Individuelle Abschlussarbeit BLJ

**CloudCore**

Autor: Jann Neuruer

Unterschrift

Datum Abgabe: 27.06.2025

Lehrfirma: EKZ Eltop

V1.2

Inhaltsverzeichnis

[Änderungstabelle 2](#_Toc200032391)

[Tagesjurnal 4](#_Toc200032392)

[Tagesjournal – Mittwoch, 5. Juni 2025 4](#_Toc200032393)

[Tagesjournal – Donnerstag, 6. Juni 2025 4](#_Toc200032394)

[Einleitung 5](#_Toc200032395)

[Aufgabenstellung 5](#_Toc200032396)

[Projektbeschreibung 5](#_Toc200032397)

[Raspberry PI einrichten 6](#_Toc200032398)

[Raspberry Pi vorbereiten mit Raspberry Pi Imager 6](#_Toc200032399)

[Mit dem Raspberry Pi verbinden (SSH) 6](#_Toc200032400)

[Statische IP-Adresse festlegen 6](#_Toc200032401)

[Docker installieren 7](#_Toc200032402)

[Selfhosted Platform Basisk 8](#_Toc200032403)

[Projektverzeichniss 8](#_Toc200032404)

[Uptime Kuma Contianer 8](#_Toc200032405)

[Admin-Konto für Uptime Kuma einrichten 9](#_Toc200032406)

[Nginx-Proxy konfigurieren 9](#_Toc200032407)

[Gemeinsames Docker-Netzwerk erstellen 10](#_Toc200032408)

[Monitoring Test 11](#_Toc200032409)

[Mein Titel 12](#_Toc200032410)

[Abbildungsverzeichnis 12](#_Toc200032411)

## Änderungstabelle

| Änderungsdatum | Version | Beschreibung | Wer |
| --- | --- | --- | --- |
| 04.06.2025 | V1.0 | Beginn Der Dokumentation | Jann Neururer |
| 05.06.2025 | V1.1 | Beginn Tag 2 | Jann Neururer |
| 06.06.2025 | V1.2 | Tag 3 | Jann Neuruer |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Tagesjurnal

### Tagesjournal – Mittwoch, 4. Juni 2025

Heute habe ich mit meinem Raspberry-Pi-Projekt gestartet. Zuerst habe ich das Image mit dem Raspberry Pi Imager auf die SD-Karte geschrieben und dabei direkt wichtige Einstellungen wie Hostname, Benutzername, Passwort, Tastatur-Layout und Zeitzone konfiguriert.

Danach habe ich mich per SSH auf den Pi verbunden. Anschliessend habe ich in der Datei /etc/systemd/network/10-eth0.network eine statische IP-Adresse gesetzt, damit mein Pi im Netzwerk zuverlässig erreichbar bleibt.

Ich habe das System mit apt update && apt upgrade auf den neusten Stand gebracht und danach Docker über das offizielle Installationsscript (get.docker.com) installiert.

Zum Schluss habe ich noch überprüft, ob Docker Compose verfügbar ist – was der Fall war. Jetzt ist der Pi bereit für den Einsatz als Selfhosting-Plattform.

### Tagesjournal – Donnerstag, 5. Juni 2025

Heute habe ich die Grundstruktur meines Projekts aufgebaut. Ich habe ein zentrales Projektverzeichnis namens cloudcore-project erstellt und darin für jeden Dienst eigene Unterordner angelegt (z. B. uptime-kuma, filebrowser, reverse-proxy, etc.).

Anschliessend habe ich den Container für Uptime Kuma gestartet – zuerst manuell mit docker run, später habe ich ein sauberes docker-compose.yml dafür erstellt. Danach konnte ich das Admin-Interface aufrufen und den ersten Benutzer anlegen.

Um Uptime Kuma übersichtlich erreichbar zu machen, habe ich Nginx als Reverse Proxy konfiguriert. Dazu habe ich eine eigene kuma.conf erstellt und Nginx per Docker Compose in einem eigenen Container gestartet.

Ich habe ausserdem ein eigenes Docker-Netzwerk (kuma-net) angelegt, damit Nginx und Kuma intern miteinander kommunizieren können. Zum Schluss habe ich überprüft, ob der Proxy funktioniert – was erfolgreich war.

