

**Connector-Module**

Jonathan Godeyne

Promotoren:

|  |  |
| --- | --- |
| Kris Vanreyten | Tobania |
| Marijke Willems | Hogeschool PXL Hasselt |



**Connector-Module**

Jonathan Godeyne

Promotoren:

|  |  |
| --- | --- |
| Kris Vanreyten | Tobania |
| Marijke Willems | Hogeschool PXL Hasselt |

Dankwoord

Om te beginnen wil ik mijn verloofde bedanken, Lana Das, zonder haar zou ik het nooit hebben aangedurfd om deze opleiding te volgen. Haar steun is hetgeen dat me altijd vooruit helpt.

Vervolgens zou ik ook alle lectoren willen bedanken voor hun kennis en het vermogen deze kennis te verspreiden met de studenten. In het bijzonder zou ik ook Marijke Willems willen bedanken die elke week paraat stond om mijn documenten te verbeteren en advies te geven. Ook zou ik Veerle Asaert willen bedanken voor haar positieve boodschappen tijdens moeilijkere periodes.

Ik zou ook mijn stagebedrijf Tobania willen bedanken, bedankt voor de hele fijne samenwerking. Ik kan niet iedereen persoonlijk bedanken want ik heb hulp van verschillende mensen gekregen in het team. Toch zou ik Kris Vanreyten en Niki Dubois willen bedanken voor de steun als bedrijfspromotor en in het speciaal Mohamed Walid Soumer voor de dagelijkse meetings en de vele kleine hulp sessies.

Ten slotte wil ik ook nog Ruud Boyden van Acerta bedanken die ondanks zijn druk schema toch af en toe tijd wou vrijmaken om up-to-date te blijven van de stage.

Abstract

Tobania is een groot Belgisch consultancy-bedrijf binnen de ICT- en businesswereld. Ze bieden hun kennis en advies aan voor bedrijven en overheden en hebben meer dan duizend *Tobians* met verschillende profielen tewerkgesteld om te voldoen aan de verwachtingen van een klant. Tobania biedt een brede waaier aan oplossingen aan zijn klanten, waaronder Business, Development, Quality & Support en Data. Deze stageopdracht bevind zich in het gedeelte Development.

Voor deze stageopdracht wordt er gewerkt voor een klant, namelijk Acerta. Acerta is een HR-firma die met een softwarepakket “Arno” werkt. Arno is een krachtige applicatie maar ze is ondertussen verouderd in het huidige digitale landschap. Daarom is Tobania bezig met een nieuwe laag code over Arno te maken omdat zodat de oude functies niet verloren gaan maar er toch nieuwe technologieën gebruikt kunnen worden. Binnen de HR-wereld werken de meeste bedrijven met hun eigen HR-softwarepakketten. Sommige van deze bedrijven stellen functies open die gebruikt kunnen worden door andere HR-bedrijven; dit gebeurt door middel van een API. Hiervoor wordt eerst een overeenkomt opgesteld zodat enkel de bedrijven die geabonneerd zijn op deze API’s ermee kunnen verbinden. Mpleo is zo’n bedrijf waarmee Acerta een overeenkomst heeft. Mpleo deelt zijn data met Acerta.

Voor deze stageopdracht wordt er een *Connector Module* gemaakt in een .NET-omgeving die de data van de Mpleo API synchroniseert met de data van Arno. Er zal ook een *mapping* gebeuren van de data want deze zijn niet standaard compatibel met elkaar. De data van de Mpleo API zal Azure API Management als tunnel gebruiken zodat dit proces beter te onderhouden wordt in de toekomst wanneer er gewerkt wordt met meerdere API’s. Vorig jaar is er als stageopdracht een aantrekkelijke en gebruiksvriendelijke WPF applicatie gemaakt die het gebruik van Arno vereenvoudigt door via een Hangfire-proces taken te starten in Arno volgens een vooraf bepaald tijdschema. Deze applicatie wordt uitgebreid om de nieuwe functionaliteiten hierin op te nemen.

Er is een onderzoek gekoppeld aan deze stage. In dit onderzoek wordt er nagegaan of Azure API Management de beste oplossing is voor dit project. Azure API Management is het API-management-systeem van Microsoft. Dit systeem zorgt ervoor dat alle API’s, die hiermee worden aangesproken, geobserveerd en uniform beheerd kunnen worden. Tobania heeft hiervoor gekozen omdat er binnen het team al veel met andere Azure-functies gewerkt wordt; dit is dus gemakkelijk in het geheel te verwerken. Als tegenhanger van Azure API Management wordt er gekeken naar Apigee van Google. Er wordt onderzocht wat de voor- en nadelen zijn van elk systeem. Via een test-API wordt er vergeleken welk systeem gebruiksvriendelijker is, welke functies mogelijk zijn en of ze van toepassing zijn op dit project. Er wordt ook onderzocht hoe het zit met de beveiliging van de data, met name of ze voldoet aan de GDPR-eisen.

TODO: conclusie

Inhoudsopgave

[Dankwoord ii](#_Toc103669846)

[Abstract iii](#_Toc103669847)

[Inhoudsopgave iv](#_Toc103669848)

[Lijst van gebruikte figuren vi](#_Toc103669849)

[Lijst van gebruikte tabellen vii](#_Toc103669850)

[Lijst van gebruikte afkortingen viii](#_Toc103669851)

[Inleiding 1](#_Toc103669852)

[I. Stageverslag 2](#_Toc103669853)

[1. Bedrijfsvoorstelling 2](#_Toc103669854)

[1.1 Situering 2](#_Toc103669855)

[1.2 Tobania.Development 2](#_Toc103669856)

[1.3 Acerta 2](#_Toc103669857)

[2. Voorstelling stageopdracht 3](#_Toc103669858)

[2.1 Connector-module 3](#_Toc103669859)

[2.1.1 Probleemstelling 3](#_Toc103669860)

[2.1.2 Doelstelling 3](#_Toc103669861)

[2.2 Automatiseringstool 3](#_Toc103669862)

[3. Uitwerking stageopdracht 4](#_Toc103669863)

[3.1 Informatie-uitwisseling 4](#_Toc103669864)

[3.2 Problemen voorkomen 4](#_Toc103669865)

[3.3 Gebruikte technologieën 4](#_Toc103669866)

[3.4 Planning 4](#_Toc103669867)

[3.5 Analyse 4](#_Toc103669868)

[3.6 Ontwikkeling 4](#_Toc103669869)

[3.7 Testen 4](#_Toc103669870)

[II. Onderzoek 5](#_Toc103669871)

[1. Vraagstelling onderzoek 5](#_Toc103669872)

[2. Methode van onderzoek 5](#_Toc103669873)

[3. Technologieën 6](#_Toc103669874)

[3.1 Wat is een API (Manager) 6](#_Toc103669875)

[3.2 Azure API Management 6](#_Toc103669876)

[3.3 Apigee Edge 7](#_Toc103669877)

[4. Literatuurstudie 9](#_Toc103669878)

[4.1 Azure API Management 9](#_Toc103669879)

[4.1.1 Belangrijkste functies 9](#_Toc103669880)

[4.1.2 Developer portaal 9](#_Toc103669881)

[4.1.3 Gateway 9](#_Toc103669882)

[4.1.4 Azure portaal 10](#_Toc103669883)

[4.1.5 Prijs en schaal 10](#_Toc103669884)

[4.1.1 Abonneren 10](#_Toc103669885)

[4.2 Apigee 11](#_Toc103669886)

[4.2.1 Developer portaal 11](#_Toc103669887)

