

Bijlagenboek

Info Support Butler

De (mobiele) persoonlijke assistent



Auteur	M. Stuivenberg
Studentnummer	12072494
Datum	21 maart 2016
Opleiding	De Haagse Hogeschool - Informatica

Bijlagenboek

Auteur	M. Stuivenberg
Studentnummer	12072494
Onderwijsinstelling	De Haagse Hogeschool Johanna Westerdijkplein 75 2521 EN, Den Haag
Opleiding	Informatica
Begeleidende examinerator	Dhr. van Doorn
Tweede examinerator	Mw. Maas
Afstudeerbedrijf	Info Support bv Kruisboog 42 3905 TG, Veenendaal
Afstudeerbegeleider	Mw. Hijn
Opdrachtgever	Dhr. Kuiper
Technisch begeleider	Dhr. Gorter

Titel	Bijlageboek
Project/Onderwerp	Info Support Butler
Versie	1.0
Status	Definitief
Datum	21-03-2016
Bestand	Afstudeerverslag
Bedrijf	Info Support bv



Info Support B.V.
Kenniscentrum

Kruisboog 42, 3905 TG, Veenendaal,
De Smalle Zijde 39, 3903 LM Veenendaal,



Tel. +31 (0) 318 55 20 20 - Fax +31 (0) 318 55 23 55
Tel. +31 (0) 318 50 11 19 - Fax +31 (0) 318 51 83 59

Historie

Versie	Status	Datum	Auteur	Verandering
1.0	Definitief	16-03-2016	Maikel Stuivenberg	Document definitief

Distributie

Versie	Status	Datum	Aan
1.0	Definitief	21-03-2016	De Haagse Hogeschool, Info Support

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

Inhoudsopgave

Bijlage B – Plan van aanpak	1
Bijlage C – Onderzoeksplan.....	26
Bijlage D – Onderzoeksrapport	35
Bijlage E – Architectuur	94
Bijlage F – Requirements	107
Bijlage G – Functioneel ontwerp	111
Bijlage H – Technisch ontwerp.....	121

Plan van aanpak

Info Support Butler



Plan van aanpak

Titel	Plan van aanpak
Project/Onderwerp	Info Support Butler
Versie	1.1
Status	Definitief
Datum	16-3-2016
Bestand	Plan van aanpak
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	25-11-2015	Maikel Stuivenberg	Creatie
0.2	Concept	30-11-2015	Maikel Stuivenberg	- Gant chart; - Git en OTP omgeving; - Betrokken personen; - Begrippenlijst; - Stijlfouten.
1.0	Definitief	07-12-2015	Maikel Stuivenberg	- Typfouten; - Testomgeving; - Management samenvatting; - Randvoorwaarden; - Ontwikkelmethode.
1.1	Definitief	08-12-2015	Maikel Stuivenberg	- Kwaliteitseisen - Risico's

Distributie

Versie	Status	Datum	Aan
0.1	Concept	25-11-2015	John Gorter
0.2	Concept	30-11-2015	Marco Kuiper
1.0	Definitief	07-12-2015	Marco Kuiper, John Gorter
1.1	Definitief	08-12-2015	Marco Kuiper, John Gorter
1.1	Definitief	11-02-2016	De Haagse Hogeschool
1.1	Definitief	29-02-2016	De Haagse Hogeschool (TTA)

Referenties

Code	Bron

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

Management samenvatting

In dit plan van aanpak wordt de opdracht “Info Support Butler” besproken. Deze opdracht zal uitgevoerd worden tussen 16 november 2015 en 18 maart 2016 door Maikel Stuivenberg (opdrachtnemer) en heeft als opdrachtgever Marco Kuiper werkzaam bij Info Support.

De ‘Info Support Butler’ is een mobiele variant van een persoonlijke assistent. Deze applicatie zal met behulp van iBeacons gaan herkennen of iemand het pand van Info Support binnenkomt of verlaat en geeft, afhankelijk van de situatie, informatie over bijvoorbeeld zijn agenda of het verkeer. Tegelijkertijd met de ontwikkeling van de Butler wordt deze opdracht ook gebruikt om af te studeren bij de Haagse Hogeschool te Den Haag. Hierdoor zijn er in dit plan ook een aantal specifieke eisen te vinden voor de opleiding.

Omdat de butler aangesloten zal worden op verschillende systemen, die allemaal verschillend werken, moet er een universele connector gemaakt worden die de applicatie met de verschillende systemen verbindt. Vanaf de 2^e tot en met de 8^e week zal in een onderzoek gekeken worden naar de mogelijkheden om de verschillende systemen te verbinden via deze universele connector en wordt gekeken hoe de gebruiker uiteindelijk kan kiezen welke systemen hij wil zien.

Voor de ontwikkeling, dat start vanaf de 8^e week (4 januari 2016), van de Butler is het nodig om zowel een frontend (applicatie voor Android en iOS) als een backend te maken. De ontwikkeling gebeurt met behulp van de Scrum methodiek, waarbij er in sprints van 2 weken functionaliteiten aan het product toegevoegd zullen worden. Voor dit proces wordt er aan het begin van de ontwikkelfase een backlog opgesteld en zal bij elk sprintreview gekozen worden welke backlog items er ontwikkeld gaan worden in de volgende sprint.

Aan het eind van dit project (18 maart 2016) worden verschillende applicaties (front- en backend) en documenten (waaronder dit plan van aanpak, onderzoeksplan en functioneel ontwerp) opgeleverd. Dit gebeurt aan zowel de opdrachtgever (per e-mail) als aan de Haagse Hogeschool (afstudeerdossier).

Inhoudsopgave

MANAGEMENT SAMENVATTING.....	4
1 BEGRIPPENLIJST	6
2 OPDRACHT	7
2.1 Betrokken personen.....	7
2.2 Opdrachtdefinitie.....	8
2.2.1 Behalen van afstudeertraject	8
2.2.2 Ontwikkelen van een persoonlijke assistent	8
2.3 Afbakening	9
2.4 Afhankelijkheden.....	9
2.5 Kwaliteitseisen	9
2.6 Uitgangspunten.....	10
2.7 Randvoorwaarden	10
3 RISICOMANAGEMENT.....	12
4 AANPAK	15
4.1 Ontwikkelmethode.....	17
4.2 Fases	17
4.2.1 Opstart van project	17
4.2.2 Onderzoek.....	18
4.2.3 Ontwikkeling.....	18
4.2.4 Omgeving	19
4.3 Versiebeheer	20
4.4 Afstudeerverslag	20
5 BEHEERSASPECTEN	21
5.1 Organisatie	21
5.1.1 Normstelling	21
5.1.2 Voortgangscontrole	22
5.2 Informatie	22
5.2.1 Normstelling	22
5.2.2 Voortgangscontrole	22
5.3 Geld	22
5.3.1 Normstelling	22
5.3.2 Voortgangscontrole	23
5.4 Middelen	23
5.4.1 Normstelling	23
5.4.2 Voortgangscontrole	23
5.5 Kwaliteit.....	23
5.5.1 Normstelling	23
5.5.2 Voortgangscontrole	24
5.6 Overlegvormen	24
6 OPLEVERING	25

1 Begrippenlijst

Begrip	Omschrijving
API	Een API definieert de toegang tot functionaliteiten van de backend. De frontend hoeft hierdoor niet de implementatie van de verschillende functionaliteiten te weten, maar hoeft alleen via de API de gegevens op te vragen.
Backend	De backend is het deel van de applicatie dat niet zichtbaar is voor de gebruiker. De frontend kan de backend aanspreken via API's.
Bluetooth Low Energie (ook BLE of Bluetooth Smart genoemd)	BLE is een protocol waarmee zenders gemaakt kunnen worden die weinig energie hebben en die specifieke informatie willen delen. (bijv. locatie-, fitness- of lichaamsgegevens)
Frontend	De frontend is de kant die de gebruiker ziet. Dit kan een mobiele applicatie zijn (locatiebepalen en tonen van informatie), maar kan ook een TV scherm of webpagina zijn.
iBeacons	Een iBeacon is een Bluetooth (Low Energie) apparaat dat continue zijn identiteit verstuurd (een nummer). Hiermee wordt het voor andere apparaten (zoals smartphones) mogelijk om te bepalen of je binnen bereik komt van een beacon.
Product backlog	De product backlog is een lijst met requirements voor dit product / project.
Repository	Een repository is een (centrale, bijv. online) opslagplaats, waarin de volledige geschiedenis van het versiebeheersysteem bijgehouden wordt. Doordat dit centraal opgeslagen is, kan dit vanaf verschillende plekken benaderd en aangepast worden.
Requirements	Een requirement is een gedocumenteerd eis in een ontwikkelproces.
Universele Connector	Een connector is 'een systeem' waar verschillende andere systemen, met verschillende protocollen, op aangesloten kunnen worden. De connector zet dit om naar één API of protocol. Hierdoor hoeft de frontend niet allerlei protocollen te ondersteunen en hoeft bij een wijziging aan een extern systeem niet de frontend geüpdatet te worden.
Versiebeheersysteem	Een versiebeheersysteem is een (computer)programma waarmee wijzigingen, die gemaakt worden in documenten of software, bijgehouden kunnen worden.

2 Opdracht

2.1 *Betrokken personen*

Opdrachtnemer

Maikel Stuivenberg
Afstudeerder
Info Support bv, Veenendaal
Maikel.Stuivenberg@infosupport.com
06-54230672

Opdrachtgever

Marco Kuiper
Enterprise Mobile
Info Support bv, Veenendaal
Marco.Kuiper@infosupport.com
06-31665786

Technisch begeleider

John Gorter
Docent Kenniscentrum
Info Support bv, Veenendaal
John.Gorter@infosupport.com
06-29234081

Afstudeerbegeleider

Pascalie Hijn
Opleidingsadviseur
Info Support bv, Veenendaal
Pascalie.Hijn@infosupport.com
06-45378507

Begeleidend examiner

E.M. van Doorn
Docent
De Haagse Hogeschool, Den Haag
e.m.vandoorn@hhs.nl

2.2 Opdrachtdefinitie

De 'Info Support Butler' opdracht kent meerdere doelstellingen:

1. Behalen van het afstudeertraject aan de Haagse Hogeschool te Den Haag.
2. Ontwikkelen van een smartphone applicatie dat zal bijhouden wanneer een medewerker van Info Support het pand binnenkomt en verlaat. Ook geeft het de gebruiker, als een assistent, persoonlijke informatie op zijn smartphone of op een extern scherm.

Beide doelstellingen hebben hun eigen aanleiding, effect en resultaat. Hieronder staat per doelstelling vermeld wat het verwachte resultaat is en welk effect dit zal hebben.

2.2.1 Behalen van afstudeertraject

De aanleiding voor deze doelstelling is de huidige opleiding (Informatica) van de opdrachtnemer, die gevolgd wordt aan de Haagse Hogeschool te Den Haag. Voor deze opleiding moet aan het eind van het opleidingstraject een afstudeeropdracht gemaakt worden.

De hogeschool verwacht dat er aan einde van de afstudeerstage een afstudeerdossier wordt opgeleverd met daarin alle gemaakte documenten (plan van aanpak, onderzoek en ontwerpdocumentatie) en applicaties (code). Naast deze documenten moet ook een afstudeerverslag gemaakt worden waarin de verschillende aspecten van de opdracht worden behandeld.

2.2.2 Ontwikkelen van een persoonlijke assistent

Op dit moment zijn er verschillende informatiebronnen die een medewerker van Info Support kan raadplegen om te zien wat zijn planning is voor een dag. Een aantal voorbeelden van deze bronnen zijn:

- Agenda (Welke activiteiten heeft hij/zij vandaag?);
- Build status (Waaraan werkt een persoon? Wat moet er gedaan worden?)

Maar ook andere bronnen worden geraadpleegd op het werk, zoals:

- Nieuws;
- Files (Hoelang is de reistijd terug naar huis);
- Aanwezigheid van een collega.

Al deze bronnen raadplegen is tijdrovend. Dit kan sneller als alle informatie bij binnenkomst (of verlaten) van het gebouw in één overzicht getoond wordt.

Het resultaat van dit project is een mobiele variant van een persoonlijke assistent. Deze applicatie zal met behulp van iBeacons moeten herkennen of je het pand van Info Support binnen komt of verlaat en geeft, gebaseerd op deze informatie, een scherm met informatie over de werkdag van de medewerker.

Omdat de butler aangesloten zal worden op verschillende systemen, die allemaal verschillend werken, moet er een universele connector gemaakt worden die de applicatie met de verschillende systemen verbindt. In een onderzoek, voorafgaand aan het ontwikkelen van de applicatie, zal gekeken gaan worden naar de mogelijkheden om de verschillende systemen te verbinden en wordt gekeken hoe de gebruiker uiteindelijk kan kiezen welke systemen hij wilt zien.

2.3 Afbakening

1. De applicatie zal alleen ontwikkeld worden voor iOS en Android, andere besturingssystemen zijn uitgesloten van dit project.
2. Het plaatsen of onderhouden van iBeacons (om de locatie van een gebruiker te bepalen) valt niet onder dit project.
3. De te maken backend zal alleen aangesloten kunnen worden op een zelf te maken database en bestaande (externe) systemen. Dit project voorziet niet in wijzigingen die gemaakt moeten worden aan een van deze (externe) systemen.

2.4 Afhankelijkheden

1. Externe systemen dienen te beschikken over een API waarvan data opgehaald kan worden.

2.5 Kwaliteitseisen

De onderstaande kwaliteitseisen zijn gekozen uit de ISO 9126 norm en zijn verplicht gesteld vanuit de afstudeereisen van de Haagse Hogeschool.

1. Herbruikbaarheid (Reusability)

De classes / code van zowel de mobiele applicatie als de backend moet op zo'n manier ontworpen en geschreven worden dat het mogelijk wordt om classes of methodes te herbruiken.

2. Testbaarheid (Testability)

De code moet zo opgebouwd worden dat het mogelijk is om vooraf opgestelde testen uit te voeren. Het streven bij het maken van deze testen is om een dekking te krijgen van 80% over alle code. Mocht er teveel tijd nodig zijn om dit streven te halen, dan kan er in overleg met de opdrachtgever besloten worden om dit percentage te verlagen.

3. Wijzigbaarheid (Modifiability)

De code moet wijzigbaar opgebouwd worden, waardoor het achteraf eenvoudig blijft om extra functionaliteit toe te voegen.

Met behulp van de testen (zie ook testbaarheid) wordt het makkelijker om te controleren of nieuwe wijzigingen zorgen voor foutieve code elders en dat het systeem zonder fouten blijft werken.

Naast de eisen aan de code, zijn er ook eisen aan de documentatie en de opdrachtnemer:

4. Zelfstandig werken

De opdrachtnemer heeft weinig sturen nodig en toont zelf initiatief door bij vragen contact op te nemen bij de technisch begeleider, afstudeerbegeleider of opdrachtgever.

5. Communicatie

Alle communicatie (zowel schriftelijk als mondeling) moet in aanvaardbaar Nederlands zijn. Dit geldt voor de communicatie met de klant, bij bijvoorbeeld overleggen en e-mails, maar daarnaast ook voor alle gemaakte documenten en rapporten.

6. Resultaatgericht werken

Tijdens dit project moet (aantoonbaar) resultaatgericht gewerkt worden. Dit kan gedaan worden door te werken volgens de opgestelde planning en doorzettingsvermogen te tonen.

2.6 Uitgangspunten

Voor de uitvoering van de in dit plan van aanpak beschreven applicaties zijn de onderstaande uitgangspunten van toepassing:

1. Voor de mobiele applicaties wordt gebruik gemaakt van het ontwikkelpakket Xamarin.
2. Ontwikkelen van de mobiele applicaties gebeurt in C# (.NET Framework).
3. Applicatie wordt gebouwd voor iOS en Android.

2.7 Randvoorwaarden

Aan uitvoering van de in dit plan van aanpak beschreven delen zijn de volgende randvoorwaarden gesteld:

1. Aan het project wordt (minimaal) 4 dagen in de week bij Info Support te Veenendaal gewerkt. Eén dag in de week kan gebruikt worden om thuis aan het project te werken.
2. Om te kunnen detecteren wanneer iemand het gebouw binnenkomt of verlaat, of om te zien waar iemand zich bevindt, dienen er op verschillende locaties iBeacons aanwezig te zijn.
3. Externe systemen dienen te beschikken over een API waarvan data opgehaald kan worden.
4. Om de applicatie te kunnen ontwikkelen voor iOS, is het nodig dat er een Mac beschikbaar is waarmee Visual Studio / Xamarin gekoppeld kan worden.
5. Voor de backend dient er een server (of cloud omgeving) beschikbaar te zijn waarop een SQL database aangemaakt kan worden waarop de ontwikkelde connector gaat draaien.

6. Om de informatie eventueel op een extern scherm te gaan tonen is het noodzakelijk dat er een scherm (bijvoorbeeld een tv) aanwezig is dat de mogelijkheid heeft om een website te tonen.
7. Er is een repository nodig om de code van zowel de frontend als backend als te beheren.
8. Voor de ontwikkeling van de iOS en Android zijn Xamarin en Visual Studio licenties nodig.

3 Risicomanagement

Voor uitvoering van de, in dit plan van aanpak beschreven delen, zijn de onderstaande risico's onderkend. Bij de risico's zijn de oorzaken en bijbehorende maatregelen ter beheersing opgenomen.

Nr.	Risico omschrijving				
	Oorzaak	Maatregel	P/S	Wie	Wanneer
1	Locatie bepaling van gebruikers kan niet gedetecteerd worden door ontbreken van iBeacons.				
	Geen iBeacons aanwezig.	Op verschillende locaties moeten iBeacons opgehangen worden.	P		
		Informatie weergeven aan de hand van het tijdstip.	S		
2	Apple / Android apparaat ondersteunt geen BLE (Bluetooth Low Energie), hierdoor kan er niet gezocht worden naar iBeacons en is het niet mogelijk om te bepalen om iemand aanwezig is.				
	Mobiël heeft geen ondersteuning voor Bluetooth 4.0 (BLE).	De applicatie niet beschikbaar maken voor deze apparaten.	P		
		Dit wordt in overleg met de opdrachtgever beperkt.	S		
3	Het is niet mogelijk om te ontwikkelen voor iOS doordat er geen mogelijkheid is om gebruik te maken van een Mac met bijbehorende eisen (zie hoofdstuk 1.4 en 4.4.1).				
	Een Mac ontbreekt.	Het ontwikkelen van de mobiele applicatie kan alleen gedaan worden voor Android.	P		
		De code voor iOS kan vast geschreven worden, waarbij in een later stadium alsnog de UI gemaakt kan worden.	S		

Nr.	Risico omschrijving				
	Oorzaak	Maatregel	P/S	Wie	Wanneer
4	Ik (opdrachtnemer) kan onverwachts ziek worden of een ongeluk krijgen				
	Ziekte of ongeluk	In de planning rekening houden met een paar dagen uitloop.	P		
		Bij ziekte dit (direct) melden bij de opdrachtgever en afstudeerbegeleider en afspraken maken om te voorkomen dat het project of afstudeer-traject mislukt.	S		
5	De opdrachtnemer kan het project niet volledig ontwikkelen doordat er niet genoeg kennis is opgedaan over de programmeertalen of architectuur				
	Te weinig kennis	Training volgen bij Info Support Kenniscentrum.	P		
		Bij vragen of onbekende onderwerpen kan de technisch begeleider benaderd worden.	S		
6	De beschreven planning blijkt niet haalbaar				
	Te krappe planning	In de planning wordt rekening gehouden met een week uitloop.	P		
		Bij uitloop, dat niet ingehaald kan worden, afspraken maken/verzetten in overleg met de opdrachtgever.	S		

Nr.	Risico omschrijving				
	Oorzaak	Maatregel	P/S	Wie	Wanneer
7	Uit het onderzoek komt naar voren dat er geen geschikte backend gemaakt kan worden. Hierdoor kan het zijn dat de afstudeerder een onvoldoende krijgt voor de opdracht.				
	Geen geschikte backend	Vooronderzoek doen, om te kijken of de opdracht haalbaar is.	P		
		Overleggen met de opdrachtgever, technisch begeleider en eventueel school over de te nemen stappen.	S		
8	De code kan onbereikbaar worden door een crash van de gebruikte software, computer. Hierdoor moet de code opnieuw gemaakt worden.				
	Storing of crash	Door de gebruik te maken van versiebeheer wordt voorkomen dat de code verloren raakt.	P		
		Er kan één keer in de week een back-up gemaakt worden, zodat nooit alle code opnieuw geschreven moet worden.	S		
9	De opdrachtgever of technisch begeleider wordt ziek of gaat met vakantie				
	Ziekte of vakantie	Er kan van te voren afspraken gemaakt worden wie zijn taak tijdelijk overneemt bij ziekte of vakantie.	P		
		Bij onverwachte situaties, zoals (langdurige) ziekte, moet met de afstudeerbegeleider contact opgenomen worden voor een oplossing.	S		

P/S = Preventief of Schadebeperkend

4 Aanpak

Het project kan opgedeeld worden in 4 fases: opstart van project, onderzoek, ontwikkeling van applicaties en het schrijven van een afstudeerverslag. Elke fase heeft zijn eigen doelstelling, product(en) en planning.

Als globale planning voor alle onderdelen kan de volgende schema aangehouden worden. Hierbij wordt uitgegaan dat week 1 start op maandag 16 november 2015 en de laatste week (18) eindigt op vrijdag 18 maart 2016.

			November			December			Januari				Februari				Maart			
			16 - 22	23 - 29	30 - 06	07 - 13	14 - 20	21 - 27	28 - 03	04 - 10	11 - 17	18 - 24	25 - 31	01 - 07	08 - 14	15 - 21	22 - 28	29 - 06	07 - 13	14 - 20
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Taak	Start	Af																		
Opstart																				
1 Plan van aanpak	1	3																		
Onderzoek																				
2 Onderzoeksplan	2	4																		
3 Onderzoeksrapport	4	8																		
Ontwikkeling																				
4 Functioneel ontwerp	8	18																		
5 Technisch ontwerp	8	18																		
6 Requirements document	8	18																		
7 Architectuur document	8	18																		
8 Android applicatie	8	18																		
9 iOS applicatie	8	18																		
10 Backend applicatie	8	18																		
11 Backend database	8	18																		
Afstuderen																				
12 Afstudeerverslag	5	18																		
Trainingen																				
14 Documentatie																				
15 Applicatie																				

Voor alle documentatie uit bovenstaand overzicht (1 t/m 12) zijn er eind opleveringen gepland aan verschillende personen. Hieronder staat in een overzicht welk document aan wie opgeleverd wordt.

Taak	Oplevering aan	Locatie	Week (t.o.v. afstudeerperiode)	Formaat
1 Plan van aanpak	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	3	.pdf
			18	.pdf
2 Onderzoeksplan	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	4	.pdf
			18	.pdf
3 Onderzoeksrapport	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	8	.pdf
			18	.pdf
4 Functioneel ontwerp	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.pdf
			8	.pdf
5 Technisch ontwerp	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.pdf
			8	.pdf
6 Requirements document	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.pdf
			18	.pdf
7 Architectuur document	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.pdf
			18	.pdf
8 Android applicatie	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.zip
			18	.zip
9 iOS applicatie	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.zip
			18	.zip
10 Backend applicatie	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.zip
			18	.zip
11 Backend database	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	17	.zip
			18	.zip
12 Afstudeerverslag	Opdrachtgever & Technisch begeleider De Haagse Hogeschool	E-Mail Afstudeerdossier	18	.pdf
			18	.pdf

In hoofdstuk 4.2 (Fases) staat per fase beschreven wanneer deze plaats zal vinden en wat daarin gedaan gaat worden. Naast de verschillende fases zijn er momenten waarop contact plaats vindt met de opdrachtgever en school, maar zijn er ook een aantal presentaties en trainingen gepland voor de opdrachtnemer. Specifieke informatie omtrent de contactmomenten is terug te vinden in hoofdstuk 4.6 (Overlegvormen).

4.1 Ontwikkelmethode

Het ontwikkelen van de opdracht zal gedaan worden met behulp van de Scrum ontwikkelmethode. Hierbij wordt in periodes van 2 weken (sprints) steeds een product opgeleverd. Omdat dit project door één persoon wordt uitgewerkt, staat in dit hoofdstuk uitgelegd hoe scrum precies wordt toegepast.

Product-owner

De product-owner is de eigenaar van het product. Bij dit project zal het gaan om de opdrachtgever: Marco Kuiper.

Ontwikkelteam

Hoewel scrum meestal wordt toegepast bij projecten waar meerdere ontwikkelaars werken aan het zelfde product, bestaat het ontwikkelteam bij dit project uit één persoon: Maikel Stuivenberg (opdrachtnemer).

Stand-up

Bij scrum projecten worden dagelijks stand-up meetings georganiseerd. Hierbij verteld ieder lid van het ontwikkelteam wat hij heeft gedaan en wat hij gaat doen. Hoewel het ontwikkelteam bestaat uit één persoon wordt er wel een stand-up meeting gehouden met de ontwikkelaars die in dezelfde kamer zitten als de opdrachtnemer.

Sprint planning

Het plannen van een sprint wordt gedaan door de opdrachtnemer in overleg met de opdrachtgever. Hierbij wordt gekozen uit backlog items welke voorafgaand aan de ontwikkelfase opgesteld zullen worden.

Retrospective

Tijdens een retrospective wordt met het ontwikkelteam gekeken wat er goed en fout ging. Doordat er geen team is, maar één persoon die gaat ontwikkelen, is een retrospective niet van toepassing.

Wel kan er met de betrokken personen (technisch begeleider, afstudeerbegeleider of opdrachtgever) tussentijds een evaluatie gedaan worden om te kijken of de ontwikkeling van de applicaties goed gaan en er geen problemen optreden.

Scrumboard en Backlog

De product-, sprint backlog en het scrumboard zullen bijgehouden worden via TFS (Team Foundation Server). Dit is een online tool waarin het hele project bijgehouden kan worden, zo is het mogelijk om een scrumboard te bekijken, maar kan een (git-) commit ook gekoppeld worden aan een taak.

4.2 Fases

4.2.1 Opstart van project

Tijdens de opstart van dit project wordt dit document (plan van aanpak) opgesteld en worden de eerste details met de opdrachtgever besproken.

Aan het eind van week 1 zal de eerste versie van het plan van aanpak ter review aangeleverd worden aan de technisch begeleider.

In week 3 (voorafgaand aan de bespreking met de opdrachtgever) zal de opdrachtgever de definitieve versie van het plan van aanpak ontvangen.

4.2.2 Onderzoek

In week 3 zal, na goedkeuring (door zowel de technische begeleider als de klant) van het plan van aanpak, begonnen worden met het onderzoek. Het onderzoek wordt opgedeeld in het schrijven van een plan (onderzoeksplan) en het onderzoek zelf (waarbij de resultaten gedocumenteerd worden).

De planning voor deze 2 onderdelen wordt op de volgende manier verdeeld:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Vooronderzoek																		
Plan																		
Resultaat																		

Vooronderzoek

Zoals in het bovenstaande planningsschema te zien is, is er in week 2 ruimte voor eventueel vooronderzoek. Dit vooronderzoek zal gedaan worden indien het werk aan het plan van aanpak tijdelijk stil ligt (wegens een review) of op moment dat het plan van aanpak voortijdig klaar is.

Onderzoeksplan

Na het gesprek met de opdrachtgever in week 3, zal in 2 weken tijd (ongeveer 6 werkdagen) het onderzoeksplan opgesteld worden. Dit document zal worden gereviewd door de technisch begeleider.

