CoGhent - DAMS

Last updated: 2022–11–25

Contents

DAMS **2**

Termen . . . 2

Services & onderdelen . . . 3

Relatie tot data.collectie.gent en de CoGent-box . . . 4

Functionaliteiten van het DAMS (MVP) . . . 4

Toevoegen en importeren van asset . . . 4

Beheren van assets . . . 5

Beheer mediabestanden . . . 5

Transcodes . . . 5

Bekijken van assets . . . 6

downloaden . . . 6

Zoeken naar assets & mediabestanden . . . 6

Entiteiten en thesauri . . . 6

Job system . . . 7

Hiërarchieën opbouwen . . . 7

Publicatie flow 6de collectie . . . 7

Gebruikers, rollen & rechten, licenties . . . 8

Gebruikers . . . 8

Rollen en rechten . . . 8

Licentiebeheer van mediabestanden . . . 8

Toekomstige functionaliteiten . . . 8

Technische gegevens . . . 9

Koppeling met LDES 9
Serverbeheer 9
Storage & backup 10
Adlib2eventstream 10
Installatie 11

DAMS

Het CoGhent DAMS - Digital Asset Management System - heeft als doel om mediabestanden te ontsluiten en beheren. Dit gaat zowel over digitale representaties van objecten uit de verschillende instellingen binnen de Stad Gent, maar ook over digital born materiaal, alsook over het materiaal dat burgers aanleveren als onderdeel van de Collectie van de Gentenaar. Het DAMS is echter niet beperkt tot deze (project gebonden) use cases. Zo zou een dienst communicatie het systeem kunnen gebruiken om het digitale materiaal behorende bij een persmededeling beschikbaar te stellen.

Het DAMS heeft volgende doelen:

- De medewerkers van instellingen en diensten ondersteunen in het beheren van het digitale materiaal. Dit door te zorgen dat mediabestanden op een zo eenvoudig en efficiënt mogelijke manier kunnen geïmporteerd worden. Anderzijds ook door de bijbehorende metadata op een zo automatisch en volledig mogelijke manier op te halen uit de bronsystemen en de koppeling te leggen tussen metadata en beeldmateriaal.
- Medewerkers en het brede publiek helpen om zo eenvoudig mogelijk beeldmateriaal snel terug te vinden over alle instellingen heen. Dankzij geavanceerde zoekfilters kunnen gebruikers gericht zoeken en in de massa aan materiaal toch de specifieke assets terug vinden die ze zoeken. Bovendien, doordat er over de grenzen van de instellingen heen gezocht kan worden en linken gelegd worden (door inzet op Linked Data), kan dit tot nieuwe inzichten leiden.
- Externe systemen voeden die gebruik willen maken van de digitale weergaves.

Termen

DAMS In dit document wordt met DAMS zowel de beheerinterface bedoeld die gebruikers in staat stelt om assets te beheren, alsook het hele achterliggende systeem dat zowel het dams.collectie.gent (hierna: 'DAMS'), data.collectie.gent (hierna 'het webplatform'), als de CoGent-box van data voorziet. DAMS staat voor Digital Asset Management System.

Mediabestand een bestand dat van waarde is om in het DAMS bij te houden. Dit kan in principe eender welk type bestand zijn. Vaak zal dit over een beeld-, video- of audiobestand gaan, of een document in PDF-formaat.

Metadata informatie die het mediabestand beschrijft. Voorbeelden zijn titel, beschrijving, datum waarop het werk gemaakt is, materiaal, . . .

Assets een asset is het geheel van één of meerdere mediabestanden, de beschrijvende metadata, technische metadata van de mediabestanden en afgeleiden.

Entiteiten de term entiteit wordt binnen DAMS gebruikt om alles aan te duiden wat metadata bevat en waarnaar kan verwezen worden. Iedere entiteit heeft een type. Het belangrijkste type entiteit binnen

DAMS is het asset. Voorbeelden van andere entiteiten zijn auteurs, thesauri-termen, organisaties, . . .

Transcode / afgeleide een transcode is een automatisch aangemaakte variant van een mediabestand. Voorbeelden hiervan zijn JPG-bestanden voor TIFFs en thumbnails voor video-bestanden. Events wanneer bepaalde zaken gebeuren binnen DAMS, bijvoorbeeld het uploaden van een nieuw mediabestand, dan zal DAMS hiervoor een event uitsturen. De verschillende micro-services kunnen zich abonneren op deze events en hiermee reageren op bepaalde acties. Het automatisch aanmaken van afgeleiden werkt bijvoorbeeld via events: wanneer een nieuw mediabestand wordt toegevoegd in DAMS dan wordt de transcode-service hiervan op de hoogte gebracht via een event, zodat deze de correcte afgeleiden kan maken.

Event-stream Een event stream is een verzameling van objecten met versiebeheer (in deze context is versie zoals een eventstream event) en kan op elk moment worden bijgewerkt in hun eigen tempo (trage & snelle data). Op deze manier kunnen consumenten gemakkelijk de laatste wijzigingen ontdekken en gebruiken.

Services & onderdelen

DAMS bestaat uit volgende services:

Collection API Dit is de kern van het DAMS systeem. De Collection API bevat alle metadata van alle assets, entiteiten en taxonomieën.

Storage API De Storage API zorgt voor het wegschrijven en beschikbaar stellen van de individuele mediabestanden. Hiervoor wordt binnen CoGhent de S3 server van District09 gebruikt.

Search API De Search API indexeert alle assets en entiteiten en werkt de index bij wanneer er aanpassingen gebeuren.

Transcode services Via 1 of meerdere transcode services worden automatisch afgeleiden (transcodes) gemaakt. Een voorbeeld hiervan zijn TIFFs die automatisch naar JPGs worden omgezet. Het originele bestand (TIFF) blijft beschikbaar, maar voor een webweergave kan dan een meer geschikt formaat (JPG) opgevraagd worden. De transcode services kunnen via de micro-architectuur uitgebreid en toegevoegd worden. Hierbij kunnen extra transcode services eventueel door andere partijen geïmplementeerd worden en/of op andere locaties (data centers) draaien. De transcode service maakt gebruik van de CloudEvents om op de hoogte gesteld te worden van nieuwe mediabestanden die een afgeleide nodig hebben, en communiceert via de standaard APIs met het systeem.

RabbitMQ is de centrale event broker waar alle componenten met bepaalde interesses zich kunnen abonneren.

