



Ontwikkeling van een Efficiënte en Gebruiksvriendelijke Foto-Management Applicatie voor Qubris

Realisatie

**Bachelor in de Toegepaste Informatica
keuzerichting APP**

Murrel Venlo

Academiejaar 2023-2024

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

Inhoudsopgave

1	INTRODUCTIE	4
1.1	Het stagebedrijf	5
1.2	De stageopdracht.....	6
2	VERWACHTING.....	7
2.1	Technische vereisten en functionaliteiten	7
3	ONDERZOEKSFASE	8
3.1	Opslagmethode	8
3.2	Bevindingen	8
3.3	Toelichting scores	9
3.4	Opslagkeuze	9
3.5	Aanpak voor opslag.....	10
4	PLANNINGSFASE.....	11
4.1	Technologieën en hulpmiddelen.....	11
4.1.1	Use-Case Diagram	11
4.1.2	Front-end.....	11
4.1.3	Backend	12
4.1.4	Database	12
4.1.5	Datamodeltering	13
4.2	Design.....	13
4.2.1	Foto's bekijken	14
4.2.2	Foto's opslaan	14
4.2.3	Foto raadplegen.....	15
4.2.4	Foto aanpassen.....	16
4.2.5	Foto verwijderen	16
4.3	Planning.....	17
4.3.1	Trello Board Structuur	17
5	ONTWIKKELINGSFASE	19
5.1	De Backend	19
5.2	De Frontend	21
5.2.1	Het inlogscherms.....	21
5.2.2	Het Dashboard.....	22
5.2.3	Menu items	22
5.2.4	Afbeelding toevoegen	22
5.2.5	Afbeelding bekijken	24
5.2.6	Afbeelding bewerken en verwijderen.....	25
6	CONCLUSIE	26
7	BRONNEN.....	27

1 INTRODUCTIE

Dit document presenteert de realisatiefase van een innovatieve applicatie die ontwikkeld wordt in opdracht van Qubris. Het doel van het project is het toevoegen en beheren van foto's op een efficiënte en gebruikersvriendelijke manier. De applicatie stelt gebruikers in staat om foto's toe te voegen aan verschillende entiteiten, ze op te slaan, te raadplegen, te bewerken en te verwijderen. Dit verslag beschrijft het gehele ontwikkelingsproces van begin tot eind, inclusief de uitdagingen en oplossingen die tijdens het project zijn aangetroffen.

1.1 **Het stagebedrijf:** In deze sectie wordt een korte introductie gegeven van het stagebedrijf, Qubris.

1.2 **De stageopdracht:** Dit hoofdstuk beschrijft de algemene doelstelling van de stage, namelijk het ontwikkelen van een applicatie voor het opslaan en beheren van foto's.

2. **Verwachtingen:** In dit hoofdstuk worden de verwachtingen voor het eindproduct besproken, met de nadruk op gebruiksvriendelijkheid, efficiëntie en betrouwbaarheid.

3. **Onderzoeksfase:** Dit hoofdstuk behandelt het onderzoek naar verschillende opslagmethoden voor foto's en de selectie van MongoDB als de meest veelbelovende optie.

4. **Planningsfase:** Hier wordt beschreven hoe de technologieën en tools zijn gekozen voor zowel de backend als de frontend, evenals het maken van een gedetailleerd ontwerp van de applicatie.

5. **Ontwikkelingsfase:** Dit hoofdstuk omvat de implementatie van zowel de backend als de frontend, met een focus op het voldoen aan de gestelde eisen door middel van een gestructureerde aanpak.

6. **Conclusie:** Een samenvatting van de resultaten en geleerde lessen.

7. **Bronnen:** Een overzicht van de geraadpleegde bronnen.

Dit document biedt een uitgebreide gids van het hele ontwikkelingsproces, van het onderzoeksproces tot de uiteindelijke oplevering van de applicatie, met een nadruk op de geleerde technieken en aanbevelingen voor toekomstig werk.

1.1 Het stagebedrijf

Wie is Qubris?

Qubris werd op 10-12-2010 opgericht onder de naam 'DIMENSYS BELGIUM'. Op 30 december 2021 werd de naam veranderd in 'Qubris'. (www.staatsbladmonitor.be)

Qubris heeft 24 jaar ervaring in productieprocessen, de absolute specialiteit van de interne technisch adviseurs en experts!

Qubris heeft het statuut van een besloten vennootschap (BV).
In 2023 waren er 13,4 FT- werknemers in dienst (www.jaarrekening.be).

Wat doet Qubris

Qubris richt zich op het vervaardigen van IT-oplossingen

Qubris adviseert nationale en internationale productiebedrijven op het gebied van software-implementatie, productie en rapportage.

Met de strategische ontwikkeling van MES (Manufacturing Execution System) systemen – out-of-the-box en afgestemd op de productieprocessen – verhogen ze de kwaliteit van de beschikbare data binnen een bedrijf en creëren zo een dieper inzicht in de onderliggende processen. Hiermee streven ze, in samenwerken met de klant, naar een efficiënter proces en kwalitatiever product.

Softwareontwikkeling

Qubris ontwikkelt flexibele en configureerbare software die is aangepast aan de specifieke behoeften van productiebedrijven. Deze software integreert naadloos met bestaande ERP-systemen.

De integratie van productielijnen met een Manufacturing Execution System (MES) verbetert de efficiëntie. MES is een zeer configureerbaar softwarepakket dat gemakkelijk te integreren is met onder andere ERP-pakketten. Dit resulteert in een snellere implementatie dankzij een verscheidenheid aan applicaties.

De app Trendwatch helpt organisaties bij het analyseren van de kwaliteit van producten en productieprocessen. Het stelt hen ook in staat om diepere oorzaken van problemen te onderzoeken. Hierdoor wordt bijgedragen aan efficiëntie en duurzaamheid in de productieprocessen.

Gegevens worden automatisch vastgelegd en omgezet in informatie. Deze concrete antwoorden op vragen over het productieproces dragen bij aan een optimaler productieproces en een verhoogde kwaliteit van zowel de producten als de processen.

1.2 De stageopdracht

De opdracht is om een applicatie te ontwikkelen die gebruikers in staat stelt om foto's op een efficiënte en gebruiksvriendelijke manier op te slaan, te koppelen aan verschillende entiteiten, zoals bijvoorbeeld machines of producten, en deze later te kunnen bewerken. De applicatie moet in staat zijn om gebruikers te laten authenticeren, foto's op te slaan in een SQL-database en deze op een overzichtelijke manier weer te geven. De applicatie neemt de vorm van een plug-in aan. Op die manier is de functionaliteit eenvoudig aan andere bestaande applicaties toe te voegen voor het beheren van foto's van verschillende entiteiten. De nadruk ligt dus vooral op de herbruikbaarheid en uitbreidbaarheid van de plug-in.

Verbeterde informatievoorziening

Het voornaamste doel van de applicatie is om gebruikers van Qubris-applicaties in staat te stellen om gebruik te maken van grafische informatie binnen de applicatie, door middel van foto's en afbeeldingen. Dit impliceert dat gebruikers gemakkelijk foto's moeten kunnen toevoegen, opslaan, en later kunnen bewerken binnen de applicatie.

Efficiënte opslag van foto's

Een belangrijk subdoel is om een efficiënte opslagmethode voor foto's te ontwikkelen. Dit houdt in dat de foto's op een gestructureerde en geoptimaliseerde manier moeten worden opgeslagen, zowel binnen een SQL-database als op een externe locatie, om de prestaties van de applicatie te verbeteren en de benodigde opslagruimte te minimaliseren.

Authenticatie-functionaliiteit

Om aan alle technische vereisten te voldoen in zaken beveiliging en autorisatie, is het noodzakelijk dat gebruikers zich kunnen authenticeren binnen de applicatie. Zo moet de gebruiker de optie hebben om in te loggen op de applicatie. De bevoegdheden worden uitgedeeld op basis van beveiligingsrollen die toegekend kunnen worden aan de verschillende gebruikers.

2 VERWACHTING

De verwachting is dat de ontwikkelde applicatie gebruikers van Qubris-applicaties voorziet van uitgebreidere en duidelijkere informatie met betrekking tot het efficiënt opslaan en beheren van foto's.

De applicatie moet in staat zijn om gebruikers toe te staan om foto's toe te voegen aan verschillende soorten entiteiten, deze op te slaan en later te bewerken. Dit impliceert een diepgaande analyse van hoe foto's worden opgeslagen en gelinkt in de database, evenals hoe ze worden weergegeven in de applicatie.

Daarnaast is het cruciaal dat de opgeslagen informatie veilig wordt vastgelegd in een SQL-database, met zorgvuldige aandacht voor de opslag en de beveiliging van gegevens.

Gebruikers moeten ook de mogelijkheid hebben om zich te authenticeren in de applicatie om toegang te krijgen tot de functionaliteiten en gegevens.

Bovendien zal de applicatie de vorm aannemen van een plug-in, waardoor het mogelijk wordt om deze te integreren in andere projecten voor het beheren van diverse entiteiten, met nadruk op herbruikbaarheid en uitbreidbaarheid.

