Technisch ontwerp

Database Factory 5

Informatieblad

ISE-Project Omgevingswet

Namens de Hogeschool van Arnhem en Nijmegen uitgevoerd voor CGI Nederland.

Het project wordt uitgevoerd door Database Factory 5.

Projectleden:

* Michiel Bos 561485
* Ricardo van Burik 569129
* Johan Heij 584299
* Duncan Luiten 553272
* Tristan de Roo 563431

Begeleiders:

* Tim de Goede
* Pim Haenen

Datum: 19 mei 2017

Locatie: Ruitenberglaan 31, 6826 CC Arnhem

Versie 1.0

Versiebeheer

|  |  |  |
| --- | --- | --- |
| Versie | Datum | Aanpassingen |
| 1.0 | 19 mei 2017 |  |
| 1.1 |  |  |
|  |  |  |

Inhoudsopgave

[1. Inleiding 5](#_Toc482969674)

[2. Netwerkarchitectuur 6](#_Toc482969675)

[3. Programmeertalen 7](#_Toc482969676)

[3.1. Beheerapplicatie 7](#_Toc482969677)

[3.2. Website 7](#_Toc482969678)

[3.3. Database 7](#_Toc482969679)

[3.4. Webserver 8](#_Toc482969680)

[4. Front-End Framework 9](#_Toc482969681)

[4.1. Beheerapplicatie 9](#_Toc482969682)

[4.2. Website 9](#_Toc482969683)

[5. Back-End Framework 10](#_Toc482969684)

[5.1. Beheerapplicatie 10](#_Toc482969685)

[5.2. Website 10](#_Toc482969686)

[6. Database 11](#_Toc482969687)

[6.1. Ondernomen stappen tot PDM 11](#_Toc482969688)

[6.1.1. Toelichting PDM 11](#_Toc482969689)

[6.2. Datakwaliteit 13](#_Toc482969690)

[6.2.1. Concurrency 13](#_Toc482969691)

[6.2.2. Lost updates 13](#_Toc482969692)

[6.3. Beveiliging 14](#_Toc482969693)

[6.4. Performance 14](#_Toc482969694)

[6.4.1. Indexen 14](#_Toc482969695)

[6.5 Stored Procedures 14](#_Toc482969696)

[6.6 Triggers 15](#_Toc482969697)

[6.7 Check constraints 15](#_Toc482969698)

[7 Opsomming niet-functionele requirements 16](#_Toc482969699)

[Literatuurlijst 17](#_Toc482969700)

# 1. Inleiding

Dit document geeft een beeld van de technische aspecten van dit project. Het laat een overzicht zien van de gemaakte technische keuzes en de gekozen frameworks.

In het hoofdstuk ‘Netwerkarchitectuur’ wordt beschreven welke componenten via het netwerk moeten communiceren, welke database is gekozen en waar de server is gelokaliseerd. Dit wordt gevolgd door een overzicht van de gebruikte programmeertalen.

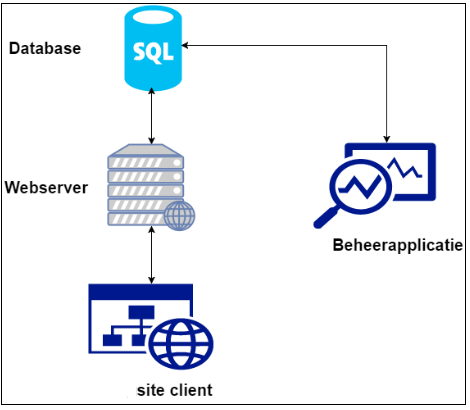
Vervolgens worden de gekozen front-end en back-end frameworks beschreven en wordt toegelicht hoe die gebruikt gaan worden. Ook wordt er uitgelegd hoe deze frameworks werken en waarom wij de keuze gemaakt hebben om ze te gebruiken.

Daarna wordt toegelicht hoe de kwaliteit van de data in de database wordt gewaarborgd, hoe het PDM is ontstaan en hoe de website gaat worden beveiligd. En tot slot worden de niet-functionele requirements weergegeven.