LDES Client leest de event streams van de verschillende Gentse cultureel erfgoedinstellingen uit en voedt hiermee de Collection API.

LDES Server stelt de data van de Collectie van de Gentenaar ('de 6de collectie') beschikbaar als een event stream, zodat andere systemen hierop kunnen subscriben en met deze data aan de slag kunnen. IIIF server stelt de mediabestanden, voor zover in het juiste formaat, beschikbaar via de IIIF Image en IIIF Presentation standaard. Op deze manier kan DAMS zelf, alsook derde partijen, afbeeldingen opvragen en hier on-the-fly manipulaties op uitvoeren.

ArangoDB is de centrale databank voor de Collection API. Arango is een Graph database. ElasticSearch wordt gebruikt door de Search API voor de indexering van metadata. Voorziet ook in het doorzoekbaar maken van de metadata, rekening houdende met typfouten e.d.

S3 voorziet in de opslag van de mediabestanden.

Beheerinterface De grafische interface die voor eindgebruikers beschikbaar wordt gesteld. In de beheerinterface kunnen gebruikers beelden toevoegen, bewerken, verwijderen, alsook opzoeken.

GraphQL Een tussenlaag voor de beheerinterface en de verschillende services. De GraphQL-laag zorgt ervoor dat de beheerinterface niet zelf naar alle verschillende services moet verbinden, maar enkel met

de GraphQL hoeft te communiceren. De GraphQL-laag voorziet tevens in configuratie van de output. Zo zal de beheerinterface dynamisch bepaalde metadata velden al-dan-niet laten zien, afhankelijk van welke metadatavelden de GraphQL-laag beschikbaar stelt voor de beheerinterface.

Relatie tot data.collectie.gent en de CoGent-box

Naast DAMS zijn er binnen CoGhent nog volgende specifieke onderdelen:

CoGent-box Deze component bestaat uitverschillende onderdelen die gebruikt worden in de box: 180-Wall, Inkom, Touchtafel. Deze worden gevoed met data vanuit het DAMS, die opgehaald worden via de CoGhent GraphQL.

Webplatform Het webplatform dat zich richt op het grote publiek. Dit wordt ook gevoed vanuit het DAMS via de CoGhent GraphQL.

CoGhent GraphQL De GraphQL-laag die voor de CoGhent specifieke componenten wordt gebruikt. Deze is verschillend van de DAMS GraphQL omdat er o.a. een onderscheid wordt gemaakt in de metadata die op het webplatform (vs DAMS) wordt getoond.

Elk gepubliceerd asset op het webplatform beschikt daarnaast over een URI conform aan de Vlaamse URI standaard en verwijst terug naar de resource URI (pagina uit de event-stream waar het object beschreven staat).

Een belangrijke opmerking hierbij is dat zowel het webplatform als de CoGent-box data gebruiken vanuit het DAMS. Naast de DAMS beheerinterface zijn deze twee interfaces dus vooral alternatieve manieren om de assets in het DAMS beschikbaar te maken en gebruikers een bepaalde ervaring te geven.

Functionaliteiten van het DAMS (MVP)

Toevoegen en importeren van asset

Assets kunnen op een aantal manieren toegevoegd worden:

Upload Voor één of enkele assets kan de gebruiker de mediabestanden via zijn browser uploaden. Bij uploads, alsook imports, worden bestanden steeds op dubbels gecontroleerd. Dit gebeurt aan de hand van de checksum (md5). Wanneer het bestand reeds in het systeem zit, zal dit geen tweede keer opgeslagen worden.

Import Wanneer meer assets geupload moeten worden die bovendien meteen gekoppeld worden aan de juiste objecten uit de event-stream kan de import functionaliteit gebruikt worden. Hierbij worden de bestanden klaargezet op een netwerk-share, in combinatie met een CSV-bestand. Dit CSV-bestand bevat informatie over hoe de koppeling automatisch gelegd kan worden. Hiervoor moet minstens de bestandsnaam en objectnummer vermeld worden. Eventueel aangevuld met informatie over rechten, rechtenhouder, fotograaf, bron en publicatiestatus van het bestand.

Toekomst: uitgebreidere import-functies: Aanleveren van beschrijvende metadata via csv. Ondersteuning voor begeleidende formaten zoals METS, waarbij de hiërarchie wordt overgenomen. Ondersteuning van alternatieve formaten, zoals ALTO bij afbeeldingen.

Beheren van assets

Nadat assets zijn aangemaakt door een import of upload kunnen deze verder beheerd worden. Hierbij kunnen deze verrijkt worden met metadata. Er wordt een onderscheid gemaakt tussen metadatavelden die werden opgevuld met gegevens vanuit een extern systeem, bijvoorbeeld data uitgelezen uit de event-streams, en vrij invulbare metadatavelden. In het kader van dataduurzaamheid en synchronisatie met andere systemen, kunnen velden die vanuit een externe bron werden aangevuld niet binnen DAMS aangepast worden. Deze dienen in het bronsysteem aangepast te worden en zullen bij een volgende synchronisatie hun nieuwe waardes in DAMS krijgen.)
Velden die niet vanuit een externe bron werden opgevuld zijn vrij invulbaar (momenteel beperkt tot 'de 6de collectie'). Hierbij kunnen velden voorgedefinieerd worden met veldnaam en type. DAMS helpt dan de gebruiker bij het invullen met validatie en mogelijke waardes. Zo zal bijvoorbeeld een numeriek veld enkel cijfers toestaan en een veld voor een auteur een auto-complete tonen met alle gekende auteurs.

Voor ieder asset is het mogelijk om aan te geven welk mediabestand moet gebruikt worden als hoofd-mediabestand en welk voor de thumbnail. Zo kan bijvoorbeeld een asset samengesteld worden voor een radio-interview met:

- De geluidsopname (hoofdmediabestand)
- Foto van de persoon die werd geïnterviewd (thumbnail)
- Transcriptie

Beheer mediabestanden

Bij het bewerken van assets is het ook mogelijk om de mediabestanden te beheren (basis CRUD functionaliteiten):

- Mediabestanden kunnen toegevoegd worden. Dit kan zowel door een nieuwe bestand op te laden, alsook door een bestaand mediabestand vanuit het systeem te selecteren. Hierbij valt op te merken dat een zelfde mediabestand via deze laatste manier (selectie) aan meerdere assets gekoppeld kan zijn. Dit kan nuttig zijn wanneer van eenzelfde werk meerdere beschrijvende records beschikbaar zijn.
- Mediabestanden kunnen verwijderd worden.
- Mediabestanden kunnen ook vervangen worden. Dit is handig wanneer er bijvoorbeeld een betere scan gemaakt werd van een pagina.
- Bij de mediabestanden kan ook metadata aangepast worden. Dit kan bijvoorbeeld gaan over de fotograaf van de foto en de rechten die op het mediabestand rusten.

