Docker Container

- Installation Ollama und OpenWebUI
- Installation Paperless und Paperless-AI
- Installation Immich-App

Installation Ollama und OpenWebUI

1. Verzeichnisstruktur vorbereiten

Alle Containerdaten liegen unter `/opt/docker`, damit System und Daten sauber getrennt bleiben:

```
sudo mkdir -p /opt/docker/ollama
sudo mkdir -p /opt/docker/open-webui
```

Optional Zugriffsrechte setzen:

```
sudo chown -R kiadmin:docker /opt/docker
```

---

2. Docker-Compose Datei

Pfad: `/opt/docker/docker-compose.yml`

```
version: '3.8'

services:
  ollama:
    image: ollama/ollama
    container_name: ollama
    ports:
      - "11434:11434"
    restart: always
    runtime: nvidia
    volumes:
      - /opt/docker/ollama:/root/.ollama

  openwebui:
    image: ghcr.io/open-webui/open-webui:ollama
    container_name: open-webui
    restart: always
    runtime: nvidia
    ports:
      - "3000:8080"
    environment:
```

```
      - OLLAMA_API_BASE_URL=http://ollama:11434
    volumes:
      - /opt/docker/open-webui:/app/backend/data
      # Eigene CSS-Anpassungen (optional)
      # - /home/pleibling/docker/ai/custom.css:/app/build/static/custom.css:ro
    depends_on:
      - ollama
```

---

3. Container starten

```
cd /opt/docker
docker compose up -d
```

Status prüfen:

```
docker ps
```

- Ollama-API → `http://<Server-IP>:11434`
- OpenWebUI → `http://<Server-IP>:3000`

---

4. Modelle installieren

Ollama lädt Modelle nach Bedarf.
Installation erfolgt z. B. über:

```
docker exec -it ollama ollama pull gpt-oss:20b
```

Verwendete LLMs in dieser Umgebung:

- **GPT-OSS:20b** → Hauptmodell (groß, sehr leistungsfähig)
- **Llama 3.1:12b**
- **Mistral:7b**
- **Gemma3:12b**
- **DeepSeek-R1:8b**
- **Qwen3:14b**

Optional kannst du auch in OpenWebUI für jedes Modell eigene Presets/Workspaces definieren.

---

5. Update der Container

```
cd /opt/docker
docker compose pull
docker compose up -d
```

---

6. Backup-Hinweis

Da alle persistenten Daten in `/opt/docker/ollama` und `/opt/docker/open-webui` liegen, reicht ein Backup dieser Ordner.

Installation Paperless und Paperless-AI

Verzeichnis anlegen (einmalig)

```
sudo mkdir -p /home/pleibling/docker/paperless/{data,media,export,consume,db,redis,ai}
sudo chown -R 1000:1000 /home/pleibling/docker/paperless
```