Ich habe zudem im Dashboard von Kuma den ersten Monitoring-Eintrag für den Nginx-Container hinzugefügt, um dessen Verfügbarkeit zu prüfen.

Insgesamt konnte ich heute das Grundsystem erfolgreich bereitstellen und vernetzen.

### Tagesjournal – Freitag, 6. Juni 2025

Heute habe ich versucht, Uptime Kuma und Filebrowser über einen gemeinsamen Nginx-Reverse-Proxy erreichbar zu machen. Dazu habe ich im Verzeichnis cloudcore-project passende docker-compose.yml angelegt.

Alle Container laufen im selben Docker-Netzwerk namens Cloudnet, damit sie miteinander kommunizieren können. Ich habe die default.conf von Nginx so eingerichtet, dass Aufrufe über /kuma und /filebrowser korrekt an die jeweiligen Container weitergeleitet werden.

Für Uptime Kuma habe ich die Variable BASE\_PATH=/kuma gesetzt und diesen Pfad auch direkt in der Datenbank gespeichert. Trotzdem wurden beim Aufruf über /kuma keine Styles und Skripte geladen, es kam zu 404-Fehlern. Nach längerer Fehlersuche habe ich herausgefunden, dass Uptime Kuma aktuell nicht gut mit Subpfaden klarkommt.

## Einleitung

## Aufgabenstellung

Während der Abschlussprojektphase vom 2. bis 27. Juni haben wir die Aufgabe, ein selbstgewähltes IT-Projekt innerhalb von maximal 12 Arbeitstagen zu planen, umzusetzen und zu dokumentieren.

Ziel ist es, ein Thema aus dem eigenen Interesse oder dem Firmenumfeld zu wählen und daraus ein konkretes, praxisnahes Projekt zu entwickeln. Dabei sollen verschiedene Phasen durchlaufen werden: von der Themenfindung über die Planung und Umsetzung bis hin zur Präsentation.

Das Projekt soll technisch sinnvoll, realistisch umsetzbar und nachvollziehbar dokumentiert sein. Es wird erwartet, dass mindestens drei Zwischenziele (Milestones) definiert und erreicht werden. Am Schluss wird das Endergebnis in einer Live-Demo vorgestellt.

## Projektbeschreibung

Im Rahmen meines Abschlussprojekts realisiere ich den Aufbau einer sicheren Selfhosting-Plattform auf Basis von Docker. Ziel ist es, verschiedene Dienste wie ein Monitoring-System (Netdata, Uptime Kuma), eine webbasierte Dateiverwaltung (Filebrowser), ein internes Wiki (Bookstack) sowie DNS-Filterung (AdGuard Home) in einer isolierten und übersichtlich verwalteten Umgebung bereitzustellen.

Die Plattform wird auf einem Linux-Server oder Raspberry Pi betrieben und mit OPNsense als zentrale Firewall abgesichert. Zusätzlich werden Mechanismen wie Firewall-Regeln, Fail2ban, Benutzerrollen, Reverse Proxy mit HTTPS und Backups integriert.

Durch dieses Projekt kann ich mein Wissen in den Bereichen Linux, Containerisierung, Netzwerksicherheit und Plattformbetrieb praxisnah anwenden und vertiefen. Die erstellte Umgebung ist modular aufgebaut und auch nach Projektabschluss privat weiter nutzbar.

## Raspberry PI einrichten

### Raspberry Pi vorbereiten mit Raspberry Pi Imager

Im Raspberry Pi Imager werden die wichtigsten Grundeinstellungen gesetzt:

Hostname:**piCloudcore**

Benutzername&Passwort festgelegt

Spracheinstellung: Europe/Zurich, Tastatur: CHWLAN deaktiviert (da per LAN verbunden)

Diese Einstellungen sorgen dafür, dass der Pi sofort im Netzwerk erkannt wird und via piCloudcore.local erreichbar ist.

