

Docker Container

- [Dockge](#)
- [Zammad](#)
- [Portainer](#)
- [UniFi Network Application](#)

Dockge

How to Install Requirements:

Docker 20+ / Podman (Podman only) podman-docker (Debian: apt install podman-docker) OS:
Major Linux distros that can run Docker/Podman such as: ☐ Ubuntu ☐ Debian (Bullseye or newer) ☐
Raspbian (Bullseye or newer) ☐ CentOS ☐ Fedora ☐ ArchLinux ☐ Debian/Raspbian Buster or lower is not
supported ☐ Windows (Will be supported later) Arch: armv7, arm64, amd64 (a.k.a x86_64) Basic
Default Stacks Directory: /opt/stacks Default Port: 5001

Create directories that store your stacks and stores Dockge's stack

```
mkdir -p /opt/stacks /opt/dockge cd /opt/dockge
```

Download the compose.yaml

```
curl https://raw.githubusercontent.com/louislam/dockge/master/compose.yaml --output  
compose.yaml
```

Start the server

```
docker compose up -d
```

If you are using docker- compose V1 or Podman docker-compose up -d

Dockge is now running on <http://localhost:5001>

Advanced If you want to store your stacks in another directory, you can generate your compose.yaml file by using the following URL with custom query strings.

Download your compose.yaml

`curl "https://dockge.kuma.pet/compose.yaml?port=5001&stacksPath=/opt/stacks" --output
compose.yaml port=5001 stacksPath=/opt/stacks` Interactive compose.yaml generator is available
on: <https://dockge.kuma.pet>

How to Update `cd /opt/dockge docker compose pull && docker compose up -d`

Zammad

dsdfdsf

sdfsddfsdfsdfs

Systemvoraussetzungen

- Docker Compose muss auf dem System installiert werden
- 4GB RAM
- Systemeinstellungen müssen wie folgt angepasst werden

```
sysctl -w vm.max_map_count=262144
```

Installation mit Docker Compose

GitHub Repo klonen

```
git clone https://github.com/zammad/zammad-docker-compose.git
```

Adjust Environment as Needed

In some cases our default environment is not what a docker-compose user is looking for. See [Docker Environment Variables](#) for details on which settings can be configured.

If you want to use a `.env` file, you can use the provided `.env.dist` file and copy it to `.env`. That way it will be picked up by Docker-Compose automatically and not overwritten during updates.

Zammad runs on port `8080` by default. If you want to use another port, you can set it via the variable `NGINX_EXPOSE_PORT`.

Portainer

Portainer Community Edition (CE)

Installation

```
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always -v  
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce
```

Update

1. Stop Docker Container

```
docker stop portainer
```

2. Remove Docker Container

```
docker rm portainer
```

3. Update Docker Image

```
docker pull portainer/portainer-ce
```

UniFi Network Application

docker-compose.yaml

```
services:
  # Define the services to run
  db:
    image: mongo
    volumes:
      - ./db:/data/db # MongoDB data persistence
      - ./db/init-mongo.js:/docker-entrypoint-initdb.d/init-mongo.js:ro
    ports:
      - 27017:27017 # MongoDB default port
    restart: always # Restart policy
    networks:
      backend: null
  app:
    image: lscr.io/linuxserver/unifi-network-application # Docker image to use
    environment:
      # Environmental variables for the container
      - PUID=1000 # User ID
      - PGID=1000 # Group ID
      - TZ=Europe/Zurich # Timezone
      - MONGO_HOST=db # MongoDB host
      - MONGO_USER= ${DB_USER} # MongoDB username
      - MONGO_PASS= ${DB_PW} # MongoDB password
      - MONGO_PORT=27017 # MongoDB port
      - MONGO_DBNAME= ${DB_NAME} # MongoDB database name
      - MEM_LIMIT=1024 #optional # Memory limit for the container
      - MEM_STARTUP=1024 #optional # Memory to allocate on container startup
    volumes:
      # Volumes to mount in the container
      - ./web:/config # Map host directory to container directory
    ports:
      # Ports to expose and forward
      - 8443:8443 # HTTPS portal
      - 3478:3478/udp # STUN service
```

- 10001:10001/udp # UniFi AP discovery
- 8080:8080 # HTTP portal
- 6789:6789 #optional # Mobile speed test port
- 161:161/udp #SNMP

restart: always # Restart policy for the container

depends_on:

- db

networks:

frontend: null

backend: null

networks:

frontend: null

backend: null

.env

DB_USER=unifi

DB_PW=password

DB_NAME=unifi_db

init-mongo.js

```
db.getSiblingDB("unifi_db").createUser({user: "unifi", pwd: "password", roles: [{role: "dbOwner", db: "unifi_db"}]});
```

```
db.getSiblingDB("unifi_db_stat").createUser({user: "unifi", pwd: "password", roles: [{role: "dbOwner", db: "unifi_db_stat"}]});
```