# 

Technisch Ontwerp

[**Database**](#_tqxwj09dbdlw) **2**

[**Front-end**](#_sp8vg6iphgna) **9**

[**Back-end**](#_fy9m4lqbkcb6) **10**

[**Netwerkstructuur**](#_i166erpm809c) **12**

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# Database

Er zijn veel verschillende databases om te gebruiken tijdens dit project.

Een paar databases die mogelijk zijn om te gebruiken zijn: MS SQL Server

In dit document zullen wij de afweging maken tussen MySql en Microsoft SQL. Waarom wij dit doen is omdat dit de 2 grootste, makkelijkst toegankelijke en best ondersteunde database types zijn. Maar om een ander perspectief te kunnen krijgen van de verschillende talen vergelijken wij MySql ook met PostgreSQL.

In dit document gaan wij ook de verschillen tussen een SQL en Nosql database laten zien.

**Verschillen tussen een SQL Database en een Nosql Database:**

* SQl gebruikt Tables, Nosql gebruikt Collections.
* SQL gebruikt Rows, Nosql gebruikt Documenten.
* SQL gebruikt Columns, Nosql gebruikt Fields.
* SQL heeft Joins, Nosql heeft Embedded Documents, Linking.
* Nosql is beter te hosten op meerdere servers dan SQL.
* SQL wordt langzamer als de databases groeien maar de hardware hetzelfde blijft.
* Nosql gebruikt geen query om data op te slaan en te zoeken en houdt zich niet aan strikte schema’s.
* Nosql is voor het handelen van grote hoeveelheden data.
* Als je geen grote hoeveelheden data hebt is SQL een betere optie.

Kort gezegd, een SQL Database gebruikt aangemaakte tables in de sql taal zelf, Nosql gebruikt documenten in plaats van tables. Een Nosql database wordt vooral gebruikt voor enorme hoeveelheden aan data omdat het makkelijk te verspreiden is over meerdere servers. En een SQl database voor kleinere hoeveelheden aan data, wat wij gaan gebruiken in dit project.

**Verschillen en overeenkomsten tussen een MySql database en een PostgreSQL database:**

* MySql ondersteund twee implementatie talen, C en C++.
* PostgreSQl ondersteund één taal, C.
* Beide database-engines zijn open source.
* MySql staat op de tweede plek waar PostgreSQl op plek vier staat.
* Beide engines worden op veel vlakken ondersteund en ondersteunen zelf veel.
* Beide database engines gebruiken een relationeel databasemodel.

**Server operating systems PostgreSQL:**

* FreeBSD
* HP-UX
* Linux
* NetBSD
* OpenBSD
* OS X
* Solaris
* Unix
* Windows

**API’s PostgreSQL:**

* Native C library
* ADO.NET
* JDBC
* ODBC

**Ondersteunde programmeertalen PostgreSQL:**

* .Net
* C
* C++
* Delphi
* Java
* Perl
* PHP
* Python
* Tcl

**API’s MySql:**

* ADO.NET
* JDBC
* ODBC

**Ondersteunde programmeertalen MySql:**

* Ada
* C
* C#
* C++
* D
* Delphi
* Eiffel
* Erlang
* Haskell
* Java
* JavaScript
* Objective-C
* OCaml
* Perl
* PHP
* Python
* Ruby
* Scheme
* Tcl

**Verschillen tussen Microsoft SQL en MySql:**

**Microsoft SQL en MySql van Oracle hebben een paar aanzienlijke verschillen.**

* MySql staat op de eerste plek en Microsoft SQL op de tweede.
* De implementatie taal die in Microsoft SQL wordt gebruikt is enkel C++ waarbij MySql gebruik maakt van C en C++.
* Server operating systems Microsoft SQL: Linux en Windows.
* Server operating systems MySql: FreeBSD, Linux, macOS, Solaris en Windows.
* License Microsoft SQL is commercial, MySql is Open Source.

**API’s Microsoft SQL:**

* OLE DB
* Tabular Data Stream(TDS)
* ADO.NET
* JDBC
* ODBC

**API’s MySql:**

* ADO.NET
* JDBC
* ODBC

**Ondersteunde programmeertalen Microsoft SQL:**

* C++
* Delphi
* Go
* Java
* Javascript
* PHP
* Python
* R
* Ruby
* Visual Basic

**Ondersteunde programmeertalen MySql:**

* Ada
* C
* C#
* C++
* D
* Delphi
* Eiffel
* Erlang
* Haskell
* Java
* JavaScript
* Objective-C
* OCaml
* Perl
* PHP
* Python
* Ruby
* Scheme
* Tcl

**Verschillen en overeenkomsten tussen MySQL en MongoDB:**

* MySQL is een SQL database, MongoDB is een Nosql database.
* MongoDB is over het algemeen sneller en makkelijker op te zetten.
* MySQL staat op de tweede plek in de wereld, MongoDB staat op de vijfde plek in de wereld.
* Beide Database-Engines zijn Open Source.
* MongoDB gebruikt geen data schema, MySQL wel.
* Geen van de twee Database-Engines zijn Cloud-based.

