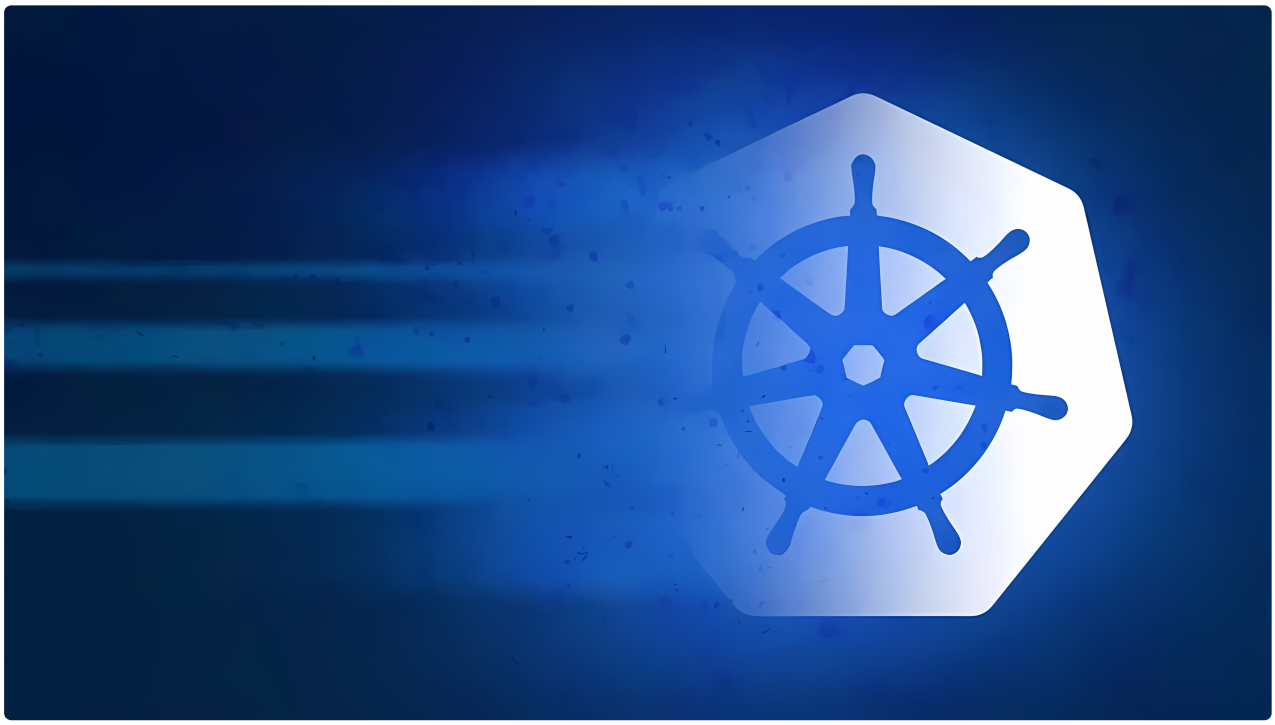
# AquaBrain Infrastructuur



**Student Naam**: Jasper Demmers
**Student Nummer**: 529633
**Student Klas**: PB-DB02
**Vak Docent**: Marco van Lee

# Inhoudsopgaven

# 1. Inleiding

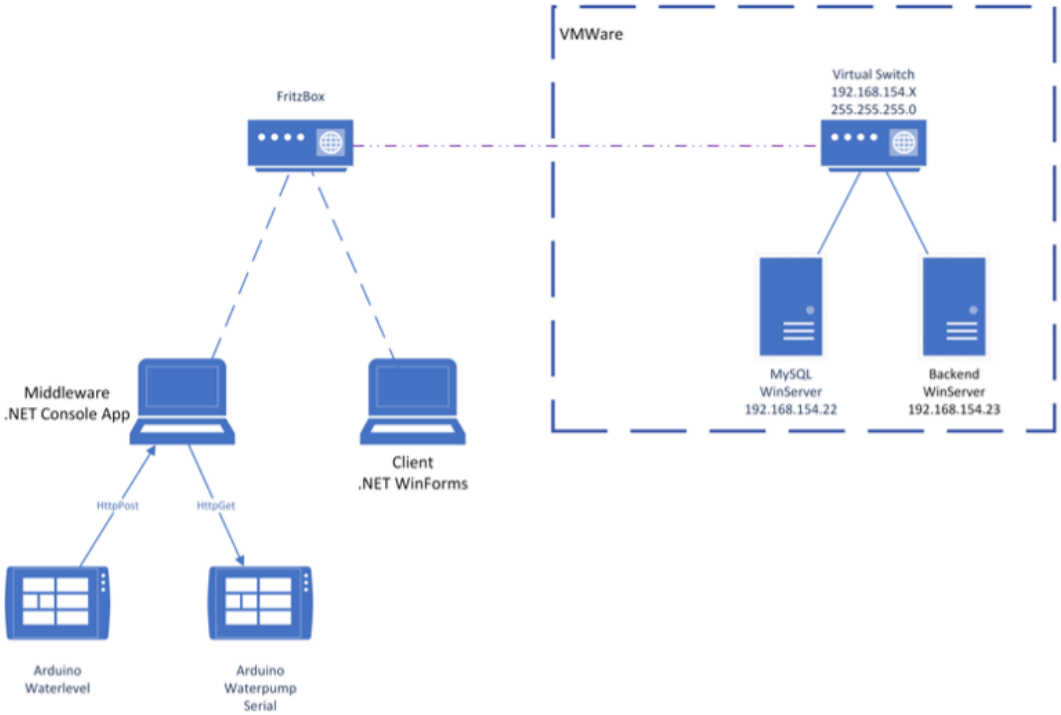
Dit verslag beschrijft de stappen die zijn genomen om onze bestaande IT-infrastructuur voor de hosting van onze API en Database te verbeteren. De focus ligt op het documenteren van wat is gedaan, waarom het is gedaan en hoe de verandering de schaalbaarheid, redundantie en beveiliging van onze systemen verbeteren.

## 1.1 Beschrijving van het Probleem

Ons huidige systeem bestaat uit twee Virtual Machines (VM's), beide draaien Windows Server. De ene host de API en de andere de Database. Dit leidt tot een aantal beperkingen en problemen, waaronder:

- **Beperkte schaalbaarheid**: Ons huidige systeem is niet ontworpen om gemakkelijk te schalen om aan toenemende vraag te voldoen.
- **Geen redundantie**: We hebben geen back-up of uitwijkmogelijkheden in geval van systeemuitval, waardoor we kwetsbaar zijn voor storingen.
- **Onvoldoende beveiliging**: Het huidige systeem heeft minimale beveiligingsmaatregelen, waardoor we blootstaan aan potentiële veiligheidsrisico's

Huidig netwerk situatie:



## 1.2 Overzicht van de Voorgestelde Oplossing

De voorgestelde oplossing voor deze problemen omvat een reeks technologische en procedurele veranderingen die gericht zijn op het verbeteren van de veerkracht, schaalbaarheid en beveiliging van onze systemen. Deze veranderingen omvatten:

- Het gebruik van een DMZ-account om ons netwerk met het internet te verbinden, met een openbaar IP-adres.
- Netwerkbeveiliging gehandhaafd door een router die zorgt voor een eerste verdedigingslinie tegen cyber dreigingen.
- Implementatie van DHCP en DNS servers voor betere IP-adresbeheer en naamresolutie binnen ons netwerk.
- Het opbouwen van een Kubernetes Cluster voor het orkestreren van onze containerized applicaties, waarmee we de schaalbaarheid en betrouwbaarheid van onze services kunnen verbeteren.
  In de volgende sectie van dit verslag zullen we deze componenten en processen in meer detail verkennen, beginnend met de analyse en dan verder naar het ontwerp en de realisatie van de oplossing.**

# 2. Analyse

In dit hoofdstuk worden verschillende analysemethodes toegepast om de requirements op te stellen. Hieruit werd een ontwerp netwerktekening gemaakt.

