THOMAS MORE

Plan van Aanpak

Sens van Aert
Student Bachelor Artificial Intelligence

2025 – 2026

Inhoudstabel

1. Ometa

Ometa staat voor Organizing Metadata en ontwikkelt oplossingen die bedrijven helpen om hun data beter te beheren en te integreren. Het bedrijf biedt een framework aan waarmee verschillende systemen en departementen binnen een organisatie met elkaar verbonden kunnen worden. Hierdoor ontstaat een centrale plaats waar data uit verschillende bronnen samenkomen.

Het Ometa framework maakt gebruik van zelf ontwikkelde interfaces om verbinding te maken met verschillende applicaties zoals Salesforce en SAP. Hierdoor kunnen organisaties hun bestaande systemen blijven gebruiken terwijl data efficiënt gedeeld kan worden tussen verschillende afdelingen.

Een belangrijk kenmerk van het Ometa framework is dat het een low-code omgeving is. Dit betekent dat oplossingen relatief snel ontwikkeld en aangepast kunnen worden aan de behoeften van de klant. In veel gevallen kan een oplossing binnen enkele dagen worden geïmplementeerd.

2. Probleem

2.1. Logging van het Framework

In het Ometa Framework wordt er zeer veel gelogd. Bij elke actie die er gebeurt, uitgevoerd door een user of niet, zal er een nieuwe log zijn. Vanwege het grote aantal logs is het moeilijk om de correcte, die het probleem aanduid, te pakken te krijgen.
Hoewel deze logs dus waardevolle informatie bevatten over de werking van het systeem, zijn er momenteel enkele problemen:

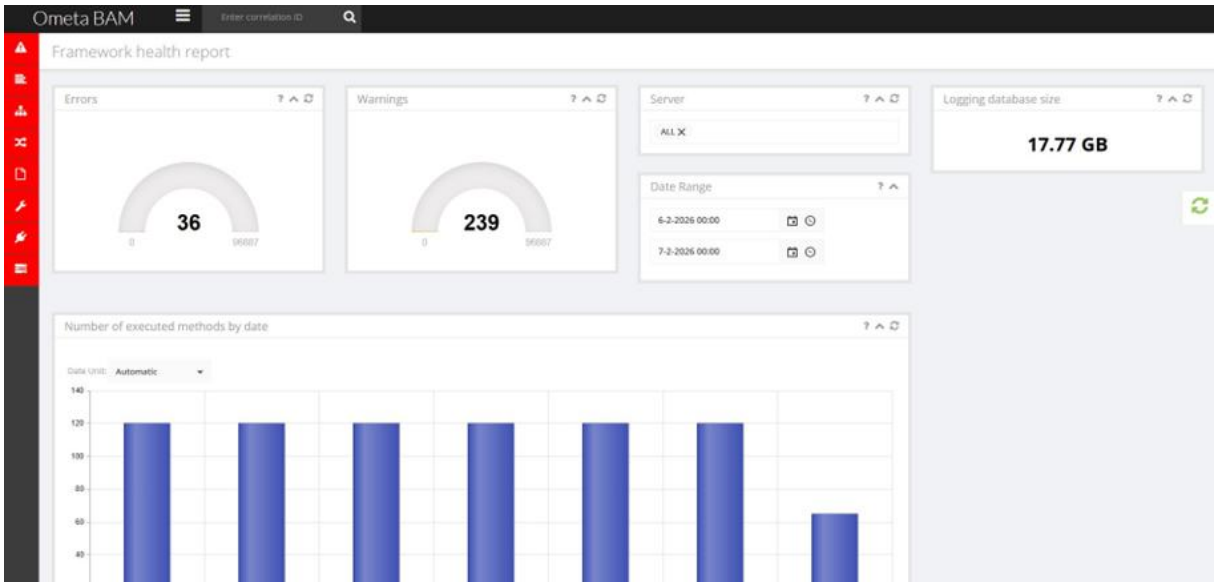
- Veel logberichten bevatten onvoldoende uitleg om snel de oorzaak van een probleem in het framework te begrijpen.
- Er worden zeer veel logs gegenereerd, waardoor belangrijke informatie of context moeilijk te vinden/onderzoeken is.
- Het bestaande dashboard, het **BAM-dashboard**, biedt onvoldoende inzicht in de logs.

