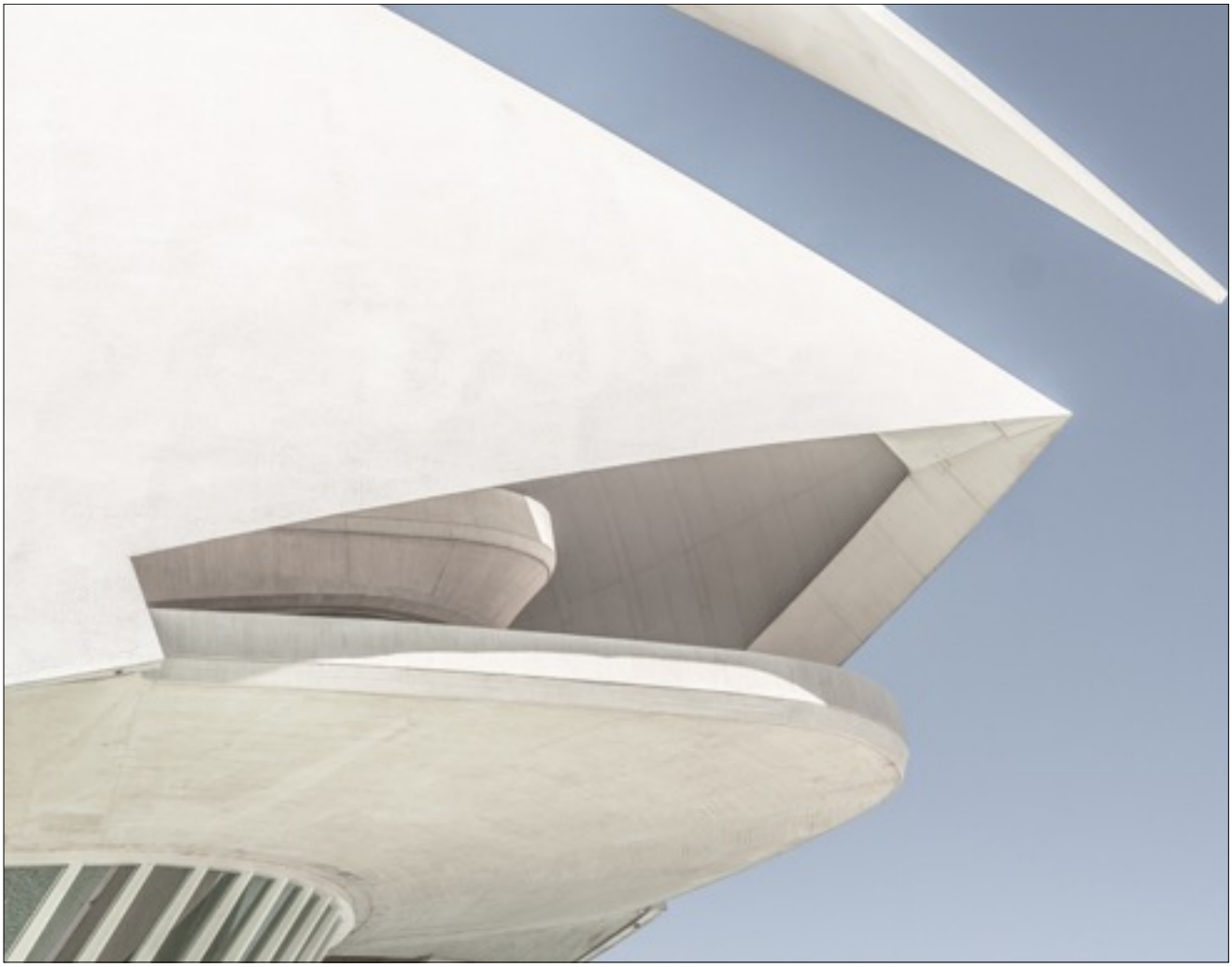


Software Architectuur Document

Opdrachtgever: Aranka Dol
Opdracht: Thema 4.2 SE, SA LT1
Auteurs: Malcolm Kindermans & Maurits van Mastrigt (Practicum duo 10B)
Datum: 25 november 2014

