



OST
Ostschweizer
Fachhochschule

eikona - Challenge Project

OST — Ostschweizer Fachhochschule

Lukas Ribi, Dominik Castelberg, Pascal Christen

May 18, 2021

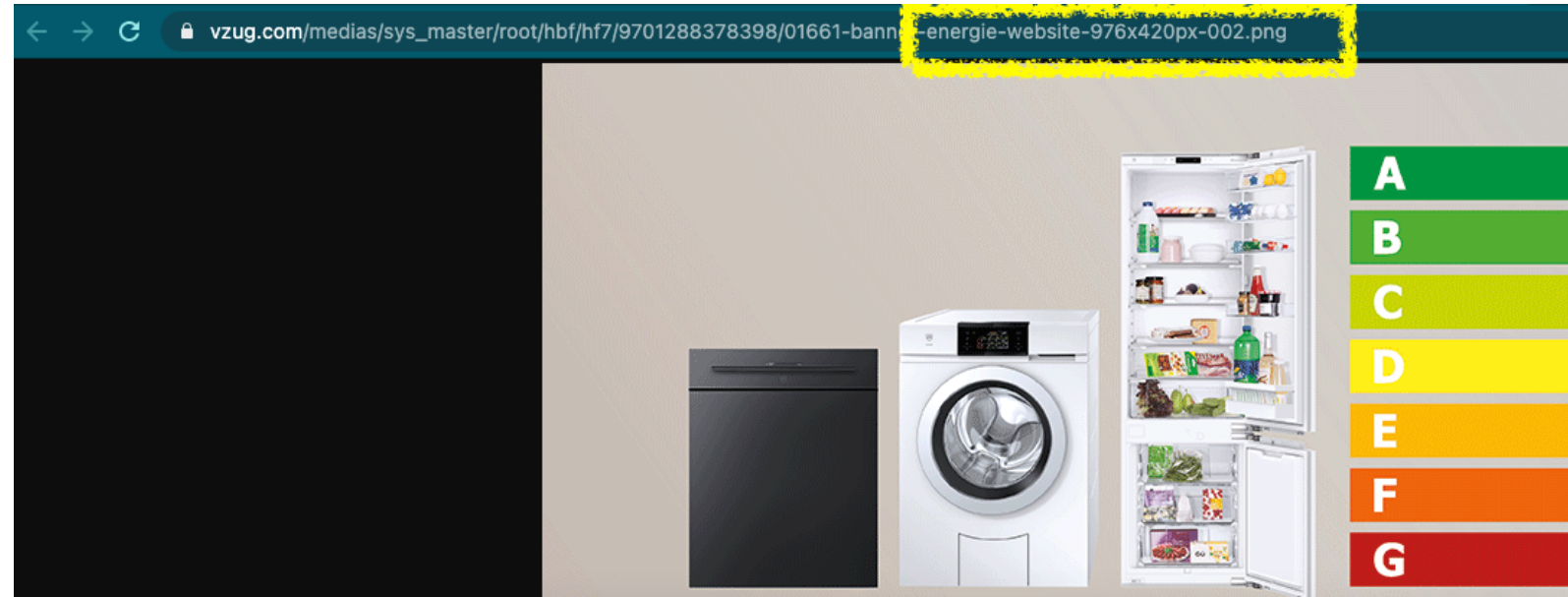
DS1 - Thomas Bocek

1. Introduction

Goal

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

- Responsive websites
- Upload of an image