2.2. Gegenereerde Rapporten

In een poging om diepere inzichten in de logs te krijgen kan men via het framework een aantal rapporten genereren. Deze rapporten gebruiken de logs om inzichten te krijgen in het aantal error, duurtijd van methodes, duurtijd van case-savings, ... Men genereerd deze rapporten wanneer er zich een probleem in het framework voordoet zoals, functionaliteit die niet werkt of zeer lange wachttijden.
Deze rapporten kunnen hier inzichten in geven. Het probleem is dat hier nog zeer veel data in zit zonder deze op een goede manier te kunnen visualiseren, waardoor het opsporen van problemen veel tijd in beslag neemt. Ook nemen ze vaak context weg van wat er gebeurde in het framework, dit is zeer belangerijk om de oorzaak van problemen in het framework op te sporen.

2.3. Huidige BAM-Dashboard

Er bestaat reeds een dashboard om deze logs te bekijken en er diepere inzichten in te krijgen. Dit dashboard heet het ‘BAM-Dashboard’. In het dashboard kan men filteren op verschillende profielen, objecten, methodes, ... van het framework. Zo kan er dieper ingezoomed worden op waar het probleem in het framework zich voordoet. Dit maakt het een betere optie dan de rapporten.
Helaas zijn de gebruikte visualisaties in het dashboard niet optimaal. Vaak geven ze geen diepere inzichten en nemen ze, net zoals de rapporten, weg van de context van wat er gaande is op dat moment. Ook kunnen de laadtijden, van het inladen van de data, erg lang zijn, waardoor veel gebruikers afhaken en in de logs in de database zelf werken.



2.4. Conclusie

Het hele proces om de oorzaak van een probleem in het framework te vinden kan zeer tijdrovend zijn. De huidige methode van werken is zeer inefficient. Ometa wil hun klantenservice verbeteren door sneller te kunnen reageren op klachten maar ook door proactief verbeteringen in het framework door te voeren. Een dashboard dat echte inzichten kan geven in de logging die gebeurt en hier dan ook proactief op te werken, kan de klantenservice zeer veel verbeteren. Ook zal het tijd vrijmaken voor medewerkers om zicht te focussen op wat er echt belangrijk is.

3. Opdracht

De opdracht van dit project is een **verbeterde oplossing ontwikkelen voor het analyseren en visualiseren van logging data,** zodat problemen sneller opgespoord en opgelost kunnen worden.

3.1. BAM-Dashboard

Binnen dit project zal een **vernieuwde versie van het BAM-dashboard** ontwikkeld worden.
Dit dashboard moet de logging data van het Ometa framework op een duidelijke en efficiënte manier visualiseren. Het doel is om medewerkers sneller inzicht te geven in mogelijke problemen en trends binnen het framework.
De belangrijkste onderdelen van het project zijn:

- Het analyseren van de huidige logging structuur.
- Het ontwerpen van een **nieuw datamodel** dat enkel relevante data bevat, maar toch historische data bewaart.
- Het filteren of structureren van logging data zodat belangrijke informatie beter zichtbaar wordt.
- Het ontwikkelen van duidelijke en betekenisvolle visualisaties.
- Het verbeteren van de performantie van het dashboard zodat laadtijden beperkt blijven.
- Het onderzoeken van mogelijke toepassingen van **AI of machine learning** om patronen of afwijkingen in logging data automatisch te detecteren.