### Mit dem Raspberry Pi verbinden (SSH)

Verbinde dich über ssh mit dem Host und Benutzernamen auf den Raspberry PI:

**ssh piCloudcore@piCloudcore.local**

### Statische IP-Adresse festlegen

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte BeschreibungBearbeite die datei: **sudo nano /etc/systemd/network/10-eth0.network**

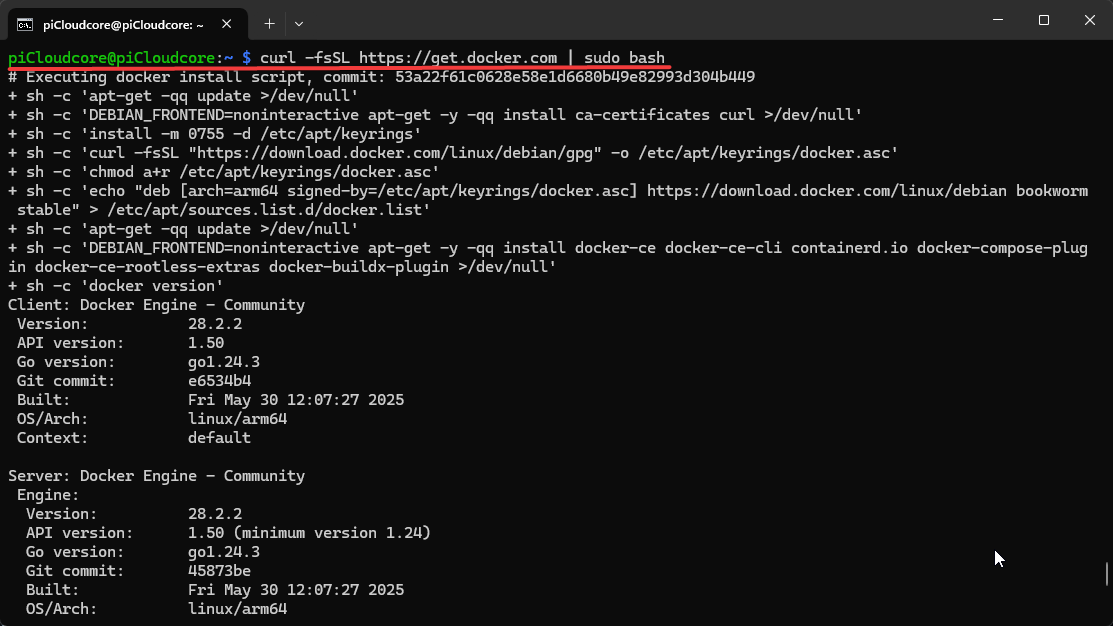
Und Fülle die Datei wie in diesem Screenshot:

Damit bekommt der PI immer dieselbe IP Adresse und dies ist Praktisch wen ich immer wieder darauf connecten muss.