DOCUMENTHISTORIE

Datum	Versie	Beschrijving	Auteur(s)
25-11-2014	1.0	Intiële versie	Malcolm Kindermands & Maurits van Mastrigt

ACCORDERING DOCUMENT

Namens	Handtekening
Aranka Dol (Hanzehogeschool Groningen)	
Malcolm Kindermans	
Maurits van Mastrigt	

INHOUD

1. INLEIDING

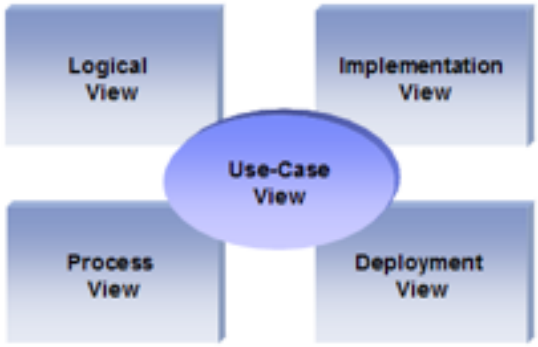
Deze sectie beschrijft het doel en de opbouw van dit document.

1.1. Doel van dit document

Dit document is geschreven om de architectuur te beschrijven van het DoIVA systeem, dat ontwikkeld is voor Aranka Dol. Het beschrijft een aantal verschillende architecturele views van het systeem om zo verschillende aspecten van het systeem te belichten.

Dit Software Architectuur Document (SAD) bevat een uitgebreide architecturele kijk op het systeem DoIVA ontwikkeld door Project Groep C. Het beschrijft een aantal verschillende architecturele views van het systeem om zo verschillende aspecten van het systeem te belichten.

Dit document beschrijft de verschillende RUP views op de software architectuur volgens het 4+1 view model.



Het 4+1 view model stelt de verschillende belanghebbenden in staat om vanuit hun eigen perspectief de invloed van de gekozen architectuur te bepalen. De Process View (communicatie van processen) is niet als los hoofdstuk uitgewerkt, maar ondergebracht bij de hoofdstukken 3.3 en 5.

1.2. Referenties

Onderstaand de referenties en desbetreffende vindplaats.

Titel	Versie	Auteur(s)	Vindplaats
Overdrachtsdocument & logboek	13 november 2014	Thema 4.1 SE, groep C	http://bit.ly/15g2bXd

1.3. Documentoverzicht

Onderstaand de hoofdstukken in dit document, de belanghebbende(n) en het doel van het hoofdstuk.

Hoofdstuk	Belanghebbende	Doel
2. Architecturele Eisen	Software Architect	Overzicht van architectureel relevante requirements.
3. Logical View	Ontwikkelaars (t.b.v. technisch ontwerp)	Inzicht in de functionele structuur van de applicatie.
4. Implementation View	Ontwikkelaars (t.b.v. de bouw)	Inzicht in de technische structuur van de applicatie.
5. Deployment View	Systeem Administrators	Inzicht in de manier waarop de applicatie wordt gedeployed en de manier waarop de (interne en externe) communicatie plaatsvindt.

2. ARCHITECTURELE EISEN

Deze sectie beschrijft de software eisen die voor het ontwikkelen van de software architectuur van belang zijn.

2.1. Niet-functionele eisen

De niet-functionele eisen zijn als volgt beschreven:

Bron	Naam	Architecturele relevantie	Geadresseerd in
Overdrachtsdocument	Belangrijke doelen, zoals ‘persoonlijke’ interacties en relevante antwoorden.	Keuzes die gemaakt moeten worden m.b.t. performance en tooling.	Paragraaf 3
Overdrachtsdocument	Cross-platform mobiele applicatie.	Beperkte mogelijkheden in platforms/tooling.	Paragraaf 5
Overdrachtsdocument	Tijdsbestek van circa 6 weken.	Beperkingen m.b.t. onderzoek naar mogelijkheden en gebruik van bepaalde tools.	Paragraaf 6
Overleg	Eenvoudig te reproduceren ontwikkelomgeving.	Structuur van de (deel)systemen.	Gesprekken met Aranka Dol.
Overleg	Beveiliging van gegevens niet essentieel in eerste versie.	Beveiliging	Gesprekken met Aranka Dol.
Installatiehandleiding	Installatie van mobiele applicatie.	Bouwen van Android applicatie (in de vorm van .APK bestand).	Paragraaf 1

