# Getting started

## Afstudeerproject: Clustering zoekresultaten afkomstig van verschillende documentbeheerssystemen in een visuele omgeving

Hieronder vind je het volledige stappenplan om onze tool werkend en functioneel te krijgen. We hebben ook beschreven hoe je dit in de toekomst zou kunnen uitbreiden.

## Installatie

### Repository

Eerst en vooral beginnen we met het clonen van deze repo. Dit kan je doen via programma's zoals 'Github desktop' of 'GitKraken'.

Een andere manier om dit te doen is via command line (Windows PowerShell, Git Bash, CMD, ...). Je navigeert naar de locatie waar je de repo zou willen clonen adhv het commando 'cd'. Eens je in de juiste directory zit dan je volgend commando uitvoeren om je repo effectief te clonen.

git clone <https://github.com/stijnsaenen/Afstudeerproject.git>

### Toegang tot databank

Voor dit deel van het stappenplan hoef je direct niet iets specifieks te doen. In onze tool maken we gebruik van toegang tot de databank server van 'Van Havermaet'. Dit betekend dat je zonder problemen toegang tot deze databank als je onze tool gebruikt binnen in het netwerk van 'Van Havermaet'. Als je dus onze tool wilt gebruiken en je bent in het netwerk van 'Van Havermaet', mag je deze stap overslaan.

Als je van buiten het netwerk van 'Van Havermaet' toegang wilt, zal je een firewall exception moeten aanmaken om toegang te verkrijgen vanuit het netwerk waar je je dan in bevindt. In deze tutorial gebruiken we hiervoor [Microsoft SQL Server Management Studio](https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15).

Eens je dit programma gedownload en geïnstalleerd heb, moet je connectie maken met de databank adhv onderstaande credentials:

- Server type: Database Engine

- Server name: vhuat.database.windows.net

- Authentication: SQL Server Authentication

- Login: azureDB\_admin

- Password: P@55w0rd

Hierna verschijnt er een pop-up venster waarin je de bepaalde firewall exception kan aanmaken adhv je IP-adres of een subnet. Na deze stap moet je je aanmelden op de azure omgeving. Dit doe je met deze credentials:

- E-mail: [admin@vanhavermaetuat.onmicrosoft.com](mailto:admin@vanhavermaetuat.onmicrosoft.com)

- Password: re@dyf0r@cti0n!

Als dit gelukt is, heb je toegang verkregen tot de databank en kan je gebruik maken van onze tool.

### ETL tool (Extraction - Transformation - Load)

### Dit deel is alleen toepasselijk als je de applicatie wilt uitbreiden! NIET nodig voor te applicatie te runnen

We hebben een ETL tool gebruikt om verschillende data sources samen te brengen en te combineren in 1 tussenlaag. Dit is in ons geval de ucllDB databank op de databank server van 'Van Havermaet'. Als er in de toekomst nog data sources zouden bijkomen die moeten toegevoegd worden aan de tussenlaag, kan dit gedaan worden via een ETL tool. Wij hebben hiervoor Pentaho Data Integration Spoon gebruikt.

In deze tutorial zullen we ook opnieuw Pentaho gebruiken. Voordat je dit programma opstart moet je ervoor zorgen dat je de juiste database driver hebt geinstalleerd. Deze vind je [hier](https://docs.microsoft.com/en-us/sql/connect/jdbc/microsoft-jdbc-driver-for-sql-server?view=sql-server-ver15).

Eens je Pentaho heb opgestart en de files open die in de repo op locatie 'Database\ETL - Pentaho' staan, zie je de transformations die wij gebruikt hebben om alles toe te voegen in de tussenlaag. In file 'vhuat db to tussenlaag(uclldb)' vind je de transformatie waar wij alle gegevens van uit de bestaande databank van 'Van Havermaet' transfereren naar onze tussenlaag. Dit werkt heel simpel: elke input node (rood) dat je ziet, staat voor een tabel en is op zijn beurt gelinkt met een output node (groen) die staat voor een tabel op de tussenlaag.