Onderzoeksresultaat

Vanaf week 5 zal begonnen worden met het onderzoek, waarbij getracht wordt de vragen uit het onderzoeksplan te beantwoorden. De resultaten op de vragen uit het onderzoeksplan zullen opgeschreven worden in een document.

4.2.3 Ontwikkeling

Tegelijk met het onderzoek wordt in week 3 ook gestart met het ontwikkelen van de applicatie. Hierbij worden week 3 tot en met 6 (met een maximum van 1 dag per week) gebruikt om, naast het onderzoek (hoofdstuk 4.2), de mogelijkheden met iBeacons, Xamarin en het maken van een backend uit te proberen.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Onderzoek																		
Definitieve ontwikkeling																		

Nadat het onderzoek (hoofdstuk 3.2) is afgerond zal de ontwikkeling van de uiteindelijk applicatie en backend gestart worden. Voor het ontwerp en de ontwikkeling hiervan wordt ongeveer 4 dagen per week uitgetrokken.

Aan het begin van de (definitieve) ontwikkelfase wordt de product backlog gevuld met alle eisen en wensen van de klant. Deze backlog kan later, tijdens een sprint review, gewijzigd of aangevuld worden met items die opgesplitst moeten worden of op moment dat er fouten uit vorige sprints zijn.

Voorafgaand aan elke sprint wordt in samenspraak met de opdrachtgever bepaald welke wensen/eisen uit de product backlog gepland worden voor de komende sprint.

Aan het eind van de ontwikkelfase worden de volgende documenten opgeleverd:

- Requirements- / Architectuur document;
- Functioneel- / Technisch ontwerp.

En zullen de volgende onderdelen opgeleverd worden:

- Android applicatie;
- iOS applicatie;
- Backend (inclusief database).

4.2.4 Omgeving

Voor de ontwikkeling van de front- en backend worden verschillende omgevingen gebruikt voor het ontwikkelen, testen en de productie. (Ook wel OTP genoemd)

Ontwikkelomgeving

Het ontwikkelen van de applicaties gebeurt op de desktop van de opdrachtnemer. Hier worden ontwikkelversies van de backend lokaal gedraaid en worden de frontend (mobiele) applicaties getest met behulp van emulators (Android op Windows, iOS op OSX) en fysieke mobiele apparaten (om de iBeacons te kunnen testen).

Testomgeving

De testomgeving bestaat uit een (virtuele) server, welke de backend code uit de testbranch (zie 3.3.2) kan ophalen en hierop automatisch unit testen kan uitvoeren. De testomgeving zal een rapport teruggeven met daarin de gevonden fouten. Op basis van dit rapport kan in de ontwikkelomgeving de code aangepast worden, zodat alle bugs opgelost worden en het gewenste resultaat gegeven gaat worden.

De frontend zal getest worden met behulp van unit testen. Deze testen worden uitgevoerd op de ontwikkelomgeving in Xamarin of Visual Studio.

Productieomgeving

Voor zowel de backend als frontend zijn er productieomgevingen nodig. De backend bestaat hierbij uit server waarop een database en de ontwikkelde webservice draait. Daarnaast moet deze productieomgeving via het externe netwerk (internet) bereikbaar zijn.

De frontend productomgeving bestaat uit een Android en iOS toestel dat voorzien kan worden van de laatste (productie) software.

4.3 **Versiebeheer**

Om de code te beheren en de juiste versies te kunnen maken voor de verschillende omgevingen (hoofdstuk 3.3.1) is het nodig een systeem te gebruiken waarin deze gescheiden delen mogelijk zijn.

Hierdoor is voor dit project gekozen om GIT te gebruiken, opgezet met minimaal 3 branches:

- Master;
- Testing;
- Development;
- En eventueel extra branches waarin functionaliteiten gemaakt of uitgeprobeerd kunnen worden.

De 'Development' branch (en eventuele extra branches) bevat hierbij de laatste code waar de opdrachtnemer mee bezig is. Na elke (gedeeltelijke) functionaliteit wordt de code gepushed, zodat altijd alle wijzigingen aanwezig zijn en bij problemen niks verloren kan gaan.

De 'Testing' branch bevat de code van complete functionaliteiten (incl. geschreven unit testen). Deze test branch kan gereleased worden naar de (backend)testomgeving, waarop de test uitgevoerd worden, maar kan daarnaast ook gebruikt worden voor de lokale frontend testen.

In de 'Master' branch wordt één keer in de twee weken, aan het eind van een sprint, een release geplaatst nadat de testen zijn doorlopen en de klant de functionaliteiten heeft goedgekeurd.

4.4 **Afstudeerverslag**

Vanaf week 5 (half december) wordt, voor het afstudeertraject aan de Haagse Hogeschool, gewerkt aan een verslag waarin verschillende onderwerpen aan pas zullen komen. De opleiding verwacht hierbij dat het afstudeerverslag minimaal het volgende bevat: (bron: afstdossier.pdf, versie 20150826, de Haagse Hogeschool)

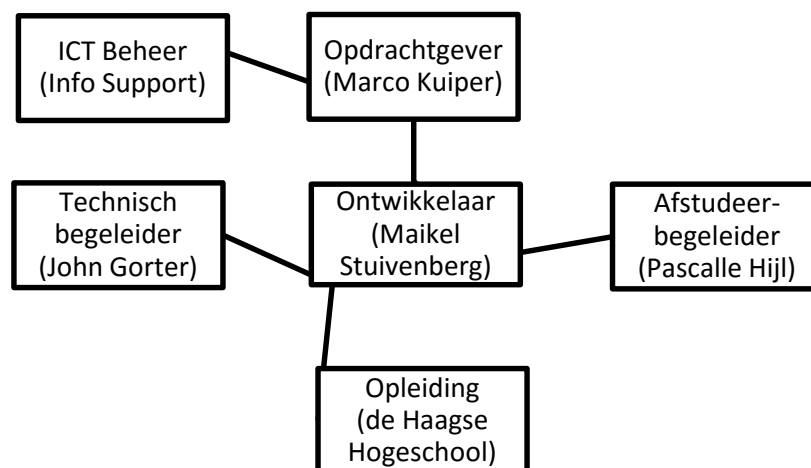
- Referaat, voorwoord, inhoudsopgave en inleiding op het afstudeerverslag;
- Beschrijving van de organisatie van de opdrachtgever en de plaats van de afstudeerder daarin;
- Beschrijving van de situatie bij aanvang van het afstuderen;
- Bespreking van mogelijke oplossingsmethoden en verdediging van de gekozen aanpak;
- Beschrijving van de werkzaamheden met toelichting op en motivatie van de gemaakte keuzes;
- Bespreking van afwijkingen ten opzichte van het afstudeerplan met motivatie
- Analyse van de (tussen)resultaten;
- Bespreking van de opgeleverde (tussen)producten los van (tussen)producten zelf;
- Evaluatie van de gebruikte aanpak tijdens de afstudeerperiode;
- Evaluatie van de opgeleverde (tussen)producten;
- Geraadpleegde literatuur;
- Afkortingenlijst, noodzakelijke, niet algemeen toegankelijke achtergrondinformatie, e.d.

Voor het verslag zal maximaal 1 dag per week uitgetrokken worden om te zorgen dat alle benodigde informatie hierin terecht gaat komen.

5 Beheersaspecten

5.1 Organisatie

5.1.1 Normstelling



Opdrachtgever / Projectleider

Marco Kuiper is binnen dit project zowel opdrachtgever als projectleider. Elke twee weken zal er een gesprek plaats vinden waarin de voortgang besproken wordt en maakt hij samen met de projectnemer afspraken over de sprints.

Technisch begeleider

De technisch begeleider zal beschikbaar zijn voor (technische) vragen over de documentatie en ontwikkeling van de applicaties.

ICT beheer

De gebruikte hardware/software (zowel ontwikkelomgeving als serveromgeving) wordt onderhouden door de ICT beheer van Info Support. Zij zullen zorg dragen voor de juiste licenties en dat de benodigde hardware ook beschikbaar is.

Opleiding

De opleiding verwacht aan het eind van dit project een afstudeerverslag, waarbij verschillende details uit dit project besproken worden. Een examinerator van de hogeschool komt op 25% van de afstudeerperiode langs bij Info Support ter controle van de activiteiten van de afstudeerder (zie ook hoofdstuk 4.6 Overlegvormen).

Afstudeerbegeleider

Info Support biedt aan de afstudeerder een begeleider aan. Met de begeleider kan de voortgang van het project besproken worden en kan, indien problemen ontstaan, meedenken naar oplossingen.

5.1.2 Voortgangscontrolle

Bij problemen binnen de organisatie wordt allereerst getracht het probleem met de persoon zelf op te lossen. Indien dit niet mogelijk is, kunnen problemen ook worden besproken met de afstudeerbegeleider (Pascalle Hijn).

De opdrachtgever ontvangt elke 2 weken een voortgangsverslag met daarin de werkzaamheden die uitgevoerd zijn. Tevens vindt iedere (ongeveer) 6 weken een gesprek plaats met Henk Brands om mijn functioneren binnen de organisatie te bespreken.

5.2 Informatie

5.2.1 Normstelling

Binnen dit project worden verschillende documenten opgesteld met informatie over het onderzoek en de applicatie.

De verschillende documentatie ontvangen de status definitief op moment dat de documenten goedgekeurd zijn door de technisch begeleider.

5.2.2 Voortgangscontrolle

Zoals ook in de normstelling beschreven, wordt de controle op documenten gedaan door de technisch begeleider (John Gorter). Hij zal de documenten op initiatief van de opdrachtnemer bespreken in tijdens een afspraak tussen de technisch begeleider en de opdrachtnemer.

Zoals besproken in de aanpak van het project (hoofdstuk 3) worden na afronding van een fase (opstart, onderzoek, ontwikkeling) de gemaakte documenten opgeleverd aan zowel de klant als de technisch begeleider.

De opdrachtnemer draagt zorg voor het op tijd opleveren van documentatie aan zowel de technisch begeleider als de klant. Indien documentatie niet tijdig opgeleverd kan worden aan de opdrachtgever zal dit gemeld worden en een nieuwe afspraak gemaakt worden om dit op te leveren.

5.3 Geld

5.3.1 Normstelling

Licenties

Voor de ontwikkeling van de applicaties zijn de volgende licenties benodigd:

1. Visual Studio 2010 Professional (of hoger);
2. Xamarin.Android;
3. Xamarin.iOS (Indien Mac beschikbaar);
4. Eventuele backend software.

5.3.2 Voortgangscontrolle

Zodra de opdrachtnemer dit aangeeft zal de opdrachtgever zorgen dat de ICT beheer voor de juiste applicaties en/of licenties gaat zorgen.

5.4 Middelen

5.4.1 Normstelling

Voor de ontwikkelfase is het nodig dat er een ontwikkelmachine en backendomgeving beschikbaar is. Hieronder staan de verschillende omgevingen beschreven met de bijbehorende eigenschappen.

Ontwikkelomgeving

De ontwikkelomgeving bestaat uit een Windows omgeving met daarop geïnstalleerd: Xamarin, Visual Studio en de 'Xamarin for Visual Studio' plug-in.

Voor de ontwikkeling van de iOS applicatie is het noodzakelijk dat er een Mac beschikbaar is met de volgende eisen / software:

- OS X (10.10 of hoger);
- Xamarin iOS SDK;
- Apple's Xcode IDE (7 of hoger);
- iOS SDK;
- Beschikbaar binnen hetzelfde netwerk als de Windows omgeving.

Deze machine moet via de Xamarin build-host optie verbonden worden met Visual Studio op de Windows omgeving.

Naast deze machine is het ook nodig (vanuit de Haagse Hogeschool) om een versiebeheersysteem te gebruiken in dit project. Hiervoor is het nodig dat er een git repository opgezet wordt.

5.4.2 Voortgangscontrolle

De opdrachtnemer zal erop toe zien dat alle genoemde middelen geleverd worden door de opdrachtgever / ICT Beheer. Indien middelen noodzakelijk zijn zal de opdrachtgever hierop geattendeerd worden en wordt verwacht dat de benodigde apparatuur / licenties zal regelen.

5.5 Kwaliteit

5.5.1 Normstelling

Het product (zowel de Android/iOS applicatie als backend) dat gebouwd zal worden is een proof of concept. Hierbij gaat het vooral om de universele connector die onderzocht en ontwikkeld moet worden. De UI van de mobile applicaties hebben geen prioriteit. Mocht er te weinig tijd zijn om de UI 'mooi' te maken, dan wordt in overleg met de opdrachtgever gekeken hoe we de informatie simpel op het scherm laten zien.

5.5.2 Voortgangscontrolle

De opdrachtgever zal tijdens de sprint reviews controleren op voortgang van de applicatie, waarbij hij tevens zal controleren of de applicatie voldoet aan de eisen van een proof of concept.

5.6 Overlegvormen

Iedere 2 weken (zie onderstaande data) zal er een gesprek plaats vinden met de opdrachtgever. Hierbij wordt de voortgang van het project besproken (sprint review) en wordt besproken welke backlog items in de volgende sprints gebouwd gaan worden.

Datum	Tijd
02-12-2015	15:00 – 16:00
15-12-2015	16:00 – 17:00
06-01-2016	15:00 – 16:00
20-01-2016	15:00 – 16:00
03-02-2016	15:00 – 16:00
17-02-2016	15:00 – 16:00
02-03-2016	15:00 – 16:00
16-03-2016	15:00 – 16:00

Naast het overleg met de opdrachtgever zal er ook (op initiatief van de opdrachtnemer) reviews plaats vinden over de documenten / ontwikkelde software. Deze datums staan niet vast maar kunnen op willekeurige momenten gepland worden door een e-mail te sturen naar de technisch begeleider.

Ook met de examiner(en) van de Haagse Hogeschool zal overleg zijn over de voortgang van de afstudeerder. Hierbij zijn de volgende vaste momenten opgesteld door de Haagse Hogeschool:

% afstudeerperiode	Soort moment
25%	Bedrijfsbezoek door begeleidende examiner aan gastbedrijf. Hierbij is opdrachtgever of bedrijfsmentor aanwezig.
45%	Voortgangsverslag van student aan begeleidende examiner.
60%	Bespreking van concept afstudeerdossier (alle, tot dan toe, gemaakte documentatie en afstudeerverslag)
80%	Uiterlijk 3 weken voor de inleverdatum vindt er op basis van het beschikbare afstudeerdossier een tussentijds assessment plaats met begeleidend en tweede examiner. Bedrijfsmentor (Pascalle) ontvangt de bevindingen hiervan.
100%	Inlevering van compleet afstudeerdossier. Dit moet in drievoud ingeleverd worden en digitaal een vierde exemplaar.

6 Oplevering

- **Opstart**

- Plan van aanpak.

De oplevering van het plan van aanpak zal aan de opdrachtgever opgeleverd worden zodra dit document de status "definitief" krijgt. De oplevering gebeurt in pdf-formaat en wordt per e-mail verzonden. Indien de opdrachtgever vragen heeft kan dit tijdens het projectoverleg besproken worden.

- **Onderzoek**

- Onderzoeksplan;
- Onderzoeksresultaat.

De documentatie van het onderzoek (zowel plan als resultaat) worden opgeleverd aan de Haagse Hogeschool en de opdrachtgever. Zij zullen de documenten in pdf-formaat ontvangen via de door hen opgegeven methode (resp. Afstudeerdossier en e-mail)

- **Ontwerpdocumentatie**

- Requirements document;
- Architectuur document;
- Functioneel ontwerp;
- Technisch ontwerp.

Zowel de opdrachtgever als de Haagse Hogeschool ontvangen alle documentatie aan het einde van het project in pdf-formaat. Net als het onderzoek zal die via het afstudeerdossier of e-mail opgeleverd gaan worden.

- **Frontend applicatie**

- iOS applicatie;
- Android applicatie.

De applicaties zullen aan de opdrachtgever opgeleverd worden als een .apk file (Android) en een .ipa file (iOS). Daarnaast is de source code van het Xamarin project ook beschikbaar via een versiebeheersysteem (Git).

De Haagse Hogeschool zal de broncode, net als de opdrachtgever, ontvangen in de vorm van een .zip file. Hierin bevindt zich het complete Xamarin project (waarin zowel de iOS als Android applicatie te vinden is).

- **Backend**

- Applicatie;
- Database.

De applicatie en database voor de backend zal aan de opdrachtgever en de Haagse Hogeschool opgeleverd worden zodra deze bestanden af zijn.

- **Afstudeerverslag**

- Op 60% van de afstudeerperiode (ongeveer 10 weken na start) moet er een conceptversie ingeleverd/besproken worden bij de toegewezen examinatoren van de Haagse Hogeschool.
- Aan het einde van de afstudeerperiode wordt een definitieve versie van het afstudeerverslag ingeleverd.

Onderzoeksplan

Info Support Butler



Onderzoeksplan

Titel	Onderzoeksplan
Project/Onderwerp	Info Support Butler
Versie	1.0
Status	Definitief
Datum	21-03-201616-3-2016
Bestand	Onderzoeksplan
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	02-12-2015	Maikel Stuivenberg	Creatie
0.2	Concept	07-12-2015	Maikel Stuivenberg	- Info Support stijl toegepast; - Onderzoeksontwerp verbeterd.
0.3	Concept	10-12-2015	Maikel Stuivenberg	- Hoofd-/deelvragen; - Begrippen; - Planning; - Bijlage toegevoegd.
0.4	Concept	18-12-2015	Maikel Stuivenberg	- Hoofd-/deelvragen; - Planning.
1.0	Definitief	20-01-2015	Maikel Stuivenberg	Plan definitief gemaakt

Distributie

Versie	Status	Datum	Aan
0.1	Concept	02-12-2015	Marco Kuiper
0.2	Concept	07-12-2015	John Gorter
0.3	Concept	11-12-2015	John Gorter
0.3	Concept	16-12-2015	Marco Kuiper
1.0	Definitief	11-02-2016	De Haagse Hogeschool
1.0	Definitief	29-02-2016	De Haagse Hogeschool (TTA)
1.0	Definitief	21-03-2016	De Haagse Hogeschool

Referenties

Code	Bron

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

1 **Aanleiding tot het onderzoek**

Dit onderzoek wordt uitgevoerd als onderdeel van de afstudeeropdracht "Info Support Butler". In deze opdracht moet een mobiele applicatie gebouwd gaan worden dat herkent wanneer een gebruiker het pand binnenkomt of het pand verlaat. Afhankelijk van de herkende situatie krijgt de gebruiker informatie op zijn mobiel te zien over zijn build status, agenda, file informatie of het nieuws.

Omdat de butler verschillende soorten informatie kan laten zien, moet de applicatie aangesloten worden op verschillende systemen, die allemaal op een verschillende manier werken. Hierdoor zal er een universele connector moeten worden gemaakt waarop deze verschillende systemen aangesloten kunnen worden. Daarnaast moet de gebruiker ook zelf kunnen kiezen welke systemen voor hem gekoppeld worden.

2 Probleemstelling

2.1 Hoofdvraag

Hoe kan er een universele connector gemaakt worden?

Toelichting:

Voor het Info Support Butler project moet er een universele connector gebouwd worden dat informatie kan ophalen uit verschillende systemen. Een systeem is hierbij een informatiebron dat door een medewerker van Info Support gebruikt wordt. Naast het ophalen van informatie moet de universele connector ook de mogelijkheid geven, aan de gebruiker, om systemen te kunnen koppelen.

2.2 Deelvragen

Om de bovenstaande hoofdvraag te kunnen beantwoorden, zijn er 3 deelvragen opgesteld die samen het antwoord van de hoofdvraag beantwoorden. Zoals in hoofdstuk 4 (Bijlage A - Deelvragen) te zien is, bestaan alle deelvragen uit sub-deelvragen. Deze sub-deelvragen zijn in overleg met de opdrachtgever uitgezocht en zorgen ervoor dat het duidelijk is welke informatie in het antwoord verwacht wordt.

1. Welke informatie wordt door een ontwikkelaar van Info Support bekeken?
2. Hoe kan data bij de verschillende systemen opgehaald worden?
3. Met welk framework kan een universele connector gebouwd?

3 Onderzoeksontwerp

3.1 Begrippen

Enkele woorden/begrippen die in dit onderzoek naar voren komen kunnen verschillende definitie bevatten. In overleg met de opdrachtgever is er een definitie vastgesteld voor deze begrippen.

Systeem

Een systeem is een informatiebron dat kan bestaan uit een softwarepakket of een website waar een medewerker van Info Support informatie uit kan halen.

Universele connector

Een connector is een systeem dat informatie uit verschillende (externe) systemen kan verzamelen en de gegevens kan converteren zodat er aan de voorkant (voor de gebruiker) er maar één ingang is waar informatie opgehaald kan worden.

3.2 Doelgroep

De doelgroep voor dit onderzoek is de opdrachtgever (Marco Kuiper) en de opdrachtnemer (Maikel Stuivenberg), zodat de uitkomst van dit onderzoek gebruikt kan worden in het "Info Support Butler" project. Daarnaast kan dit onderzoek ook gebruikt worden door andere projecten die meerdere systemen willen koppelen en resultaat moeten teruggeven aan een gebruiker.

3.3 Vorm

Alle genoemde deelvragen worden beantwoord door verschillende wijze te hanteren.

Deelvraag 1 zal bestaan uit een aantal interviews met ontwikkelaars van Info Support. Zij zullen de vragen beantwoorden, zodat er een lijst opgesteld kan worden met de gebruikte systemen en de informatie die hieruit gehaald wordt.

Deelvraag 2 bestaat uit literatuuronderzoek. Waarbij gezocht kan worden naar (betrouwbare) informatiebronnen om tot het antwoord van de verschillende sub-deelvragen te komen.

Bij de laatste deelvraag, deelvraag 3, worden eerst alle functionaliteiten opgesomd waaruit de universele connector zou moeten bestaan en wordt daarna onderzoek gedaan naar bestaande oplossingen.

Het onderzoek zal gestart worden in week 4 van het project (start 7 december 2015). Tijdens het onderzoek gedurende 5 weken (tot 10 januari 2016), zullen de hoofd- en deelvragen beantwoord gaan worden.

	Week 1	Week 2	Week 3	Week 4	Week 5
Deelvraag 1					
Deelvraag 2					
Deelvraag 3					
Hoofdvraag					

Voor het opstellen van een lijst met de verschillende systemen die door medewerkers van Info Support gebruikt worden (deelvraag 1) zijn ongeveer 2 dagen nodig. Dit onderdeel zal in de eerste week van het onderzoek afgerond zijn.

De 2^e deelvraag is een literatuuronderzoek nodig dat ongeveer 2 a 3 dagen duurt. Dit zal aan het begin van week 2 uitgevoerd gaan worden.

Voor deelvraag 3 is, in verhouding met deelvraag 2, iets meer tijd nodig om alle vragen te kunnen beantwoord. Hierbij moet onderzocht worden of er bestaande software is dat een basis kan vormen als universele connector. Voor deze vraag is ongeveer een week nodig om mogelijke oplossingen op te zoeken en eventueel uit te proberen of de functionaliteiten in deze pakketten aanwezig zijn.

De laatste week van de planning wordt gebruik voor eventuele uitloop en wordt de hoofdvraag beantwoord en een conclusie geschreven.

4 Bijlage A - Deelvragen

1. Welke informatie wordt door een ontwikkelaar van Info Support bekeken?

1.1	Welke systemen worden er gebruikt?
1.2	Waar wordt elk systeem voor gebruikt?
1.3	Waar is het systeem te vinden?
1.4	Voor wie is het systeem bereikbaar?

2. Hoe kan data bij de verschillende systemen opgehaald worden?

2.1	Welke gegevens van een systeem moeten opgehaald worden?
2.2	Hoe kan deze API benaderd worden?
2.2.1	<i>Waar is de API te vinden?</i>
2.2.2	<i>In wat voor formaat is de API</i>
2.2.3	<i>Zijn er kosten voor deze API</i>
2.2.4	<i>Zijn er restricties aan de API?</i>

3. Met welke framework kan een universele connector gemaakt worden?

3.1	Welke functionaliteiten moet de connector krijgen?
3.2	Welke frameworks zijn er?
3.2.1	<i>Wat zijn de kosten?</i>
3.2.2	<i>Op welk platform draait elk framework?</i>
3.2.3	<i>Wie is de ontwikkelaar / fabrikant?</i>
3.2.4	<i>Welke eisen worden ondersteunt?</i>
3.3	Wat is het meest geschikte framework?

Onderzoeksrapport

Info Support Butler



Onderzoeksrapport

Titel	Onderzoeksrapport
Project/Onderwerp	Info Support Butler
Versie	1.1
Status	Definitief
Datum	21-03-2016
Bestand	Onderzoeksrapport
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	07-01-2015	Maikel Stuivenberg	Creatie
0.2	Concept	20-01-2015	Maikel Stuivenberg	- Spel-/typfouten - Antwoord op deelvraag 3 toegevoegd - Conclusies toegevoegd aan alle deelvragen - Conclusie op hoofdvraag aangepast
1.0	Definitief	01-02-2016	Maikel Stuivenberg	- Hoofdvraag duidelijker vermeld
1.1	Definitief	22-02-2016	Maikel Stuivenberg	- Tekstuele fouten gewijzigd.

Distributie

Versie	Status	Datum	Aan
0.1	Concept	07-01-2016	Marco Kuiper, John Gorter
1.0	Definitief	11-02-2016	De Haagse Hogeschool
1.1	Definitief	29-02-2016	De Haagse Hogeschool (TTA)
1.1	Definitief	21-03-2016	De Haagse Hogeschool

Referenties

Code	Bron

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

1 **Samenvatting**

In dit onderzoeksrapport is onderzoek gedaan naar het realiseren van een universele connector voor de "Info Support Butler" opdracht. Om deze connector te kunnen is het nodig te weten welke systemen er door medewerkers van Info Support gebruikt worden en of het mogelijk is om data bij deze systemen op te halen. Daarnaast wordt er in dit onderzoek gekeken met welke frameworks en eisen de connector gebouwd kan worden.

In het onderzoek zijn verschillende systemen naar voren gekomen die (dagelijks) door medewerkers van Info Support gebruikt worden. Zo kijken ontwikkelaars 's ochtends of de build 's nachts goed zijn uitgevoerd en of alle automatische testen zijn geslaagd. Naast de ontwikkelhulpmiddelen die geraadpleegd worden, wordt er ook dagelijks gebruik gemaakt van systemen zoals het nieuws van de NOS en Nu.nl en wordt, voordat een medewerker weggaat, gekeken of er flitsers zijn en waar de files staan.

Voor de gebruiker is het mogelijk om alle systemen te benaderen, maar voor een webservice zal dit lastiger zijn om aan alle informatie te komen. Hiervoor is in hoofdstuk 3.1.5 onderzocht hoe data van verschillende systemen opgehaald kan worden.

De bruikbare systemen uit deelvraag 2 en eisen vanuit de opdrachtgever vormen samen een lijst met eisen voor de keuze van een framework. In de derde deelvraag van dit onderzoek wordt gekeken welke systemen. Zoals in de conclusie geschreven staat, is er niet één 'goede' oplossing om een universele connector te maken, maar zijn er meerdere mogelijkheden met elk zijn eigen voor- en nadelen.