2.2. Functionele eisen

In SCRUM is het gebruikelijk User Stories op te stellen, om zo de klant- en gebruikerswensen vast te leggen. Onderstaand de gedefinieerde User Stories, welke dienen als de functionele systeemeisen.

#	User Story
1	As a USER I want a secure and personal APP as my virtual assistant.
2	As a USER I want to ask a question so that I can get an answer to my question.
3	As a USER I want the virtual assistant ask me for feedback about the given answers.
4	As a USER I want to dictate my question so that I do not have to write them.
5	As a USER of the APP I want to give a suggestion on the given ANSWER.
6	As a USER I want a visual avatar to give emotional feedback so that I know that my question has been received or aknowledged.
7	As a USER I want to have the answers read back to me, so that I don't have to read them.
8	As a RESEARCHER I want the answers on user questions to be as accurate as possible so that their needs are fulfilled.
9	As an ADMIN I want to sign into the CMS so that my information is secure.
10	As an ADMIN I want to have a log on the dashboard in the CMS to have insight in statistics.
11	As an ADMIN I want to MANAGE the user accounts that can use the app.
12	As an ADMIN I want to MANAGE the questions so that they can be changed/created at any moment.
13	As an ADMIN I want to MANAGE unanswered questions so that I can increase the rate of answers.
14	As an ADMIN I want to manage the AI in a proper way.
15	As a TESTER I want automated tests for the admin controller.
16	As a TESTER I want automated tests for the answer controller.
17	As a TESTER I want automated tests for the answer feedback controller.
18	As a TESTER I want automated tests for the question controller.
19	As a TESTER I want automated tests for the subject controller.
20	As a TESTER I want automated tests for the suggestion controller.
21	As a TESTER I want automated tests for the user controller.

3. LOGICAL VIEW

Deze sectie beschrijft de architectureel significante logische opbouw van het systeem.

3.1. Lagen

De opbouw van het systeem kan worden onderverdeeld in vier lagen.

Laag	Doel
Presentatie	De grafische interface voor de interactie met de eindgebruikers.
Service	Het communiceren tussen de presentatielaag en de domeinlaag. Dit wordt gedaan m.b.v. een API.
Domein	De applicatielogica, zoals het presenteren en verwerken van gegevens.
Data	Het aanbieden en persisteren van gebruikersdata.

3.2. Deelsystemen

Het systeem bestaat uit drie deelsystemen, namelijk:

Deelsysteem	Doel
Mobiel internet apparaat (smartphone)	Dit apparaat heeft de app geïnstalleerd, wat de gebruiker in staat stelt tot interactie met het systeem.
Personal Computer	Met behulp van een webbrowser kan er worden ingelogd in het Content Management Systeem (CMS). In het CMS kan het systeem worden beheerd en kunnen er gebruikersstatistieken worden ingezien.
Webserver	Dit systeem verzorgd de API, die wordt gebruikt door de mobiele app en het CMS. Tevens worden de CMS webpagina's geserveerd.
Database server	De database bevat de gebruikersdata en stelt de webserver in staat deze data op te halen en te muteren.

