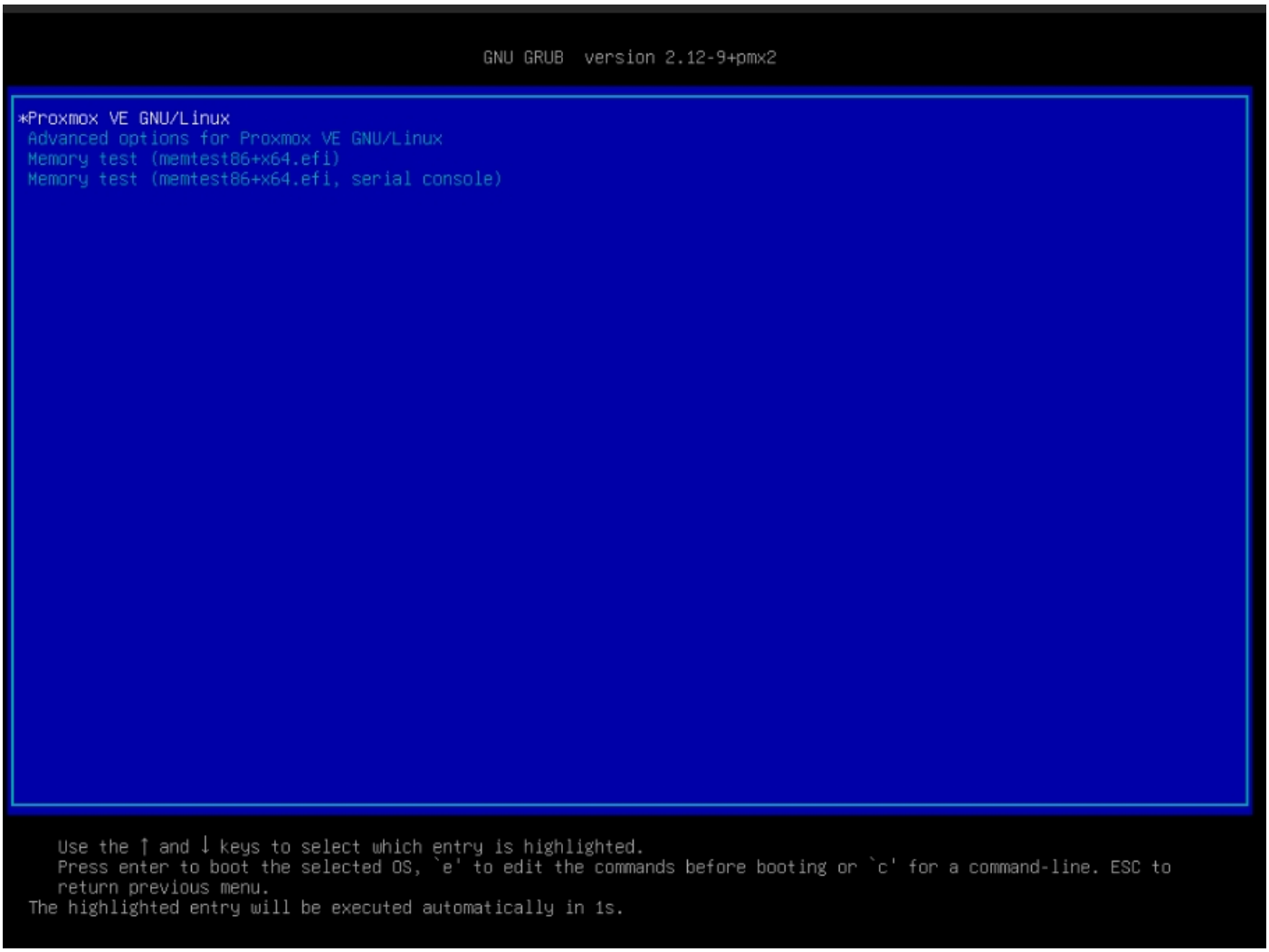
Proxmox + CT ollama + VM ubuntu docker OpenwebUI