[4.2.2 API proxy 11](#_Toc103669888)

[4.2.3 Observeren van API’s 11](#_Toc103669889)

[4.2.4 Soorten Apigee Edge 11](#_Toc103669890)

[4.2.1 Prijs 11](#_Toc103669891)

[4.2.1 Bescherming 11](#_Toc103669892)

[5. Proof of Concept 13](#_Toc103669893)

[6. Conclusie 13](#_Toc103669894)

[7. Bronnen 15](#_Toc103669895)

[Bijlagen 16](#_Toc103669896)

Lijst van gebruikte figuren

[Figuur 1 Simpele representatie van stageopdracht 3](#_Toc103672613)

[Figuur 2 Automatiseringstool 4](#_Toc103672614)

[Figuur 3 Azure API Management onderdelen 8](#_Toc103672615)

[Figuur 4 Apigee overzicht 9](#_Toc103672616)

[Figuur 5 Apigee API Analytics 9](#_Toc103672617)

[Figuur 6 Inleiding Azure API Management 10](#_Toc103672618)

Lijst van gebruikte tabellen

Lijst van gebruikte afkortingen

|  |  |
| --- | --- |
| API | Application programming interface |
| APIM | API management |
| DDoS | distributed denial-of-service |
| GDPR | General Data Protection Regulation |
| URL | Uniform Resource Locator |
| CQRS | Command and Query Responsibility Segregation |
| XAML | Extensible Application Markup Language |
| WPF | Windows Presentation Foundation |
| GPU | Graphics Processing Unit |
| CPU | Central Processing Unit |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Inleiding

Ik ben met de stageopdrachten van Tobania in contact gekomen via PXL. Ik had al wel van hun gehoord via seminaries. Toen ik zag dat ze vooral in de .NET-omgeving bezig zijn had ik direct interesse. Na alle opdrachten te hebben doorgelezen vond ik er verschillende tussen die ik zou willen doen. Ik heb dan ook snel contact opgenomen met hun om eens te horen wat de mogelijkheden waren. Tijdens een eerste gesprek werd al snel duidelijk dat er een goede connectie was tussen Tobania en mezelf. Ik heb dan ook een stageopdracht van hun gekozen. Hun ligging is voor mij ook belangrijk want Hasselt is snel en makkelijk bereikbaar van mijn thuis.

De stageopdracht is een project voor Acerta, een klant van Tobania. Acerta wilt hun data synkroniseren met de data van een externe partij Mpleo. Dit zal gebeuren via een API. Deze API zal worden beheerd in Azure API Management. Het is ook de bedoeling dat het synkroniseren van data gebeurd via een applicatie die vorig jaar ontwikkeld is door een andere stagiair. Het ontwikkelproces zal verder beschreven worden in het stageverslag.

Er wordt voor deze opdracht gebruik gemaakt van Azure API Management voor de externe API’s te beheren. Is dit de enige of de juiste technologie voor dit project? Dat zal het onderzoek moeten uitwijzen.

1. Stageverslag

# Bedrijfsvoorstelling

Er wordt een korte schets gemaakt over hoe Tobania in elkaar zit en van welk team deze opdracht deel zal uitmaken.

## Situering

Tobania is een groot Belgisch consultancy-bedrijf binnen de ICT- en businesswereld. Ze bieden hun kennis en advies aan voor bedrijven en overheden en hebben meer dan duizend *Tobians* met verschillende profielen tewerkgesteld om te voldoen aan de verwachtingen van een klant. Tobania bestaat uit vier grote units .Development, .Business, .Quality & Support en .Data. “Met een omzet van 85 miljoen euro in 2018 en meer dan 1000 IT-specialisten en business consultants (“Tobians”), behoort Tobania tot een van de grote spelers in België.” [1] Het hoofdgebouw van Tobania bevind zich is in Strombeek-Bever maar ze hebben nog andere kantoren in Gent en Hasselt. Tobania heeft vele consultants en zij kunnen zich over heel België situeren. [1]

## Tobania.Development

Voor deze stageopdracht wordt er gewerkt in de .Development afdeling van Tobania. Deze unit van Tobania richt zich vooral op het maken van applicaties. Binnen .Development zijn er twee verdelingen. Als eerste wordt er aan traditioneel consultancy gedaan. Dit wil zeggen dat de developers deel gaan uitmaken van een team van een klant om deze te versterken. Het andere luik van .Development is wat Tobania het “factorymodel” noemt. Met dit model zal een klant naar Tobania komen met een probleemstelling en dan zal Tobania zelf uitzoeken welke developers hierop gezet worden. Ze zullen dan samen met de klant een oplossing zoeken voor hun probleem. Binnen dit model bevinden zich nog twee groepen. Het java team en de .NET team. Het java team bevind zich voornamelijk in Brussel en het .NET team zit in Hasselt. De stage bevind zich in het “factorymodel” van .Development in het .NET team en zal dus doorgaan in Hasselt.

## Acerta

Acerta is de klant van Tobania waarvoor deze stageopdracht gemaakt wordt. “Acerta is een Belgische organisatie actief binnen de human resources-dienstverlening. Het bedrijf heeft 25 kantoren over het hele land. … Acerta heeft een aanbod voor starters, zelfstandigen, kmo’s en grote ondernemingen. Ze zijn gericht op zowel de private als de publieke sector. Daarnaast richt het bedrijf zich ook op de social profit.” [2]

# Voorstelling stageopdracht

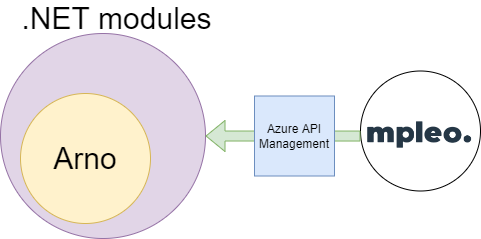
Deze stageopdracht heeft als doel een functionaliteit te ontwikkelen voor Acerta die klant zijn van Tobania.

## Connector-module

Rond Arno heeft Tobania nieuwe functies gemaakt die kunnen uitgevoerd worden. Deze worden “modules” genoemd een voorbeeld van zo een bestaande module is Voucher-module. Deze module kan gebruikt worden om maaltijdcheques te beheren van werknemers. Voor deze stageopdracht zal er een nieuwe module aangemaakt worden namelijk de “connector-module”.

### Probleemstelling

Acerta wilt binnen hun reeds bestaand HR-programma Arno de optie hebben om data van een externe bron te synkroniseren met hun eigen data. Deze externe data zal voor deze stageopdracht bestaan uit werknemers gegevens van Mpleo, een andere HR-firma. Er zal een aanpassing moeten gebeuren in Arno om dit mogelijk te maken. De gegevens van Mpleo moeten door een gateway van Azure API Management gaan. Ook moet er gekeken worden dat de nieuwe functionaliteiten kunnen opgenomen worden in de automatiseringstool die vorig jaar ontwikkeld is tijdens de stage van een andere student. Omdat Arno verouderde software is maar toch nog krachtig heeft Tobania al een nieuwe laag met modules over de bestaande code gemaakt. Dus nieuwe functies zullen een uitbreiding zijn van deze nieuwe laag code.



