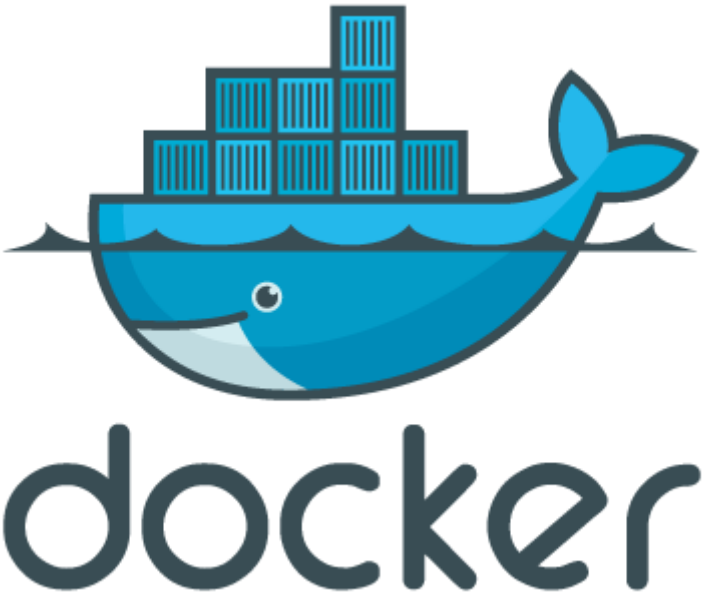
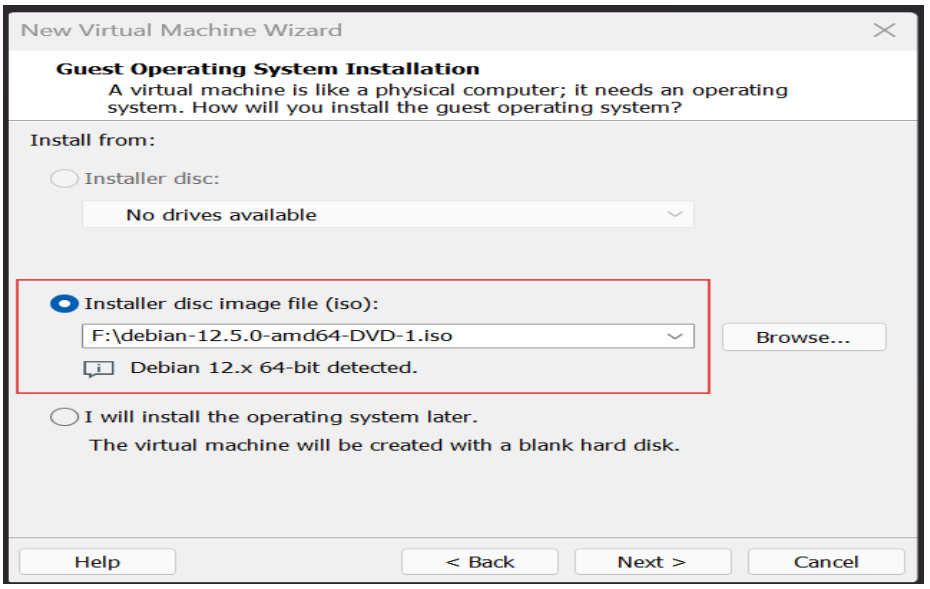
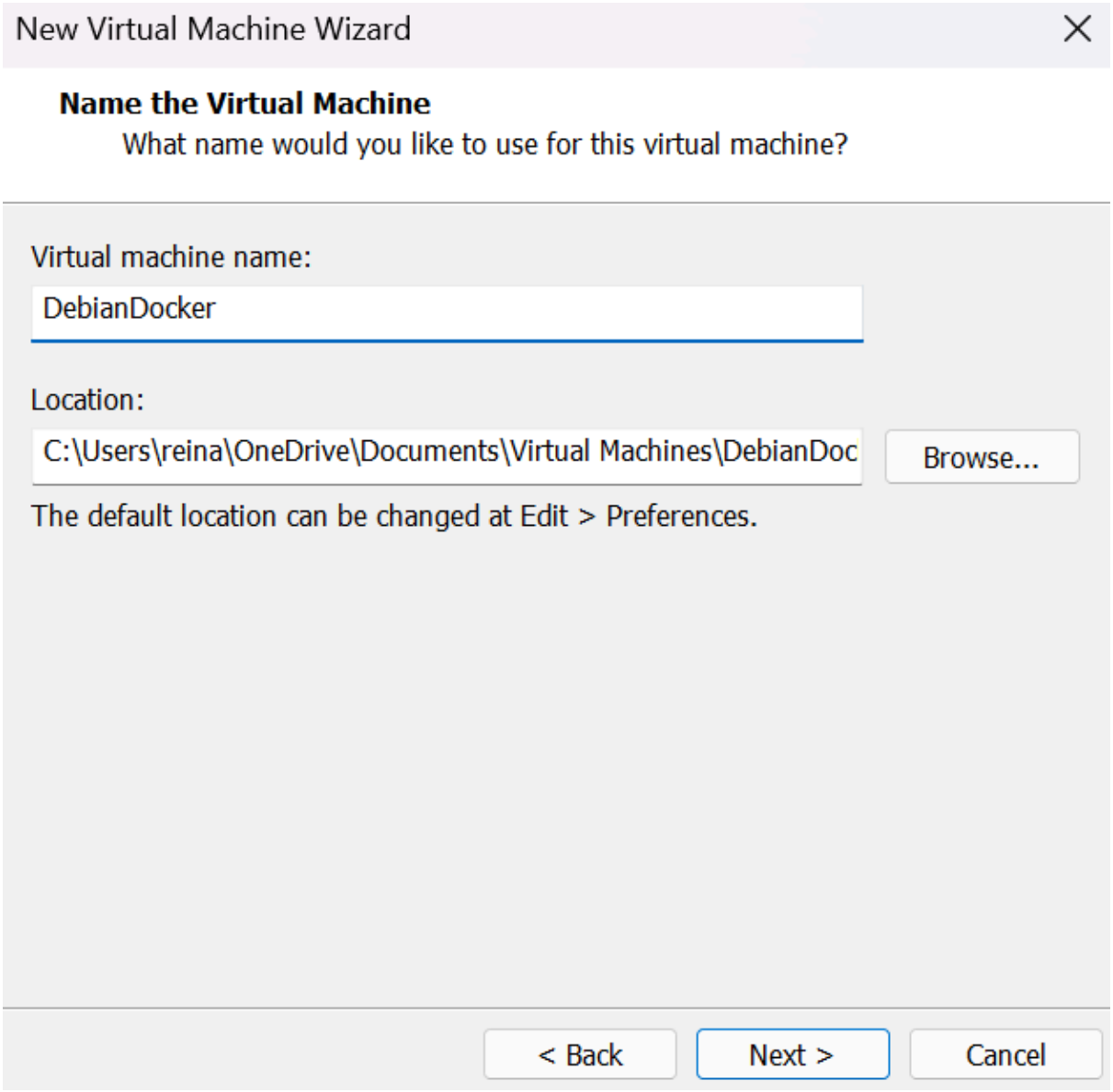


Job 01 :

Création d'une VM Debian minimaliste et y installer Docker CLI uniquement :
Une VM Debian en mode console 8 Go DD / 1 Go RAM / 1 vCPU :