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

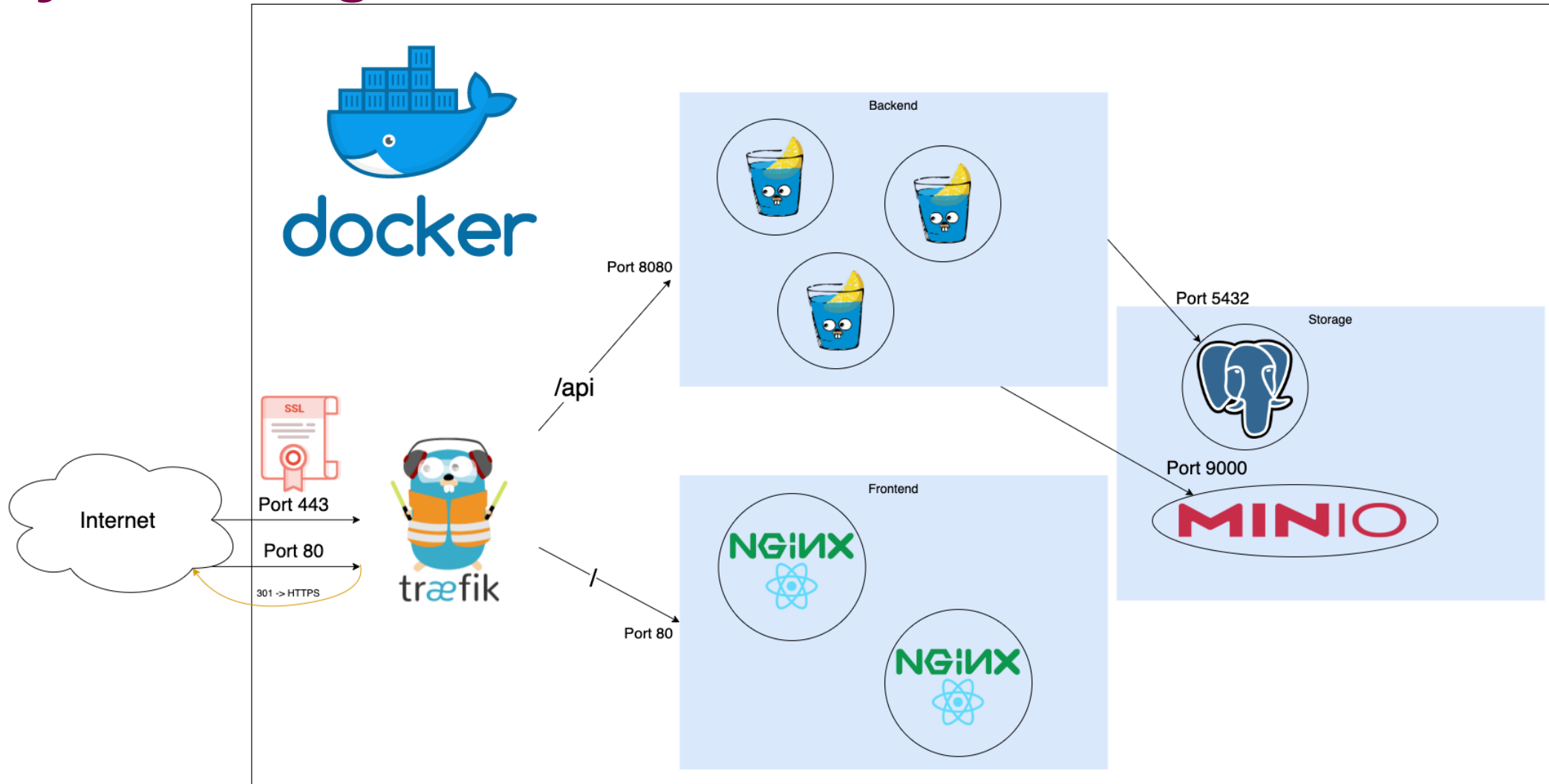
2. Technology

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**

2. Technology

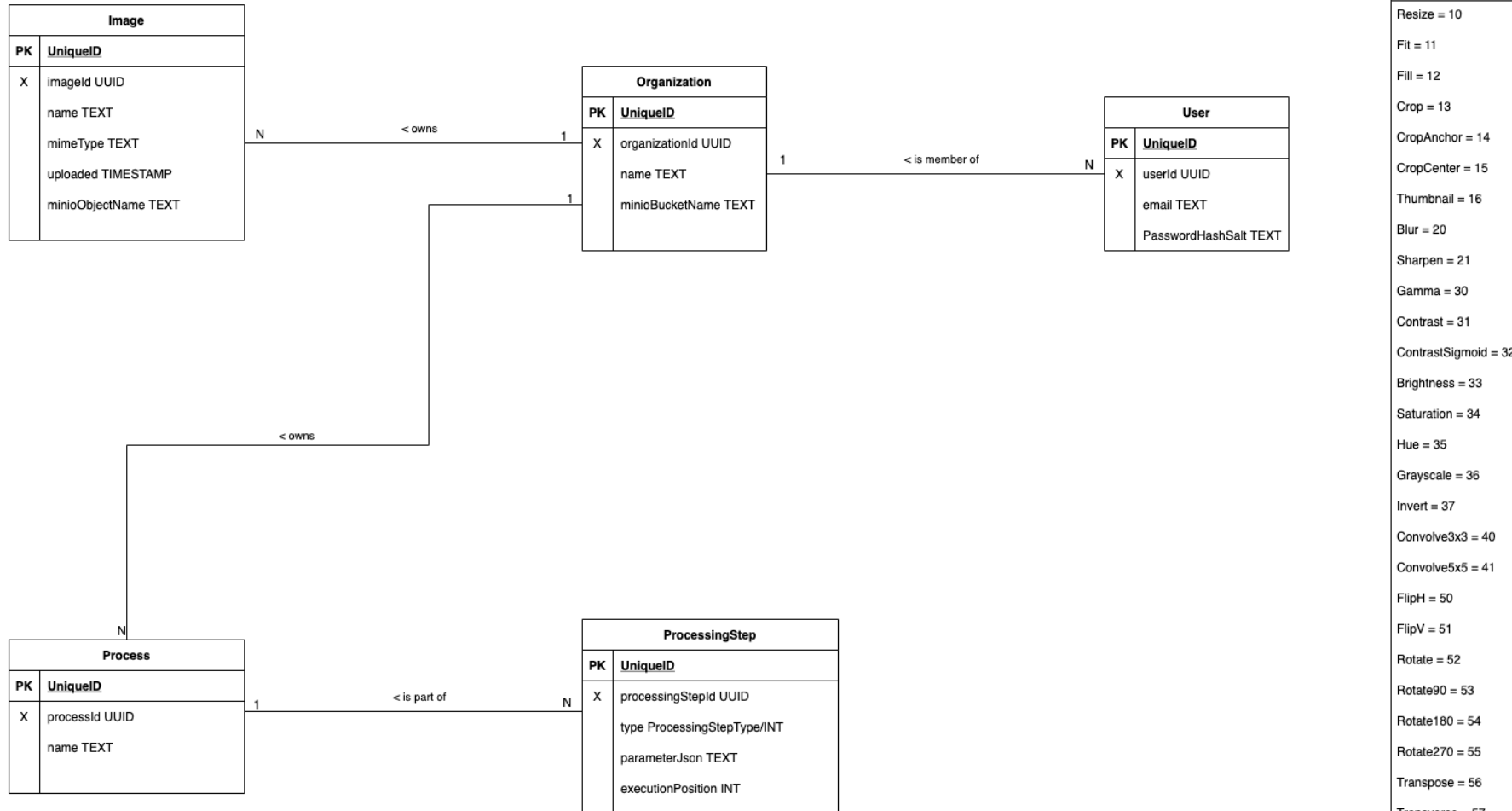
System Diagram



3. Development

Data Model

imgProcessing Entity Relations



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

3. Development

Github Actions

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

📦 2 packages		
📦 frontend latest	Published 8 days ago by imgProcessing in imgProcessing/backend	↓ 276
📦 backend latest	Published 8 days ago by imgProcessing in imgProcessing/backend	↓ 270

3. Development

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

3. Development

Loadbalancer

Traefik:

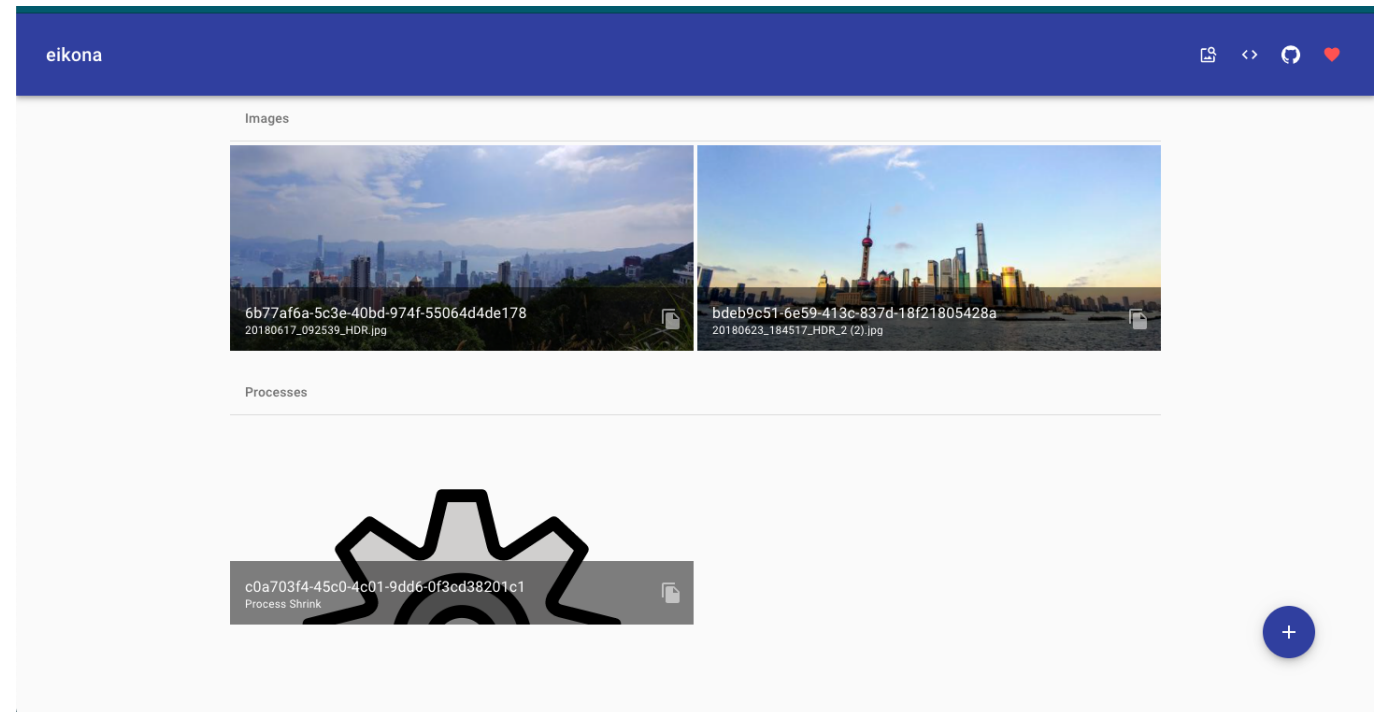
- Docker Socket mounted:
 - + Autodiscover
 - - Security
- HTTP -> HTTPS Middleware
- Healthchecks

3. Development

Frontend

ReactJS with Material-UI:

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



3. Development

Backend

Golang:

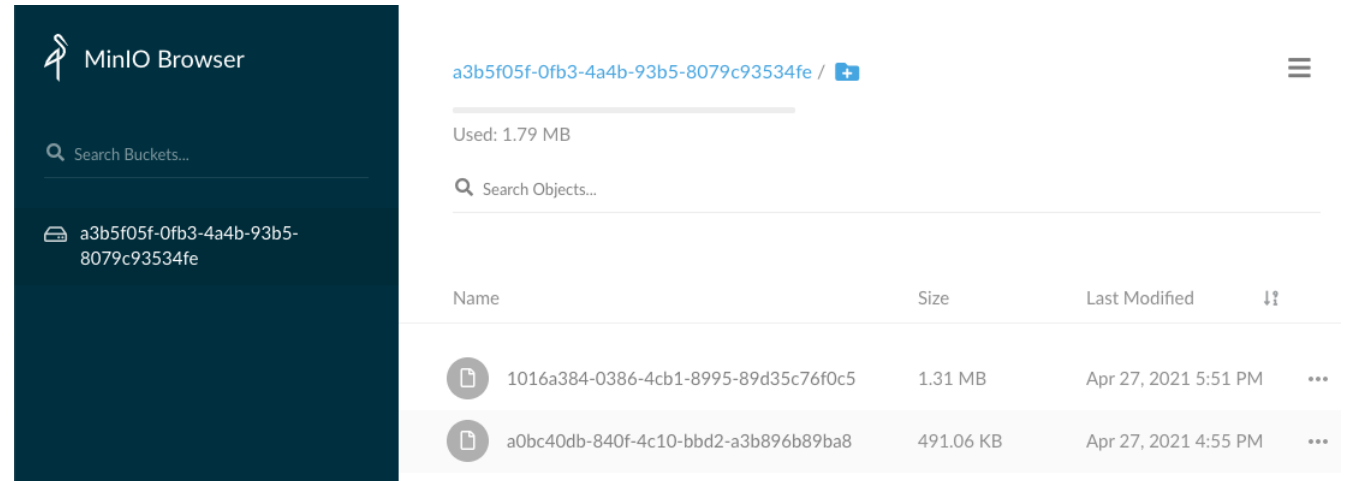
- Image filtering using **gift**:
 - **gift vs. imaging**
 - Advanced image processing capabilities and filter chaining
 - 25 implemented operations right now (expanding...)
- REST API using **gin**:
 - JWT authentication using middleware
 - API Documentation using **swagger**
- Quick win: Simple caching 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

3. Development

Storage

MinIO:

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



4. Legal

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](#)