Het uiteindelijke resultaat moet een dashboard zijn dat medewerkers van Ometa ondersteunt bij het **sneller identificeren van fouten en problemen binnen het framework**.

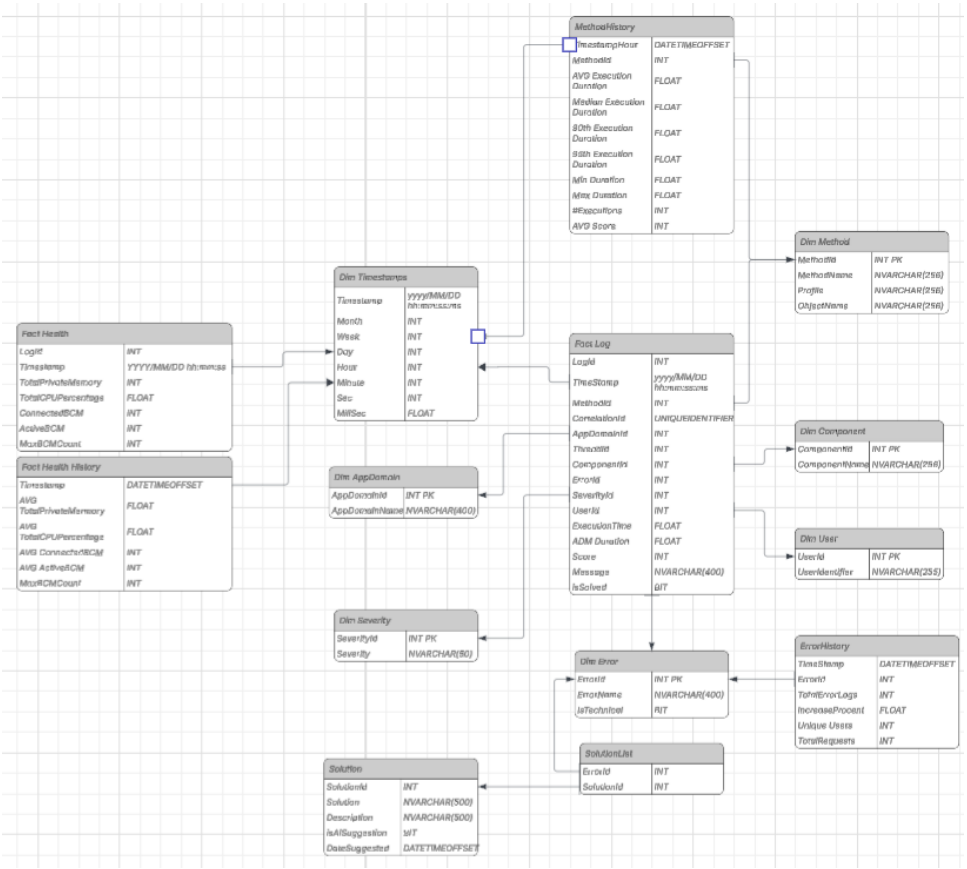
Mijn deel in dit project is de database en het backend. Er zal een diepe analyse op de huidige data moeten gebeuren om uit te zoeken wat er belangrijk in de data is, of we dit op een betere manier beschikbaar kunnen stellen en of we meer historische data kunnen bijhouden.Er zal een backend opgestelt moeten worden om de data in de database beschikbaar te stellen aan de frontend, hierbij zal er ook bekeken moeten worden welke berekeningen waar moeten gebeuren, in de database, in het backen of op het frontend.

3.2. Database

Om te voorkomen dat er onnodig veel data wordt opgeslagen, terwijl er toch meer inzichten uit de data gehaald moeten worden, zal er een nieuw datamodel ontwikkeld moeten worden. Voor de structuur van dit model is het aangewezen om een **stermodel (star schema)** te gebruiken.