Ajout de l'image ISO

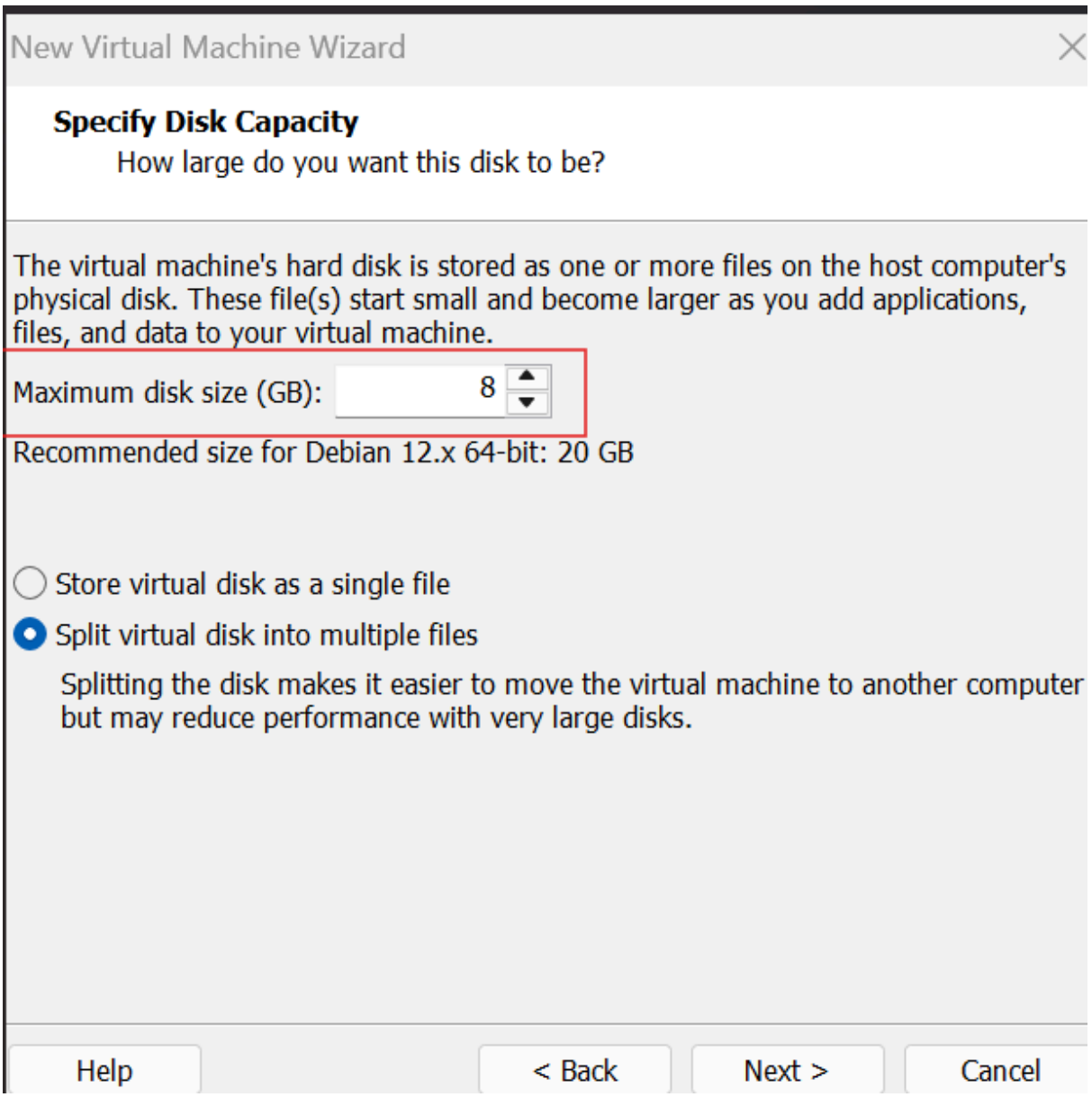


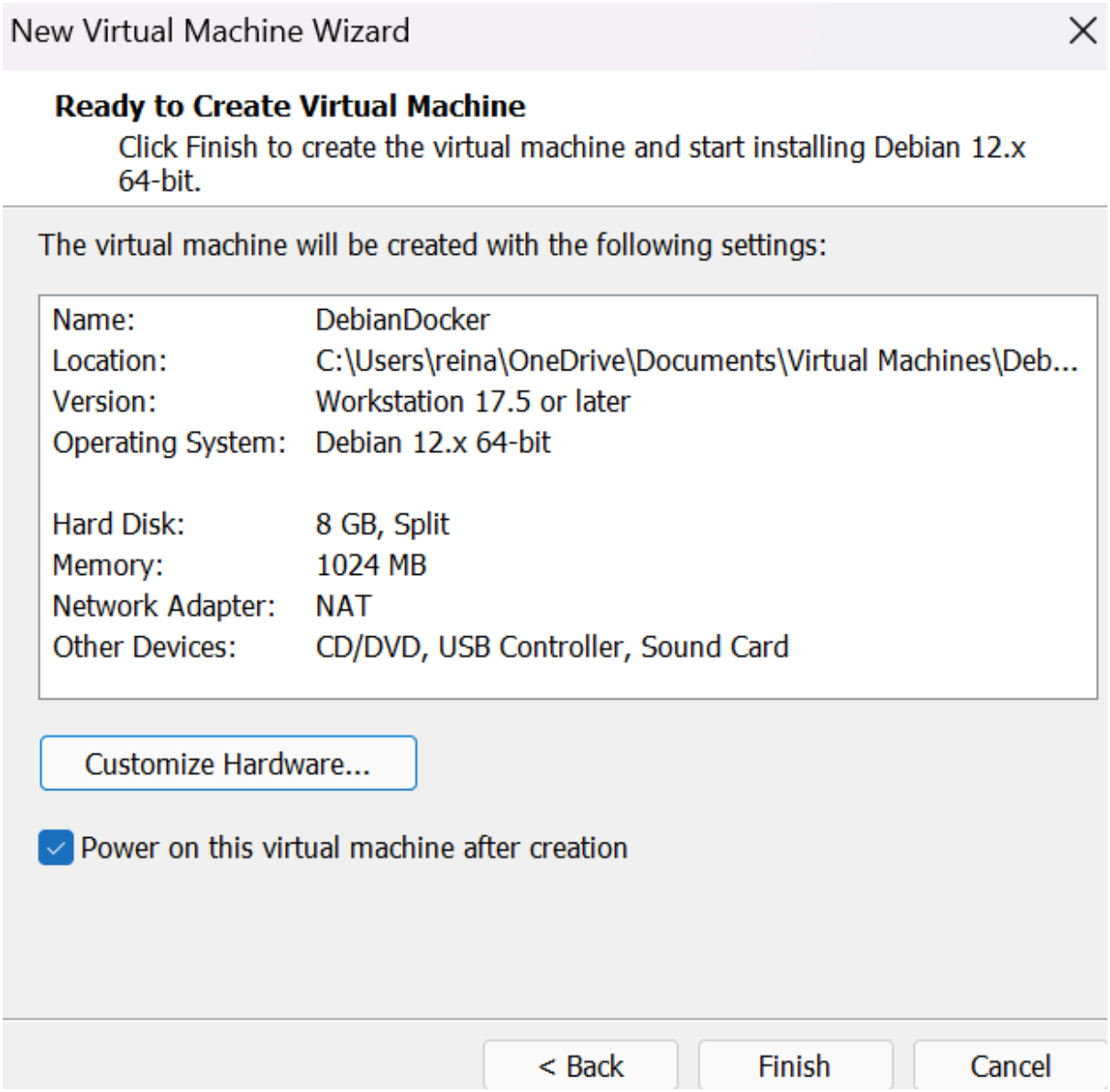


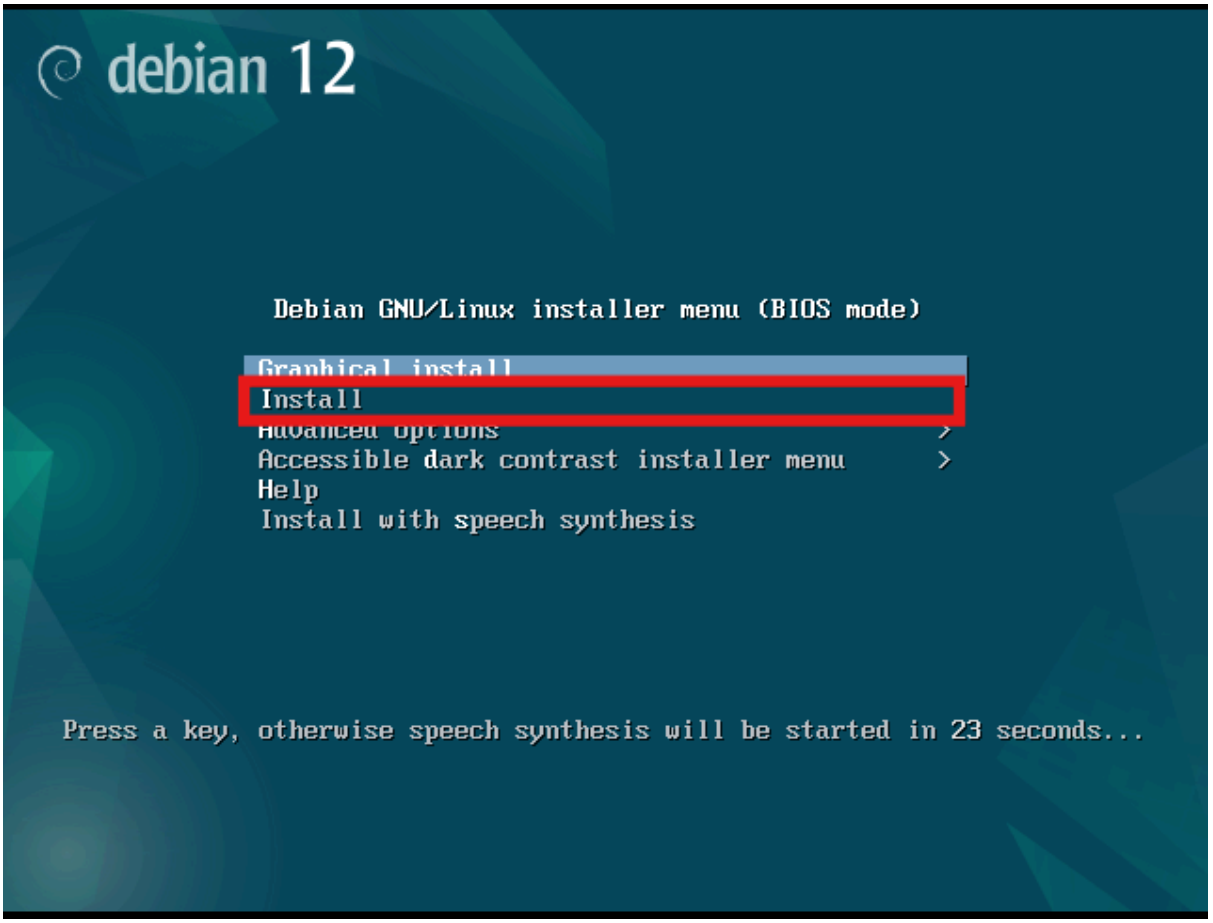
1 VCPU

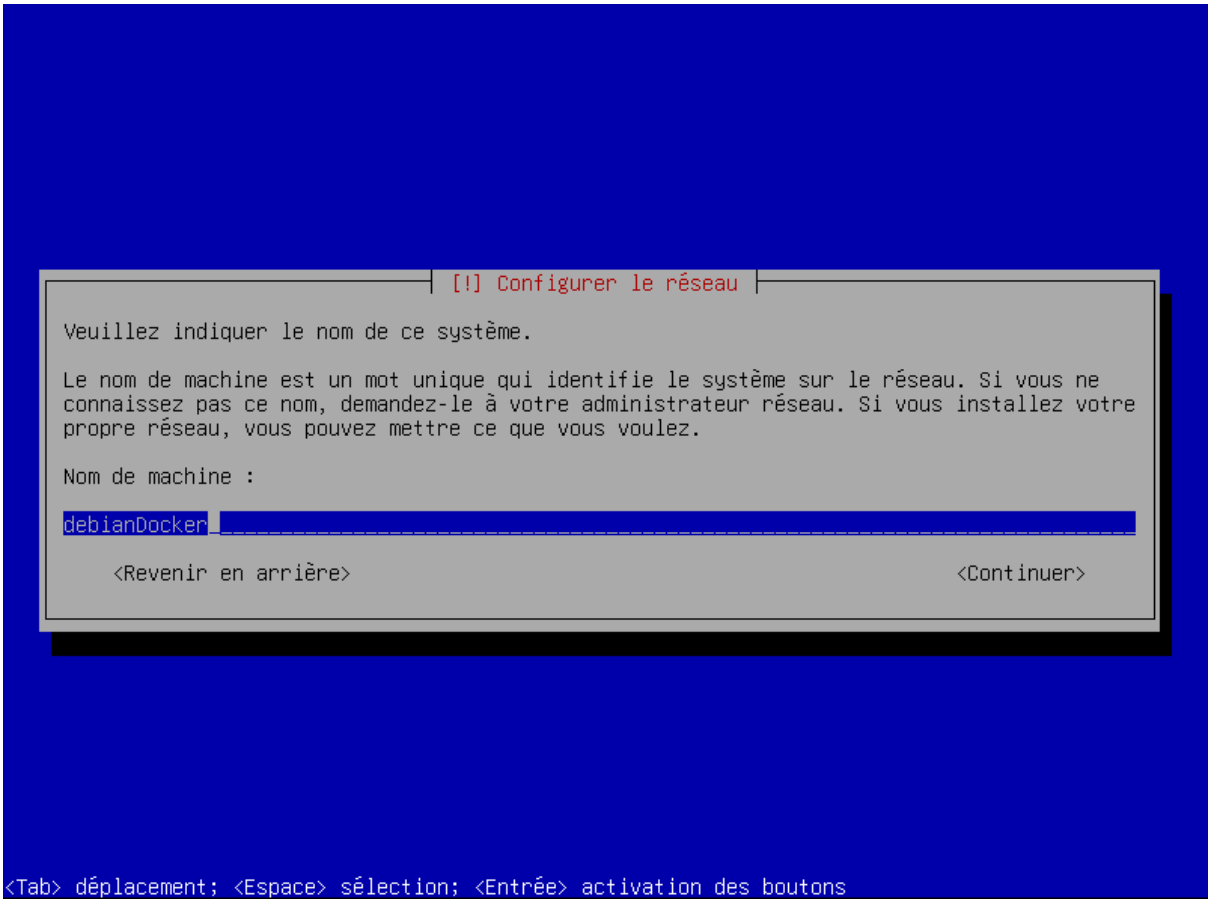
1 Go Ram

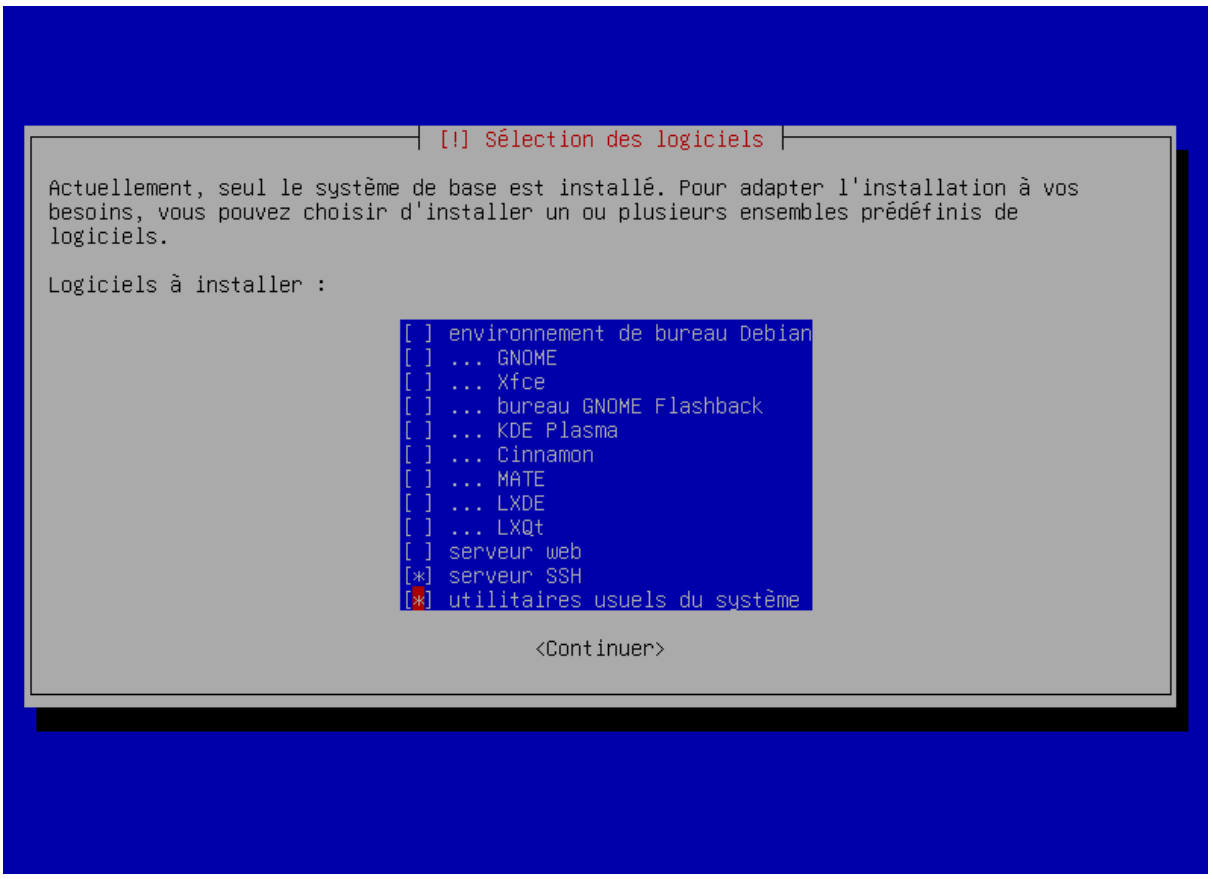
8 Go de DD


New Virtual Machine Wizard
Specify Disk Capacity
How large do you want this disk to be?
The virtual machine's hard disk is stored as one or more files on the host computer's physical disk. These file(s) start small and become larger as you add applications, files, and data to your virtual machine.
Maximum disk size (GB): 8
Recommended size for Debian 12.x 64-bit: 20 GB
Store virtual disk as a single file
Split virtual disk into multiple files
Splitting the disk makes it easier to move the virtual machine to another computer but may reduce performance with very large disks.
Help
< Back
Next >
Cancel