3.3. Realisatie van functionele eisen

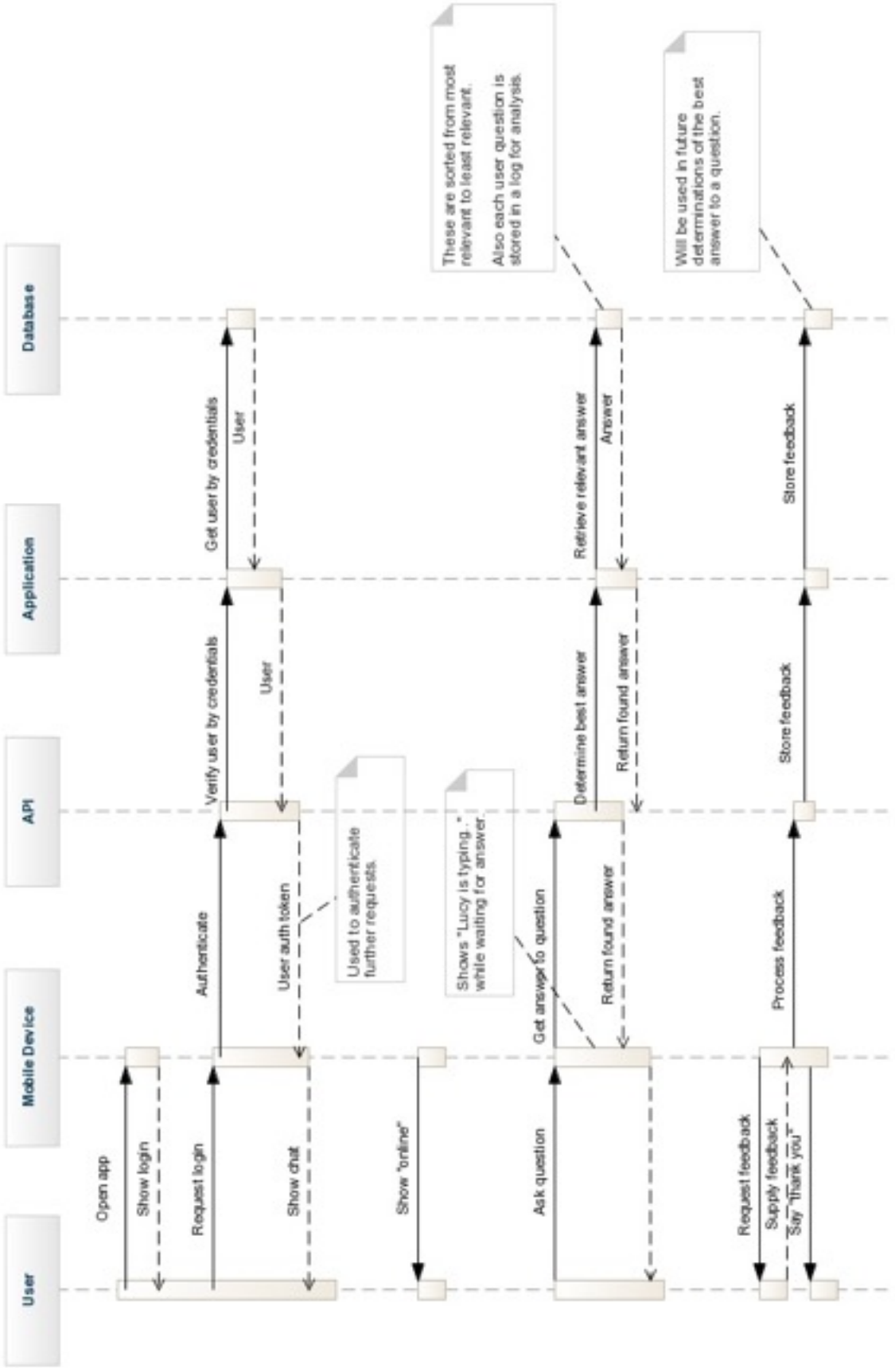
De onderstaande tabel maakt de implementatie status van de functionele eisen inzichtelijk.