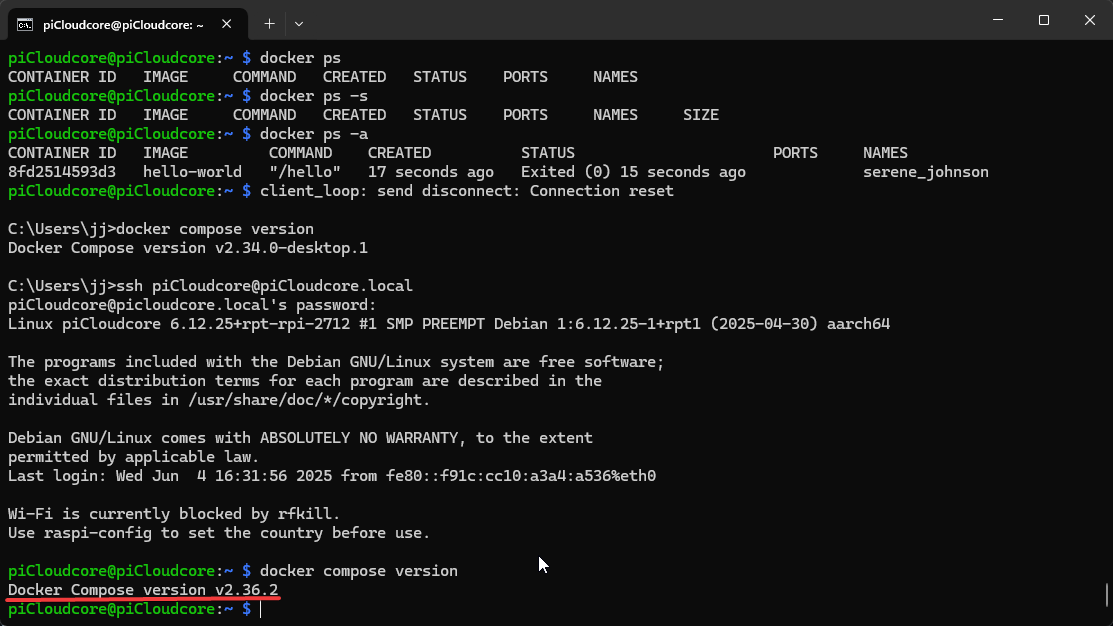
### Docker installieren

Benutze diesen Einzeiler hier wie im Screenshot:

curl -fsSL https://get.docker.com | sudo bash

Docker Compose muss nicht mehr installiert werden da bei den neueren versionen dies gleich mitinstalliert wird.

Die Installation kann man nun mit Docker version checken und mit Docker compose version kann man nachsehen, ob Docker Compose schon installiert ist.

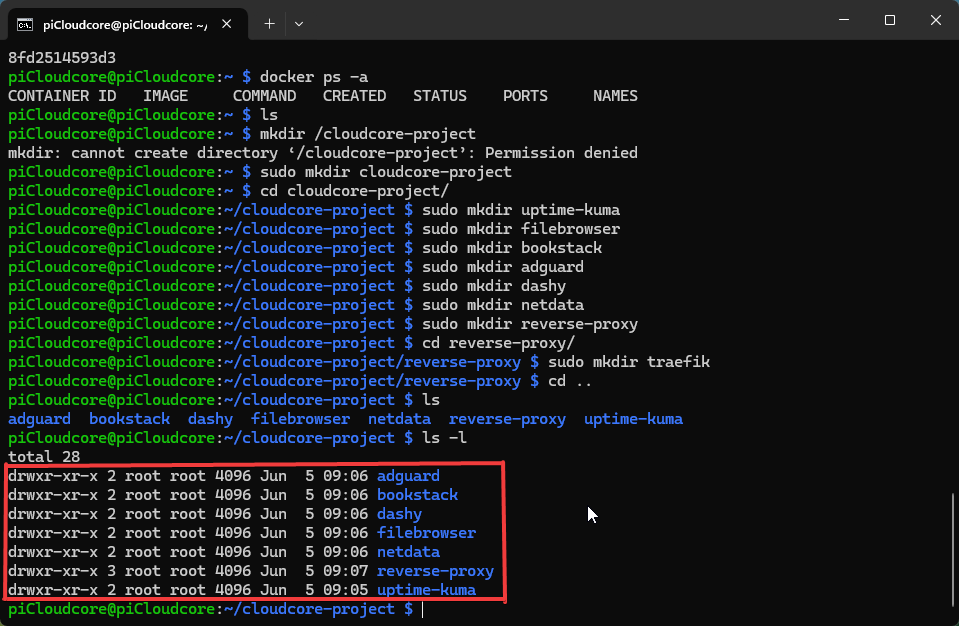


## Selfhosted Platform Basisk

Ein Bild, das Symbol, Logo, Grafiken, Kreis enthält.