2.1 Technische vereisten en functionaliteiten

Om aan de verwachtingen te voldoen, moeten diverse technische vereisten en functionaliteiten in overweging worden genomen. Dit omvat onder andere het opslaan van foto's op elke soort entiteit, met de mogelijkheid tot later bewerken. Aangezien de foto's ook kunnen worden opgeslagen in een andere database, zoals een NoSQL database (bijvoorbeeld MongoDB), moet de applicatie flexibel genoeg zijn om met verschillende soorten databases te kunnen werken.

Naast het opslaan van foto's, is de implementatie van gebruikersauthenticatie van cruciaal belang om de veiligheid van de applicatie te waarborgen. Dit kan worden bereikt door middel van een backend met een REST API, waarmee gebruikers zich kunnen authenticeren voordat ze toegang krijgen tot de functionaliteiten en gegevens van de applicatie.

Voor een optimale gebruikerservaring is de ontwikkeling van een intuïtieve gebruikersinterface van essentieel belang. Hiervoor zal gebruik worden gemaakt van een frontend, hoogstwaarschijnlijk in Vue.js of Nuxt.js, waarbij interactiviteit en reactiviteit centraal staan.

Verder is het belangrijk om onderzoek te doen naar en de implementatie van versiebeheer en de ontwikkelstandaarden binnen het bedrijf te waarborgen. Dit zorgt voor een gestructureerde en georganiseerde ontwikkeling van de applicatie, wat essentieel is voor het behoud van kwaliteit en consistentie.

Het grondig testen van de applicatie is een cruciaal onderdeel om fouten te voorkomen en een soepele werking te garanderen. Na het testen moet de applicatie worden geïmplementeerd in productie onder begeleiding van de stagebegeleider, om er zo voor te zorgen dat alle aspecten van het project correct worden uitgevoerd.

Tot slot moet de applicatie de mogelijkheid bieden tot het ontwikkelen van een plug-in voor integratie in andere projecten. Dit vereist een focus op herbruikbaarheid en uitbreidbaarheid, zodat de applicatie flexibel genoeg is om te worden geïntegreerd in verschillende omgevingen en systemen.

3 ONDERZOEKSFASE

Eén van de belangrijkste onderdelen in de wereld van softwareontwikkeling is het onderzoeken van de meest geschikte technische oplossingen en architecturen.

Ook in dit project staat onderzoek voorop. Met behulp van een gestructureerde aanpak zijn verschillende methodes voor het opslaan van afbeeldingen onderzocht, geëvalueerd en vergeleken. De gevolgde methodiek staat hieronder beschreven.

3.1 Opslagmethode

Uit markt onderzoek bleken er verschillende soorten databases en opslag methodes zich te lenen voor het opslaan van foto's. Na een korte verdieping leken de volgende methodes het meest interessant om verder te onderzoeken:

- Opslag van bestandpaden in de database
- Opslag van afbeeldingen als BLOB's in de database
- Opslag van afbeeldingen in een gespecialiseerde opslagservice
- Opslag van afbeeldingen in een NoSql Database:
 - MongoDB (on-premise)
 - CouchDB (on-premise)
- Firestore (cloud)
- CDN

Belangrijke aspecten van de onderzochte methoden zijn onder andere de leercurve, implementatie, schaalbaarheid, beheer, kosten en vooral de prestaties.

3.2 Bevindingen

Allereerst zijn alle methodes gescoord op basis van de verschillende aspecten. Met behulp van een gewogen matrix waarbij het gewicht toegekend aan een aspect het relatief belang uitdrukt, is de totaal score van iedere methode bepaald. Dit leidde tot een objectieve scoring waarbij de hoogte score overeenkomt met de methode die zit het beste leent bij de gestelde problematiek.

Factor	Tota le Score	Koste n	Prestati es	Leercur ve	Schaalbaarh eid	Implementa tie	Installa tie + Beheer
Methode	Wegin g	4	5	2	4	2	4
Bestandspade n		3	2	4	2	3	4
Blob-Storage		4	1	3	2	3	5
Gespecialisee rde service		2	5	2	5	3	2

MongoDB		4	4	3	4	3	3
CouchDB		4	4	3	4	3	3
Firestore		3	4	3	4	3	3
CDN		2	5	2	4	3	2

3.3 Toelichting scores

Opslagkosten: Dit criterium evalueert de verwachte opslagkosten voor afbeeldingen. Een hogere score duidt op waarschijnlijk lagere kosten, wat gunstig kan zijn voor het budget.

Prestatie: Deze score geeft aan hoe goed de methode presteert, vooral wat betreft de snelheid en efficiëntie bij het opslaan en ophalen van afbeeldingen. Een hogere score duidt op betere prestaties, wat cruciaal kan zijn voor de algehele gebruikerservaring van de applicatie.

Leercurve: Een hogere score betekent meer complexiteit, wat kan leiden tot extra tijd en middelen voor ontwikkeling, implementatie en onderhoud.

Schaalbaarheid: Deze geeft aan hoe goed de methode kan omgaan met een toenemend aantal afbeeldingen en groeiende eisen aan de applicatie. Een hogere score wijst op betere schaalbaarheid, wat belangrijk is voor een applicatie die naar verwachting in de loop van de tijd zal groeien.

Implementatie: Dit criterium beoordeelt het implementatiegemak van de methode. Een hogere score betekent dat de methode gemakkelijker te implementeren is, wat tijd en middelen kan besparen tijdens ontwikkeling en implementatie.

Installatie en beheren: De score voor installatie en beheer beoordeelt hoe gemakkelijk het is om de methode bij de klant te installeren en te beheren. Een hogere score geeft aan dat de installatie en het beheer eenvoudiger zijn, wat belangrijk is voor een soepele integratie en bedrijfsvoering.

3.4 Opslagkeuze

De gewogen matrix, weergegeven in bovenstaande tabel, toont de beoordeling van verschillende opslagmethoden op belangrijke factoren zoals kosten, prestaties, leercurve, schaalbaarheid, implementatiegemak en installatiebeheer.

Na evaluatie blijkt MongoDB de hoogste totale score te behalen, gevolgd door CouchDB en Firestore. Dit maakt MongoDB, samen met CouchDB en Firestore, tot de meest veelbelovende opties voor het opslaan en ophalen van afbeeldingen binnen onze applicatieontwikkeling.

De keuze voor MongoDB biedt meerdere voordelen. Zo biedt het een hoge mate van schaalbaarheid, geavanceerde afbeeldingsverwerking en is het relatief eenvoudig te implementeren en te beheren. Deze factoren dragen bij aan een verbeterde gebruikerservaring door snelle laadtijden te garanderen en tijd en middelen te besparen tijdens de ontwikkeling en implementatie.

Naast de technische aspecten moeten ook andere factoren in overweging worden genomen, zoals de beperkingen van Firestore met betrekking tot offline beheer en de slechte community support van CouchDB.

Na deze overwegingen te hebben genomen, is de conclusie dat MongoDB de beste optie is voor de opslag van afbeeldingen. Als de klant echter geen gebruik wil maken van

MongoDB, kunnen we ook werken met de tweede beste optie uit de matrix, namelijk een gespecialiseerde service.

3.5 Aanpak voor opslag

Tot slot is vastgesteld dat voorafgaand aan de implementatie twee tabellen nodig zijn voor de opslag van foto's: één voor de foto's zelf en één voor de koppeling tussen de foto en de entiteit. De tabel voor de foto's bevat essentiële gegevens zoals een unieke ID, de naam van de foto en de gebruiker die de foto heeft opgeslagen. De unieke ID wordt gebruikt om de foto te koppelen aan verschillende entiteiten in de tweede tabel.

