1.6.1 Docker VM Preperation

1.6.1.1 Docker VM Installation

The first VM we will create is the Docker VM. This VM will serve for all our docker compose files and applications that are not mission critical and are light weight applications. What is and what is not mission critical is of course for up to debate ;) Personally I have three dedicated VM's

1. One for Ente Photos
2. One for my Servarr Stack
3. My Docker VM for all my lightweight applications

In this section we will setup a Docker VM in proxmox i.e most likely your first VM. Luckily this will be extremely ease due to Proxmox Helper scripts. Proxmox Helper scripts walks you extremely easy through the whole installation process of any LXC container and also certain VM's. In this case we will make use of it as well to create our Docker VM. Super easy, no configuration necessary from your end. Just hit the following link in your PVE Shell and you can start

```
bash -c "$(curl -fsSL https://raw.githubusercontent.com/community-scripts/ProxmoxVE/main/vm/docker-vm.sh)"
```

1. Choose Install Docker Vm

2. Choose Advanced Setup

3. Select the newest form of Debian as of writing that is Debian 13

4. Enable Cloud Init: Yes

5. Provision SSH keys for Cloud-Init VM: -> Choose "manual Paste a single public key "

6. Copy paste in your SSH key from the machine you are planning to access the Docker VM from.

7. Set a Docker VM ID: Choose what fits for you. Most often I start with 1001

8. Choose type: q35

9. Choose the amount of Storage: I would recommend to start with 16G and then you can always add more over time. It is easy to add but hard to remove storage from your VM. Hence, start low and add more if necessary.

10.Disk Cache: None

11. Choose hostname: hostname

12. Choose CPU model: host

13. Allocate CPU cores: Start with 4 and you can always add more if necessary

14. RAM: start with 4096

15. Interface: Choose the interface on which the VLAN sits that you want to use for your docker VM.

16. Set a MAC address: Leave as is

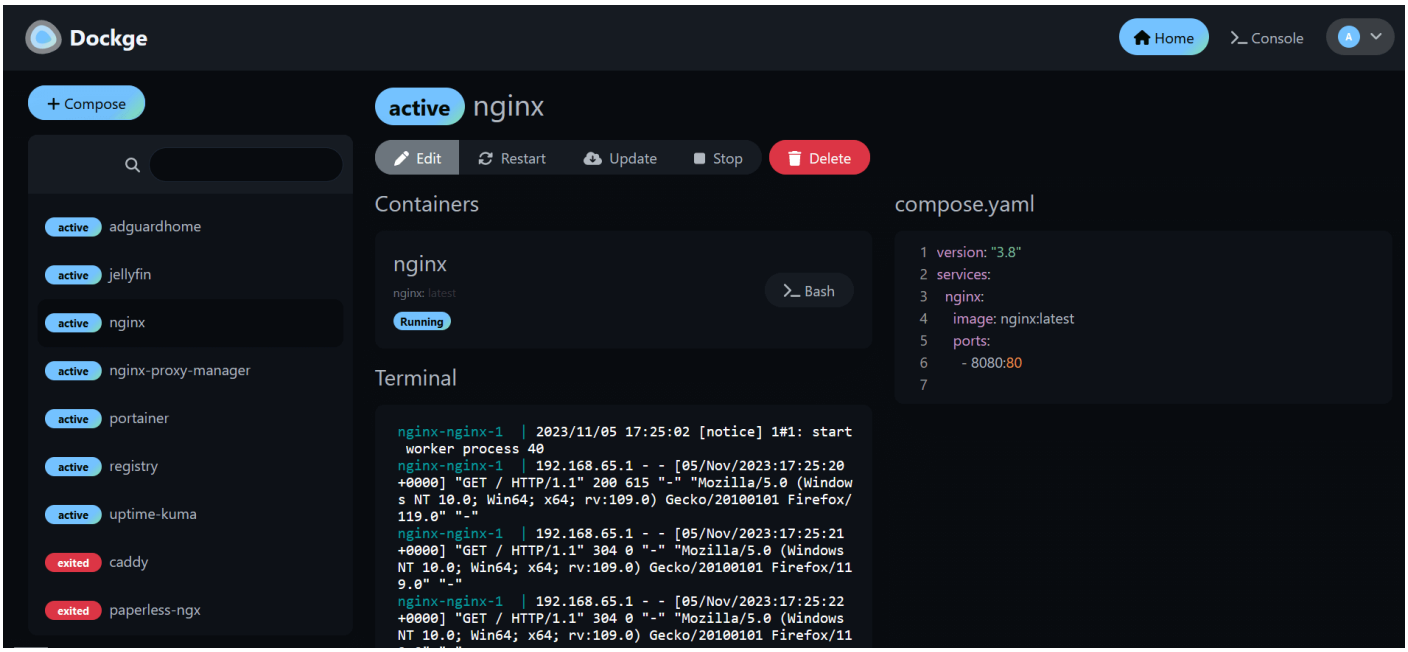
17. Set VLAN: The VLAN you want to host the docker VM on.