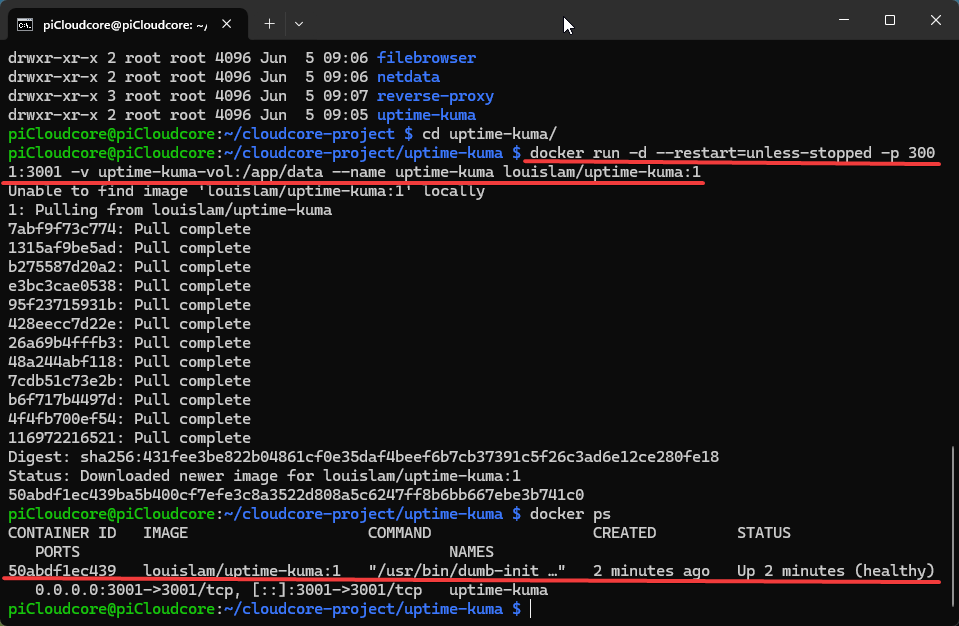
Automatisch generierte Beschreibung

### Projektverzeichniss

Es ist sinnvoll ein Projekt Verzeichniss zu erstellen wie hier. Als Main Directory cloudcore-project und darunter kommen die ganzen unterordner pro dienst mit den Docker yaml Files drin. So kann ich die Dienste einfach und übersichtlich mit Docker Compose up -d starten.

### Uptime Kuma Contianer

Um den Kuma Container zu erstellen benutze ich den command vom [Kuma Wiki](https://github.com/louislam/uptime-kuma/wiki/%F0%9F%94%A7-How-to-Install):

docker run -d --restart=unless-stopped -p <YOUR\_PORT>:3001 -v <YOUR\_DIR OR VOLUME>:/app/data --name uptime-kuma louislam/uptime-kuma:1

Das zieht das Image, erstellt das Volume und startet den Container erfolgreich.

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

Diesen command kann man dan auch noch in ein yaml reinschreiben um beim starten des containers nur noch docker compose up benutzen muss.

### Admin-Konto für Uptime Kuma einrichten

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte BeschreibungBeim ersten Starten öffnet sich ein Login Bildschirm nach dem Verbinden auf <http://10.80.4.210:3001> da wird das Login festgelegt für den Zugang auf das Dashboard.

Benutzername: piCloudecore

Passwort: Zli12345

### Nginx-Proxy konfigurieren

Gehe und erstelle das verzeichniss reverse-proxy/nginx/ Darin muss die Datei: [kuma.conf](https://github.com/louislam/uptime-kuma/wiki/Reverse-Proxy) angelegt werden:

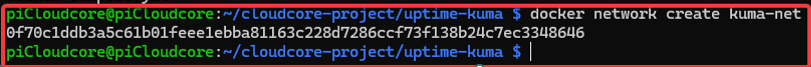
Ein Bild, das Text, Screenshot, Schrift enthält.

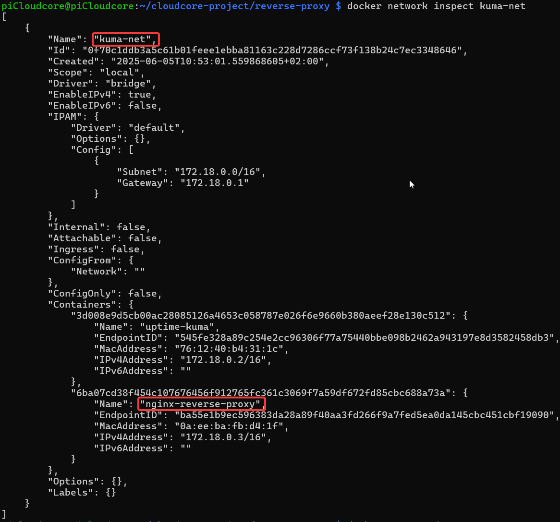
Automatisch generierte Beschreibung

Durch diese Datei weis der Proxy dann was er machen muss wen auf den Port 80 connected wird und ich auf den Kuma möchte.