Inhoudsopgave

1	Samenvatting	37
2	Inleiding	39
3	Resultaten.....	40
3.1	Welke informatie wordt door een ontwikkelaar van Info Support bekeken? .	41
3.1.1	Welke systemen worden er gebruikt?.....	44
3.1.2	Waar wordt elk systeem voor gebruikt?.....	45
3.1.3	Waar is het systeem te vinden?.....	47
3.1.4	Voor wie is het systeem bereikbaar?	48
3.1.5	Conclusie	50
3.2	Hoe kan data bij de verschillende systemen opgehaald worden?.....	51
3.2.1	Welke gegevens van een systeem moeten opgehaald worden?.....	56
3.2.2	Hoe kan deze API benaderd worden?	58
3.2.3	Conclusie	68
3.3	Met welke framework kan een universele connector gemaakt worden?	69
3.3.1	Aan welke eisen moet het framework voldoen?	70
3.3.2	Welke frameworks zijn er?	72
3.3.3	Wat is het meest geschikte framework?	83
3.3.4	Conclusie	84
4	Conclusie	85
5	Literatuur.....	87
6	Begrippenlijst	90
7	Bijlage A – Interview	92
7.1	Interview met Marco Kuiper	92
7.2	Interview met Thomas Bruggink.....	93
7.3	Interview met Willem Meints.....	93

2 Inleiding

Dit onderzoek is geschreven naar aanleiding van het Info Support Butler (afstudeer)project. In deze opdracht moet er een universele connector gebouwd worden dat informatie uit verschillende systemen kan halen en kan geven aan een client (Android of iOS applicatie).

Uit vooronderzoek is gebleken dat er geen software bestaat waarmee het mogelijk is om een systeem op te zetten waarmee een universele connector gemaakt kan worden voor het Info Support Butler project. Hierdoor is dit onderzoek opgezet om te onderzoeken met welke frameworks een universele connector zelf ontwikkeld kan gaan worden.

In dit onderzoek wordt gekeken welke systemen er door medewerkers van Info Support gebruikt worden en wat de mogelijkheden zijn om data uit de verschillende systemen te halen. Naast de verschillende systemen wordt ook gekeken met welk framework een universele connector gebouwd kan worden. Om dit te achterhalen worden alle eisen aan het framework achterhaalt en wordt onderzocht welke frameworks hieraan voldoen.

3 Resultaten

Toelichting

Voor het Info Support Butler project moet er een universele connector gebouwd worden dat informatie kan ophalen uit verschillende systemen. Een systeem is hierbij een informatiebron dat door een medewerker van Info Support gebruikt wordt. Naast het ophalen van informatie moet de universele connector ook de mogelijkheid geven, aan de gebruiker, om systemen te kunnen koppelen.

Voor al deze gegevens is de volgende hoofdvraag opgesteld:

Hoe kan er een universele connector gemaakt worden?

In dit onderzoek is deze hoofdvraag opgedeeld in 3 deelvragen (en een aantal sub-deelvragen).

Deelvraag 1. Welke informatie wordt door een ontwikkelaar van Info Support bekeken?

1.1	Welke systemen worden er gebruikt?
1.2	Waar wordt elk systeem voor gebruikt?
1.3	Waar is het systeem te vinden?
1.4	Voor wie is het systeem bereikbaar?

Deelvraag 2. Hoe kan data bij de verschillende systemen opgehaald worden?

2.1	Welke gegevens van een systeem moeten opgehaald worden?
2.2	Hoe kan deze API benaderd worden?
2.2.1	<i>Waar is de API te vinden?</i>
2.2.2	<i>In wat voor formaat is de API</i>
2.2.3	<i>Zijn er kosten voor deze API</i>
2.2.4	<i>Zijn er restricties aan de API?</i>

Deelvraag 3. Met welke framework kan een universele connector gemaakt worden?

3.1	Welke functionaliteiten moet de connector krijgen?
3.2	Welke frameworks zijn er?
3.2.1	<i>Wat zijn de kosten?</i>
3.2.2	<i>Op welk platform draait elk framework?</i>
3.2.3	<i>Wie is de ontwikkelaar / fabrikant?</i>
3.2.4	<i>Welke eisen worden ondersteunt?</i>
3.3	Wat is het meest geschikte framework?

3.1 Welke informatie wordt door een ontwikkelaar van Info Support bekeken?

Toelichting

Medewerkers van Info Support gebruiken dagelijks systemen om zichzelf te voorzien van informatie. Om te achterhalen welke systemen zij gebruiken zijn er interviews gehouden met 3 ontwikkelaars van Info Support (zie bijlage A). Uit deze interviews is een lijst met systemen naar voren gekomen, met daarbij de informatie die zij uit dit systemen halen.

Om de deelvraag te kunnen beantwoorden is de vraag opgesplitst in 4 (sub-) deelvragen. De antwoorden op deze vragen zijn samengevoegd en zijn hieronder beschreven.

1.1 Welke systemen worden er gebruikt?

1.2 Waar wordt elk systeem voor gebruikt?

1.3 Waar is het systeem te vinden?

1.4 Voor wie is het systeem bereikbaar?

Onderzoek

Buienradar

Omschrijving	Met het bekijken van het weer kan er bepaald worden of de paraplu meegenomen moeten worden uit de auto. (Vooral belangrijk voor het einde van de werkdag)
Locatie	http://www.buienradar.nl
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Flitsmeister

Omschrijving	Tijdens het rijden wordt Flitsmeister gebruikt om op de hoogte gehouden te worden van flitsers op de route van de gebruiker.
Locatie	App op website (Zie ook http://www.flitsmeister.nl)
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Flitsservice

Omschrijving	Hiermee wordt vooraf gekeken of er op weg naar huis nog flitsers gesignaleerd zijn.
Locatie	http://m.flitsservice.nl/
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Google agenda

Omschrijving	Hierin staan overige afspraken in, die een medewerker die dag heeft.
Locatie	https://calendar.google.com/
Bereikbaarheid	Iedereen die een Google account heeft, kan hier op inloggen

Google verkeersinformatie

Omschrijving	Hiermee kan gekeken worden welke route genomen moet worden naar huis en hoe laat de gebruiker thuis is.
Locatie	https://www.google.nl/maps
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Kibana

Omschrijving	Kibana is een tool dat grafisch weergeeft hoeveel error-/log meldingen er zijn en wanneer deze meldingen zijn binnengekomen.
Locatie	http://knownow-elk.cloudapp.net
Bereikbaarheid	Normaal gesproken is Kibana zonder account te bekijken. Het knowNow team heeft echter een authenticatie gebouwd zodat niet iedereen bij alle gegevens kan.

knowNow

Omschrijving	knowNow is een (online) platform waar medewerkers van Info Support kennis met elkaar kunnen delen.
Locatie	http://knownow.infosupport.com
Bereikbaarheid	Iedereen van Info Support kan inloggen met zijn/haar (Cronos) account.

New Relic

Omschrijving	In New Relic kunnen de server statussen bekeken worden.
Locatie	http://newrelic.com
Bereikbaarheid	De gegevens in New Relic zijn te bekijken met een apart account

NOS

Omschrijving	De website van de NOS wordt gebruikt om het laatste nieuws te bekijken.
Locatie	http://nos.nl
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Nu.nl

Omschrijving	Nu.nl wordt, net als de NOS website, gebruikt om het laatste nieuws te bekijken.
Locatie	http://nu.nl
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Slack

Omschrijving	Slack is een communicatiemiddel die door medewerkers van Info Support gebruikt wordt. Naast de gesprekken van de medewerkers wordt hier (automatisch) ook een statusupdate gepost (in de vorm van een bericht) zodra een build geslaagd/mislukt is.
Locatie	Applicatie op desktop of mobiel, zie ook http://slack.com
Bereikbaarheid	Elke Slack gebruiker heeft een eigen account nodig om gebruik te maken van deze dienst. Daarnaast moet een gebruiker bij 'private channels' geregistreerd zijn om berichten te kunnen bekijken.

TFS Portal

Omschrijving	In de TFS portal kunnen de statussen bekeken worden van de Builds (die bijvoorbeeld 's nachts draaien). Daarnaast is het met TFS ook mogelijk om te bekijken of testen zijn geslaagd/mislukt
Locatie	https://tfs.rhea.infosupport.net/tfs
Bereikbaarheid	Iedereen van Info Support kan inloggen op de TFS, maar krijgt alleen informatie over zijn eigen projecten.

Tomtom routeinformatie

Omschrijving	Hiermee kan gekeken worden welke route genomen moet worden naar huis en hoe laat de gebruiker thuis is.
Locatie	http://routes.tomtom.nl/
Bereikbaarheid	De informatie op deze website is voor iedereen toegankelijk.

Werkagenda

Omschrijving	Hierin staan (deels) de afspraken die een medewerker die dag heeft.
Locatie	https://mobile.infosupport.com/owa/
Bereikbaarheid	Eigen agenda is beschikbaar via de Exchange server met een Info Support email adres.

Werk email

Omschrijving	Outlook (email) wordt gebruikt om te communiceren met andere medewerkers en klanten.
Locatie	https://mobile.infosupport.com/owa/
Bereikbaarheid	Eigen email is beschikbaar via de Exchange server met een Info Support email adres.

3.1.1 Welke systemen worden er gebruikt?

Toelichting

De vraag "Welke systemen worden er gebruikt?" is een onderdeel van deelvraag 1. In deze vraag wordt, door middel van een interview, onderzocht welke systemen gebruikt worden door medewerkers van Info Support. Hiervoor zijn korte interviews gehouden met 3 medewerkers (zie Bijlage A).

Onderzoek

De onderstaande lijst met systemen is gebaseerd op de interviews uit Bijlage A. Hierbij zijn dubbele resultaten samengevoegd en is de lijst gesorteerd op alfabetische volgorde.

- Buienradar
- Flitsmeister
- Flitsservice
- Google agenda
- Google verkeersinformatie
- Kibana
- knowNow
- New Relic
- NOS
- Nu.nl
- Slack
- TFS Portal
- Tomtom routeinformatie
- Werkagenda
- Werk email

3.1.2 Waar wordt elk systeem voor gebruikt?

Toelichting

De vraag "Waar wordt elk systeem voor gebruikt?" is een onderdeel van deelvraag 1 en gaat verder in op de vorige sub-deelvraag, waar aan lijst met systemen uit is gekomen. Met deze vraag wordt beantwoord waarvoor elk systeem gebruikt wordt door medewerkers van Info Support. Om de resultaten op deze vraag te achterhalen, zijn korte interviews gehouden met 3 medewerkers (zie Bijlage A).

Onderzoek

De onderstaande lijst met systemen is gebaseerd op de interviews uit Bijlage A. Hierbij is de lijst met systemen aangehouden uit sub-deelvraag 1.1 en is per systeem vermeld waar het voor gebruikt wordt.

- Buienradar
 - o Met het bekijken van het weer kan er bepaald worden of de paraplu meegenomen moeten worden uit de auto. (Vooral belangrijk voor het einde van de werkdag)
- Flitsmeister
 - o Tijdens het rijden wordt Flitsmeister gebruikt om op de hoogte gehouden te worden van flitsers op de route van de gebruiker.
- Flitsservice
 - o Hiermee wordt vooraf gekeken of er op weg naar huis nog flitsers gesignaleerd zijn.
- Google agenda
 - o Hierin staan overige afspraken in, die een medewerker die dag heeft.
- Google verkeersinformatie
 - o Hiermee kan gekeken worden welke route genomen moet worden naar huis en hoe laat de gebruiker thuis is.
- Kibana
 - o Kibana is een tool dat grafisch weergeeft hoeveel error-/log meldingen er zijn en wanneer deze meldingen zijn binnengekomen.
- knowNow
 - o knowNow is een (online) platform waar medewerkers van Info Support kennis met elkaar kunnen delen.
- New Relic
 - o In New Relic kunnen de server statussen bekeken worden.
- NOS
 - o De website van de NOS wordt gebruikt om het laatste nieuws te bekijken.
- Nu.nl
 - o Nu.nl wordt, net als de NOS website, gebruikt om het laatste nieuws te bekijken.

- Slack
 - o Slack is een communicatiemiddel die door medewerkers van Info Support gebruikt wordt. Naast de gesprekken van de medewerkers wordt hier (automatisch) ook een statusupdate gepost (in de vorm van een bericht) zodra een build geslaagd/mislukt is.
- TFS Portal
 - o In de TFS portal kunnen buildstatussen bekeken worden, van Builds die bijvoorbeeld 's nachts draaien. Daarnaast is het met TFS ook mogelijk om te bekijken of testen zijn geslaagd/mislukt.
- Tomtom routeinformatie
 - o Hiermee kan gekeken worden welke route genomen moet worden naar huis en hoe laat de gebruiker thuis is.
- Werkagenda
 - o Hierin staan (deels) de afspraken die een medewerker die dag heeft
- Werk email
 - o Outlook (email) wordt gebruikt om te communiceren met andere medewerkers en klanten.

3.1.3 Waar is het systeem te vinden?

Toelichting

De vraag "Waar is het systeem te vinden?" is een onderdeel van deelvraag 1. Met deze vraag wordt beantwoord hoe een medewerker van Info Support elk systeem (uit vraag 1.1) benaderd. Om de resultaten op deze vraag te achterhalen, zijn korte interviews gehouden met 3 medewerkers van Info Support (zie Bijlage A).

Onderzoek

- Buienradar
 - o <http://www.buienradar.nl>
- Flitsmeister
 - o App op website (Zie ook <http://www.flitsmeister.nl>)
- Flitsservice
 - o <http://m.flitsservice.nl>
- Google agenda
 - o <http://calendar.google.com>
- Google verkeersinformatie
 - o <http://www.google.nl/maps>
- Kibana
 - o <http://knownow-elk.cloudapp.net>
- knowNow
 - o <http://knownow.infosupport.com>
- New Relic
 - o <http://newrelic.com>
- NOS
 - o <http://nos.nl>
- Nu.nl
 - o <http://nu.nl>
- Slack
 - o Applicatie op desktop of mobiel, zie ook <http://slack.com>
- TFS Portal
 - o <https://tfs.rhea.infosupport.net/tfs>
- Tomtom routeinformatie
 - o <http://routes.tomtom.nl>
- Werkagenda
 - o <https://mobile.infosupport.com/owa/>
- Werk email
 - o <https://mobile.infosupport.com/owa/>

3.1.4 Voor wie is het systeem bereikbaar?

Toelichting

De vraag "Voor wie is het systeem bereikbaar?" is een onderdeel van deelvraag 1. Met deze vraag wordt beantwoord of iedereen de gegevens van een systeem kan bekijken, of dat er een login vereist is. Om de resultaten op deze vraag te achterhalen, zijn korte interviews gehouden met 3 medewerkers van Info Support (zie Bijlage A).

Onderzoek

- Buienradar
 - o Beschikbaar voor iedereen.
- Flitsmeister
 - o Beschikbaar voor iedereen.
- Flitsservice
 - o Beschikbaar voor iedereen.
- Google agenda
 - o Eigen agenda is toegankelijk met een Google account.
- Google verkeersinformatie
 - o Beschikbaar voor iedereen.
- Kibana
 - o Zonder account toegankelijk, knowNow team heeft echter een eigen authenticatie module ingebouwd.
- knowNow
 - o Beschikbaar voor iedereen met een Info Support (Cronos) account.
- New Relic
 - o Eigen servers zijn beschikbaar met een New Relic account.
- NOS
 - o Beschikbaar voor iedereen.
- Nu.nl
 - o Beschikbaar voor iedereen.
- Slack
 - o Eigen account om te communiceren en groepen (channels) te bekijken.
- TFS Portal
 - o Eigen projecten zijn beschikbaar voor iedereen van een Info Support (Cronos) account.

- Tomtom routeinformatie
 - o Beschikbaar voor iedereen
- Werkagenda
 - o Eigen agenda is beschikbaar via de Exchange server met een Info Support email adres.
- Werk email
 - o Eigen email is beschikbaar via de Exchange server met een Info Support email adres.

3.1.5 Conclusie

Uit gesprekken met een aantal medewerkers van Info Support zijn 15 systemen gekomen die dagelijks gebruikt worden als informatievoorziening. Hieronder zijn systemen als nieuws en weer bronnen te vinden, maar zijn ook systemen te vinden die voor specifieke projecten gebruikt worden.

De lijst met systemen, voortgekomen uit deelvraag 1: (zie hoofdstuk 3.1)

- Buienradar
- Flitsmeister
- Flitsservice
- Google agenda
- Google verkeersinformatie
- Kibana
- knowNow
- New Relic
- NOS
- Nu.nl
- Slack
- TFS Portal
- Tomtom routeinformatie
- Werkagenda
- Werk email

3.2 Hoe kan data bij de verschillende systemen opgehaald worden?

Toelichting

Voor elk systeem uit hoofdstuk 3.1 moet bekeken gaan worden of het mogelijk is om via een API-informatie op te halen uit het systeem, waar deze API zich bevindt en of alle benodigde informatie (uit de vorige deelvraag) ook opgehaald kan worden.

Om de deelvraag te kunnen beantwoorden is de vraag opgesplitst in 2 vragen. De antwoorden op deze vragen zijn samengevoegd en zijn hieronder beschreven.

2.1 Welke gegevens van een systeem moeten opgehaald worden?

2.2 Hoe kan deze API benaderd worden?

2.2.1 Waar is de API te vinden? 3.2.2.1

2.2.2 In wat voor formaat is de API?

2.2.3 Zijn er kosten voor deze API?

2.2.4 Zijn er restricties aan de API?

Onderzoek

Buienradar

Benodigde informatie	Weer voor de komende dag.
Locatie API	<i>Niet aanwezig</i>
Formaat	N.v.t.
Kosten	N.v.t.
Restricties	N.v.t.

Flitsmeister

Benodigde informatie	Flitserlocaties
Locatie API	<i>Niet aanwezig</i>
Formaat	N.v.t.
Kosten	N.v.t.
Restricties	N.v.t.

Flitsservice

Benodigde informatie	Flitserlocaties
Locatie	Flitserlocaties http://www.flitsservice.nl/rss.xml
Formaat	XML
Kosten	Gratis
Restricties	Alleen voor privédoeleinde. Alleen de eerste 40 tekens van elke regel worden weergegeven. (Flitsservice RSS Feed, sd)

Google agenda

Benodigde informatie	Afspraken voor de komende dag.
Locatie	Ophalen van verschillende agenda's https://www.googleapis.com/calendar/v3/users/me/calendarList?key=xxx Ophalen van agenda items https://www.googleapis.com/calendar/v3/calendars/{calendarId}/events?key=xxx
Formaat	JSON
Kosten	Gratis, met een maximum van 1.000.000 query's per dag. <i>Let op: dit aantal is een som van alle query's naar de calendar API.</i>

Google verkeersinformatie

Locatie	Route naar een adres https://maps.googleapis.com/maps/api/directions/{output}?origins=xxx&destinations=xxx&key=xxx
Formaat	JSON, XML <i>De gewenste output moet aangegeven worden in de url ("{"output}").</i>
Kosten	Gratis, met een maximum van 2500 requests per dag.
Restricties	Login door middel van OAuth. De ontwikkelaar heeft een account nodig waarmee een client ID aangevraagd kan worden. (Google Calendar API v3 Authorizing, sd)
Locatie	Route naar een adres https://maps.googleapis.com/maps/api/directions/{output}?origins=xxx&destinations=xxx&key=xxx

Kibana

Benodigde informatie	Het aantal error-/log meldingen.
Locatie API	<i>Niet aanwezig</i>
Formaat	N.v.t.
Kosten	N.v.t.
Restricties	N.v.t.

knowNow

Benodigde informatie	Laatste toegevoegde artikelen.
Locatie	De nieuwste artikelen https://knownow.infosupport.com/api/search/new
Formaat	JSON
Kosten	Gratis
Restricties	De gebruiker moet ingelogd zijn door middel van OAuth via de ADFS van Info Support.

New Relic

Benodigde informatie	Huidige server statussen.
Locatie	Huidige server statussen https://api.newrelic.com/v2/servers.json
Formaat	JSON, XML <i>Door in de URL ".json" te wijzigen naar ".xml" wordt er een xml output gegenereerd.</i>
Kosten	Gratis
Restricties	Elk account moet zijn eigen API Key aanmaken. Deze moet meegegeven worden bij elke request door middel van een "X-API-Key" header.

NOS

Benodigde informatie	Laatste nieuwsberichten.
Locatie API	<i>Niet aanwezig</i>
Formaat	N.v.t.
Kosten	N.v.t.
Restricties	N.v.t.

Nu.nl

Benodigde informatie	Laatste nieuwsberichten.
Locatie	Laatste nieuwsberichten http://www.nu.nl/rss.html
Formaat	XML (Atom)
Kosten	Gratis
Restricties	<i>Geen</i>

OpenWeatherMap

Benodigde informatie	Weer voor de komende dag.
Locatie	Weer voor de komende dagen http://api.openweathermap.org/data/2.5/forecast?q=Veenendaal,NL&mode=json&appid=xxx
Formaat	JSON, XML of HTML <i>Door middel van de 'mode' variabele kan aangegeven worden (keuze uit de bovenstaande 3 opties) in welk formaat de API data terug moet geven.</i>
Kosten	Gratis, met een limiet van 60 keer per minuut en 50.000 keer op een dag.
Restricties	<i>Er is een API key nodig om weersvoorspellingen op te kunnen halen.</i> (OpenWeatherMap Prices, sd)
Opmerking	OpenWeatherMap komt niet voort uit voorgaande hoofdstukken, waarbij onderzocht is welke systemen gebruikt worden. Zoals te lezen in hoofdstuk 3.3.2.1. is dit systeem toegevoegd als alternatief voor andere weersvoorspellingssystemen.

Slack

Benodigde informatie	Laatste berichten in een kanaal (channel).
Locatie	Laatste berichten in een kanaal (channel) https://slack.com/api/groups.history?token={xxx}&channel={G000}&count=5
Formaat	JSON
Kosten	Gratis
Restricties	Login door middel van OAuth. Met OAuth kan een Access Token opgevraagd worden die niet verloopt (Slack OAuth, sd)

TFS Portal

Benodigde informatie	Laatste buildstatussen; Aantal gelukke/mislukte testen.
Locatie	Overzicht van builds https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/builds?api-version=2.0[&\$top=10] Build informatie https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/Builds/{buildid}/timeline?api-version=2.0
Formaat	JSON
Kosten	Gratis
Restricties	

Tomtom routeinformatie

Benodigde informatie	Welke route naar huis; Waar staan de files; Hoelang is het rijden naar huis.
Locatie	Route naar huis - Verwachte aankomsttijd - Waar staan de files https://api.tomtom.com/routing/1/calculateRoute/point1:point2?key=xx
Formaat	JSON of XML <i>Standaard wordt er een JSON-output gegeven, dit is te wijzigen naar XML door de "Content-type" header, in de request, toe te voegen of aan te passen.</i>
Kosten	Gratis om te testen. Voor productie moet er een productie API key aangevraagd worden, waar welk kosten aan zitten. (TomTom Developer Support FAQ, sd)
Restricties	Om informatie op te vragen is een API key nodig. Deze kan, voor test doeleinde, gratis aangevraagd worden (aanvraag kan 24 uur duren). (TomTom Developer Support FAQ, sd)

Werkagenda

Benodigde informatie	Afspraken voor de komende dag
Locatie API	<i>Niet aanwezig</i>
Formaat	N.v.t.
Kosten	N.v.t.
Restricties	N.v.t.

Werk e-mail

Benodigde informatie	Laatste binnengekomen e-mails
Locatie API	<i>Niet aanwezig</i>
Formaat	N.v.t.
Kosten	N.v.t.
Restricties	N.v.t.

3.2.1 Welke gegevens van een systeem moeten opgehaald worden?

Toelichting

Voor elk systeem uit hoofdstuk 3.1 moet bepaald worden welke informatie opgehaald gaat worden. Dit wordt gedaan aan de hand van de uitkomst van hoofdstuk 3.1.2 (Waar wordt elk systeem voor gebruikt?), waarin staat beschreven waar elk systeem, door een medewerker van Info Support, voor gebruikt wordt. Het antwoord op deze vraag wordt gebruikt in de volgende sub-deelvragen om de juiste API's te kunnen opzoeken.

Onderzoek

- Buienradar
 - o Weer voor de komende dag.
- Flitsmeister
 - o Flitserlocaties.
- Flitsservice
 - o Flitserlocaties.
- Google agenda
 - o Afspraken voor de komende dag.
- Google verkeersinformatie
 - o Route naar huis.
- Kibana
 - o Het aantal error-/log meldingen.
- knowNow
 - o Laatste toegevoegde artikelen.
- New Relic
 - o Huidige server statussen.
- NOS
 - o Laatste nieuwsberichten.
- Nu.nl
 - o Laatste nieuwsberichten.
- Slack
 - o Laatste berichten in een kanaal (channel)
- TFS Portal
 - o Laatste buildstatussen;
 - o Aantal gelukke/mislukte testen.

- Tomtom routeinformatie
 - o Route naar huis
 - Verwachte aankomsttijd
 - Waar staan de files
- Werkagenda
 - o Afspraken voor de komende dag
- Werk email
 - o Laatste binnengekomen e-mails

3.2.2 Hoe kan deze API benaderd worden?

Toelichting

De vraag "Hoe kan deze API benaderd worden?" is onderdeel van deel vraag 2. Het ophalen van gegevens heeft verschillende eigenschappen zoals een locatie, bestandsformaat en mogelijke authenticatie.

Om de huidige vraag goed te kunnen beantwoorden is de vraag verder opgedeeld in de volgende 4 vragen:

2.2.1 Waar is de API te vinden? 3.2.2.1

2.2.2	In wat voor formaat is de API?
2.2.3	Zijn er kosten voor deze API?
2.2.4	Zijn er restricties aan de API?

Onderzoek

Flitsservice

Locatie	Flitserlocaties http://www.flitsservice.nl/rss.xml
Formaat	XML
Kosten	Gratis
Restricties	Alleen voor privédoeleinde. Alleen de eerste 40 tekens van elke regel worden weergegeven. (Flitsservice RSS Feed, sd)

Google Agenda

Locatie	Ophalen van verschillende agenda's https://www.googleapis.com/calendar/v3/users/me/calendarList?key=xxx Ophalen van agenda items https://www.googleapis.com/calendar/v3/calendars/{calendarId}/events?key=xxx
Formaat	JSON
Kosten	Gratis, met een maximum van 1.000.000 query's per dag. <i>Let op: dit aantal is een som van alle query's naar de calendar API.</i>
Restricties	Login door middel van OAuth. De ontwikkelaar heeft een account nodig waarmee een client ID aangevraagd kan worden. (Google Calendar API v3 Authorizing, sd)

Google Verkeersinformatie

Locatie	Route naar een adres https://maps.googleapis.com/maps/api/directions/{output}?origins=xxx&destinations=xxx&key=xxx
Formaat	JSON, XML <i>De gewenste output moet aangegeven worden in de url ("{"output}").</i>
Kosten	Gratis, met een maximum van 2500 requests per dag.

Restricties	Login door middel van OAuth. De ontwikkelaar heeft een account nodig waarmee een client ID aangevraagd kan worden. (Google Calendar API v3 Authorizing, n.d.)
-------------	---

knowNow

Locatie	De nieuwste artikelen https://knownow.infosupport.com/api/search/new
Formaat	JSON
Kosten	Gratis
Restricties	De gebruiker moet ingelogd zijn door middel van OAuth via de ADFS van Info Support.

New Relic

Locatie	Huidige server statussen https://api.newrelic.com/v2/servers.json
Formaat	JSON, XML <i>Door in de URL ".json" te wijzigen naar ".xml" wordt er een xml output gegenereerd.</i>
Kosten	Gratis
Restricties	Elk account moet zijn eigen API Key aanmaken. Deze moet meegegeven worden bij elke request door middel van een "X-API-Key" header.