## 2.1 SWOT-analyse

De SWOT-analyse bestaat uit 4 delen. Strengths, Weaknesses, Opportunities en Threats. De SWOT-analyse heeft de focus op het product.

**Strengths**:

- **Innovatie**: Onze slimme watertonnen bieden een unieke, technologisch geavanceerde oplossing voor waterbeheer.
- **Connectiviteit**: Met de mogelijkheid om met de backend te communiceren, kunnen gebruikers de status en het gebruik van hun watertonnen op afstand volgen en beheren.

**Weaknesses**:

- **Connectiviteitsbeperkingen**: Afhankelijk van de internetverbinding van de klant kunnen er problemen zijn met de realtime functionaliteit van de producten.
- **Technische Kennis**: De producten kunnen te technisch zijn voor sommige gebruikers om volledig te benutten of zelf te installeren en onderhouden.

**Opportunities**:

- **Groene Technologie**: In een steeds milieubewuster wordende wereld, kunnen slimme watertonnen bijdragen aan duurzaam watergebruik.
- **IoT Groeimarkt**: De vraag naar internet of things (IoT) apparaten neemt toe en dit kan leiden tot een uitbreiding van de markt voor slimme watertonnen.

**Threats**:

- **Competitie**: Hoewel het product mogelijk uniek is, kunnen andere bedrijven vergelijkbare of betere slimme watertonnen ontwikkelen en op de markt brengen.
- **Privacy en Beveiliging**: Concerns over de verzameling en opslag van gebruikersinformatie en de beveiliging van de bijbehorende datatransmissies kunnen het vertrouwen van potentiële klanten verminderen.

## 2.2 Requirements

Met behulp van 2.1 SWOT-analyse, zijn er requirements opgesteld.

#### Functionele Eisen

1. **Hoge beschikbaarheid**: Het nieuwe systeem moet zijn ontworpen voor *hoge beschikbaarheid* en er moet een *failover plan* zijn ingebouwd om *downtime tot een minimum te beperken*.
2. **Automatisering van taken**: De werklast van het beheer moet worden verminderd door middel van automatisering tools dat repetitieve processen kunnen uitvoeren.

#### Niet-functionele Eisen

1. **Schaalbaarheid**: Het systeem moet *schaalbaar* zijn om aan *toekomstige vraag* te kunnen voldoen.
2. **Beveiliging**: Het systeem moet zijn ontworpen voor beveiliging met real-time bescherming tegen bedreigingen, gereguleerde toegangscontrole, en geregeld bijwerken van veiligheidsmaatregelen.
3. **Compatibiliteit**: Het nieuwe systeem moet compatibel zijn met bestaande technologieën en flexibel genoeg om toekomstige technologische innovaties te kunnen accommoderen.

#### Wensen

1. **Containerization**: Bij voorkeur maakt het nieuwe systeem gebruik van containerization technieken zoals Docker of Kubernetes om schaalbaarheid en onderhoudbaarheid te verbeteren. (Kubernetes is de Enterprise standaard)
2. **Beheerbaarheid**: Het nieuwe systeem moet gemakkelijk te beheren en te onderhouden zijn.
3. **Toekomstige integratie van aanvullende services:** Het systeem zou de mogelijkheid moeten hebben om in de toekomst extra services te integreren, indien nodig. Denk aan een website
   Deze eisen en wensen zullen als basis dienen voor het ontwerpen en implementeren van de nieuwe IT-infrastructuur.