4 PLANNINGSFASE

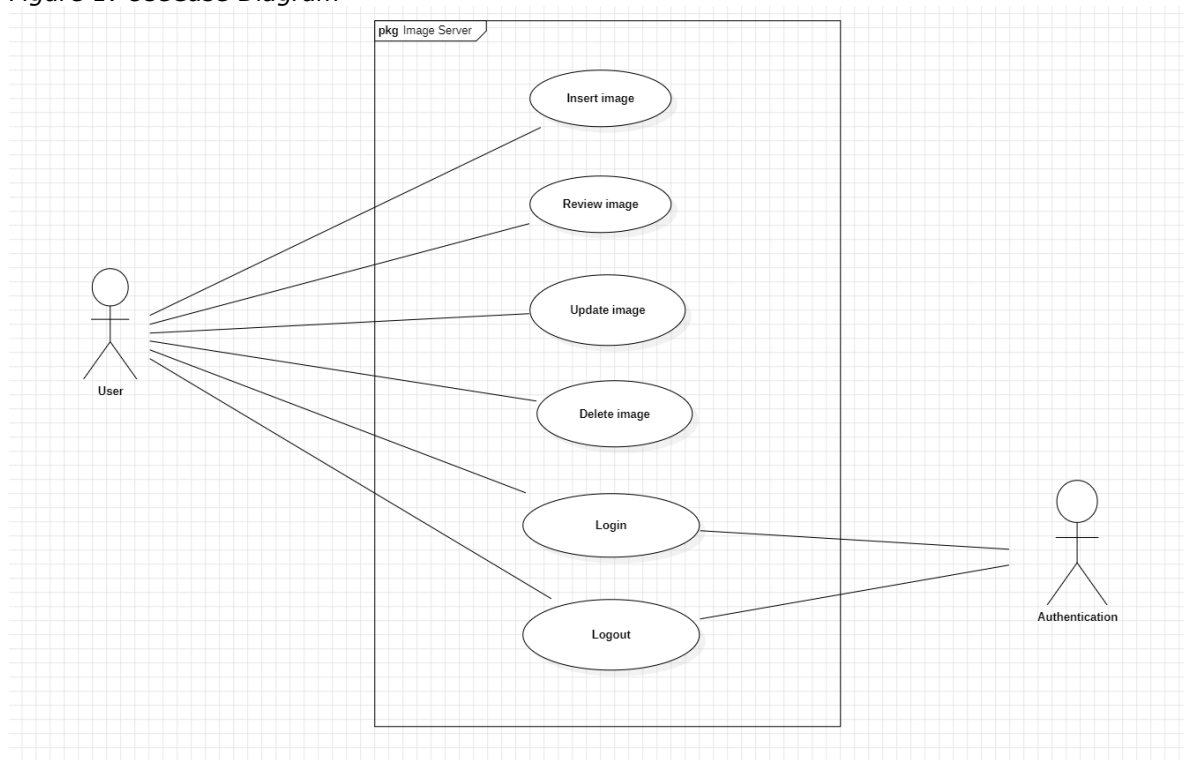
4.1 Technologieën en hulpmiddelen

Voor de uitvoering van de opdracht hebben mijn stagebegeleider en ik samen verschillende technologieën en hulpmiddelen onderzocht en geselecteerd. Vervolgens zijn deze toegepast in zowel de backend als de frontend van de applicatie. Hieronder volgt een korte uitleg van de gebruikte technologieën en hulpmiddelen.

4.1.1 Use-Case Diagram

Volgende diagram beschrijft de belangrijkste use cases. Met behulp van dit diagram kunnen de verschillende functionaliteiten van de applicatie gebruikt worden vanuit het standpunt van de gebruiker.

Figure 1: UseCase Diagram



Zoals u hierboven te zie is, moet de aangemelde gebruiker, foto's kunnen opslaan, bekijken, bewerken en verwijderen. De functionaliteiten zullen later via de prototypes verduidelijkt worden.

4.1.2 Front-end

Het front-end van een applicatie is het deel dat gebruikers kunnen zien. In het kader van deze opdracht, waarbij het voornaamste doel het opslaan van foto's is, is het front-end van essentieel belang voor de gebruikerservaring.

Voor het front-end van de applicatie is er gekozen voor Nuxt.Js, gezien deze framework compatibel is met de bestaande frameworks die bij Qubris worden gebruikt. Vue.js en bij uitbreiding Nuxt.js staan bekend om hun eenvoudige syntax, modulariteit

en krachtige functionaliteiten, waardoor ze ideale keuzes zijn voor het ontwikkelen van moderne en responsieve gebruikersinterfaces.

4.1.3 Backend

Tegenwoordig is het gebruikelijk dat elke grote applicatie een backend heeft vanwege de flexibiliteit die het biedt. Een backend stelt applicaties in staat om gegevens op te slaan, te verwerken en te beheren op een centrale locatie, waardoor de gebruikersinterface soepel kan functioneren. Het is belangrijk omdat het de mogelijkheid biedt om complexe logica uit te voeren, gebruikersgegevens veilig te bewaren, en om interactie met externe systemen mogelijk te maken. Kortom, de backend is essentieel voor het creëren van een robuuste en schaalbare applicatie die aan de behoeften van gebruikers kan voldoen.

Voor deze opdracht is er besloten om .NET als ons framework te kiezen, in lijn met de bestaande praktijken bij Qubris. Het .NET framework staat bekend om zijn robuustheid, veiligheid en uitgebreide mogelijkheden voor het ontwikkelen van schaalbare backend-systemen.

4.1.4 Database

Uit het onderzoek blijkt dat de beste oplossing een NoSql Database is. Na een verdiepende vergelijking komt MongoDB als beste kandidaat uit de bus. Deze keuze is gebaseerd op verschillende belangrijke factoren, waaronder kosten, prestaties, schaalbaarheid en implementatiegemak. MongoDB scoorde hoog op al deze criteria en biedt daarnaast geavanceerde functies voor afbeeldingsverwerking, zoals automatische schaalverkleining en optimalisatie, waardoor het een ideale keuze is voor het efficiënt en effectief opslaan van afbeeldingen.


Figuur 2: MongoDB



4.1.5 Datamodellering

GenericMappingTable		
PK	MappingId	GUID
	TableName	VARCHAR(100)
	KeyValuePairs	VARCHAR(255)
	CreatedBy	VARCHAR(100)
	CreatedAtUtc	DATETIME
	ModifiedBy	VARCHAR(100)
	ModifiedAtUtc	DATETIME
	RowVersion	INTEGER

Figuur A: Erd Diagram voor MSSQ tabel voor het mappen van entiteiten

images		
	_id	objectId NN
	ImageName	string NN
	ImageData	binData NN
	CreatedBy	string NN
	CreatedAtUtc	date NN
	ModifiedBy	string NN
	ModifiedAtUtc	date NN
	RowVersion	int NN
	MappingId	string NN

Figuur B: Erd diagram voor MongoDB collectie voor Foto's

Zoals te zien is in Figuur A en B, is dit geen normaal datamodel. Links zie je de entiteit GenericMappingTable, die verantwoordelijk is voor het creëren van een unieke sleutel per entiteit zonder het database model te verstoren. Dit mapt geen entiteiten met elkaar, maar laat juist toe om foto's toe te kennen aan specifieke entiteiten op basis van de nieuwe gegenereerde ids. Dit is besloten om te voorkomen dat de bestaande applicaties en hun functionaliteit verstoord worden, en om de impact van de plugin op het datamodel te minimaliseren. Met deze nieuwe primaire sleutel is het eenvoudig om elke entiteit te linken aan de verschillende foto's.

4.2 Design

De functionaliteiten van een app worden tot leven gebracht door een goed doordacht gebruikersinterface (UI) ontwerp. Dit hoofdstuk richt zich op het ontwerp van de UI om de functionaliteiten van de app te optimaliseren en de gebruikerservaring te verbeteren.

Van het navigeren tussen schermen tot het uitvoeren van complexe taken, het UI-design speelt een essentiële rol bij het bieden van een intuïtieve en plezierige ervaring voor gebruikers.

Het hoofdstuk richt zich op het onderzoeken van hoe verschillende ontwerpbeslissingen bijdragen aan het efficiënt benutten van de functionaliteiten van de app, met daarbij aandacht voor een naadloze interactie en een aantrekkelijke visuele presentatie.

4.2.1 Foto's bekijken

Functionaliteit: Als gebruiker wil ik foto's kunnen bekijken

Voorwaarde: Je moet ingelogd zijn als gebruiker/klant

Qubris

Entity

Report

John Doe

Entities

Name	DisplayName	Comment
Entity 1	<div></div> DisplayName Entity 1	Lorem Ipsum is simply dummy text
Entity 2	<div></div> DisplayName Entity 2	Lorem Ipsum is simply dummy text
Entity 3	<div></div> DisplayName Entity 3	Lorem Ipsum is simply dummy text

Figuur 5: Scherm met entiteiten

De gebruiker komt terecht op dit scherm en ziet een lijst van entiteiten. Als deze gebruiker op een entiteit klikt komt er een pop-up scherm met gegevens van de geklikte entiteit.

4.2.2 Foto's opslaan

Functionaliteit: Als gebruiker wil ik foto's kunnen opslaan

Voorwaarde: Je moet ingelogd zijn als gebruiker/klant

Figuur 6: Scherm voor het opslaan van een foto

Image modal		
<div>Upload the image here</div> <div>Choose image</div>	Entity 1	DisplayName Entity 1
	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.	

Met behulp vanbovenstaand scherm kan de gebruiker eenvoudig een nieuwe foto toevoegen aan de selecteerde entiteit.