New Virtual Machine Wizard
Ready to Create Virtual Machine
Click Finish to create the virtual machine and start installing Debian 12.x 64-bit.
The virtual machine will be created with the following settings:
Name: DebianDocker
Location: C:\Users\reina\OneDrive\Documents\Virtual Machines\Deb...
Version: Workstation 17.5 or later
Operating System: Debian 12.x 64-bit
Hard Disk: 8 GB, Split
Memory: 1024 MB
Network Adapter: NAT
Other Devices: CD/DVD, USB Controller, Sound Card
Customize Hardware...
Power on this virtual machine after creation
< Back
Finish
Cancel


debian 12
Debian GNU/Linux installer menu (BIOS mode)
Graphical install
Install
Advanced options
Accessible dark contrast installer menu
Help
Install with speech synthesis
Press a key, otherwise speech synthesis will be started in 23 seconds...


[!] Configurer le réseau
Veuillez indiquer le nom de ce système.
Le nom de machine est un mot unique qui identifie le système sur le réseau. Si vous ne connaissez pas ce nom, demandez-le à votre administrateur réseau. Si vous installez votre propre réseau, vous pouvez mettre ce que vous voulez.
Nom de machine :
debianDocker
<Revenir en arrière>
<Continuer>
<Tab> déplacement; <Espace> sélection; <Entrée> activation des boutons


[!] Sélection des logiciels
Actuellement, seul le système de base est installé. Pour adapter l'installation à vos besoins, vous pouvez choisir d'installer un ou plusieurs ensembles prédéfinis de logiciels.
Logiciels à installer :
[ ] environnement de bureau Debian
[ ] ... GNOME
[ ] ... Xfce
[ ] ... bureau GNOME Flashback
[ ] ... KDE Plasma
[ ] ... Cinnamon
[ ] ... MATE
[ ] ... LXDE
[ ] ... LXQt
[ ] serveur web
[*] serveur SSH
[*] utilitaires usuels du système
<Continuer>

```
Debian GNU/Linux 12 debianDocker tty1

debianDocker login: youcef
Password:
Linux debianDocker 6.1.0-32-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.129-1 (2025-03-06) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
youcef@debianDocker:~$ _
```

apt install -y docker.io

```
root@debdocker:/etc/apt# apt update
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Tous les paquets sont à jour.
root@debdocker:/etc/apt# qpt updqte 77 qpt upgrqde )y
-bash: erreur de syntaxe près du symbole inattendu « ) »

root@debdocker:/etc/apt#
root@debdocker:/etc/apt# apt update
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Tous les paquets sont à jour.
root@debdocker:/etc/apt# apt upgrade
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Calcul de la mise à jour... Fait
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
root@debdocker:/etc/apt# apt install -y docker.io
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  binutils binutils-common binutils-x86-64-linux-gnu cgroupfs-mount containerd criu git git-man iptables libbinutils libctf-nobfd0 libctf0 liberror-perl
  libgprofng0 libintl-perl libintl-xs-perl libip6tc2 libmodule-find-perl libnet1 libnetfilter-conntrack3 libnfnetlink0 libnl-3-200 libproc-processtable-perl
  libprotobuf32 libsort-naturally-perl libterm-readkey-perl needrestart patch python3-protobuf runc sgml-base tini
Paquets suggérés :
  binutils-doc containernetworking-plugins docker-doc aufs-tools btrfs-progs debootstrap rinse rootlesskit xfsprogs zfs-fuse | zfsutils-linux git-daemon-run
  | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn firewalld needrestart-session | libnotify-bin ed diffutils-doc
  sgml-base-doc
Les NOUVEAUX paquets suivants seront installés :
  binutils binutils-common binutils-x86-64-linux-gnu cgroupfs-mount containerd criu docker.io git git-man iptables libbinutils libctf-nobfd0 libctf0
  liberror-perl libgprofng0 libintl-perl libintl-xs-perl libip6tc2 libmodule-find-perl libnet1 libnetfilter-conntrack3 libnfnetlink0 libnl-3-200
  libproc-processtable-perl libprotobuf32 libsort-naturally-perl libterm-readkey-perl needrestart patch python3-protobuf runc sgml-base tini
0 mis à jour, 33 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 84,3 Mo dans les archives.
Après cette opération, 360 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 runc amd64 1.1.5+ds1-1+deb12u1 [2 710 kB]
Réception de :2 http://deb.debian.org/debian bookworm/main amd64 containerd amd64 1.6.20~ds1-1+deb12u1 [25,9 MB]
24% [2 containerd 22,2 MB/25,9 MB 86%]                                                                    233 kB/s 4min 15s
```

docker --version

```
root@debdocker:/etc/apt# docker --version
Docker version 20.10.24+dfsg1, build 297e128
root@debdocker:/etc/apt#
```

Job 02

Tester l’installation de docker avec le conteneur « hello-world » et se familiariser avec les commandes.

docker run hello-world

Explication :

docker run :
C'est la commande pour lancer un conteneur Docker à partir d'une image.

hello-world :
C'est le nom de l'image Docker que l’on veut exécuter dans notre exemple.
Si elle n’est pas présente localement sur la machine, Docker va télécharger depuis Docker Hub (le registre officiel).

```
root@debdocker:/etc/apt# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:7e1a4e2d11e2ac7a8c3f768d4166c2defeb09d2a750b010412b6ea13de1efb19
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

root@debdocker:/etc/apt# _
```

Commandes gestion d’images :

docker pull <image> : Télécharge une image depuis Docker Hub vers votre machine locale.

docker images : Affiche la liste des images Docker présentes localement.

docker rmi <image> : Supprime une image Docker de votre machine

Commandes gestion des containers :

docker run <image> : Crée et démarre un nouveau conteneur à partir d'une image.

docker run -d <image> : Démarre un conteneur en arrière-plan (mode détaché).

docker run -it <image> bash : Démarre un conteneur en mode interactif avec un terminal bash.

docker ps : Affiche les conteneurs en cours d'exécution.

docker ps -a : Affiche tous les conteneurs, y compris ceux arrêtés.

docker stop <nom|id> : Arrête un conteneur en cours d'exécution.

docker start <nom|id> : Démarre un conteneur arrêté.

docker restart <nom|id> : Redémarre un conteneur.

docker rm <nom|id> : Supprime un conteneur arrêté.

Commandes Clean :

docker system prune : Supprime tous les objets inutilisés (conteneurs arrêtés, images non utilisées, volumes non utilisés, etc.).

docker volume ls : Liste tous les volumes Docker.

docker volume rm <nom> : Supprime un volume Docker.

Commandes utiles diverses :

docker exec -it <nom|id> bash : Ouvre un terminal interactif dans un conteneur en cours d'exécution.

docker logs <nom|id> : Affiche les logs (journaux) d'un conteneur.

docker inspect <nom|id> : Affiche des informations détaillées sur un conteneur ou une image.

Job 03

Utilisation de « Dockerfile » pour recréer le conteneur « helloworld » depuis une image Debian minimum.

Un **Dockerfile** est un **fichier texte contenant un ensemble d'instructions** qui servent à **construire une image Docker**. Ces instructions décrivent **comment configurer l'environnement** dans lequel une application va tourner, de manière automatisée et reproductible

Le Dockerfile sert à :

1. **Créer une image Docker personnalisée** pour ton application.

2. **Définir tout ce qu'il faut pour exécuter ton application** : système de base, bibliothèques, dépendances, fichiers sources, ports à exposer, etc.

3. **Faciliter la portabilité** : tout ce qui est dans le Dockerfile peut être exécuté de la même manière sur n'importe quelle machine ayant installé Docker.

Création du dossier et du Dockerfile :

mkdir ~/docker-helloworld
cd ~/docker-helloworld
nano Dockerfile

```
root@debdocker:/home/jordan/docker-helloworld# ls
Dockerfile
```

```
  GNU nano 7.2                                                    D
#image de base

FROM debian:bookworm-slim

#MAJ
RUN apt-get update && apt-get install -y curl && apt-get clean

#Commande exécutée

CMD echo "Hello from Jordan et Youcef pour le sujet la plateforme Docker"
```

Construction de l’image :

docker build -t helloworld-debian .

```
root@debdocker:/home/jordan/docker-helloworld# docker build -t helloworld-debian .
```

```
Successfully built 00c24e58b7fe
Successfully tagged helloworld-debian:latest
```

Lancement du container :

docker run helloworld-debian

```
Successfully tagged helloworld-debian:latest
root@debdocker:/home/jordan/docker-helloworld# docker run helloworld-debian
Hello from Jordan et Youcef pour le sujet la plateforme Docker
```

Job 04

Utilisation de Dockerfile pour créer une image SSH (compte : root et mot de passe : root123) sans utiliser une image SSH existante, voir redirection des ports (utiliser un autre port que le 22) , créé et lancez le conteneur et se connecter pour vérifier l'accès SSH.

Création du dossier et du Dockerfile

mkdir ~/docker-ssh
cd ~/docker-ssh

```
root@debdocker:/home/jordan# mkdir docker-ssh
```

nano Dockerfile

```
  GNU nano 7.2                                                  Dockerfile *
FROM debian:bookworm-slim

#MAJ + installation SSH
RUN apt-get update && apt-get install -y openssh-server && apt-get clean

#MDP pour root
RUN echo "root:root123" | chpasswd


#Création repertoire SSH
RUN mkdir /var/run/sshd

#Permettre accès root via SSH
RUN sed -i 's/^#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config

#Changer le port SSH
RUN sed -i 's/#Port 22/Port 2222/' /etc/ssh/sshd_config

#Exposer le port 2222
EXPOSE 2222


#Commande de démarrage du service SSH
CMD ["/usr/sbin/sshd", "-D"]
```

Création de l'image et lancement du container

docker build -t ssh-server .

```
root@debdocker:/home/jordan/docker-ssh# docker build -t ssh-server .
```

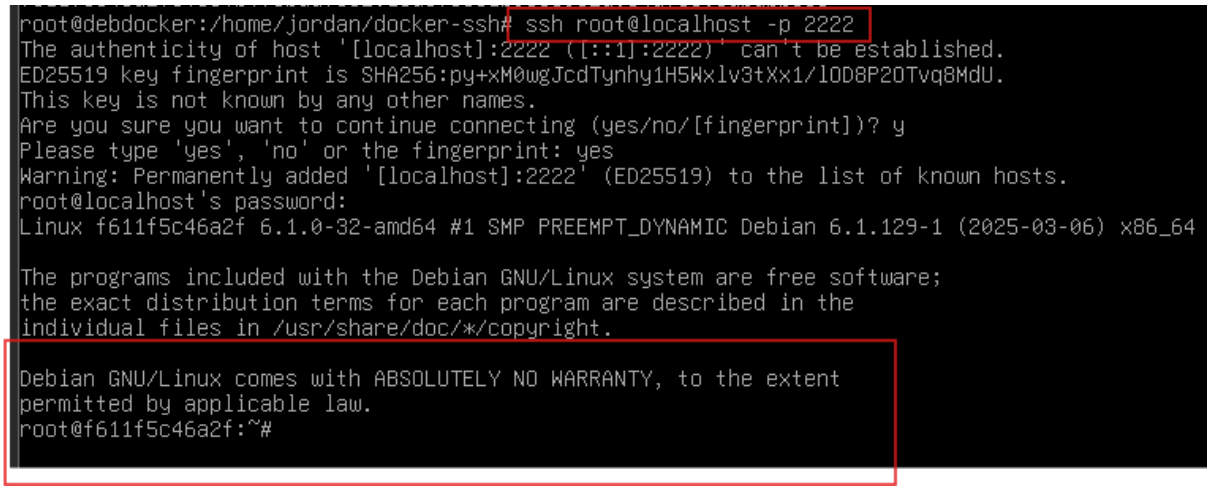
```
Successfully built a7dfbebd80b0
Successfully tagged ssh-server:latest
```

docker run -d -p 2222:2222 --name test-ssh ssh-server
port machine hote: port container

```
root@debdocker:/home/jordan/docker-ssh# docker run -d -p 2222:2222 --name test-ssh ssh-server
f611f5c46a2f87637b7fcbdaf56201cacfecc1b98389c2d0e4df380eabdbb8e5
```

Test du ssh avec l'utilisateur root

ssh root@localhost -p 2222



Job 05

Tout informaticien étant « flemmard » , il faut faire des alias pour les commandes docker en CLI , à mettre dans ~/.bashrc pour manipuler les images / containers.

