THOMAS MORE

Internship Summary - Arxus

Internship Summary Document

Sepp Eyckmans
Student Bachelor in de Elektronica-ICT – Cloud & Cyber Security

2025 – 2026

Inhoudsopgave

1. Inleiding

Dit document geeft een concreet overzicht van mijn volledige stagetraject bij Arxus. Het bevat een samenvatting van alle documenten die werden opgesteld voor zowel mijn stageproject als mijn stageperiode bij Arxus.

Eerst wordt de Project Charter besproken, met daarin de projectomschrijving, projectdoelstellingen en een korte toelichting van de succescriteria. Vervolgens komt het Realisatie Document aan bod, waarin wordt toegelicht wat er gerealiseerd is, inclusief architectuur- en ontwerpdiagram en alle grote onderdelen van het project. Deze onderdelen worden onderbouwd via deliverables zoals code snippets. Tot slot wordt het Reflectie Document overlopen, waarin ik reflecteer op zowel mijn stageproject als mijn algemene stage-ervaring bij Arxus.

Het doel van dit document is om snel en concreet een volledig overzicht te bieden van zowel mijn stage als mijn stageproject, zonder het volledige gedetailleerde document te hoeven lezen.

Indien er technische termen worden gebruikt die onduidelijk of onbekend zijn, kan men hiervoor terecht in de woordenlijst (Glossary) op het einde van dit document in bijlage. Wenst u meer informatie of bijkomende details, dan raad ik aan het volledige document te raadplegen. U kunt dit terugvinden op mijn portfolio via de volgende link:

https://seyckmans.be/

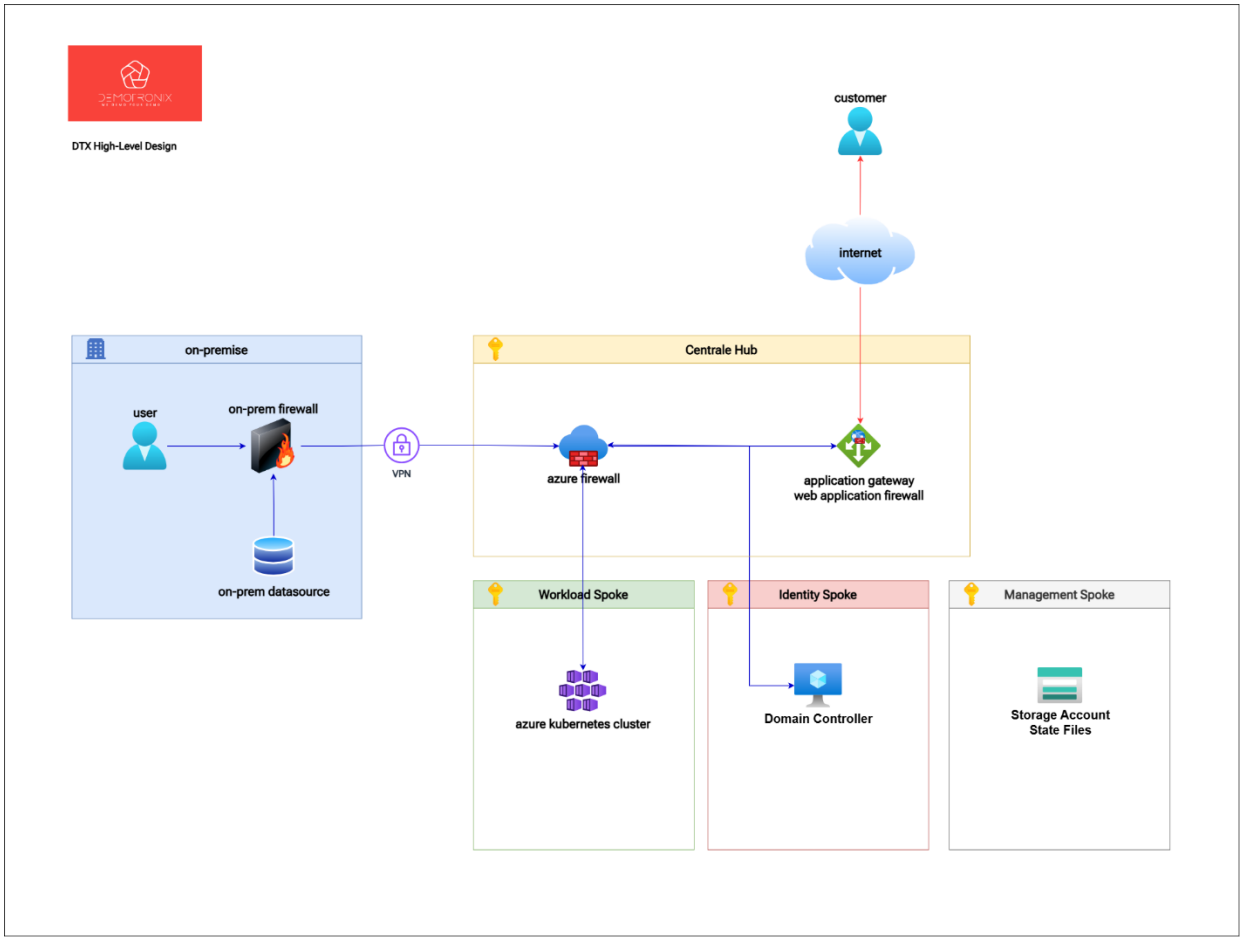
2. Probleemomschrijving

Arxus is een Cloud Service Provider en een Microsoft Azure Expert Managed Service Provider (MSP)-partner. Om klanten een consistente en betrouwbare service te bieden, zet Arxus sterk in op het standaardisatie van Azure Landing Zones, een volledige functionele cloud omgeving voor bedrijven, en op de automatisering van Azure deployments. Deze implementaties worden uitgevoerd volgens de richtlijnen van het Microsoft Cloud Adoption Framework (CAF), een Microsoft richtlijn voor het uitrollen van Landing Zones omgevingen.