# 2. Netwerkarchitectuur

Dit project heeft vier componenten die via het netwerk moet communiceren:

* Microsoft SQL Server
  + De database die gebruikt wordt heet Microsoft SQL Server. Hier wordt alle informatie in opgeslagen. De database communiceert via de webserver met de website en direct met de beheerapplicatie
* Website
  + De website draait op de webserver. De website communiceert alleen met de webserver. Als de website iets wilt opslaan in de database, wordt dit via de webserver gecommuniceerd.
* Beheerapplicatie
  + Met de beheerapplicatie wordt alle data in de database beheert. Alle data die hier word bewerkt, wordt direct naar de database gecommuniceerd.
* Webserver
  + De webserver is het middelpunt van de communicatie tussen de website en de database. De website die op de webserver staat, is zichtbaar met behulp van het internet.



Figuur 1 Netwerkarchitectuur.

# 3. Programmeertalen

## 3.1. Beheerapplicatie

Voor de beheerapplicatie is uiteindelijk de keuze gemaakt voor C#. Als eerst is MS Acces gevallen. Hoewel MS Acces goed is om een demo/prototype applicatie te maken, is C# beter voor dit project omdat het bedrijf de intentie heeft om het project uit te breiden. Dit komt omdat C# meer functionaliteiten tot zijn beschikking heeft.

## 3.2. Website

* De website wordt opgebouwd met HTML en CSS. Om hier niet teveel tijd aan te besteden is er gekozen om te zoeken naar een goede template dat sterk samenhangt met Bootstrap. Zo kan er meer aandacht worden besteed aan de functionaliteiten van de webapplicatie, in plaats van de opmaak. Naast het besparen van tijd hebben we Bootstrap gekozen om de volgende redenen:
  + Alle pagina’s zijn consistent
  + Ervaring binnen de groep
  + Makkelijk te integreren
  + Responsive
* Om de webapplicatie dynamisch te maken, wordt er verbinding gemaakt met een MSSQL Database. Deze database zal overigens ook in verbinding staan met de beheerapplicatie.
* Data wordt met behulp van het PHP Framework Laravel uit de database gehaald. Ook maakt Laravel het makkelijk om verschillende onderdelen van een site te herbruiken. Verdere punten die mee hebben gespeeld in het besluit zijn:
  + Goede meegeleverde documentatie
  + Veel online learning sessions (Laracast, Pluralsight)
  + Veelgebruikte bestanden aanmaken via Artisan commands (Models, Controllers)
  + Blade Templates, hiermee kan er overzichtelijk gemaakte laravel functies in de html worden verwerkt.
  + Makkelijke Dependency Injection.
  + Automated Authentication.

## 3.3. Database

De keuze is op Microsoft SQL Server gevallen, omdat alle projectleden hier de meeste ervaring mee hebben. Vanuit school is MSSQL aanbevolen en wordt ook het meest ondersteund. Vergeleken met een database zoals MySQL heeft MSSQL het voordeel dat *CHECKS* uitgeschreven kunnen worden. Voor de Omgevingswet applicatie, waar zich veel *CHECK* constraints in bevinden, is dit dus ideaal.

## 3.4. Webserver

Op dit moment is het onduidelijk op welk webserver de website gehost gaat worden omdat dit gehost gaat worden op de webserver van CGI.

# 4. Front-End Framework

In dit hoofdstuk worden de keuzes met betrekking tot het front-end framework toegelicht voor zowel de beheerapplicatie als de webapplicatie.

## 4.1. Beheerapplicatie

Voor de beheerapplicatie is de keuze gemaakt om de Graphical User Interface (GUI) framework Windows Presentation Foundation (WPF) te gebruiken. WPF is een grafisch subsysteem dat deel uitmaakt van het DotNet framework versie 3 of hoger van Microsoft. De opmaaktaal waarin gewerkt wordt heet Extensible Application Markup Language (XAML). XAML is een mix van de opmaaktalen HTML, CSS en XML. Wat in de XAML code wordt gemaakt, wordt real time geüpdate in het designer scherm dat standaard boven de XAML code staat.

WPF wordt gebruikt omdat:

* Het een moderner GUI oplevert dan de nog veelgebruikte standaard front-end frameworks voor C# (winforms).
* Met WPF een programmeur op een hoger niveau aanpassingen kan maken aan een grafisch element dan met een winforms framework. Dit betekent dat de sub-elementen van een grafisch element kunnen worden aangepast.
* Een programmeur met het grid systeem op gemakkelijke wijze een consistent formulier kan maken.
* Met een resource dictionary file genaamd ‘app.xaml’ kan een programmeur alle stijlen aanpassen die voor de hele applicatie geldt. De programmeur kan in de app.xaml aangeven dat alle knoppen in de applicatie 80 pixels hoog en 80 pixels breed moeten worden en de kleur blauw moeten zijn.