Nu.nl

Locatie	Laatste nieuwsberichten http://www.nu.nl/rss.html
Formaat	XML (Atom)
Kosten	Gratis
Restricties	Geen

OpenWeatherMap

Locatie	Weer voor de komende dagen http://api.openweathermap.org/data/2.5/forecast?q=Veenendaal,NL&mode=json&appid=xxx
Formaat	JSON, XML of HTML <i>Door middel van de 'mode' variabele kan aangegeven worden (keuze uit de bovenstaande 3 opties) in welk formaat de API-data terug moet geven.</i>
Kosten	Gratis, met een limiet van 60 keer per minuut en 50.000 keer op een dag.
Restricties	Er is een API key nodig om weersvoorspellingen op te kunnen halen. (OpenWeatherMap Prices, n.d.)
Opmerking	OpenWeatherMap komt niet voort uit voorgaande hoofdstukken, waarbij onderzocht is welke systemen gebruikt worden. Zoals te lezen in hoofdstuk 3.3.2.1. is dit systeem toegevoegd als alternatief voor andere weersvoorspellingssystemen.

Slack

Locatie	Laatste berichten in een kanaal (channel) https://slack.com/api/groups.history?token={xxx}&channel={G000}&count=5
Formaat	JSON
Kosten	Gratis
Restricties	Login door middel van OAuth. Met OAuth kan een Acces Token opgevraagd worden die niet verloopt (Slack OAuth, n.d.)

TFS Portal

Locatie	Overzicht van builds https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/builds?api-version=2.0[&\$top=10] Build informatie https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/Buils/{buildid}/timeline?api-version=2.0
Formaat	JSON
Kosten	Gratis
Restricties	De TFS portal zit net als knowKnow achter een login van Info Support. Hierdoor is het nodig om via de ADFS ingelogde gebruiker te hebben (dit kan door middel van OAuth)

Tomtom Routeinformatie

Locatie	Route naar huis - Verwachte aankomsttijd - Waar staan de files https://api.tomtom.com/routing/1/calculateRoute/point1:point2?key=xx
Formaat	JSON of XML <i>Standaard wordt er een JSON-output gegeven, dit is te wijzigen naar XML door de "Content-type" header, in de request, toe te voegen of aan te passen.</i>
Kosten	Gratis om te testen. Voor productie moet er een productie API key aangevraagd worden, waar welk kosten aan zitten. (TomTom Developer Support FAQ, n.d.)
Restricties	Om informatie op te vragen is een API key nodig. Deze kan, voor test doeleinde, gratis aangevraagd worden (aanvraag kan 24 uur duren). (TomTom Developer Support FAQ, n.d.)

3.2.2.1 Waar is de API te vinden?

Toelichting

Voor elk van de benodigde gegevens (zie hoofdstuk 3.2.1) moet onderzocht worden of er een API beschikbaar is om deze gegevens op te kunnen halen. Indien het mogelijk is om een de gegevens op te halen, zal de locatie van de API gegeven worden in het onderstaande overzicht. De mogelijkheden van de API's zullen onderzocht worden door middel van een literatuuronderzoek (op het internet)

Onderzoek

- **Buienradar**
 - o Weer voor de komende dagen.
Buienradar biedt geen API aan, wel is het mogelijk om via OpenWeatherMap gegevens over het weer te krijgen.
- **Flitsmeister**
 - o Flitserlocaties
Flitsmeister biedt geen API aan.
- **Flitsservice**
 - o Flitserlocaties
<http://www.flitsservice.nl/rss.xml>
- **Google agenda**
 - o Afspraken voor de komende dag
 - Ophalen van verschillende agenda's
<https://www.googleapis.com/calendar/v3/users/me/calendarList?key=xxx>
 - Ophalen van agenda items
<https://www.googleapis.com/calendar/v3/calendars/{calendarId}/events?key=xxx>
- **Google verkeersinformatie**
 - o Route naar een adres
<https://maps.googleapis.com/maps/api/directions/{output}?origins=xx&destinations=xxx&key=xxx>
- **Kibana**
 - o Het aantal error-/log meldingen
Kibana bied zelf geen API aan, wel is het mogelijk om het onderliggende systeem (ElasticSearch) aan te roepen door middel van een query naar de Search API. Echter heeft elk systeem zijn eigen variabelen / eigenschappen, waardoor het niet mogelijk is om een query te schrijven die voor meerdere gebruikers zal werken.
- **knowNow**
 - o De nieuwste artikelen
<https://knownow.infosupport.com/api/search/new>
- **New Relic**
 - o Huidige server statussen
<https://api.newrelic.com/v2/servers.json>

- **NOS**
 - o Laatste nieuwsberichten
De NOS biedt geen API (meer) aan om het nieuws op te halen. NOS.nl biedt wel een RSS-feed aan, echter is de lijst met feeds op moment van schrijven (11-12-2015) niet beschikbaar.
- **Nu.nl**
 - o Laatste nieuwsberichten
<http://www.nu.nl/rss.html>
- **OpenWeatherMap**
 - o Weer voor de komende dagen
<http://api.openweathermap.org/data/2.5/forecast?q=Veenendaal,NL&mode=json&appid=xxx>
- **Slack**
 - o Laatste berichten in een kanaal (channel)
<https://slack.com/api/groups.history?token={xxx}&channel={G000}&count=5>
- **TFS Portal**
 - o Laatste buildstatussen
 - Overzicht van builds
[https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/builds?api-version=2.0\[&\\$top=10\]](https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/builds?api-version=2.0[&$top=10])
 - Build informatie
https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/Buids/{buildid}/timeline?api-version=2.0
 - o Aantal gelukke/mislukte testen;
[https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/test/runs?includeRunDetails=true&api-version=1.0\[&\\$top=10&...\]](https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/test/runs?includeRunDetails=true&api-version=1.0[&$top=10&...])
- **Tomtom routeinformatie**
 - o Route naar huis
 - Verwachte aankomsttijd
 - Waar staan de files<https://api.tomtom.com/routing/1/calculateRoute/point1:point2?key=xx>
- **Werkagenda**
 - o Afspraken voor de komende dag
Deze zijn op te halen via een Exchange server, echter is er niet voldoende documentatie gevonden over de wijze waarop de data opgehaald kan worden.
- **Werk email**
 - o Laatste e-mails
Deze zijn op te halen via een Exchange server, echter is er niet voldoende documentatie gevonden over de wijze waarop de data opgehaald kan worden.

3.2.2.2 In wat voor formaat is de API?

Toelichting

Elke API uit hoofdstuk 3.2.2.1 heeft zijn eigen eigenschappen, in deze sub-deelvraag wordt gekeken in welk formaat de API-data teruggeeft.

Onderzoek

- **Flitsservice**
 - o <http://www.flitsservice.nl/rss.xml>
Output: XML
- **Google agenda**
 - o <https://www.googleapis.com/calendar/v3/users/me/calendarList?key=xxx>
Output: JSON
 - o <https://www.googleapis.com/calendar/v3/calendars/{calendarId}/events?key=xxx>
Output: JSON
- **Google verkeersinformatie**
 - o <https://maps.googleapis.com/maps/api/directions/{output}?origins=xxx&destinations=xxx&key=xxx>
Output: JSON, XML
De gewenste output moet aangegeven worden in de url ("{"output}").
- **knowNow**
 - o <https://knownow.infosupport.com/api/search/new>
Output: JSON
- **New Relic**
 - o <https://api.newrelic.com/v2/servers.json>
Output: JSON, XML
Door in de URL ".json" te wijzigen naar ".xml" wordt er een xml output gegenereerd.
- **Nu.nl**
 - o <http://www.nu.nl/rss.html>
Output: XML (Atom)
- **OpenWeatherMap**
 - o <http://api.openweathermap.org/data/2.5/forecast?q=Veenendaal,NL&mode=json&appid=xxx>
Output: JSON, XML of HTML
Door middel van de 'mode' variabele kan aangegeven worden (keuze uit de bovenstaande 3 opties) in welk formaat de API-data terug moet geven.

- **Slack**
 - o <https://slack.com/api/groups.history?token={xxx}&channel={G000}&count=5>
Output: JSON
- **TFS Portal**
 - o [https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/builds?api-version=2.0\[&\\$top=10\]](https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/builds?api-version=2.0[&$top=10])
Output: JSON
 - o https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/Builds/{buildid}/timeline?api-version=2.0
Output: JSON
 - o https://tfs.rhea.infosupport.net/tfs/DefaultCollection/{project}/_apis/build/Builds/{buildid}/timeline?api-version=2.0
Output: JSON
- **Tomtom routeinformatie**
 - o <https://api.tomtom.com/routing/1/calculateRoute/point1:point2?key=xx>
Output: JSON of XML
Standaard wordt er een JSON-output gegeven, dit is te wijzigen naar XML door de "Content-type" header, in de request, toe te voegen of aan te passen.

3.2.2.3 Zijn er kosten voor deze API?

Toelichting

Elke API uit hoofdstuk 3.2.2.1 heeft zijn eigen eigenschappen, in deze sub-deelvraag wordt gekeken of de API gratis beschikbaar is, of dat hier kosten aan verbonden zijn.

Onderzoek

- **Flitsservice**
Kosten: Gratis.
- **Google agenda**
Kosten: Gratis, met een maximum van 1.000.000 query's per dag.
Let op: dit aantal is een som van alle query's naar de calendar API.
- **Google verkeersinformatie**
Kosten: Gratis, met een maximum van 2500 requests per dag.
- **knowNow**
Kosten: Gratis, wordt gehost binnen Info Support
- **New Relic**
Kosten: Gratis
- **Nu.nl**
Kosten: Gratis
- **OpenWeatherMap**
Kosten: Gratis, met een limiet van 60 keer per minuut en 50.000 keer op een dag.
- **Slack**
Kosten: Gratis
- **TFS Portal**
Kosten: Gratis, wordt gehost binnen Info Support
- **Tomtom routeinformatie**
Kosten: Gratis om te testen. Voor productie moet er een productie API key aangevraagd worden, waar welk kosten aan zitten. (TomTom Developer | Support FAQ, n.d.)

3.2.2.4 Zijn er restricties aan de API?

Toelichting

Elke API uit hoofdstuk 3.2.2.1 heeft zijn eigen eigenschappen, in deze sub-deelvraag wordt gekeken of er naast de kosten (hoofdstuk 3.2.2.3) ook nog andere restricties kunnen zijn aan de verschillende API's.

Onderzoek

- **Flitsservice**
Restricties: Alleen voor privédoeleinde.
Alleen de eerste 40 tekens van elke regel worden weergegeven.
(Flitsservice RSS Feed, n.d.)
- **Google agenda**
Restricties: Login door middel van OAuth.
De ontwikkelaar heeft een account nodig waarmee een client ID aangevraagd kan worden.
(Google Calendar API v3 | Authorizing, n.d.)
- **Google verkeersinformatie**
Restricties: De ontwikkelaar heeft een API key nodig om de API te kunnen gebruiken.
(Google Maps API | API Key, n.d.)
- **knowNow**
Restricties: De gebruiker moet ingelogd zijn door middel van OAuth via de ADFS van Info Support.
- **New Relic**
Restricties: Elk account moet zijn eigen API Key aanmaken. Deze moet meegegeven worden bij elke request door middel van een "X-API-Key" header.
- **Nu.nl**
Restricties: *Geen*
- **OpenWeatherMap**
Restricties: *Er is een API key nodig om weersvoorspellingen op te kunnen halen.*
(OpenWeatherMap Prices, n.d.)
- **Slack**
Restricties: Login door middel van OAuth. Met OAuth kan een Acces Token opgevraagd worden die niet verloopt (Slack OAuth, n.d.)
- **TFS Portal**
Restricties: De TFS portal zit net als knowKnow achter een login van Info Support. Hierdoor is het nodig om via de ADFS ingelogde gebruiker te hebben (dit kan door middel van OAuth).
- **Tomtom routeinformatie**

Restricties: *Om informatie op te vragen is een API key nodig. Deze kan, voor test doeleinde, gratis aangevraagd worden (aanvraag kan 24 uur duren). (TomTom Developer | Support FAQ, n.d.)*

3.2.3 Conclusie

In deelvraag 2 (hoofdstuk 3.2) is onderzocht of, bij elk systeem van deelvraag 1, het mogelijk is om door middel van een API data op te halen bij alle verschillende systemen. Uit onderzoek blijkt dat het niet mogelijk is om dit voor elk systeem te doen, waarbij vooral informatie uit systemen die publiekelijk toegankelijk zijn, minder vaak een API beschikbaar hebben.

Naast de systemen uit deelvraag 1, is er in deze deelvraag ook een alternatief gezocht voor de systemen die geen API beschikbaar hebben. In de onderstaande tabel is een compleet overzicht te zien van alle systemen die onderzocht zijn in deelvraag 2, met eventuele alternatieven.

Systeem	API beschikbaar
Buienradar	Nee, alternatief: OpenWeaterMap
Flitsmeister	Nee, alternatief: Geen
Flitsservice	Nee, alternatief: Geen
Google agenda	Ja
Google verkeersinformatie	Ja
Kibana	Nee, alternatief: Geen
knowNow	Ja
New Relic	Ja
NOS	Nee, alternatief: Nu.nl
Nu.nl	Ja
OpenWeaterMap	Ja
Slack	Ja
TFS Portal	Ja
Tomtom routeinformatie	Ja
Werkagenda	Ja
Werk e-mail	Ja

3.3 Met welke framework kan een universele connector gemaakt worden?

Toelichting

Voor het "Info Support Butler" project moet een universele connector gebouwd worden waar de verschillende systemen uit hoofdstuk 3.1.5 aan verbonden kunnen worden. In deze vraag wordt gekeken welke frameworks er zijn om deze universele connector te gaan ontwikkelen.

Om de deelvraag te kunnen beantwoorden is de vraag opgesplitst in 3 sub-deelvragen:

3.1	Aan welke eisen moet het framework voldoen ?
3.2	Welke frameworks zijn er?
3.2.1	Wat zijn de kosten?
3.2.2	Op welk platform draait elk framework?
3.2.3	Wie is de ontwikkelaar / fabrikant?
3.2.4	Welke eisen worden ondersteunt?
3.3	Wat is het meest geschikte framework?

Onderzoek

Er zijn 9 eisen waaraan een framework moet voldoen, zodat een universele connector gebouwd kan worden:

1. Ondersteuning voor JSON en XML.
2. Mogelijkheid tot opslaan van gegevens
3. Instellen van headers
4. De client moet informatie van verschillende systemen kunnen ontvangen
5. De connector data kunnen ontvangen van een client
6. De connector moet verbonden kunnen worden met een database
7. De connector moet uitbreidbaar zijn.
8. De universele connector moet om kunnen gaan met verschillende gebruikers.
9. De webservice moet data ontvangen en/of terug geven in verschillende versies.

In dit onderzoek zijn 6 frameworks onderzocht of deze voldoen aan de bovenstaande eisen. Zoals te zien in de onderstaande tabel, worden de eerste 6 eisen door alle frameworks ondersteund en zijn de laatste 2 eisen minder beschikbaar in de onderzochte frameworks.

Framework	9. Meerdere API versies	8. Meerdere gebruikers	7. Uitbreidbaar	6. Database ondersteuning	5. Ontvangen van data	4. Terugsturen van data	3. Instellen van headers	2. Opslaan van data	1. JSON / XML
	-	✓	✓	✓	✓	✓	✓	✓	✓
	✗	✗	✓	✓	✓	✓	✓	✓	✓
	✓	✓	✓	✓	✓	✓	✓	✓	✓
	✗	✗	✓	✓	✓	✓	✓	✓	-
	✓	✗	✓	✓	✓	✓	✓	✓	✓
	-	✓	✓	✓	✓	✓	✓	✓	✓

In overleg met de opdrachtgever zijn aan de verschillende eisen gewichten gehangen, omdat bepaalde eisen belangrijker zijn dan andere eisen. Hierbij is gekozen voor een gewicht tussen de 1 en 3 (laag, normaal en hoog).

Eisen	Totaal aantal punten	9. Meerdere API versies	8. Meerdere gebruikers	7. Uitbreidbaar	6. Database ondersteuning	5. Data ontvangen	4. Terugsturen van data	3. Instellen van headers	2. Opslaan van data	1. JSON / XML
	19	1	2	2	3	3	3	1	3	1
Gewicht										

In de onderstaande tabel zijn de ✓, - en ✗ vervangen door de gewichten, hierbij wordt aan de - niet het volledige gewicht gekoppeld, maar de helft van het gewicht.

ASP.NET Web API	1	3	1	3	3	3	2	2	0,5	18,5
Slim Framework	1	3	1	3	3	3	2	0	0	16
Django REST Framework	1	3	1	3	3	3	2	2	1	19
Phalcon	0,5	3	0,5	3	3	3	2	0	0	15
Play framework	1	3	1	3	3	3	2	0	1	17
Jersey	1	3	1	3	3	3	2	2	0,5	18,5

3.3.1 Aan welke eisen moet het framework voldoen?

Toelichting

De vraag "Aan welke eisen moet het framework voldoen?" bestaat uit het opstellen van de eisen. Deze eisen komen voort uit hoofdstuk 3.1.5 (Data van verschillende systemen) samen met overige eisen, die door de opdrachtgever gesteld worden.

Onderzoek

Eisen voortgekomen uit eerdere deelvragen:

1. Ondersteuning voor JSON en XML.
2. Mogelijkheid tot opslaan van gegevens (zoals API keys voor OAuth authenticatie)
3. Instellen van (custom) headers (zie Tomtom verkeersinformatie)

Overige eisen door de opdrachtgever:

4. De client (informatiescherm van de gebruiker) moet informatie van verschillende systemen kunnen ontvangen
Gegevens van verschillende systemen moeten verzameld worden en terug gestuurd kunnen worden naar de client. De meest voorkomende manier om data terug te sturen naar een client is door middel van een API met JSON of XML als data protocol (IBM | Using Internet data in Android applications, sd)
5. De connector moet data kunnen ontvangen van een client
De mobiele applicatie moet deze data kunnen versturen naar de connector. Dit kan net als bij 4. ook gedaan worden door een webservice op te zetten waar in een JSON of XML formaat data naartoe gestuurd kan worden.
6. De connector moet verbonden kunnen worden met een database (voor het opslaan van gekoppelde systemen, profielen van gebruikers en eventuele andere informatie).
7. De connector moet uitbreidbaar zijn.
Aan de connector moet door de ontwikkelaar systemen toegevoegd kunnen worden, die vervolgens door de gebruiker gekoppeld/gekozen kunnen worden. Dit kan bereikt worden door OOP (object-georiënteerd programmeren) te programmeren en verschillende design patterns toe te passen. (Gamma, Helm, Johnson, & Vlissides, 1994)
8. De universele connector moet om kunnen gaan met verschillende gebruikers.
9. De webservice (zie 4. en 5.) moet data ontvangen en/of terug geven in verschillende versies.
Bij wijzigingen aan de webservice moet de oude versie beschikbaar blijven voor de huidige applicaties, terwijl de nieuwe versie voor de nieuwe applicatie moet werken.

3.3.2 Welke frameworks zijn er?

Toelichting

Met de vraag "Welke frameworks zijn er?" wordt gezocht naar frameworks waarmee het mogelijk wordt om een universele connector te bouwen. Naast het zoeken van mogelijke frameworks, wordt in het antwoord ook besproken aan welke eisen (uit hoofdstuk 3.3.1) dit framework voldoet.

3.2.1 *Wat zijn de kosten?*

3.2.2 *Op welk platform draait dit?*

3.2.3 *Wie is de ontwikkelaar / fabrikant van deze connector?*

3.2.4 *Welke eisen worden ondersteunt?*

Onderzoek

In vooronderzoek zijn verschillende frameworks gevonden waarmee een universele connector gebouwd zou kunnen worden. Voor elk frameworks wordt in deze vraag opgezocht wat de mogelijkheden zijn.

ASP.NET Web API

Kosten	Gratis
Platform	ASP.NET Windows
Ontwikkelaar	Microsoft

Slim Framework

Kosten	Gratis
Platform	PHP Windows, Linux & OS X
Ontwikkelaar	Josh Lockhart, Andrew Smith, Rob Allen, and the Slim Framework Team

Django REST Framework

Kosten	Gratis
Platform	Python Windows, Linux & OS X
Ontwikkelaar	Tom Christie

Phalcon

Kosten	Gratis
Platform	PHP Windows, Linux & OS X
Ontwikkelaar	Vast team: Andres Gutierrez, Eduar Carvajal, Nikolas Dimopoulos, Nikolay Kirsh en Serghei Iakovlev. Daarnaast zijn er ook nog bijdrage van vele andere mensen.

Play framework

Kosten	Gratis
Platform	Java & Scala Windows, Linux & OS X
Ontwikkelaar	Typesafe / Community

Jersey

Kosten	Gratis
Platform	Java Windows, Linux & OS X
Ontwikkelaar	Oracle

In de onderstaande tabel staat per framework (uit de bovenstaande lijst) aangegeven aan welke eisen uit hoofdstuk 3.3.1 het framework voldoet. Dit is aangegeven door middel van een **✓** (standaard ingebouwd), **–** (zie opmerking) of **✗** (niet ingebouwd). Uitgebreide informatie over de features, per framework, zijn te lezen in de sub-vraag (hoofdstuk 3.3.2.4).

Framework	10. JSON / XML	11. Opslaan van data	12. Instellen van headers	13. Terugsturen van data	14. Ontvangen van data	15. Database ondersteuning	16. Uitbreidbaar	17. Meerdere gebruikers	18. Meerdere API versies
ASP.NET Web API	✓	✓	✓	✓	✓	✓	✓	✓	– ¹
Slim Framework	✓	✓	✓	✓	✓	✓	✓	✗	✗
Django REST Framework	✓	✓	✓	✓	✓	✓	✓	✓	✓
Phalcon	– ²	✓	– ³	✓	✓	✓	✓	✗	✗
Play framework	✓	✓	✓	✓	✓	✓	✓	✗	✓
Jersey	✓	✓	✓	✓	✓	✓	✓	✓	– ¹

Opmerkingen:

¹ Om dit mogelijk te maken moeten er een kleine wijziging gedaan worden in de code.

² Heeft alleen ondersteuning voor JSON in de response, daarnaast moeten er eigen implementaties gemaakt worden voor overige ondersteuning.

³ Er is geen ondersteuning voor het aanroepen van andere systemen, hiervoor zal een eigen implementatie gemaakt moeten worden i.c.m. met PHP.

3.3.2.1 Wat zijn de kosten?

Toelichting

De vraag "Wat zijn de kosten?" is een onderdeel van de vraag "Welke frameworks zijn er?". Met deze vraag wordt beantwoord wat de kosten zijn om het framework te kunnen gebruiken.

Onderzoek

Voor alle onderstaande frameworks geldt dat het gebruik van het framework gratis is. De enige kosten die voor kunnen komen, zijn kosten voor het hosten van de applicatie en het gebruik van een IDE.

ASP.NET Web API

Slim framework

Django Rest framework

Phalcon

Play framework

Jersey

3.3.2.2 Op welk platform draait elk framework?

Toelichting

De vraag "Op welk platform draait elk framework?" is een onderdeel van de vraag "Welke frameworks zijn er?". Met deze vraag wordt de lijst met frameworks uit hoofdstuk 0 genomen en verder onderzocht op welke platforms deze frameworks draaien.

Onderzoek

ASP.NET Web API

Zoals de naam van het framework al aangeeft, draait dit framework op ASP.NET (Windows) platform.

Slim framework

Slim is een framework dat gebruik maakt van PHP. Hierdoor is het mogelijk om het framework te gebruiken op onder andere Windows, OS X en Linux systemen. (PHP | Install, sd)

Django Rest framework

Django is een framework gebouwd op Python. Doordat er gebruik gemaakt wordt van Python kan het op (bijna) alle platforms, waaronder Windows, Linux en OS X, worden gedraaid. (Python | Downloads, sd)

Phalcon

Net als het Slim framework is Phalcon gebouwd op PHP. Hierdoor is het mogelijk om het op onder andere Windows, OSX en Linux te gebruiken.

Play framework

Hey Play framework draait op Java en Scala en zijn beschikbaar voor onder andere Windows, OS X en Linux (Scala | Download 2.12, sd)

Jersey

Jersey is gebouwd op Java, hier door kan het net als vele andere frameworks gebruikt worden op de meeste besturingssystemen. (Windows, OS X en Linux)

3.3.2.3 Wie is de ontwikkelaar / fabrikant?

Toelichting

De vraag "Wie is de ontwikkelaar / fabrikant?" is een onderdeel van de vraag "Welke frameworks zijn er?". Met deze vraag wordt de lijst met frameworks uit hoofdstuk 03.3.2 genomen en verder onderzocht op welke platforms deze frameworks draaien.

Onderzoek

ASP.NET Web API

Het ASP.NET Web API framework wordt ontwikkeld door Microsoft. (The ASP.NET Site | Web API, sd)

Slim framework

Het Slim framework wordt ontwikkeld door Josh Lockhart, Andrew Smith, Rob Allen, en het Slim Framework Team. (Slim Framework, sd)

Django REST framework

Het Django REST framework is van Tom Christie, maar is opengesteld op GitHub zodat iedereen aan de sourcecode kan bijdragen. (Django Rest Framework | Contributing, sd)

Phalcon

Phalcon wordt gemaakt door een vast team van mensen (Andres Gutierrez, Eduar Carvajal, Nikolas Dimopoulos, Nikolay Kirsh en Serghei Iakovlev) en heeft daarnaast ook bijdragen van heel veel andere mensen gekregen. (Phalcon | Team, sd)

Play framework

Het Play framework wordt ontwikkeld door Typesafe, daarnaast kan door iedereen een bijdrage gedaan worden aan de code. (Play Framework | Code, sd)

Jersey

Jersey is een framework dat voornamelijk ontwikkeld wordt door ontwikkelaars van Oracle. (Jersey | Team List, sd)

3.3.2.4 Welke eisen worden ondersteunt?

Toelichting

De vraag "Welke eisen worden ondersteunt?" wordt beantwoord door de lijst met eisen (hoofdstuk 3.3.1) te nemen en op het internet onderzoek te doen naar de mogelijkheden van het framework.

Onderzoek

De punten uit het onderstaande antwoord komen overeen met de nummering uit hoofdstuk 3.3.1.