18. Finally choose the storage pool: Recommended on your VM mirrored SSD ZFS pool if you have that configured

1.6.1.2 Dockge Introduction

As we will see we can either be in the terminal all the time editing docker compose files for our applications or we can attach a GUI to it.

1. Website
2. Github Page



So yeahh that is basically what Dockge is including many other features such as

- □□□ Manage your `compose.yaml` files
  - Create/Edit/Start/Stop/Restart/Delete
  - Update Docker Images
- ⌨ Interactive Editor for `compose.yaml`
- □□ Interactive Web Terminal
- □□ (1.4.0 □□) Multiple agents support - You can manage multiple stacks from different Docker hosts in one single interface
- □□ Convert `docker run ...` commands into `compose.yaml`
- □□ File based structure - Dockge won't kidnap your compose files, they are stored on your drive as usual. You can interact with them using normal `docker compose` commands
- □□ Reactive - Everything is just responsive. Progress (Pull/Up/Down) and terminal output are in real-time
- □□ Easy-to-use & fancy UI - If you love Uptime Kuma's UI/UX, you will love this one too

So in this Book we will also make sure to use Dockge for all our installations. It is for sure not necessary but it is so easy and comfortable. In the next page we will do the installation.

1.6.1.3 Dockge Installation

Assuming you already have installed your Docker VM it is time to install Dockge on top of it to handle all your docker compose files ;)

1. Go to the official dockge page and basically copy the docker compose file that looks like this

```
services:
  dockge:
    image: louislam/dockge:1
    restart: unless-stopped
    ports:
      - 5001:5001
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - ./data:/app/data
      # Stacks Directory
      # ⚠ READ IT CAREFULLY. If you did it wrong, your data could end up writing into a WRONG
PATH.
      # ⚠ 1. FULL path only. No relative path (MUST)
      # ⚠ 2. Left Stacks Path === Right Stacks Path (MUST)
      - /opt/stacks:/opt/stacks
    environment:
      # Tell Dockge where to find the stacks
      - DOCKGE_STACKS_DIR=/opt/stacks
      # This is great for entering the Docker VM console from the WebGUI
      - DOCKGE_ENABLE_CONSOLE=true
```

2. Then open your terminal with your access machine and SSH into it. You added the SSH key of your machine in the Cloud Init section so you should be entering without any interruptions. Simply use

> ssh username@IP-Address-Docker-VM

3. Once inside we will create a new dockge folder with

```
mkdir docgke
```

4. Then we will enter the folder with

```
cd dockge
```

5. Inside this folder we will create the docker compose file by copying and pasting in the docker compose content from above

```
nano compose.yml
```

6. Then save it with

CTRL X and hit Y

7. Then we will start the dockge container with

```
docker compose up -d
```

8. If everything is green and alright we should be able to access the WebGui at

> IP-Address-DockerVM:5001

9. Then create an account with a username and password. Now, normally I would say you would save the credentials in your Bitwarden account but we have not yet setup that container. However, choose a non typical username and a password of 20 characters and copy paste it somewhere safe s.t you can find it. After we have created a vaultwarden container you will be able to copy paste it inside of it with all your other future credentials.