## 2.3 GAP-analyse

De GAP-analyse analyseert welke stappen er genomen moeten worden om van de huidige situatie naar de gewenste situatie te komen.

**Huidige Situatie**

- Twee afzonderlijke Windows Server VM's, één voor API met IIS en één voor MySQL-database.
- Geen duidelijk gesegmenteerde subnetten.
- Geen containerization of orkestratieoplossing.
- Geen failover plan.
- Beperkte automatisering van beheerprocessen.

**Gewenste Situatie**

- Beveiligd netwerk met openbaar IP gehost achter een router.
- Gesegmenteerde netwerken, verhoogd beveiliging, betere foutisolatie en verbeterde prestaties.
- Containerization met behulp van Kubernetes, Kubernetes is het Enterprise standaard voor container orkestratie. Containerization maakt schaalbaarheid gemakkelijk
- Failover plan in plaats voor hoge beschikbaarheid.
- Beheer met behulp van een tool die automatisering van repetitieve processen mogelijk maakt, gemakkelijk beheer.

**De Gaps**

1. **Netwerkarchitectuur**: In de huidige staat is er geen duidelijke netwerksegmentatie. Dit moet worden aangepakt om het systeem in lijn te brengen met de gesegmenteerde structuur beschreven in de gewenste staat.
2. **Beheer van Containers en Orkestratie**: De site maakt momenteel geen gebruik van containerization- en orkestratietechnologieën zoals die in de gewenste staat worden beschreven met Kubernetes.
3. **Failover plan**: Er is nog geen failover plan aanwezig. Dit moet worden ontwikkeld en geïmplementeerd.
4. **Automatisering**: De huidige staat bevat beperkte automatisering van beheerprocessen. Er moet een tool worden geïmplementeerd om deze processen te automatiseren.
5. **Beveiliging**: Hoewel de systemen momenteel beveiligingsmaatregelen bevatten, zullen deze moeten worden herzien en aangescherpt om te voldoen aan de hogere beveiligingseisen in de gewenste staat.
   Deze GAP-analyse geeft ons een duidelijk pad om te volgen bij het verplaatsen van de huidige naar de gewenste staat. Elk van de gaps vertegenwoordigt een kans voor verbetering en innovatie in ons systeem.

## 2.4 MoSCoW-analyse

De MoSCoW analyse helpt bij het stellen van prioriteiten. Wat moeten we hebben, wat zouden we moeten hebben, wat kunnen we hebben en wat hebben we niet.

**Must Have**:

- Een beveiligd netwerk met openbaar IP gehost achter een router.
- Implementatie van subnetten voor de verschillende servers en Kubernetes-componenten.
- Opsporen en versterken van beveiligingsmaatregelen binnen het volledige systeem.

**Should Have**:

- Implementatie van Kubernetes voor containerization en orkestratie om de schaalbaarheid en het beheer te verbeteren.
- Ontwikkelen en implementeren van een failover plan om downtime te minimaliseren.
- Implementatie van een automatiseringstool voor repetitieve taken om systeembeheer te vereenvoudigen.

**Could Have**:

- Geavanceerde beveiligingsmaatregelen zoals intrusie detectiesystemen of verbeterde data-encryptie.
- Implementatie van geavanceerde monitoringtools voor diepgaande inzichten in systeemprestaties.

**Won't Have**:

- Serverless Architecture: Hoewel serverless-architecturen de potentie hebben om flexibiliteit en kostenbeheersing te verbeteren, zal het in deze fase van de doorontwikkeling niet worden meegenomen.

Met deze MoSCoW-analyse krijgen we een duidelijk begrip van de implementatieprioriteiten voor onze geplande IT-infrastructuur. Dit zal ons helpen bij de planning en ontwikkeling van de nieuwbouw en eventuele toekomstige verbeteringen.