Transcodes

Niet alle bestandsformaten zijn in alle situaties even geschikt om mee te werken. Zo zijn bestandsformaten voor een hoogwaardig drukwerk bijvoorbeeld minder geschikt voor weergave op het web. Om dit probleem op te lossen maakt het DAMS gebruik van 'transcodes', ook 'afgeleiden' genoemd. Deze transcodes kunnen per omgeving via de micro-services geconfigureerd worden. Een concreet voorbeeld hiervan zijn mediabestanden in het TIFF-formaat. Deze bestanden zijn vaak heel groot waardoor ze moeilijk bruikbaar zijn voor weergaven in een webbrowser. Daarom maakt DAMS voor iedere TIFF automatisch een JPG-afgeleide met dezelfde resolutie en minimale compressie. Deze JPG-afgeleide is daarom beter geschikt voor een web-weergave en dient ook als basis voor de IIIF-weergave.

Bekijken van assets

Binnen DAMS kunnen assets bekeken worden. Hierbij wordt standaard het hoofdmediabestand als eerste mediabestand weergegeven. Binnen DAMS kunnen verschillende bestandstypes bekeken worden. Het kan hierbij gaan om afbeeldingen (via IIIF), video's, audiofragmenten, pdf's... Hierbij zal steeds het formaat gekozen worden dat het meest geschikt is en voldoet aan de rechten van een gebruiker (zie licentiebeheer).

downloaden

Afhankelijk van de rechten van de gebruiker en de rechtenstatus van het mediabestand kan een gebruiker een beeld downloaden. Hierbij wordt ook rekening gehouden met de rechten en zal automatisch het bestand in de hoogst mogelijke kwaliteit ter download worden aangeboden. Indien de gebruiker (collectie)medewerker is van de instelling die het asset beheert, zal deze gebruiker het beeld in het originele formaat kunnen downloaden. In andere gevallen hangt het van de rechten af van mediabestand en gebruiker en zal de gebruiker enkel een afgeleide (in lagere kwaliteit) kunnen downloaden, of zelfs helemaal geen mediabestand te zien krijgen.

Zoeken naar assets & mediabestanden

Een belangrijk aspect van het DAMS is het helpen van gebruikers om heel gericht assets terug te vinden. Zeker wanneer er tien- of honderdduizenden assets geïmporteerd zijn in het DAMS is dit erg belangrijk. Het DAMS helpt de gebruiker hierbij door allerlei filters ter beschikking te stellen. Deze filters kunnen geconfigureerd worden voor alle mogelijke metadatavelden. Net als bij het bewerken van metadata kent het DAMS het type van deze velden en zal de gebruiker ook hierbij helpen met auto-complete waardes en validaties waar nodig. Bij iedere filter kan de gebruiker ook aangeven hoe de filter juist moet zoeken. Dit hangt af van het type filter. Zo kan voor een tekstfilter bijvoorbeeld aangegeven worden dat het stuk tekst een exacte overeenkomst moet hebben, één van de woorden moet bevatten, of zelfs juist geen enkel van de woorden mag bevatten.
Na een zoekactie kan de gebruiker verder filters toevoegen of verwijderen om de zoekresultaten te verfijnen of juist verbreden. Via het permissie systeem is het ook mogelijk om bepaalde filters enkel voor bepaalde groepen van gebruikers beschikbaar te stellen.

Entiteiten en thesauri

DAMS is opgebouwd rond entiteiten. Een entiteit is ieder object dat metadata kan bevatten. Binnen DAMS kunnen een onbeperkt aantal types gedefinieerd worden. Voorbeelden zijn 'persoon', 'instelling', 'materiaal'. Entiteiten kunnen onderlinge relaties naar elkaar hebben. Iedere entiteit kan via metadata beschreven worden. Het Asset is een speciaal type van entiteit welke centraal staat binnen DAMS. Dankzij de verschillende entiteiten kunnen zaken zoals personen centraal beschreven en beheerd worden en er volgens het Linked Data principe relaties gelegd worden. Dit zorgt er voor dat gegevens van een persoon maar in één enkele entiteit moeten aangepast worden om de metadata, vb geboortedatum, toe te voegen. Dit geeft ook overzichtelijke lijsten, bijvoorbeeld voor het beheer van thesauri.

Job system

Om gebruikers die beherende taken uitvoeren meer inzicht te geven in allerlei zaken bevat DAMS een job systeem. Hier krijgen beheerders een duidelijk overzicht van de voortgang en uitkomst van langdurige taken. Een voorbeeld hiervan is een import. Afhankelijk van hoeveel bestanden dit zijn kan dit enkele minuten tot uren duren. DAMS start voor iedere import een job met een sub job per bestand. Op deze manier kan de gebruiker de voortgang van de gehele import job zien, alsook het resultaat van ieder individueel bestand. Ook wanneer er andere zaken achter de schermen gebeuren ten gevolge van een actie in DAMS wordt hier een job voor gemaakt, zodat de gebruiker hier een duidelijk inzicht in krijgt en kan zien wat het resultaat is. Een voorbeeld hiervan is het automatisch aanmaken van een transcode. Dankzij het job-systeem wordt dit inzichtelijk voor de beheerder.