#	User Story	Implementatie
1	As a USER I want a secure and personal APP as my virtual assistant.	✓ Volledig
2	As a USER I want to ask a question so that I can get an answer to my question.	✓ Volledig
3	As a USER I want the virtual assistant ask me for feedback about the given answers.	✓ Volledig
4	As a USER I want to dictate my question so that I do not have to write them.	✓ Volledig
5	As a USER of the APP I want to give a suggestion on the given ANSWER.	✓ Volledig
6	As a USER I want a visual avatar to give emotional feedback so that I know that my question has been received or aknowledged.	✗ Niet geïmplementeerd.
7	As a USER I want to have the answers read back to me, so that I don't have to read them.	✗ Niet geïmplementeerd.
8	As a RESEARCHER I want the answers on user questions to be as accurate as possible so that their needs are fulfilled.	✓ Volledig
9	As an ADMIN I want to sign into the CMS so that my information is secure.	✓ Volledig
10	As an ADMIN I want to have a log on the dashboard in the CMS to have insight in statistics.	✓ Volledig
11	As an ADMIN I want to MANAGE the user accounts that can use the app.	✓ Volledig
12	As an ADMIN I want to MANAGE the questions so that they can be changed/ created at any moment.	✓ Volledig
13	As an ADMIN I want to MANAGE unanswered questions so that I can increase the rate of answers.	✓ Volledig
14	As an ADMIN I want to manage the AI in a proper way.	✗ Niet geïmplementeerd.
15	As a TESTER I want automated tests for the admin controller.	✓ Volledig
16	As a TESTER I want automated tests for the answer controller.	✓ Volledig
17	As a TESTER I want automated tests for the answer feedback controller.	✓ Volledig

#	User Story	Implementatie
18	As a TESTER I want automated tests for the question controller.	✓ Volledig
19	As a TESTER I want automated tests for the subject controller.	✓ Volledig
20	As a TESTER I want automated tests for the suggestion controller.	✓ Volledig
21	As a TESTER I want automated tests for the user controller.	✓ Volledig

Een groot deel van de functionaliteit bestaat uit de tweede User Story: *"As a USER I want to ask a question so that I can get an answer to my question."*. Het sequentiediagram van deze User Story maakt duidelijk inzichtelijk hoe de verschillende componenten communiceren en bijdragen aan de functionaliteit.

Zie de volgende pagina voor het sequentiediagram.