Als er nieuwe data in de tussenlaag moet komen, die komt van de bestaande databank van 'Van Havermaet' kan je gewoon een configuratie kopieren en de gegevens aanpassen (kiezen van welke, naar welke tabel er moet worden getransfereerd).

De tweede file 'csv-to-db' is een transformatie die we gebruikt hebben om csv data, die we via een script van de active directory omgeving hebben gehaald, om te zetten daar een tabel en deze ook weer in te voegen in de tussenlaag. Net zoals de vorige, kan je deze gewoon kopiëren en al de info van de nodes aanpassen om het om te zetten naar de csv/tabel die je nodig hebt.

### Installatie Package

Het is belangrijk dat je de correcte python versie gebruikt. Versie 3.7.x is een vereiste, bij voorkeur gebruik je versie 3.7.6.

Deze tutorial (https://www.youtube.com/watch?v=bXWlyOMYpRE) kan je helpen om python correct te installeren.

Bij versie selecteer je 3.7.6, volg de tutorial tot 7:45.

Je kan de installatie controleren met:

python --version

### Installatie packages

Vervolgens installeren we de benodigde packages. Om dit makkelijker te maken is er een requirements.txt file in je repo, waarin alle packages opleglijst staan.

Navigeer in cmd naar de /Flask map in je repo.

Je kan de packages installeren met het commando:

pip install -r requirements.txt

Het is belangrijk dat requirements.txt in dezlefde map staan als de map waarin je het commando uit voert.

### Serve de applicatie lokaal

Flask, het package die we gebruiken voor de backend, heeft een ingebouwde lokale webserver, je kan deze runnen in de /Flask map met het commando:

flask run

Indien dit niet werkt kan je simpelweg de app.py uitvoeren met het commando:

py app.py

of

python app.py

Tip: als je aanpassingen wil maken in de backend is het handig om Flask debug mode te activeren, dan herstart de server automatisch bij elke aanpassing:

SET FLASK\_DEBUG = 1

### Overzicht file structure

Naast app.py, die de backend bevat, toont de volgende boom de frontend files. De overige files, die hier niet getoond worden, bevatten voornamelijk libraries die in deze files worden gebruikt.

Flask

│

│

├───static

│ │

│ │

│ ├───js

│ ├───network.js

│ └───searchbar.js

│

│

└───templates

|───index.html

network.js is de kern van de frontend en bevat de code de het netwerdiagram construeert en alles er rond.

searchbar.js is de code met betrekking tot de zoekalk en de autocomplete.

Deze bestanden moeten verplicht in de /static map.

index.html is het hoofd html bestand, dit is het bestand dat allereerst wordt ingeladen als je de pagina bezoekt.

De bovenstaande js files zorgen voor de invulling ervan.

html files moeten verplicht in de templates map.

### Docker

In dit deel beschrijven we hoe we via Docker ervoor zorgen dat we de applicatie kunnen installeren en runnen via localhost:5000. Voor Docker heb je een linux machine nodig, je kan dit ook doen via een virtuele linux machine op je Windows computer.

1. Navigeer in de repo naar de Flask folder

2. Volg [deze tutorial](https://docs.docker.com/install/linux/docker-ce/ubuntu/#install-using-the-repository) om docker te installeren in Linux. Je moet enkel het deel "Install using the repository" volgen en uitvoeren.

3. Voer de volgende commando's uit

sudo usermod -aG docker $UsernameVanRootGebruiker

docker build -t flask:latest .

docker run -d -p 5000:5000 flask:latest

docker ps

4. Dan zou je iets zien zoals dit:

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS

9701 flask python app.py 3 min ago Up 4 min 0.0.0.0:5000

Dit betekend dat de applicatie succesvol draait op de localhost:5000 (Als je hiernaar surft zou je de applicatie moeten zien).