`docker-compose.yml` (unter `/home/pleibling/docker/paperless/docker-compose.yml`)

```
version: "3.8"

services:
  paperless-ngx:
    image: ghcr.io/paperless-ngx/paperless-ngx:latest
    container_name: paperless-ngx
    restart: always
    environment:
      - PAPERLESS_REDIS=redis://paperless-redis:6379
      - PAPERLESS_DBHOST=paperless-db
      - PAPERLESS_DBUSER=paperless
      - PAPERLESS_DBPASS=PapPW050725
      - PAPERLESS_SECRET_KEY=G2v3eKLZRIkpMeUcGkLor0Lt6vtzHodKLCRVvYHHjtE=
      - PAPERLESS_AI_ENABLED=1
      - PAPERLESS_AI_PROVIDER=ollama
      - PAPERLESS_AI_MODEL=llama3.1:8b
      - PAPERLESS_AI_HOST=http://192.168.33.200:11434
      - PAPERLESS_ALLOWED_HOSTS=dms.leibling.de,192.168.33.200,localhost
      -
PAPERLESS_CSRF_TRUSTED_ORIGINS=https://dms.leibling.de,http://192.168.33.200:8080,http://local
host:8080
      - USERMAP_UID=1000
      - USERMAP_GID=1000
```

```
      - PAPERLESS_TIKA_ENABLED=true
      - PAPERLESS_TIKA_ENDPOINT=http://tika:9998
      - PAPERLESS_CONVERT_OFFICE=true
      - PAPERLESS_TIKA_CONVERT_OFFICE=true
      - PAPERLESS_GOTENBERG_ENABLED=true
      - PAPERLESS_GOTENBERG_ENDPOINT=http://gotenberg:3000
      - RAG_SERVICE_ENABLED=true
      - RAG_SERVICE_URL=http://paperless-ai:8000
    ports:
      - "8080:8000"
    volumes:
      - /opt/docker/paperless/data:/usr/src/paperless/data
      - /opt/docker/paperless/media:/usr/src/paperless/media
      - /opt/docker/paperless/export:/usr/src/paperless/export
      - /opt/docker/paperless/consume:/usr/src/paperless/consume
    depends_on:
      - paperless-db
      - paperless-redis
  gotenberg:
    image: gotenberg/gotenberg:7
    container_name: paperless-gotenberg
    restart: always
    ports:
      - "3002:3000"
    security_opt:
      - no-new-privileges:true
    cap_drop:
      - ALL

  tika:
    image: apache/tika
    container_name: paperless-tika
    restart: always
    ports:
      - "3003:9998"
    security_opt:
      - no-new-privileges:true
    cap_drop:
      - ALL
```

```
  paperless-db:
    image: postgres:15
    container_name: paperless-db
    restart: always
    environment:
      - POSTGRES_DB=paperless
      - POSTGRES_USER=paperless
      - POSTGRES_PASSWORD=PapPW050725
    volumes:
      - /opt/docker/paperless/db:/var/lib/postgresql/data

  paperless-redis:
    image: redis:7
    container_name: paperless-redis
    restart: always
    volumes:
      - /opt/docker/paperless/redis:/data

  paperless-ai:
    image: clusterzx/paperless-ai
    container_name: paperless-ai
    restart: always
    environment:
      - PUID=1000
      - PGID=1000
      - PAPERLESS_AI_PORT=3000
      - RAG_SERVICE_ENABLED=true
      - RAG_SERVICE_URL=http://paperless-ai:3000
    ports:
      - "3010:3000"
    volumes:
      - /opt/docker/paperless/ai:/app/data
    deploy:
      resources:
        reservations:
          devices:
            - capabilities: ["gpu"]
```

Wichtig:

- Deine **Ollama-Instanz** läuft in einem anderen Stack (Ollama/OpenWebUI). Zugriff erfolgt hier **per IP/Port** (`http://192.168.33.200:11434`) – das ist ideal, da unterschiedliche Compose-Netzwerke sich sonst nicht automatisch sehen.
- Wenn du später **TLS/Reverse Proxy** (z. B. Traefik/Caddy/Nginx) nutzt, passe `PAPERLESS_ALLOWED_HOSTS` und `PAPERLESS_CSRF_TRUSTED_ORIGINS` an.

Start/Update

```
cd /home/pleibling/docker/paperless
docker compose up -d
# Update:
docker compose pull && docker compose up -d
```

Kurzer Funktionstest

- Paperless-NGX: `http://<VM-IP>:8080`
- RAG-API (intern gemappt): `http://<VM-IP>:3010/health` → sollte `ok` liefern
- Tika: `http://<VM-IP>:3003/version`
- Gotenberg: `http://<VM-IP>:3002/health`

Installation Immich-App

Immich – Installation (Minimal Setup)

```
cd /opt/docker
git clone https://github.com/immich-app/immich.git immich-app
cd immich-app
```

`.env` erstellen

Datei: `/opt/docker/immich-app/.env`

```
# You can find documentation for all the supported env variables at
https://immich.app/docs/install/environment-variables

# The location where your uploaded files are stored
UPLOAD_LOCATION=./library

# The location where your database files are stored. Network shares are not supported for the
database
DB_DATA_LOCATION=./postgres

# To set a timezone, uncomment the next line and change Etc/UTC to a TZ identifier from this
list: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones#List
TZ=Europe/Berlin

# The Immich version to use. You can pin this to a specific version like "v1.71.0"
IMMICH_VERSION=release

# Connection secret for postgres. You should change it to a random password
# Please use only the characters `A-Za-z0-9`, without special characters or spaces
DB_PASSWORD=kdlIHdjdfifhj7f7ehekdhuvzdjenfifhHJZHGjdu65w7sh
```

```
# The values below this line do not need to be changed
###################################################################################
DB_USERNAME=postgres
DB_DATABASE_NAME=immich
```

Starten

```
docker compose up -d
```

Aufruf im Browser

`http://<VM-IP>:2283`

Erster registrierter Benutzer = Administrator.