Hiërarchieën opbouwen

In verschillende gevallen zal de structuur van een asset complexer blijken dan een mediabestand met metadata. Daarom zijn in het DAMS een aantal manieren voorzien om meer complexe structuren op te bouwen.
Een eerste mogelijkheid is om meerdere mediabestanden toe te voegen aan een asset. Dat kan bijvoorbeeld nuttig zijn wanneer een boek wordt gedigitaliseerd waarbij iedere pagina een aparte scan is. In dit geval kan iedere pagina als apart mediabestand toegevoegd worden. Hierbij kan ieder mediabestand zijn eigen metadata hebben, dat de specifieke pagina beschrijft of rechten weergeeft. Ieder mediabestand zal bovendien zijn eigen afgeleiden krijgen. Binnen het asset kan er een mediabestand worden aangeduid dat als 'primair' mediabestand zal dienen. Dit is het mediabestand dat als eerste getoond zal worden bij het openen van het asset. Verder kan een asset worden aangeduid dat gebruikt zal worden voor de thumbnail. Standaard is dat hetzelfde mediabestand als het primaire mediabestand, maar in sommige gevallen kan het wenselijk zijn hiervan af te wijken.
Een andere mogelijkheid is om alles aan elkaar te linken en op deze manier een volledig hiërarchie op te bouwen. Dit kan nuttig zijn wanneer assets een onderling verwantschap hebben, maar ook op zichzelf bestaande assets zijn. Een voorbeeld hiervan zou een reeks tekeningen zijn die samen een reeks vormen. Een ander voorbeeld is een tijdschrift dat uit verschillende jaargangen bestaat. Daarbij kan een overkoepelend asset gemaakt worden die het volledige magazine representeert, een asset per jaargang, en daaronder een asset per editie. Deze assets kunnen op hun beurt dan verschillende mediabestanden hebben met de scans van de verschillende pagina's.

Publicatie flow 6de collectie

De publicatie van assets in de '6de collectie' verloopt als volgt:

1. Een eindgebruiker kan via het webplatform inloggen en één of meerdere assets opladen. Hierbij kan extra metadata ingegeven worden door de eindgebruiker.
2. Via het DAMS kunnen alle nog te valideren werken gevonden worden. Dit kan gedaan worden met behulp van de filters in het zoekscherm.
3. Een asset dat gevalideerd dient te worden kan bewerkt worden. Hierbij kan de zichtbaarheid en licentie aangepast worden door de validerende instelling.
4. Verder is het mogelijk om extra metadata toe te voegen of aan te passen.

Gebruikers, rollen & rechten, licenties

Gebruikers

Gebruikers van DAMS worden binnen Keycloak beheerd. Dit is een centraal systeem voor gebruikersbeheer. Binnen Keycloak kunnen gebruikers rollen krijgen. Eén van de voordelen van Keycloak is dat dit eenvoudig gekoppeld kan worden met andere 'entity providers' zoals Mijn Gent en het login systeem voor medewerkers van Stad Gent. Zo kunnen mensen hun reeds bestaande account gebruiken binnen DAMS en is er geen noodzaak om voor DAMS een specifieke account te maken. Het gebruikersbeheer wordt hierdoor ook eenvoudiger voor de applicatiebeheerders van DAMS. Immers wanneer een account van een medewerker centraal wordt uitgeschakeld is deze ook meteen binnen DAMS uitgeschakeld.

Rollen en rechten

Binnen DAMS krijgen alle gebruikers één of meerdere rollen toegekend. Via deze rollen zullen de gebruikers een set aan permissies krijgen, die hen toelaten om al dan niet bepaalde zaken te bekijken en aan te passen. Via dit rechtensysteem kunnen gewone gebruikers van het systeem DAMS alle publieke assets bekijken. Medewerkers van de instellingen daarentegen kunnen ook niet-publieke assets bekijken van de musea en de assets van hun eigen instelling bewerken (bijvoorbeeld werkbeelden). Dit systeem is volledig aan te passen via technische configuratie zodat rechten van de verschillende gebruikersgroepen op elk moment herbekeken kunnen worden.

Licentiebeheer van mediabestanden

Het licentiebeheer van mediabestanden is nauw verwant aan de rechten van een gebruiker. Zo is het immers mogelijk dat bepaalde mediabestanden niet bekeken mogen worden door gewone gebruikers, maar wel door museummedewerkers.
Rechten worden bepaald door een combinatie van metadata op het asset en op het mediabestand:

- Op het niveau van een asset kan bepaald worden of het asset publiek is, of in zijn geheel niet zichtbaar voor gewone gebruikers van het DAMS (en web portaal).
- Per mediabestand kan vervolgens ook aangegeven worden of er nog copyright op het bestand rust. Bovendien kan ook de rechthebbende aangegeven worden. Voor deze indicatie wordt gebruik gemaakt van de internationaal erkende Creative Commons licenties aangevuld met Rights Statements.

De combinatie van deze twee parameters, in combinatie met de rol van de gebruiker zal bepalen of de gebruiker het asset al-dan-niet te zien krijgt en welke mediabestanden zichtbaar en downloadbaar zijn.

Op het webplatform worden enkel assets getoond die minstens één publiek zichtbaar mediabestand hebben. In het detailscherm van een asset worden enkel de publiek beschikbare mediabestanden getoond.

Toekomstige functionaliteiten

Het DAMS wordt nog steeds verder ontwikkeld en zal in de toekomst verschillende nieuwe functionaliteiten krijgen. Onderstaande is een lijst van functies die op de planning staan. Deze lijst is echter verre van limitatief en zal in de toekomst uitgebreid worden.

Bulkbewerkingen en acties Momenteel is het mogelijk om asset per asset te bewerken. Vaak zal er

echter nood zijn aan het kunnen bewerken van enkele, of vele, assets in één bewerking. Een voorbeeld hiervan kan het aanpassen zijn van de rechten op meerdere assets om deze publiek vrij te geven. Ook het downloaden van meerdere assets in één beweging zal een nuttige toevoeging zijn.

Opslaan van zoekacties Via de geavanceerde zoekfilters kan efficiënt gezocht worden naar assets en andere entiteiten. Voor zoekacties die vaak herhaald worden zal het mogelijk zijn om deze op te slaan. Op deze manier kan deze zoekactie eenvoudig herhaald worden op een later tijdstip. Daarbij kan een zoekactie opgeslagen worden voor privé-gebruik, maar ook, indien de gebruiker de juiste rechten heeft, beschikbaar gesteld worden voor alle gebruikers.

Uitbreiding rechten en rollen Momenteel worden de rechten voor een gebruiker bepaald door een aantal rollen, waaronder een rol per instelling. In de toekomst zullen bepaalde entiteiten kunnen aangeduid worden als een 'rechtengevende entiteit'. Dit kan dan gaan over de instelling, maar bijvoorbeeld ook over een collectie-entiteit. Via de rechtengevende entiteiten kunnen gebruikers extra rollen krijgen die enkel van toepassing zijn op assets die verwant zijn aan deze entiteiten. In de praktijk kunnen mensen zo bijvoorbeeld rechten krijgen om terug alle assets van een instelling te bewerken. Dit laat echter ook toe om bijvoorbeeld een jobstudent aan te nemen voor het metadateren van een specifieke deelcollectie. Dankzij dit systeem kan de student dan enkel bewerkrechten krijgen voor deze deelcollectie, waar in het huidige systeem nog bewerkrechten voor de volledige instelling gegeven moeten worden.