## 4.2. Website

Het front-end framework dat wordt gebruikt voor de webapplicatie is Bootstrap. Zoals eerder aangegeven heeft Bootstrap een aantal voordelen die erg toepasselijk zijn op het project Omgevingswet. De webapplicatie gaat gebruikt worden door bewoners van een gemeente. Het moet dus het ‘visitekaartje’ worden van de gemeente. Met het gebruik van Bootstrap kan er snel een goed uitziende basis worden neergezet, waar vervolgens de benodigde functionaliteiten in ingebouwd kunnen worden.

Ondanks dat de huidige applicatie eerst gebruikt gaat worden als demo, is het goed om na te denken over de toekomst. Wij denken dat met het gebruik van Bootstrap de ‘overdracht’ van het project naar een uiteindelijk development team van CGI makkelijker wordt. Bootstrap wordt namelijk wereldwijd veel gebruikt.

# 5. Back-End Framework

In dit hoofdstuk worden de keuzes met betrekking tot het back-end framework toegelicht voor zowel de beheerapplicatie als de webapplicatie.

## 5.1. Beheerapplicatie

Aangezien voor de beheerapplicatie de front-end framework WPF is gekozen, is het gebruik van het back-end framework DotNet verplicht. WPF heeft functionaliteiten van DotNet nodig om goed te kunnen functioneren. Dit zijn voornamelijk functionaliteiten die het afhandelen van data betreffen.

DotNet wordt gebruikt omdat:

* WPF gebruik maakt van DotNet om te functioneren.
* Het DotNet framework helpt bij:
  + Het ophalen van data uit de database
  + Het afhandelen van data die is opgehaald uit de database
  + Het verbinden van de data aan een datagrid.

## 5.2. Website

Het PHP framework Laravel wordt gebruikt als back-end framework voor de webapplicatie. Om de functionaliteiten binnen de applicatie te kunnen gebruiken, moeten gebruikers een account aanmaken. Dit is een cruciaal onderdeel van de applicatie. Echter is het inloggen van een gebruiker niet aan bod gekomen tijdens het eerste gesprek. Alhoewel inloggen ‘vanzelfsprekend’ is, kan het realiseren van de functionaliteit veel tijd kosten. Door gebruik te maken van Laravel kan er al snel automatisch een basic authentication gemaakt worden. Dit was niet de voornaamste reden om Laravel te gebruiken. Na het inventariseren van de benodigde functionaliteiten konden we concluderen de website kan worden gebouwd met behulp van PHP. Echter, om onze eigen kennis te verbreden, hebben we gekozen voor Laravel. Hierbij kan er worden vastgesteld dat naarmate het project vordert, we meer profijt van Laravel krijgen. Het framework maakt bijvoorbeeld het gebruik van queries makkelijker, aangezien het “denken” op de achtergrond van Laravel plaatsvindt.

