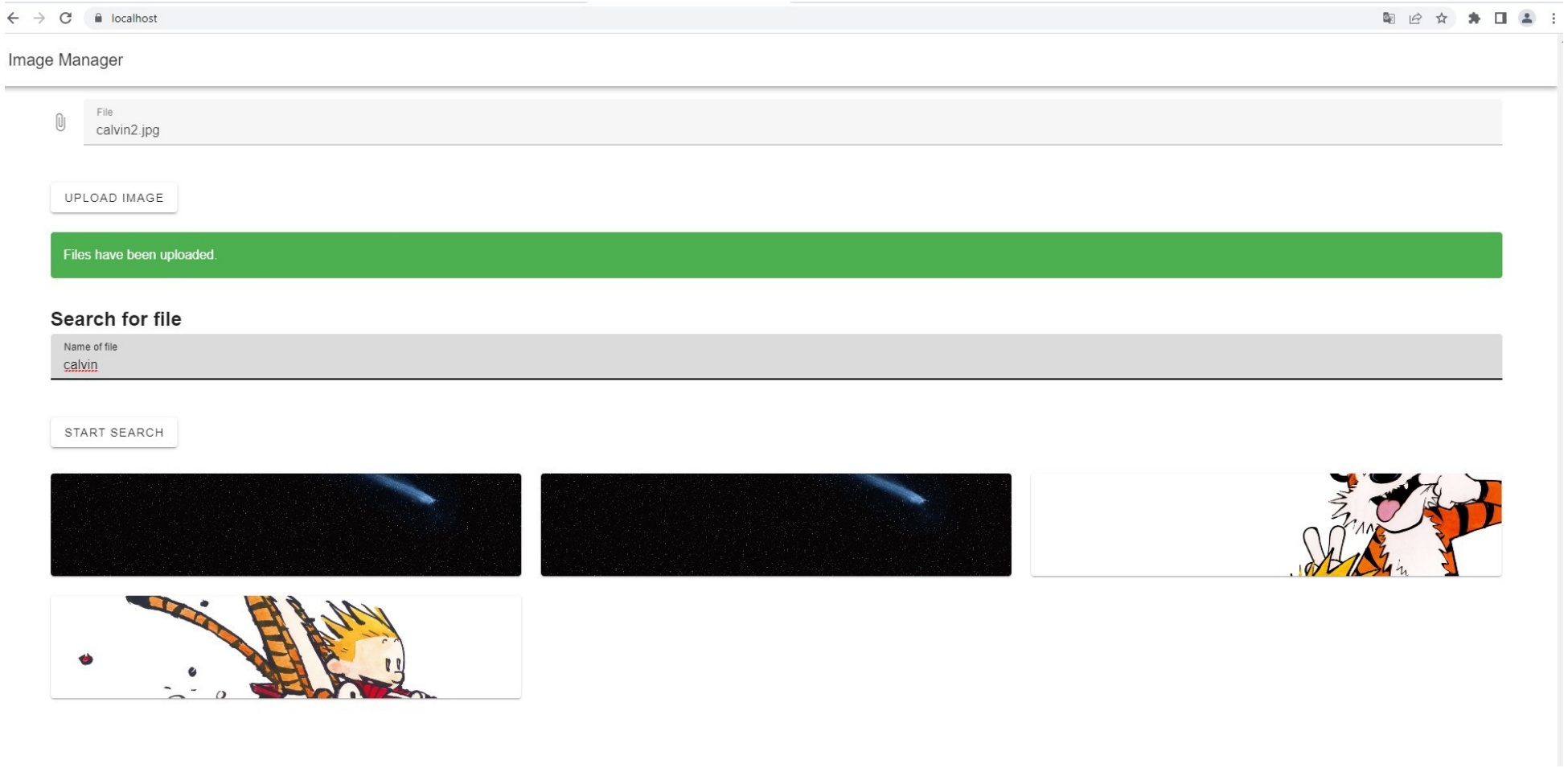
Distributed Systems FS 23
Challenge Task
Image Manager

