OST
Ostschweizer
Fachhochschule

eikona - Challenge Project

OST — Ostschweizer Fachhochschule

Lukas Ribi, Dominik Castelberg, Pascal Christen

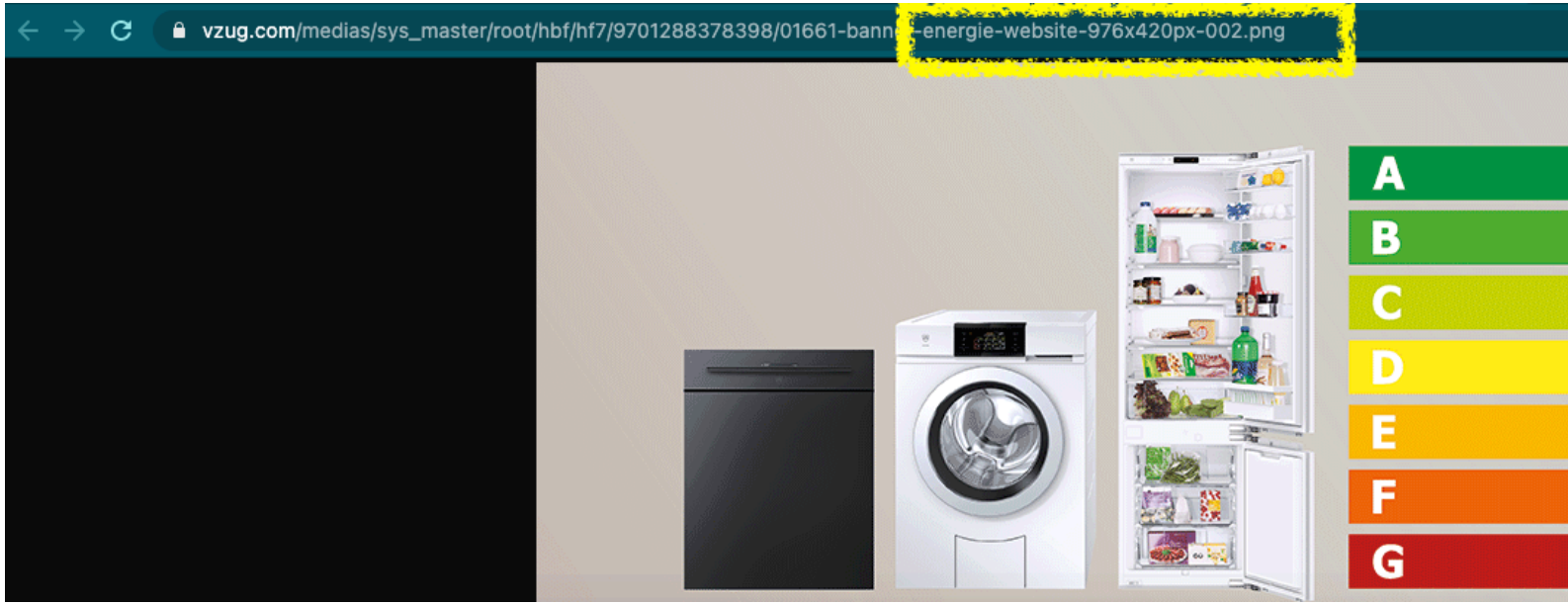
May 18, 2021

DS1 - Thomas Bocek

Goal

Problem: Often you need to resize, adjust quality, and convert on-demand for:

- Responsive websites
- Upload of an image

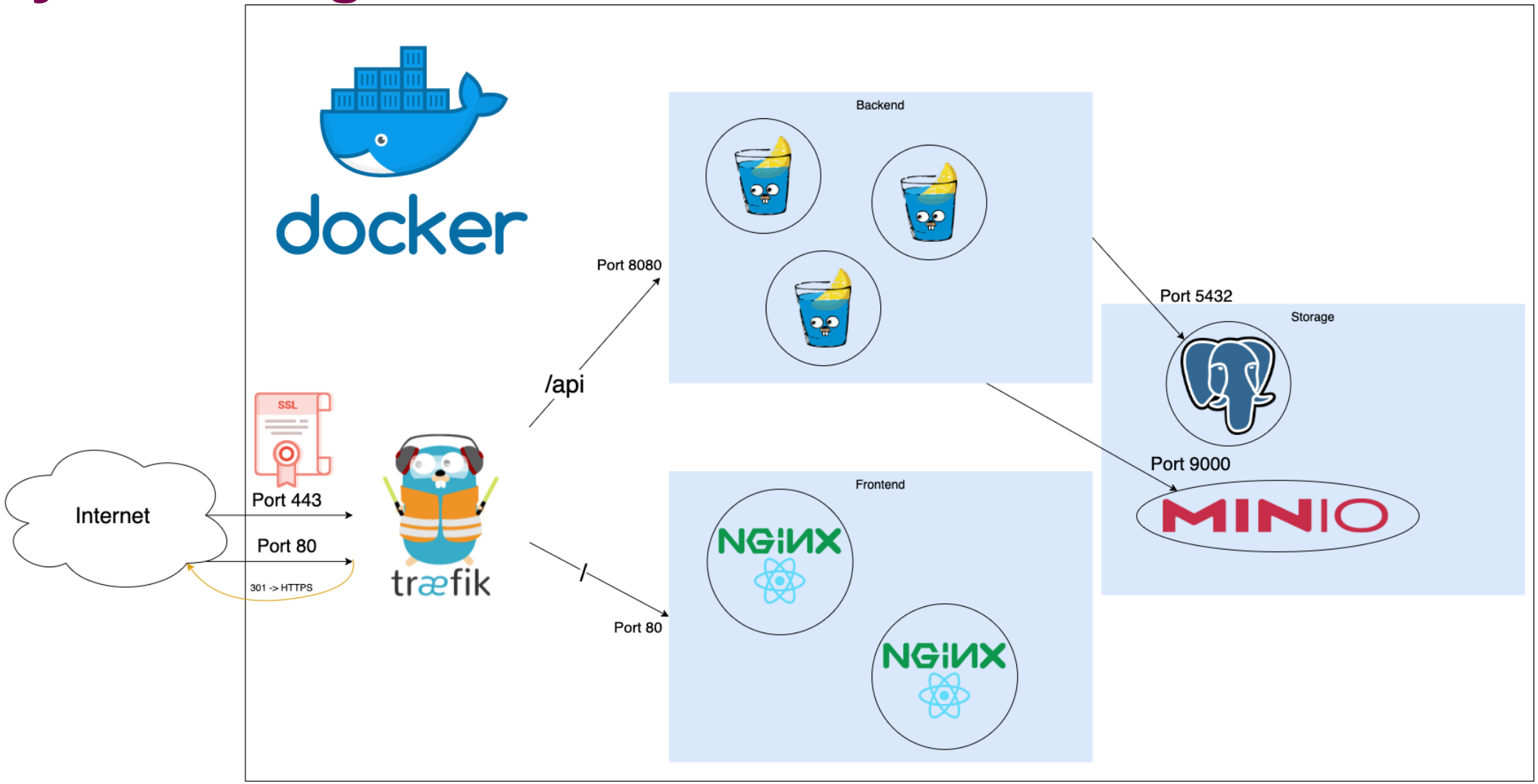


Solution: Our *εικόνα [iˈkɔna]* service converts the image **on-the-fly** and provides a simple *REST API*.