Binnen deze context bestaat er voor de referentieklant “Demotronix” een standaard Azure Landing Zone architectuur. Het huidige probleem is dat deze architectuur voornamelijk gedefinieerd staat op conceptueel niveau en dient uitgewerkt te worden naar een geautomatiseerde infrastructuur implementatie. Bovendien zou deze omgeving on-demand moeten opgebouwd en afgebroken kunnen worden voor demonstratiedoeleinden aan klanten waarna de mogelijkheid bestaat om deze omgeving af te zetten om kostenbesparend te werken.

Het doel van dit project is om deze architectuur om te zetten naar Infrastructure as Code met behulp van Terraform, zodat de volledige uitrol van het infrastructuur omgeving geautomatiseerd wordt en in code gedefinieerd staat. Hiervoor wordt gebruikgemaakt van de interne Terraform module library van Arxus. De infrastructuur zal automatisch uitgerold worden naar een bestaande Azure demo tenant via GitHub Actions pipelines, dit laat toe om veranderingen in de omgeving continue en eenvoudig te implementeren en on-demand omgevingen te bouwen en af te breken.
(zie Glossary)

Het project heeft als doel een reproduceerbare en gestandaardiseerde implementatie van de Azure Landing Zone te realiseren volgens de principes en beste praktijken van het Microsoft Cloud Adoption Framework. De volledige details, waaronder de scope, worden verder toegelicht in het Project Charter document.
(Eyckmans, Project Charter, 2026)



Bijlage 1: High-level Infrastructuur Design Diagram

(Eyckmans, High-level Infrastructuur Design Diagram, 2026)

3. Doelstelling

Het doel van dit project is het ontwikkelen van een geautomatiseerde en gestandaardiseerde Azure omgeving die on-demand opgebouwd en afgebroken kan worden voor demonstratiedoeleinden aan klanten. Deze omgeving zal dienen als voorbeeldimplementatie voor klanten en zal de architectuurprincipes van een Azure Landing Zone, volgens het Microsoft CAF framework, volgen.

Dit SMART-doelstelling moet voldoen aan alle 5 vereisten, namelijk:

3.1. Specifiek

Het project heeft als doel een Azure Landing Zone omgeving te implementeren met behulp van Terraform en de interne Terraform module library van Arxus. De infrastructuur wordt uitgerold via GitHub Actions pipelines naar een Azure demo-tenant en volgt de beste praktijken van Microsoft.

3.2. Meetbaar

Het project wordt als succesvol beschouwd wanneer:

- De volledige Azure omgeving automatisch kan worden uitgerold via verschillende GitHub Actions workflows.
- De omgeving on-demand volledig opgebouwd en afgebroken kan worden via een automatisch proces.
- De implementatie maakt gebruik van de bestaande Arxus Terraform module library voor het uitrollen van alle Azure componenten.
- De Azure Landing Zone omgeving wordt volledig opgezet volgens de Microsoft-richtlijnen en volledig gebaseerd op de vooropgestelde architectuur design.

3.3. Acceptabel

De oplossing sluit aan bij de bestaande standaarden, tools en werkwijzen die binnen Arxus worden gebruikt voor Azure implementaties en automatisering.

3.4. Realistisch

Het project maakt gebruik van bestaande architectuurdocumentatie- en diagrammen, een al bestaande Terraform module library en een bestaande Azure demo-tenant. Waardoor de implementatie binnen de beschikbare tijd van de stage haalbaar is.

3.5. Tijdsgebonden

De implementatie, demonstratie en resolutie documentatie (thesis) van de geautomatiseerde Azure Landing Zone omgeving worden afgerond binnen de looptijd van de stage.