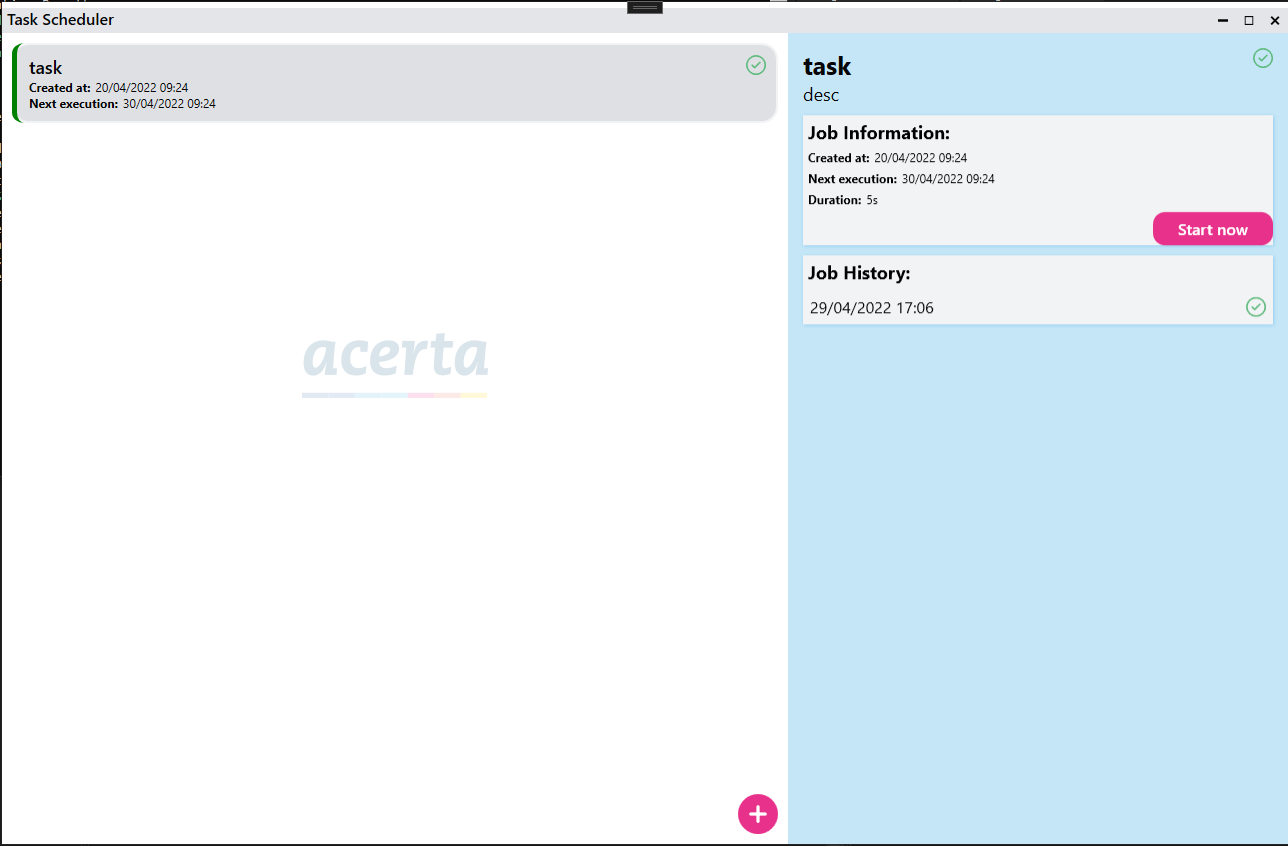
Figuur 1 Eenvoudige representatie van stageopdracht

### Doelstelling

Tijdens de stage wordt er een nieuwe module voorzien in Arno, namelijk de connector-module. Deze zal gegevens ophalen van Mpleo via Azure API Management. Er wordt gewerkt met Azure API Management omdat er in de toekomst met meerdere data bronnen gewerkt zal worden (API’s) en op deze manier kan er een beter overzicht worden gehouden op de API’s. Azure API Management moet geconfigureerd worden zodat de data van Mpleo eerst door een API-gateway gaat. Er zal een implementatie gebeuren van de nieuwe Arno functies zodat deze kunnen opgestart worden in de automatiseringstool.

## Automatiseringstool

Er is vorig jaar een automatiseringstool ontwikkeld tijdens de stage van een student. Deze tool maakt het mogelijk om taken uit te voeren in een vooraf bepaald tijdschema. Deze automatiseringstool heeft een aantrekkelijke interface zodat het makkelijk is om ermee te werken. Acerta was hier heel tevreden mee en daarom zal deze tool verder worden ontwikkeld zodat de nieuwe functies van Arno hierin gestart kunnen worden.



Figuur 2 Automatiseringstool

## Gebruikte technologieën

### .NET Core & .NET Framework

.NET Core en .NET Framework zijn twee gratis en opensourceframeworks van Microsoft. .NET Core is de opvolger van .NET Framework en het grote verschil is dat .NET Core cross-platform werkt. Hiermee bedoeld men dat het zowel op Windows, Linux en MacOs werkt. [3]

Voor de automatiseringstool is er gewerkt met .NET Core maar voor de modules rond Arno is er gewerkt met .NET Framework.

### CQRS pattern

CQRS pattern is een architectuur patroon waarvan gebruik gemaakt wordt in de modules. CQRS staat voor *Command and Query Responsibility Segregation*. Dit concept is dat er gewerkt wordt met queries en commands. Een query wordt gebruikt om resultaten op te halen, ze passen een systeem niet aan. In deze opdracht zal er een query gemaakt worden om data op te halen vanuit Arno. Commands passen het systeem wel aan, deze zullen gebruikt worden om data toe te voegen aan Arno of om bestaande gegevens te updaten binnen Arno. [4]

### Azure API Management

Azure API Management is de API management tool van Azure. Azure is het cloud platform van Microsoft. Dit APIM wordt gebruikt om de externe data van Mpleo te verkrijgen. Er is een vergelijkende studie gedaan over Azure API Management in het onderdeel *Literatuustudie.*

### Automapper

Automapper is een kleine *library* om data van een bepaald type te converteren naar een ander type. Dit is niet altijd evident om te doen, maar dankzij Automapper is dit makkelijker. Automapper wordt in deze opdracht gebruikt om de data van Arno om te vormen naar een type waarmee we kunnen werken. [5]

### SpecFlow

SpecFlow is een software tool om automatische test oplossing voor .NET te maken. Dankzij deze tool kunnen er integratie testen geschreven worden in menselijke taal zodat deze ook leesbaar zijn voor niet technische personen. [6] In deze opdracht zal er gebruik gemaakt worden van SpecFlow om een integration test te voorzien waarmee de module getest kan worden.

### WPF (Windows Presentation Foundation)

WPF is een gratis opensourcefrontendframework dat ontwikkeld is door Microsoft. Het maakt gebruik van XAML (Extensible Application Markup Language) om de frontend te programmeren. WPF is een Windows applicatie en wordt ondersteund door alle Microsoft Windows versies vanaf Vista. Indien nodig kunnen er ook bibliotheken geïnstalleerd worden op oudere versies van Microsoft Windows om WPF te laten werken.

Als eerste maakt WPF gebruik van Direct3D, dit zorgt ervoor dat Windows gedeeltelijk de grafische berekeningen zal doen die de applicatie nodig heeft om te renderen. Windows zal deze berekeningen doen op de GPU (Graphics Processing Unit) waardoor de CPU minder belast wordt.

Vervolgens werkt WPF met *data binding* dit zorgt ervoor dat data koppelen aan frontendelementen eenvoudiger is. Dankzij *databinding* moeten we geen referenties hebben naar de view in de viewmodel.

Ook kan er binnen WPF sjablonen gebuikt worden die de frontendelementen kunnen manipuleren.

Ten slotte biedt WPF animaties aan om de applicatie er dynamischer uit te doen zien.