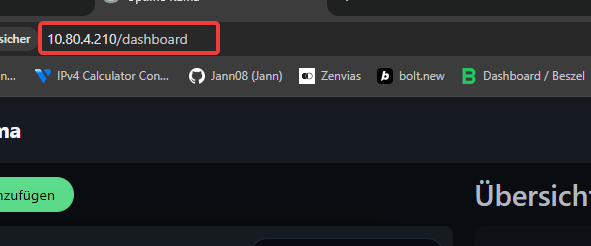
### Gemeinsames Docker-Netzwerk erstellen

Wir benötigen ein Gemeinsames Netzwerk das sich der Nginx und der Kuma sich im Netzwerk sehen können.



Die beiden Container kommunizieren jetzt direkt über diese Namen: uptime-kuma & nginx-reverse-proxy, Dass kann man auch nochmals mit dem Befehl docker network inspect kuma-net überprüfen:

Jetzt kann man nur noch die IP eingeben und man kommt auf das richtige Dashboard da die anfrage dank des Proxys weitergeleitet wird.



Ein Bild, das Symbol, Logo, Grafiken, Kreis enthält.

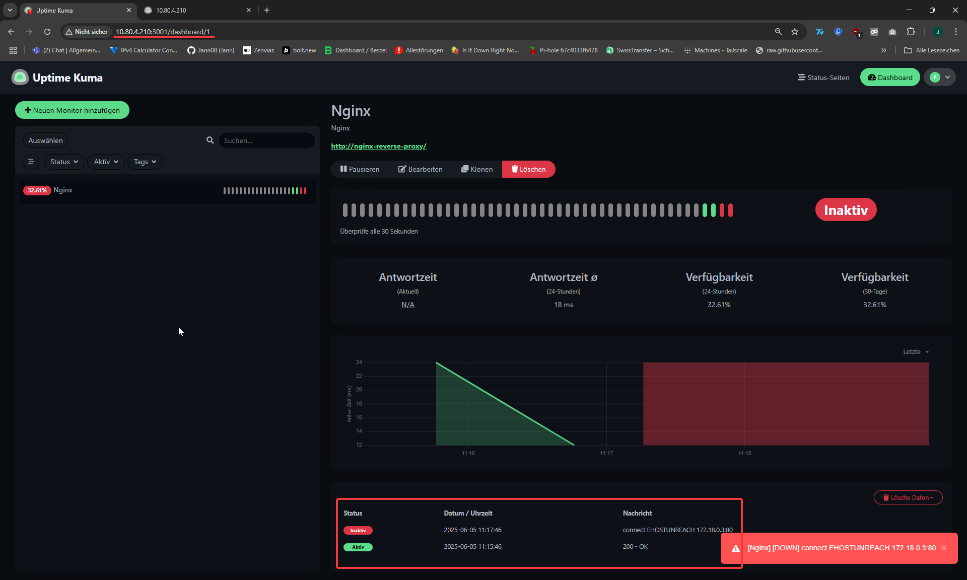
Automatisch generierte Beschreibung

### Monitoring Test

Um zu sehen ob das ganze richtig kommuniziert kann man Testweise ein Monitor hinzufügen und das habe ich hier gemacht und den Nginx container hinzugefügt.

Ein Bild, das Screenshot, Text, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

Im Monitoring kann ich sehen, ob der Service aktiv oder inaktiv ist:

## Filebrowser + Uptime Kuma hinter Nginx

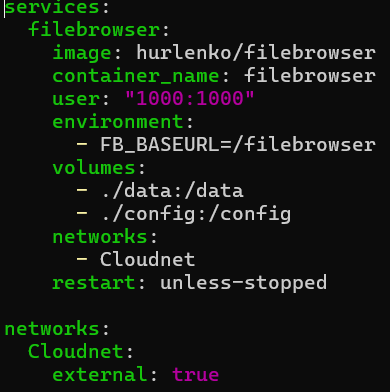
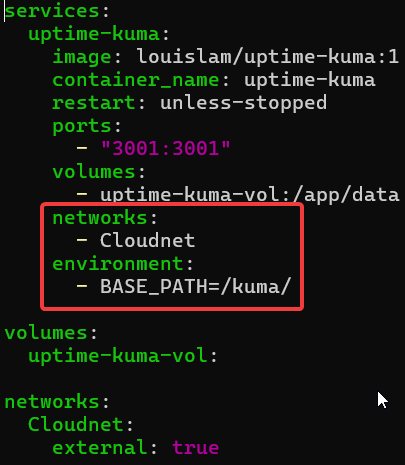
Um den Filebrowser zu installieren kann man dieses [Github](https://github.com/hurlenko/filebrowser-docker) benuzen und daraus das Docker compose file und die config datei für den Nginx Server grösstenteils übernehmen.