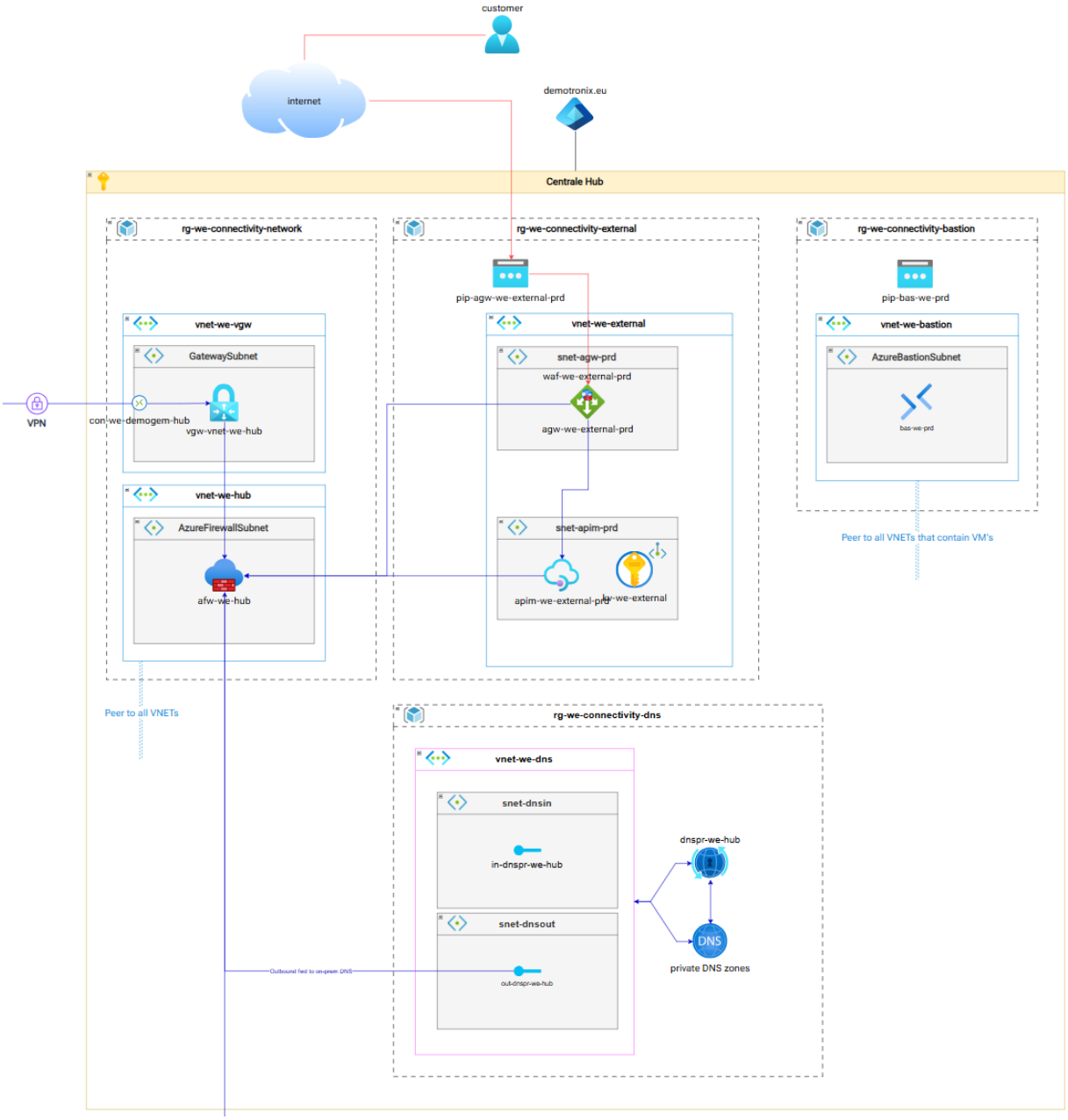
4. Succescriteria

Het project wordt als succesvol beschouwd wanneer de vooropgestelde doelstellingen gerealiseerd zijn en de opgeleverde oplossing voldoet aan de technische en functionele verwachtingen. De volgende criteria worden gebruikt om het succes van het project te evalueren.

- **Geautomatiseerde deployment van de omgeving**
- **On-demand opzetten en verwijderen van de omgeving**
- **Gebruik van gestandaardiseerde infrastructuurcode**
- **Werkende demo omgeving in de Azure demo-tenant**
- **Goedkeuring door de technische begeleiding**
- **Project oplevering binnen de deadline**

Meer details over deze criteria zijn terug te vinden in het Project Charter document. (Eyckmans, Project Charter, 2026)

5. Architectuur- en Ontwerpoverzicht

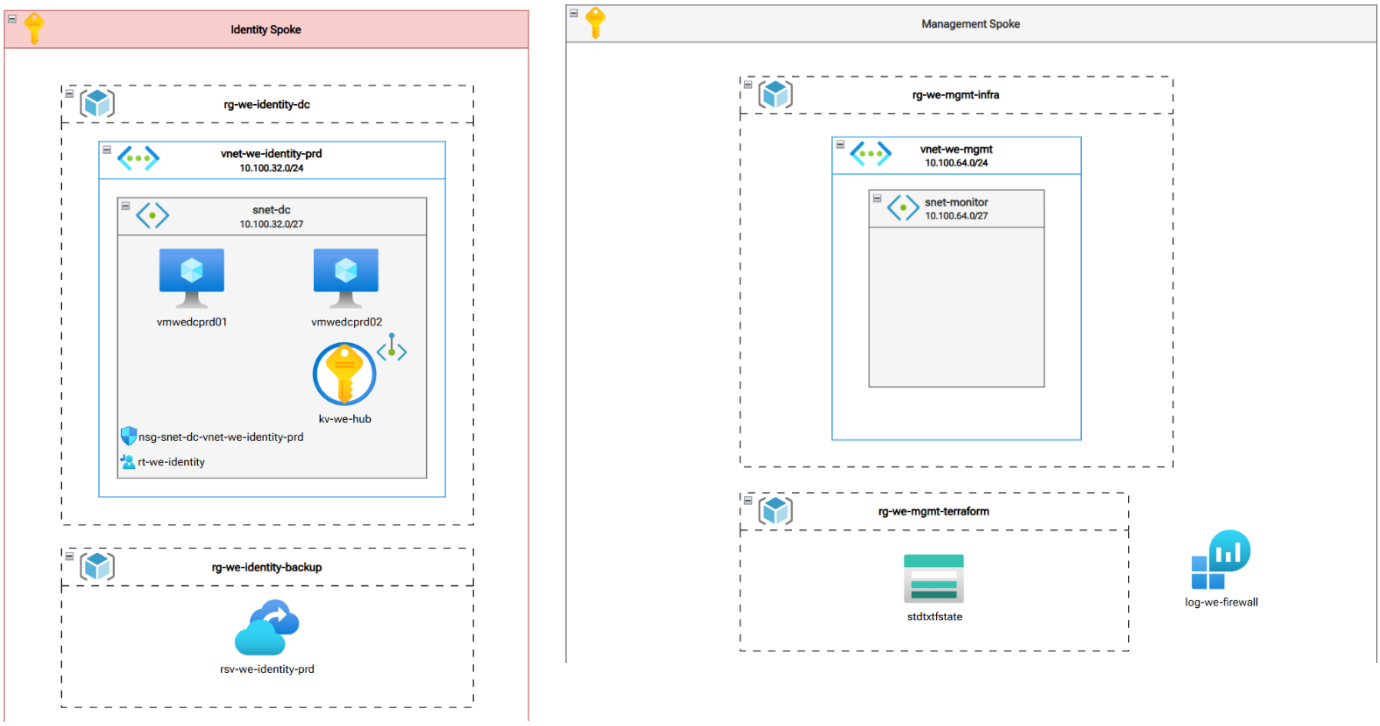


Bijlage 2: Hub Design

(Eyckmans, Realisatie Document, 2026)

Een standaard Azure Landing Zone volgt het Hub-en-Spoke topologie model voor zijn architectuurdesign. Dit zorgt voor een flexibel en modulaire structuur van de omgeving. Via deze structuur kan eenvoudig nieuwe onderdelen toegevoegd worden zonder grote architectuur veranderingen. Ieder stuk van de omgeving is opgesplitst in zijn eigen onderdeel, genaamd een spoke. Ieder spoke heeft zijn eigen individuele subscriptie waarin alle netwerk en infrastructuur onderdelen onder vallen.