ASP.NET Web API

1. De Web API framework biedt ondersteuning om gegevens naar XML of JSON om te zetten en terug te converteren. (ASP.NET | Web API | JSON and XML serialization, sd)
2. Gegevens zoals API keys kunnen opgeslagen worden in een database. De Web API heeft ingebouwde ondersteuning voor SQL Server (van Microsoft). (ASP.NET | SQL Server Database, sd)
3. Het lezen van headers kan gedaan worden uit de HttpRequestMessage, hierin zit een 'Header' collectie met alle meegestuurde headers (MSDN | HttpRequestMessage Class, sd). Het meesturen van headers kan gedaan worden door dit in de HttpResponseMessage mee te sturen (MSDN | HttpResponseMessage Class, sd).
4. Zoals beschreven in de eis, is de meest gebruikte manier om gegevens van een server naar doormiddel van een API in JSON of XML formaat. In 1. is te lezen dat ASP.NET Web API ondersteuning biedt voor beide protocollen.
5. Met de Web API kan een API gemaakt worden dat de mobiele applicatie aan kan roepen zodra er een gebruiker binnenkomt of weggaat. Dit kan net als in 4. gedaan worden door het JSON of XML formaat te gebruiken.
6. Zie 2. voor database ondersteuning.
7. Het ontwikkelen met de Web API kan gedaan worden in meerdere classes en namespaces, door het gebruik van design patterns (zie ook hoofdstuk 3.3.1) wordt het mogelijk om het systeem zo te bouwen dat er achteraf eenvoudig systemen toegevoegd kunnen worden.
8. Web API heeft ingebouwde ondersteuning voor Authenticatie en Autorisatie. Met deze ondersteuning kan doormiddel van attributen boven classes aangegeven worden of een bepaalde API call voor iedereen toegankelijk is, alleen voor bepaalde gebruikers of voor gebruikers met een bepaalde rol. (ASP.NET | Web API | Authentication and Authorization, sd)
9. Het framework heeft geen ingebouwde ondersteuning voor meerdere versie, echter is eenvoudig om dit wel in te bouwen. Dit kan gerealiseerd worden door een wijziging te maken in de 'Route' class en versie nummers te gebruiken in de namespaces. (MSDN Blogs | Using namespaces to version web apis, sd)

Slim framework

1. Het Slim framework heeft ondersteuning voor JSON, XML en URL-gecodeerde data. (Slim Framework | Request, sd)
2. Slim biedt zelf geen extra functionaliteiten aan om gegevens op te slaan, wel kan de (standaard) functionaliteiten van PHP gebruikt worden te verbinden met onder andere MySQL, MS SQL, Oracle en meer.. (PHP | PDO Drivers, sd)
3. Slim heeft ondersteuning om Headers op te vragen (Slim Framework | Request, sd) en mee te geven bij een response (Slim Framework | Response, sd). Dit gaat echter alleen over de request met de client. Bij het aanroepen van een extern systeem moet er teruggevallen worden op de methodes van PHP.
4. Zoals beschreven in de eis, is de meest gebruikte manier om gegevens van een server naar doormiddel van een API in JSON of XML formaat. In 1. is te lezen dat het Slim framework ondersteuning biedt voor beide protocollen.
5. Met het Slim framework kan een API gemaakt worden dat de mobiele applicatie aan kan roepen zodra er een gebruiker binnenkomt of weggaat. Dit kan net als in 4. gedaan worden door het JSON of XML formaat te gebruiken.
6. Zie 2. voor database ondersteuning.
7. Het ontwikkelen met Slim gebeurt in de taal PHP. Hoewel het met PHP niet verplicht is om op een Object georiënteerde manier te programmeren, maakt het Slim framework gebruik van OOP. Door bij de ontwikkeling van een webapplicatie met het Slim framework ook de OOP manier te gebruiken, wordt het later eenvoudig om extra systemen toe te voegen aan de applicatie.
8. Het Slim framework heeft geen ondersteuning voor meerdere gebruikers of het authenticeren van een gebruiker. Hiervoor zal een eigen implementatie gebouwd of gezocht moeten worden, waarmee dit wel mogelijk wordt.
9. Het Slim framework biedt geen ondersteuning voor verschillende versies van een API.

Django REST framework

1. Het Django REST framework biedt ondersteuning voor onder andere JSON en XML en kan op verschillende manieren ingesteld / gebruikt worden. (Django REST Framework | Renderers, sd)
2. Django heeft ondersteuning ingebouwd voor MySQL, PostgreSQL en SQLite en kan daarnaast ook nog ondersteuning krijgen voor andere databases met behulp van 3rd-party software. (Django | Databases, sd)
3. Met Django kunnen meegegeven headers bekeken worden (Django REST Framework | Responses, sd). Daarnaast is het ook mogelijk om bij een request naar een externe webservice headers mee te geven aan de request. (Django REST Framework, sd)
4. Zoals beschreven in de eis, is de meest gebruikte manier om gegevens van een server naar doormiddel van een API in JSON of XML formaat. In 1. is te lezen dat Django ondersteuning biedt voor beide protocollen.
5. Met het Django REST framework kan een API gemaakt worden dat de mobiele applicatie aan kan roepen zodra er een gebruiker binnenkomt of weggaat. Dit kan net als in 4. gedaan worden door het JSON of XML formaat te gebruiken.
6. Zie 2. voor database ondersteuning.
7. Net als andere ontwikkeltalen die in dit onderzoek besproken zijn, is ook Python object georiënteerd en kunnen design patterns, zoals besproken in de eisen, ook toegepast worden binnen Django/Python. (Python | Classes, sd)
8. Django REST Framework heeft standaard ingebouwde ondersteuning voor gebruikers en bied daarvoor ook meerdere manieren aan om de authenticatie te regelen (Django REST Framework | Authentication, sd)
9. Django REST Framework biedt ondersteuning voor API versies, welke op verschillende manieren meegegeven kunnen worden (onder andere headers, url en hostname) (Django REST Framework | Versioning, sd)

Phalcon

1. Phalcon biedt in de response alleen ondersteuning voor JSON (Phalcon | Http Response, sd). Daarnaast moeten eigen implementaties met behulp van PHP methodes gemaakt worden om XML en JSON te kunnen gebruiken voor andere gegevens.
2. Phalcon heeft ondersteuning om data op te slaan met behulp van de Model classes. Hierin kan verbinding gemaakt worden met verschillende databases, waaronder MySQL, PostgreSQL en SQLite. (Phalcon | DB, n.d.)
3. Er is geen ingebouwd ondersteuning voor het aanroepen van externe API's. Hierdoor moet er een eigen methode gebouwd worden, met behulp van PHP, waarmee de systemen aangeroepen kunnen worden en eventuele headers meegestuurd kunnen worden.
4. De meest gebruikte manieren om data te versturen naar een webservice is door middel van een API dat JSON of XML accepteert. Met Phalcon is het mogelijk om uit de request JSON of de tekst (zoals opgestuurd) op te halen. (Phalcon | HTTP Request, n.d.)
5. Het framework kan data van de client ontvangen in JSON of tekst formaat (zie ook 4.). Hierdoor kan de client de status van de gebruiker versturen. Daarnaast kan de status ook opgeslagen worden in verschillende soorten databases (zie 2.)
6. Zie 2. voor database ondersteuning.
7. Het Phalcon framework is gebouwd met de taal PHP. Net als bij het Slim framework is het hierdoor mogelijk om te ontwikkelen op een object georiënteerde manier.
8. Phalcon heeft geen ingebouwde ondersteuning voor meerdere gebruikers binnen een API. Hierdoor moet er een eigen implementatie gemaakt worden voor eventuele authenticatie.
9. Er is geen ondersteuning aanwezig om meerdere versies van een API bij te houden. De enige mogelijk die gebruikt kan worden, is het aanmaken van routes voor elke locatie en versie.

Play framework

1. Het Play framework heeft ondersteuning voor zowel JSON als XML in requests en responses (Play framework | JSON, sd; Play framework | XML, sd)
2. Play ondersteunt verschillende databases om gegevens in op te slaan en op te halen, zoals MySQL, PostgreSQL en SQLite (Play framework | Database, 2016)
3. Met Play is het mogelijk om andere systemen/webservices aan te roepen, tijdens het aanroepen van de systemen kunnen headers ingesteld worden. (Play framework | WS, n.d.)
4. Zoals in punt 3 en 1 te lezen is, kan Play informatie uit andere systemen ophalen en heeft het ondersteuning om de gebruiker informatie (een response) te sturen in JSON of XML formaat.
5. Het Play framework kan data ontvangen (zoals in punt 1 te lezen is) in zowel JSON als XML formaat. Deze data kan opgeslagen worden in verschillende database (zie punt 2).
6. Zie punt 2 voor de ondersteuning van de verschillende databases.
7. Het Play framework werkt op Java/Scala, hierdoor kan de connector gebouwd worden met verschillende design patterns (zie eis 7). Hierdoor wordt het eenvoudiger om later extra systemen toe te voegen aan de connector.
8. In het Play framework is geen ondersteuning ingebouwd voor meerdere gebruikers en de authenticatie daarvan.
9. Met Play kan eenvoudig geconfigureerd worden, zodat er een versie nummer meegegeven kan worden bij het aanroepen van de connector (Play framework | Routing, sd)

Jersey

1. Jersey biedt ondersteuning voor zowel JSON als XML (Jersey | Documentation XML, sd; Jersey | Documentation JSON, sd)
2. Jersey heeft geen eigen methodes om de database aan te spreken, wel kan gemakkelijk de Java methodes gebruikt worden om data op te halen en op te slaan.
3. Jersey heeft een Client API waarmee gegevens opgehaald kunnen worden van andere webservices. Hierin is het ook mogelijk om headers mee te geven en aan te passen. (Jersey | Documentation Client API, sd)
4. Met behulp van de Client API (zie punt 3) kan informatie uit verschillende systemen gehaald worden, vervolgens kunnen deze samengevoegd worden en met JSON of XML (zie punt 1) terug gestuurd worden naar de client.
5. Jersey kan data versturen en ontvangen in verschillende formaten (plain tekst, JSON en XML). Met behulp van een '@consumes' tag kan aangegeven worden wat voor formaat er verwacht wordt. (Jersey | Documentation @Consumes, sd)
6. Zie 2. voor database ondersteuning.
7. Net als bij vele andere talen, kan ook in Java object-georiënteerd geschreven worden.
8. Jersey heeft de mogelijkheid om op verschillende authenticatie toe te voegen, hierdoor wordt het mogelijk om meerdere gebruikers te ondersteunen in de connector (Jersey | Documentation, sd)
9. In Jersey is geen ondersteuning voor meerdere API versies. Om dit wel te kunnen is het mogelijk om op basis van headers of de url een andere methode aan te roepen.

3.3.3 Wat is het meest geschikte framework?

Toelichting

Om de vraag "Wat is het meest geschikte framework?" te kunnen beantwoorden moet aan de eisen uit hoofdstuk 3.3.1. gewichten gehangen worden en kunnen deze vervolgens naast de tabel uit hoofdstuk 3.3.2 (Welke frameworks zijn er?) gelegd worden.

Onderzoek

Omdat bepaalde eisen zwaarder wegen dan de overige eisen zijn er 'gewichten' gehangen aan de verschillende eisen. In het onderstaande tabel zijn de verschillende eisen voorzien van gewichten tussen de 1 (Laag) en de 3 (Hoog).

Eisen	10. JSON / XML	11. Opslaan van data	12. Instellen van headers	13. Terugsturen van data	14. Data ontvangen	15. Database ondersteuning	16. Uitbreidbaar	17. Meerdere gebruikers	18. Meerdere API versies	Totaal aantal punten
Gewicht	1	3	1	3	3	3	2	2	1	19

In de volgende tabel met frameworks, zijn de ✓, – en ✗ vervangen door de gewichten, hierbij wordt aan de – niet het volledige gewicht gekoppeld, maar de helft van het gewicht.

ASP.NET Web API	1	3	1	3	3	3	2	2	0,5	18,5
Slim Framework	1	3	1	3	3	3	2	0	0	16
Django REST Framework	1	3	1	3	3	3	2	2	1	19
Phalcon	0,5	3	0,5	3	3	3	2	0	0	15
Play framework	1	3	1	3	3	3	2	0	1	17
Jersey	1	3	1	3	3	3	2	2	0,5	18,5

Zoals te zien in de bovenstaande tabel, liggen de resultaten van de verschillende frameworks dicht bij elkaar. De belangrijkste eisen (3) aan het framework worden door alle frameworks ondersteund. Het verschil zit vooral in de eisen met de minste prioriteit (1). De frameworks die aan alle eisen voldoen (met eventueel een kleine aanpassing) zijn:

- ASP.NET Web API
- Django REST Framework
- Jersey

3.3.4 Conclusie

Na het samen voegen van de verschillende eisen (die voortkomen uit dit onderzoek of gesprekken met de opdrachtgever), zijn de volgende eisen opgesteld:

1. Ondersteuning voor JSON en XML.
2. Mogelijkheid tot opslaan van gegevens
3. Instellen van headers
4. De client moet informatie van verschillende systemen kunnen ontvangen
5. De connector data kunnen ontvangen van een client
6. De connector moet verbonden kunnen worden met een database
7. De connector moet uitbreidbaar zijn.
8. De universele connector moet om kunnen gaan met verschillende gebruikers.
9. De webservice moet data ontvangen en/of terug geven in verschillende versies.

Hoewel er geen kant-en-klaar pakket is gevonden, waarmee de universele connector opgezet kan worden, zijn er wel verschillende frameworks waarmee de connector zelf ontwikkeld kan worden. Hierdoor zijn 6 verschillende frameworks onderzocht op functionaliteiten en naast de lijst met eisen gehouden om te controleren of een framework ondersteuning biedt voor deze eisen.

Uit het bovenstaande onderzoek blijkt dat alle frameworks dicht bij elkaar liggen qua score en dat alle frameworks de belangrijkste eisen (met gewicht 3) ondersteunen. Als er gezocht wordt naar frameworks die alle functionaliteiten ondersteunen (eventueel met een kleine aanpassing) dan kunnen de volgende framework geselecteerd worden als mogelijke kandidaat:

- ASP.NET Web API
- Django REST Framework
- Jersey

4 Conclusie

Uit gesprekken met een aantal medewerkers van Info Support zijn 15 systemen gekomen die dagelijks gebruikt worden als informatievoorziening. Hieronder zijn systemen als nieuws en weer bronnen te vinden, maar zijn ook systemen te vinden die voor specifieke projecten gebruikt worden.

- Buienradar
- Flitsmeister
- Flitsservice
- Google agenda
- Google verkeersinformatie
- Kibana
- knowNow
- New Relic
- NOS
- Nu.nl
- Slack
- TFS Portal
- Tomtom routeinformatie
- Werkagenda
- Werk email

Hoewel het voor gebruikers mogelijk is om deze systemen te benaderen, is uit onderzoek gebleken dat het niet mogelijk is om dit voor elk systeem gegevens op te vragen door middel van een beschikbare webservice. Hierbij is informatie uit systemen die publiekelijk toegankelijk zijn, minder vaak beschikbaar.

Om de gebruiker toch de mogelijkheid te bieden informatie uit bovenstaande systemen te halen, is er een alternatief gezocht voor de systemen die geen API beschikbaar hebben. In de onderstaande tabel is een compleet overzicht te zien van alle systemen die onderzocht zijn met eventuele alternatieven.

Systeem	API beschikbaar
Buienradar	Nee, alternatief: OpenWeaterMap
Flitsmeister	Nee, alternatief: Geen
Flitsservice	Nee, alternatief: Geen
Google agenda	Ja
Google verkeersinformatie	Ja
Kibana	Nee, alternatief: Geen
knowNow	Ja
New Relic	Ja
NOS	Nee, alternatief: Nu.nl
Nu.nl	Ja
OpenWeaterMap	Ja
Slack	Ja
TFS Portal	Ja
Tomtom routeinformatie	Ja
Werkagenda	Ja
Werk e-mail	Ja

Na het samen voegen van de verschillende eisen (die voortkomen uit dit onderzoek of gesprekken met de opdrachtgever), zijn de volgende eisen opgesteld:

1. Ondersteuning voor JSON en XML.
2. Mogelijkheid tot opslaan van gegevens
3. Instellen van headers
4. De client moet informatie van verschillende systemen kunnen ontvangen
5. De connector data kunnen ontvangen van een client
6. De connector moet verbonden kunnen worden met een database
7. De connector moet uitbreidbaar zijn.
8. De universele connector moet om kunnen gaan met verschillende gebruikers.
9. De webservice moet data ontvangen en/of terug geven in verschillende versies.

Uit het bovenstaande onderzoek blijkt dat alle frameworks dicht bij elkaar liggen qua score en dat alle frameworks de belangrijkste eisen ondersteunen. Als er gezocht wordt naar frameworks die alle bovenstaande eisen ondersteunen (eventueel met een kleine aanpassing) dan kunnen de volgende framework geselecteerd worden als mogelijke kandidaat:

- ASP.NET Web API
- Django REST Framework
- Jersey

Om tussen één van deze frameworks te kiezen, zou een vervolgonderzoek gehouden kunnen worden, waarbij onderzoek gedaan wordt naar de mogelijke performance verschillen of extra eisen die door o.a. de opdrachtgever gesteld worden.

5 Literatuur

- Akka*. (sd). Opgeroepen op Januari 5, 2016, van <http://akka.io/>
- ASP.NET | SQL Server Database*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.asp.net/whitepapers/aspnet-data-access-content-map#sqlserver>
- ASP.NET | Web API | Authentication and Authorization*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.asp.net/web-api/overview/security/authentication-and-authorization-in-aspnet-web-api>
- ASP.NET | Web API | JSON and XML serialization*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.asp.net/web-api/overview/formats-and-model-binding/json-and-xml-serialization>
- Django | Databases*. (sd). Opgeroepen op Januari 7, 2016, van <https://docs.djangoproject.com/en/1.9/ref/databases/>
- Django REST Framework*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.django-rest-framework.org/api-guide/requests/>
- Django REST Framework | Responses*. (sd). Opgehaald van <http://www.django-rest-framework.org/api-guide/responses/>
- Django REST Framework | Authentication*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.django-rest-framework.org/api-guide/authentication>
- Django Rest Framework | Contributing*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.django-rest-framework.org/topics/contributing/>
- Django REST Framework | Renderers*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.django-rest-framework.org/api-guide/renderers/>
- Django REST Framework | Versioning*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.django-rest-framework.org/api-guide/versioning/>
- DocForge - Caching*. (sd). Opgeroepen op December 15, 2015, van http://docforge.com/wiki/Web_application/Caching
- Elastic Search API*. (sd). Opgeroepen op December 14, 2015, van <https://www.elastic.co/guide/en/elasticsearch/reference/current/search.html>
- Flitsservice RSS Feed*. (sd). Opgeroepen op December 16, 2015, van <http://www.flitsservice.nl/algemeen/rss.php>
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*.
- Google Calendar API v3*. (sd). Opgeroepen op December 15, 2015, van <https://developers.google.com/google-apps/calendar/v3/reference/>
- Google Calendar API v3 | Authorizing*. (sd). Opgeroepen op December 28, 2015, van <https://developers.google.com/google-apps/calendar/auth>
- Google Calendar API v3 | Events List*. (sd). Opgeroepen op December 15, 2015, van <https://developers.google.com/google-apps/calendar/v3/reference/events/list>
- Google Maps API | API Key*. (sd). Opgeroepen op December 28, 2015, van <https://developers.google.com/maps/documentation/directions/get-api-key>
- IBM | Using Internet data in Android applications*. (sd). Opgeroepen op Januari 4, 2016, van <http://www.ibm.com/developerworks/library/x-dataAndroid/>
- Jersey | Documentation*. (sd). Opgehaald van <https://jersey.java.net/documentation/1.19/jax-rs.html>

- Jersey | Documentation @Consumes.* (sd). Opgeroepen op Januari 5, 2016, van <https://jersey.java.net/documentation/latest/user-guide.html#d0e2167>
- Jersey | Documentation Client API.* (sd). Opgeroepen op Januari 5, 2016, van <https://jersey.java.net/documentation/latest/user-guide.html#d0e4606>
- Jersey | Documentation JSON.* (sd). Opgeroepen op Januari 5, 2016, van <https://jersey.java.net/documentation/1.19/json.html>
- Jersey | Documentation XML.* (sd). Opgehaald van <https://jersey.java.net/documentation/1.19/xml.html>
- Jersey | Team List.* (sd). Opgeroepen op Januari 5, 2016, van <https://jersey.java.net/team-list.html>
- knowNow API documentation.* (n.d.). Retrieved December 11, 2015, from <https://knownow.infosupport.com/api/help>
- Listing your server ID and metric data.* (sd). Opgeroepen op December 15, 2015, van NewRelic: <https://docs.newrelic.com/docs/apis/rest-api-v2/server-examples-v2/listing-your-server-id-metric-data-v2>
- Luracast | Restler Features.* (sd). Opgeroepen op December 29, 2015, van <http://www.luracast.com/products/restler/features>
- MSDN | HttpRequestMessage Class.* (sd). Opgeroepen op Januari 4, 2016, van [https://msdn.microsoft.com/en-us/library/system.net.http.httprequestmessage\(v=vs.118\).aspx](https://msdn.microsoft.com/en-us/library/system.net.http.httprequestmessage(v=vs.118).aspx)
- MSDN | HttpResponseMessage Class.* (sd). Opgeroepen op Januari 4, 2016, van [https://msdn.microsoft.com/en-us/library/system.net.http.httpresponsemessage\(v=vs.118\).aspx](https://msdn.microsoft.com/en-us/library/system.net.http.httpresponsemessage(v=vs.118).aspx)
- MSDN Blogs | Using namespaces to version web apis.* (sd). Opgeroepen op Januari 4, 2016, van <http://blogs.msdn.com/b/webdev/archive/2013/03/08/using-namespaces-to-version-web-apis.aspx>
- New Relic API Keys.* (sd). Opgeroepen op December 11, 2015, van <https://docs.newrelic.com/docs/apis/rest-api-v2/requirements/api-keys>
- OpenWeatherMap Prices.* (sd). Opgeroepen op December 15, 2015, van <http://openweathermap.org/price>
- Phalcon | DB.* (sd). Opgeroepen op Januari 5, 2016, van <https://docs.phalconphp.com/en/latest/reference/db.html#connecting-to-databases>
- Phalcon | HTTP Request.* (sd). Opgeroepen op Januari 5, 2016, van https://docs.phalconphp.com/en/latest/api/Phalcon_Http_Request.html
- Phalcon | Http Response.* (sd). Opgeroepen op Januari 5, 2016, van https://docs.phalconphp.com/en/latest/api/Phalcon_Http_Response.html
- Phalcon | Team.* (sd). Opgeroepen op Januari 5, 2016, van <https://phalconphp.com/en/team>
- PHP | Install.* (sd). Opgeroepen op Januari 5, 2016, van <http://php.net/manual/en/install.php>
- PHP | PDO Drivers.* (sd). Opgeroepen op Januari 4, 2016, van <http://php.net/manual/en/pdo.drivers.php>
- PHP OAuth.* (sd). Opgeroepen op December 29, 2015, van <http://php.net/manual/en/book.oauth.php>
- Play Framework | Code.* (sd). Opgeroepen op Januari 5, 2016, van <https://www.playframework.com/code>

- Play framework | Database.* (2016, January 5). Opgehaald van <https://www.playframework.com/documentation/2.0.4/ScalaDatabase>
- Play framework | JSON.* (sd). Opgeroepen op Januari 5, 2016, van <https://www.playframework.com/documentation/2.2.x/ScalaJson>
- Play framework | Routing.* (sd). Opgeroepen op Januari 5, 2016, van <https://www.playframework.com/documentation/2.1.1/ScalaRouting>
- Play framework | WS.* (sd). Opgeroepen op Januari 5, 2016, van <https://www.playframework.com/documentation/2.3.x/ScalaWS>
- Play framework | XML.* (sd). Opgeroepen op Januari 5, 2016, van <https://www.playframework.com/documentation/2.2.x/ScalaXmlRequests>
- Python | Classes.* (sd). Opgeroepen op Januari 6, 2016, van <https://docs.python.org/2/tutorial/classes.html>
- Python | Downloads.* (sd). Opgeroepen op Januari 4, 2016, van www.python.org/downloads
- Scala | Download 2.12.* (sd). Opgeroepen op Januari 5, 2016, van <http://www.scala-lang.org/download/2.12.0-M3.html>
- Slack OAuth.* (sd). Opgeroepen op December 14, 2015, van <https://api.slack.com/docs/oauth>
- Slack Web API.* (sd). Opgeroepen op December 11, 2015, van <https://api.slack.com/web>
- Slim Framework.* (sd). Opgeroepen op Januari 4, 2016, van <http://www.slimframework.com/>
- Slim Framework | Request.* (sd). Opgeroepen op Januari 4, 2016, van <http://www.slimframework.com/docs/objects/request.html>
- Slim Framework | Request.* (sd). Opgeroepen op Januari 4, 2016, van <http://www.slimframework.com/docs/objects/request.html#the-request-headers>
- Slim Framework | Response.* (sd). Opgeroepen op Januari 4, 2016, van <http://www.slimframework.com/docs/objects/response.html#the-response-headers>
- Slim Framework Response.* (sd). Opgeroepen op December 29, 2015, van <http://www.slimframework.com/docs/objects/response.html#the-response-headers>
- Spray.* (sd). Opgeroepen op Januari 5, 2016, van <http://spray.io/>
- Spray | Contributing.* (sd). Opgeroepen op Januari 5, 2016, van <http://spray.io/project-info/contributing>
- The ASP.NET Site | Web API.* (sd). Opgeroepen op Januari 4, 2016, van <http://www.asp.net/web-api>
- TomTom Developer | Support FAQ.* (sd). Opgeroepen op December 28, 2015, van <http://developer.tomtom.com/support/faq>
- Visual Studio Team Services REST API Reference.* (sd). Opgeroepen op December 11, 2015, van <https://www.visualstudio.com/en-us/integrate/api/overview>

6 Begrippenlijst

API	Een API definieert de toegang tot functionaliteiten van de backend. De frontend hoeft hierdoor niet de implementatie van de verschillende functionaliteiten te weten, maar hoeft alleen via de API de gegevens op te vragen.
Authenticatie	Authenticatie is het controleren of een gebruiker is wie hij zegt dat hij is. Dit kan gedaan worden met bijvoorbeeld wachtwoord (dat bij de gebruikersnaam van de gebruiker hoort)
Autorisatie	Autorisatie is de stap na authenticatie. Hierin wordt aangegeven of een gebruiker rechten heeft om een bepaalde actie uit te voeren (zoals het uitlezen van projecten of verwijderen van bestanden)
Backend	De backend is het deel van de applicatie dat niet zichtbaar is voor de gebruiker. De frontend kan de backend aanspreken via API's.
Build	Met een build wordt de laatste versie van de gemaakte code('s) gecompileerd (omgezet) naar software dat gebruikt of geïnstalleerd kan worden.
Content Delivery Network (CDN)	Een CDN is een netwerk van servers die verspreid staan over verschillende landen, waarop onder andere teksten, afbeeldingen en video's geplaatst kunnen worden. Hierdoor kan een gebruiker snel de data ophalen bij dichtbijgelegen server.
Integrated development environment (IDE)	Een IDE is een softwareprogramma waarmee een ontwikkelaar software kan ontwikkelen. Iedere IDE is meestal gericht op één specifieke taal zodat de editor gericht kan helpen bij het ontwikkelen in die taal. Enkele voorbeelden van een IDE zijn: Visual Studio, Eclipse, Netbeans, pyCharm en PhpStorm.
OAuth	OAuth is een standaard om de authenticatie van gebruikers te kunnen regelen.
REST	REST is een software-architectuurstijl voor het internet. REST wordt meestal gebruikt om te communiceren over HTTP in combinatie met GET, POST, PUT etc. (dit wordt ook gebruikt door webbrowsers om data op te vragen en te versturen).
RSS	RSS is een XML bestand dat meestal door nieuwssites en weblogs gebruikt wordt om gebruikers op de hoogte te houden van nieuwe content. Een RSS-feed kan uitgelezen worden met RSS-lezers, zoals Digg reader en Feedly.
Software design pattern	Pattern is precies zoals het vertaald kan worden, een patroon. Door te ontwikkelen met bepaalde patronen (zoals Singleton, Factory en Façade) kunnen veel voorkomende ontwikkelproblemen opgelost worden.

Team Foundation Server (TFS)	TFS is een server waar teams code kunnen delen, werk kunnen verdelen, testen (automatisch) kunnen laten uitvoeren en builds kunnen uitvoeren. In een IDE als Visual Studio of online kan verschillende onderdelen van TFS bekeken of gewijzigd worden.
Testrun	Een testrun is het ronde waarin alle gedefinieerde testen uitgevoerd worden voor een bepaald (gedeelte van een) project. Uit een testrun komt een rapport met daarin het aantal testen dat gelukt/mislukt zijn en welke testen gerepareerd moeten worden.
Universele Connector	Een connector is 'een systeem' waar verschillende andere systemen, met verschillende protocollen, op aangesloten kunnen worden. De connector zet dit om naar één API of protocol. Hierdoor hoeft de frontend niet allerlei protocollen te ondersteunen en hoeft bij een wijziging aan een extern systeem niet de frontend geüpdatet te worden.