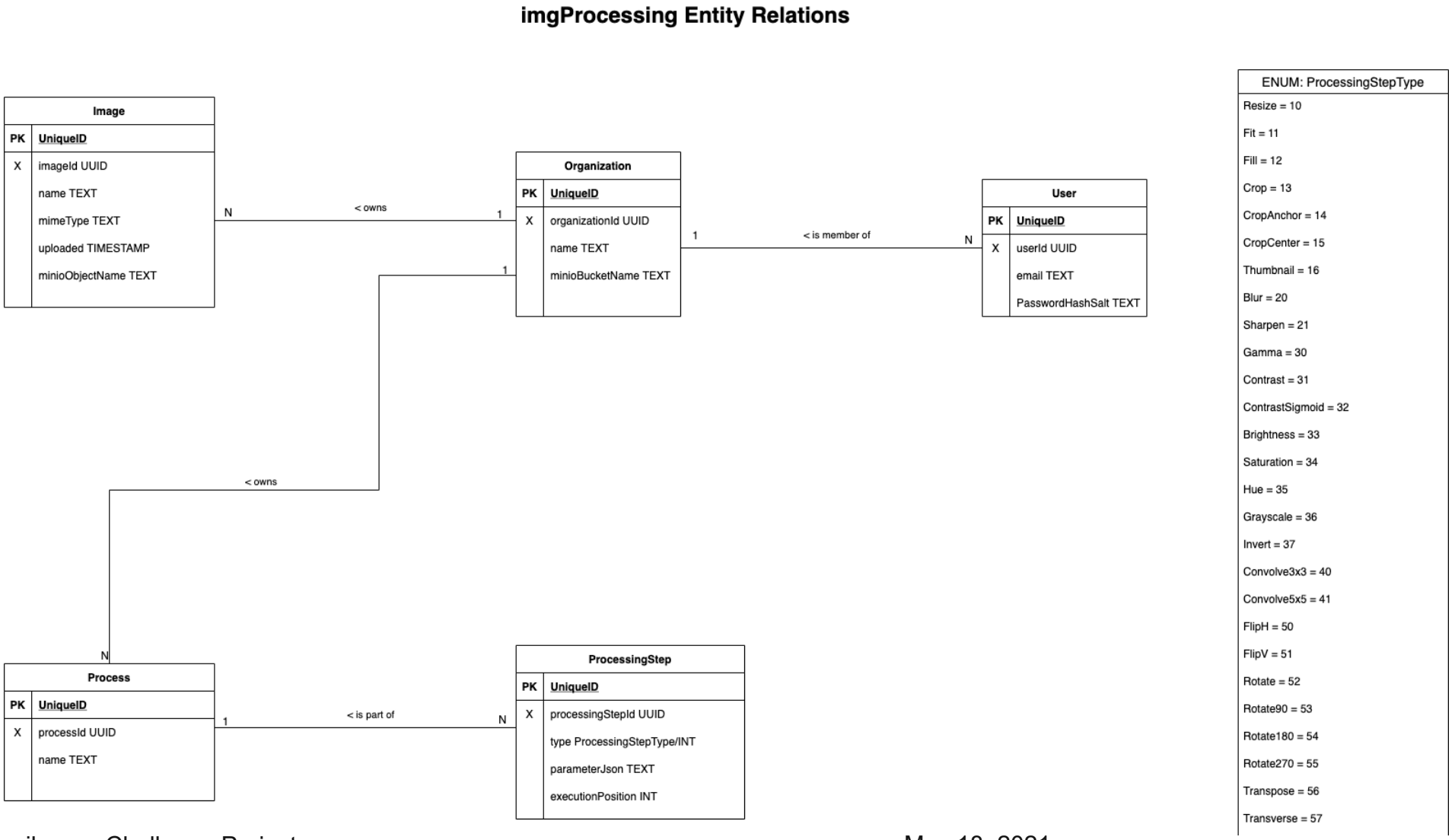
Setup

- Backend
  - go (1.16.4)
    - gin (1.7.1) with Ristretto (0.0.3)
    - JWT
- Frontend
  - ReactJS (17.0.2)
    - Material-UI
    - Served by Nginx (1.20.0)
- Loadbalancer
  - Traefik (2.4.8)
- Storage
  - PostgreSQL (13.2)
  - MinIO - Object Storage (RELEASE.2021-03-26)
- Presentation
  - LaTeX