Technische gegevens

Koppeling met LDES

Ieder uur importeert het DAMS automatisch de LDES-stromen van de verschillende instellingen. Hierbij wordt er gecontroleerd adhv de objectnummer of er reeds een asset bestaat voor het record. Indien er reeds een asset bestaat, wordt dit bijgewerkt met de recentste gegevens. Indien er nog geen asset bestond wordt er een nieuw asset gemaakt.

Serverbeheer

De nomad servers draaien op een VMware ESX die beheerd wordt door district09. Inuits heeft een account op vSphere waarvan de credentials te vinden zijn in de inuits vault.

Er zijn 3 ESX-en voorzien met de volgende machines:

- srvesxcog01.gentgrp.gent.be
  - es01.coghentdev
  - es01.coghentprod
  - nomad01.coghentprod
- srvesxcog02.gentgrp.gent.be
  - es02.coghentdev
  - es02.coghentprod
  - nomad02.coghentprod
- srvesxcog03.gentgrp.gent.be
  - es03.coghentdev
  - es03.coghentprod
  - nomad03.coghentprod

Alle vm's gebruiken puppet als config management. en worden dus voornamelijk beheerd door een puppet agent. SSH is ook voorzien en wordt door de technische mensen gebruikt om toegang te krijgen tot diverse tools (consul / vault / nomad / ...)

Storage & backup

Er zijn 2 backups die genomen worden:

- `consul`: Deze worden iedere 30 minuten gemaakt en bevatten de volgende gegevens:
  - consul kv:
    - Terraform statefiles
    - vault backend storage
    - configuratie paramaters
    - deployment parameters
  - consul services
  - consul sessions
- `ArangoDB`
  - Volledige dump van productie
  - Volledige dump van uat

Voor ElasticSearch is geen backup voorzien, dit om meerdere redenen:

- Data kan opnieuw gegenereerd worden indien lost
- De data is reeds een cluster over 3 nodes, dus data loss is onwaarschijnlijk

Beide backups worden voor een maximale duur van 21 dagen bijgehouden.

De backup en restore gebeurt door een nomad scheduled task. Bij een restore dien je de data directory door te geven waar je backup staat.

De backup van bovenstaande gegevens worden gestockeerd op een extern NFS volume:

Host	Mountpoint on nomad nodes
svm_ac010.gentgrp.gent.be:/cog_backup	/nomad/nfs_backup

Adlib2eventstream

Adlib2eventstream stelt Adlib databases bloot als event streams. Een event stream is een verzameling van objecten met versiebeheer (in deze context is versie zoals een eventstream event) en kan op elk moment worden bijgewerkt in hun eigen tempo (trage & snelle data). Op deze manier kunnen consumenten gemakkelijk de laatste wijzigingen ontdekken en gebruiken.

Application Er zijn 2 NodeJS 14 toepassingen die het adlib2eventstream proces ondersteunen:

- Adlib-backend: Eerst wordt een Adlib-database opgehaald en wordt de data gemapt naar Linked Data in JSON-LD formaat waarna het wordt opgeslagen in een database.

- Eventstream-API: Hierna worden de Linked Data Fragmenten ontsloten vanuit deze database volgens de Evenstream specificatie.

Technology

- De Adlib WebAPI is een REST web service geleverd door Axiell ALM. Zie Axiell WebAPI site - search .
- Zowel de adlib-backend als de eventstream API zijn NodeJS toepassingen.
- De database draait op MSSQL server.

Infrastructuur De adlib2eventstream draait op Red Hat OpenShift.

Git repositories (GitHub) De NodeJS toepassingen zijn beschikbaar als Open Source repositories op GitHub:

- StadGent/node_service_adlib-backend
- StadGent/node_service_eventstream-api

Installatie

Volgende `docker-compose` & `.env` geeft een inzicht in hoe een volledig systeem kan opgezet worden en hoe alle services aan elkaar gelinked worden.