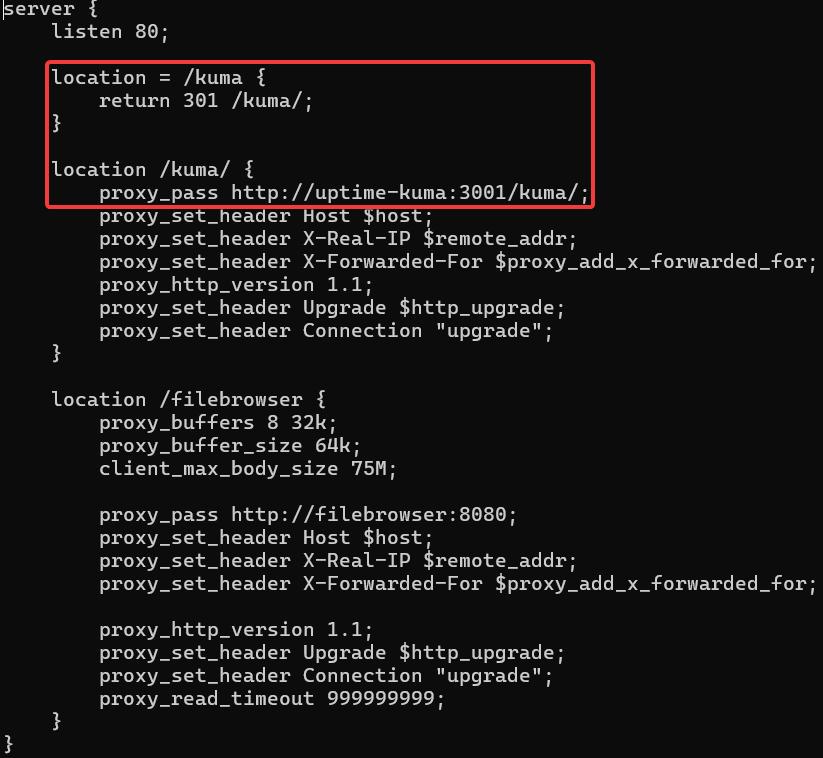
### Gemeinsames Netzwerk Cloudnet

Alle Container wurden in das manuell erstellte Netzwerk Cloudnet eingebunden:

**docker network create --attachable --driver bridge Cloudnet**

### Docker Compose files

Uptime Kuma Monitoring File browser

 Config Datei für reverse proxy über Nginx

## Erwartetes Ergebniss:

Theoretisch sollte ich mit diesen einstellungen über diese zwei Adressen:

http://10.80.4.210/kuma → Uptime Kuma Dashboard

http://10.80.4.210/filebrowser → Filebrowser Webinterface

Auf die richtigen dienste weitergeleitet werden jedoch gibt es Probleme beim Kuma.

### Problem

Uptime-Kuma lädt im Nginx-Reverse-Proxy unter /kuma nicht korrekt.