System Diagram

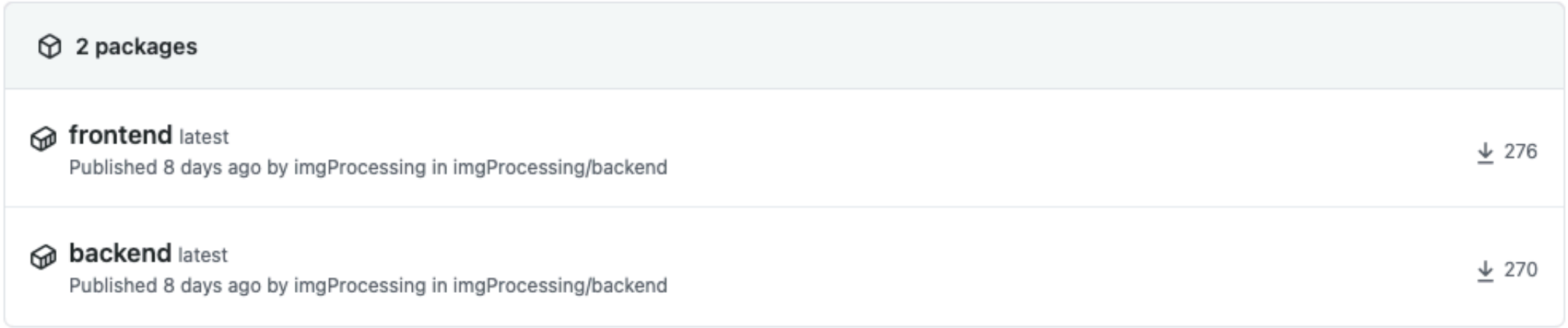


Data Model


imgProcessing Entity Relations
Image
PK UniqueID
X imageId UUID
name TEXT
mimeType TEXT
uploaded TIMESTAMP
minioObjectName TEXT
N
< owns
1
Organization
PK UniqueID
X organizationId UUID
name TEXT
minioBucketName TEXT
1
< is member of
N
User
PK UniqueID
X userId UUID
email TEXT
PasswordHashSalt TEXT
1
< owns
N
Process
PK UniqueID
X processId UUID
name TEXT
1
< is part of
N
ProcessingStep
PK UniqueID
X processingStepId UUID
type ProcessingStepType/INT
parameterJson TEXT
executionPosition INT
ENUM: ProcessingStepType
Resize = 10
Fit = 11
Fill = 12
Crop = 13
CropAnchor = 14
CropCenter = 15
Thumbnail = 16
Blur = 20
Sharpen = 21
Gamma = 30
Contrast = 31
ContrastSigmoid = 32
Brightness = 33
Saturation = 34
Hue = 35
Grayscale = 36
Invert = 37
Convolve3x3 = 40
Convolve5x5 = 41
FlipH = 50
FlipV = 51
Rotate = 52
Rotate90 = 53
Rotate180 = 54
Rotate270 = 55
Transpose = 56
Transverse = 57

Github Actions

- Docker Images - builded and versioned by Github Actions
  - Frontend
  - Backend
- Images hosted by GitHub Packages (ghcr.io)
- Security by scanning for CVE



docker-compose

- Development environment:
  - Debugging with delve
  - Live reloading with air
  - Local certificates with mkcert
- Production environment:
  - Only exposing Loadbalancer network (Port 443/80)
- Using multiple networks
- Working with environments

Loadbalancer

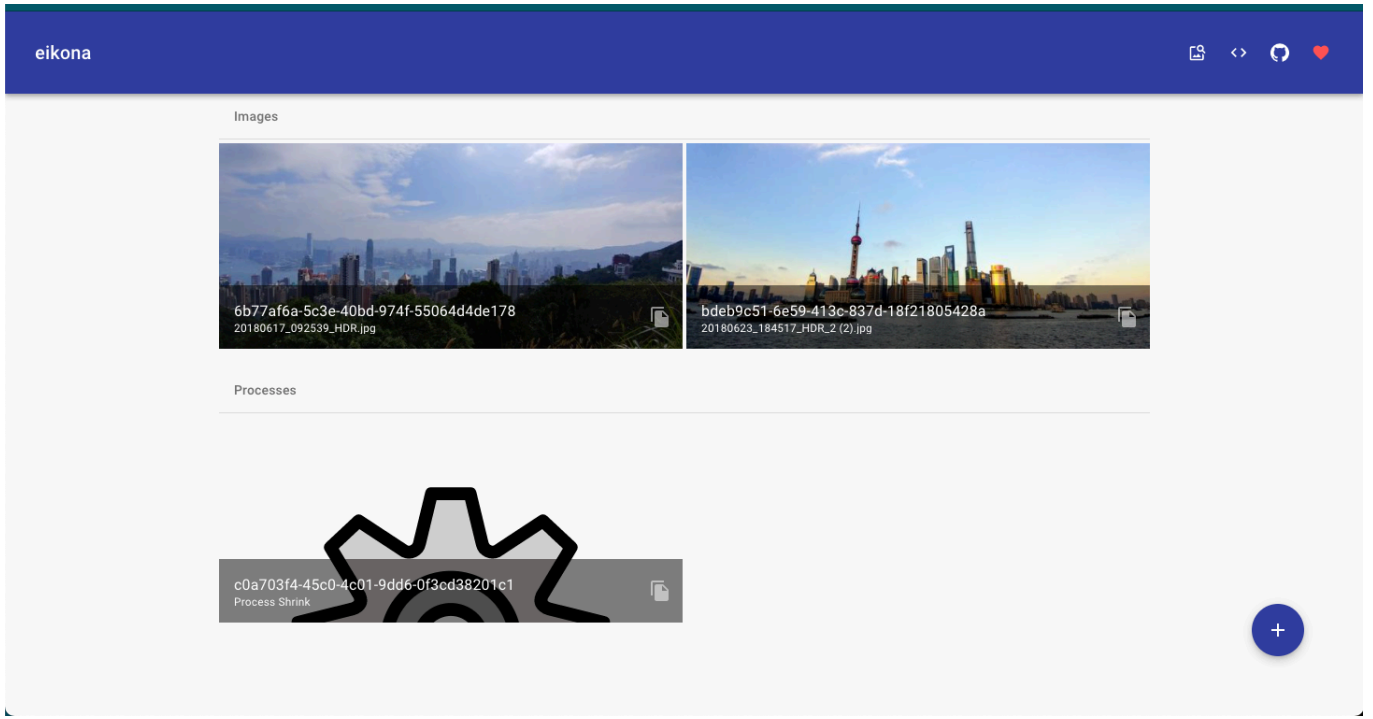
Traefik:

- Docker Socket mounted:
  - + Autodiscover
  - - Security
- HTTP -> HTTPS Middleware
- Healthchecks

Frontend

ReactJS with Material-UI:

- Responsive design
- Simple error handling
- Functions:
  - Dashboard
  - Upload Image
  - Create Process
  - API Docs



Backend

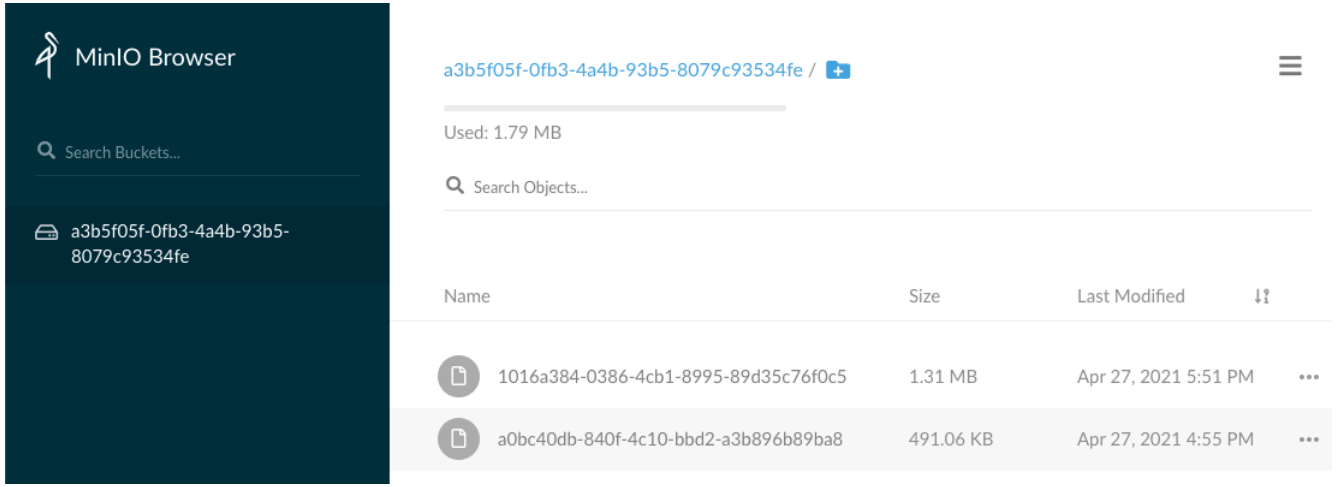
Golang:

- Image filtering using gift:
  - gift vs. imaging
  - Advanced image processing capabilities and filter chaining
  - 25 implemented operations right now (expanding...)
- REST API using gin:
  - JWT authentication using middleware
  - API Documentation using swagger
- Quick win: Simple caching using Ristretto at the single process level
  - Caching based on request (parameters / image id & pipeline combination
  - Enhancement: Use varnish, nginx or Traefik enterprise in front of the application to handle image caching

Storage

MinIO:

- High Performance, Kubernetes Native Object Storage
- Ready for distributed usage
- Compatible with Amazon S3
- Isolated buckets per organization with source images



Fulfillment of Requirements

- Dockerized: Used (images hosted on Github)
- Load balancing: Traefik
- Scalable service: Backend and Frontend
- JWT Authentication: Used
- Latest FOSS: Used

Demo

https://eikona.pesc.xyz

End

Source Code