7 Bijlage A – Interview

Aan 2 ontwikkelaars van Info Support zijn de volgende vragen gesteld om informatie voor deelvraag 1 te achterhalen:

1. Wat voor systemen gebruik je voorafgaand, tijdens of aan het einde van je werkzaamheden bij Info Support? (Denk hierbij aan werk gerelateerde informatie bronnen zoals een serverstatus, maar ook aan andere bronnen zoals weer, verkeer of nieuws)
2. Met welk doeleinde gebruik je de verschillende systemen? (Welke informatie haal jij hier uit?)
3. Waar zijn deze systemen te vinden? (Draait het als applicatie op je desktop, staat het op een lokale server of een andere website?)
4. Is de informatie uit dit systeem voor iedereen beschikbaar of zitten hier restricties op? (Gebruik je een Info Support account of heb je een apart account voor dit systeem?)

7.1 Interview met Marco Kuiper

Bij aankomst:

- Het weer, zodat ik kan kijken of ik een paraplu mee moet nemen uit de auto (vooral wanneer ik weer naar huis ga). Dit is gewoon online, op een willekeurige website met weersverwachtingen.
- De buildstatus, zodat ik kan kijken of ik direct met m'n werkzaamheden kan beginnen, of dat eerst de build gerepareerd moet worden. Dit systeem draait in RHEA op TFS, waarbij ik ingelogd moet zijn om het te bekijken met mijn RHEA account.
- M'n agenda, zodat ik weet welke afspraken ik vandaag kan verwachten. Dit zit deels in mijn werk agenda (<https://mobile.infosupport.com/owa/>), en in Google Calendar (<https://calendar.google.com/>). Voor de eerste heb ik mijn RHEA account nodig, voor de tweede mijn persoonlijke Google account.

Bij vertrek:

- Files, zodat ik kan kijken of ik om moet rijden en/of ik later thuis ben (en hoe laat). Ik gebruik hier TomTom (<http://routes.tomtom.nl/>) of Google Maps (<https://www.google.nl/maps>) voor. Systemen zijn zonder inloggen te benaderen.
- Flitsinformatie, zodat ik weet of er op mijn weg flitsers gesignaleerd zijn. Ik gebruik hier Flitsservice voor (<http://m.flitsservice.nl/>), en tijdens het rijden Flitsmeister (<https://www.flitsmeister.nl/>). Systemen zijn zonder inloggen te benaderen.

7.2 Interview met Thomas Bruggink

- **TFS Portal en/of Slack**
In de TFS portal staat of de automatische builds (die 's nachts om 3 uur draaien) gelukt / mislukt zijn.
Daarnaast wordt deze informatie ook (automatisch) op Slack gepost.
- **Test Management (In TFS)**
Elke ochtend bekijken we of alle testen (350 / 400) afgelopen nacht zijn geslaagd of errors geven.
- **New Relic**
Wat is de status van onze servers? Draaien ze nog of is er iets gestopt?
- **Kibana**
Hier komen foutmeldingen uit verschillende bronnen binnen, en worden ook andere meldingen gelogd. Interessant om te weten is hoeveel logs/errors er de vorige dag(en) waren. Kibana is alleen grafisch interface, alle informatie wordt opgeslagen in ElasticSearch.
Normaal gesproken is deze website vrij beschikbaar, alleen
- **NOS**
Bekijken van nieuws op de website van de NOS.
Wat voor nieuws?
Eigenlijk van alles, behalve sport.
- **KnowNow**
Wat ik interessant zou vinden om te zien:
Zijn er nieuwe artikelen op knowNow geplaatst? Zijn er updates in de communities waar ik lid van ben?

7.3 Interview met Willem Meints

- **Outlook**
Wordt gebruikt om e-mails te lezen en te bekijken of er afspraken zijn (agenda)
- **TFS**
In TFS staan de builds en testen
- **Nu.nl**
Nu.nl gebruik ik om het nieuws te bekijken
- **Buienradar**
Af en toe kijk ik op buienradar.nl (tegenwoordig niet zo heel vaak meer)

Architectuur

Info Support Butler



Architectuur

Titel	Architectuur
Project/Onderwerp	Info Support Butler
Versie	1.0
Status	Definitief
Datum	21-03-2016
Bestand	Architectuur
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	28-11-2016	Maikel Stuivenberg	Creatie
0.2	Concept	11-02-2016	Maikel Stuivenberg	- Schermen -> Instellingen aangepast
0.3	Concept	23-02-2016	Maikel Stuivenberg	- Sprint 4 -> Extra informatie over authenticatie bij extern systeem
1.0	Definitief	21-03-2016	Maikel Stuivenberg	Document definitief

Distributie

Versie	Status	Datum	Aan
0.1	Concept	29-11-2016	John Gorter
0.1	Concept	02-02-2016	Marco Kuiper
0.2	Concept	11-02-2016	De Haagse Hogeschool
0.3	Concept	29-02-2016	Marco Kuiper, De Haagse Hogeschool (TTA)
1.0	Definitief	21-03-2016	De Haagse Hogeschool

Referenties

Code	Bron

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

Inhoudsopgave

INHOUDSOPGAVE	97
1 INFRASTRUCTUUR	98
1.1 Overzicht.....	98
1.2 Onderdelen.....	98
1.2.1 Beacons	99
1.2.2 Mobiele applicatie.....	99
1.2.3 ASP.NET Webserver.....	99
1.2.4 Externe webservices	99
2 SOFTWARE	100
2.1 Overzicht.....	100
2.2 Mobiele applicatie.....	100
2.2.1 Bluetooth	100
2.2.2 Notificatie.....	101
2.2.3 UI	101
2.2.4 Connector	101
2.3 Backend (Universele connector)	101
2.3.1 Web API.....	101
2.3.2 Authentication	101
2.3.3 Systems.....	101
2.3.4 Database.....	102
3 SCHERMEN	103
3.1 Opstarten applicatie.....	103
3.1.1 Inloggen	103
3.1.2 Registreren.....	104
3.2 Instellingen	104
3.2.1 Uitzonderingen.....	105
3.3 Overzichtsscherm	106
3.3.1 Binnenkomst of verlaten Info Support.....	106

1 Infrastructuur

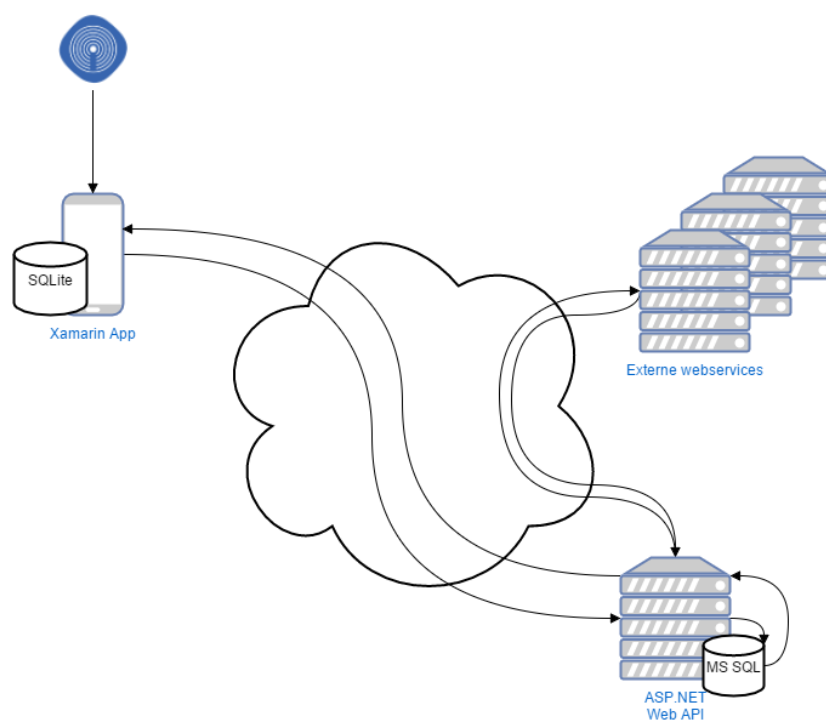
De infrastructuur van de Butler applicatie bestaat uit 4 onderdelen:

- Beacons;
- Xamarin applicatie;
- ASP.NET Web API Webserver;
- Externe webservices.

Hierbij worden de applicatie, webserver en webservices met elkaar verbonden door middel van het internet.

Hieronder staat in een overzicht uitgelegd hoe de infrastructuur van de Butler applicatie in elkaar zit. Daarnaast wordt er per onderdeel ook beschreven wat het is en hoe dit een relatie heeft met andere onderdelen uit de infrastructuur.

1.1 Overzicht



1.2 Onderdelen

Elk onderdeel heeft zijn eigen eigenschappen en kan op verschillende manier verbonden zijn met andere onderdelen van de infrastructuur.

Zoals in het bovenstaande overzicht te zien is, lopen (bijna) alle verbindingen door een cloud. Via het internet worden deze apparaten verbonden met elkaar, waarbij ze door middel van verschillende protocollen (zie ook de onderstaande hoofdstukken) met elkaar zullen communiceren.

1.2.1 Beacons

Beacons (ook wel iBeacons genoemd) zijn BLE (Bluetooth Low Energie) apparaten die via bluetooth continue een ID versturen. Aan de hand van deze ID kunnen de beacons herkend worden door verschillende mobiele apparaten.

Het signaal van een beacon gaat één richting op, doordat er alleen signaal verzonden wordt vanaf de beacon. Het mobiele apparaat hoeft geen signaal te versturen, maar alleen te luisteren naar signalen van andere apparaten.

Aan de hand van een beacon moet de applicatie kunnen herkennen of een gebruiker naar binnen of naar buiten gaat. Om dit te realiseren worden er 2 beacons opgehangen bij Info Support. Eén aan de binnenkant van de ingang en één aan de buitenkant van de ingang. Door beide beacons te herkennen kan bepaald worden of een gebruiker binnenkomt of het gebouw verlaat.

1.2.2 Mobiele applicatie

De mobiele applicatie is een applicatie gebouwd in Xamarin dat geschikt is voor zowel Android als iOS. De applicatie moet beacons (zie hoofdstuk 1.2.1) kunnen herkennen en moet in verbinding staan met het internet om gegevens van de webserver (zie hoofdstuk 1.2.3) te kunnen ontvangen.

Naast de applicatie zal op het mobiele apparaat een SQLite database geïnstalleerd worden, waarin gegevens over de gebruiker of gekoppelde systemen lokaal bijgehouden kunnen worden.

1.2.3 ASP.NET Webserver

De webserver (ook wel de universele connector genoemd) zal worden ontwikkeld met ASP.NET Web API. Deze connector zal als geheel fungeren tussen alle externe systemen en de applicaties bij de verschillende gebruikers.

De connector zal door middel van een REST service verbonden worden met de mobiele applicatie. Hierbij kan de applicatie een aanroep doen naar de webserver om aan te geven dat het bijvoorbeeld een nieuwe gebruiker wilt creëren (POST), gegevens wilt ophalen (GET) of een koppeling wilt verwijderen (DELETE).

Het ophalen van gegevens bij de verschillende webservices (zie ook hoofdstuk 1.2.4) kan met verschillende protocollen, afhankelijk van de webservice, afgehandeld worden door de webserver.

1.2.4 Externe webservices

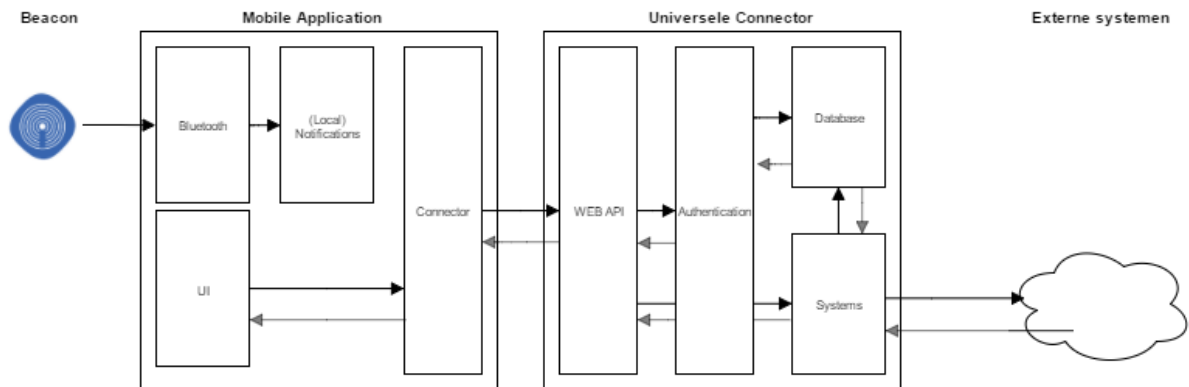
De externe webservices zijn bestaande systemen die verbonden zullen worden met de universele connector. Hierbij kan de universele connector, op moment dat de mobiele applicatie hierom vraagt, data ophalen bij een extern system. Voorbeelden van mogelijke systemen zijn: Nu.nl, OpenWeatherMap en TomTom.

Doordat de externe webservices niet in eigen beheer zijn, is het niet mogelijk om wijzigingen aan te brengen in deze webservices. Hierdoor kunnen de ondersteunende protocollen van elk systeem anders zijn. Meer over deze systemen (welke systemen, formaten en overige protocollen) zijn terug te vinden het onderzoeksrapport.

2 Software

Voor het Butler project moeten er 2 applicaties ontwikkeld worden die met elkaar en met externe webservices kunnen communiceren. Daarnaast moet het voor de gebruiker mogelijk worden om zelf te kiezen van welke systemen hij/zij informatie wilt ontvangen.

2.1 Overzicht



Zoals in de bovenstaande afbeelding te zien is, ligt de focus in dit project op de mobiele applicatie en de universele connector. Deze worden op bestaande onderdelen (beacons en externe systemen) aangesloten.

2.2 Mobiele applicatie

De mobiele applicatie bestaat, zoals te zien in het overzicht, uit 4 modules die gebouwd gaan worden:

- Bluetooth
- Notificatie
- UI
- Connector

2.2.1 Bluetooth

De bluetooth module ontvangt Bluetooth signalen en controleert of dit een beacon van Info Support is. Daarnaast kan met de Bluetooth module gekeken worden of een gebruiker naar binnen of buiten gaat. Dit wordt gedaan aan de hand van de volgorde van 2 beacons.

Bij het binnenkomen van het gebouw van Info Support, zal de gebruiker eerst buiten langs een beacon lopen en vervolgens binnen een beacon tegenkomen. Hierdoor weet de Bluetooth module dat de gebruiker naar binnen gaat. Bij het verlaten van het pand is de volgorde precies andersom. De applicatie zal eerst de beacon binnen herkennen en vervolgens de beacon buiten tegenkomen.

Bij beide acties geeft de bluetooth module door aan de Notificatie module dat er actie moet gebeuren (event handler).

2.2.2 **Notificatie**

De notificatie module ontvangt events van de bluetooth module en zorgt ervoor dat de gebruiker een notificaties te zien krijgt.

2.2.3 **UI**

De UI module zorgt voor het weergeven van schermen, zoals uitgelegd in hoofdstuk 3. Deze module staat in contact met de connector module, waarmee de benodigde data opgevraagd kan worden.

2.2.4 **Connector**

De connector module is het gedeelte dat data ophaalt uit de Universele Connector en zorgt dat de UI de juiste gegevens krijgt, die getoond kunnen worden aan de gebruiker.

Het ophalen van data gebeurt door middel van een webservice dat beschikbaar gesteld wordt door de universele connector. Met REST calls kunnen gebruikers worden aangemaakt, systemen worden gekoppeld en gebruikersinformatie worden opgehaald.

2.3 **Backend (Universele connector)**

2.3.1 **Web API**

De Web API is de communicatie laag tussen de systemen en de client (mobiele applicatie). Hierin wordt data uit de systeem en authenticatie modules omgezet naar een JSON formaat, zodat dit met een REST service teruggestuurd kan worden naar een gebruiker.

Aan de hand van een request van een gebruiker bepaald een Web API aan welk systeem hij wat moet vragen en bepaald de Web API ook wat er teruggestuurd moet worden. Bij het terugsturen van gegevens is het ook mogelijk dat de Web API een statuscode meestuurt, zodat de mobiele applicatie weet of een request juist is (200), geen juiste authenticatie bevat (403) of een niet bestaande API probeert aan te roepen (404).

2.3.2 **Authentication**

De authenticatie module handelt de gebruikerslogin en -registratie af en geeft aan de Web API terug of een authenticatie poging correct is.

De Authenticatie module kan de gegevens van een gebruiker opslaan en controleren door middel van de verbinding met de database module. Met deze module kunnen opgeslagen gebruikers opgevraagd en nieuwe gegevens doorgegeven worden.

2.3.3 **Systems**

De 'systems' module is de belangrijkste schakel in de universele connector en zorgt voor de communicatie tussen de Web API en de externe webservices. Hierin zit alle logica vermeld om verschillende soorten data op te halen (zoals json, xml en html) en kan data converteren naar een formaat dat terug gestuurd kan worden met de Web API module.

Hiernaast heeft de systeem module ook de mogelijkheid om met de database te communiceren via de database module. Hieruit kan bijvoorbeeld informatie opgehaald worden die benodigd zijn voor een systeem (zoals een API key)

2.3.4 Database

De database module zorgt voor het contact met de database. Hierin staan gegevens vermeld over gebruikers, systemen en de koppeling tussen beide onderdelen.

Naast het bewaren van de gegevens is de database module ook verantwoordelijk voor het correct teruggeven van gegevens aan de authenticatie module (bijvoorbeeld gebruikersgegevens) en het geven van gegevens aan de systeem module (bijvoorbeeld: welke systemen zijn beschikbaar en actief en zijn gekoppeld met een gebruiker)

3 Schermen

3.1 Opstarten applicatie

Bij het opstarten van de applicatie wordt, als de gebruiker niet eerder ingelogd is, het onderstaande scherm getoond. Hierbij krijgt de gebruiker de mogelijkheid om een gebruikersnaam in te voeren en te kiezen voor Inloggen of Registreren. Beide knoppen hebben hierbij een eigen acties, die hieronder beschreven staan:



Zoals in de bovenstaande afbeelding te zien is, is voor het inloggen geen wachtwoord vereist. Dit is in overleg met de opdrachtgever besloten, omdat het bij het Butler project gaat om een Proof of Concept.

3.1.1 Inloggen

De inlog knop voert een controle op uit een correcte gebruikersnaam, door een request te sturen naar de webservice. Indien de gebruikersnaam niet bekend is bij de webserver, zal het onderstaande scherm getoond worden.



Bij het een correcte login wordt de gebruiker doorgestuurd, naar het beacon scherm (zie 3.3)

3.1.2 Registreren

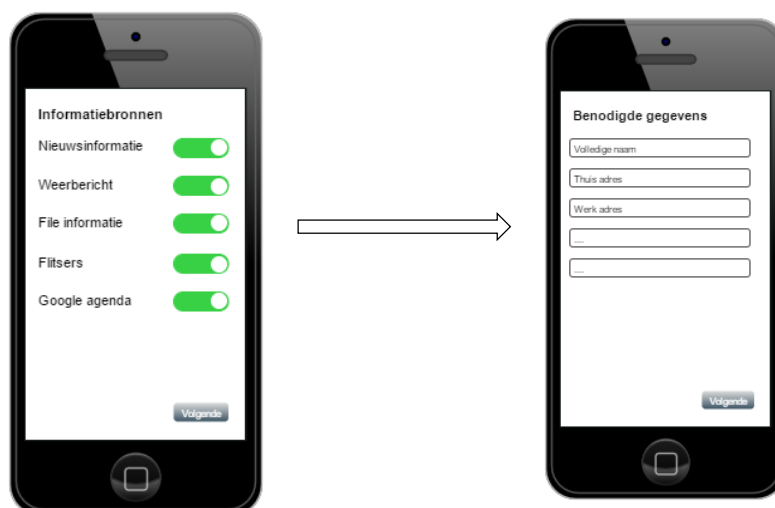
De knop registreren voert twee controles uit op de gegevens gebruikersnaam. Er wordt gecontroleerd of er een gebruikersnaam is opgegeven (lokaal) en of deze nog niet bezet is (via de webservice).



Na het succesvol klikken op de registreer knop wordt de gebruiker doorgestuurd naar het instellingen scherm (zie 3.2).

3.2 Instellingen

Zodra de gebruiker in het instelling scherm komt, dit kan na het registreren (zie 3.1.2) zijn of vanaf de beacon pagina, krijgt de gebruiker de volgende schermen te zien:



Afhankelijk van de beschikbare systemen, kan de gebruiker door een lijst met systemen scrollen en ze koppelen aan een actie (vertrek of binnenkomst). Na een aantal systemen gekozen te hebben en doorgaat naar het volgende scherm, krijgt de gebruiker een scherm te zien waarin gevraagd wordt om extra gegevens op te geven.

Het aantal gegevens dat benodigd zijn, is afhankelijk van de gekozen systemen. Indien de gebruiker alleen het systeem 'Nieuwsinformatie' heeft aanstaan, hoeft de gebruiker geen extra gegevens op te geven. In geval van 'File informatie', moet de gebruiker opgeven waar hij woont, zodat op de achtergrond gekeken kan worden wat de route (+file) van de gebruiker is.

Systemen als 'Google Agenda' en 'Slack' vereisen dat gebruiker toestemming geeft van het systeem dat gekoppeld gaat worden. Hiervoor is een extra scherm vereist waarin om deze toestemming wordt gevraagd. Na het activeren van een systeem dat authenticatie vereist, wordt gelijk een scherm getoond van het desbetreffende systeem.

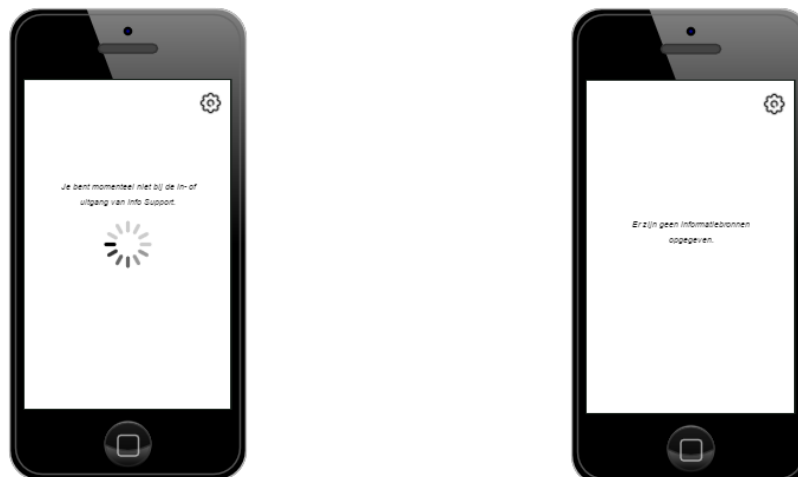
3.2.1 Uitzonderingen

Indien de gebruiker geen toestemming geeft voor het benaderen van zijn gegevens (via een extern systeem) of niet alle benodigde gegevens (correct) invult, is het niet mogelijk om gegevens op te halen uit deze systemen.

Hierdoor zal bij het binnenkomen/verlaten van Info Support geen informatie getoond worden uit dit systeem in het overzichtsscherm (zie 3.3).

3.3 Overzichtsscherm

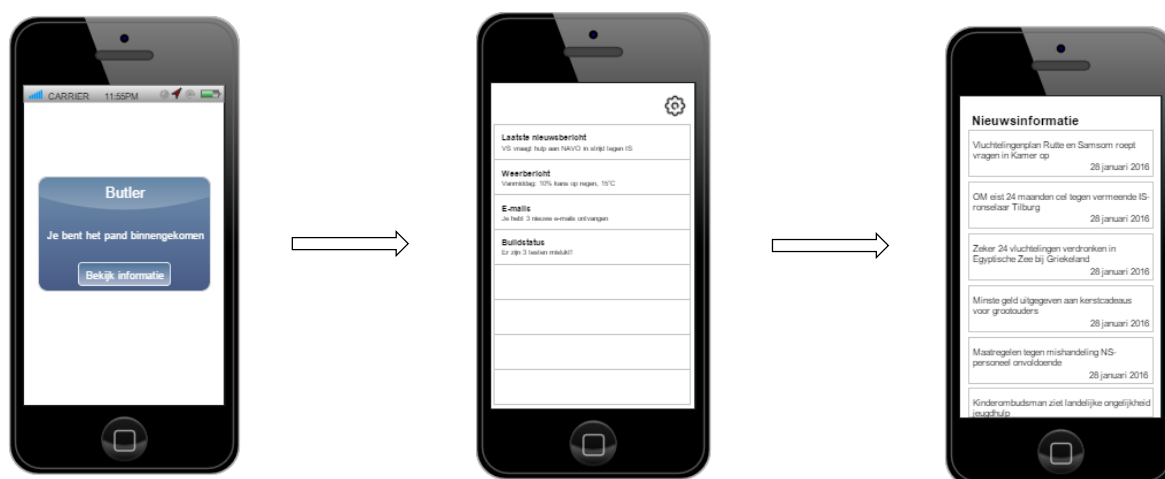
Het overzichtsscherm wordt getoond na het inloggen of registreren van een gebruiker. Het tonen van gegevens is alleen mogelijk als de gebruiker systemen heeft geselecteerd en bij het pand van Info Support naar binnen of buiten gaat.



Zoals te zien op beide afbeeldingen, is het vanuit beide schermen mogelijk om rechtsboven op een 'Instellingen' icoon te klikken. Hiermee wordt de gebruiker naar de instellingen pagina gestuurd (zie 3.2) waar hij/zij de gekozen systemen en gegevens weer kan aanpassen.

3.3.1 Binnenkomst of verlaten Info Support

Zodra de gebruiker het pand van Info Support binnenkomt of verlaat, zal de applicatie een notificatie tonen. Bij het klikken op de notificatie wordt een overzicht gegeven met alle systemen die gekoppeld zijn, met daarbij een beknopt overzicht van de gegevens (bijvoorbeeld het laatste nieuwsbericht, het aantal nieuwe e-mails of het aantal mislukte testen).



Na het klikken op een informatiebron (2e afbeelding) wordt aan de gebruiker gedetailleerde informatie getoond uit het systeem (3e afbeelding). Hierin kan de gebruiker bijvoorbeeld de laatste 10 nieuwsberichten zien, de titels van de nieuwe e-mails en welke testen mislukt zijn.

Requirements

Info Support Butler



Requirements

Titel	Requirements
Project/Onderwerp	Info Support Butler
Versie	1.0
Status	Definitief
Datum	21-03-2016
Bestand	Requirements
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	28-01-2016	Maikel Stuivenberg	Creatie
0.2	Concept	11-02-2016	Maikel Stuivenberg	- Req. FR06 aangepast.
0.3	Concept	19-02-2016	Maikel Stuivenberg	- FR03 aangepast
0.4	Concept	09-03-2016	Maikel Stuivenberg	- FR04 aangepast
1.0	Definitief	21-03-2016	Maikel Stuivenberg	Document definitief

Distributie

Versie	Status	Datum	Aan
0.1	Concept	29-01-2016	John Gorter
0.1	Concept	02-02-2016	Marco Kuiper
0.2	Concept	11-02-2016	De Haagse Hogeschool
0.3	Concept	29-02-2016	Marco Kuiper, De Haagse Hogeschool (TTA)
1.0	Definitief	21-03-2016	De Haagse Hogeschool

Referenties

Code	Bron

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

1 Requirements

1.1 Functioneel

ID	Requirement
FR01	De applicatie moet automatisch herkennen wanneer een gebruiker het pand binnenkomt of verlaat.
FR02	De gebruiker wilt informatie kunnen ontvangen van systemen.
FR03	De gebruiker moet zich kunnen identificeren door middel van een gebruikersnaam.
FR04	De gebruiker moet systemen kunnen koppelen aan een binnenkomst/verlaten actie.
FR05	De gebruiker moet gegevens van verschillende systemen in één overzicht zien.
FR06	De gebruiker moet (afhankelijk van de gekozen systemen) informatie kunnen opgeven zoals een woonadres of vertrektijd.
FR07	De gebruiker moet kunnen registreren, indien hij nog geen gebruikersnaam heeft.

