# Externe API:

# Auth0 Management API

## Essentie

Voor onze externe API in het applied softwareproject hebben we gekozen om gebruik te maken van de management API van Auth0. Deze hebben we verwerkt in het admingedeelte van de opdracht. Met de management API verkrijgen we extra functionaliteit in het adminpaneel om bepaalde zaken verder te automatiseren en een zo eenvoudig mogelijk product af te leveren aan de klant.

De Auth0 management API is bedoeld om door backend servers gebruikt te worden om administratieve zaken in verband met Auth0 uit te voeren zonder dat je hiervoor het Auth0 dashboard zelf moet nuttigen. Algemeen kan men zeggen dat alles wat je via het Auth0 dashboard kan doen, je ook via de API kunt uitvoeren.

Het is natuurlijk belangrijk dat deze API enkel en alleen maar gebruikt wordt door vertrouwde partijen, aangezien je met deze API volledige controle hebt over alles in de specifieke Auth0 account.

Specifiek hebben we in ons project gebruik gemaakt van de API om ervoor te zorgen dat we via het admin dashboard van onze eigen applicatie Auth0 gebruikers kunnen beheren. Dit houdt in:

* Aanmaken
* Aanpassen
* Verwijderen

Dit wil zeggen dat zo goed als alle belangrijkste functionaliteit dat te maken heeft met het beheer van gebruikers via de applicatie kan gebeuren en je niet apart via het dashboard van Auth0 moet werken.

We hebben gekozen om dit te implementeren omdat het volgens ons een grote meerwaarde toevoegt aan de applicatie in zijn geheel. Het vergemakkelijkt het proces van het beheren van gebruikers zodat men niet twee verschillende dashboards moet gebruiken voor iets wat ook op deze manier kan aangeboden worden. Het grote voordeel is vooral dat alles wat aangenamer en vlotter werken wordt.

## Functionaliteit

Eén van de vereisten van het project was het gebruik van Auth0 om de authenticatie te voorzien. Het beschermen van de data/API is de belangrijkste use case waarom we authenticatie zouden willen implementeren. Dit gebeurt natuurlijk door accounts te voorzien via Auth0 met verschillende privileges/rollen om zo specifiek te bepalen wie toegang heeft tot welke data en welke front-end. Medewerkers moeten enkel en alleen hun persoonlijke taken kunnen raadplegen via de voorziene angular webapplicatie en hebben geen toegang tot het admin dashboard. Admins moeten daarentegen alle data van de API kunnen aanroepen en hebben vanzelfsprekend ook toegang nodig tot het admin dashboard dat voorzien is via het ASP.NET gedeelte.

Er komt dus heel wat kijken bij het beheer van deze accounts. Accounts moeten aangemaakt worden met de juiste gegevens zoals: email, voornaam, achternaam, wachtwoord, …. Daarnaast moeten deze accounts ook de juiste privileges en rollen toegewezen krijgen zodat de accounts de correcte toegang krijgen tot wat ze nodig hebben. Je moet deze ze natuurlijk ook kunnen aanpassen wanneer nodig bijvoorbeeld een gebruiker extra rechten toewijzen of afnemen. Ten slotte is het ook belangrijk dat deze accounts verwijderd kunnen worden wanneer nodig.

Al deze zaken kan je verrichten in het admin dashboard van Auth0 zelf. Maar het is wel duidelijk dat als je via deze manier in praktijk gaat werken dit een hele grote rompslomp zal worden. In de database houden we extra gegevens bij van de werknemers, idealiter komen deze overeen met de gegevens die we voor de accounts op Auth0 moeten bijhouden. Dit wil dus zeggen dat men eerst een werknemer via onze applicatie moet aanmaken om dan nog eens apart dezelfde werknemer aan te maken op Auth0 om een account te voorzien waarmee deze werknemer kan aanmelden. Dit wil ook zeggen dat wanneer men deze werknemer wil wijzigen, dit moet gebeuren via beide dashboards. Hetzelfde geldt voor het verwijderen. Dit dubbel werk is niet ideaal en zouden we graag vermijden.

Om deze redenen hebben we besloten om de management API in ons project te implementeren om zo bepaalde functionaliteit van het Auth0 dashboard in onze applicatie te verwerken en ervoor te zorgen dat onze applicatie 1 opzichzelfstaand geheel wordt, waarbij het beheer van de Auth0 gebruikers geautomatiseerd wordt via onze applicatie zonder dubbel werk te hoeven verrichten.

Zo hebben we dus specifiek voor volgende functionaliteit in ons admin dashboard gezorgd:

* Het aanmaken van een werknemer in de database zal ook een account aanmaken op Auth0 met dezelfde gegevens. Eveneens krijgt deze account meteen de juiste rollen toegewezen dat we kunnen kiezen in onze applicatie bij het aanmaken van een medewerker.
* Het aanpassen van een werknemer in de database zal ook de nodige aanpassingen doen voor de gebruiker op Auth0. Enkel wanneer nodig.
* Wanneer men een werknemer verwijdert, wordt deze ook verwijderd op Auth0.