After proxmov deployment

4 lspci | grep NVIDIA

5 lsmod | grep nouveau ### nouveau 要禁用

6 cd /

7 ls -l /dev/dri

8 ls -l /dev/nvidia*

9 echo -e "blacklist nouveau\noptions nouveau modeset=0" > /etc/modprobe.d/blacklist-nouveau.conf && update-initramfs -u && reboot

10 cat /etc/modprobe.d/blacklist-nouveau.conf

root@pve:~# cat /etc/modprobe.d/blacklist-nouveau.conf

blacklist nouveau

options nouveau modeset=0

11 lsmod | grep nouveau

12 lsmod | grep nouveau

13 lspci | grep NVIDIA

wget https://us.download.nvidia.com/XFree86/Linux-x86_64/580.95.05/NVIDIA-Linux-x86_64-580.95.05.run

121 chmod +x NVIDIA-Linux-x86_64-580.95.05.run

123 ./NVIDIA-Linux-x86_64-580.95.05.run

124 reboot

Nvidia-smi verify driver version

###PVE 完成

create CT

61 pveam download local ubuntu-24.04-standard_24.04-2_amd64.tar.zst

Create CT101 ollama

root@pve:~# cat /etc/pve/lxc/101.conf

=== [NVIDIA GPU %E6%98%A0%E5%B0%84%E8%A8%AD%E5%AE%9A - %E8%AB%8B%E5%8B%99%E5%BF%85%E8%B2%BC%E5%9C%A8%E9%85%8D%E7%BD%AE%E6%AA%94%E6%9C%AB%E5%B0%BE] ===

%E6%AC%8A%E9%99%90%E6%8E%88%E6%AC%8A (%E4%BD%BF%E7%94%A8%E6%82%A8%E7%A2%BA%E8%AA%8D%E9%81%8E%E7%9A%84%E7%B7%A8%E8%99%9F)

%E6%98%A0%E5%B0%84 NVIDIA %E5%B0%88%E7%94%A8%E8%A3%9D%E7%BD%AE

%E6%98%A0%E5%B0%84 DRM %E6%B8%B2%E6%9F%93%E8%A3%9D%E7%BD%AE

arch: amd64

cores: 16

features: nesting=1

hostname: ollama-ct

memory: 32768

nameserver: 192.168.55.254

net0: name=eth0,bridge=vmbr0,gw=192.168.55.254,hwaddr=BC:24:11:BF:DD:D1,ip=192.168.55.181/24,type=veth

ostype: ubuntu

rootfs: local-lvm:vm-101-disk-0,size=200G

searchdomain: nx3230.local

swap: 1024

unprivileged: 1

lxc.cgroup2.devices.allow: c 195:* rwm

lxc.cgroup2.devices.allow: c 511:* rwm

lxc.cgroup2.devices.allow: c 226:* rwm

lxc.mount.entry: /dev/nvidia0 dev/nvidia0 none bind,optional,create=file

lxc.mount.entry: /dev/nvidiactl dev/nvidiactl none bind,optional,create=file

lxc.mount.entry: /dev/nvidia-uvm dev/nvidia-uvm none bind,optional,create=file

lxc.mount.entry: /dev/nvidia-uvm-tools dev/nvidia-uvm-tools none bind,optional,create=file

lxc.mount.entry: /dev/dri/card0 dev/dri/card0 none bind,optional,create=file

lxc.mount.entry: /dev/dri/card1 dev/dri/card1 none bind,optional,create=file

lxc.mount.entry: /dev/dri/renderD128 dev/dri/renderD128 none bind,optional,create=file

root@pve:~#

195:

📍 **範例：映射 NVIDIA GPU**

對於 NVIDIA GPU，LXC 容器設定通常會使用一個特殊選項來處理所有必要的節點：

```
lxc.cgroup2.devices.allow: c 195:* rwm  # 允許 NVIDIA 設備
lxc.mount.entry: /dev/nvidia-uvm dev/nvidia-uvm none bind,optional,create=file
lxc.mount.entry: /dev/nvidia0 dev/nvidia0 none bind,optional,create=file
lxc.mount.entry: /dev/nvidiactl dev/nvidiactl none bind,optional,create=file
```

或者，您可以使用 Proxmox 的 `lxc.hook` 腳本來自動處理。

511

您的下一個步驟是執行 **步驟 1: 驗證 NVIDIA 專用裝置節點**，然後開始編輯您的 CT 配置檔。

```
root@pve:~# ls -l /dev/nvidia*
crw-rw-rw- 1 root root 195,   0 Dec  2 16:09 /dev/nvidia0
crw-rw-rw- 1 root root 195, 255 Dec  2 16:09
/dev/nvidiactl
crw-rw-rw- 1 root root 511,   0 Dec  2 16:09 /dev/nvidia-...
```

太好了！您提供的輸出**確認了所有 NVIDIA 專有裝置節點**都已經在 Proxmox 主機上成功創建並準備就緒。

這些節點（尤其是 `/dev/nvidia0`、`/dev/nvidiactl`、`/dev/nvidia-uvm`）是容器 (CT) 存取 GPU 進行計算或轉碼的關鍵。

現在，我們可以精確地配置您的 CT 映射檔案。

226

```
root@pve:~#  ls -l /dev/dri
total 0
drwxr-xr-x 2 root root        100 Dec  3 13:38 by-path
crw-rw---- 1 root video  226,   0 Dec  3 13:38 card0
crw-rw---- 1 root video  226,   1 Dec  3 13:38 card1
crw-rw---- 1 root render 226, 128 Dec  3 13:38 renderD128
```

c 195:* 和 **c 511:*** 是根據您輸出的 Major Number (例如 195, 0 和 511, 0) 設置的 **NVIDIA 專有裝置**權限。

c 226:* 是針對您 /dev/dri/ 下的 **顯示渲染裝置 (DRM)** 權限。

lxc.mount.entry 負責將主機上的實體檔案路徑映射到容器內部相同路徑。

##進到 CT

86 pct start 101

87 pct enter 101

```
    5  vi /etc/apt/sources.list
root@ollama-ct:~#  cat /etc/apt/sources.list
deb http://archive.ubuntu.com/ubuntu noble main restricted universe multiverse
deb http://archive.ubuntu.com/ubuntu noble-updates main restricted universe multiverse
deb http://archive.ubuntu.com/ubuntu noble-security main restricted universe multiverse

deb http://archive.ubuntu.com/ubuntu noble main restricted universe multiverse
deb http://archive.ubuntu.com/ubuntu noble-updates main restricted universe multiverse
deb http://archive.ubuntu.com/ubuntu noble-security main restricted universe multiverse

    8  apt install libnvidia-cuda-toolkit
    9  apt install libnvidia-cuda-toolkit libnvidia-ml-dev -y

   11  apt install nvidia-utils-580 libnvidia-compute-580 libnvidia-decode-580 -y
   12  curl -fsSL https://ollama.com/install.sh | sh
   13  apt install curl
   14  curl -fsSL https://ollama.com/install.sh | sh
   15  ollama
   34  apt search libnvidia-compute | grep  580
   35  apt install nvidia-utils-580
   36  nvidia-smi       ####CT 與 PVE 版本要一到才跑得起來
   37  ollama run llama3

root@ollama-ct:/etc/systemd/system# pwd
/etc/systemd/system
root@ollama-ct:/etc/systemd/system# cat ollama.service
[Unit]
```

Description=Ollama Service

After=network-online.target

[Service]

ExecStart=/usr/bin/ollama serve

User=ollama

Group=ollama

Restart=always

RestartSec=3

Environment="PATH=/sbin:/bin:/usr/sbin:/usr/bin"

Environment="OLLAMA_HOST=0.0.0.0" ####這個後面或現在要改 沒有加只有 127.0.0.1 自己能存取 ，要改 openwebui 才拿得到

