THOMAS MORE

Refelectie Stage Ometa

Sens van Aert
Student Bachelor Artificial Intelligence

2025 – 2026

Table of Contents

1. Stage opdracht

Mijn stage heeft plaatsgevonden bij Ometa, zij leveren een framework aan hun klanten waarmee ze verschillende informatiesystemen kunnen verbinden zoals SAP, SharePoint, Salesforce en nog vele anderen. In dit framework wordt elke actie die uitgevoerd wordt, al dan niet door een gebruiker, als log opgeslagen in een database.
Inzicht verkrijgen in deze logs is niet gemakkelijk. Als er fouten gemeld worden, zoals verkeerde verwerking van informatie of een vertraagde werking in het framework, wordt er gebruik gemaakt van deze logs om het probleem op te lossen.
De huidige manier van analyseren door middel van Excel-rapporten en ook een bestaand dashboard is onoverzichtelijk, traag en moeizaam.
Mijn taak, samen met 3 andere studenten, was het creëren van een dashboard waarop de analyse van de logs, en dus het oplossen van problemen, gemakkelijker wordt gemaakt. Bijkomend zou het dashboard een proactieve werking kunnen toepassen in plaats van een reactieve werking. Hierdoor zou men problemen kunnen oplossen voordat de klant er last van heeft en ook problemen in de code kunnen voorkomen.

2. Realisatie

In dit hoofdstuk zal ik de effectieve realisatie beschrijven die ik heb gemaakt. Zoals ik al vermeldde, heb ik samen met drie andere studenten aan dit project gewerkt. Mijn taak binnen het project was het bijhouden van historische data en het toegankelijk maken van de data. Dit heb ik gedaan door een stermodel te creëren in de database waarin de logs opgeslagen worden. Om de data toegankelijk te maken heb ik een .NET 8 API gebouwd.

2.1. Database

Het grootste probleem met de huidige manier van werken is dat er maar data van enkele dagen beschikbaar is en dat deze data niet snel toegankelijk is. Dit probleem kan voor een groot deel in database opgelost worden.
Dit heb ik gedaan door middel van het creëren van een stermodel. Dit stermodel maakt gebruik van de data in de logs om ingevuld te worden. Elk uur zal er een script draaien dat deze data migratie uitvoert. Van de huidige data zal er, afhankelijk van de hoeveelheid data die er gelogd wordt, ongeveer 1% bijgehouden worden. Dit maakt dat we langer data kunnen bijhouden zonder de database extreem te laten groeien. Deze 1% kunnen we bereiken door enkel het hoogstnoodzakelijke op te slaan, zoals errors, methode-uitvoeringen en gezondheidsrapportering van het framework, en door gebruik te maken van de dimensies waardoor bepaalde waarden niet dubbel opgeslagen moeten worden.

Eigen aan een stermodel is dat het goed is voor analyses, door middel van de dimensies die gekozen zijn zullen we snel informatie over een bepaald onderdeel van het framework efficiënt kunnen opvragen.

In het migratiescript zal er ook een scoreberekening gebeuren voor elke methode-uitvoering, deze beschrijft hoe de methode zich gedraagt (in tijd van uitvoering) ten opzichte van vorige uitvoeringen.

Voor elk uur zal er een samenvatting gemaakt worden per soort error, per soort methode en van de algemene gezondheid van het framework.

2.2. Backend API

De backend API is gemaakt in .NET en heeft als doel het doorgeven van de data van de database naar de frontend. Hiervoor zijn er endpoints voorzien die zo dicht mogelijk aansluiten bij de grafieken op de frontend, zodat hier nog maar minimale verwerking moet gebeuren. Aan elke route zijn filters voorzien waarmee we de tijdspanne waarin we data willen zien kunnen opvragen. Om overbelasting van de server te voorkomen, zal er telkens een limiet staan op de opgevraagde data. Dit kan in de vorm zijn van een maximale tijdspanne, bijvoorbeeld twee dagen, of een maximum aantal records. Dit is vaak 1.000.000.

Andere filters zijn de dimensies in het stermodel. Deze zijn niet beschikbaar op elke route. Dit komt doordat we toegang willen tot de originele logs, die niet verwerkt zijn in het migratiescript. Bij het opvragen van deze data kunnen we niet rechtstreeks gebruikmaken van de dimensies in het stermodel. Ook zal de gezondheidsrapportering niet dezelfde dimensies of filters hebben als de methode-uitvoeringen.

Historische data, in de vorm van de eerder genoemde uurlijkse samenvatting, zal ook toegang hebben tot minder filters vanwege groepering.