By: Linus Flury, Natalia Gerasimenko, Kailing Peng

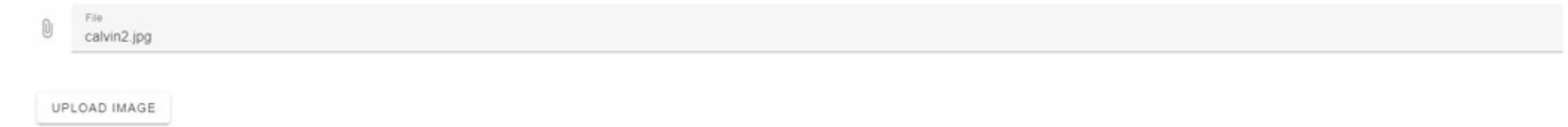
OST
Ostschweizer
Fachhochschule

Image Manager



Functionality

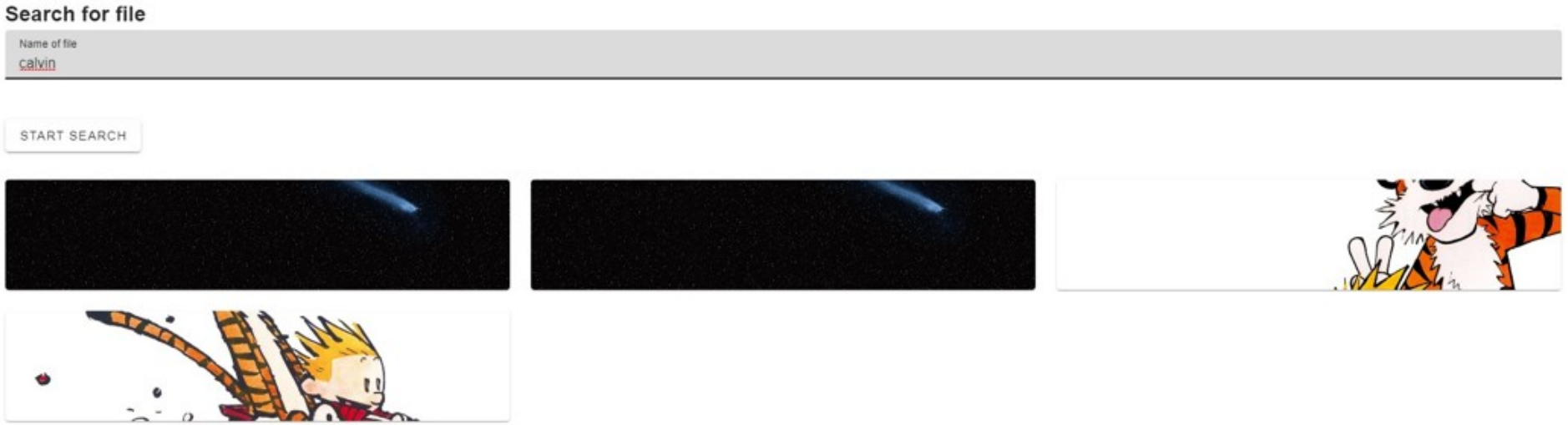
Upload images:



- Click paperclip to chose image
- Press „UPLOAD IMAGE“ to store image in database