Een stermodel bestaat uit een centrale **fact table**, waarin de meetbare gegevens worden opgeslagen, en meerdere **dimensietabellen** die extra context geven aan deze gegevens (bijvoorbeeld tijd, gebruiker of type actie). Deze structuur is bijzonder geschikt voor analytische toepassingen en data-analyse.

Aangezien er veel analyses op de data uitgevoerd moeten worden, is het belangrijk dat de database performant blijft. Een stermodel is hiervoor ideaal omdat het de complexiteit van queries vermindert en het uitvoeren van analytische queries versnelt. Bovendien biedt deze structuur ook voordelen op vlak van opslag en onderhoud, doordat gegevens logisch worden opgesplitst in feiten en dimensies.



Om de data in het dashboard relevant en up-to-date te houden, zal er elk uur een migratie plaatsvinden van de huidige **BAM data table** naar het nieuwe stermodel. Tijdens dit proces wordt de nieuwe data verwerkt en correct verdeeld over de fact- en dimensietabellen. Het is hierbij belangrijk dat dit proces de werking van het bestaande framework zo weinig mogelijk beïnvloedt.

Om dit te realiseren zullen de scripts die hiervoor nodig zijn opgeslagen worden als **stored procedures** in de database. Stored procedures zijn vooraf gedefinieerde SQL-scripts die in de database worden opgeslagen en die herhaaldelijk kunnen worden uitgevoerd. Dit biedt verschillende voordelen. Zo kunnen stored procedures de performantie verbeteren omdat ze vaak al gecompileerd zijn en door de database gecachet kunnen worden. Daarnaast zorgen ze ook voor een beter beheer van de logica, omdat alle verwerking op een centrale plaats wordt opgeslagen.

Deze stored procedures kunnen vervolgens automatisch uitgevoerd worden met behulp van **een Job in SQL Server Agent.** SQL Server Agent laat toe om taken op vaste tijdstippen te laten uitvoeren. In dit geval zal er elk uur een job worden gestart die de stored procedures uitvoert en zo de data uit de BAM tabel verwerkt en migreert naar het nieuwe stermodel. Op deze manier blijft de data in het dashboard steeds actueel, zonder dat dit de normale werking van het framework verstoort.

De migratie van de data zal bovendien in **batches** gebeuren. Dit wordt gedaan om te voorkomen dat er langdurige blokkades op de betrokken tabellen ontstaan. Door de data in kleinere batches te verwerken, blijft de belasting op de database beperkt en kunnen andere processen binnen het framework blijven functioneren zonder merkbare vertraging.

3.3. Backend

Naast het ontwikkelen van het datamodel zal er ook een **backend** ontwikkeld moeten worden die de communicatie tussen de database en het dashboard verzorgt. Deze backend zal geïmplementeerd worden als een **API geschreven in .NET 8**. Door gebruik te maken van een API kan de data op een gestructureerde en veilige manier beschikbaar gemaakt worden voor het dashboard of andere toepassingen.

De backend zal gebruikmaken van **modellen die gebaseerd zijn op het stermodel** dat in de database werd ontworpen. Deze models stellen de verschillende fact- en dimensietabellen voor en zorgen ervoor dat de data op een consistente manier kan worden opgehaald en verwerkt binnen de applicatie. Op deze manier blijft de structuur van de backend nauw afgestemd op de structuur van de database.

Voor het ophalen van de data zal de API werken met **aparte endpoints per grafiek of visualisatie** in het dashboard. Elke grafiek zal dus een specifieke API-call uitvoeren die enkel de relevante data ophaalt. Dit zorgt ervoor dat de hoeveelheid data die wordt verstuurd beperkt blijft en dat het dashboard performant kan blijven werken. Bovendien maakt deze aanpak het eenvoudiger om nieuwe grafieken of analyses toe te voegen zonder dat de bestaande functionaliteit aangepast moet worden.