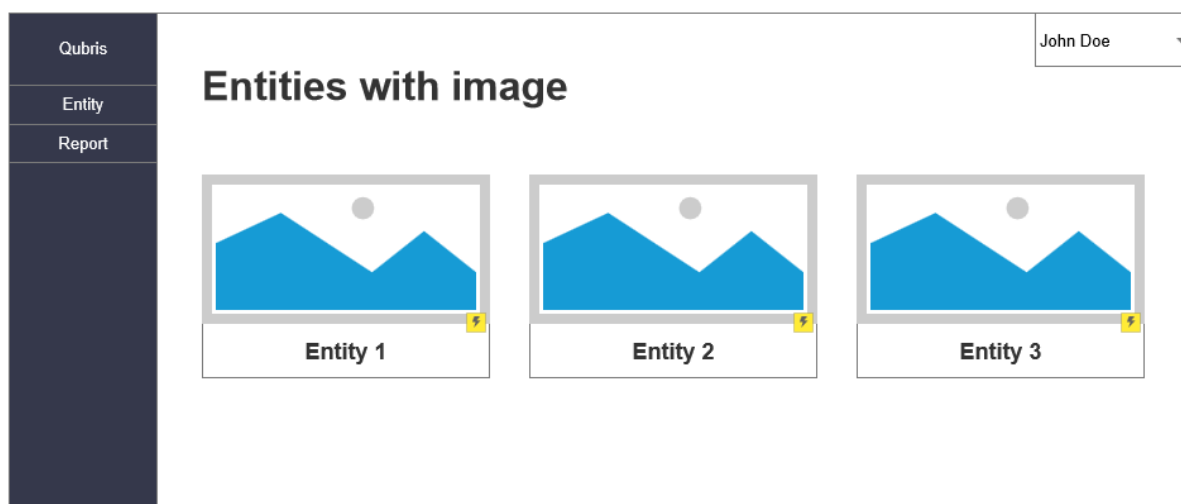
4.2.3 Foto raadplegen

Functionaliteit: Als gebruiker wil ik entiteiten met bijbehorende foto kunnen raadplegen

Voorwaarde: Je moet ingelogd zijn als gebruiker/klant

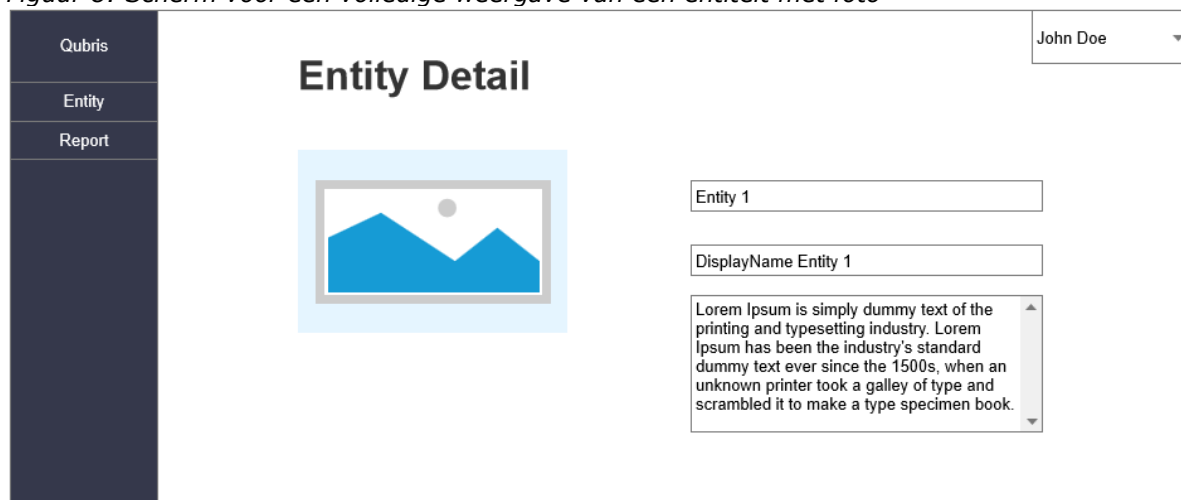
Deze UC is analoog met UC "Review image".

Figuur 7: Scherm om entiteiten te bekijken



De gebruiker kan ook entiteiten met hun bijbehorende foto's zien. Wanneer hij op een entiteit met een foto klikt, kan hij een gedetailleerde weergave van de entiteit bekijken. Hieronder is een voorbeeld van een detail weergave te zien.

Figuur 8: Scherm voor een volledige weergave van een entiteit met foto



4.2.4 Foto aanpassen

Functionaliteit: Als gebruiker wil ik een foto van een entiteit kunnen aanpassen

Voorwaarde: Je moet ingelogd zijn als gebruiker/klant

Deze UC is analoog met UC "Update image".

Figuur 9: Scherm om entiteiten te bekijken en aan te passen

The screenshot shows a web application interface titled 'Entity Detail'. On the left is a dark sidebar with three menu items: 'Qubris', 'Entity', and 'Report'. The 'Entity' item is highlighted. In the top right corner, there is a user profile dropdown showing 'John Doe'. The main content area features a large placeholder image of a blue mountain range. Below the image are two icons: a red trash can and a green pencil. To the right of the image, there are three input fields: 'Entity 1', 'DisplayName Entity 1', and a text area containing Lorem Ipsum text.

Naast het bekijken van een gedetailleerde weergave van een entiteit, biedt het scherm ook de mogelijkheid om een foto van de entiteit bij te werken. Het groene icoontje op het scherm is specifiek bedoeld voor het bijwerken van de foto. Door op dit icoontje te tikken, kunnen gebruikers gemakkelijk een nieuwe foto selecteren of een bestaande foto vervangen, waardoor de entiteit wordt bijgewerkt met de meest recente afbeelding.

4.2.5 Foto verwijderen

Functionaliteit: Als gebruiker wil ik een foto van een entiteit kunnen aanpassen

Voorwaarde: Je moet ingelogd zijn als gebruiker/klant

Deze UC is analoog met UC "Update image".

Figuur 10: Scherm om entiteiten te bekijken en te verwijderen

This screenshot is identical to Figure 9, showing the 'Entity Detail' screen. It includes the same sidebar with 'Qubris', 'Entity', and 'Report' options, the 'John Doe' user profile in the top right, and the main content area with the placeholder image, trash and edit icons, and the input fields for 'Entity 1', 'DisplayName Entity 1', and the Lorem Ipsum text area.

In het scherm met een gedetailleerde weergave van een entiteit is het mogelijk om een foto van de entiteit te verwijderen. Het rode icoontje, dat duidelijk herkenbaar is, is bedoeld voor het verwijderen van de foto. Gebruikers kunnen eenvoudig op dit icoontje tikken om de foto te verwijderen en de entiteit bij te werken.

4.3 Planning

Een planning is opgesteld om het project op een tijdige manier tot een goed einde te brengen. Hierop staan alle mogelijke taken opgesomd in sprints, categorie en urgentie. Met dit document had ik een goede houvast om een planning in Trello te maken voor de ontwikkelingsfase van de opdracht.

Figuur 11: Planning Scrum Document

Scrum document Stageopdracht 25/03/2024						Student: Murrel Venlo		
Id	Sprint	Domain + Epic Story	Sprint planning		Business Value	Estimated "effort" (h)	Execution	
			User Story				Team member(s)	Actual "effort" (h)
1	Sprint 1	Mongodb-data Backend	Integrate MongoDB into the Domain layer	high		1		
2	Sprint 1	Mongodb-data Backend	Integrate MongoDB into the Data layer	high		8		
3	Sprint 1	Mongodb-data Backend	Integrate MongoDB into the Application layer	high		1		
4	Sprint 1	Mongodb-data Backend	Adding CRUD operation for pictures	high		4		
5	Sprint 1	Mongodb-data Backend	Adding Endpoints to application	high		1		
6	Sprint 1	Mongodb-data Backend	Adding request validation	high		4		
7	Sprint 1	Mongodb-data Backend	Testing	high		8		
8	Sprint 1	Mongodb-data Backend	Code review, process feedback & refactoring	high		8		
9	Sprint 2	SSMS-data Backend	Creating API for intermediate Data	high		4		
10	Sprint 2	SSMS-data Backend	Integrate GenericMappingTable into the Domain layer	high		2		
11	Sprint 2	SSMS-data Backend	Integrate GenericMappingTable into the Data layer	high		4		
12	Sprint 2	SSMS-data Backend	Create algorithm to map entities to genericMappingTable	high		8		
13	Sprint 2	SSMS-data Backend	Adding CRUD operation for GenericMappingTable	high		8		
14	Sprint 2	SSMS-data Backend	Adding Endpoints for GenericMappingTable	high		1		
15	Sprint 2	SSMS-data Backend	Adding request validation	high		4		
16	Sprint 2	SSMS-data Backend	Testing	high		8		
17	Sprint 2	SSMS-data Backend	Code review, process feedback & refactoring	high		8		
18	Sprint 3	Integral-data Front-end	Fetch list GenericMappingTable on webpage	high		4		
19	Sprint 3	Integral-data Backend	Add picture to Equipment	high		4		
20	Sprint 3	Integral-data Front-end	Add picture to Equipment on webpage	high		4		
21	Sprint 3	Integral-data Front-end	Dummy view on entities	high		2		
22	Sprint 3	Integral-data Front-end	Link pictures to entities	high		8		
23	Sprint 3	Integral-data Front-end	View picture	high		3		
24	Sprint 3	Integral-data Front-end	Update picture	high		4		
25	Sprint 3	Integral-data Front-end	Delete picture	medium		2		
26	Sprint 3	Integral-data Front-end	Testing	high		4		
27	Sprint 3	Integral-data Front-end	Code review, process feedback & refactoring	high		8		
28	Sprint 4	Expansion Front-end	Allow pictures to other entities (equipmentOnHoldReasons)	high		40		
29	Sprint 4	Integral-data Front-end	Read PackageOnHoldReason	high		4		
30	Sprint 4	Integral-data Front-end	Add picture to PackageOnHoldReason	high		8		
31	Sprint 4	Integral-data Front-end	Get picture per selected PackageOnHoldReason	high		8		
32	Sprint 4	Integral-data Front-end	Update picture per selected PackageOnHoldReason	high		8		
33	Sprint 4	Integral-data Front-end	Delete picture per selected PackageOnHoldReason	high		4		
34	Sprint 5	SSMS-data Backend	Add Validation logic for Authentication	medium		4		
35	Sprint 5	SSMS-data Backend	Crud-Authentication	medium		8		
36	Sprint 5	SSMS-data Backend	Integrate Authentication service	medium		4		
37	Sprint 5	SSMS-data Backend	Add User validation Backend	medium		4		
38	Sprint 5	Integral-data Front-end	Add User validation Frontend	medium		8		