De centrale hub beheert alle netwerkverkeer, interne connectiviteit en connectiviteit van buiten Azure naar binnen. In bijlage 2 zie je dat deze onderdelen bevat, zoals Application Gateway, Azure Firewall, API Management, Bastion en alle DNS-componenten. Bovendien bevindt het VPN Gateway oplossing voor connectiviteit tussen on-prem en Azure ook binnen de hub.



Bijlage 3: Identity + Management Spoke

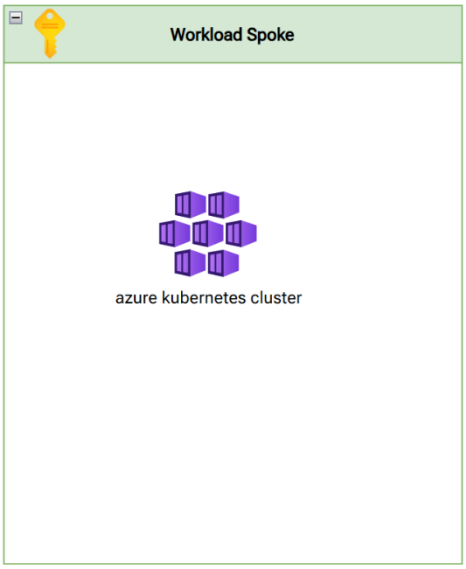
(Eyckmans, Realisatie Document, 2026)

Vervolgens hebben we de identity spoke, zoals te zien in bijlage 3. Deze spoke beheert alle componenten die dienen voor identiteit en authenticatie. Net als de Domain Controller-machines, die dienen als identiteits- en authenticatieoplossing binnen de omgeving, bevindt ook de Recovery Services Vault zich hier voor het back-uppen en herstellen van deze machines.

Daarnaast hebben we ook de management spoke, ook te zien in bijlage 3. Deze dient voor alle componenten die helpt bij het beheren en monitoren van deze Azure omgeving. Zoals de Storage Account voor de Terraform state files dat helpt bij het beheren en automatiseren van de omgeving. Hebben we ook de log onderdelen van de firewall en de fundering voor een monitoring system.

Tot slot hebben we de aks spoke. Dit bevat de aks cluster waarin alle workloads zullen draaien. Binnen de scope van dit project is het doel om in deze spoke enkel de basis te voorzien, zodat hier in de toekomst eenvoudig een cluster kan worden uitgerold.

Door tijdsbeperkingen binnen het project is het niet gelukt om een aparte AKS spoke te implementeren. Toch is het relevant om deze spoke te vermelden aangezien dit de start kan vormen van een mogelijk toekomstig project en een basis kan bieden voor verdere uitbreiding van de architectuur.

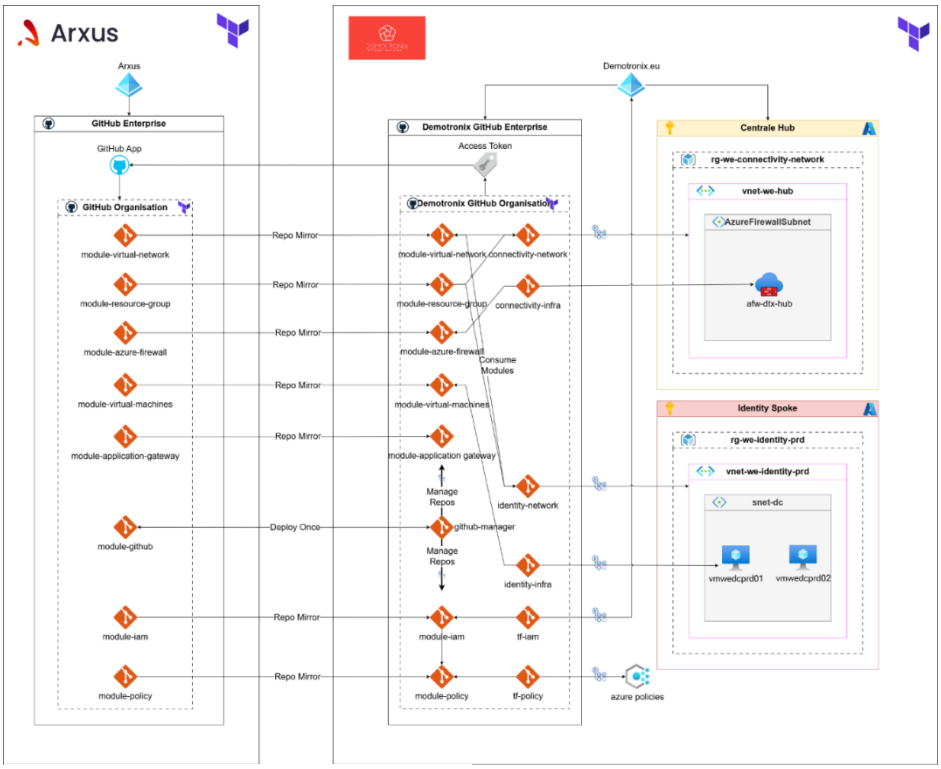


Bijlage 4: Workload Spoke

(Eyckmans, Realisatie Document, 2026)

Voor meer details over de architectuur, inclusief de functionaliteit van elk onderdeel binnen Demotronix, wordt verwezen naar het Realisatie document waar dit verder wordt toegelicht. (Eyckmans, Realisatie Document, 2026)

6. Terraform Module Library



Bijlage 5: Terraform Module Library & Sync
(Eyckmans, Realisatie Document, 2026)