### Hangfire

In de automatisering tool wordt er gebruik gemaakt van Hangfire. Hangfire wordt gebruikt om taken automatisch op de achtergrond uit te voeren. Dit zal achterliggend de Arno modules laten werken op het voorop bepaalde tijdstip.

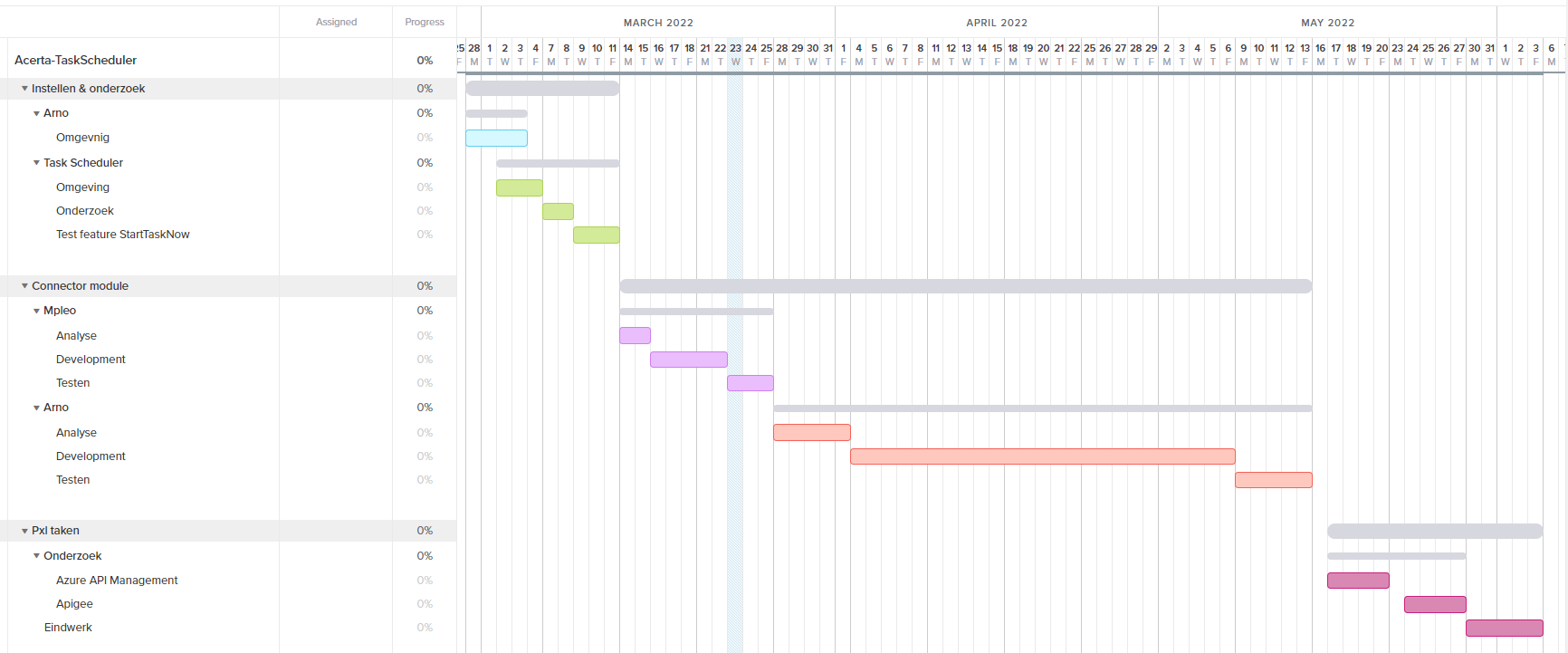
# Uitwerking stageopdracht

## Informatie-uitwisseling

Er zal dagelijks veel informatie uitgewisseld worden. Zo zal er elke dag een stand-up zijn met alle stagiairs maar ook een stand-up met Walid en Joren waar eventueel Bart, Kris en Ruud ook deel van zullen uitmaken als ze tijd hebben. Deze stand-ups zullen altijd via Microsoft Teams doorgaan omdat niet iedereen altijd samen op kantoor aanwezig kan zijn. Ook voor spontane communicatie zal er gebruik gemaakt worden van Microsoft Teams, deze tool maakt het communiceren met het hele development team gemakkelijk.

## Planning

Tijdens de eerste weken van de stage is er gekozen om te werken met een Gant-grafiek. Dit is een tijdlijn voor het project waarin er aangeduid kan worden hoe het project zal verlopen. Dankzij deze grafiek zal er bepaald worden hoelang het project zal duren en welke middelen er nodig zijn. Er kan ook gekeken worden of het project nog op schema ligt. Voor deze grafiek wordt er gebruik gemaakt van de gratis onlinetool TeamGantt.



Figuur 3 Gantt-grafiek

Het project is verdeeld in drie grote delen Instellen & onderzoek, Connector module en PXL taken. Tijdens de Instellen & onderzoek fase zal er vooral gezorgd worden dat elle reeds bestaande code werkt op het apparaat van de stagiair. Er moet gezorgd worden dat er zowel met Arno en met de Automatiseringstool kan gewerkt worden. Om te leren werken met de code zal er een test functie gemaakt worden in de automatiseringstool.

Tijdens het tweede deel Connector-module zullen alle taken zitten die gaan over het ophalen van de Mpleo data en het wegschrijven van deze data in Arno. Hier wordt tijd voorzien voor analyse, development en testen.

Als laatste deel is er nog de PXL taken, hier valt het onderzoek onder en portfolio.

## Analyse

Er is al een analyse gebeurd door de medewerkers van het Acerta project. Samen met hun is er ook een kanbanbord voorzien met alle userstory’s op.

### Userstory’s

Userstory’s zijn kleine beschrijvingen van wat de applicatie allemaal moet doen. Dit geeft een duidelijk beeld over wat allemaal moet voorzien zijn bij de eindapplicatie.