4.3.1 Trello Board Structuur

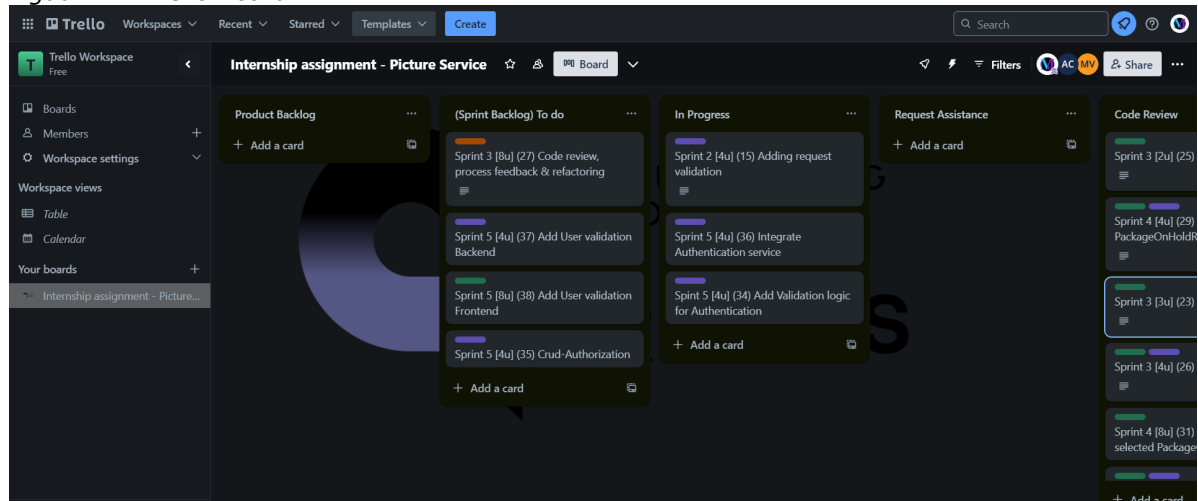
Het Trello-board is georganiseerd in verschillende lijsten om de voortgang van taken te beheren. De structuur is als volgt:

- **Product Backlog:** Hier worden alle toekomstige taken en ideeën verzameld die nog niet zijn ingepland.
- **Sprint Backlog (To do):** Dit is een subset van de Product Backlog, met taken die zijn geselecteerd per sprint en nog moeten worden opgenomen.
- **In Progress:** Taken die momenteel in uitvoering zijn.
- **Request Assistance:** Taken waar hulp of overleg voor nodig is.

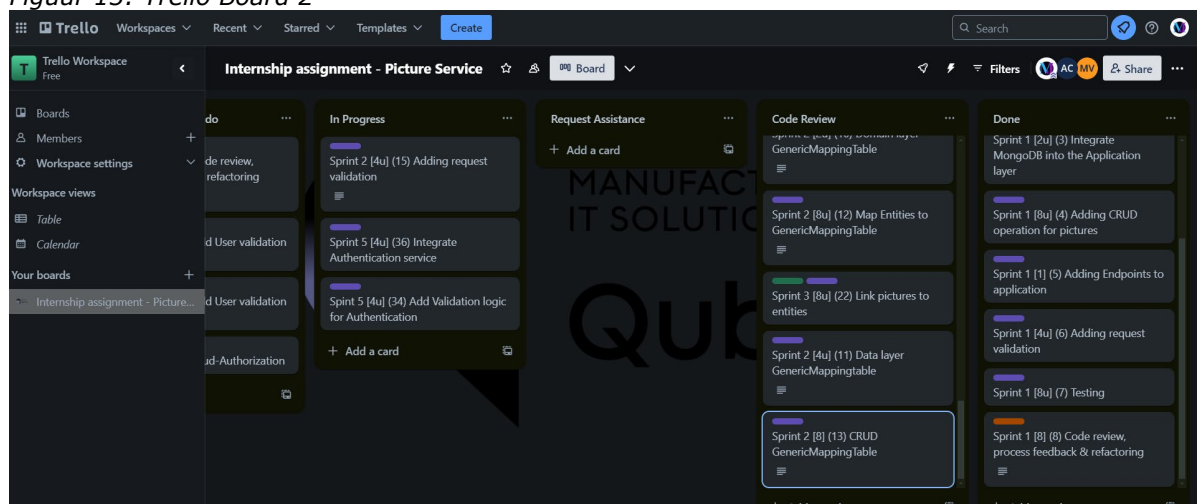
- **Code Review:** Taken die zijn voltooid maar nog moeten worden beoordeeld door mijn stagebegeleiders.
- **Done:** Taken die volledig zijn afgerond en goedgekeurd.

Hieronder is er een weergave van de planning in Trello te zien.

Figuur 12: Trello Board 1



Figuur 13: Trello Board 2



Deze indeling helpt om een duidelijk overzicht te houden van de voortgang van de stageopdracht en zorgt ervoor dat elke taak de juiste aandacht krijgt tijdens elke fase van ontwikkeling. Deze indeling helpt om een duidelijk overzicht te houden van de voortgang van de stageopdracht. Daarnaast fungeert het als een belangrijk instrument tijdens de wekelijkse planningsmeeting, waarin we evalueren of het project nog op schema zit en waarbij dit board helpt om het overzicht te behouden. Dit zorgt ervoor dat elke taak de juiste aandacht krijgt tijdens elke fase van ontwikkeling.

5 ONTWIKKELINGSFASE

De ontwikkelingsfase van dit project speelde een cruciale rol bij het omzetten van de geplande functies en ontwerpen naar werkende software. In deze fase werden de theoretische concepten uit de vorige fasen daadwerkelijk geïmplementeerd. Dit hoofdstuk beschrijft de processen, methoden en tools die tijdens de ontwikkeling zijn gebruikt, evenals de structuur en organisatie van het werk.

5.1 De Backend

De kern van deze applicatie is afbeeldingen toevoegen aan entiteiten. Hieronder kunt u een deel van implementatie in de Backend zien.

Figuur 14: Handler voor foto-opslag

```
0 references | Murrel Venio, 10 days ago | 1 author, 3 changes
public async Task<Picture> Handle(AddPictureToEntity request, CancellationToken cancellationToken)
{
    var tableExists = await _mappingTablesReader.Exist(request.TableName);

    if (!tableExists)
        throw new BusinessException("The specified table does not exist in the database.");

    var mappingExists = await _mappingTablesReader.MappingExists(request.TableName, request.KeyValuePairs);

    Guid mappingId;

    if (!mappingExists)
    {
        // If the mapping doesn't exist, create a new one
        mappingId = await _mappingTableCreator.Add(new GenericMappingTable()
        {
            TableName = request.TableName,
            KeyValuePairs = request.KeyValuePairs,
        });
    }
    else
    {
        // If the mapping exists, retrieve its ID
        mappingId = await _mappingTablesReader.ByTable(request.TableName, request.KeyValuePairs);
    }

    // Create and add the picture
    var picture = new Picture
    {
        Name = request.ImageName,
        ImageData = request.ImageData,
        MappingId = mappingId,
        RowVersions = 1
    };

    await _picturesRepository.InsertOneAsync(picture);

    return _picturesRepository.FindById(picture.Id);
}
```

Figuur 15 Controller voor foto-opslag

```

/// <summary>
/// Add picture to entity
/// </summary>
/// <param name="storePictureVM"></param>
/// <returns></returns>
[HttpPost("v1/pictures")]
[ProducesResponseType(StatusCodes.Status201Created, Type = typeof(PictureVM))]
[ProducesResponseType(StatusCodes.Status400BadRequest)]
[ProducesResponseType(StatusCodes.Status500InternalServerError)]
0 references | Murrel Venlo, 7 days ago | 1 author, 7 changes
public async Task<IActionResult> Add([FromForm] StorePictureVM storePictureVM)
{
    if (storePictureVM == null)
        return BadRequest();

    var imageData = _mapper.Map<byte[]>(storePictureVM.ImageData);
    var result = await _mediator.Send(new AddPictureToEntity(
        storePictureVM.Name,
        imageData,
        storePictureVM.TableName,
        storePictureVM.KeyValuePair));

    var resultVM = _mapper.Map<PictureVM>(result);
    return Ok(resultVM);
}

```

Zoals in Figuur 16 te zien is, zijn dit screenprints van de API-Calls. Deze end-points zijn belangrijk bij de implementatie in de Frontend. In deze interface zijn enkele belangrijke functies op te merken, zoals het toevoegen van een afbeelding aan een entiteit, het ophalen, bewerken en verwijderen van afbeeldingen.