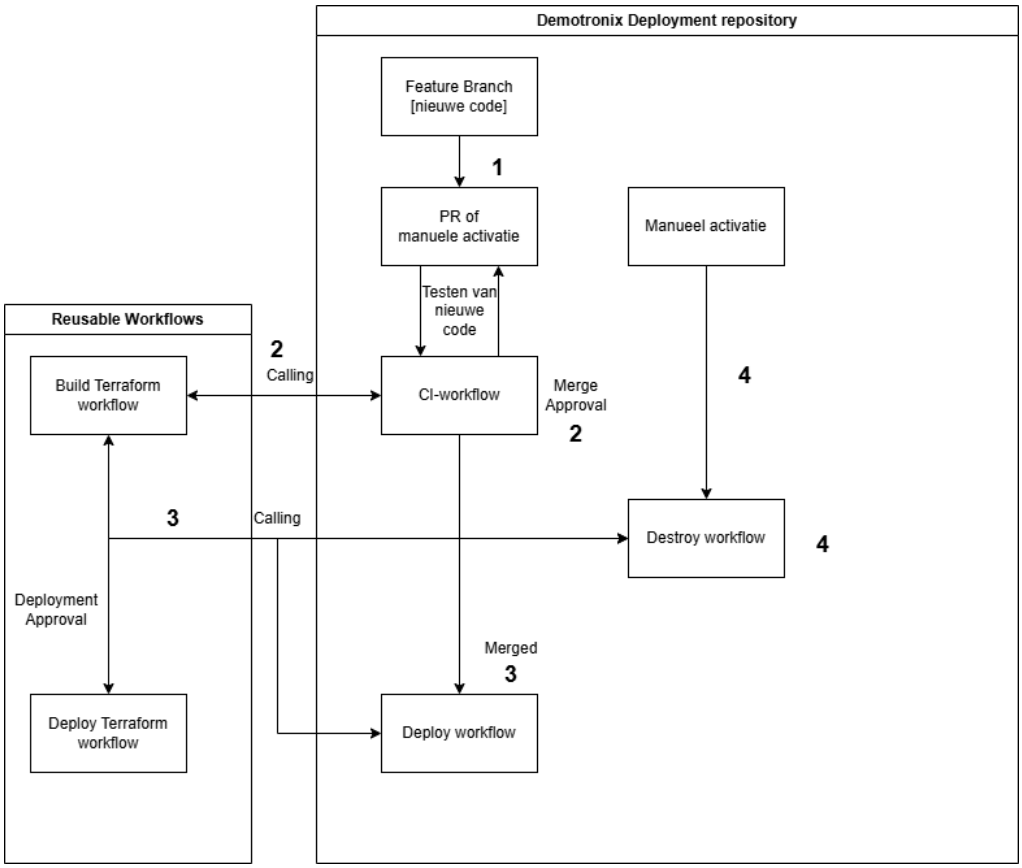
De Terraform Module Library is de interne en zelfgemaakte Terraform module collectie van Arxus. Iedere Azure service (Resource Group, Virtual Network, Firewall, DNS Private Resolver,…) heeft zijn eigen Terraform module om die service te beheren en uit te rollen met Terraform. Alle modules van alle Azure services zijn te vinden in deze library. Bovendien is deze library flexibel genoeg om infrastructuur uit te rollen in iedere Azure tenant naar keuze en iedere development omgeving binnen een tenant (dev, tst, acc, prd).

Deze library en zijn modules zullen gebruikt worden bij het uitrollen van alle infrastructuur binnen de Demotronix omgeving. Het gebruik van deze library en het schrijven van de code zal gebeuren volgens de standaarden voorgelegd door Arxus.

Bovendien worden deze modules beheerd en automatisch geüpdatet dankzij de module sync functionaliteit. Hiermee worden nieuwe versies, gemaakt door Arxus, doorgevoerd naar de module library van iedere klant dat een bezit. Bijlage 5 toont de structuur van de module sync bij de klant. Door de module sync kunnen nieuwe functionaliteiten toegevoegd worden, zonder dit manueel bij iedere klant te hoeven updaten. Als er bugs of fouten zijn, kan dankzij de module sync dit snel hersteld worden.

Voor een gedetailleerde toelichting op Terraform, de module library, module sync en hun onderlinge samenhang bij het opzetten en beheren van een Azure omgeving, verwijs ik naar het Realisatie document. (Eyckmans, Realisatie Document, 2026)
(zie Glossary)

7. GitHub Actions Workflows



Bijlage 6: Deployment Workflow Structuur
(Eyckmans, Realisatie Document, 2026)

GitHub Actions workflows zijn de automatische CI/CD workflows binnen elke Demotronix repository. Ze zorgen ervoor dat er veilig wordt ingelogd op de Demotronix Azure omgeving en voeren vervolgens Terraform code uit. Op die manier wordt de infrastructuur in Azure automatisch opgebouwd, aangepast of verwijderd.

Deze workflows zijn modulair en methodiek opgebouwd met behulp van calling workflows en reusable workflows. Dit zorgt voor een duidelijke structuur, een betere herbruikbaarheid van bestaande workflows en een optimalisatie van bestaande code door redundante code te vermijden. Bovendien zorgt dit voor een eenvoudiger beheer en onderhoud van de workflows. (zie Glossary)

Zoals eerder vermeld, zijn er binnen dit project 2 type workflows gebruikt:
- **Calling workflows**
- **Reusable workflows.**

Calling workflows zijn normale workflows die tijdens een bepaalde stap in hun workflow een reusable workflow kunnen oproepen. Deze workflows zijn meestal het startpunt waaraan een serie van reusable workflows aangekoppeld worden om zo grote en lange taken uit te voeren.

Terwijl reusable workflows herbruikbare workflows zijn die kunnen aangeroepen worden in elke calling workflows zonder dat je code moet copy-pasten van bestaande workflows. Dit vermijdt redundante code en maakt de workflows meer modulair. Alle reusable workflows zitten in hun eigen repository, die manueel bij de klant gezet wordt, waarna deze worden opgeroepen van elke deployment repository bij de klant.