Historische data over errors kan niet tijdens de verwerking van het migratiescript worden aangemaakt, omdat er op dat moment nog geen groeperingen van errors bestaan. Elk uur zal er een groepering van errors in de backend gebeuren door middel van een job. Deze job wordt uitgevoerd na de data migratie. Na de groepering zal dan een samenvatting van het afgelopen uur gemaakt worden met behulp van een tweede job die in backend wordt uitgevoerd, het resultaat zal in de database opgeslagen worden.

Een scoreberekening over het algemene welzijn van het framework zal ook met een job gebeuren in de backend. Deze wordt elk uur uitgevoerd. De score is gebaseerd op het aantal errors, de gemiddelde methodescore, het CPU-verbruik en het RAM-verbruik van het framework.

2.3. Performance en testing

Belangrijk bij dit project was het niet overbelasten van de server, maar toch snel leveren van de data. Hierbij speelt het stermodel zoals eerder vermeld een grote rol. De migratie van de data naar dit stermodel moet ook op een efficiënte manier gebeuren zonder overbelasting te veroorzaken. Hier heb ik gebruik gemaakt van MAXDOP, ofwel Maximum Degree of Parallelism. Zo kan er een maximumaantal operaties tegelijk gebeuren, waardoor we de belasting op de server kunnen beperken.
In de backend werd er gebruikgemaakt van een maximum aantal opvraagbare records. Hierbij werd ook gekeken naar de efficiëntie van de gemaakte queries naar de database.

Om de goede werking van de verschillende endpoints te garanderen, zijn er unit tests ontwikkeld die deze werking valideren.

Om de effectieve belasting op de server te testen, werden er load tests uitgevoerd met JMeter, tijdens deze tests keek ik met Logman welke processen de meeste CPU en RAM in beslag namen. Hier zijn niet veel duidelijke conclusies getrokken. Dit moet in de toekomst verder bekeken worden.

3. Verdere ontwikkeling

De verdere ontwikkeling van de applicatie zal bestaan uit het proactiever maken van de applicatie. Dit gebeurt al deels door de scores die berekend worden. We kunnen problemen zien opkomen voordat de klant het merkt, maar de problemen zijn dan wel al aanwezig. Dit kan beter met de ontwikkeling van Iljas, een student die tijdens zijn stage onderzoek deed naar onder andere het voorspellen van anomalieën. Zijn

oplossing is nog niet geïntegreerd in de applicatie. Deze integratie zou de applicatie proactiever kunnen maken.
Verder zou er nog meer naar de load testing gekeken moeten worden om zeker te zijn dat de impact van onze applicatie zo klein mogelijk is en niet merkbaar is voor de rest van het framework.
Ook de eenvoudige installatie bij de klant moet verder bekeken worden. Nu is er redelijk veel configuratie nodig. Dit zal geminimaliseerd moeten worden.

4. Persoonlijke reflectie

Tijdens mijn stage bij Ometa heb ik zeer veel geleerd. Ik heb gekende technologieën in de praktijk kunnen toepassen, maar ook kennisgemaakt met nieuwe technologieën en manieren van werken. MySQL is zeer bekend voor mij, net als het maken van een stermodel, maar de echte voordelen van het stermodel waren voor mij nog moeilijk te zien. Dit werd snel duidelijk toen ik tijdens de ontwikkeling zag hoeveel sneller onze oplossing was ten opzichte van de bestaande oplossing. Ook in het beperken van de opgeslagen data kon ik zien hoeveel een stermodel bespaarde door veelvoorkomende waarden niet dubbel op te slaan.

Met .NET had ik nog nooit eerder gewerkt. De syntax was volledig onbekend voor mij. Ik merkte dat de logica die ik al had toegepast bij bijvoorbeeld een oplossing in Java, hier ook nog altijd gebruikt kon worden, weliswaar met een andere syntax. Het werken met entiteiten was ook al bekend voor mij, maar DTO's, of de vorm die teruggestuurd moet worden, waren nieuw. Dit was niet opgelegd door de stagegever, maar ik zag hier zelf wel de meerwaarde van.

Het maken van jobs of scripts die elk uur worden uitgevoerd was ook nieuw voor mij.

Ook heb ik tijdens de stage de kans gehad om contact te hebben met klanten van Ometa, om hier problemen op te lossen en ook om onze applicatie bij hen op te zetten. Dit heeft me geleerd om in een professionele omgeving met klanten om te gaan. Het heeft me ook geleerd om op een niet-technische manier te praten over technische problemen. Denk bijvoorbeeld aan vragen zoals: "Op welke knop druk je in de applicatie op dit moment?" Zo kan je toch een technisch probleem vinden dat de klant zelf niet kent. Hier kan ik nog wel beter in worden, vind ik zelf.

Ik heb mijn stage als zeer leerrijk ervaren en vond het ook een leuke ervaring. Ik had goede ondersteuning van mijn medestudenten, maar ook zeker van de collega's van Ometa. Daar ben ik hen zeer dankbaar voor!