# 3. Ontwerp

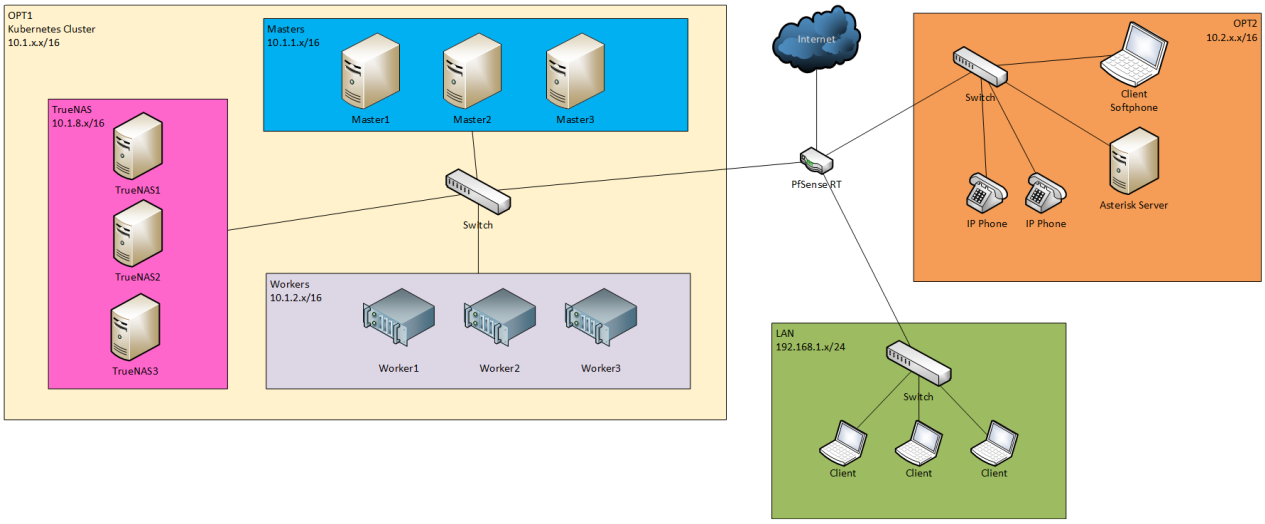
Onderstaande afbeelding zal ons ontwerp zijn. In OPT1 bevindt zich een high availability Kubernetes (K8s) cluster. Hier zal de MySQL database, API en eventueel een website plaatsvinden. Om ervoor te zorgen dat de cluster highly available is, zullen er minimaal 3 control-plane (master) nodes aanwezig zijn en minimaal 3 worker nodes. 2 control-plane nodes is ook voldoende, maar oneven getallen zijn gemakkelijk met bepalen van de leader bij een machine of zone failure (Bron: Creating Highly Available Clusters with kubeadm). Voor de K8s cluster is ook opslag nodig. Hiervoor wordt een Network File Server (NFS) gebruikt. Deze wordt opgesteld door middel van TrueNAS.

**OPT1**:
Het subnet van OPT1 is 10.1.0.1/16. Dit betekent dat de IP-adressen tot 10.1.255.255 beschikbaar zijn. Dat zijn maar liefst 65536 adressen. Dit hebben we verder gesegmenteerd. De DHCP range is aangepast om IP-adressen uit te delen van 10.1.0.10 tot 10.1.0.245. Vervolgens zitten alle control-plane nodes statisch op 10.1.1.X, de worker nodes op 10.1.2.X en TrueNAS op 10.1.8.X. Later in dit verslag komt er een stukje over MetalLB. MetalLB zal IP-adressen uit delen van 10.1.4.10 tot 10.1.4.245.

**OPT2**:
Het subnet van OPT2 is 10.2.0.1/16.

> WIP

**LAN**:
Het subnet van LAN is 192.168.1.1/24. Hier zullen clients plaatsvinden. De DHCP range is van 192.168.1.10 tot 192.168.1.245.



# 4. Kubernetes (K8s)

> WIP