Modification du bashrc pour créer les alias

nano ~/.bashrc

```
  GNU nano 7.2                                                    .bashrc
esac

# enable color support of ls and also add handy aliases
if [ -x /usr/bin/dircolors ]; then
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"
    alias ls='ls --color=auto'
    #alias dir='dir --color=auto'
    #alias vdir='vdir --color=auto'

    #alias grep='grep --color=auto'
    #alias fgrep='fgrep --color=auto'
    #alias egrep='egrep --color=auto'
fi

# colored GCC warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
#alias ll='ls -l'
#alias la='ls -A'
#alias l='ls -CF'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

# Docker alias
alias d='docker'
alias dps='docker ps'
```

Test

Exemple avec la commande **docker ps** et son alias **dps**

```
jordan@debdocker:~$ docker ps
CONTAINER ID   IMAGE        COMMAND               CREATED          STATUS          PORTS                                       NAMES
f611f5c46a2f   ssh-server   "/usr/sbin/sshd -D"   24 minutes ago   Up 24 minutes   0.0.0.0:2222->2222/tcp, :::2222->2222/tcp   test-ssh
jordan@debdocker:~$ dps
CONTAINER ID   IMAGE        COMMAND               CREATED          STATUS          PORTS                                       NAMES
f611f5c46a2f   ssh-server   "/usr/sbin/sshd -D"   24 minutes ago   Up 24 minutes   0.0.0.0:2222->2222/tcp, :::2222->2222/tcp   test-ssh
jordan@debdocker:~$
```

Job 06

Se renseigner sur l'utilisation de volumes entre deux conteneurs et la gestion des volumes.

Volume Docker : En termes simples, un volume Docker est un répertoire ou un espace de stockage dédié qui peut être partagé entre les conteneurs et persiste même lorsque le conteneur associé est arrêté ou supprimé. Cette fonctionnalité fait des volumes un outil essentiel pour la gestion des données dans un environnement Docker.

C'est quoi un volume commun ?

Un volume commun, c'est un volume partagé entre plusieurs conteneurs.
Dans le cadre de l'exercice on veut :

Le conteneur FTP reçoit le fichier index.html via FileZilla.

Le conteneur nginx affiche ce fichier dans le navigateur.

Pour que les deux accèdent au **même fichier**, on doit créer un **volume commun monté dans les deux conteneurs**, **dans le même dossier** (par exemple /usr/share/nginx/html).

https://blog.stephane-robert.info/docs/conteneurs/moteurs-conteneurs/volumes/#:~:text=Les%20volumes%20Docker%20standards%20repr%C3%A9sentent,la%20portabilit%C3%A9%20et%20la%20s%C3%A9curit%C3%A9.

Job 07

À l'aide d'un fichier yml , de docker-compose, faire deux conteneurs : nginx et FTP liés entre eux. Création d'un volume commun pour accéder au dossier web.
Créer sur votre PC un fichier index.html, dans ce fichier faites afficher votre nom/prénom). Installer FileZilla sur votre PC , se connecter en FTP sur le conteneur FTP pour envoyer le fichier index.html, et regarder le résultat.

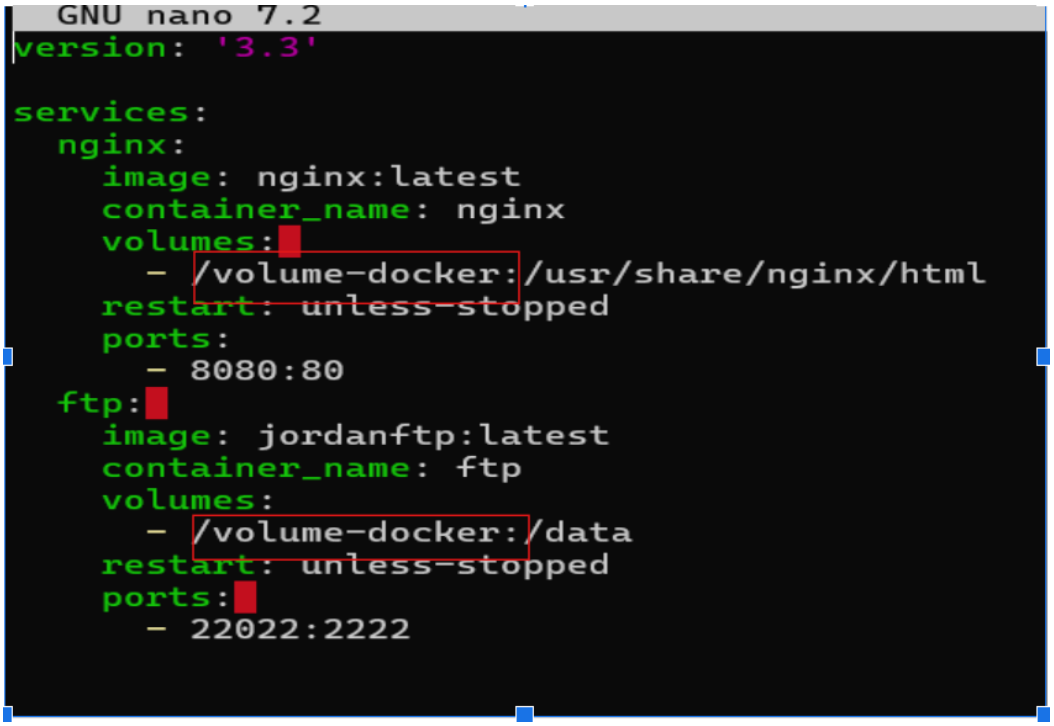
Docker Compose est un outil qui sert à **définir et gérer plusieurs conteneurs Docker en une seule commande** à l'aide d'un fichier de configuration

Installation Docker Compose :

sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

Création du fichier yml pour le job07



docker-compose up -d

```
root@debdocker:/home/jordan/job07# docker-compose up -d
Creating network "job07_default" with the default driver
Creating volume "job07_web_data" with default driver
Pulling nginx (nginx:latest)...
latest: Pulling from library/nginx
6e909acdb790: Already exists
5eaa34f5b9c2: Pull complete
417c4bccf534: Pull complete
e7e0ca015e55: Pull complete
373fe654e984: Pull complete
97f5c0f51d43: Pull complete
c22eb46e871a: Pull complete
Digest: sha256:124b44bfc9ccd1f3cedf4b592d4d1e8bddb78b51ec2ed5056c52d3692baebc19
Status: Downloaded newer image for nginx:latest
Pulling ftp (fauria/vsftpd:)...
latest: Pulling from fauria/vsftpd
2d473b07cdd5: Pull complete
33b95e46f70b: Pull complete
e22029c8d9a7: Pull complete
e1871c5d8fc9: Pull complete
c17c1255c529: Pull complete
ddcbab051542: Pull complete
1c68b0b593f1: Pull complete
dadb66293c59: Pull complete
99a54b7a405b: Pull complete
200facf93d0a: Pull complete
16ecaf7d0305: Pull complete
Digest: sha256:6d71d7c7f1b0ab2844ec7dc7999a30aef6d758b6d8179cf5967513f87c79c177
Status: Downloaded newer image for fauria/vsftpd:latest
Creating ftp_server ... done
Creating nginx_web  ... done
root@debdocker:/home/jordan/job07#
```

Vérification

docker inspect nginx_web | grep -A 10 Mounts

```
root@debdocker:/home/jordan/job07# docker inspect nginx_web | grep -A 10 Mounts
        "Mounts": [
            {
                "Type": "volume",
                "Name": "job07_web_data",
                "Source": "/var/lib/docker/volumes/job07_web_data/_data",
                "Destination": "/usr/share/nginx/html",
                "Driver": "local",
                "Mode": "rw",
                "RW": true,
                "Propagation": ""
            }
```

docker inspect ftp_server | grep -A 10 Mounts

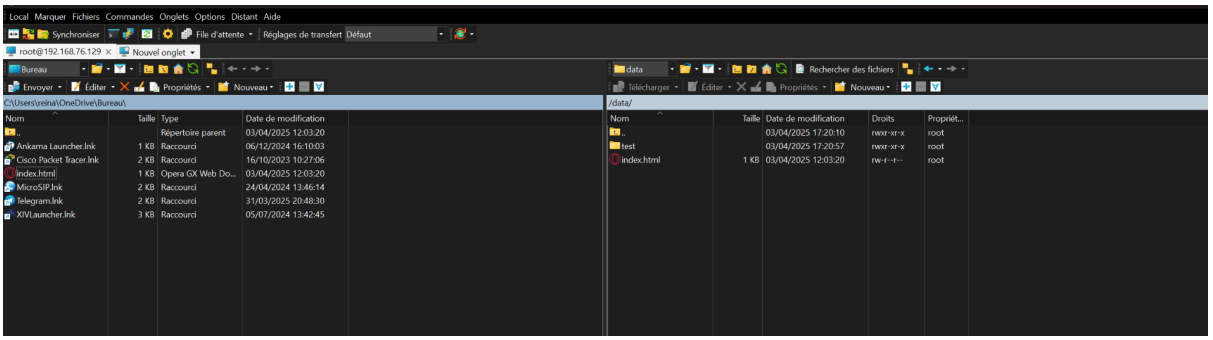
```
root@debdocker:/home/jordan/job07# docker inspect ftp_server | grep -A 10 Mounts
        "Mounts": [
            {
                "Type": "volume",
                "Name": "job07_web_data",
                "Source": "/var/lib/docker/volumes/job07_web_data/_data",
                "Destination": "/home_vsftpd",
                "Driver": "local",
                "Mode": "rw",
                "RW": true,
                "Propagation": ""
            }
```

docker ps

```
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                                                 NAMES
9f516b4844be   nginx:latest       "/docker-entrypoint.…"   48 minutes ago   Up 48 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp                 nginx
169b4078b7a3   jordanftp:latest   "/usr/sbin/sshd -D"      48 minutes ago   Up 48 minutes   0.0.0.0:22022->2222/tcp, :::22022->2222/tcp           ftp
```

Test du transfert de fichier :

WIN



Bienvenue, Jordan !

Job 08

**Sans utiliser une image nginx existante , utilisation de « Dockerfile » pour
créer un conteneur nginx, voir redirection des ports, et se connecter.**

Création dossier job08 et fichier essentiels

default.conf

```
 GNU nano 7.2                                                default.conf
server {
    listen 80;
    server_name localhost;

    location / {
        root /usr/share/nginx/html;
        index index.html;
    }
}
```

Dockerfile

```
 GNU nano 7.2                                                  Dockerfile
FROM debian:bullseye

RUN apt-get update && \
    apt-get install -y nginx && \
    apt-get clean

# Supprimer la config par défaut
RUN rm /etc/nginx/sites-enabled/default

# Copier ta config à la bonne place
COPY default.conf /etc/nginx/sites-enabled/default
COPY index.html /usr/share/nginx/html/index.html

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
```

docker-compose-yml

```
 GNU nano 7.2
version: '3.3'

services:
  nginx_custom:
    build: .
    ports:
      - "8088:80"
    container_name: nginx_custom_job08
```

index.html

```
 GNU nano 7.2                                                    index.html
<!DOCTYPE html>
<html>
<head>
  <title>Job 08</title>
</head>
<body>
  <h1>Bienvenue, Jordan !</h1>
</body>
</html>
```

Construction de l'image et création des containers

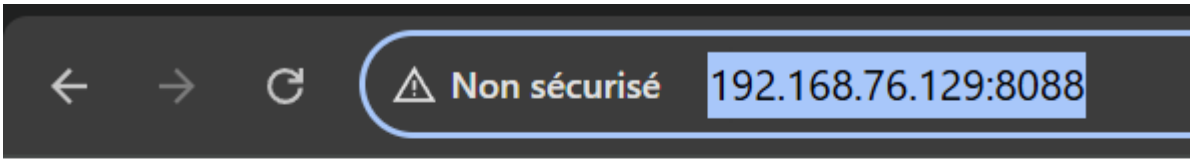
docker-compose build --no-cache

docker-compose up -d

Vérification

docker ps

```
jordan@debdocker:~/job08$ docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED         STATUS         PORTS                                   NAMES
1609c792d936   job08_nginx_custom   "nginx -g 'daemon of…"   6 minutes ago   Up 6 minutes   0.0.0.0:8088->80/tcp, :::8088->80/tcp   nginx_custom_job08
```



Bienvenue, Jordan !