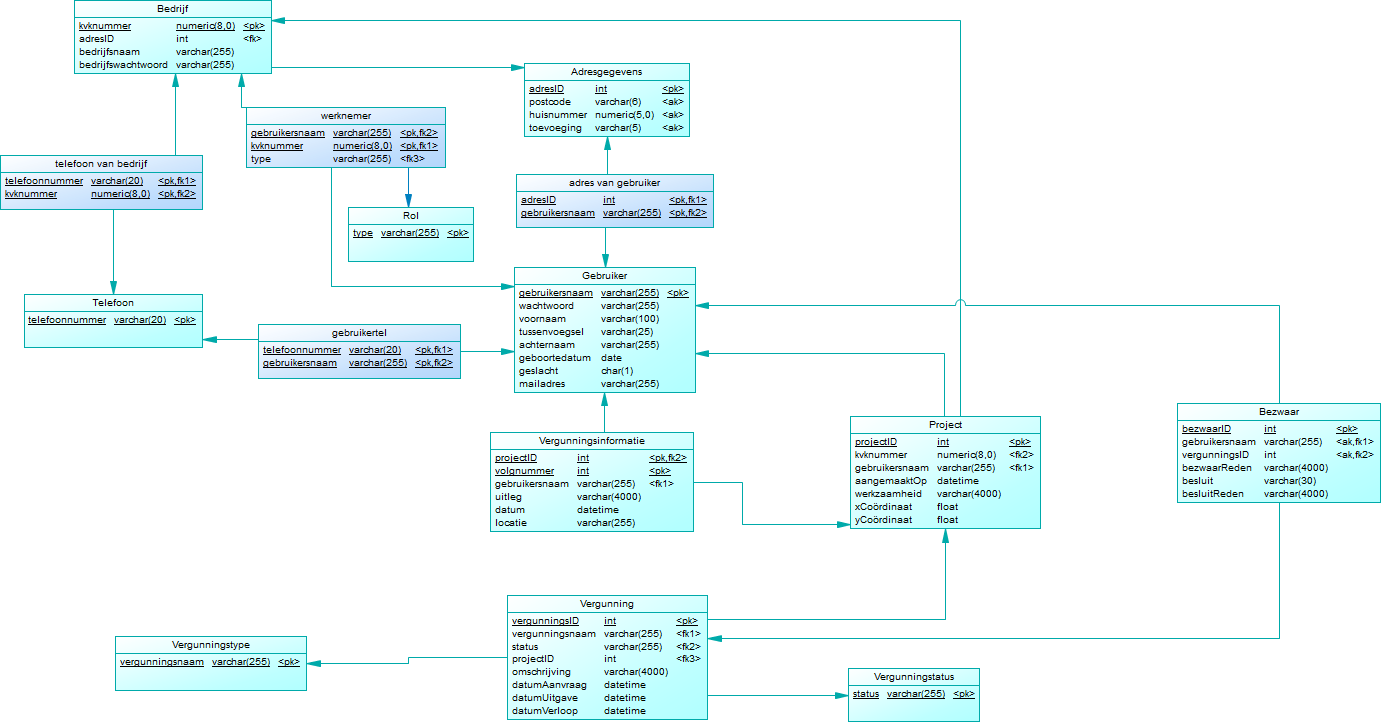
De connectie met MSSQL binnen Laravel is ook gemakkelijk te leggen, aangezien de basis al ingebouwd is. Enkel de servernaam, databasenaam, gebruikersnaam en wachtwoord moeten worden ingevuld, Laravel doet de rest.

# 6. Database

De database is misschien wel het belangrijkste product dat opgeleverd gaat worden. Het is in elk geval het meest belaste product. Dat wil zeggen dat er veel gebruikers tegelijkertijd indirect gebruik van maken, via de webapplicatie en de beheerapplicatie. Daarnaast is veiligheid een belangrijk en actueel thema, en is datakwaliteit ook van belang.

## 6.1. Ondernomen stappen tot PDM

Het PDM is automatisch gegenereerd vanuit het CDM.



Figuur 2 Physical Data Model

### 6.1.1. Toelichting PDM

In onderstaande tabel worden de verschillende entiteiten welke aanwezig zijn in het PDM uitgelegd.