**API’s MongoDB:**

* Werkt vooral met JSON files.

**API’s MySql:**

* ADO.NET
* JDBC
* ODBC

**Ondersteunde programmeertalen MongoDB:**

* Actionscript
* C
* C#
* C++
* Clojure
* ColdFusion
* D
* Dart
* Delphi
* Erlang
* Go
* Groovy
* Haskell
* Java
* Javascript
* Lisp
* Lua
* MatLab
* Perl
* PHP
* PowerShell
* Prolog
* Python
* R
* Ruby
* Scala
* Smalltalk

**Server operating system MongoDB:**

* Linux
* OS X
* Solaris
* Windows

**Server operating system MySQL:**

* FreeBSD
* Linux
* OS X
* Solaris
* Windows

**Ondersteunde programmeertalen MySql:**

* Ada
* C
* C#
* C++
* D
* Delphi
* Eiffel
* Erlang
* Haskell
* Java
* JavaScript
* Objective-C
* OCaml
* Perl
* PHP
* Python
* Ruby
* Scheme
* Tcl

Wij kiezen voor MS SQL Server omdat het een makkelijker te leren taal is, het is makkelijk te hosten en goed te onderhouden, verder is het een meer toegankelijke taal.

# 

# 

# Front-end

We gaan gebruik maken van **HTML**, **JavaScript** en **CSS** omdat:

* Dit de industry-standard is.
* Op school hebben we ook met JavaScript en CSS gewerkt, dus we hebben al ervaring ermee.
* Het heeft de grootste community, dus is het heel makkelijk om antwoorden op je vragen te krijgen.
* Zeer handig voor later omdat bijna elk bedrijf HTML, Javascript en CSS gebruikt.

**Waarom preprocessors CSS:**

* Nettere code.
* Code is herbruikbaar.
* Beter georganiseerd.
* Kleinere bestandsgrootte, dus snellere laadtijd voor de website.

**Waarom gaan we SASS gebruiken en geen LESS:**

* Betere support bij SASS, zoals bootstrap compatible.
* SASS is efficiënter dan LESS.
* SASS heeft de grootste community, dus makkelijker om antwoorden op je vragen te krijgen.
* Aan SASS hebben we meer in de toekomst.

**Dit zijn een paar frameworks voor Javascript.**

**Angular 2+**

* Framework met volledige eigenschappen.
* Heel erg populair.
* Angular CLI, wat ervoor zorgt dat het heel makkelijk wordt gemaakt om een applicatie te maken dat meteen werkt. Je kunt er ook de applicatie mee testen.
* Typescript, dat zorgt voor automatisch aanvullen, geavanceerde navigatie en makkelijke herstructurering.
* Ionic framework, wat te gebruikt wanneer je mobiele applicaties wil maken, wat in ons geval geen toegevoegde waarde heeft.

**React**

* Lichtgewicht en efficiënt framework.
* Erg populair.
* CLI, net als Angular 2+.
* React native, wat je gebruikt voor mobiele applicaties, ook in dit geval hebben we daar niks aan.
* React Windows, wat je gebruikt voor Windows applicaties, hebben we in ons geval ook niks aan.

**Vue.Js**

* Heel simpel om te gebruiken.
* Erg flexibel en productief.
* Zeer snel.
* Vue-CLI, met dezelfde eigenschappen als de CLI van React en Angular 2+.

Vue.Js gaan we gebruiken omdat het heel simpel is om te leren en gebruiken, in tegenstelling tot react en angular 2+. Het is zeer snel en flexibel, en heeft net als de andere twee frameworks een CLI.

# Back-end

De talen waarin wij een keuze moeten gaan maken zijn Python en PHP.  
Dit omdat deze talen het meest gebruikt gaan worden in de toekomst. Wij vinden deze talen ook leuk en willen graag meer leren van deze talen. Nodejs of andere backend talen willen wij ook graag leren. Maar wij maken nu de keuze uit PHP en Python.

**Python en PHP overeenkomsten:**

* Makkelijk te leren.
* Beide hebben ze een Integrated development environment(debugger, automatiseringshulpmiddelen).
* Beide hebben een grote en vriendelijke community.