[Install]

WantedBy=default.target

root@ollama-ct:/etc/systemd/system#

74 sudo systemctl daemon-reload

75 sudo systemctl restart ollama

##CT ollama 完成

PVE CVM install Ubuntu OS

user@vm-103:~$ history

1 sudo apt update

4 sudo apt install ca-certificates curl gnupg lsb-release -y

5 sudo install -m 0755 -d /etc/apt/keyrings

6 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

7 sudo chmod a+r /etc/apt/keyrings/docker.gpg

8 echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

9 $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

10 sudo apt update

11 sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y

12 sudo docker run hello-world

13 sudo usermod -aG docker $USER

14 docker run -d --name open-webui -p 3000:8080 -e OLLAMA_BASE_URL=http://192.168.55.181:11434 -v open-webui-data:/app/backend/data --restart always ghcr.io/open-webui/open-webui:main

15 exit <<<<<<跳出重登 !13 才會生效

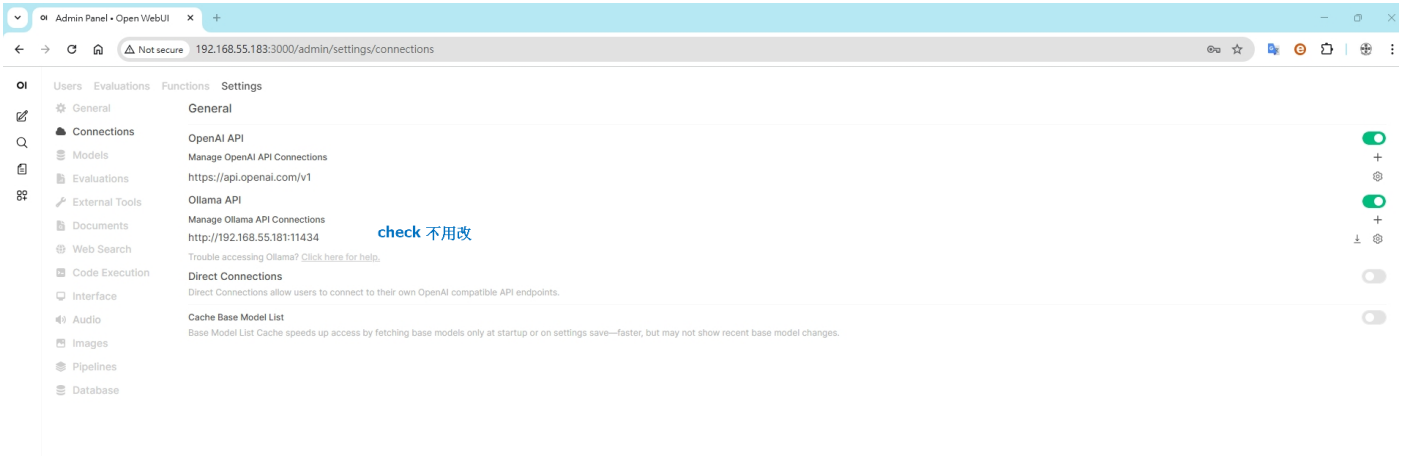
16 docker ps

17 docker run

18 docker run -d --name open-webui -p 3000:8080 -e OLLAMA_BASE_URL=http://192.168.55.181:11434 -v open-webui-data:/app/backend/data --restart always ghcr.io/open-webui/open-webui:main