|  |  |
| --- | --- |
| **Entiteit** | **Uitleg** |
| Adresgegevens | In de entiteit worden alle gegevens welke betrekking hebben tot een adres opgeslagen. De postcode samen met het huisnummer en de toevoeging zijn de alternative key. Deze drie kolommen zijn uniek genoeg om één unieke locatie op een kaart te krijgen. |
| Adres van gebruiker | Elke gebruiker woont op een adres. De tabel ‘Adres van gebruiker’ is een koppeltabel, aangezien meerdere gebruikers op hetzelfde adres kunnen wonen. |
| Bedrijf | Een gebruiker kan een bedrijf zijn. Bedrijven moeten namelijk ook invloed uit kunnen oefenen op hun leefomgeving. |
| Bezwaar | Belanghebbenden van een project kunnen bezwaar maken. Een bezwaar wordt gebruikt om aan de gemeente te laten weten dat mensen in de omgeving van het project het niet eens zijn met de gestelde plannen. |
| Gebruiker | Een persoon die de applicatie wil gebruiken moet zich registreren. Er wordt dan een gebruiker binnen het systeem aangemaakt. |
| Gebruikertelefoon | Elke gebruiker is verplicht een telefoonnummer op te geven. Echter is de tabel Gebruikertelefoon een koppeltabel, omdat meerdere gebruikers hetzelfde telefoonnummer kunnen hebben en een gebruiker ook meerdere nummers kan bezitten. |
| Project | Alle informatie met betrekking tot een project wordt opgeslagen in de tabel Project. Aan een project zit een gebruiker vast, welke de initiatiefnemer is. |
| Rol | Een gebruiker kan verschillende rollen binnen het systeem hebben. Met gebruik van rollen kan er goed worden afgevangen welke functionaliteiten beschikbaar zijn voor gebruikers. (denk hierbij aan bijvoorbeeld admins, vertegenwoordigers) |
| Telefoon | Telefoon is de hoofdtabel waarin alle telefoonnummers van gebruikers en bedrijven worden opgeslagen. |
| Telefoon van bedrijf | Elk bedrijf moet een telefoonnummer hebben. De tabel ‘Telefoon van bedrijf’ is een koppeltabel tussen telefoon en bedrijf. |
| Vergunning | Aan een project kunnen verschillende vergunningen gekoppeld worden. Zo kan er makkelijk worden gespecificeerd wat er precies nodig is om een project te laten voltooien. |
| Vergunningsinformatie | Alle informatie die wordt aangeleverd bij een project of vergunning wordt opgeslagen in de vergunningsinformatie. Denk hierbij aan documenten met informatie en links naar relevante webpagina’s. |
| Vergunningstatus | Elke vergunning heeft een status wanneer deze wordt aangevraagd. De status kan gedurende het project wijzigen. De verschillende statusmogelijkheden zijn opgeslagen in de tabel ‘Vergunningstatus’. |
| Vergunningstype | Vergunningen hebben verschillende types. Zo is er bijvoorbeeld een evenementvergunning en een monumentvergunning. De verschillende vergunningstype worden opgeslagen in deze tabel. |
| Werknemer | Werknemer is een koppeltabel tussen bedrijf en gebruiker. Elke gebruiker binnen een bedrijf kan handelingen voor het bedrijf ondernemen. Belangrijk hierbij is dat het altijd mogelijk is om na te gaan welke werknemer een handeling namens het bedrijf heeft gedaan. |

Tabel 1 Toelichting PDM.

## 6.2. Datakwaliteit

Het beschermen van de datakwaliteit wordt gedaan door middel van constraints. Het gaat hierbij om basisfunctionaliteiten zoals primary keys, alternative keys en foreign keys, maar ook om check constraints, triggers, stored procedures en functions. Voor alle gemaakte constraints wordt uitgelegd of hierbij problemen kunnen optreden met betrekking tot concurrency en lost updates.

### 6.2.1. Concurrency

Dit is nog niet aan de orde gekomen.

### 6.2.2. Lost updates

Dit is nog niet aan de orde gekomen.

## 6.3. Beveiliging

De beveiliging van de database wordt gerealiseerd op twee manieren. Aan de ene kant is er de beveiliging tegen doelbewust hacken (zoals SQL injectie). Aan de andere kant kunnen gebruikers verschillende permissies hebben. Als deze permissies niet goed gekoppeld worden aan de gebruikers, dan kan er mogelijk data gelekt worden.

Een oplossing, om gebruikersdata toch enigszins veilig te stellen, is om wachtwoorden te hashen. Het hashen van wachtwoorden wordt opgevangen in de front-end van de applicatie. Laravel Authentication wordt standaard geleverd met *bcrypt.* Het voordeel van bcrypt is dat de developer zelf kan aangeven hoe sterk een wachtwoord gehashed moet worden. Ook wordt *salt* gebruikt. Dit zorgt ervoor dat de kans dat het wachtwoord gekraakt zal worden minimaal is.

## 6.4. Performance

Dit is nog niet aan de orde gekomen.

### 6.4.1. Indexen

Dit is nog niet aan de orde gekomen.

## 6.5 Stored Procedures

In onderstaande tabel worden de Stored Procedures weergegeven, welke functie ze hebben en waar ze in het programma worden gebruikt.

|  |  |  |
| --- | --- | --- |
| **Naam** | **Functie** | **Gebruik** |
| spInsertUser | Het inserten van een user in de database. Het adres en het telefoonnummer wordt direct gecontroleerd of deze al bestaan. Als deze nog niet bestaan worden ze eerst toegevoegd, waarna ze als Foreign Key aan de gebruiker worden gekoppeld. | Webapplicatie registratieformulier. |
| spAddProject | Het toevoegen van een project. Hierin wordt de datum en tijd op het moment van opslaan automatisch ingevuld. Verder zal de x-coördinaat en y-coördinaat worden meegegeven, net zoals de gebruiker die het project toevoegt. | Webapplicatie. De Stored Procedure wordt aangeroepen op het moment dat de gebruiker een locatie heeft aangewezen en een omschrijving heeft toegevoegd. |

Tabel 2 Toelichting Stored Procedures.