We hebben uiteraard ook voorzien dat wanneer er iets fout loopt met het aanmaken, aanpassen of verwijderen van een account op Auth0 dit correct afgehandeld wordt in de backend zodat dit geen verandering in de database zal veroorzaken. Zo blijft alles netjes op elkaar afgestemd. Daarnaast hebben we ook voorzien dat dit duidelijk op de front-end vermeld wordt zodat de gebruiker weet wat zijn/haar fout was of wat er precies is fout gelopen met de juiste foutmeldingen wanneer mogelijk.

## Implementatie

Omdat we werken via het CQRS-patroon waarbij we alle CRUD-operaties opdelen in aparte commands per operatie, is de meest logische plaats om de management API te implementeren bij de specifieke commands die verantwoordelijk zijn voor de operaties in verband met werknemers:

* Command om werknemer aan te maken
* Command om werknemer te updaten
* Command om werknemer te verwijderen

De manier waarop de API aangeroepen wordt, werkt in het algemeen op dezelfde manier voor deze drie commands. Hieronder zullen we de implementatie van één van de drie, namelijk de command om een werknemer aan te maken, iets gedetailleerder toelichten aan de hand van enkele voorbeelden uit de code.

We roepen de API van Auth0 aan met behulp van “RestSharp”, een REST API client library voor .NET.

### Management token

Om te bewijzen dat je de juiste rechten en toegang hebt om de specifieke Auth0 tenant aan te roepen, heb je een token nodig. Dit token kan je verkrijgen door de management API die voor je domein voorzien is, aan te roepen op het "oauth/token" endpoint. Uiteraard moet je enkele parameters meegeven die bewijzen dat je rechten hebt om de token te verkrijgen. Deze parameters zijn:

* Client ID
* Client Secret
* Audience
* Grant Type

De waarden voor deze parameters zijn te vinden in het admin dashboard van Auth0.



In de response van deze call zit de access token die we nodig hebben om verdere calls naar de management API te kunnen doen. We vangen deze op in een eigen klasse “TokenObject”.



Als laatste controleren we of de call naar de API succesvol was, zo niet dan beëindigen we meteen de command met de foutmelding dat er iets is fout gelopen met het verkrijgen van de access token. Deze zal dan getoond worden op de front-end.

### Aanmaken gebruiker

Eens we de access token in ons bezit hebben, kunnen we de endpoints aanroepen die effectief het beheer van de gebruikers voorzien. In volgende snippet roepen we de “/users” endpoint aan om een gebruiker aan te maken. Het is zeer belangrijk om de access token niet te vergeten meegeven in de header van de call.



Er zijn tal van parameters die we kunnen meegeven aan deze call om extra info i.v.m. de user op te slaan of het account aan te passen naar wat je nodig zou hebben. Wij hebben ervoor gekozen om enkel het nodige mee te geven zoals email, wachtwoord en een naam, aangezien Auth0 enkel gebruikt wordt om accounts te voorzien voor onze applicatie, en we dus geen andere functionaliteit nodig hebben.

De response van deze call vangen we weeral op in een eigen klasse “Auth0User”, om zo enkel de properties bij te houden die van belang zijn voor ons.



De user id die we hier verkrijgen bewaren we in de database, om zo een medewerker te kunnen koppelen aan zijn Auth0 account. Deze id zal verder nog van pas komen als we de werknemer willen aanpassen, of rollen willen toekennen.

Ook hier controleren we eerst of het aanmaken van de werknemer succesvol was, zo niet dan beëindigen we meteen de command met de foutmelding dat er iets is fout gelopen met het aanmaken van de gebruiker, en eventuele andere specifiekere foutmeldingen die we vanuit Auth0 verkrijgen. Deze zullen dan getoond worden op de front-end. Dit gebeurt op dezelfde manier als bij het aanvragen van de access token.

### Rollen toekennen

De laatste stap in het aanmaken van een gebruiker op Auth0 is het toekennen van de rollen die de gebruiker nodig heeft. Dit moet in een aparte call naar de API gebeuren.



We maken hier gebruik van de unieke Auth0 ID van de gebruiker, die we verkregen hebben uit de response van het aanmaken van de gebruiker, om rollen toe te kennen aan de juiste gebruiker, namelijk degene die we net hebben aangemaakt. We kijken ook na of de gebruiker al dan niet de adminrol toegekend moet krijgen vooraleer we de call doen naar de API.

Ook hier controleren we weer of de call succesvol was of niet, met de correcte foutmeldingen mocht dit nodig zijn.

Als dit alles succesvol is, dan is de gebruiker aangemaakt op Auth0 en kunnen we de gebruiker met al zijn gegevens ook opslaan in de database.