19 docker ps

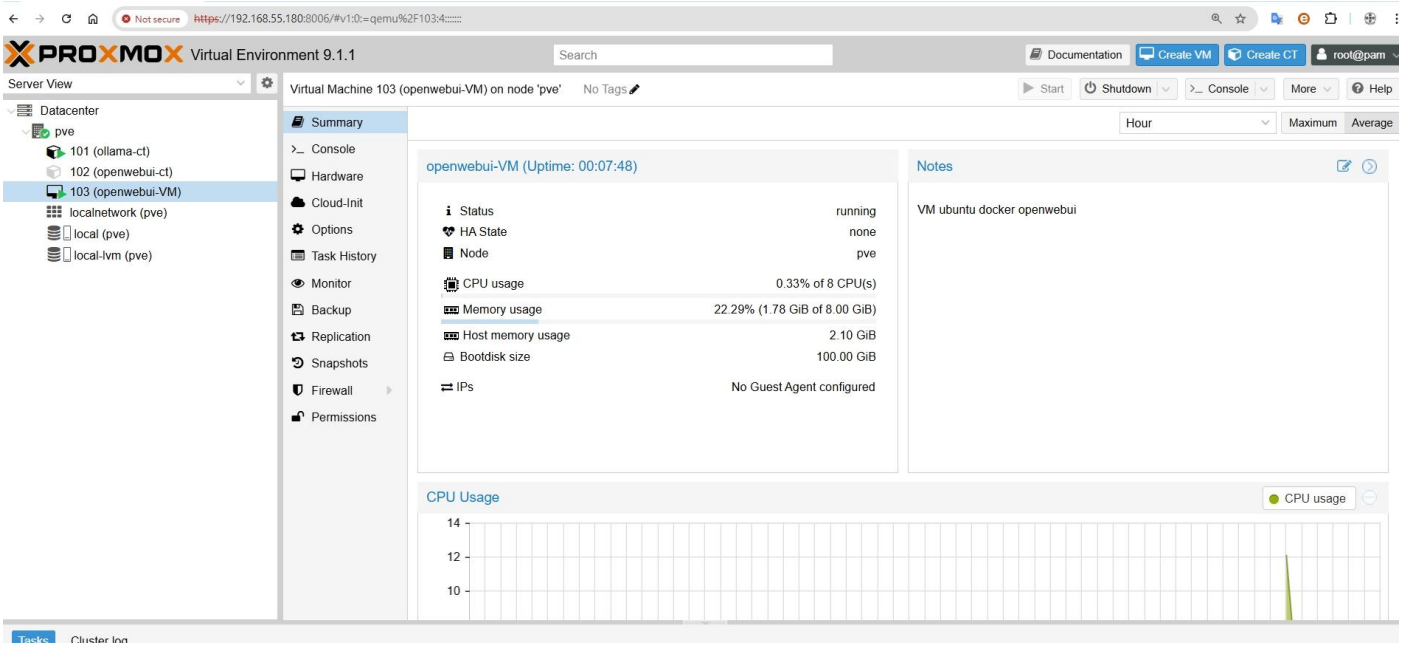
22 sudo ufw status

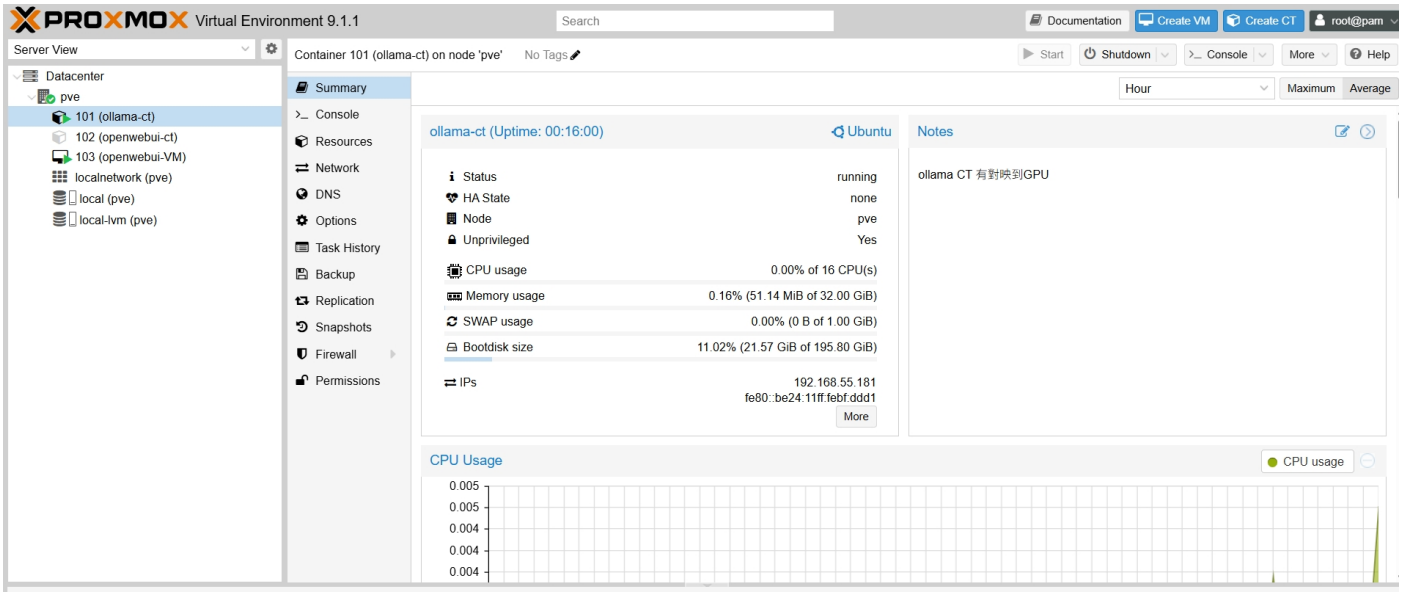


對映到 CT 已經是 Environment="OLLAMA_HOST=0.0.0.0"

如此才能確保 OpenWebUI 能連到 Ollama-IP :11434

下載 Model 的功能才接下去


Not secure https://192.168.55.180:8006/#v1:0:=qemu%2F103:4::::::
PROXMOX Virtual Environment 9.1.1
Search
Documentation
Create VM
Create CT
root@pam
Server View
Datacenter
pve
101 (ollama-ct)
102 (openwebui-ct)
103 (openwebui-VM)
localnetwork (pve)
local (pve)
local-lvm (pve)
Virtual Machine 103 (openwebui-VM) on node 'pve'
No Tags
Start
Shutdown
Console
More
Help
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Backup
Replication
Snapshots
Firewall
Permissions
Hour
Maximum
Average
openwebui-VM (Uptime: 00:07:48)
Status running
HA State none
Node pve
CPU usage 0.33% of 8 CPU(s)
Memory usage 22.29% (1.78 GiB of 8.00 GiB)
Host memory usage 2.10 GiB
Bootdisk size 100.00 GiB