Méthode Dockerfile simple

Création du dossier et du Dockerfile :

```
FROM debian:bookworm-slim

RUN apt-get update && apt-get install -y nginx && apt-get clean

# Modifier la conf nginx pour qu'il écoute sur 7500
RUN sed -i 's/listen 80 default_server;/listen 7500 default_server;/' /etc/nginx/sites-available/default && \
    sed -i 's/listen \[::\]:80 default_server;/listen [::]:7500 default_server;/' /etc/nginx/sites-available/default

EXPOSE 7500

CMD ["nginx", "-g", "daemon off;"]
```

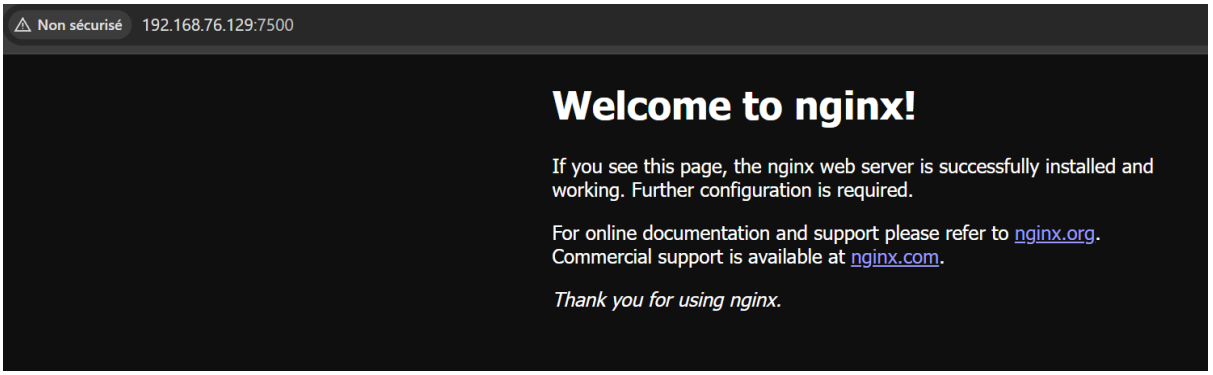
build et lancement de l'image :

docker build -t nginx-custom .
docker run -d -p 7500:7500 --name mon-nginx nginx-custom

```
jordan@debdocker:~/nginx-custom$ docker build -t nginx-custom .
Sending build context to Docker daemon  2.048kB
Step 1/5 : FROM debian:bookworm-slim
 ---> 59429810452a
Step 2/5 : RUN apt-get update && apt-get install -y nginx && apt-get clean
 ---> Running in 018eb1a5137d
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8792 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [512 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [258 kB]
Fetched 9305 kB in 2s (5301 kB/s)
Reading package lists...
Reading package lists...
Building dependency tree...
Reading state information...
```

```
Successfully built 49a4dfd7c533
Successfully tagged nginx-custom:latest
jordan@debdocker:~/nginx-custom$ docker run -d -p 7500:7500 --name mon-nginx nginx-custom
a2da5773dc0c1b57a198c5bd54f3a787c
```

test :



Job09

Création et utilisation d'un "registry" local. Et ajouter une « UI » pour le gérer
depuis une interface web.

Création dossier du job et de l’espace de stockage

mkdir dossier du job

mkdir data (espace stockage pour image du registry qui servira de volume)

Création fichier yml

fichier yml :

```
 GNU nano 7.2                                                    docker-compose.yml
version: '3.3'

services:
  registry:
    image: registry:2
    container_name: registry-server
    ports:
      - "5000:5000"
    volumes:
      - ./data:/var/lib/registry
    environment:
      REGISTRY_STORAGE_DELETE_ENABLED: "true"
      REGISTRY_HTTP_HEADERS_Access-Control-Allow-Origin: '["http://192.168.76.129:8081"]'
      REGISTRY_HTTP_HEADERS_Access-Control-Allow-Methods: '["GET", "OPTIONS", "PUT", "DELETE"]'
      REGISTRY_HTTP_HEADERS_Access-Control-Allow-Headers: '["Authorization", "Accept"]'

  registry-ui:
    image: joxit/docker-registry-ui:latest
    container_name: registry-ui
    ports:
      - "8081:80"
    depends_on:
      - registry
    environment:
      REGISTRY_TITLE: "Registry Local Privé"
      SINGLE_REGISTRY: "true"
      REGISTRY_NAME: "LocalRegistry"
      DELETE_IMAGES: "true"
      REGISTRY_SECURED: "false"
      REGISTRY_URL: "http://192.168.76.129:5000"
```

explication du fichier yml :

REGISTRY_STORAGE_DELETE_ENABLED : autorise la suppression des images du registre (par défaut, c’est désactivé).

REGISTRY_HTTP_HEADERS_Access-Control-Allow-Origin : autorise l’accès au registre depuis l’interface web à l’adresse spécifiée (nécessaire pour éviter les erreurs CORS).

REGISTRY_HTTP_HEADERS_Access-Control-Allow-Methods : définit les méthodes HTTP autorisées pour l’UI (lecture, envoi, suppression…).

REGISTRY_HTTP_HEADERS_Access-Control-Allow-Headers : indique les en-têtes HTTP acceptés (ex : Authorization, Accept), utilisés dans les requêtes de l'interface.

REGISTRY_TITLE: nom affiché dans l'interface.

SINGLE_REGISTRY: un seul registre géré.

DELETE_IMAGES: autorise la suppression via l'UI.

REGISTRY_SECURED: ici `false` → pas de HTTPS.

REGISTRY_URL: adresse locale du registry (doit correspondre à l'IP et port exposé).

docker tag job08_nginx_custom localhost :5000/job08_nginx_ custom :latest

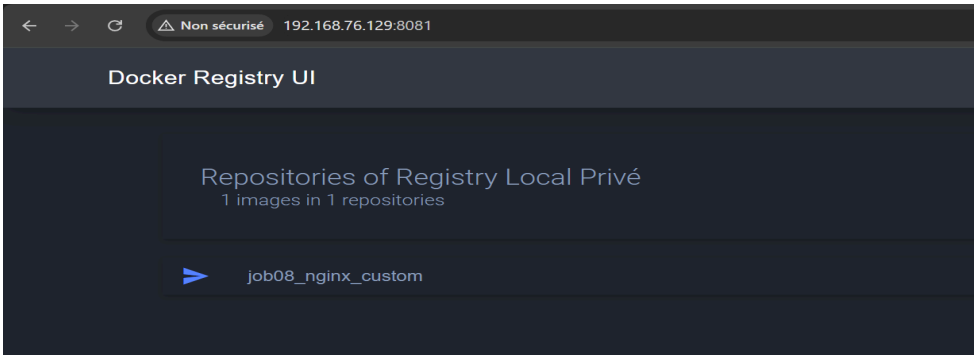
docker push localhost:5000/job08_nginx_custom:latest

```
jordan@debdocker:~/job09$ docker push localhost:5000/job08_nginx_custom:latest
The push refers to repository [localhost:5000/job08_nginx_custom]
f77aaece06d9: Pushed
14bd1175ce8d: Pushed
affaf525a517: Pushed
ebb8067cc083: Pushed
3dfea33954f7: Pushed
latest: digest: sha256:28d42840905d0b5757c3bb6049f0397bdac35bd084b59959edd2853ee49e6ec2 size: 1362
```

explication commande :

docker tag : prépare l'image à être poussée vers un registre
docker push : envoie l'image dans le registre docker local

Test :



Job 10

Faire un script bash , pour effacer totalement docker (les images , volumes ,conteneurs, et paquets correspondants) et rendre le système propre.
Et aussi un script pour automatiser l'installation de docker sur une machine Debian.

Fichier de désinstallation

```
  GNU nano 7.2                                                                                    uninstall.sh
#!/bin/bash

echo "🧹 Suppression complète de Docker..."

# 🔴 Arrêt des conteneurs s'ils existent
docker ps -aq &>/dev/null && docker stop $(docker ps -aq) 2>/dev/null
docker system prune -a -f --volumes

# 🗑 Suppression explicite de docker.io si installé depuis les dépôts Debian
apt-get purge -y docker.io || true

# 🗑 Suppression d'autres variantes Docker (au cas où)
apt-get purge -y docker-ce docker-ce-cli containerd containerd.io docker-buildx-plugin docker-compose-plugin || true

# 🧽 Nettoyage des dépendances
apt-get autoremove -y
apt-get clean

# 🧹 Suppression manuelle des fichiers liés à Docker
rm -rf /var/lib/docker
rm -rf /var/lib/containerd
rm -rf /etc/docker
rm -rf /etc/systemd/system/docker.service.d
rm -rf /usr/local/bin/docker*
rm -rf /usr/bin/docker*
rm -rf ~/.docker

echo -e "\n✅ Docker et tous ses composants ont été supprimés proprement."
```

sudo sh uninstall.sh

```
jordan@debdocker:~/job10$ sudo sh uninstall.sh
🧹 Suppression complète de Docker...
uninstall.sh: 6: docker: not found
uninstall.sh: 7: docker: not found
uninstall.sh: 6: docker: not found
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  cgroupfs-mount containerd criu git git-man iptables liberror-perl libintl-perl libintl-xs-perl libip6tc2 libmodule-find-perl libnet1 libnetfilter-conntrack3 libnfnetlink0 libnl-3-200 libproc-processtable-perl libprotobuf32 libsort-naturally-perl libterm-readkey-perl
  needrestart patch python3-protobuf runc sgml-base tini
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
Les paquets suivants seront ENLEVÉS :
  docker.io*
0 mis à jour, 0 nouvellement installés, 1 à enlever et 0 non mis à jour.
Après cette opération, 151 Mo d'espace disque seront libérés.
(Lecture de la base de données... 37245 fichiers et répertoires déjà installés.)
Suppression de docker.io (20.10.24+dfsg1-1+deb12u1) ...
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
(Lecture de la base de données... 37028 fichiers et répertoires déjà installés.)
Purge des fichiers de configuration de docker.io (20.10.24+dfsg1-1+deb12u1) ...
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Le paquet « docker-ce » n'est pas installé, et ne peut donc être supprimé
Le paquet « docker-ce-cli » n'est pas installé, et ne peut donc être supprimé
E: Impossible de trouver le paquet containerd.io
E: Impossible de trouver de paquet correspondant à l'expression rationnelle « containerd.io »
E: Impossible de trouver de paquet correspondant à l'expression rationnelle « containerd.io »
E: Impossible de trouver le paquet docker-buildx-plugin
E: Impossible de trouver le paquet docker-compose-plugin
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets suivants seront ENLEVÉS :
  cgroupfs-mount containerd criu git git-man iptables liberror-perl libintl-perl libintl-xs-perl libip6tc2 libmodule-find-perl libnet1 libnetfilter-conntrack3 libnfnetlink0 libnl-3-200 libproc-processtable-perl libprotobuf32 libsort-naturally-perl libterm-readkey-perl
  needrestart patch python3-protobuf runc sgml-base tini
0 mis à jour, 0 nouvellement installés, 25 à enlever et 0 non mis à jour.
Après cette opération, 174 Mo d'espace disque seront libérés.
(Lecture de la base de données... 37026 fichiers et répertoires déjà installés.)
Suppression de cgroupfs-mount (1.4) ...
Suppression de containerd (1.6.20~ds1-1+deb12u1) ...
Suppression de criu (3.17.1-2+deb12u1) ...
Suppression de git (1:2.39.5-0+deb12u2) ...
Suppression de git-man (1:2.39.5-0+deb12u2) ...
Suppression de iptables (1.8.9-2) ...
Suppression de liberror-perl (0.17029-2) ...
Suppression de libintl-xs-perl (1.33-1) ...
Suppression de needrestart (3.6-4+deb12u3) ...
Suppression de libintl-perl (1.33-1) ...
Suppression de libip6tc2:amd64 (1.8.9-2) ...
Suppression de libmodule-find-perl (0.16-2) ...
Suppression de libnet1:amd64 (1.1.6+dfsg-3.2) ...
Suppression de libnetfilter-conntrack3:amd64 (1.0.9-3) ...
Suppression de libnfnetlink0:amd64 (1.0.2-2) ...
Suppression de libnl-3-200:amd64 (3.7.0-0.2+b1) ...
Suppression de libproc-processtable-perl:amd64 (0.634-1+b2) ...
Suppression de python3-protobuf (3.21.12-3) ...
Suppression de libprotobuf32:amd64 (3.21.12-3) ...
Suppression de libsort-naturally-perl (1.03-4) ...
Suppression de libterm-readkey-perl (2.38-2+b1) ...
Suppression de patch (2.7.6-7) ...
Suppression de runc (1.1.5+ds1-1+deb12u1) ...
Suppression de sgml-base (1.31) ...
```

```
ordan@debdocker:~/job10$ which docker
ordan@debdocker:~/job10$ dpkg -l | grep docker
ordan@debdocker:~/job10$ |
```