1.2 Niet functioneel

ID	Requirement
NF01	De applicatie moet op de achtergrond de iBeacon herkennen.
NF02	De gebruiker moet (nadat hij is ingelogd) eerst systemen koppelen en daarna profielinformatie opgeven.
NF03	De gebruiker hoeft als account alleen een gebruikersnaam op te geven (wachtwoord is niet nodig in de PoC).
NF04	Na het herkennen van een iBeacon, moet de applicatie een notificatie weergeven.
NF05	Indien een systeem authenticatie vereist wordt na het koppelen een authenticatiescherm weergeven.

Functioneel ontwerp

Info Support Butler



Functioneel ontwerp

Titel	Functioneel ontwerp
Project/Onderwerp	Info Support Butler
Versie	1.0
Status	Definitief
Datum	21-03-2016
Bestand	Functioneel ontwerp
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	11-01-2016	Maikel Stuivenberg	Creatie
0.2	Concept	08-02-2016	Maikel Stuivenberg	- Use Cases bijgewerkt - "Systeem" vervangen door "informatiebron" - Profielinformatie toegevoegd
0.3	Concept	15-02-2016	Maikel Stuivenberg	- Use case en Globaal overzicht diagram opnieuw uitgewerkt
0.4	Concept	23-02-2016	Maikel Stuivenberg	Sprint 3: - Authencatie bij informatiebron koppelen
0.5	Concept	09-03-2016	Maikel Stuivenberg	Sprint 4: - Actie bij informatiebron koppelen
1.0	Definitief	21-03-2016	Maikel Stuivenberg	Document definitief

Distributie

Versie	Status	Datum	Aan
0.1	Concept	01-02-2016	Marco Kuiper
0.2	Concept	11-02-2016	De Haagse Hogeschool
0.2	Concept	12-02-2016	John Gorter
0.3	Concept	17-02-2016	Marco Kuiper
0.4	Concept	29-02-2016	Marco Kuiper, De Haagse Hogeschool (TTA)
1.0	Definitief	21-03-2016	De Haagse Hogeschool

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

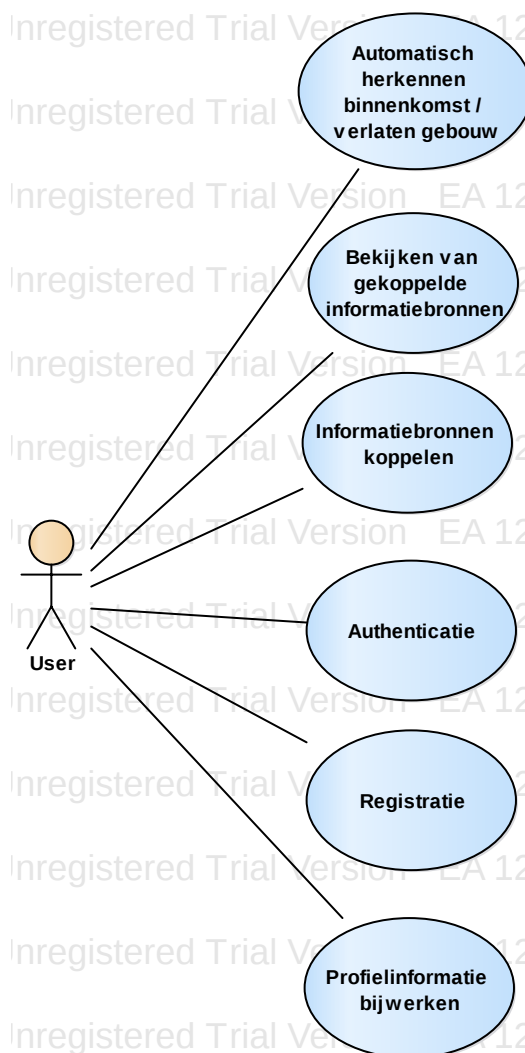
No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

Inhoudsopgave

1	USE CASE	115
1.1	Beschrijving.....	116
2	GLOBAL OVERZICHT.....	119
3	GEGEVENS OVERZICHT.....	120

1 Use case



1.1 Beschrijving

Use Case 'Automatisch herkennen binnenkomst / verlaten gebouw'	
ID	1
Primaire actor	Gebruiker
Secundaire actor	N.v.t
Requirement	FR01
Precondities	1. De gebruiker moet ingelogd zijn.
Main flow	1. De gebruiker komt binnen bij of verlaat het gebouw van Info Support. 2. Het systeem herkent de gebruiker 3. De gebruiker krijgt een melding van het systeem
Resultaat	De gebruiker heeft een notificatie waarmee hij de app kan starten.
Alternatieve flow	

Use Case 'Informatie bekijken van gekoppelde informatiebronnen'	
ID	2
Primaire actor	Gebruiker
Secundaire actor	N.v.t.
Requirement	FR02, FR05
Precondities	1. De gebruiker moet het pand binnenkomen of verlaten (Use Case 1). 2. De gebruiker moet ingelogd zijn.
Main flow	1. Het systeem geeft een overzicht van alle gekoppelde informatiebronnen. 2. De gebruiker kiest een informatiebron. 3. Het systeem geeft alle gegevens van de gekozen informatiebron.
Resultaat	Gebruiker kan van elk gekoppelde bron informatie bekijken.
Alternatieve flow	

Use Case 'Informatiebronnen koppelen'	
ID	3
Primaire actor	Gebruiker
Secundaire actor	N.v.t.
Requirement	FR04, NF05
Precondities	1. De gebruiker moet ingelogd zijn.
Main flow	1. De gebruiker kiest informatiebronnen die hij wilt koppelen aan een actie. (vertrek / binnenkomst) 2. Het systeem slaat deze bronnen op. 3. De gebruiker geeft extra benodigde profielinformatie op. 4. Het systeem slaat de extra informatie op
Resultaat	De gebruiker krijgt de nieuw gekozen systemen te zien bij het binnenkomen of verlaten van Info Support. (Use Case 2)
Alternatieve flow	Koppelen van systeem dat authenticatie vereist 1. De gebruiker kiest de informatiebronnen die hij die hij wilt koppelen aan een actie. (vertrek / binnenkomst) 2. Het systeem vraagt de gebruiker om zich te authenticeren. 3. Het systeem onthoudt de authenticatie. 4. Het systeem slaat de informatiebronnen op. 5. De gebruiker geeft extra benodigde profielinformatie op. 6. Het systeem slaat de extra informatie op.

Use Case 'Authenticatie'	
ID	4
Primaire actor	Gebruiker
Secundaire actor	N.v.t.
Requirement	FR03, NF03
Precondities	1. De gebruiker heeft een account.
Main flow	1. De gebruiker geeft zijn gebruikersnaam 2. Het systeem authenticiseert de gebruiker
Resultaat	De gebruiker krijgt voortaan een melding als hij binnenkomt (of weggaat) bij het pand van Info Support. (Use Case 1)
Alternatieve flow	Incorrect gebruikersnaam 1. De gebruiker geeft zijn gebruikersnaam 2. Het systeem controleert de gebruikersnaam -> Indien niet bestaand: Terug naar stap 1.

Use Case 'Registratie'	
ID	5
Primaire actor	Gebruiker
Secundaire actor	N.v.t.
Requirement	FR07
Precondities	1. De gebruiker heeft nog geen account. 2. De gebruiker is nog niet ingelogd.
Main flow	1. De gebruiker geeft een gebruikersnaam op. 2. Het systeem slaat de gebruikersnaam op 3. [Use Case 3]
Resultaat	De gebruiker heeft een account, is ingelogd in de applicatie en krijgt voortaan een notificatie als hij binnenkomt (of weggaat) bij het pand van Info Support.
Alternatieve flow	Gebruikersnaam al in gebruik 1. De gebruiker geeft een gebruikersnaam op. 2. Het systeem controleert de gebruikersnaam. -> Indien in gebruik: Terug naar stap 1.

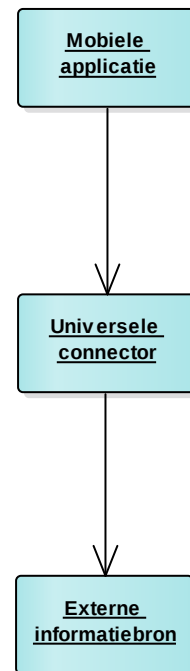
Use Case 'Profielinformatie bijwerken'	
ID	6
Primaire actor	Gebruiker
Secundaire actor	N.v.t.
Requirement	FR06
Precondities	1. De gebruiker is ingelogd
Main flow	1. De gebruiker wijzigt zijn gegevens 2. Het systeem controleert de gegevens
Resultaat	De gegevens van de gebruiker zijn geüpdatet en krijgt voortaan informatie gebaseerd op deze gegevens
Alternatieve flow	

2 Globaal overzicht

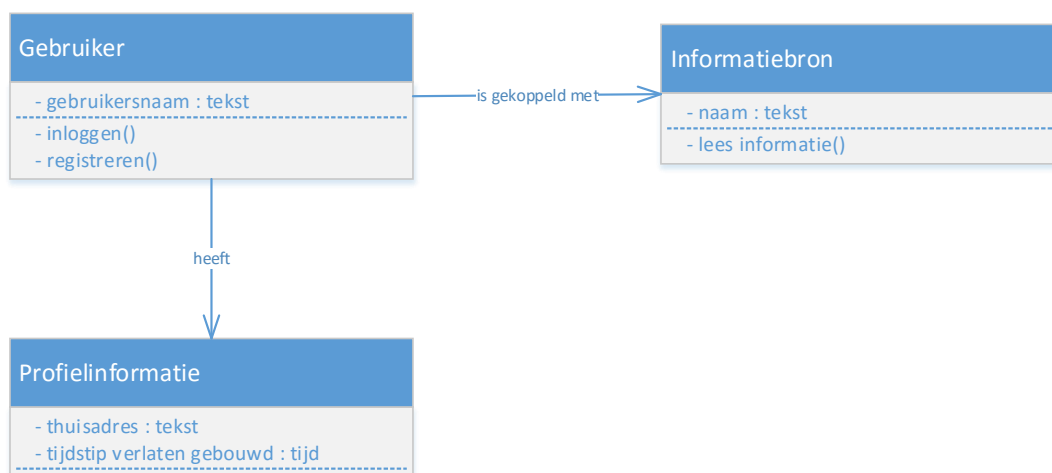
Om alle use cases (zie hoofdstuk 1) te bouwen, worden er twee onderdelen ontwikkeld:

- De mobiele applicatie
Zorgt dat de gebruiker systemen kan koppelen en gegevens uit de systemen kan lezen
- Een universele connector
De applicatie haalt zijn data uit een universele connector die de gegevens uit externe systemen gaat halen.

De externe informatiebronnen zijn bestaande systemen als Google Maps, Nu.nl en OpenWeatherMap, waaruit informatie opgehaald kan worden.



3 Gegevensoverzicht



Technisch ontwerp

Info Support Butler



Technisch ontwerp

Titel	Technisch ontwerp
Project/Onderwerp	Info Support Butler
Versie	1.0
Status	Definitief
Datum	21-03-2016
Bestand	Technisch ontwerp
Bedrijf	Info Support bv

Historie

Versie	Status	Datum	Auteur	Verandering
0.1	Concept	11-01-2016	Maikel Stuivenberg	Creatie
0.2	Concept	08-02-2016	Maikel Stuivenberg	- Use cases (uit FO) vermeld - Universele connector opnieuw uitgewerkt
0.3	Concept	15-02-2016	Maikel Stuivenberg	- Mobiele applicatie opnieuw uitgewerkt
0.4	Concept	23-02-2016	Maikel Stuivenberg	Sprint 4 - Kleine beschrijving toegevoegd aan diagrammen - 2.1.2 sequence vervangen door activity diagram
0.5	Concept	09-03-2016	Maikel Stuivenberg	Sprint 5 - Actie aan 2.2.2 en 2.3 toegevoegd
1.0	Definitief	21-03-2016	Maikel Stuivenberg	Document definitief

Distributie

Versie	Status	Datum	Aan
0.1	Concept	02-02-2016	Marco Kuiper, John Gorter
0.2	Concept	12-02-2016	John Gorter, De Haagse Hogeschool
0.3	Concept	17-02-2016	Marco Kuiper
0.4	Concept	29-02-2016	Marco Kuiper, De Haagse Hogeschool (TTA)
1.0	Definitief	21-03-2016	De Haagse Hogeschool

© Info Support, Veenendaal 2014

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande toestemming van Info Support.

No part of this publication may be reproduced in any form by print, photo print, microfilm or any other means without written permission by Info Support.

Prijsopgaven en leveringen geschieden volgens de Algemene Voorwaarden van Info Support B.V., gedeponeerd bij de K.v.K. te Utrecht onder nr. 30135370. Een exemplaar zenden wij u op uw verzoek per omgaande kosteloos toe.

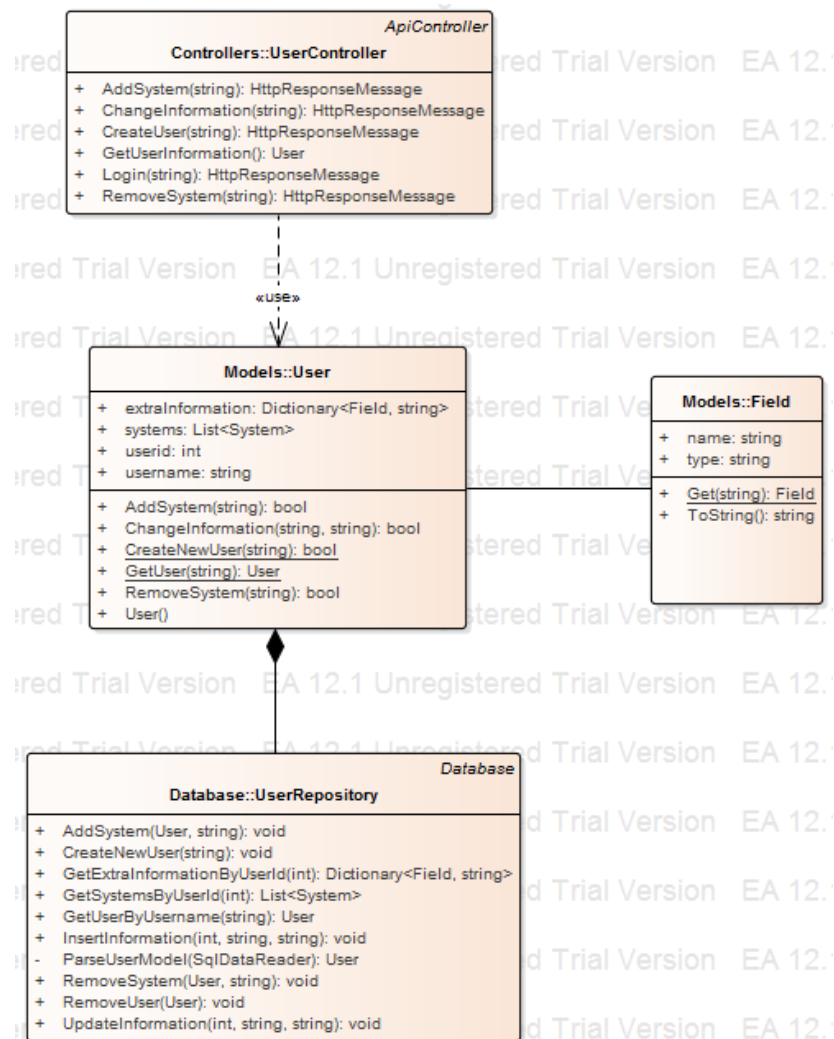
Inhoudsopgave

1	UNIVERSELE CONNECTOR	125
1.1	Gebruikers.....	125
1.1.1	Authenticatie	126
1.1.2	Wijzigen profielinformatie.....	126
1.2	Systemen.....	127
1.2.1	Informatie ophalen uit een systeem.....	128
1.3	Database.....	129
2	MOBIELE APPLICATIE	130
2.1	Bluetooth	130
2.1.1	Starten beacon herkenning.....	130
2.1.2	Herkennen van een beacon	131
2.2	Gebruikers.....	132
2.2.1	Registratie.....	133
2.2.2	Systemen koppelen	133
2.3	Informatiebronnen.....	134
2.3.1	Overzichtspagina.....	135
2.3.2	Enkele informatiebron bekijken	135

1 Universele connector

1.1 Gebruikers

Met de Universele Connector kunnen gebruikers worden geauthentiseerd en kunnen gegevens van gebruikers opgehaald, gewijzigd en aangemaakt worden. De API kan met behulp van de onderstaande classes deze use cases uitvoeren.

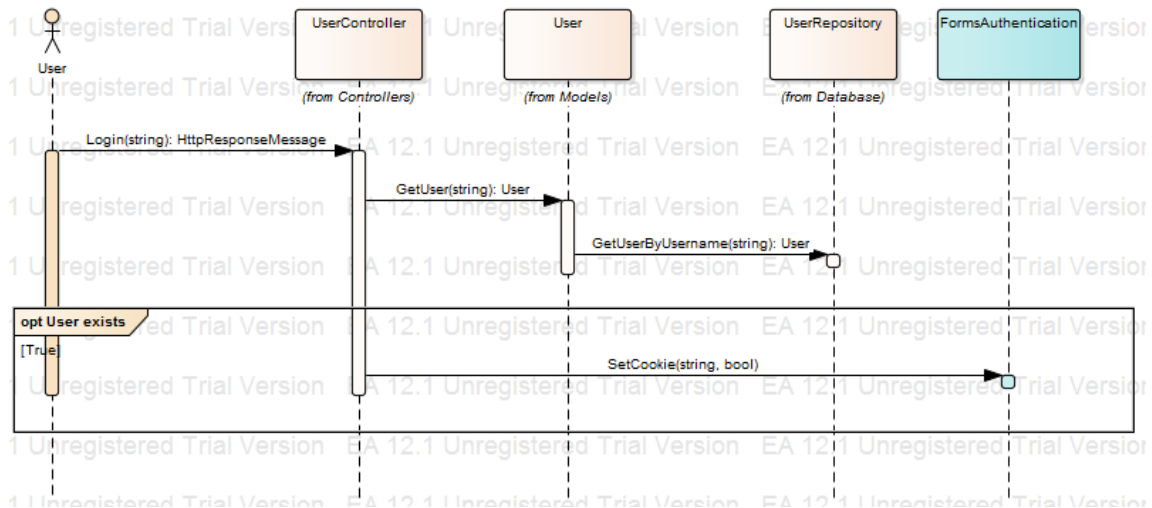


De code is zo opgezet dat het voor de controller eenvoudig is om een actie uit te voeren op een user. Een controller krijgt een user object en het user object communiceert met de database class.

In de volgende 2 sub-hoofdstukken zijn voorbeelden gegeven van de authenticatie en wijzigen van profielinformatie processen.

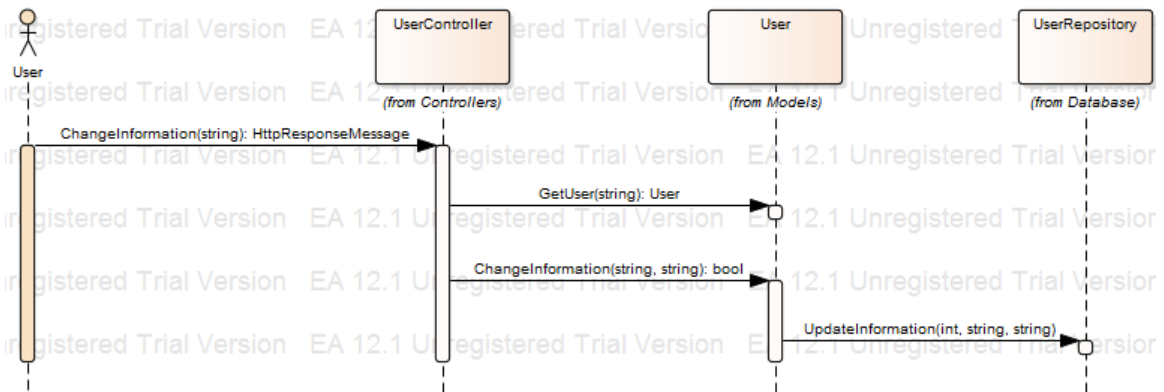
1.1.1 Authenticatie

Use case 4, het authenticeren van gebruikers, maakt gebruik van ASP.Net Forms authenticatie. Hierdoor kan er een cookie gezet worden als de gebruiker correct ingelogd is en kan deze cookie meegegeven worden bij elke nieuwe request.



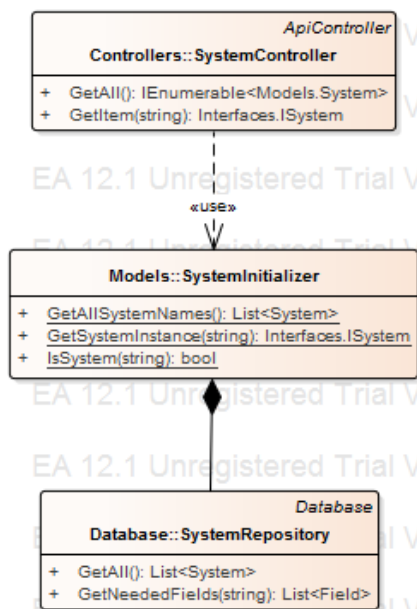
1.1.2 Wijzigen profielinformatie

Use case 6, het wijzigen van profielinformatie, is net zo eenvoudig opgezet als het Authenticeren van een gebruiker.

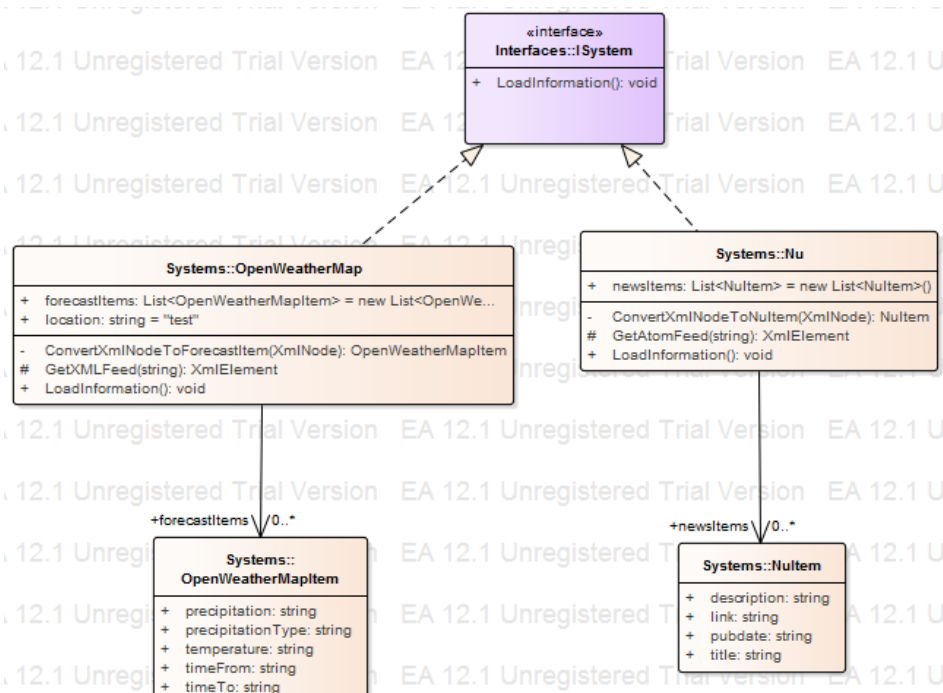


1.2 Systemen

De universele connector kan een lijst geven van alle beschikbare systemen (use case 3) en kan informatie geven uit de verschillende systemen (use case 2). Hiervoor is, zoals in het onderstaande diagram te zien, een SystemController opgezet dat via de SystemInitializer systemen kan ophalen.



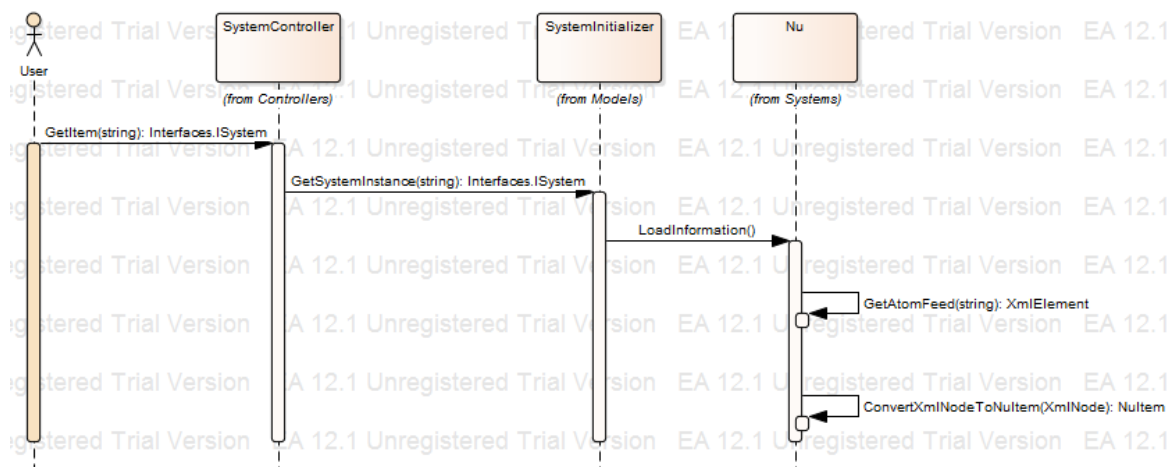
Met behulp van de SystemInitializer kan een instantie aangemaakt worden van een systeem dat het ISystem interface implementeert. In het onderstaande diagram zijn 2 voorbeeld systeem weergegeven.



1.2.1 Informatie ophalen uit een systeem

Het ophalen van informatie (use case 2) gaat via de SystemController. Deze controller laat de SystemInitializer een ISystem object aanmaken en vullen met informatie en geeft deze terug aan de controller.