Figuur 16: API-Calls Pictures

Picture Service API v1 OAS3
<https://localhost:7277/swagger/v1/swagger.json>

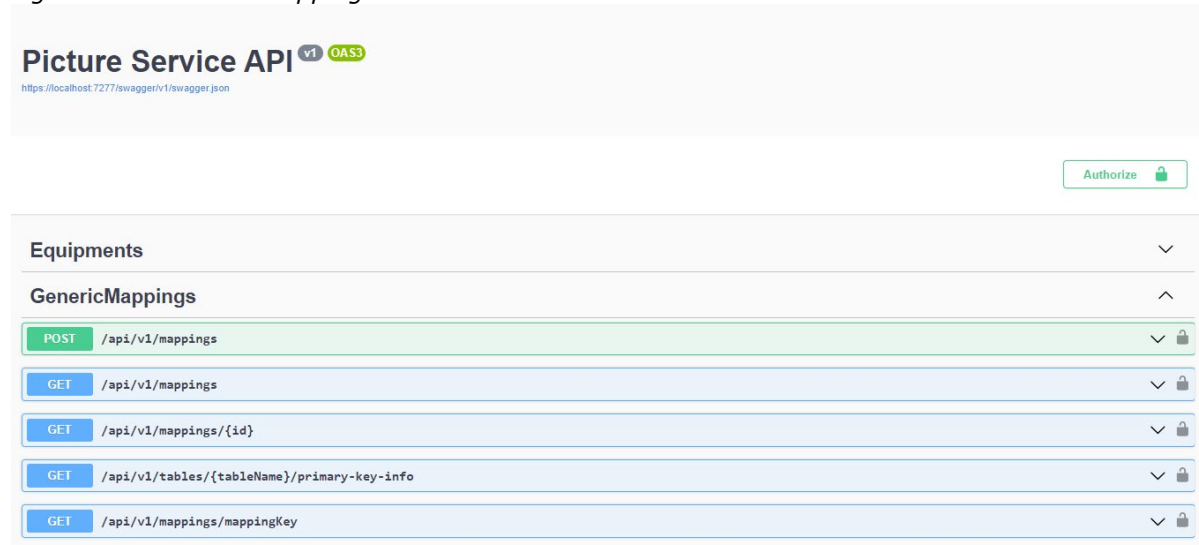
[Authorize](#)

Equipments	▼
GenericMappings	▼
Persons	▼
Pictures	▲
POST /api/v1/pictures	▼
GET /api/v1/pictures	▼
GET /api/v1/pictures/{id}	▼
PUT /api/v1/pictures/{id}	▼
DELETE /api/v1/pictures/{id}	▼
GET /api/v1/pictures/get-multiple	▼
GET /api/v1/pictures/value-pair	▼

De endpoints die hieronder worden weergegeven, zijn relevant voor de relatie tussen de afbeeldingencollectie en de bestaande Qubris Database. Wanneer een afbeelding wordt toegevoegd, wordt eerst gecontroleerd of er al een relatie bestaat op basis van

de naam van de entiteit en de sleutel(s) van die entiteit. Deze sleutels worden aan hun waarden toegewezen, waarbij een zogenoemd 'Key-Value-Pair' wordt gevormd.

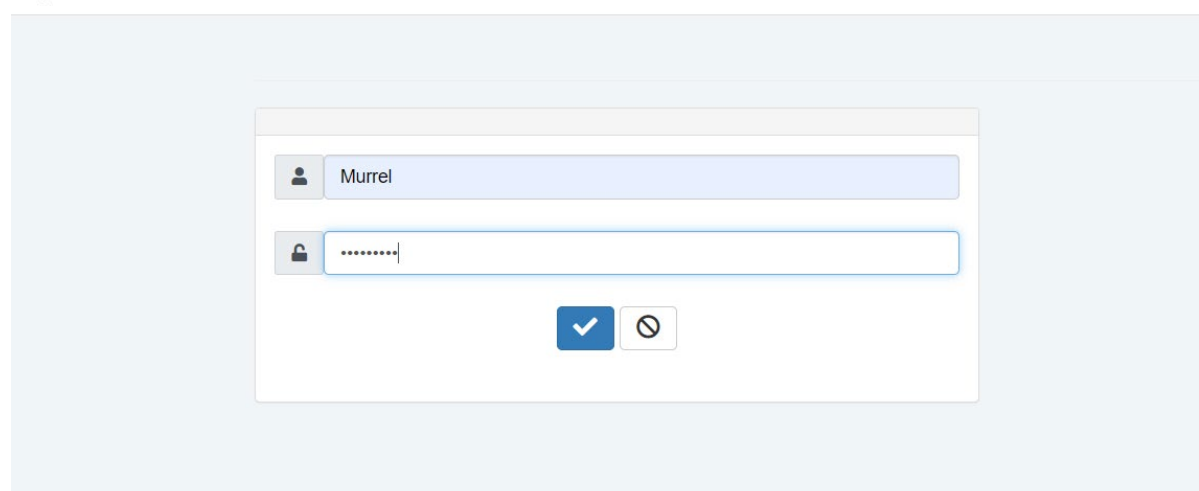
Figuur 17: API-Calls Mappings



5.2 De Frontend

5.2.1 Het inlogscherf

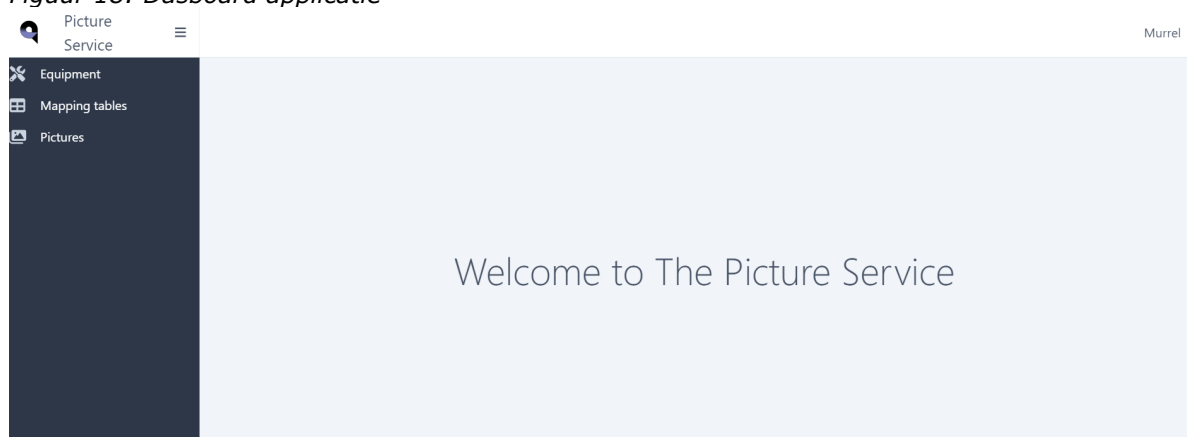
Op het inlogscherf kunt u uw gebruikersnaam en wachtwoord invoeren om toegang te krijgen tot de applicatie. Na een succesvolle login wordt u doorgestuurd naar het dashboard, waar u toegang heeft tot alle functies van de applicatie. Het inlogscherf zorgt ervoor dat alleen geautoriseerde gebruikers toegang hebben tot de applicatie.



5.2.2 Het Dashboard

Op de screenprint kunt u het dashboard van de applicatie zien. Als ingelogde gebruiker kunt u links een menu krijgen met de entiteiten.

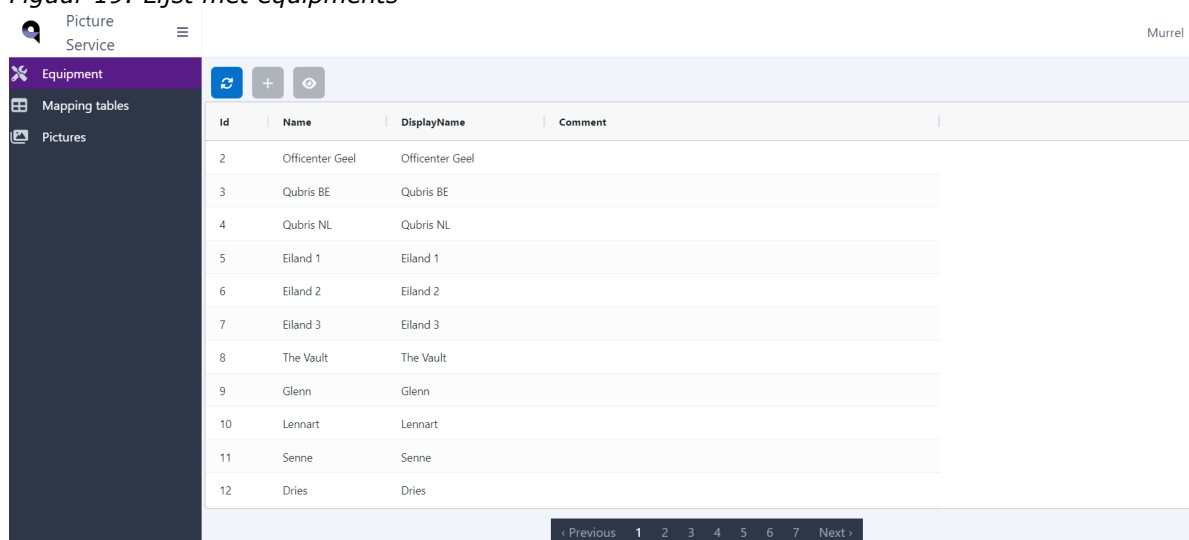
Figuur 18: Dashboard applicatie



5.2.3 Menu items

Als de gebruiker een entiteit aanklik, ziet hij een lijst van items. Selecteert hij vervolgens een item dan heeft hij de mogelijkheid om meteen een afbeelding toe te voegen of eerst een lijst met afbeeldingen per selectie te bekijken.