IPs No Guest Agent configured
Notes
VM ubuntu docker openwebui
CPU Usage
CPU usage
14
12
10
Tasks
Cluster log


PROXMOX Virtual Environment 9.1.1
Search
Documentation
Create VM
Create CT
root@pam
Server View
Datacenter
pve
101 (ollama-ct)
102 (openwebui-ct)
103 (openwebui-VM)
localnetwork (pve)
local (pve)
local-lvm (pve)
Container 101 (ollama-ct) on node 'pve'
No Tags
Start
Shutdown
Console
More
Help
Summary
Console
Resources
Network
DNS
Options
Task History
Backup
Replication
Snapshots
Firewall
Permissions
Hour
Maximum
Average
ollama-ct (Uptime: 00:16:00)
Ubuntu
Status running
HA State none
Node pve
Unprivileged Yes
CPU usage 0.00% of 16 CPU(s)
Memory usage 0.16% (51.14 MiB of 32.00 GiB)
SWAP usage 0.00% (0 B of 1.00 GiB)
Bootdisk size 11.02% (21.57 GiB of 195.80 GiB)
IPs 192.168.55.181
fe80::be24:11ff:febf:ddd1
More
Notes
ollama CT 有對映到GPU
CPU Usage
CPU usage
0.005
0.005
0.004
0.004
0.004