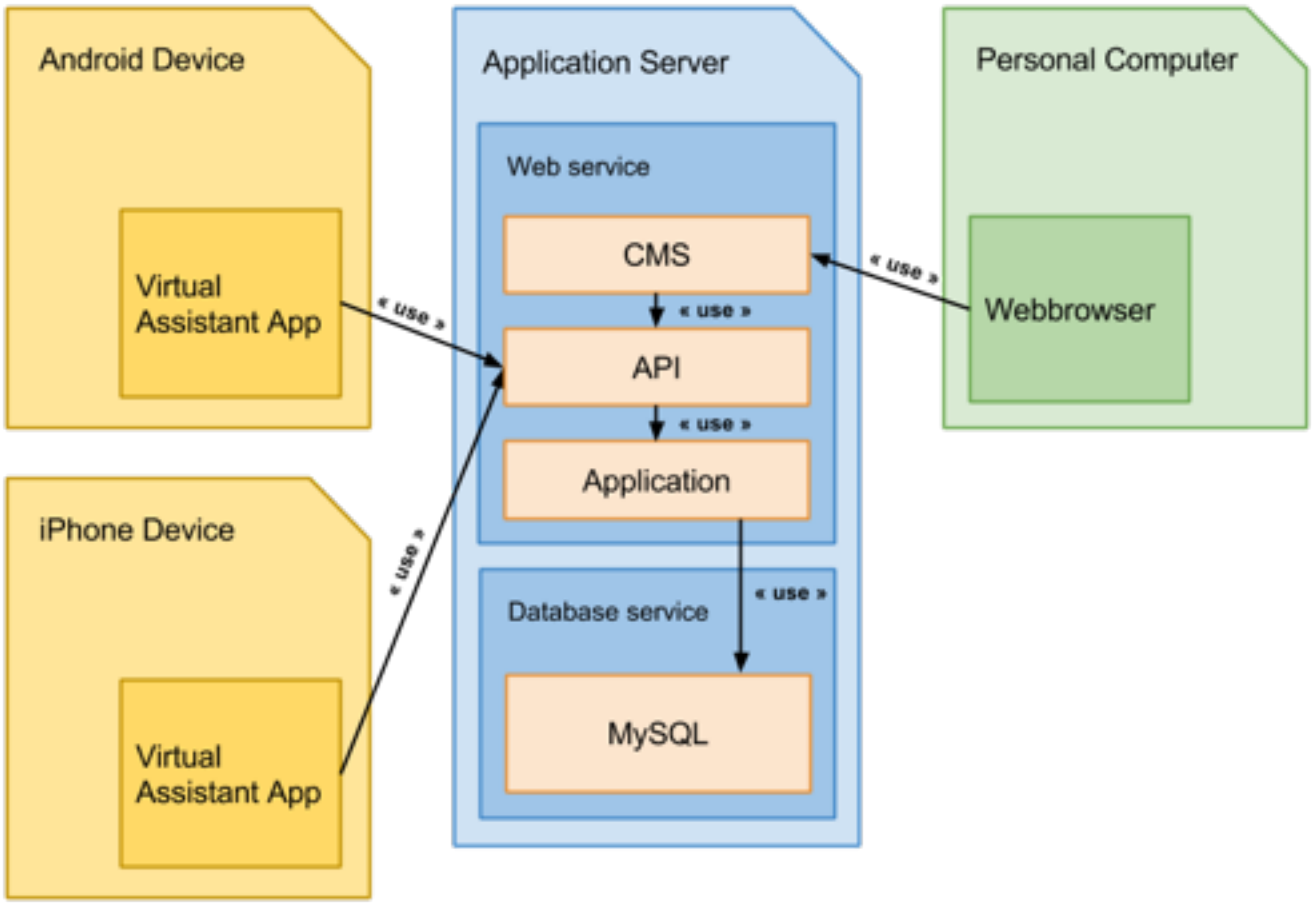

User
Mobile Device
API
Application
Database
Open app
Show login
Request login
Authenticate
Verify user by credentials
Get user by credentials
User
User
User auth token
Used to authenticate further requests
Show chat
Show "online"
Shows "Lucy is typing..." while waiting for answer
Ask question
Get answer to question
Determine best answer
Retrieve relevant answer
Answer
Return found answer
Return found answer
These are sorted from most relevant to least relevant.
Also each user question is stored in a log for analysis.
Request feedback
Process feedback
Store feedback
Store feedback
Supply feedback
Say "thank you"
Will be used in future determinations of the best answer to a question.

4. IMPLEMENTATION VIEW

Deze sectie beschrijft de technische invulling van de logical view.

4.1. Packagestructuur

Het onderstaande diagram geeft weer hoe de deelsystemen met elkaar interacteren.



Figuur 4.1 • Package diagram

Dit diagram weerspiegeld hoe de mobiele applicatie interacteert met de API en hoe een systeem administrator via een webbrowser gebruik kan maken van het CMS. De onderliggende laag, met name de applicatielaag, biedt alle functionaliteiten en presenteert de (MySQL) database gegevens, aan de API en het CMS, in het vereiste formaat.

4.2. Invulling van lagenstructuur

Deze paragraaf beschrijft de technische invulling van de, in de logica view, onderscheiden lagen.

PRESENTATIELAAG

De presentatie laag omvat de deelsystemen die een interface bieden aan de eindgebruikers en beheerders van het systeem. Dit wordt gedaan door middel van een mobiele applicatie en een Content Management Systeem (CMS). Beide onderdelen communiceren, via de service laag, met de domein (applicatie) laag voor het ophalen en muteren van gegevens.

Onderstaand worden de keuzes toegelicht (regels voor de componenten):

Mobiele Applicatie

De mobiele applicatie moet een cross-platform applicatie zijn die eenvoudig is in gebruik. Hierdoor is er voor bestaande oplossingen als Cordova en OnsenUI gekozen (zie paragraaf 4.3). Dit neemt werk uit handen, waardoor er meer focus gelegd kan worden op de gebruikerservaring, terwijl de applicatie toch grafisch aantrekkelijk blijft.