### Ausgangslage:

Uptime-Kuma läuft in einem eigenen Docker-Container hinter Nginx

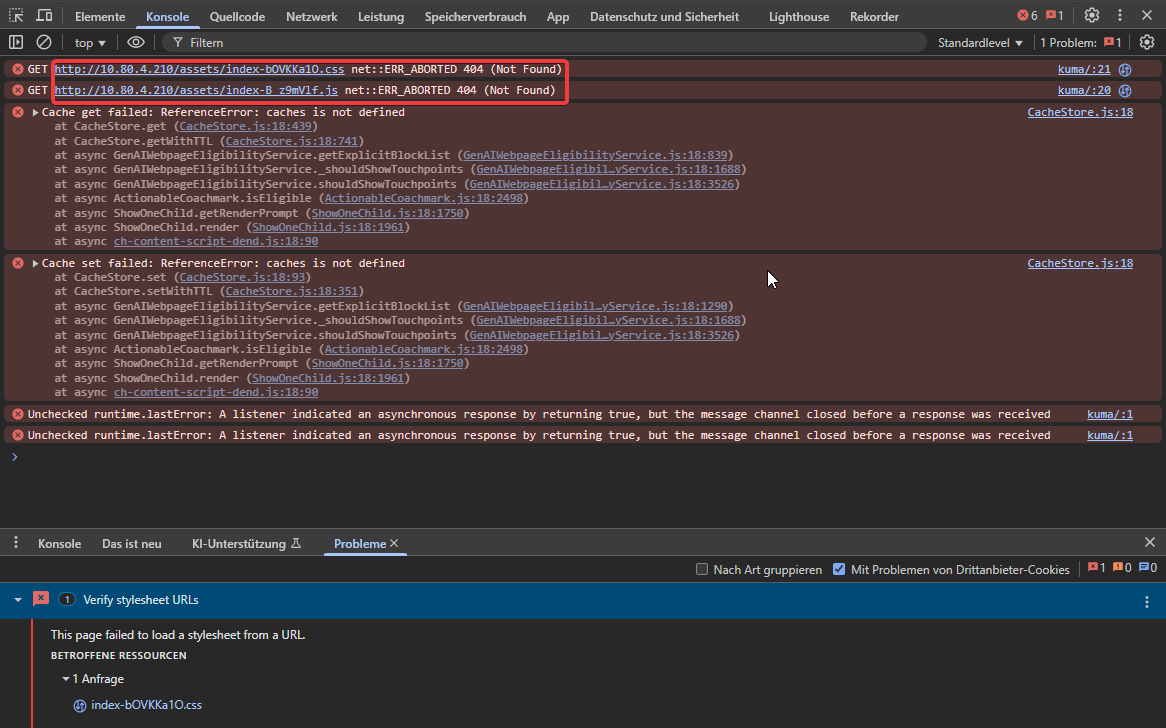
Zugriff erfolgt über: http://10.80.4.210/kuma

Container wurde korrekt mit BASE\_PATH=/kuma gestartet so das der Nginx weis wohin er die Assets und Files weiterleiten muss oder her holen muss.

basePath wurde zusätzlich direkt in der SQLite-DB (kuma.db) als basePath = /kuma gesetzt sodas es ganz sicher als basePath benutzt wird.

Ich habe das alte Volumen Gelöscht, das aus diesem mögliche Konfigurationen weg sind, und ein neues Volumen angelegt.

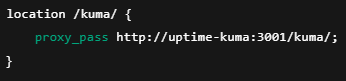
### Fehler:

Beim Aufrufen der webseite <http://10.80.4.210/kuma> wird nur ein weisser bildschirm angezeigt und in den Error logs ist das zu sehen:

Das bedeutet das die assets ohne den /kuma pfad geladen werden.

Die App versucht auf <http://10.80.4.210/assets/> die assets zu finden diese sind jedoch umter <http://10.80.4.210/kuma/assets/> vorhanden.

### Ursache

Nginx-Proxy verwendet:

basePath ist korrekt gespeichert, aber:

Uptime-Kuma rendert weiterhin Hardcoded-Paths ohne Präfix.

Dieser Fehler tritt auf, wenn:

Der erste Aufruf **nicht** über /kuma/ erfolgt

Browser-Cache alte Pfade enthält

Kuma initialisiert seine config.json bzw. Frontend-State nicht korrekt nach BasePath-Änderung

Ein Bild, das Symbol, Logo, Grafiken, Kreis enthält.

Automatisch generierte Beschreibung

### Workaround-Versuche/ Debugging mit Chat GPT

manuelles Setzen von basePath in kuma.db

Restart des Containers

Neu anlegen vom Volume

Aufruf nur über /kuma/

Nginx-Fix mit location = /kuma { return 301 /kuma/; }

Alles hat aber nichts gebracht, Uptime-Kuma kann probleme machen mit Reverse-Proxies und BASE\_PATH. Diese Probleme betreffen insbesondere statische Assets, da sie nicht dynamisch relativ zum basePath geladen werden.