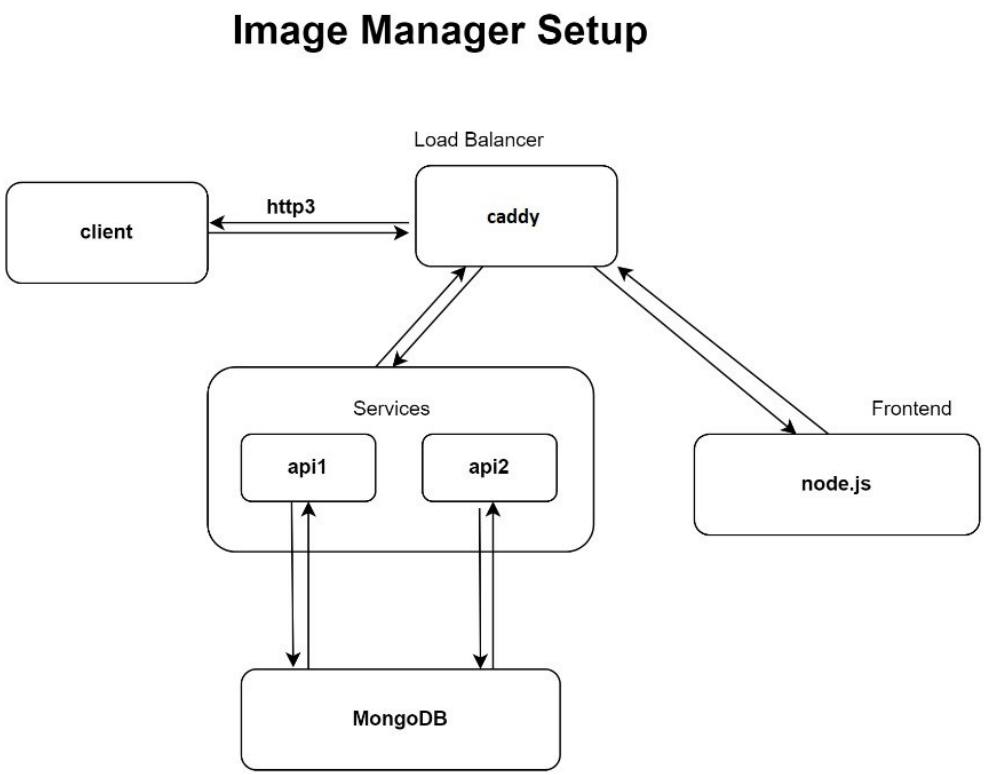
Functionality

Search images:



- Enter part of a filename that you want to look for
- Press „START SEARCH“ to show all stored images that contain the search term
- Click image for full view

Setup



- Dockerized setup, each smaller shape represent a container
- Caddy loadbalancer / reverse proxy, entrypoint
- Node.js webserver for webpage, frontend
- Node.js webservers for backend services
- MongoDB for persistence

Docker

Why Docker?

- Environment independent development
- Easy central managment of multiple server instances (including lifespan)
- Was mandated

What do we use it for

- Creating instance images of all parts
- Ordered creation of images (dependencies)
- Executing initial server startup commands for server images
- Starting/stopping backend services to prove proper healthchecks of loadbalancer

Loadbalancer

Why Caddy?

- http3 compatible (experimental)
- Loadbalancing + reverse proxy
- Caddy and Traefik were tried
- Traefik was used at the for the most part
- Actual http3 communication was only achieved with Caddy

What do we use it for

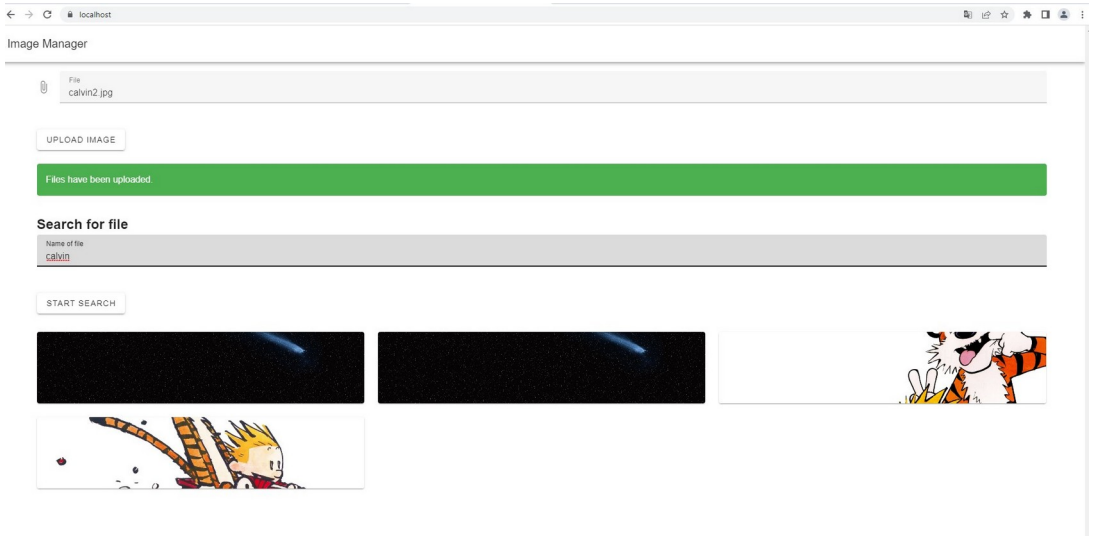
- http3 advertising
- http port redirecting to https
- https setup
- Routing: paths to docker container
- Loadbalancing for backend services, including healthchecks