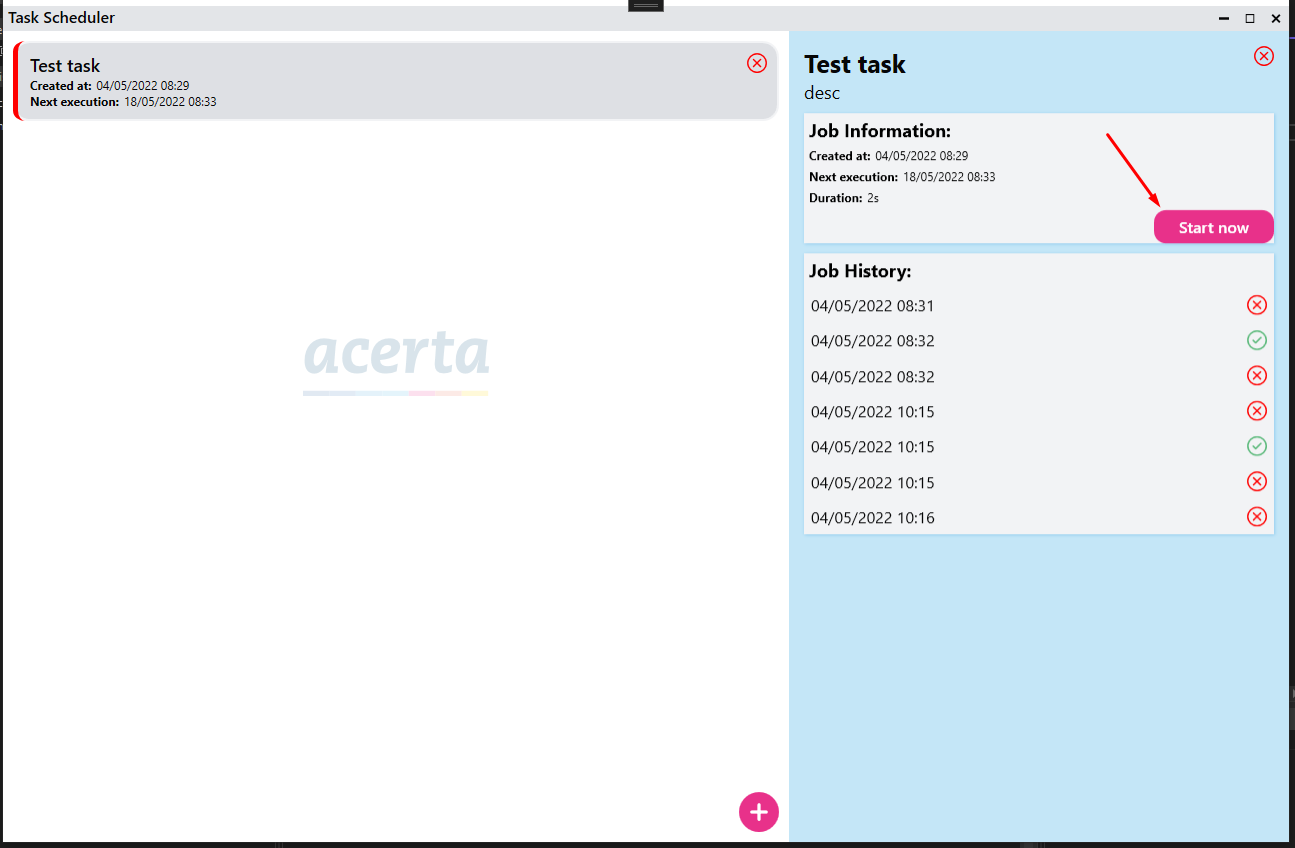
In dit project wordt er op een Agile manier gewerkt. Dus zal en een *backlog* gemaakt worden met alle userstory’s en zo kan er elke sprint een aantal story’s opgenomen worden om af te werken. Deze userstory’s worden bijgehouden in Jira.

Afbeelding met tafel

Automatisch gegenereerde beschrijving

Figuur 4 Kanbanbord Azure DevOps

### Hands-on Automatiseringstool

Omdat er moet gewerkt worden met twee reeds bestaande projecten is er besloten om een kleine nieuwe functie toe te voegen aan de automatiserinstool om zo de code beter te begrijpen. Deze functie zou ervoor dienen om een reeds bestaande *Task* onmiddellijk te starten in plaats van te wachten tot het juiste tijdstip. 

Figuur 5 Automatiseringstool met extra functie

Dankzij deze extra functie is het makkelijker om de code te begrijpen om deze in de toekomst aan te passen voor de nieuwe Arno modules.

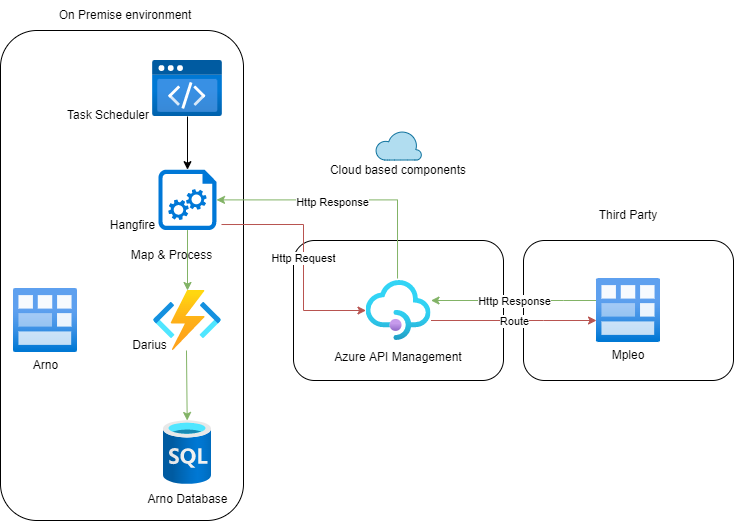
### Proof of Concept Mpleo data ophalen

Er is ook een *proof of concept* gemaakt voor het ophalen van de data van Mpleo via Azure API Management. Dit is een alleenstaand programma dat de data van Mpleo zal ophalen en ze in een tekst bestand zal opslaan. Deze *proof of concept* is een simpele console applicatie. Dankzij deze applicatie is er al gewerkt met Azure API Management. Dit staat al klaar om geïmplementeerd te worden in de Arno module.

## Ontwerp

### Schema

Er is een *high-level* schema gemaakt om alle componenten van de applicatie aan te duiden. Dit geeft een overzicht van alle verschillende frameworks die gebruikt worden. Dit diagram is gemaakt met de onlinetool Draw.io [7]



Figuur 6 High level diagram connector-module

Een gebruiker van de applicatie begint bij de *Task Scheduler* (automatiseringstool) en creëert een taak die op een bepaald tijdschema wordt uitgevoerd (bijvoorbeeld één keer per maand). Achterliggend houdt een Hangfire server de taak bij en voert deze uit op het juiste moment. Op het correcte moment wordt er een request gestuurd naar Azure API Management waarbij de data wordt opgehaald bij Mpleo, vervolgens wordt deze data teruggestuurd naar Arno. Binnen Arno wordt de data dan gesynkroniseerd met de Arno databank via Darius.

### Design

Voor het design van de Automatiseringstool wordt er altijd verder gegaan op het reeds bestaande design. Bijvoorbeeld als er extra knoppen voorzien worden moeten die dezelfde stijl hebben als de reeds bestaande knoppen. Er mag geen breuk zijn in het design van de bestaande applicatie.

## Ontwikkeling

### Projectstructuur

Het Arno project is zeer groot en het moet ook duidelijk zijn waar de nieuwe code staat. Dus zal er in de solution onder de map *Services* een nieuw project gemaakt worden met de naam Acerta.Arno.Services.ConnectorModule.

Afbeelding met tekst, monitor, schermafbeelding

Automatisch gegenereerde beschrijving

Figuur 7 Structuur Connector-module

Binnen het connector-module project wordt ook een structuur aangehouden.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Figuur 8 Projectstructuur Arno

Er is een mappenstructuur voorzien zodat alle code op een logische plaats staat. In de map *Data* bevinden alle Query’s en Commands. De map *Interfaces* bevat alle interfaces die gebruikt zijn in het connector-module project. Verder is er ook nog de map *Mapping* staan de mappers waar Automapper gebruik van maakt. De laatste map *Models* bevat de modellen die nodig zijn voor dit project. Als laatste zijn er nog de algemene classen die rechtstreeks in het project staan.

Er wordt ook nog gebruik gemaakt van bestaande onderdelen van het Arno project maar omdat deze door verschillende modules gebruikt worden staan ze anders georganiseerd.

### Integratie

Voor het laten werken van de connector-module zijn er eerst enkele voorbereidende stappen nodig. Eerst zal er een configuratie moeten gebeuren van Azure API Management. Binnen Azure wordt er een nieuwe resourceaangemaakt van het type API Management. Aan deze resource wordt er een abonnement gekoppeld dat het financiële zal afhandelen. Wanneer de resource klaar is kan er een implementatie gebeuren zodat de Mpleo API hierin zal werken. Na deze implementatie wordt er een sleutel gegenereerd die nodig is om gebruik te maken van Azure API Management.