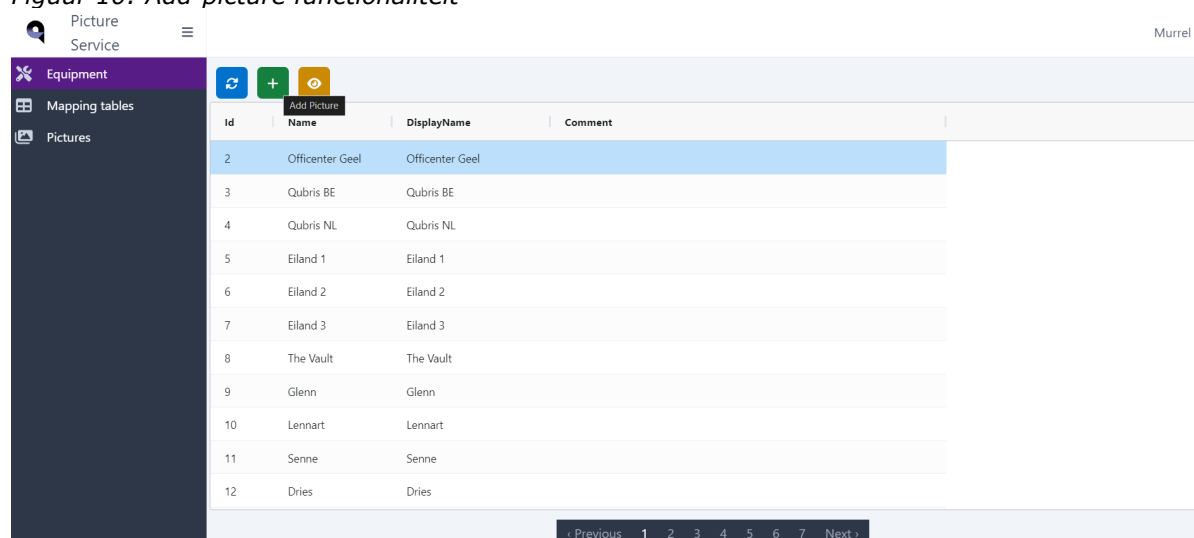
Figuur 19: Lijst met equipments



5.2.4 Afbeelding toevoegen

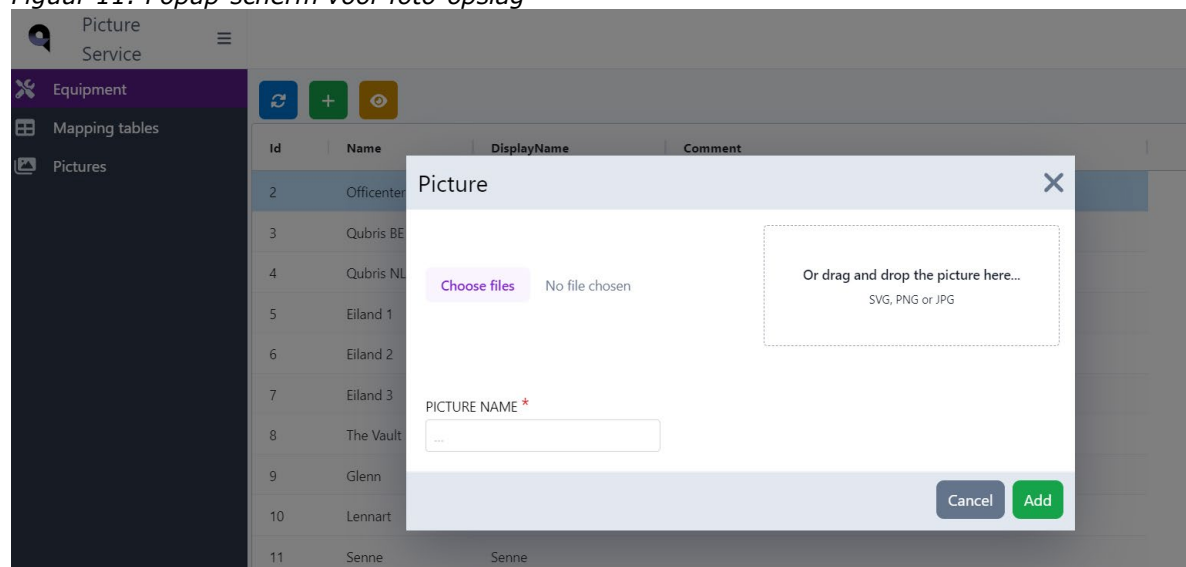
Hieronder kan de gebruiker het "Add Picture"-knopje aanklikken

Figuur 10: Add-picture functionaliteit

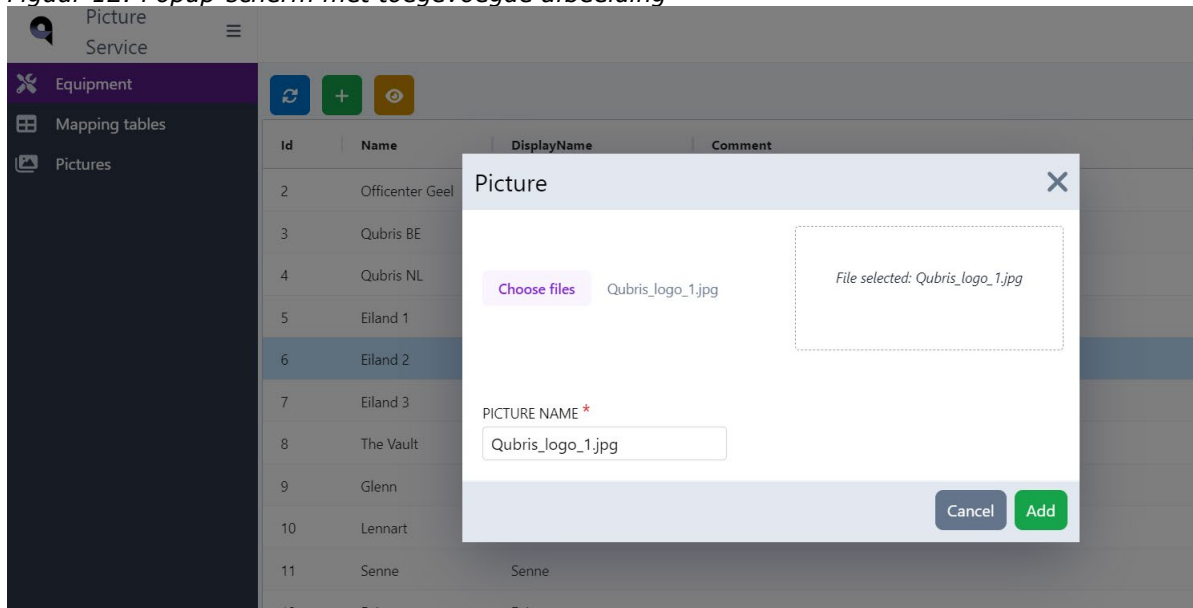


Na het "Add Picture"-knopje aangeklikt te hebben, krijgt de gebruiker een pop-upschermd om een afbeelding toe te voegen.

Figuur 11: Popup-schermd voor foto-opslag



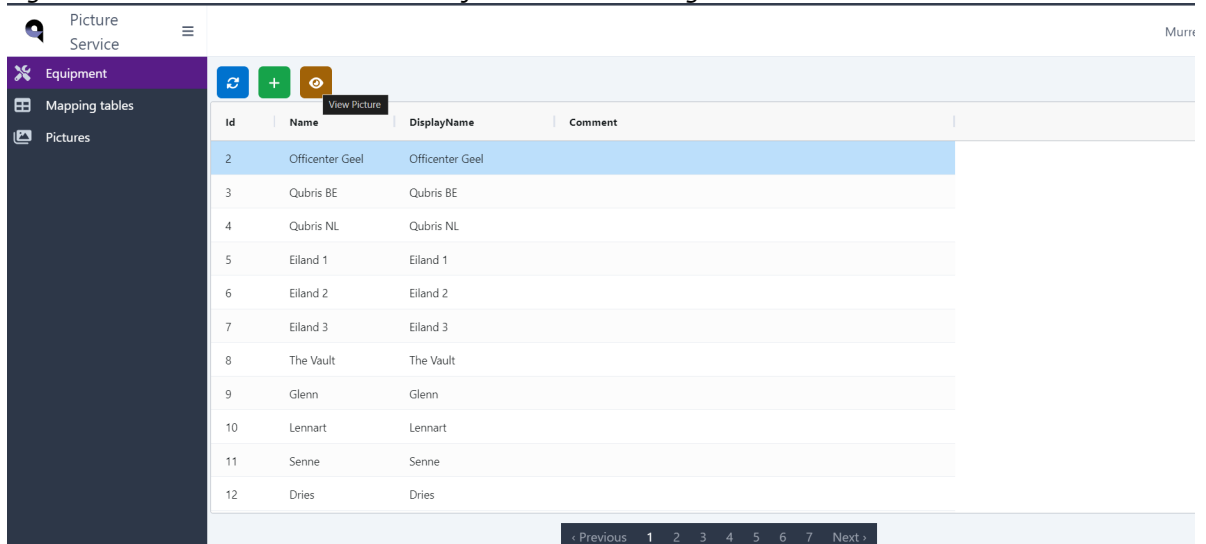
Figuur 12: Popup-scherm met toegevoegde afbeelding