docker-compose.yml

```
---
version: "3.4"

services:

  # Traefik: https://doc.traefik.io/traefik/
  traefik:
    image: traefik:v2.5
    volumes:
      - "/var/run/docker.sock:/var/run/docker.sock:ro"
    command:
      - --log.level=INFO
      - --providers.docker=true
      - --providers.docker.exposedbydefault=false
      - --api.insecure=true
      - --entrypoints.web.address=:80
      - --accesslog=true
    networks:
      - traefik
      - inuits-dams
```

```
    ports:
      - "${PROJECT_PORT}:80"
      - "8080"
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-traefik.rule=Host(`traefik.${PR
      - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-traefik.loadbalancer.server.po
    restart: unless-stopped

  collection-api:
    image: registry-dev.cloud.inuits.io/inuits/dams-collection-api:latest
    build:
      context: collection-api
      dockerfile: docker/Dockerfile
    environment:
      - DOCKER_BUILDKIT

      - DB_ENGINE

      - ARANGO_DB_HOST
      - ARANGO_DB_USERNAME
      - ARANGO_DB_PASSWORD
      - ARANGO_DB_NAME

      - MONGO_DB_HOST
      - MONGO_DB_PORT
      - MONGO_DB_USERNAME
      - MONGO_DB_PASSWORD
      - MONGO_DB_NAME

      - RABMQ_RABBITMQ_URL
      - RABMQ_SEND_EXCHANGE_NAME
      - EVENT_DELAY=0

      - COLLECTION_API_URL
      - IMAGE_API_URL_EXT
      - JOB_API_URL
      - STORAGE_API_URL
      - STORAGE_API_URL_EXT

      - REQUIRE_TOKEN
      - STATIC_JWT
      - STATIC_ISSUER
      - STATIC_PUBLIC_KEY
      - ROLE_PERMISSION_FILE
      - SUPER_ADMIN_ROLE
      - REMOTE_TOKEN_VALIDATION

      - APPS_MANIFEST
    depends_on:
```

```
    - mongo
    - arangodb
    - rabbitmq
  networks:
    inuits-dams:
      aliases:
        - job-api
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-collection-api.service=${TRAEFI
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-collection-api.rule=Host(`colle
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-collection-api.loadbalancer.se
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-job-api.service=${TRAEFIK_PROJE
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-job-api.rule=Host(`job-api.${PR
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-job-api.loadbalancer.server.po
  profiles:
    - backend
    - import

iiif-manifest:
  image: registry-dev.cloud.inuits.io/inuits/dams-iiif-manifest:latest
  build:
    context: inuits-dams-iiif-manifest-mapper
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT

    - COLLECTION_API_URL
    - IMAGE_API_URL
    - IMAGE_API_URL_EXT
    - JOB_API_URL
    - PRESENTATION_API_URL

    - REQUIRE_TOKEN
    - STATIC_JWT
    - STATIC_ISSUER
    - STATIC_PUBLIC_KEY
    - ROLE_PERMISSION_FILE
    - SUPER_ADMIN_ROLE
    - REMOTE_TOKEN_VALIDATION
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-iiif-manifest.rule=Host(`iiif-m
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-iiif-manifest.loadbalancer.ser
  profiles:
    - backend

storage-api:
```

```
  image: registry-dev.cloud.inuits.io/inuits/dams-storage-api:latest
  build:
    context: storage-api
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT

    - RABMQ_RABBITMQ_URL
    - RABMQ_SEND_EXCHANGE_NAME

    - MINIO_ENDPOINT
    - MINIO_ACCESS_KEY
    - MINIO_SECRET_KEY
    - MINIO_BUCKET

    - COLLECTION_API_URL
    - STORAGE_API_URL
    - JOB_API_URL

    - REQUIRE_TOKEN
    - STATIC_JWT
    - STATIC_ISSUER
    - STATIC_PUBLIC_KEY
    - ROLE_PERMISSION_FILE
    - SUPER_ADMIN_ROLE
    - REMOTE_TOKEN_VALIDATION
  depends_on:
    - minio
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-storage-api.rule=Host(`storage-
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-storage-api.loadbalancer.serve
  profiles:
    - backend
    - import

search-api:
  image: registry-dev.cloud.inuits.io/inuits/dams-search-api:latest
  build:
    context: search-api
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT

    - ARANGO_DB_HOST
    - ARANGO_DB_USERNAME=${ARANGO_READER_USERNAME}
    - ARANGO_DB_PASSWORD=${ARANGO_READER_PASSWORD}
    - ARANGO_DB_NAME
```

```
    - SEARCH_ENGINE=elastic

    - COLLECTION_API_URL
    - ELASTIC_URL

    - REQUIRE_TOKEN
    - STATIC_JWT
    - STATIC_ISSUER
    - STATIC_PUBLIC_KEY
    - ROLE_PERMISSION_FILE
    - SUPER_ADMIN_ROLE
    - REMOTE_TOKEN_VALIDATION
  depends_on:
    - es
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-search-api.rule=Host(`search-ap
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-search-api.loadbalancer.server
  profiles:
    - backend

csv-import-service:
  image: registry-dev.cloud.inuits.io/inuits/dams-csv-import-service:latest
  build:
    context: dams-csv-import-service
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT

    - RABMQ_RABBITMQ_URL
    - RABMQ_SEND_EXCHANGE_NAME

    - COLLECTION_API_URL
    - JOB_API_URL

    - REQUIRE_TOKEN
    - STATIC_JWT
    - STATIC_ISSUER
    - STATIC_PUBLIC_KEY
    - ROLE_PERMISSION_FILE
    - SUPER_ADMIN_ROLE
    - REMOTE_TOKEN_VALIDATION

    - UPLOAD_SOURCE=/app/import
  volumes:
    - ./inuits-dams-data-seed-service/images:/app/import
  depends_on:
```

```
    - collection-api
    - storage-api
    - rabbitmq
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-csv-import-service.rule=Host(`c
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-csv-import-service.loadbalance
  profiles:
    - backend

ldes-import-service:
  image: registry-dev.cloud.inuits.io/inuits/dams-ldes-import-service:latest
  build:
    context: ldes-import-service
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT
    - EVENT_STREAM_BASE_URL=http://data-seed-service
    - REQUEST_DELAY=0
    - COLLECTION_API_URL
    - STATIC_JWT
  depends_on:
    - collection-api
    - storage-api
    - data-seed-service
  profiles:
    - import
  networks:
    - inuits-dams

data-seed-service:
  image: nginx:stable-alpine
  environment:
    - DOCKER_BUILDKIT
  volumes:
    - ./inuits-dams-data-seed-service/ldes:/usr/share/nginx/html:ro
  profiles:
    - import
  networks:
    - inuits-dams

convert-images:
  image: dpokidov/imagemagick
  environment:
    - DOCKER_BUILDKIT
  volumes:
    - ./inuits-dams-data-seed-service/images:/images
  working_dir: "/images"
```

```
    entrypoint: "/images/convert_images.sh"
    profiles:
      - import
    networks:
      - inuits-dams

  import-images:
    image: curlimages/curl
    environment:
      - DOCKER_BUILDKIT

      - CSV_IMPORTER_URL

      - STATIC_JWT
    volumes:
      - ./inuits-dams-data-seed-service/images:/images
    working_dir: "/images"
    entrypoint: "/images/import_images.sh"
    profiles:
      - import
    networks:
      - inuits-dams

  import-sixth-collection-entity:
    image: curlimages/curl
    environment:
      - DOCKER_BUILDKIT

      - COLLECTION_API_URL

      - STATIC_JWT
    volumes:
      - ./inuits-dams-data-seed-service/sixth_collection_data_body.json:/sixth_collection_
      - ./inuits-dams-data-seed-service/import_sixth_collection_entity.sh:/import_sixth_co
    working_dir: "/"
    entrypoint: "/import_sixth_collection_entity.sh"
    profiles:
      - import
    networks:
      - inuits-dams

  elastic-indexer-service:
    image: registry-dev.cloud.inuits.io/inuits/dams-elastic-indexer-service:latest
    build:
      context: dams-elastic-indexer-service
      dockerfile: docker/Dockerfile
    environment:
      - DOCKER_BUILDKIT

      - COLLECTION_API_URL
```

```
    - ELASTIC_URL

    - RABMQ_RABBITMQ_URL
    - RABMQ_SEND_EXCHANGE_NAME

    - STATIC_JWT
  depends_on:
    - collection-api
    - es
    - rabbitmq
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-elastic-indexer-service.rule=Ho
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-elastic-indexer-service.loadba
  profiles:
    - backend

transcode-service:
  image: registry-dev.cloud.inuits.io/inuits/dams-transcode-service:latest