Content Management Systeem

Het beheersysteem, of CMS, is tevens geïmplementeerd aan de hand van bestaande componenten. Voor een aantrekkelijk grafische interface is er Twitter Bootstrap gebruikt. De applicatielogica aan de client kant is gedaan met behulp van AngularJS. Ook deze punten worden nader toegelicht in paragraaf 4.3.

Het serveren van de CMS pagina's wordt gedaan met behulp van een webservice. Hierbij is er gekozen voor het Laravel framework, omdat dit framework makkelijk is in gebruik, geen stijle leercurve heeft, en er al de nodige ervaring in de groep aanwezig was. De keuzecriteria zijn dus met name ontwikkelsnelheid en onderhoudbaarheid (maintainability) van het systeem.

SERVICE LAAG

De communicatie gebeurd met behulp van HTTP requests. Hierbij wordt het RESTful design principe gebruikt. Dit houdt in dat alle entiteiten op één uniforme manier, via de Application Programming Interface (API), benaderd kunnen worden. Dit vereenvoudigd het redeneren over de communicatie (HTTP requests) en het opsporen en oplossen van fouten.

Het HTTP protocol heeft veel overhead, maar biedt daarentegen veel structuur en garandeert een stabiele omgeving. Tevens wordt dit protocol door webbrowsers gebruikt, waardoor er veel informatie over te vinden is, wat het implementeren eenvoudiger maakt.

DOMEINLAAG

De applicatielogica is volledig uitgewerkt in de PHP scripttaal. Er is hierbij gekozen voor een framework, omdat hiermee sneller en transparanter ontwikkeld kan worden. Dit is voordelig voor alle ontwikkelaars. Tevens is het gekozen framework zeer expressief en biedt het veel mogelijkheden. Dit voorkomt dat ontwikkelaars onnodig tijd besteden aan het “opnieuw uitvinden van het wiel” - wat een gevaar voor elk project is. Het gebruiken van bestaande tools stelt de ontwikkelaars in staat zich meer te richten op de functionaliteiten en minder op implementatie technieken.

DATALAAG

Voor het persisteren van data zijn er veel opties. Omdat de gegevens het best in een relationeel model passen, is er de keuze gemaakt voor een relationele database. Hierbij is er gekozen voor MySQL, vanwege het gemak in gebruik. Er is veel documentatie beschikbaar en er zijn weinig ontwikkelaars die niet met dit softwarepakket gewerkt hebben. Ook dit scheelt ontwikkeltijd, waardoor er meer gericht kan worden op de functionaliteit in plaats van implementatie.

Tevens is MySQL zeer ver doorontwikkeld, waardoor dingen zoals stabiliteit en hardware specificatie eigenlijk geen rol meer spelen.

4.3. (Her)gebruik van componenten en frameworks

Deze paragraaf beschrijft de componenten en frameworks die in het project zijn gebruikt/toegepast.

PRESENTATIELAAG

De presentatielaag is op te delen in de mobiele applicatie en de front-end van het CMS.

Mobiele App

De volgende componenten/frameworks zijn gebruikt bij het ontwikkelen van de mobiele applicatie:

- **Cordova** – Vanwege de mogelijkheid om te ontwikkelen zonder gebruik te maken van de native APIs van devices;
- **Onsen UI** – Vanwege de mogelijkheid om simpel een native “look and feel” te creëren in een webomgeving. Daarnaast heeft Onsen UI een betere performance voor mobiele apparaten dan een regulier HTML5 UI Framework.

Front-end CMS

De volgende componenten/frameworks zijn gebruikt bij het ontwikkelen van de front-end:

- **Twitter Bootstrap** – Vanwege de moeilijkheidsgraad om er een mooi product mee te maken;
- **AngularJS** – De leercurve van AngularJS is gelukkig niet te stijl, zeker ten opzichte van KnockoutJS. Het is een JavaScript framework, wat gebruikt is om functionaliteit te offloaden naar de webbrowser van de gebruiker;
- **D3 plugins** – Voor visualisatie van de donut/pie chart – deze bleken het makkelijkste te implementeren.