Fichier d’installation

```
  GNU nano 7.2                                                        install_docker_debian.sh
#!/bin/bash

echo "📦 Installation de Docker depuis les dépôts officiels Debian..."

# Mise à jour de la liste des paquets
sudo apt update

# Installation du paquet docker.io fourni par Debian
sudo apt install -y docker.io

# Activation du service Docker au démarrage
sudo systemctl enable docker

# Démarrage immédiat du service
sudo systemctl start docker

# Affichage de la version installée
echo -e "\n✅ Docker a été installé avec succès :"
docker --version

# Test rapide (facultatif)
echo -e "\n🔍 Test rapide : docker run hello-world"
sudo docker run hello-world
```

```
jordan@debdocker:~/job10$ sudo chmod +x install_docker_debian.sh
jordan@debdocker:~/job10$ sudo sh install_docker_debian.sh
📦 Installation de Docker depuis les dépôts officiels Debian...
Réception de :1 http://security.debian.org/debian-security bookworm-security InRelease [48,0 kB]
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Réception de :3 http://deb.debian.org/debian bookworm-updates InRelease [55,4 kB]
Réception de :4 http://security.debian.org/debian-security bookworm-security/main Sources [129 kB]
Réception de :5 http://security.debian.org/debian-security bookworm-security/main amd64 Packages [258 kB]
Réception de :6 http://security.debian.org/debian-security bookworm-security/main Translation-en [155 kB]
645 ko réceptionnés en 1s (587 ko/s)
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
7 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  cgroupfs-mount containerd criu git git-man iptables liberror-perl libintl-perl libintl-xs-perl libip6tc2 libmodule-find-perl libnet1 libnetfilter-conntrack3 libnfnetlink0 libnl-3-200 libproc-processtable-perl libprotobuf32 libsort-naturally-perl libterm-readkey-perl
  needrestart patch python3-protobuf runc sgml-base tini
Paquets suggérés :
  containernetworking-plugins docker-doc aufs-tools btrfs-progs debootstrap rinse rootlesskit xfsprogs zfs-fuse | zfsutils-linux git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn firewalld needrestart-session
  | libnotify-bin ed diffutils-doc sgml-base-doc
Les NOUVEAUX paquets suivants seront installés :
  cgroupfs-mount containerd criu docker.io git git-man iptables liberror-perl libintl-perl libintl-xs-perl libip6tc2 libmodule-find-perl libnet1 libnetfilter-conntrack3 libnfnetlink0 libnl-3-200 libproc-processtable-perl libprotobuf32 libsort-naturally-perl libterm-readkey-perl
  needrestart patch python3-protobuf runc sgml-base tini
0 mis à jour, 26 nouvellement installés, 0 à enlever et 7 non mis à jour.
Il est nécessaire de prendre 77,9 Mo dans les archives.
Après cette opération, 325 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 runc amd64 1.1.5+ds1-1+deb12u1 [2 710 kB]
Réception de :2 http://deb.debian.org/debian bookworm/main amd64 containerd amd64 1.6.20~ds1-1+deb12u1 [25,9 MB]
Réception de :3 http://deb.debian.org/debian bookworm/main amd64 libip6tc2 amd64 1.8.9-2 [19,4 kB]
Réception de :4 http://deb.debian.org/debian bookworm/main amd64 libnfnetlink0 amd64 1.0.2-2 [15,1 kB]
Réception de :5 http://deb.debian.org/debian bookworm/main amd64 libnetfilter-conntrack3 amd64 1.0.9-3 [40,7 kB]
Réception de :6 http://deb.debian.org/debian bookworm/main amd64 iptables amd64 1.8.9-2 [360 kB]
Réception de :7 http://deb.debian.org/debian bookworm/main amd64 tini amd64 0.19.0-1 [255 kB]
Réception de :8 http://deb.debian.org/debian bookworm/main amd64 docker.io amd64 20.10.24+dfsg1-1+deb12u1 [36,2 MB]
Réception de :9 http://deb.debian.org/debian bookworm/main amd64 sgml-base all 1.31 [15,4 kB]
Réception de :10 http://deb.debian.org/debian bookworm/main amd64 cgroupfs-mount all 1.4 [6 276 B]
Réception de :11 http://deb.debian.org/debian bookworm/main amd64 libprotobuf32 amd64 3.21.12-3 [932 kB]
Réception de :12 http://deb.debian.org/debian bookworm/main amd64 python3-protobuf amd64 3.21.12-3 [245 kB]
Réception de :13 http://deb.debian.org/debian bookworm/main amd64 libnet1 amd64 1.1.6+dfsg-3.2 [60,3 kB]
Réception de :14 http://deb.debian.org/debian bookworm/main amd64 libnl-3-200 amd64 3.7.0-0.2+b1 [63,1 kB]
Réception de :15 http://deb.debian.org/debian bookworm/main amd64 criu amd64 3.17.1-2+deb12u1 [665 kB]
Réception de :16 http://deb.debian.org/debian bookworm/main amd64 liberror-perl all 0.17029-2 [29,0 kB]
Réception de :17 http://deb.debian.org/debian bookworm/main amd64 git-man all 1:2.39.5-0+deb12u2 [2 053 kB]
Réception de :18 http://deb.debian.org/debian bookworm/main amd64 git amd64 1:2.39.5-0+deb12u2 [7 260 kB]
```

```
✅ Docker a été installé avec succès :
Docker version 20.10.24+dfsg1, build 297e128
-e
🔍 Test rapide : docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:c41088499908a59aae84b0a49c70e86f4731e588a737f1637e73c8c09d995654
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/


jordan@debdocker:~/job10$
```

Job11

Découverte de portainer.
Refaire les Job 2 à 9 , en utilisant l'interface graphique de portainer. Se renseigner sur les alternatives à Portainer.