Loadbalancer – http3

http3 involving client, frontend and backend services:

200	GET	localhost	VSnackbar.css	HTTP/3	script	js	3.79 KB	3.57 KB
200	GET	localhost	VSwitch.css	HTTP/3	script	js	3.02 KB	2.79 KB
200	GET	localhost	VSystemBar.css	HTTP/3	script	js	1.88 KB	1.66 KB
200	GET	localhost	VTabs.css	HTTP/3	script	js	2.31 KB	2.09 KB
200	GET	localhost	VTab.css	HTTP/3	script	js	1.25 KB	1.03 KB
200	GET	localhost	VTable.css	HTTP/3	script	js	5.74 KB	5.51 KB
200	GET	localhost	VTextarea.css	HTTP/3	script	js	2.11 KB	1.89 KB
200	GET	localhost	VThemeProvider.css	HTTP/3	script	js	915 B	687 B
200	GET	localhost	VTimeline.css	HTTP/3	script	js	17.61 KB	17.39 KB
200	GET	localhost	VTooltip.css	HTTP/3	script	js	1.43 KB	1.21 KB
200	GET	localhost	VVirtualScroll.css	HTTP/3	script	js	977 B	749 B
200	GET	localhost	materialdesignicons-webfont.woff2?v=7.2.96	HTTP/3	font	woff2	387.68 KB	387.43 ...
404	GET	localhost	vite.svg	HTTP/3	img	svg	127 B	0 B
200	GET	localhost	image	HTTP/3	xhr	json	377 B	175 B
	GET	localhost	1684787446282-image-Bildschirmfoto 2023-05-21 um 12.16.04.		img		0 B	0 B
308	GET	localhost	1684787446282-image-Bildschirmfoto 2023-05-21 um 12.16.04.	HTTP/1.1	img	png	209.02 KB	208.80...
200	GET	localhost	1684787446282-image-Bildschirmfoto 2023-05-21 um 12.16.04.	HTTP/3	img	png	208.89 KB	208.80...

Frontend



Vite

Frontend

Why Vite.js?

- Development server based on Node.js
- Supports Typescript
- Simplifies development through hot-reloading
- Default server for Vue.js projects
- A lot of prior experience by teammember

What do we use it for

- Serving web application
- Responding to user interaction
- Sending requests to backend

Frontend

Why Vue.js?

- JavaScript framework for single page applications
- Simple, lightweight, less overhead than e.g. Angular
- A lot of prior experience by teammember
- Styling via Vuetify

Why Vuetify?

- Open source UI library with good integration into Vue.js
- Includes styling tools and predefined styles and features
- Saves having to define shapes from scratch

Backend

Backend

Why Express.js?

- Based on Node.js
- We used open source code for webserver connected to mongoDB (do not reinvent the wheel) as base
- Adjusted express api endpoint handling to fit our needs
- Easy to create API endpoints
- No need for styling
- Alternatives: Fastify
- We had a working prototype using Express.js, decided to focus on core parts of task

Why mongoDB?

- Tiny project scale (big entries, but very few)
- DB type not that relevant
- Working prototype was found using mongoDB
- Needed very few adjustments
- Alternatives: PostgreSQL, SQLite, ….

Backend - API

Endpoints on services:

- GET /api/search/{searchTerm}
  - Returns filepath for stored image to be created on service out of db
- GET /api/files/{filepath}
  - Builds image from db for specified filepath and returns it
- POST /api/upload
  - Upload an image using GridFs Storage to images as binary in db
- *Get /api/health*
  - *Obsolete, was used for Traefik healthcheck, returns OK code 200 when active, was used as precaution to not interfere with routing*