**Python en PHP verschillen:**

* Python's syntax is makkelijker om te leren omdat het veel overeenkomsten heeft met het normale engels.
* Python kan niet gemixt worden met html (heeft een template library nodig), terwijl PHP dat wel kan.
* Python is duur om te hosten en moeilijker te hosten.
* PHP heeft een grotere community dan Python.
* Lexical scoping van Python is rommelig in tegenstelling tot PHP.

De backend gaat bestaan uit PHP. Door bovenstaande punten is onze keuze op PHP gevallen. PHP wordt ook door vrijwel alles ondersteund. Het is een makkelijke taal om te leren en brengt ons veel kennis op voor in de toekomst.

**Waarom zou je een framework van PHP gebruiken?**

* Het programmeren wordt makkelijker en sneller door frameworks!
* Het is overzichtelijk.
* Ze zorgen voor betere beveiliging.

De 3 frameworks die wij uit hebben gekozen zijn : **Symfony, Laravel en Yii.**

**Symfony:**

* kan samen werk met Javascript framework “Angular”.
* 30 componenten om uit te kiezen.
* Kan gebruikt worden voor third-party.
* De componenten zijn reusable waardoor websites sneller zullen worden.
* Symfony heeft een grote community en is nog in bloei!
* Modulair.

**Laravel:**

* Laravel heeft de grootste community en is het populairst.
* Wordt veel gebruikt voor professionele websites.
* Uitstekende beveiliging.

**Yii:**

* Goede beveiliging.
* High-performance web framework.
* Yii is de snelste framework.
* jQuery integration ( makkelijk te combineren met html framework “ jquery”.
* Makkelijk uit te breiden.

Laravel is onze keuze omdat zij de grootste community hebben en er veel te leren valt bij laravel. Ook is de beveiliging erg belangrijk bij onze applicatie. Ook is er veel over te vinden op internet en zal deze framework ons meer informatie en kennis brengen voor in de toekomst

# Netwerkstructuur

**Server-side:**

* De talen die het meeste worden gebruikt I.C.M server-side zijn:
* PHP, Python, ASP.Net, C++, Visual Basic.
* Server-side wordt vooral gebruikt voor:
* Pagina’s weergeven en input van de gebruiker verwerken.

**Server-side rendering Voordelen:**

* Maakt de SEO beter ( Search Engine Optimization ) ( zoeken).
* Het eerste gedeelte laden van de pagina gaat sneller.
* Werkt goed voor statische sites.

**Server-side rendering Nadelen:**

* Vaak informatie opvragen ( wat tijd kan kosten).
* de gehele website laden duurt langer.
* Als je refresht dan moet hij opnieuw requesten.

**Client-side:**

* Javascript, HTML, CSS.
* Voor bijvoorbeeld interactieve webpagina’s, dynamische acties op de pagina en verzoeken sturen naar de server en data ophalen.

**Client-side rendering voordelen:**

* Na het eerste loading screen is de website sneller.
* Werkt goed voor webapplicaties.

**Client-side rendering nadelen:**

* Het eerste beeld renderen kan langer duren.
* SEO kan langer duren.

Een server-side API gebruiken we voor het renderen van de tabel met facturen. Dit is het enige dynamische onderdeel van de website. Om het overzicht van facturen apart te laden via de server-side, zal als voordeel hebben dat de gehele web-app sneller zal laden. Client-side gebruiken we voor de rest van de website.

**API in Laravel en Vue.js**

Een API kan gemaakt worden in Laravel, met Laravel 5.5 is het ook makkelijk te maken door de toevoeging van API resources. Laravel's resource classes maken het mogelijk om gemakkelijk je models en model collections te transformeren naar JSON.

Een API werkt ook met Vue.js. Je kunt request libraries gebruiken zoals jQuery, Axios of vue-resource.

**Verschillen tussen webhosting en server hosting**

**Bij webhosting horen de volgende voordelen:**

* Over het algemeen goedkoper
* Je kunt een veilige server opzetten
* Simpeler op te zetten
* Makkelijkere backups
* Makkelijker schaalbaar

**Bij server hosting horen deze:**

* Grotere opslagcapaciteit
* Krachtiger
* Ondersteund groter cache geheugen

Wij gaan voor webhosting, omdat webhosting goedkoper is en wij niet zo veel kracht en geheugen nodig hebben.

**Web server OS**

De twee grootste en belangrijkste OS’en zijn Windows en Linux. MySQL kan op beide draaien dus nu is de vraag welke voor ons het meest aantrekkelijk is.

Voor ons zou Linux het beste zijn. Dat komt omdat het betere uptime heeft en stabieler is, het veiliger is, het over het algemeen goedkoper is dan Windows en het een sterke community/support heeft. Windows is daarentegen alleen beter voor Windows-specifieke technologieën zoals .NET of een Microsoft SQL server. Daarbij loopt PHP makkelijker onder Linux.