Binnen Arno is de structuur als volgt: er zal een *Task* gemaakt worden die de data zal synkroniseren. Deze *Task* gaat beginnen met het ophalen van data uit Azure API Management, met behulp van de sleutel die we van Azure API Management gekregen hebben. Ook gaat de *Task* de werknemers data van Arno ophalen aan de hand van een query. Als deze twee data sources opgehaald zijn kan er een vergelijking gebeuren. Aan de hand van deze vergelijking wordt er bepaald of het gaat over een nieuwe werknemer die niet gekend is in Arno of een bestaande werknemer. Als het een nieuwe werknemer is wordt deze toegevoegd aan Arno met behulp van een *command*. Wanneer het gaat over een bestaande werknemer worden zijn gegevens geüpdatet in Arno, dit gebeurd ook met een *command*.

Als gebruiker kan er via de automatiseringstool gewerkt worden om een nieuwe taak aan te maken. De gebruiker kan dan kiezen voor “Synkroniseer werknemers Mpleo” en kan dan een tijdschema kiezen. Op het juiste tijdstip zal Hangfire dan de *Task* uitvoeren.

### Eindresultaat

TODO

## Testen

### Integratie testen

Omdat dit geen gewoonlijk project is kan je niet zomaar alle code uitvoeren. Dus om een stuk code te testen op zichzelf zal er worden gewerkt met integratie testen. Via SpecFlow kan je een test schrijven die er leerbaar uitziet voor niet technische mensen en dit is ook de beste manier om de gecreëerde stukken code uit te voeren.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Figuur 9 Voorbeeld SpecFlow integratie test

Zoals te zien op de figuur wordt voor deze test eerst een Arno applicatie gestart. Dan zullen er enkele functies uitgevoerd worden en vervolgens gebeurd er een controle

TODO aanpassen naar eigen test (nog niet af)

1. Onderzoek

# Vraagstelling onderzoek

Wanneer er in een groot project gebruik gemaakt wordt van verschillende *Application Programming Interfaces* (API’s) is het niet altijd voor de hand liggend hoe deze met elkaar gaan communiceren. Als oplossing hiervoor kan er gebruik gemaakt worden van een API management (APIM) systeem zodat er een duidelijke structuur gevolgd kan worden.

Voor dit project is er gekozen om te werken met Azure API Management. Dit is het API management systeem van Microsoft. Er is hiervoor gekozen omdat er binnen het team al veel met andere Azure functies gewerkt wordt, dit is dus gemakkelijk in het geheel te verwerken.

Is Azure API Management de beste keuze voor het project? Dat zal dit onderzoek moeten uitwijzen. Als tegenhanger van Azure API Management is er gekozen om Apigee van Google te vergelijken. Zijn deze gelijkaardig? Wat zijn de voor- en nadelen van elk? Hoe zit het met het kostenplaatje? Hoe is het met de beveiliging van data? Hoe zit het met GDPR?

Dit zijn al enkele vragen die gesteld kunnen worden en waarvan de antwoorden onderzocht zullen worden.

# Methode van onderzoek

Voor deze stageopdracht moet er gewerkt worden met externe API’s. Omdat deze API’s van verschillende ontwikkeling omgevingen komen is het overzichtelijker om ze te beheren via een API management systeem. Binnen dit project is ervoor gekozen om te werken met Azure API Management.

Om vast te stellen of het gekozen management systeem de beste keuze is voor dit project zal deze vergeleken worden met een gelijkaardige tool, namelijk Apigee. Er zal met een test API gewerkt worden, namelijk [*elephant-api.herokuapp.com*](https://elephant-api.herokuapp.com/). Er zal een implementatie gebeuren van de test API in de twee API management systemen. Dan zal er gekeken worden welke manipulaties er gedaan kunnen worden op de data door het API management systeem.

Omdat Azure APIM en Apigee cloud systemen zijn, zijn er ook kosten verbonden aan het gebruik. De twee systemen hebben een proefversie die gebruikt kan worden om ze te testen met een start budget. Via deze proefversies met een start budget kan er ook een vergelijking worden gemaakt met de prijzen. Er kan ook een vergelijking worden gemaakt over het gebruiksgemak bij het opzetten van de systemen.

Ten slotte zal er ook onderzocht worden hoe het zit met beveiliging van de data die door de API management systemen komt.

# Technologieën

In dit hoofdstuk zullen de gekozen technologieën worden toegelicht. Er zal ook extra uitleg volgen over wat er bedoeld wordt met API’s.

## Wat is een API (Manager)

Wat is een API? Geetima Dutta beschrijft een API als een tussenpersoon wanneer twee partijen niet rechtstreeks met elkaar kunnen communiceren. [7] In de IT-wereld houdt dit in dat er applicaties zijn waarmee men data kan manipuleren. Hiermee wordt er bedoeld dat er data kan worden opgehaald, bewerkt, toegevoegd en verwijderd van een bepaalde databank. En deze manipulaties gebeuren via een API.

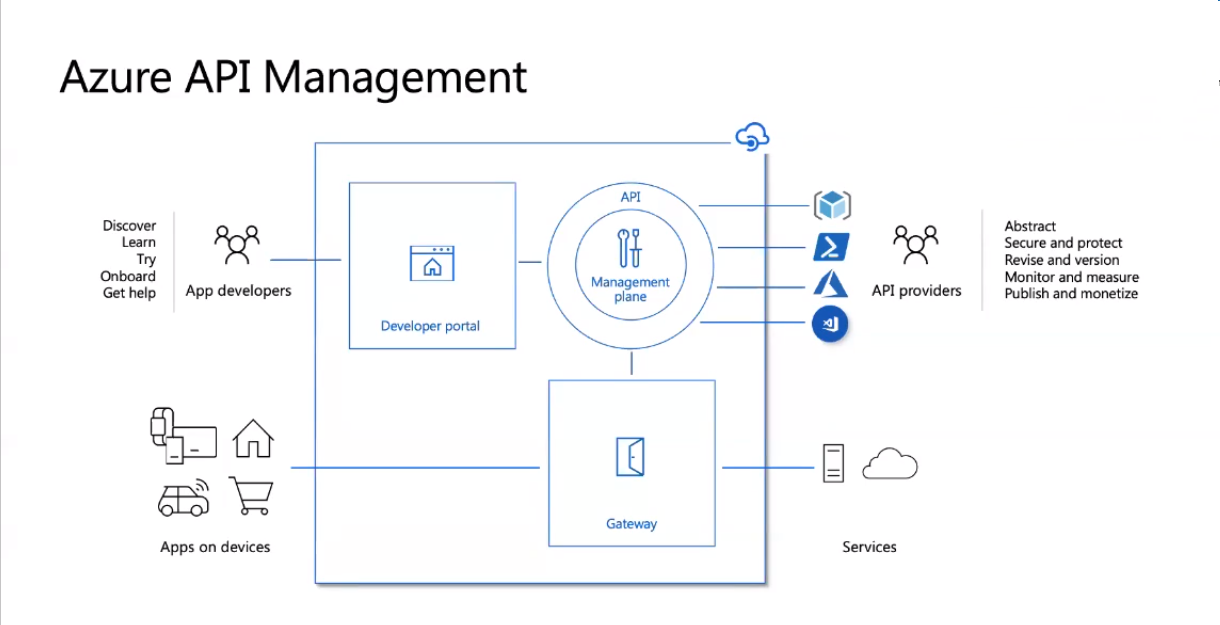
Als er voor een project gewerkt wordt met meerder API’s is het nuttig om met een API Management tool te werken. Hierin kunnen verschillende API’s aan één project gelinkt worden. Dankzij de API Management tool kunnen de API’s geobserveerd en gemanaged worden. Ook is beveiliging vaak een belangrijk deel van API Management. [8] Een andere functie van een APIM is dat consumenten zich kunnen abonneren op bepaalde API’s. Je kan as project eigenaar je API’s openstellen en *subscription key’s* geven aan consumenten die API’s mogen gebruiken.