  build:
    context: dams-transcode-service
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT

    - RABMQ_RABBITMQ_URL
    - RABMQ_SEND_EXCHANGE_NAME

    - COLLECTION_API_URL
    - STORAGE_API_URL
    - JOB_API_URL

    - REQUIRE_TOKEN
    - STATIC_JWT
    - STATIC_ISSUER
    - STATIC_PUBLIC_KEY
    - ROLE_PERMISSION_FILE
    - SUPER_ADMIN_ROLE
    - REMOTE_TOKEN_VALIDATION
  depends_on:
    - collection-api
    - storage-api
    - rabbitmq
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-transcode-service.rule=Host(`tr
```

```
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-transcode-service.loadbalancer
  profiles:
    - backend

antivirus-service:
  image: registry-dev.cloud.inuits.io/inuits/dams-antivirus-service:latest
  build:
    context: dams-antivirus-service
    dockerfile: docker/Dockerfile
  environment:
    - DOCKER_BUILDKIT

    - RABMQ_RABBITMQ_URL
    - RABMQ_SEND_EXCHANGE_NAME

    - COLLECTION_API_URL
  depends_on:
    - collection-api
    - storage-api
    - rabbitmq
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-antivirus-service.rule=Host(`an
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-antivirus-service.loadbalancer
  profiles:
    - backend

minio:
  image: minio/minio:RELEASE.2021-07-27T02-40-15Z
  environment:
    - MINIO_ROOT_USER=${MINIO_ACCESS_KEY}
    - MINIO_ROOT_PASSWORD=${MINIO_SECRET_KEY}
    # Do NOT use MINIO_DOMAIN or MINIO_SERVER_URL with Traefik
    - MINIO_BROWSER_REDIRECT_URL=http://minio-console.${PROJECT_DOMAIN}:${PROJECT_PORT}
  volumes:
    - minio:/data
  entrypoint: sh
  command:
    - "-c"
    - rm -rf /data/dams-iiif-cache && mkdir -p /data/dams && mkdir -p /data/dams-iiif-ca
  networks:
    - inuits-dams
  expose:
    - "9000"
    - "9001"
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-minio.service=${TRAEFIK_PROJECT
```

```
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-minio.rule=Host(`minio.${PROJEC
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-minio.loadbalancer.server.port
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-minio-console.service=${TRAEFIK
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-minio-console.rule=Host(`minio-
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-minio-console.loadbalancer.ser
  profiles:
    - backend
    - import

cantaloupe:
  image: registry-dev.cloud.inuits.io/inuits/dams-cantaloupe:latest
  build:
    context: dams-cantaloupe
    dockerfile: docker/Dockerfile
  environment:
    - COLLECTION_API_URL
    - STORAGE_API_URL

    - STATIC_JWT
  volumes:
    - ./docker-compose/cantaloupe/cantaloupe.properties:/etc/cantaloupe/cantaloupe.prope
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-cantaloupe.rule=Host(`cantaloup
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-cantaloupe.loadbalancer.server
  profiles:
    - backend

mongo:
  image: mongo:3.6.23
  volumes:
    - mongo:/data/db
  command: --nojournal
  networks:
    - inuits-dams
  expose:
    - "27017"
  profiles:
    - backend
    - import

arangodb:
  image: arangodb:3.9
  environment:
    - ARANGO_DB_NAME
    - ARANGO_ROOT_PASSWORD
    - ARANGO_READER_USERNAME
    - ARANGO_READER_PASSWORD
```

```
  volumes:
    - arango:/var/lib/arangodb3
    - ./docker-compose/arango/create-arango-users.js:/docker-entrypoint-initdb.d/create-
    - ./docker-compose/arango/entrypoint.sh:/entrypoint.sh
  networks:
    - inuits-dams
  expose:
    - "8529"
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-arangodb.rule=Host(`arangodb.${
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-arangodb.loadbalancer.server.p
  profiles:
    - backend
    - import

keycloak:
  hostname: keycloak.${PROJECT_DOMAIN} # Run keycloak on the same host as the external
  image: jboss/keycloak:11.0.2
  environment:
    - "DB_VENDOR=h2"
    - "KEYCLOAK_USER=${LOCAL_KEYCLOAK_MASTER_USER-admin}"
    - "KEYCLOAK_PASSWORD=${LOCAL_KEYCLOAK_MASTER_PASSWORD-admin}"
    - "KEYCLOAK_IMPORT=/tmp/realm-export.json"
    - "KEYCLOAK_LOGLEVEL=DEBUG" # Default is "INFO"
    - "KEYCLOAK_HTTP_PORT=${PROJECT_PORT}" # Only to be used in the "assure user" scrip
  volumes:
    - keycloak:/opt/jboss/keycloak/standalone/data
    - ./docker-compose/keycloak/realm-export.json:/tmp/realm-export.json
  command: "-b 0.0.0.0 -Djboss.http.port=${PROJECT_PORT}" # Run keycloak on the same po
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-keycloak.rule=Host(`keycloak.${
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-keycloak.loadbalancer.server.p
  profiles:
    - backend

dashboard:
  image: registry-dev.cloud.inuits.io/inuits/inuits-dams-frontend:latest
  build:
    context: inuits-dams-frontend
    dockerfile: docker/Dockerfile
    args:
      NPM_CONFIG_ALWAYS_AUTH: ${NPM_CONFIG_ALWAYS_AUTH}
      NPM_CONFIG_REGISTRY: ${NPM_CONFIG_REGISTRY}
      NPM_CONFIG__AUTH_TOKEN: ${NPM_CONFIG__AUTH_TOKEN}
  environment:
    - VUE_APP_AUTH=false
```

```
    - VUE_APP_INDEX=false
    - VUE_APP_CONFIG_URL=/api/config
    - NPM_CONFIG_ALWAYS_AUTH
    - NPM_CONFIG_REGISTRY
    - NPM_CONFIG__AUTH_TOKEN
  profiles:
    - frontend
  networks:
    - inuits-dams
  labels:
    - "traefik.enable=true"
    - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-dashboard.rule=Host(`dashboard.
    - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-dashboard.loadbalancer.server.

graphql:
  image: registry-dev.cloud.inuits.io/inuits/inuits-dams-graphql:latest
  env_file:
    - ".env"
  build:
    context: inuits-dams-graphql-service
    dockerfile: docker/Dockerfile
    args:
      NPM_CONFIG_ALWAYS_AUTH: ${NPM_CONFIG_ALWAYS_AUTH}
      NPM_CONFIG_REGISTRY: ${NPM_CONFIG_REGISTRY}
      NPM_CONFIG__AUTH_TOKEN: ${NPM_CONFIG__AUTH_TOKEN}
  environment:
    - PORT=4001

    - APOLLO_GRAPHQL_PATH=/api/graphql
    - APOLLO_INTROSPECTION=true
    - APOLLO_PLAYGROUND=false
    - APOLLO_CLIENT_SECRET
    - APOLLO_SESSION_SECRET

    - OAUTH_BASE_URL
    - OAUTH_BASE_URL_FRONTEND
    - OAUTH_CLIENT_ID
    - OAUTH_TOKEN_ENDPOINT=/protocol/openid-connect/token
    - OAUTH_AUTH_ENDPOINT=/protocol/openid-connect/auth
    - OAUTH_API_CODE_ENDPOINT=/api/auth_code

    - COLLECTION_API_URL
    - CSV_IMPORTER_URL
    - DAMS_FRONTEND_URL
    - GRAPHQL_ENDPOINT=/api/graphql
    - IMAGE_API_URL
    - IMAGE_API_URL_EXT
    - SEARCH_API_URL
    - STORAGE_API_URL
```

```
      - STATIC_JWT

      - IGNORE_PERMISSIONS
    networks:
      - inuits-dams
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-graphql.rule=Host(`dashboard.${
      - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-graphql.loadbalancer.server.po
    profiles:
      - frontend

  es:
    image: docker.elastic.co/elasticsearch/elasticsearch-oss:7.10.2
    environment:
      - action.auto_create_index=false
      - discovery.type=single-node
      - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
    volumes:
      - es:/usr/share/elasticsearch/data
    networks:
      - inuits-dams
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-elasticsearch.rule=Host(`elasti
      - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-elasticsearch.loadbalancer.ser
    profiles:
      - backend

  rabbitmq:
    domainname: rabbitmq.com
    image: rabbitmq:3.8-management-alpine
    expose:
      - 5672
    networks:
      - inuits-dams
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.${TRAEFIK_PROJECT_NAMESPACE}-rabbitmq.rule=Host(`rabbitmq.${
      - "traefik.http.services.${TRAEFIK_PROJECT_NAMESPACE}-rabbitmq.loadbalancer.server.p
    healthcheck:
      test: rabbitmq-diagnostics -q ping
      interval: 10s
      timeout: 10s
      retries: 10
    profiles:
      - backend
      - import

networks:
```

```
  traefik:
  inuits-dams:
    name: inuits-dams

volumes:
  minio:
  arango:
  mongo:
  es:
  keycloak:
```