Création du volume et lancement du container Portainer

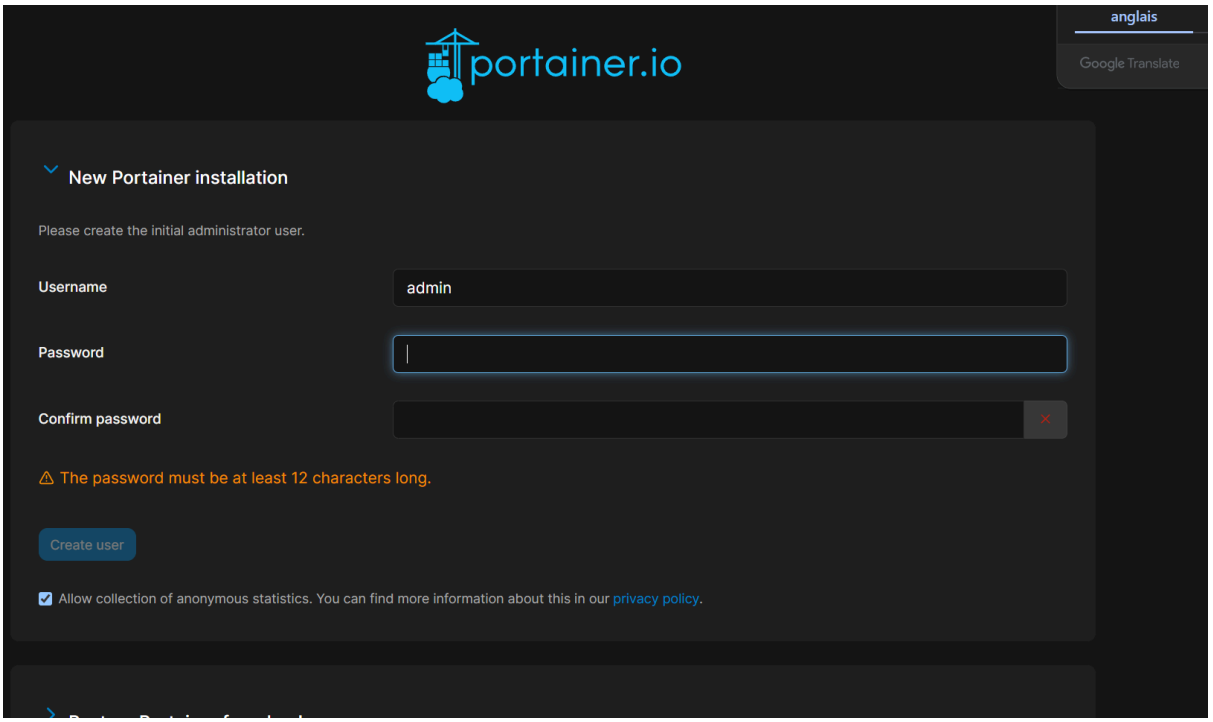
docker volume create portainer_data

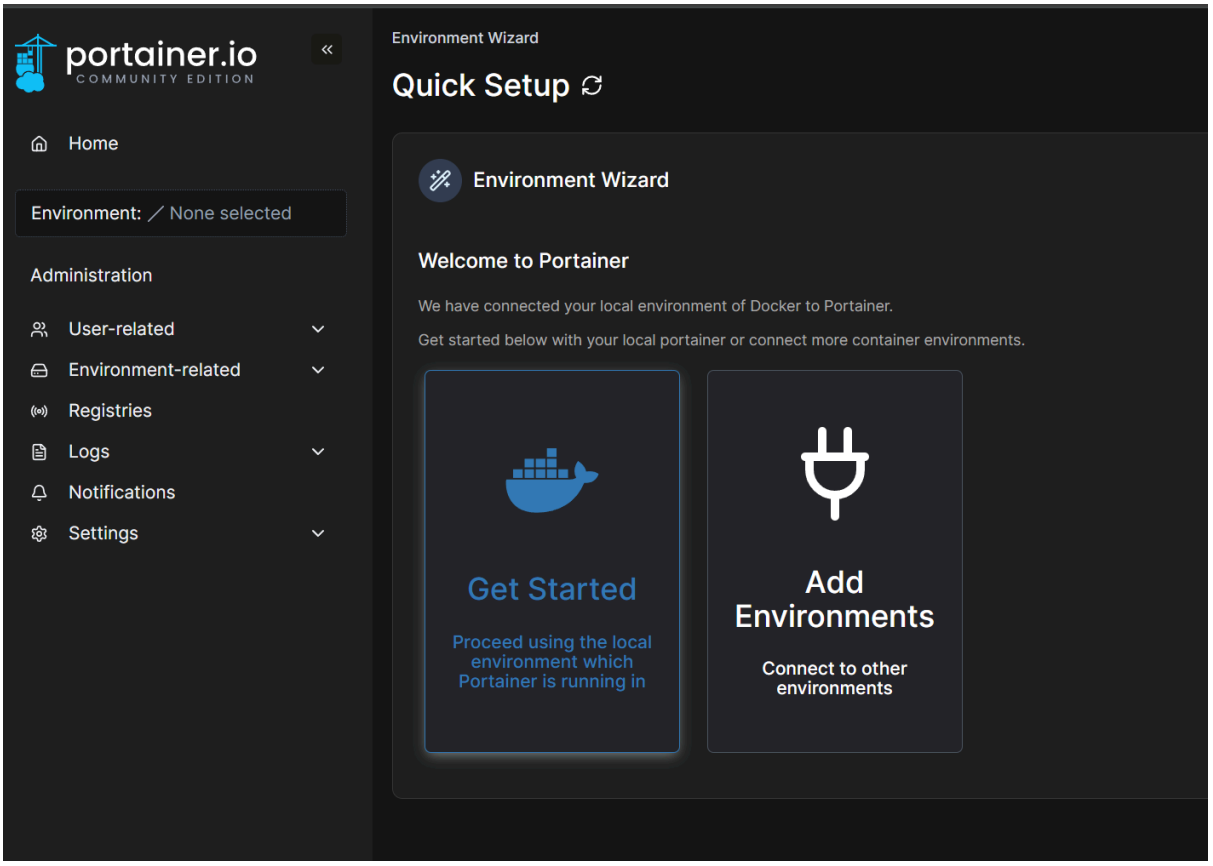
docker run -d -p 9000:9000 -p 9443:9443 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce

```
jordan@debdocker:~/job10$ docker run -d -p 9000:9000 -p 9443:9443 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce
Unable to find image 'portainer/portainer-ce:latest' locally
latest: Pulling from portainer/portainer-ce
6c1edd96ce20: Pull complete
cd0b075a95f6: Pull complete
04de093ad5ed: Pull complete
a9ff7abff372: Pull complete
e09df2601140: Pull complete
0e930de42c29: Pull complete
0df7a6683c39: Pull complete
df1dbbbff985: Pull complete
63bfee07d9ce: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:5f3f03a7d2e34fe836837f00183aa52ba1e914c9f51b9ef359aa5651a474ff5b
Status: Downloaded newer image for portainer/portainer-ce:latest
9a40d0b1cc5ea1e5a534ede137c4a44ca5a60893b39fa79d8e8c0613dac8b0ab
```

```
jordan@debdocker:~/job10$ docker ps
CONTAINER ID   IMAGE                  COMMAND        CREATED          STATUS          PORTS                                                                                                                                  NAMES
9a40d0b1cc5e   portainer/portainer-ce   "/portainer"   50 seconds ago   Up 48 seconds   0.0.0.0:9000->9000/tcp, :::9000->9000/tcp, 8000/tcp, 0.0.0.0:9443->9443/tcp, :::9443->9443/tcp   portainer
jordan@debdocker:~/job10$
```

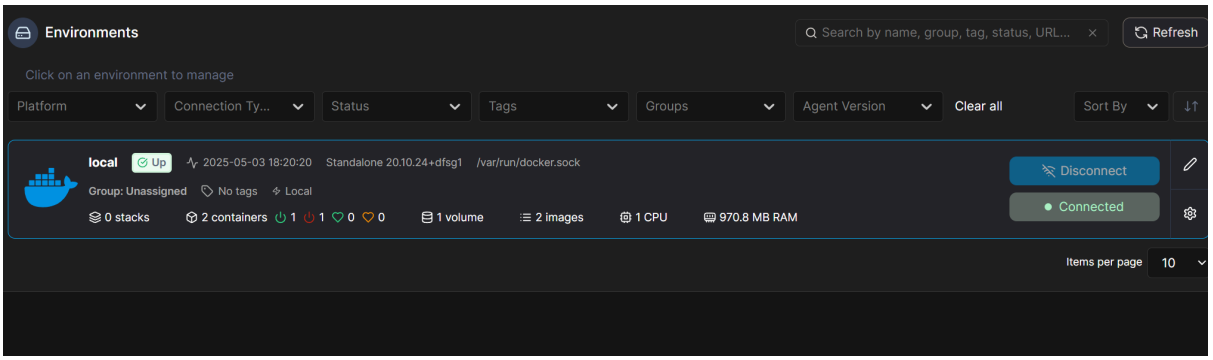
Création de l'Admin Portainer

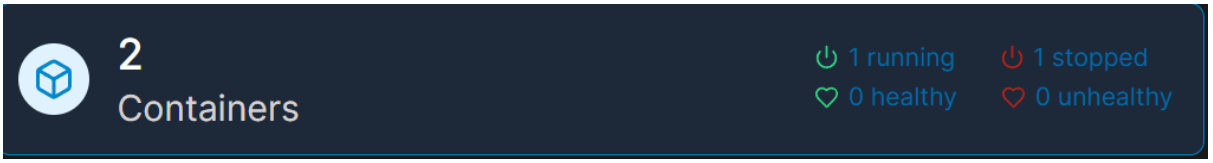


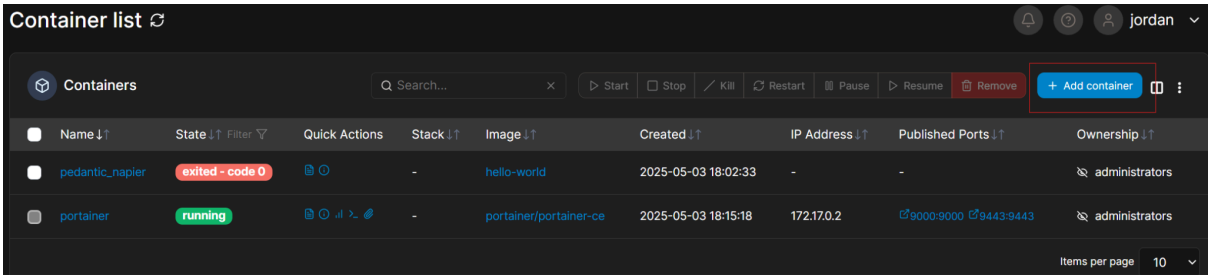

portainer.io
COMMUNITY EDITION
Home
Environment: None selected
Administration
User-related
Environment-related
Registries
Logs
Notifications
Settings
Environment Wizard
Quick Setup
Environment Wizard
Welcome to Portainer
We have connected your local environment of Docker to Portainer.
Get started below with your local portainer or connect more container environments.
Get Started
Proceed using the local environment which Portainer is running in
Add Environments
Connect to other environments

Job 2 version portainer

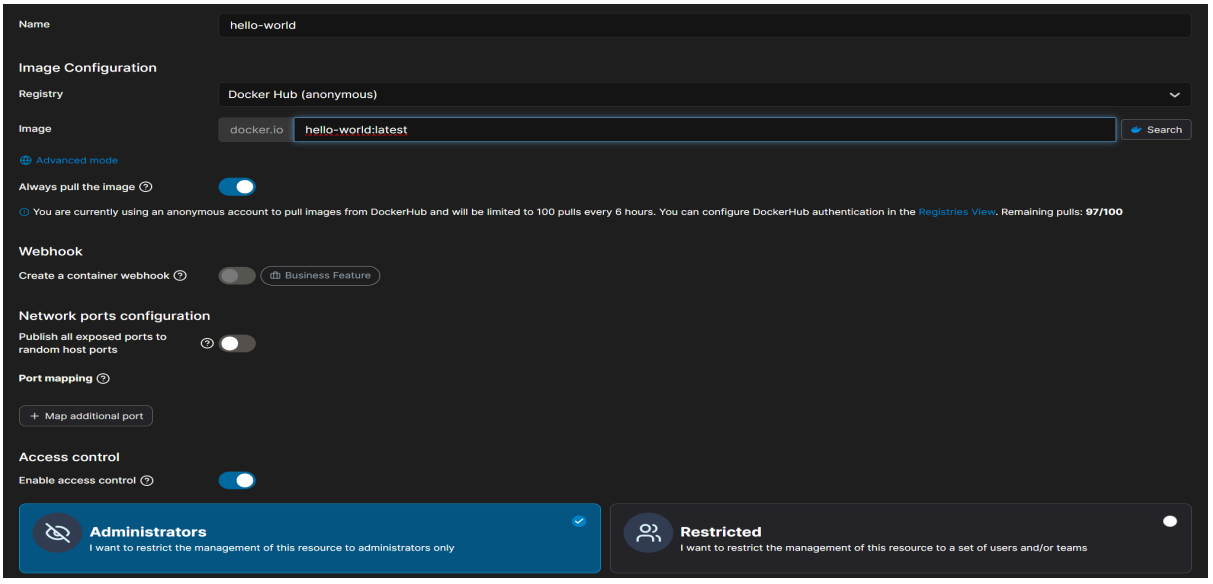
On clique sur local



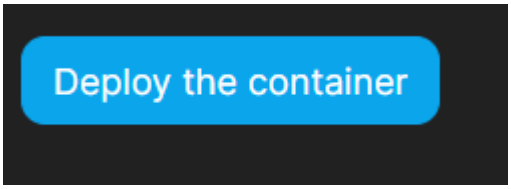




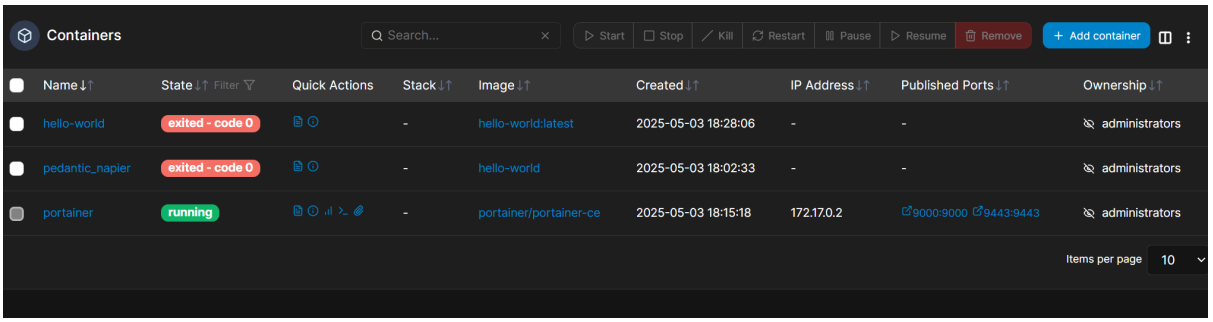
On clique sur sur container add container et on cherche l’image officielle de hello-world présente dans dockerhub