Binnen Demotronix, heeft iedere deployment repo de volgende calling workflows:

- CI-workflow
- Deploy workflow
- Destroy workflow

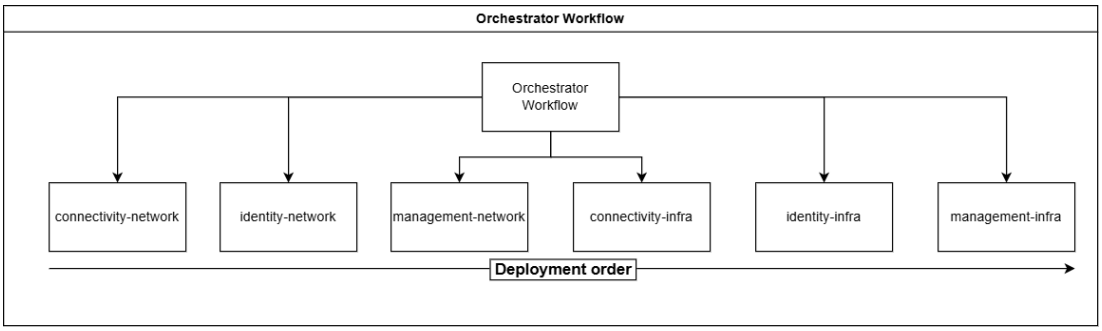
Op hun beurt, spreken ze een van of beide reusable workflows aan:

- Build Terraform workflow
- Deploy Terraform workflow

Deze workflows hangen nauw met elkaar samen en vormen één geheel. Hun gezamenlijke doel is het testen van nieuwe infrastructuur code door een infrastructuur plan te maken en, na goedkeuring, het uitrollen of verwijderen van de bijbehorende infrastructuur.
(zie Glossary)

Voor een meer gedetailleerde uitleg over de CI/CD workflows en hun onderlinge samenhang verwijs ik naar het Realisatie document.
(Eyckmans, Realisatie Document, 2026)

8. Orchestrator Workflow



(Eyckmans, Realisatie Document, 2026)

Bijlage 7: Orchestrator Workflow

Voor een deployment moet in elke deployment repository afzonderlijk een deploy workflow worden gestart. Bij een volledige deployment betekent dit dat je momenteel in zeven repositories manueel een workflow moet opstarten, bovendien in de correcte volgorde. Om dit proces te automatiseren en te stroomlijnen, werd de Orchestrator workflow ontwikkeld.

De Orchestrator workflow is een GitHub Actions workflow die de volledige infrastructuur van de Demotronix Azure omgeving kan uitrollen via de activatie van enkele 1 workflow. Deze start niet alleen automatisch alle nodige workflows, maar doet dit ook in de juiste deployment volgorde, zoals te zien in bijlage 7. Dit is ideaal voor ontwikkelaars die snel een demo omgeving willen opzetten bij klanten. Echter hoeft het niet alles of niks te zijn, met deze workflow kan je zelf kiezen welke deployment repositories je wilt aanspreken. Tot slot werkt dit voor zowel het opbouwen als het afbreken van infrastructuur.

Deze workflow functioneert door API-calls te maken naar de gewenste repositories via de GitHub API om het deploy workflow of destroy workflow te activeren. Maar dit is enkel het eerste deel, het methodieken deployment volgorde is een apart probleem De oplossing hiervoor is de “sort-repositories” action, die ik zelfgemaakt heb voor dit project.

Een Action workflow, ook wel Composite Actions genoemd in GitHub, is een herbruikbaar en modulair oplossing voor het uitvoeren van kleine, specifieke taken binnen een workflow. In tegenstelling tot een reusable workflow, die meerdere acties kan bevatten en als een volledige workflow functioneert, bundelt een Composite Action meerdere stappen tot een enkele actie. Deze actie kan vervolgens eenvoudig in andere workflows worden gebruikt.

Voor taken die vaak terugkomen en mogelijk ook in toekomstige workflows nodig zijn, is het aanmaken van een Composite Action ideaal. Het helpt om herhaling te vermijden, workflows overzichtelijk te houden en onderhoud eenvoudiger te maken.
(zie Glossary)

Voor extra details omtrent de functionaliteit en structuur van de orchestrator workflow, verwijs ik hier naar het Realisatie document.
(Eyckmans, Realisatie Document, 2026)

9. Inhoudelijke reflectie

Het hoofddoel van dit stageproject was het uitrollen van een geautomatiseerde, gestandaardiseerde en herbruikbare Azure Landing Zone die voor demonstraties en interne toepassingen kan gebruikt worden. Dit zou on-demand een Azure omgeving van nul kunnen opbouwen en afbreken. Dit allemaal volgens de beste praktijken en standaarden van zowel Arxus als het Microsoft CAF. De project deliverables bestaan uit infrastructuur en workflow code. Deze code is, via GitHub repositories, aan Arxus overhandigd op het einde van de stageperiode.

Voor Arxus is de toegevoegde waarde van dit project om hun automatisatie en Azure Landing Zone uitrol live te kunnen demonstreren aan potentiële klanten. Zo kunnen ze hun services beter tonen en verkopen. Maar ook heeft dit intern een potentieel nut. Doordat dit project gestandaardiseerde infrastructuur code heeft gemaakt voor een standaard Azure Landing Zone, zou dit intern kunnen gebruikt worden als template voor het uitrollen van Landing Zones. Zo moeten ontwikkelaars niet van nul beginnen maar hebben ze een fundamentele basis als uitgangspunt.

Gebaseerd op de succes criteria die zijn vastgelegd in het Project Charter, is er geëvalueerd dat het project succesvol is afgerond. Echter zijn er mogelijkheden voor uitbreiding van de demo omgeving via het toevoegen van extra spokes, een monitoringsysteem, extra workloads, etc. Doordat dit project modulair is gemaakt, zou toekomstige projecten eenvoudig geïntegreerd kunnen worden met de Demotronix Landing Zone. Beknopt gezegd is het project volledig voltooid. Na de hand-over van het project aan Arxus, zou dit onmiddellijk kunnen gebruikt worden voor klantendemo’s of als basis voor de ontwikkeling van omgevingen.