.env.dist

```
PROJECT_DOMAIN=dams.localhost
PROJECT_PORT=8100
TRAEFIK_PROJECT_NAMESPACE=dams

COLLECTION_API_URL=http://collection-api:5000/
COLLECTION_API_URL_EXT=http://collection-api.${PROJECT_DOMAIN}:${PROJECT_PORT}/
CSV_IMPORTER_URL=http://csv-import-service:5000
DAMS_FRONTEND_URL=http://dashboard.${PROJECT_DOMAIN}:${PROJECT_PORT}
ELASTIC_URL=http://es:9200
IMAGE_API_URL=http://cantaloupe:8182
IMAGE_API_URL_EXT=http://cantaloupe.${PROJECT_DOMAIN}:${PROJECT_PORT} # Cantaloupe refuses
JOB_API_URL=${COLLECTION_API_URL}
PRESENTATION_API_URL=http://iiif-manifest.${PROJECT_DOMAIN}:${PROJECT_PORT}/
SEARCH_API_URL=http://search-api:5000/
SEARCH_API_URL_EXT=http://search-api.${PROJECT_DOMAIN}:${PROJECT_PORT}/
STORAGE_API_URL=http://storage-api:5000/
STORAGE_API_URL_EXT=http://storage-api.${PROJECT_DOMAIN}:${PROJECT_PORT}/

APOLLO_CLIENT_SECRET=c9d9c9f7-e4b2-4bf3-b5a7-ad5e53d7ee31
APOLLO_SESSION_SECRET=heelgeheim

APPS_MANIFEST=apps/app_list.json

ARANGO_DB_HOST=http://arangodb:8529
ARANGO_DB_NAME=dams
ARANGO_DB_PASSWORD=***
ARANGO_DB_USERNAME=***
ARANGO_READER_PASSWORD=***
ARANGO_READER_USERNAME=***
ARANGO_ROOT_PASSWORD=***

COMPOSE_FILE=docker-compose.yml,docker-compose-dev.yml
COMPOSE_PATH_SEPARATOR=,
COMPOSE_PROJECT_NAME=dams

DB_ENGINE=arango
```

```
DOCKER_BUILDKIT=1

LOCAL_KEYCLOAK_CONTAINER=keycloak
LOCAL_KEYCLOAK_MASTER_PASSWORD=***
LOCAL_KEYCLOAK_MASTER_USER=admin
LOCAL_KEYCLOAK_PASSWORD=***
LOCAL_KEYCLOAK_REALM=dams
LOCAL_KEYCLOAK_USER=

MINIO_ACCESS_KEY=minio
MINIO_BUCKET=dams
MINIO_ENDPOINT=http://minio:9000
MINIO_SECRET_KEY=minio_secret

MONGO_DB_HOST=mongo
MONGO_DB_NAME=dams
MONGO_DB_PASSWORD=***
MONGO_DB_PORT=27017
MONGO_DB_USERNAME=***

NPM_CONFIG_ALWAYS_AUTH=true
NPM_CONFIG_REGISTRY=https://nexus.inuits.io/repository/npm
NPM_CONFIG__AUTH_TOKEN=

OAUTH_BASE_URL=http://keycloak:8100/auth/realms/dams
OAUTH_BASE_URL_FRONTEND=http://keycloak.dams.localhost:8100/auth/realms/dams
OAUTH_CLIENT_ID=dams-dashboard

RABMQ_RABBITMQ_URL=amqp://rabbitmq:5672?heartbeat=7200
RABMQ_SEND_EXCHANGE_NAME=dams

REMOTE_TOKEN_VALIDATION=False
REQUIRE_TOKEN=False
ROLE_PERMISSION_FILE=role_permission.json
SUPER_ADMIN_ROLE=role_super_admin

SIXTH_COLLECTION_ID=

IGNORE_PERMISSIONS='true'
```