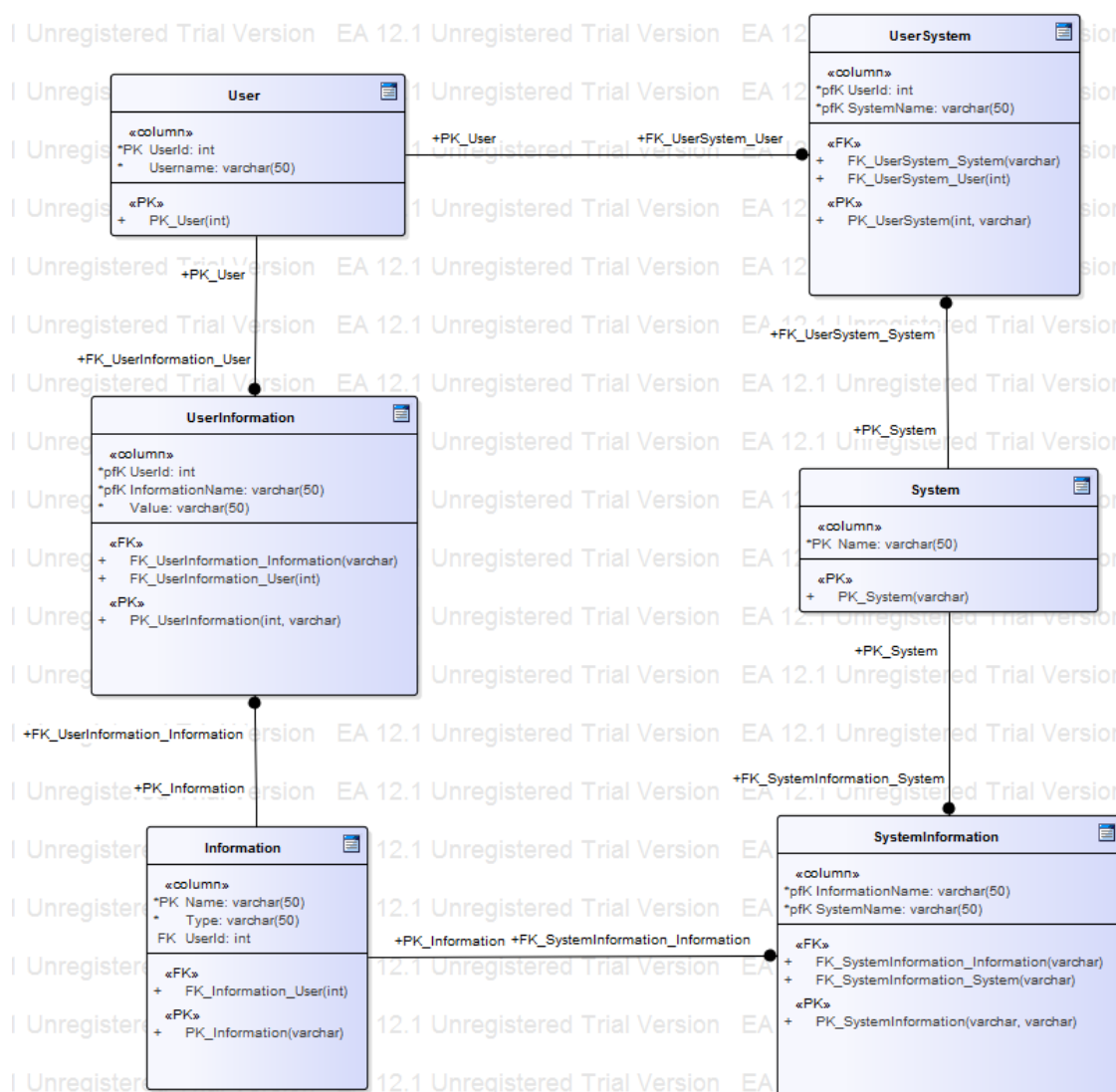
In het onderstaande diagram is het proces uitgewerkt voor Nu.nl, alle overige informatiebronnen zijn op vergelijkbare manieren uitgewerkt.



1.3 Database

Om gegevens van de gebruiker, systemen en benodigde profielinformatie op te slaan, worden de gegevens in de universele connector opgeslagen in een database (zie het Architectuur document voor meer informatie).

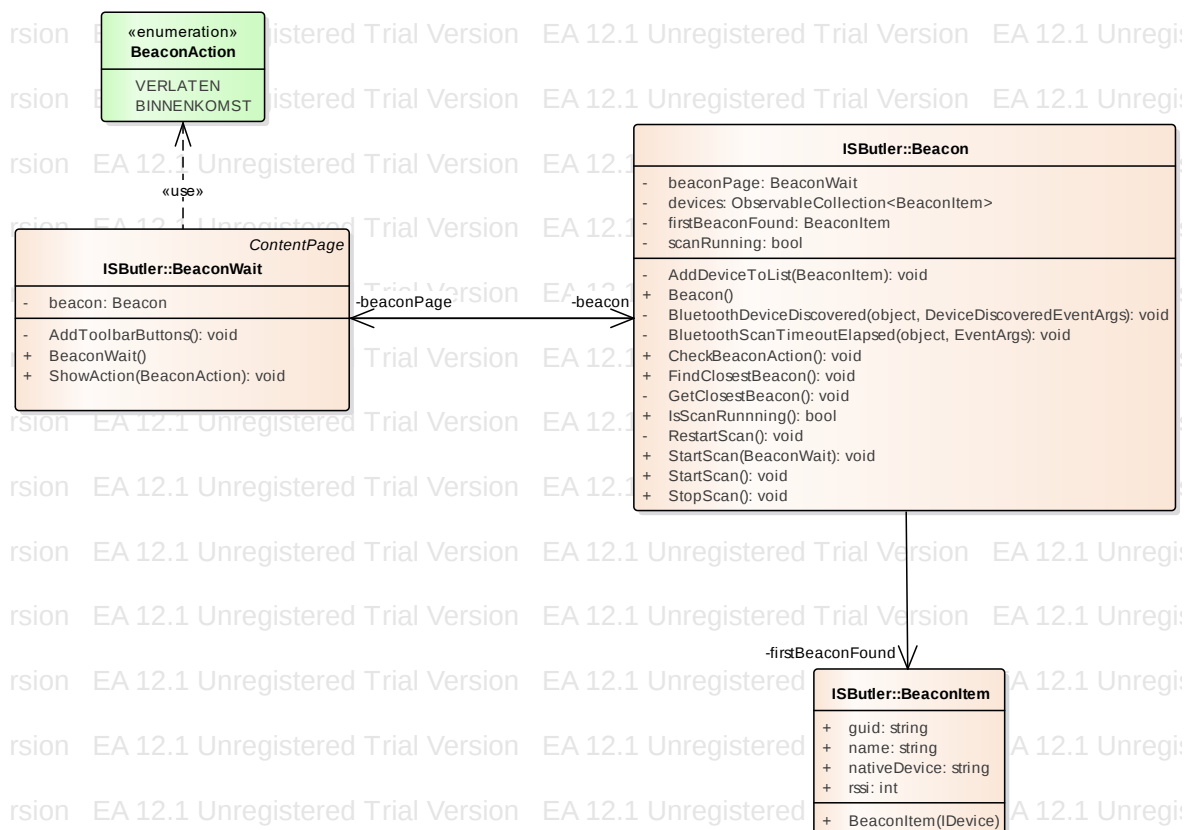
In het onderstaande diagram staan alle tabellen, relaties en constraints weergegeven.



2 Mobiele Applicatie

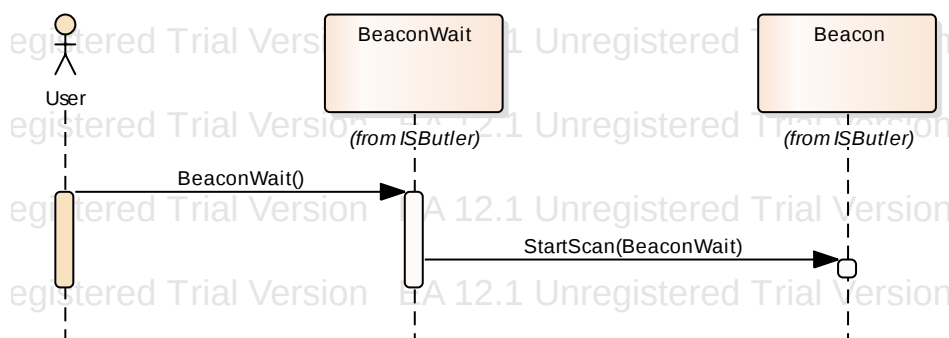
2.1 Bluetooth

Het bluetooth gedeelte van de mobiele applicatie maakt hoofdzakelijk gebruik van de volgende classes.



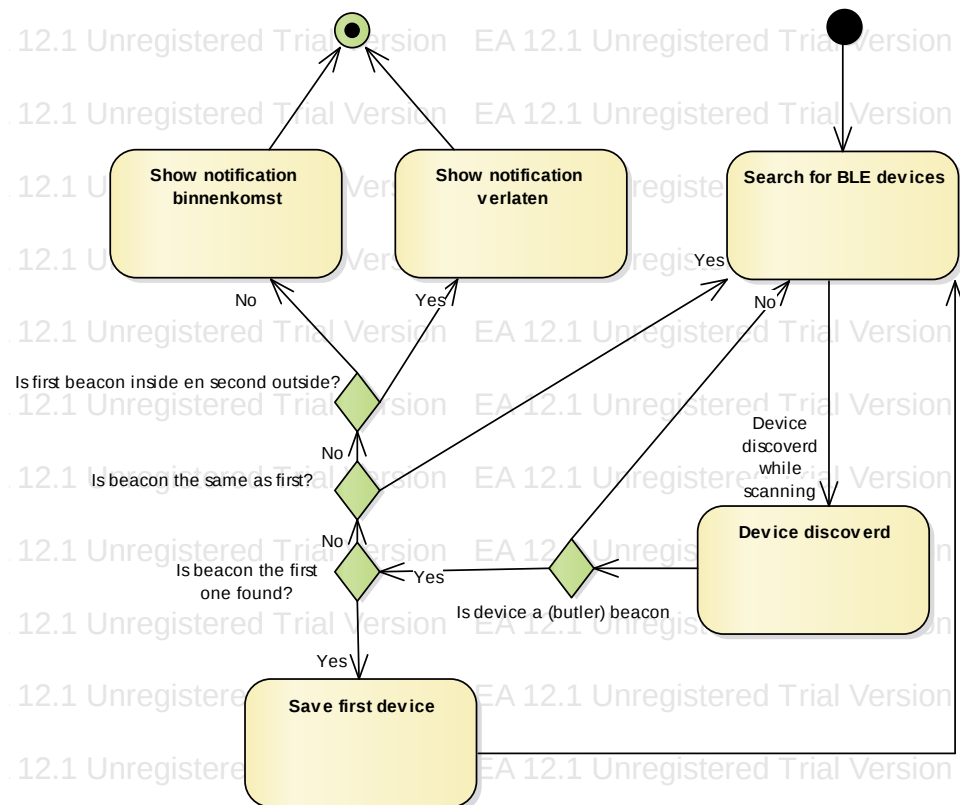
2.1.1 Starten beacon herkenning

Bij het opstarten van de applicatie wordt de **StartScan** methode van de beacon class aangeroepen. Indien deze methode hergebruikt



2.1.2 Herkennen van een beacon

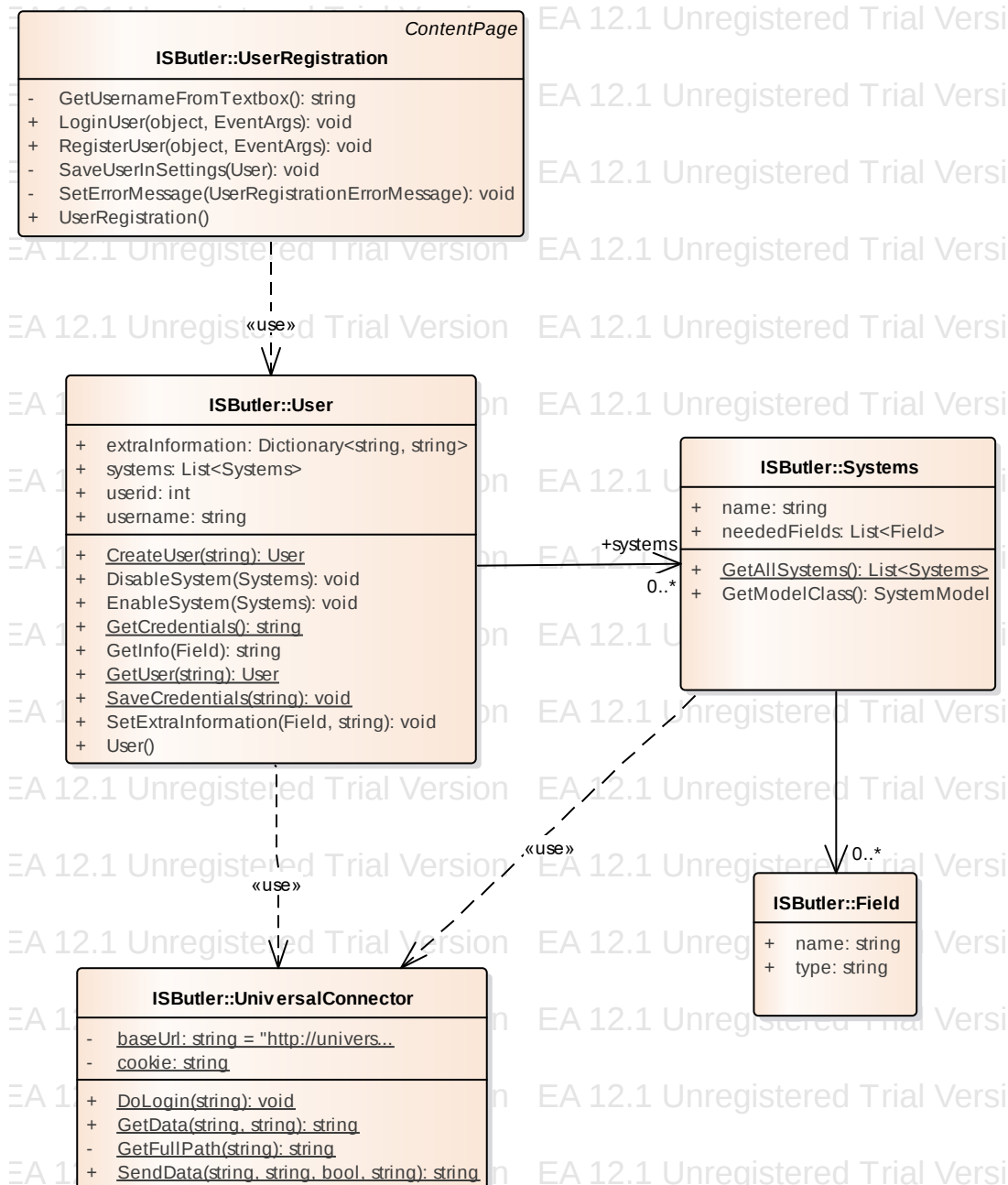
Tijdens het zoeken van beacons, kunnen verschillende devices gevonden. In de onderstaande activity diagram wordt getoond aan welke voorwaarden een bluetooth device moet doen om aan de gebruiker een notificatie te tonen.



2.2 Gebruikers

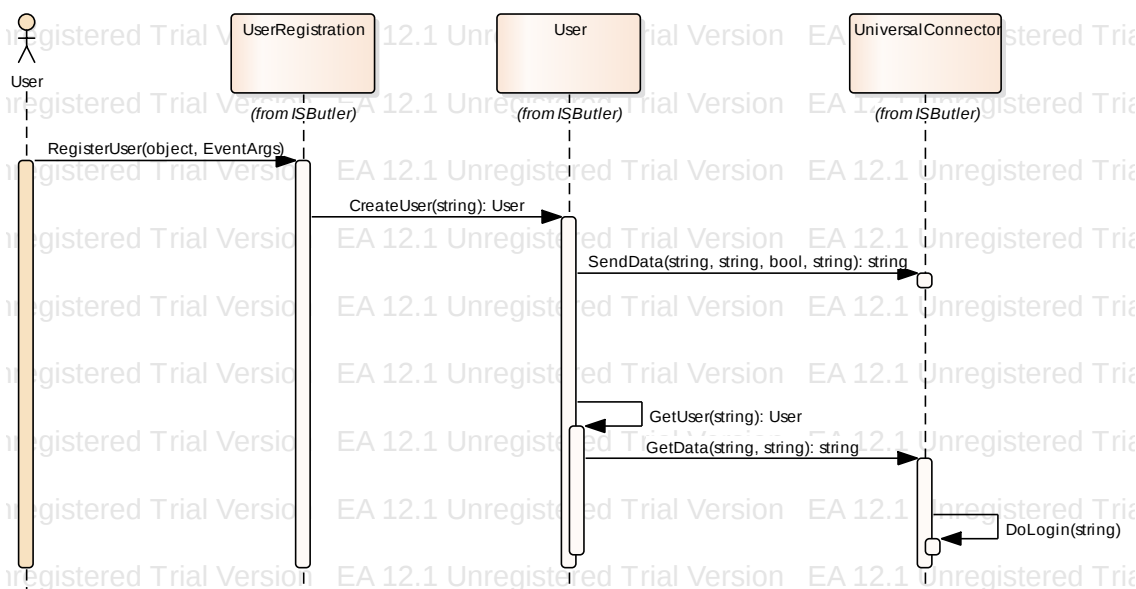
In de mobiele applicatie kunnen gebruikers zich registreren (use case 5), inloggen (use case 4), systemen koppelen (use case 3) en profielinformatie wijzigen (use case 6).

Om dit mogelijk te maken kan elke pagina een gebruiker opvragen of aanmaken via de User class (static methodes). Daarnaast bevat elke gebruiker ook informatie over de gekoppelde systemen en heeft het (extra) profielinformatie opgeslagen



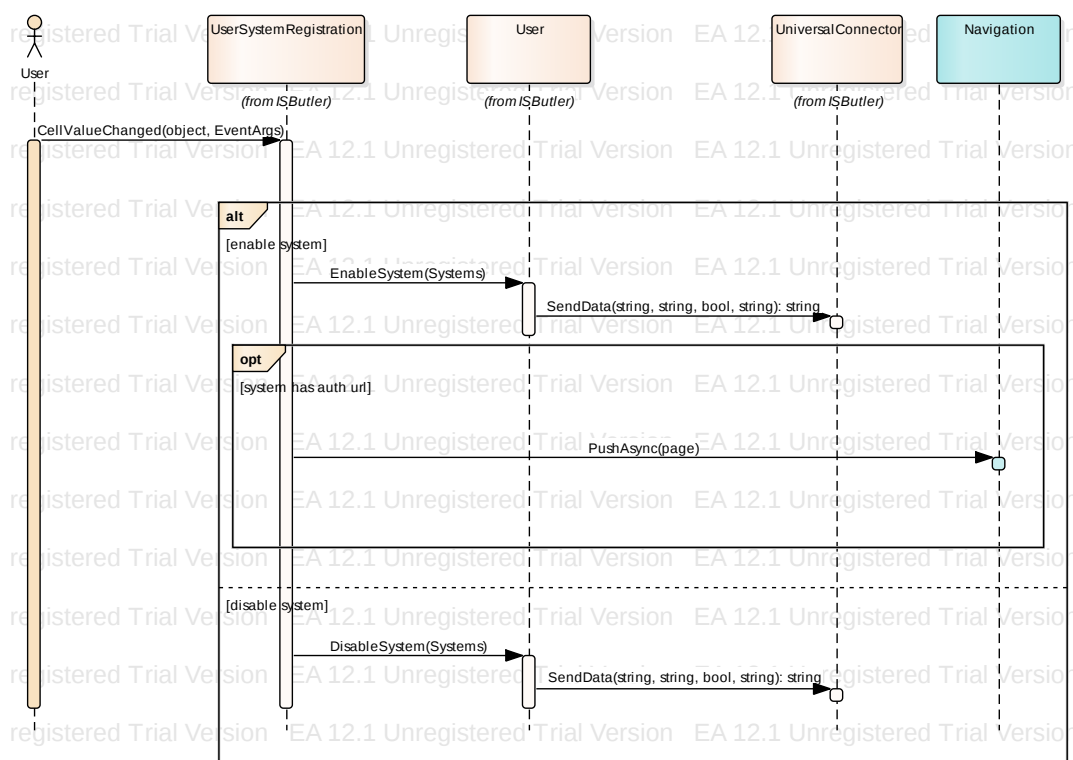
2.2.1 Registratie

De registratie pagina kan een gebruiker aanmaken met behulp van de static GetUser methode. Zoals in het onderstaande sequence diagram te zien is, wordt de gebruiker aangemaakt bij de universele connector en gelijk weer opgehaald, zodat de applicatie over een compleet user object beschikt (inclusief userid).



2.2.2 Systemen koppelen

Het koppelen van een systeem is een vrij eenvoudig proces. Op moment dat een gebruiker op de enable/disable knop klikt, wordt deze data doorgestuurd naar de universele connector. In de universele connector wordt deze vervolgens opgeslagen.

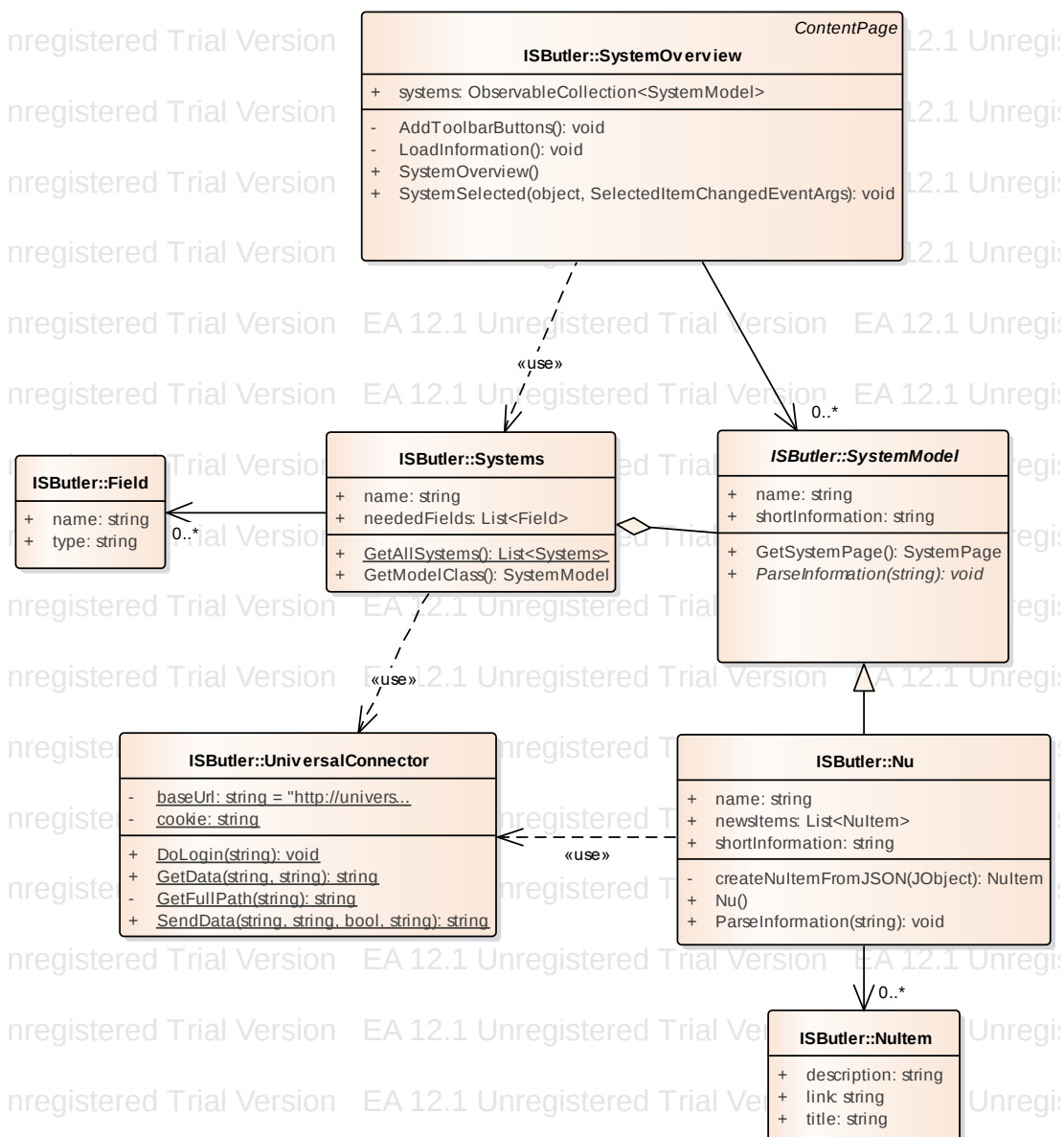


Zodra een systeem een authenticatie url meegeeft, moet de gebruiker zich authenticeren via de opgegeven url (dit scherm automatisch geopend). Nadat het online authenticatie proces is doorlopen wordt automatisch de applicatie weer geopend en wordt de autorisatiecode doorgegeven aan de universele connector

2.3 Informatiebronnen

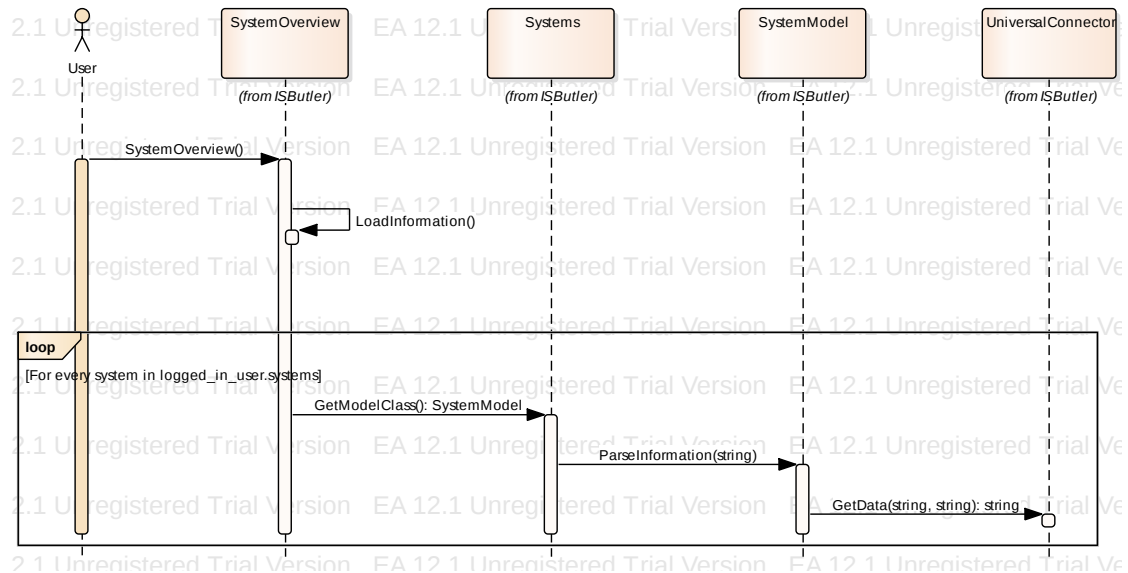
Naast gebruikers kan de mobiele applicatie ook informatie ophalen van verschillende systemen (use case 2). Zoals in hoofdstuk 2.2 te zien is, heeft elke gebruiker een list van gekoppelde systemen. Een pagina kan door deze systemen heen itereren en informatie opvragen uit de bijbehorende SystemModel class van het betreffende system object.

In het onderstaande model zijn niet alle systemen meegenomen. Ter voorbeeld worden alleen de systemen OpenWeatherMap en Nu weergegeven, de overige systemen zijn vergelijkbaar geïmplementeerd.



2.3.1 Overzichtspagina

Bij binnenkomst of verlaten van het gebouw krijgt de gebruiker een overzichtspagina te zien met alle systemen die gekoppeld zijn. In de onderstaande sequence diagram wordt dit proces beschreven.



2.3.2 Enkele informatiebron bekijken

Doordat bij het overzichtspagina elk systeem ingeladen is, hoeft een systeem niet opnieuw ingeladen te worden bij het aanroepen van een pagina van een enkele informatiebron.

In het onderstaande voorbeeld is een sequence diagram uitgewerkt van het bekijken van het systeem Nu.nl.