Tot slot volgt hier een overzicht van alle succesdoelstellingen zoals vastgelegd in de Project Charter, als bewijs van de succesvolle afronding van het stageproject:

- Een volledig geautomatiseerde uitrol van een Azure Landing Zone binnen een Azure demo-tenant.
  - Via Terraform en GitHub Actions
- On-demand bouwen en afbreken van volledige omgeving.
  - Hierdoor kunnen werknemers snel omgevingen opzetten voor demo presentaties.
  - Automatisch afzetten van deze omgeving na presentatie voor kosten te besparen.
- Volledige oplossing is gebouwd volgens Arxus standaarden en beste praktijken.
  - Met gestandaardiseerde infrastructuur code
  - Via Terraform module library van Arxus
- Volledige oplossing is gestandaardiseerd volgens Microsoft CAF.
- Binnen deze uitgerolde demo omgeving werkt alle functionaliteit.
- Zoals; netwerking, DNS, back-ups, Private Endpoints, App Gateway, Bastion, …
- Goedgekeurd door alle betrokken stakeholders.
  - Zowel op architecturaal als technisch vlak.
- Het volledige project is opgeleverd binnen de vastgelegde deadline van 22 Mei.

10. Persoonlijke reflectie

Bij aanvang van mijn stage had ik een bepaald beeld van hoe een professionele werkomgeving eruit zou zien en hoe mijn ervaring daarin zou zijn. Al snel bleek echter dat deze realiteit aanzienlijk verschilt van een schoolomgeving. Desondanks heeft Arxus het mij zeer gemakkelijk gemaakt om mijn inwerkperiode zo soepel mogelijk te laten verlopen. Ik heb het bedrijf leren kennen als een aangename, ondersteunende en professionele werkomgeving.

Tijdens mijn stage kwam ik echter al snel in contact met enkele leercurves. Zowel op technisch als op praktisch vlak. Technisch gezien lagen de grootste uitdagingen bij het begrijpen van de meer genuanceerde aspecten van de Terraform module library. Vervolgens was het oplossen van complexe problemen via trial-and-error bij het opzetten van nieuwe GitHub Actions workflows een sterke uitdaging. Verder ook het verwerven van diepgaande kennis van complexe Azure functionaliteiten was zeer belangrijk. Deze elementen vereisten niet alleen doorzettingsvermogen, maar ook analytisch denkvermogen en een gestructureerde aanpak om problemen efficiënt te identificeren en op te lossen. Dit zijn enkele nieuwe competenties die ik verworven heb.

Naast de technische aspecten waren er ook praktische en communicatieve uitdagingen. Het ging hierbij om eenvoudige, maar essentiële vaardigheden zoals het reserveren van bureaus en meeting rooms tot het inplannen van de meetings zelf. Deze ervaringen hebben mij geholpen om mijn organisatorische en communicatieve competenties verder te ontwikkelen en mij beter te integreren binnen een professionele context.

Op persoonlijk vlak heeft deze stage voor mij een grote meerwaarde betekend. Ik heb niet alleen mijn technische kennis aanzienlijk uitgebreid, maar ook geleerd om zelfstandig te werken, initiatief te nemen en out-of-the-box te denken. Mijn kennis van Azure, evenals mijn vaardigheden in automatisatie met Terraform en GitHub Actions, zijn sterk gegroeid sinds mijn start bij Arxus. Door weken van diepgaand onderzoek en analyse, gecombineerd met maanden aan praktische ervaring, heb ik de nodige competenties ontwikkeld om dit project succesvol af te ronden.

Conclusie

Kortom kan ik concluderen dat deze stage een zeer waardevolle en leerrijke ervaring is geweest. Het heeft mij niet alleen voorbereid op mijn toekomstige carrière, maar mij ook een duidelijker inzicht gegeven in mijn eigen sterktes en ontwikkelpunten. Deze ervaring vormt zonder twijfel een sterke basis voor mijn verdere professionele ontwikkeling. Ik ben zeer trots op het resultaat dat ik geleverd heb. Het feit dat mijn project door collega's van Arxus gebruikt kan worden voor demonstraties aan potentiële klanten, en tegelijk een basis vormt voor ontwikkelaars om nieuwe projecten op te starten, maakt het geheel minder conceptueel en meer een tastbare realisatie.

Als afsluiter zou ik graag Jan de Munck en Brent Boutmans oprecht bedanken voor hun voortdurende steun, begeleiding en vertrouwen tijdens mijn periode bij Arxus. Dankzij hun hulp, feedback en betrokkenheid heb ik veel kunnen bijleren en mijzelf verder kunnen ontplooien, zowel op professioneel als persoonlijk vlak.

Verder zou ik Cas Magnus graag willen bedanken voor zijn ondersteuning en waardevolle feedback tijdens mijn stageperiode. Zijn begeleiding en steun bij het opstellen van mijn project charter en het realiseren van mijn realisatiedocument hebben mij enorm geholpen om mijn documenten gestructureerd en doelgericht uit te werken. Zijn inzichten, advies en algemene ondersteuning hebben een belangrijke bijdrage geleverd om dit project en de gerelateerde documentatie tot een goed eind te brengen.

Tot slot zou ik iedereen bij Arxus willen bedanken voor de aangename werksfeer die ze voor mij hebben gecreëerd. De manier waarop ik als nieuwe stagiair meteen werd betrokken om deel uit te maken van het team, heb ik enorm geapprecieerd. Dankzij hun openheid, steun en vriendelijke aanpak voelde ik mij snel welkom binnen de groep. Dit heeft mijn stage niet alleen leerrijk gemaakt, maar ook zeer aangenaam op persoonlijk vlak.