5.2.5 Afbeelding bekijken

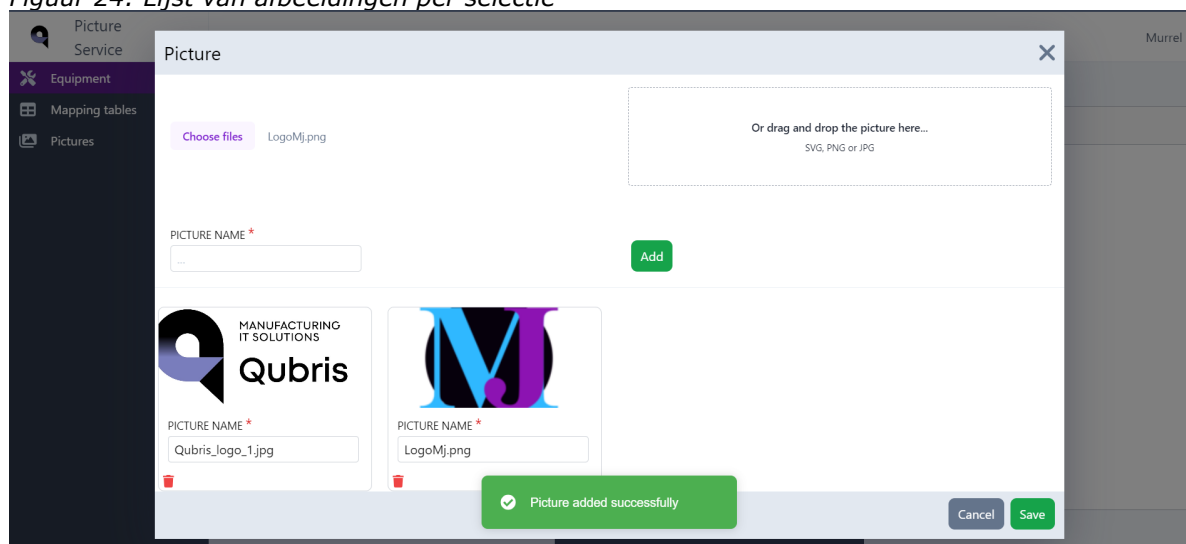
Na het "View Picture"-knopje aangeklikt te hebben, krijgt de gebruiker een pop-upscherm met de lijst van afbeeldingen per selectie.

Figuur 13: Fuctionaliteit voor het bekijken van afbeeldingen



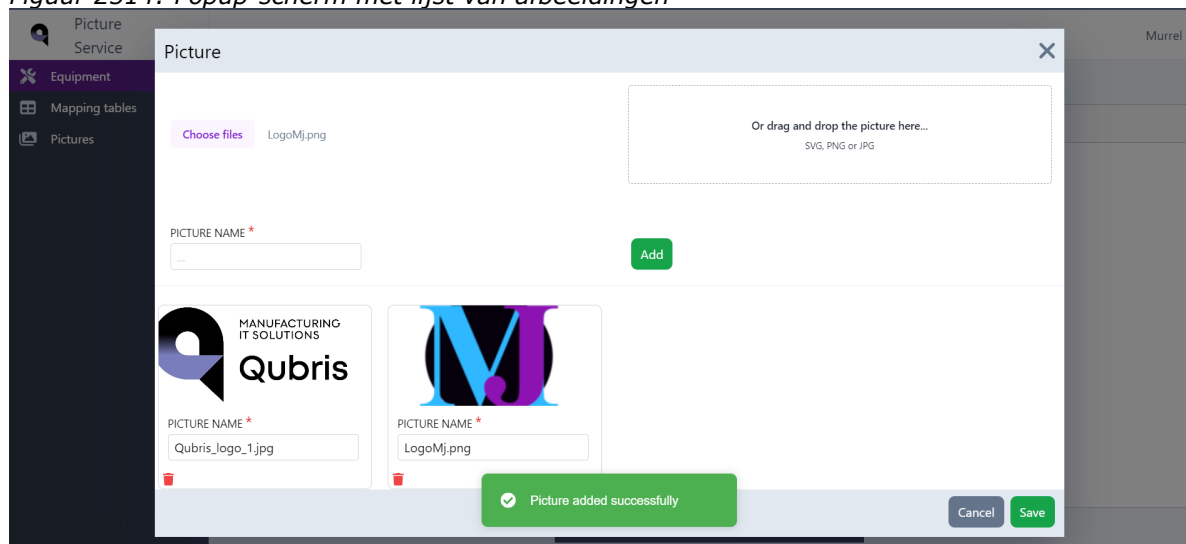
Op de screensprint hieronder is de lijst van afbeeldingen per selectie zien.

Figuur 24: Lijst van afbeeldingen per selectie



5.2.6 Afbeelding bewerken en verwijderen

Figuur 2514: Popup-scherm met lijst van afbeeldingen



Op de screenprint hierboven is te zien dat bij elke afbeelding de mogelijkheid bestaat om deze te verwijderen, aangeduid door een passend icoontje. Overigens is het ook mogelijk om de naam van de afbeelding aan te passen. Dit is mogelijk door de tekst in het tekstveld aan te passen en vervolgens op de save knop te drukken.

6 CONCLUSIE

Het ontwikkelingsproces van de innovatieve applicatie voor Qubris heeft zich gericht op het leveren van een product dat voldoet aan de behoeften van gebruikers door een naadloze ervaring te bieden voor het opslaan en beheren van foto's. Gedurende dit project zijn belangrijke aandachtspunten zoals doelstellingen, verwachtingen, technologieën en hulpmiddelen grondig geanalyseerd en toegepast om een hoogwaardig eindproduct te realiseren.

Het streven naar een eindproduct dat niet alleen voldoet aan de gestelde vereisten, maar ook uitblinkt in gebruiksvriendelijkheid, efficiëntie en betrouwbaarheid is centraal gesteld.

Dit heeft geleid tot de keuze voor een stack waarbij .NET wordt gebruikt voor de backend, MongoDB en Microsoft SQL als databases, en Vue.js voor de frontend. Deze technologieën zijn zorgvuldig geselecteerd vanwege hun compatibiliteit, betrouwbaarheid en prestaties.

Het UI-design, behandeld in het hoofdstuk UI Design, heeft zich gericht op het optimaliseren van de functionaliteiten van de app en het verbeteren van de gebruikerservaring.

Door middel van functionele eisen en gebruikscases zijn verschillende ontwerpbeslissingen genomen om een intuïtieve, responsieve en aantrekkelijke gebruikersinterface te creëren.

In het bijzonder zijn functionaliteiten zoals het opslaan, raadplegen, aanpassen en verwijderen van foto's geïmplementeerd met oog op gebruiksgemak en efficiëntie. Het UI-design heeft een cruciale rol gespeeld bij het tot leven brengen van deze functionaliteiten, waardoor gebruikers een soepele en plezierige ervaring hebben bij het gebruik van de applicatie.

Het ontwikkelingsproces van de applicatie heeft voor Qubris geleid tot een hoogwaardig eindproduct dat voldoet aan de gestelde verwachtingen en doelstellingen.

Door middel van zorgvuldige planning, analyse en implementatie van technologieën en UI-designprincipes, ben ik erin geslaagd om een applicatie te ontwikkelen die niet alleen functioneel en efficiënt is, maar ook aantrekkelijk en gebruiksvriendelijk voor de eindgebruikers.

7 BRONNEN

Best way to store image(s) in web application development: Best Practices and Considerations

<https://medium.com/@hassaanistic/best-way-to-store-image-in-any-web-application-development-best-practices-and-considerations-4fbacdf066d2>

Storing Images in a Database: Methods and Considerations

<https://www.beekeeperstudio.io/blog/how-to-store-images-in-a-database#:~:text=One%20method%20is%20to%20store,it%20directly%20in%20the%20database.>

Create New Cluster

<https://cloud.mongodb.com/v2/65315e6edef11f74af6c0486#/clusters/edit>

Apache CouchDB

<https://www.capterra.com/p/210370/Apache-CouchDB/>

Amazon S3 pricing

<https://aws.amazon.com/s3/pricing/>

Content Delivery Network pricing

https://azure.microsoft.com/en-us/pricing/details/cdn/?ef_id=k_CjwKCAiAloavBhBOEiwAbtAJ06zFi3M7LqT5XkY0mgtQVpC0J-FZOJO5vRVT0FuVUreRLYu1SoFf4RoCRygQAvD_BwE_k_OCID=AIDcmmbnk3rt9z_SEM_k_CjwKCAiAloavBhBOEiwAbtAJ06zFi3M7LqT5XkY0mgtQVpC0J-FZOJO5vRVT0FuVUreRLYu1SoFf4RoCRygQAvD_BwE_k_gad_source=1&gclid=CjwKCAiAloavBhBOEiwAbtAJ06zFi3M7LqT5XkY0mgtQVpC0J-FZOJO5vRVT0FuVUreRLYu1SoFf4RoCRygQAvD_BwE