## 6.6 Triggers

In onderstaande tabel worden de Triggers weergegeven, welke functie ze hebben en waar ze in het programma worden gebruikt.

|  |  |  |
| --- | --- | --- |
| **Naam** | **Functie** | **Gebruik** |
| trgBezwaarMakenVanaf18 | Een gebruiker mag wel een account hebben maar mag niet bezwaar maken op een project als zijn/haar leeftijd onder de 18 ligt. | Webapplicatie, maken van bezwaar op een project. |

Tabel 3 Toelichting Triggers.

## 6.7 Check constraints

In onderstaande tabel worden de Check constraints weergegeven, welke functie ze hebben en waar ze in het programma worden gebruikt.

|  |  |  |
| --- | --- | --- |
| **Naam** | **Functie** | **Gebruik** |
| CKC\_KVKNUMMER\_BEDRIJF | Het KVK nummer van een bedrijf mag niet langer dan 8 karakters lang zijn. | Webapplicatie, registratieformulier. |
| CKC\_GEBRUIKERSNAAM\_GEBRUIKE2 | De gebruikersnaam mag moet langer dan 4 characters zijn. | Webapplicatie registratieformulier. |
| CKC\_GEBOORTEDATUM\_GEBRUIKE | De gebruiker moet een minimale leeftijd van 18 jaar hebben. | Webapplicatie registratieformulier. |
| CKC\_GESLACHT\_GEBRUIKE | De gebruiker kan een man, vrouw of onbekend zijn. | Webapplicatie registratieformulier. |
| CKC\_TELEFOONNUMMER\_GEBRUIKE | Het telefoonnummer moet groter 8 characters zijn. | Webapplicatie registratieformulier. |
| CKC\_STATUS\_VERGUNNI2 | De vergunning moet verschillende statussen kunnen hebben, namelijk: Aangevraagd, Afgewezen, Uitgegeven, Verlopen, Bezwaar. | Webapplicatie aanvragen project, bekijken projecten.  Beheerapplicatie. |

Tabel Toelichting Check constraints.

# 7 Opsomming niet-functionele requirements

De requirements die niet kunnen worden onderverdeeld in een use case.

|  |  |
| --- | --- |
| **Code** | **Omschrijving** |
| **Usability** | |
| NFRU01 | Een ingelogde gebruiker moet met één muisklik zijn projectenoverzicht kunnen bekijken. |
|  |  |
| **Reliability** | |
| NFRR01 | Bij onverwachte systeemuitval moet het systeem binnen 5 minuten kunnen herstarten. |
| NFRR02 | Het systeem mag maximaal 24 uur per jaar onbeschikbaar zijn vanwege fouten in de geleverde software. Dit omvat **niet** de tijd die door de systeembeheerders nodig is om eventueel onderhoud uit te voeren. |
| **Performance** | |
| NFRP01 | De in de gebruikers- en beheerapplicatie gebruikte informatiebehoeften op de tabellen Project en Gebruiker mogen maximaal 1 seconde per 1.000.000 records duren. |
| **Supportability** | |
| NFRS01 | De webapplicatie moet ondersteund worden op de platformen Windows (7 en hoger), Linux (in ieder geval Ubuntu 16.04), Android (2.6 en hoger) en iOS (10.0 en hoger). |
| NFRS02 | De beheerapplicatie moet draaien op Windows 7 en hoger. |
| NFRS03 | De webapplicatie moet te gebruiken zijn vanuit de browsers Mozilla Firefox, Google Chrome, Microsoft Edge en Safari. |
| **Security** | |
| NFR+01 | Persoonlijke gegevens van gebruikers (met uitzondering van gebruikersnaam, voornaam, tussenvoegsel en achternaam), mogen alleen inzichtelijk zijn voor de gebruikers zelf, het systeembeheer en de bevoegde gezagen. |

Tabel 5 Overzicht niet-functionele requirements.

# Literatuurlijst

*Front End Frameworks Statistics*. (2017, Mei 18). Retrieved from keycdn: https://www.keycdn.com/blog/front-end-frameworks/#Front-End-Frameworks-Statistics