Glossary

Term	Definitie
Azure demo-tenant	Het afgebakend stuk van Microsoft Azure waarin alle Clouddiensten, gebruikers en instellingen van een organisatie beheerd worden.
Azure Kubernetes Service (AKS)	Een managed service binnen Azure voor het implementeren, beheren en schalen van Kubernetes clusters. AKS vereenvoudigt het beheren van gecontaineriseerde applicaties in Azure zonder de onderliggende controle plane te onderhouden.
Azure Landing Zone	Gestructureerde en geoptimaliseerde Azure omgeving. Een omgeving ontworpen om zo veilig, flexibel en optimaal mogelijk te zijn. Kan modulair aangepast worden aan de behoeften van een organisatie dankzij het hub-spoke design.
Continuous Integration & Continuous Delivery (CI/CD)	Verschillende werkwijzen om het software ontwikkeling proces te automatiseren en te versnellen. Bij nieuwe code veranderingen in de code repository zou dit automatisch deze code testen, integreren met de bestaande code en het uitrollen van de infrastructuur. Dit laat toe om frequente veranderingen, reparaties en nieuwe implementaties snel door te voeren. Deze automatie implementatie wordt soms ook een deployment pipeline of gewoon een pipeline genoemd.
GitHub Actions pipeline	Het Continuous Integration & Delivery dienst van GitHub. Het volledige proces van code migreren, testen en uit te rollen gebeurt rechtstreeks vanuit de code repository in GitHub en gebeurt automatisch met minimale manuele interactie zodra er veranderingen zijn in de code.
GitHub Organisation	Een privé stuk in GitHub waarin alle code repositories van een organisatie kan bewaart en beheerd worden. Allemaal privé en geïsoleerd.
GitOps	Een werkwijze waarbij je infrastructuur en applicatie configuratie beheert via Git repositories, die als enige bron van waarheid dienen. Deployments gebeurt automatisch op basis van wijzigingen in Git.
Infrastructure as Code (IaC)	Het automatisch opzetten en beheren van IT-infrastructuur via code in plaats van handmatige interacties. Hierdoor krijg je consistente, herhaalbare en snel uitrolbare omgevingen.
Ingress	Een unieke Kubernetes onderdeel die inkomende HTTP/HTTPS verkeer van buiten de cluster naar interne services routeert via regels (zoals host- of pad gebaseerde routing). Meestal wordt dit beheert door een ingress controller.
Microsoft Azure Expert MSP partner	Het officiële Microsoft certificaat om als Cloud Solution Provider bedrijf samen te werken met Microsoft als partners. Dit certificaat bewijst dat dit bedrijf uitblinkt in het verkopen en beheren van Azure services bij klanten. Met deze partnership krijgt dit bedrijf verschillende voordelen zoals kortingen op Azure licenties & subscripties en prioriteit in Microsofts "referral engine", dat het bedrijf helpt om nieuwe klanten te kunnen krijgen. Maar een partner worden is een complex proces. Dit omvat verschillende audits en allerlei hoge verwachtingen. Dit certificaat moet ook jaarlijks hernieuwd worden.
Microsoft Cloud Adoption Framework (CAF)	Richtlijnen en best practices om bedrijven te helpen Azure Cloud te implementeren in hun huidige IT-omgeving.
Pods	De kleinste uitvoerbare eenheid in Kubernetes. Een Pod bevat een of verschillende containers die samen draaien, dezelfde netwerkconfiguratie delen en toegang hebben tot gedeelde opslag. Pods worden gebruikt om applicaties of onderdelen van applicaties te draaien in een Kubernetes cluster.
SKU (Stock Keeping Unit)	Een identificatiecode voor specifieke productvarianten binnen Azure te onderscheiden. Een SKU is een specifieke configuratie of versie van een Azure service die bepaalt welke functies, prestaties en prijs van toepassing zijn. Elke SKU komt overeen met een bepaalde capaciteit of service niveau binnen een Azure product. Zoals VM size, opslagtype of database tier.
Terraform	Terraform is een Infrastructure as Code tool waarmee infrastructuur automatisch wordt opgezet en beheert via code, grotendeels onafhankelijk van de Cloud provider.
Terraform module library van Arxus	De interne en zelfgemaakte Terraform module collectie. Deze collectie wordt gebruikt om de volledige infrastructuur omgeving in code te definiëren en te bouwen. Het gebruik van deze collectie standaardiseert het proces van Azure resource deployment via Terraform.
Weighted Ranking Method	De Weighted Ranking Method is een gestructureerde beslissingsmethode waarbij verschillende alternatieve systematisch met elkaar vergeleken wordt op basis van vooraf gelegde criteria. Elk criteria krijgt een bepaald gewicht gebaseerd op het belang ervan. Elk alternatief wordt per criteria gescoord, om zo uiteindelijk een totaalscore te berekenen. Hierdoor kan er constructief en systematisch keuzes onderbouwd worden.

Generatieve AI Policy

Tijdens het schrijven van deze thesis, is er generatieve AI gebruikt om het geschreven werk te verbeteren. Specifiek de Copilot tool van Arxus is gebruikt om mijn geschreven werk te verbeteren. Zoals controleren op grammatica, spelling, formaliteit en algemene zinsbouw. Alle prompts die gebruikt zijn bij het schrijven van deze stagedocumenten zijn allemaal in de aard van "Verbeter deze tekst op grammatica, spelling, formaliteit en algemene zinsbouw." geformuleerd. Waarna ik deze output zelf nog gecontroleerd heb op kwaliteit, accuraatheid en aangepast aan mijn persoonlijke voorkeur. Dit allemaal om tot de definitieve uitkomst te geraken.

AI is altijd voorzichtig gebruikt en met een korrel zout opgenomen, het is nooit blindelings geïmplementeerd in het document. Want AI is niet perfect, het maakt regelmatig fouten of geeft een zelfverzonnen antwoord zonder bron. Bovendien is het partijdig (Copilot heeft voorkeur naar Microsoft producten bijvoorbeeld) en beperkt tot bronnen die dateren tot en met 2025. Hierdoor is AI niet een definitieve bron van waarheid.

Referentielijst

Eyckmans, S. (2026). High-level Infrastructuur Design Diagram.
Eyckmans, S. (2026). *Project Charter.*
Eyckmans, S. (2026). *Realisatie Document.*