Daarnaast zal de backend ook fungeren als **verbindingslaag tussen de AI-component en de database**. De AI kan via de API toegang krijgen tot de benodigde data uit het stermodel en deze gebruiken voor verdere analyse of het genereren van inzichten. De backend zorgt hierbij voor een gecontroleerde toegang tot de database en kan eventueel extra logica toevoegen, zoals filtering, validatie of aggregaties.
Door deze architectuur wordt een duidelijke scheiding gecreëerd tussen **data opslag, verwerking en presentatie**, wat de onderhoudbaarheid en uitbreidbaarheid van het systeem ten goede komt.

4. Business Case

Dit project kan op verschillende manieren waarde creëren voor Ometa.
Ten eerste zal een verbeterd dashboard ervoor zorgen dat problemen in het framework **sneller geïdentificeerd en opgelost kunnen worden**. Hierdoor kan de klantenservice efficiënter werken.
Daarnaast kan het dashboard ook helpen om **proactief problemen te detecteren**, nog voordat klanten er last van ondervinden. Dit kan de kwaliteit van de dienstverlening verbeteren.
Een ander voordeel is dat medewerkers minder tijd moeten besteden aan het manueel doorzoeken van logs of databases. Hierdoor kunnen zij zich meer focussen op taken met een hogere toegevoegde waarde.
Samengevat kan dit project bijdragen aan:

- betere inzichten in de werking van het framework
- snellere probleemdetectie
- efficiëntere klantenservice
- tijdsbesparing voor medewerkers

5. Stakeholders

Binnen dit project zijn verschillende stakeholders betrokken.

- **Het Ometa Team**

Het team van Ometa spendeert veel tijd om problemen op te zoeken en dan ook op te lossen, door middel van dit dashboard kan er zeer veel tijd vrijgemaakt worden voor belangrijkere taken.

- **Klanten van Ometa**

Sommige klanten maken ook zelf gebruik van het dashboard. Ze kunnen zelf hun oplossing uitbreiden en met behulp van dit dashboard ook monitoren.

6. Risico's

Tijdens het project kunnen verschillende risico's optreden.

- **Complexiteit van de logging data**

  De grote hoeveelheid logs kan het moeilijk maken om relevante informatie te filteren en te structureren. Ook is het een risico dat bij het bijhouden van historische data de database nog verder zal exploderen dan het nu al doet. Klanten kunnen vaak hun opslag niet groter maken. Vanuit Ometa is opgelegd dat de extra data die wij zullen aanmaken onder 10% van de huidige grootte moet zijn.

- **Performantieproblemen**

  Bij grote hoeveelheden data kan het dashboard trager worden als het datamodel niet efficiënt ontworpen is. Hierdoor zullen lange laadtijden op het dashboard niet verholpen worden. Ook zal de druk op de server te hoog kunnen worden door de hoeveelheid data en de complexiteit van de aanvragen naar de database. Onze applicatie zou de werking van het Ometa-framework en andere systemen op dezelfde server kunnen beïnvloeden, wat absoluut niet de bedoeling is.

- **AI-integratie**

  Indien AI gebruikt wordt voor analyse van logs, kan het een uitdaging zijn om betrouwbare resultaten te verkrijgen.

Deze risico's kunnen beperkt worden door regelmatig te testen, iteratief te ontwikkelen en feedback te verzamelen van gebruikers. Duidelijkheid in de logging data kan deels door werknemers van Ometa gebracht worden alsook het exploreren van de data.
Om te voorkomen dat de afwezigheid van AI heel het project in het water laat vallen, zullen er vervangende visualisaties en berekeningen aangebracht worden om zo goed mogelijke inzichten te creëren.

7. Timeline

7.1. Week 1-2 Analyse

In de eerste en tweede week van de stage zal ik me focussen op de analyse van het project. We leren het project kennen, maar ook het bedrijf en op wie onze oplossing effect zal hebben. De data waarover ons project gaat zal geëxploreerd moeten worden. Ook wat voor data model het beste voor onze oplossing is en hoe we hiervan gebruik kunnen maken.