## Azure API Management

*“Azure API Management is een hybride, multicloudbeheerplatform voor API's in alle omgevingen.”* [9]

Azure API Management (APIM) is van Microsoft. Het is deel van de cloudomgeving van Azure. Het bestaat uit drie grote delen, een API-gateway, een *developer portal* en een *management plane* ook wel het Azure portaal genoemd in andere bronnen. Deze drie onderdelen worden volledig beheerd door Azure. [9]

De API-gateway zorgt ervoor dat alle aanvragen van een applicatie één aanspreekpunt hebben. Dit wil zeggen dat alle aanvragen van verschillende API’s via deze gateway zullen gaan. De gateway zal als een façade dienen. Deze gateway geeft ook de mogelijkheid om instellingen te configureren zoals beveiliging, routing … [9]



Figuur 10 Azure API Management onderdelen

Vervolgens is er ook het *management plane* dat te vinden is op het Azure portaal, hier worden de instellingen beheerd en bekeken.

Ten slotte is er het developerportaal, dit is een automatisch gegenereerde, aanpasbare website waar de documentatie van de API’s te vinden is. [9]

## Apigee Edge

Apigee X is het APIM van Google Cloud. Er bestaan verschillende versies van Apigee. Maar voor dit onderzoek zal er enkel gekeken worden naar Apigee Edge. Apigee bestaat uit twee grote componenten. Apigee *services* en Apigee *runtime.*

Apigee *services* handelt met alles over het maken en consumeren van API’s. [10]

Apigee *runtime* zorgt voor tools die het mogelijk maken om API *proxies* toe te voegen en de beheren. Dankzij deze *proxies* kunnen er beleids protocollen aangemaakt worden op vlak van beveiliging, rate beperking, bemiddeling, caching … Ook kan het gedrag van de API’s aagepast worden door het toepassen van scripts. [10]

High level overview
  of the Apigee architecture.

Figuur 11 Apigee overzicht

Tot slot is er ook nog Apigee API Analytics dat sterke hulpmiddelen voorziet om korte en lange termijn gebruik van je API’s op te volgen. [10]

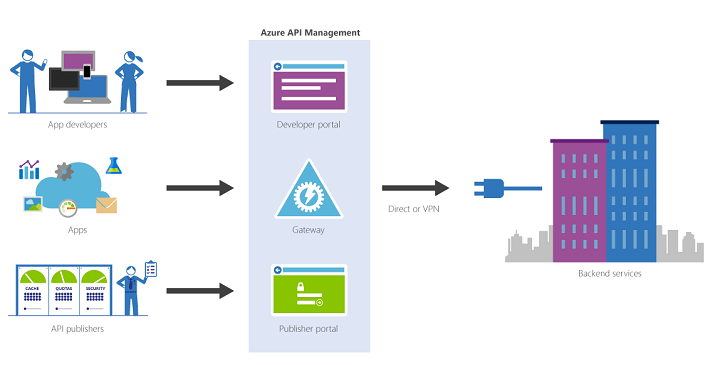
An analytics dashboard that shows the number of policy errors in graph and
    tabular form.

Figuur 12 Apigee API Analytics

# Literatuurstudie

## Azure API Management

### Belangrijkste functies

Romeel Khan vind het belangrijk dat een API-manager een omgeving is waarin je als ontwikkelaar gemakkelijk API’s kan ontwikkelen, *deployen* en beheren. [11] Specifiek wil dit zeggen dat er een API-gateway moet aanwezig zijn dat zal zorgen voor beperkte toegang tot de backend. Ook moet er een manier zijn om de snelheid te beperken en er moeten veiligheid protocollen opgesteld kunnen worden. Hij vind het ook nog belangrijk dat er goede documentatie en publiceer hulpmiddelen zijn, een *access portal* en er moet voldoende mogelijkheid zijn om de API’s te observeren en te analyseren. [11] Nicolás Andres Mijoch vind dan weer dat één van belangrijkste functies dat je enkel moet betalen voor wat je gebruikt. Dit komt door de *consumption price tier layer.* [12]

Figuur 13 Inleiding Azure API Management

### Developer portaal

In het developer portaal vinden we een automatisch gegenereerde API catalogus, documenten en code voorbeelden. Dit is de plaats waar de API-consument kan komen om alle informatie te vinden over een bepaalde API die in een project zit. Dit portaal bevat ook de *subcription* *key* voor de consumenten die zich geabonneerd hebben op een bepaald API. Er is ook een console om de API uit te testen. [13] Romeel Khan haalt ook nog aan dat de documentatie die gegenereerd wordt in verschillende formaten kan komen bijvoorbeeld Swagger Open API, metadata of WADL *document* (XML). [11] Volgens Nicolás Andres Mijoch is het heel handig dat Azure APIM dit automatisch doet. Hij zegt dat deze functie veel tijd kan besparen bij een saaie taak namelijk documenteren. [12]

### Gateway

Een API-gateway is de toplaag van de backend service. Het is frontend dat werkt als een proxy. Alle API-*requests* komen langs deze gateway. De gateway zal deze aanvragen dan doorsturen naar de juiste services en de resultaten terugzenden. Authenticatie, autorisatie en beperkingen worden via deze laag beheerd. [13] Het hoofddoel van een API-gateway is om toegang te beveiligen en het beperken van toegang, dus ook hoe vaak een API per dag of uur mag aangesproken worden. [11] Nicolás Andres Mijoch voegt ook nog to dat je toegang kan verlenen aan groepen administrators, developers en gasten. Maar je kan ook zelf nog op maat gemaakte groepen toevoegen. Nicolás zegt ook nog dat je *policies* kan toevoegen aan API’s die uitgevoerd zullen worden samen met een API-*request*. “They [policies] are related to access restriction going through validation like check HTTP headers or remove them if this information could compromise your backend. Authentication policies, cross-domain validations, and any other security measure could be checked before reaching your back-ends.” [12]

### Azure portaal

Via het Azure portaal kunnen developers alles instellen in verband met de API’s. Romeel Khan zegt dat het Azure portaal een administratieve interface is voor het beheren van API programmas. Het bevat belangrijke functies zoals een dashboard met gepubliseerde API’s. Een mogelijkheid om je *policies*, en consumenten te beheren ook. Ten slotte bevat het ook *analytics* voor het observeren van de API’s. [11]

### Prijs en schaal

Er zijn verschillende prijs categorieën waarmee er gewerkt kan worden. De developer categorie waarmee een developer de tool kan testen. Dit is niet bedoeld om te gebruiken in effectieve projecten. [12] De volgende categorie is *Basic, Standard & Premium* volgens Nicolás Andres Mijoch vallen deze drie onder dezelfde categorie omdat ze hetzelfde aanbieden maar met een groter volume. Dit houd ook in dat dit makkelijk kan vergroot worden als het project groter wordt. Zo kan er geüpgraded worden van basic naar premium in simpele stappen. Bij deze categorie staat je service altijd aan en betaal je per uur. [12] Als laatste categorie is er ook nog de *consumption* categorie, hier zal er enkel aangerekend worden wanneer er een API-*request* gestuurd wordt. De prijzen lopen van €0,06/uur tot €3,45/uur en voor de *consumption* categorie is het €0.0315 per 10,000 API-*requests*. [14] Het is opvallend dat Microsoft heel transparant is over de prijzen van zijn producten in Azure.