SERVICE LAAG

De service laag bestaat uit:

- **RESTful API** – Vanwege de eenvoud van de benadering is er voor gekozen om de API RESTful te implementeren. Complexiteit wordt hierdoor zoveel mogelijk verminderd.

Deze API is geïmplementeerd in het Laravel framework. Zie hiervoor ook het kopje “Domeinlaag”.

DOMEINLAAG

De domeinlaag is de applicatielogica en valt te omvatten in de term “back-end”. Hieronder valt het stukje CMS en de applicatielogica. Hiervoor is het volgende framework gebruikt:

- **Laravel (PHP-framework)** – Vanwege de hoeveelheid documentatie die er over beschikbaar is en de duidelijke MVC-structuur die het biedt zonder te veel op te leggen.

DATALAAG

De datalaag bestaat uit het volgende onderdeel:

- **MySQL** – Als database voor de opslag van gegevens, vanwege het sterke relationele karakter van onze data en de kleine pool van proefpersonen.

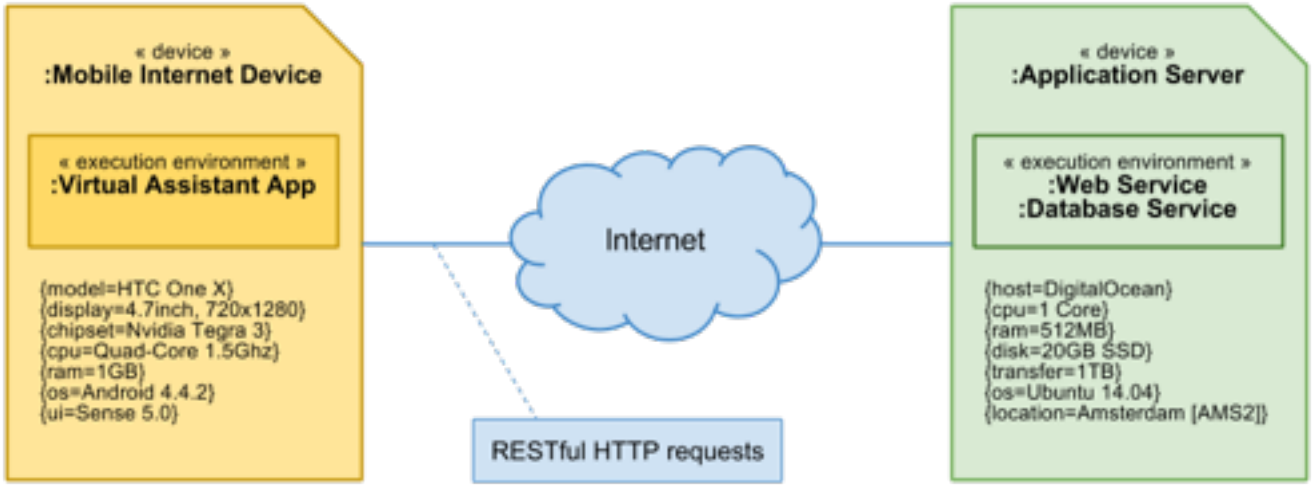
5. DEPLOYMENT VIEW

Deze sectie beschrijft de fysieke netwerk (hardware) configuraties waarop de software gaat draaien.

Naam	Type	Omschrijving
Mobile Internet Device	Smartphone, HTC One X, Android	De mobiele applicatie draait op dit toestel. Met de app kan ingelogd worden, om vervolgens vragen te stellen aan de Virtuele Assistent.
Applicatie Server	VPS, Ubuntu 14.04	De applicatie server verzorgt de webservice (CMS, API, en applicatielogica) en de database service (MySQL).

5.1. Deployment-diagram

Onderstaand het deployment-diagram van het systeem.



Figuur 5.1 • Deployment diagram

Zoals aangegeven in het diagram wordt de communicatie via het internet gedaan en dus niet via een lokale LAN. Dit introduceert mogelijke beveiligingsproblemen. Deze kunnen worden opgelost door gebruik te maken van een SSL certificaat en dus in plaats van HTTP requests, HTTPS requests (S voor Secure) te doen.