7.2. Week 3-4 Database/Analyse Backend

In week drie en vier ligt de focus al op het realiseren van de database en scripts om de datatabellen te populeren. Ook zal er nu een diepere analyse op de backend gebeuren, wat voor API we gaan gebruiken, welke calls we nodig hebben, maar ook natuurlijk welke technologieën we gaan gebruiken. API calls zullen per grafiek ontworpen worden zodat de verwerking op de frontend minimaal blijft.

7.3. Week 5-10

Nu de analyse is afgerond zitten we volledig in de ontwikkelingsfase. De database is al grotendeels in orde en we focussen ons nu vooral op de backend. API calls worden ontwikkeld en geconsumeerd door de frontend. Er wordt gekeken naar performantie zodat de server niet te veel beïnvloed wordt. Aanpassingen worden gemaakt in de ontworpen endpoints om een zo goed en snel mogelijk werking te verwezenlijken.

7.4. Week 11-13

De laatste fase van de stage. Het project is grotendeels werkende, er wordt gefocust op het oplossen van bugs en verbeteringen door te voeren, deels voorgesteld door stakeholders. Ook zullen er extra testen plaatsvinden om de invloed van de applicatie te bekijken en beperken. Er wordt gekeken naar het mogelijks installeren bij een klant en hoe dit zou kunnen gebeuren. Documentatie is zeer belangrijk en zal in deze fase nagekeken en verder uitgeschreven worden, zowel van de code zelf en van het opzetten in een omgeving.

8. Rapportering en informatie

Tijdens het project zal er regelmatig gerapporteerd worden over de voortgang.
Dit kan onder andere gebeuren via:

- regelmatige meetings met de stagebegeleider of projectverantwoordelijke

- tussentijdse demo’s van het dashboard
- Projectopvolging aan de hand van Git
- documentatie van het datamodel en de gebruikte technieken
- Wekelijkse rapporten
- een eindrapport waarin het volledige project wordt beschreven

Daarnaast zal alle relevante informatie over het project duidelijk gedocumenteerd worden zodat toekomstige ontwikkelaars het dashboard eenvoudig kunnen begrijpen en verder uitbreiden.

9. Conclussie

Dit project heeft als doel om de analyse van logging data binnen het Ometa Framework duidelijker, sneller en efficiënter te maken. De huidige manier van werken, waarbij medewerkers gebruikmaken van logs, rapporten en het bestaande BAM-dashboard, vraagt veel tijd en biedt niet altijd voldoende context om problemen snel te vinden. Door de grote hoeveelheid logging data is het moeilijk om belangrijke informatie te onderscheiden van minder relevante gegevens.

Met de ontwikkeling van een vernieuwd BAM-dashboard, een nieuw stermodel in de database en een .NET 8 backend wordt er een oplossing voorzien die beter aansluit bij de noden van Ometa. Het stermodel zorgt ervoor dat historische data op een efficiënte manier kan worden bijgehouden, zonder de database onnodig sterk te laten groeien. De backend maakt deze data op een gestructureerde manier beschikbaar voor de frontend, zodat visualisaties snel en gericht de juiste informatie kunnen tonen.

Daarnaast biedt dit project ook kansen om proactiever te werken. Door trends, fouten en afwijkingen sneller zichtbaar te maken, kunnen problemen mogelijk worden opgespoord voordat klanten hier hinder van ondervinden. Dit kan de klantenservice verbeteren en medewerkers helpen om minder tijd te verliezen met het manueel doorzoeken van logs.

Samengevat kan dit project een duidelijke meerwaarde bieden voor Ometa. Het vernieuwde dashboard moet niet alleen zorgen voor betere inzichten in de werking van het framework, maar ook voor snellere probleemdetectie, betere ondersteuning van klanten en een efficiëntere manier van werken voor de medewerkers.