### Abonneren

Azure API Management werkt op basis van abonnees. De beheerder van een project op Azure APIM moet toegang verlenen aan consumenten voordat deze een API kunnen gebruiken. Dit gebeurd aan de hand van een sleutel die de developer uitdeelt. [13] De sleutel verwijst naar een groep gebruikers die hun eigen instellingen hebben op vlak van beveiliging en welke API’s ze kunnen consumeren. [12] Dit is een belangrijk concept bij Azure APIM want anders kan iedereen met de juist URL’s aan de data omdat alles van Azure APIM in de cloud zit.

## Apigee

### Developer portaal

Apigee Edge heeft een makkelijk aan te passen developer portaal, hier kunnen developers de API’s bestuderen en zich aanmelden als ze een API willen gebruiken. Het bied ook een volledige oplossing voor het beheren van toegang tot de API’s volgens Pedro Dantas. [15] Madhuri Yerukala voegt hier nog aan toe dat: “It designs a structure for publishing interactive SmartDOCs API documentation and Smart essential management application for safe registration, key generation, and enlisting of developers whether they are from internal, external, or partner.” [16] Ten slotte zegt Vaibhav Gupta ook nog dat het app developers toegang geeft tot API-documentatie voor de beschikbare producten en het beheren van API-sleutels. [17]

### API proxy

Zoals er bij Azure APIM gewerkt wordt met een gateway wordt dit in Apigee een *proxy* genoemd. Vaibhav Gupta zegt dat de proxy laag een abstractie is voor de backend API’s. Hij zegt ook dat een *proxy* een soort schild is voor de echte achterliggende API’s. [17] Pedro Dantas noemt dit een *proxy endpoint*, dit is hetgene dat de consumenten zien en voor hun zal dit de API zijn in plaats van de achterliggende code. [15] Dus consumenten van een API zullen nooit echt in contact komen met een API maar altijd door deze *proxy* laag. [16]

### Observeren van API’s

“Edge also has a powerful monitoring module that allows you to check the operation of your APIs, including pre-configured dashboards for the most common metrics such as traffic, errors, geomapping, etc.” [15] Apigee Edge voorziet verschillende visuele hulpmiddelen zoals een dashboard (dat een algemene kijk geeft over alle API-programmas), aangepaste rapporten en *GeoMaps.* Het bied ook de mogelijkheid aan om trends te zien in het gebruik per individuele API. [16]

### Soorten Apigee Edge

Madhuri Yerukala haalt nog aan dat Apigee op drie verschillende manier gebruikt kan worden. In een private cloud, een publieke cloud of hybride. Een private cloud houd in dat alles op locatie zal gebeuren. Als er voor deze formule gekozen wordt dan zal niemand buiten de omgeving aan de API’s kunnen. Voor een publieke cloud zal Apigee in de cloud gehost worden. Iedereen met de juiste sleutel kan dan aan de API’s. Tot slot kan er ook een hybride optie gekozen worden waarmee er een combinatie van de vorige twee zal gebruikt worden. [16] Vaibhav Gupta en Pedro Dantas bevestigen deze formules van Apigee. [17] [15]

### Prijs

Over de prijs van Apigee is er niet veel te vinden. Google is niet erg transparant op dit vlak.

### Bescherming

Apigee bied ook bescherming aan voor de backend API’s wanneer er ongebruikelijk data verkeer is. Het zal een *Spike Arrest* doen wanneer er plots veel meer data verkeer is dan normaal. Dit is onder andere een bescherming tegen DDoS aanvallen. Er zal ook aan *cashing* gebeuren dit zal ervoor zorgen dat data performanter kan opgehaald worden. Ten slotte *Quotas and rate limits*  limiteren het aantal connecties dat een API kan hebben. Dit zorgt ervoor dat het systeem niet overbelast wordt.

# Proof of Concept

# Conclusie

# Bronnen

|  |  |
| --- | --- |
| [1] | „Tobania,” [Online]. Available: https://nl.everybodywiki.com/Tobania. [Geopend 29 04 2022]. |
| [2] | „Acerta,” [Online]. Available: https://nl.wikipedia.org/wiki/Acerta. [Geopend 29 4 2022]. |
| [3] | „.NET Core,” [Online]. Available: https://nl.wikipedia.org/wiki/.NET\_Core. [Geopend 17 5 2022]. |
| [4] | „Apply simplified CQRS and DDD patterns in a microservice,” [Online]. Available: https://docs.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/apply-simplified-microservice-cqrs-ddd-patterns. [Geopend 17 05 2022]. |
| [5] | „AutoMapper,” [Online]. Available: https://automapper.org/. [Geopend 17 5 2022]. |
| [6] | „Welcome to SpecFlow’s documentation!,” [Online]. Available: https://docs.specflow.org/projects/specflow/en/latest/. [Geopend 17 5 2022]. |
| [7] | G. Dutta, „How do you explain API to a 5-year-old?,” [Online]. Available: https://geetima.medium.com/how-do-you-explain-api-to-a-5-year-old-7d70f6986a15. [Geopend 26 04 2022]. |
| [8] | „API Management: What It Is and Why You May Need It,” [Online]. Available: https://www.altexsoft.com/blog/api-management/. [Geopend 27 04 2022]. |
| [9] | „Meer informatie over API Management,” [Online]. Available: https://docs.microsoft.com/nl-be/azure/api-management/api-management-key-concepts. [Geopend 27 04 2022]. |
| [10] | „What is Apigee?,” [Online]. Available: https://cloud.google.com/apigee/docs/api-platform/get-started/what-apigee. [Geopend 28 04 2022]. |
| [11] | R. Khan, „AZURE API MANAGEMENT,” [Online]. Available: https://www.reply.com/solidsoft-reply/en/content/azure-api-management. [Geopend 29 04 2022]. |
| [12] | N. A. Mijoch, „Azure API Management Tools & Functions,” [Online]. Available: https://bizbrains.com/blog/azure-api-management/. [Geopend 29 04 2022]. |
| [13] | U. Aytaş, „What is Azure API Management? Why Should We Use It? What are the benefits? (Part 1),” [Online]. Available: https://medium.com/devopsturkiye/what-is-azure-api-management-why-should-we-use-it-what-are-the-benefits-part-1-11233376ac52. [Geopend 29 4 2022]. |
| [14] | „API Management pricing,” [Online]. Available: https://azure.microsoft.com/en-gb/pricing/details/api-management/. [Geopend 29 4 2022]. |
| [15] | P. Dantas, „An entire new experience with Apigee Edge.,” [Online]. Available: https://polarising.com/techinside/an-entire-new-experience-with-apigee-edge/. [Geopend 29 4 2022]. |
| [16] | M. Yerukala, „What is Apigee Edge,” [Online]. Available: https://mindmajix.com/what-is-apigee-edge. [Geopend 29 4 2022]. |
| [17] | V. Gupta, „Empowering Apigee with Google Cloud Platform,” [Online]. Available: https://www.hcltech.com/blogs/empowering-apigee-google-cloud-platform. [Geopend 29 4 2022]. |
| [18] | „Apigee API Management,” [Online]. Available: https://cloud.google.com/apigee. [Geopend 27 04 2022]. |

Bijlagen

1. Omschrijving Bijlage A
2. Omschrijving Bijlage B
3. Omschrijving Bijlage C
4. Omschrijving Bijlage A
5. Omschrijving Bijlage B
6. Omschrijving Bijlage C