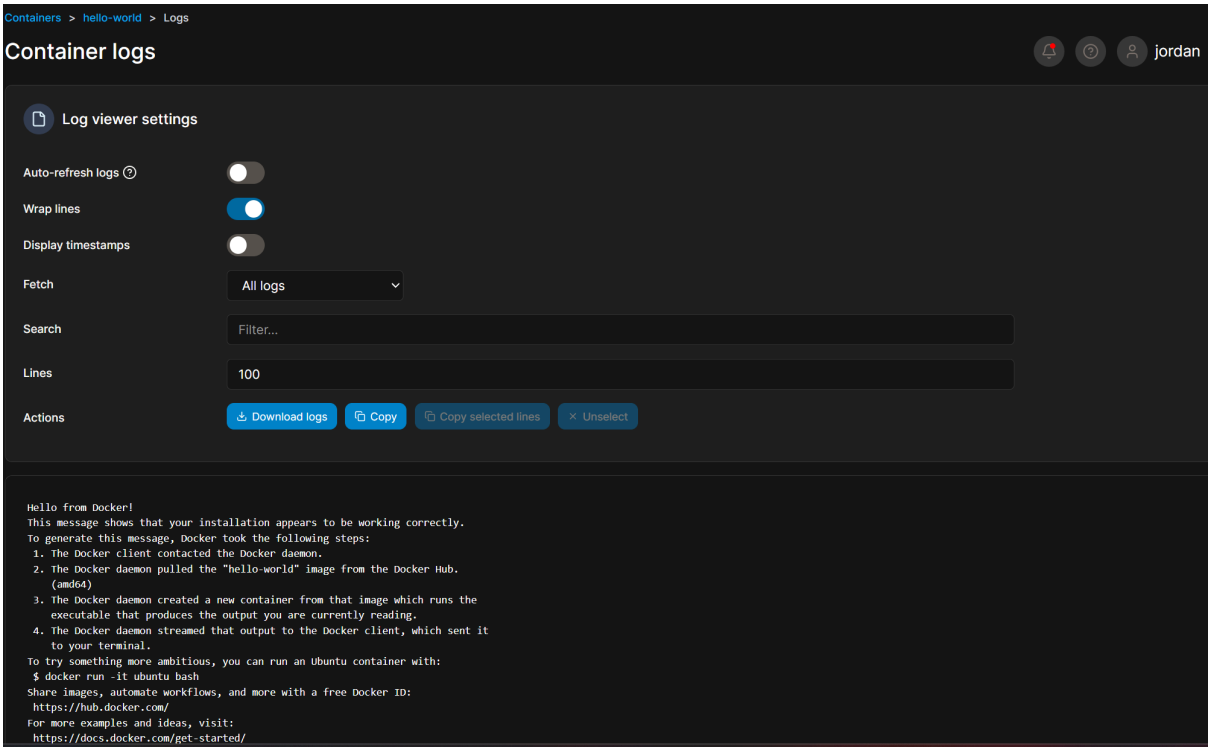
On déploie le container



On peut voir qu’il est maintenant bien présent, ce qui n’était pas le cas avant

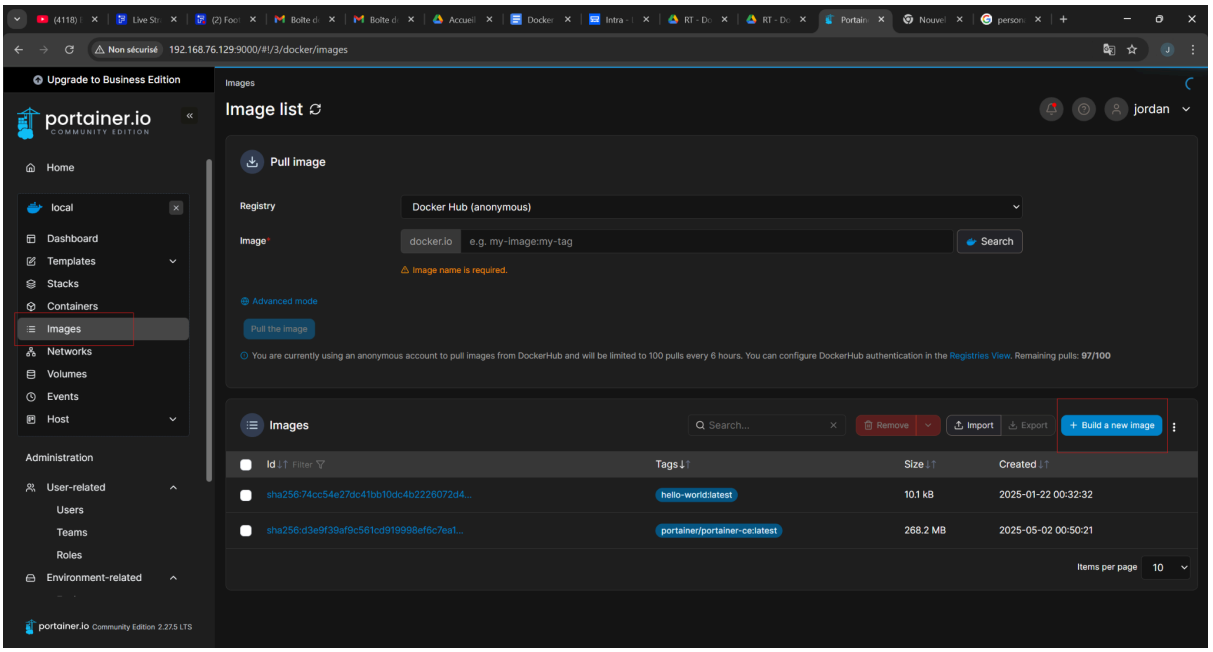


Vérification que le hello-world s’est bien lancé via les logs du container

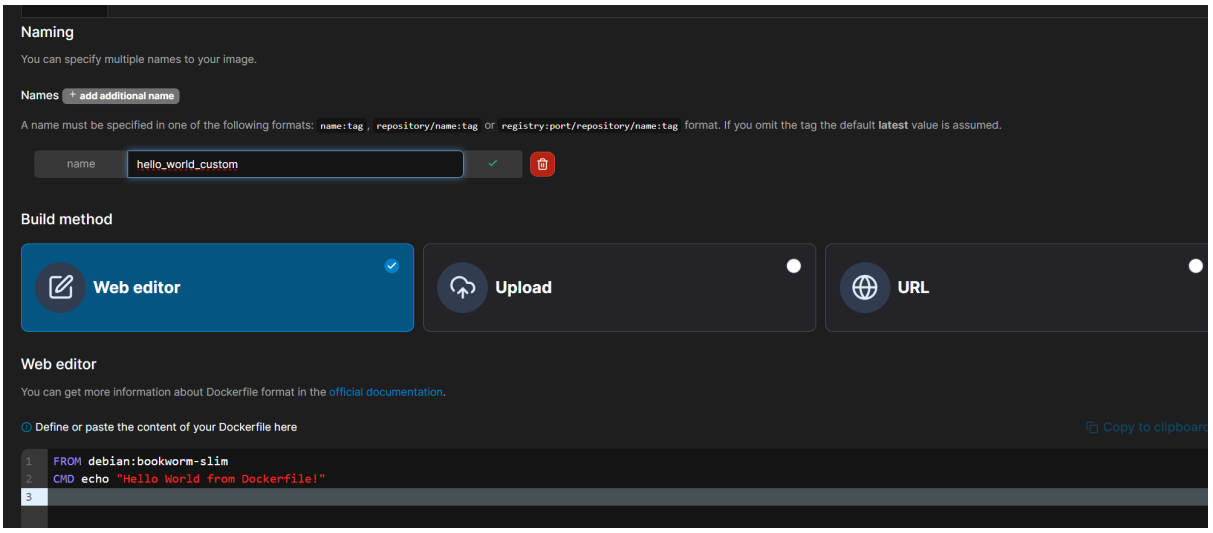


Job 03 version portainer

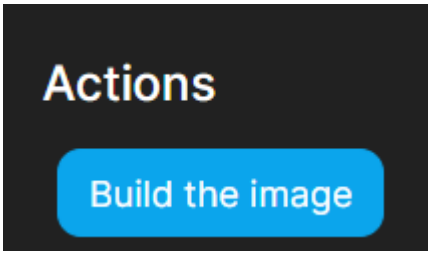
Construction de l’image

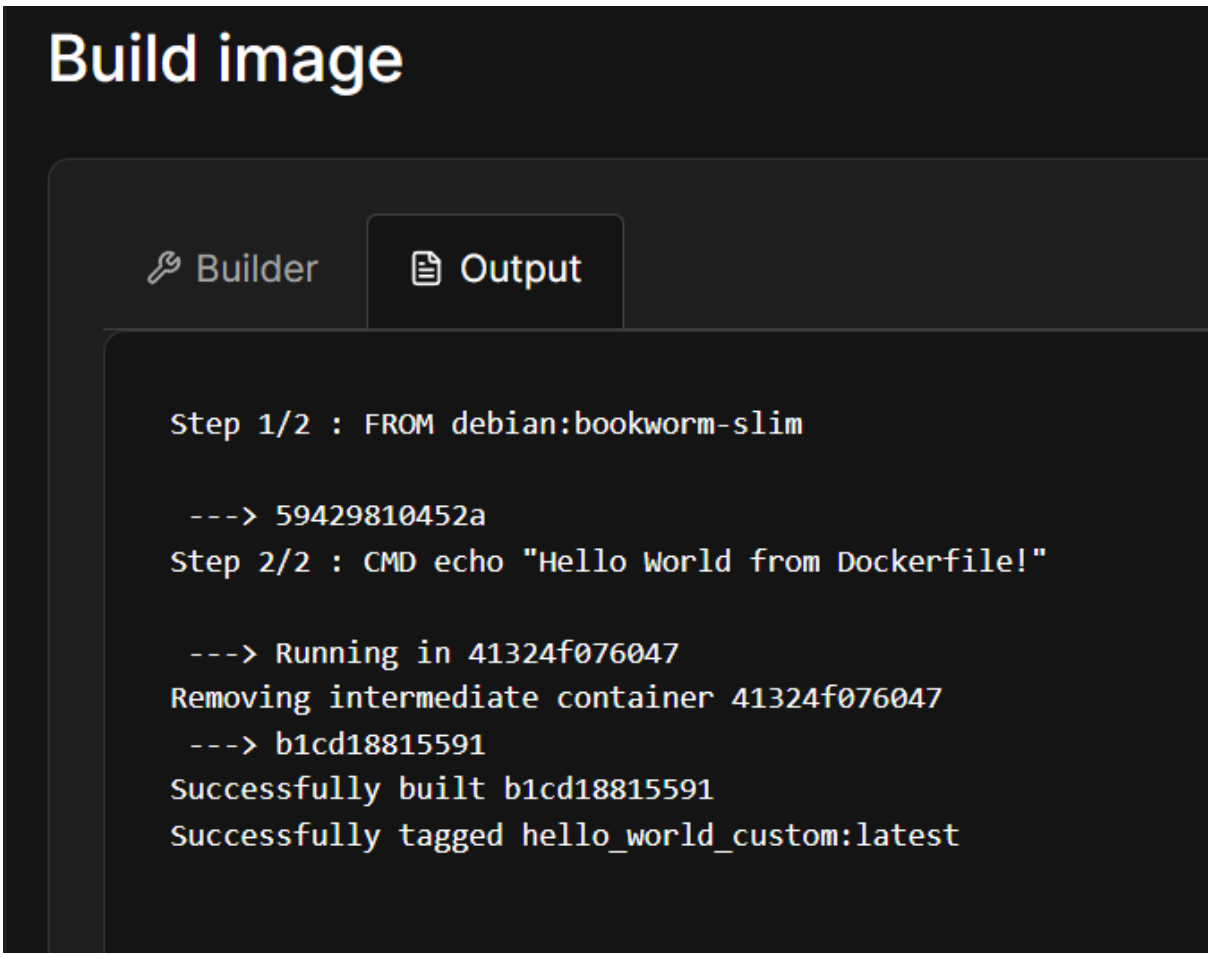


Création de l’image avec le Dockerfile



Finalisation de la construction de l’image





Création du container avec l'image custom build précédemment

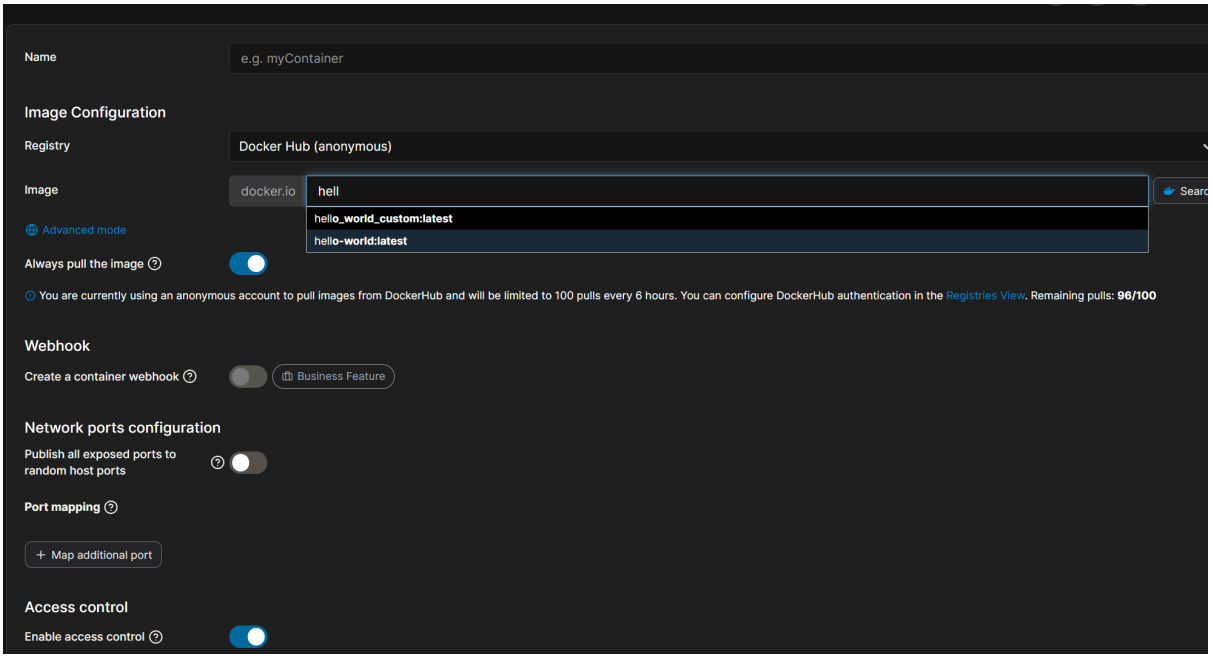
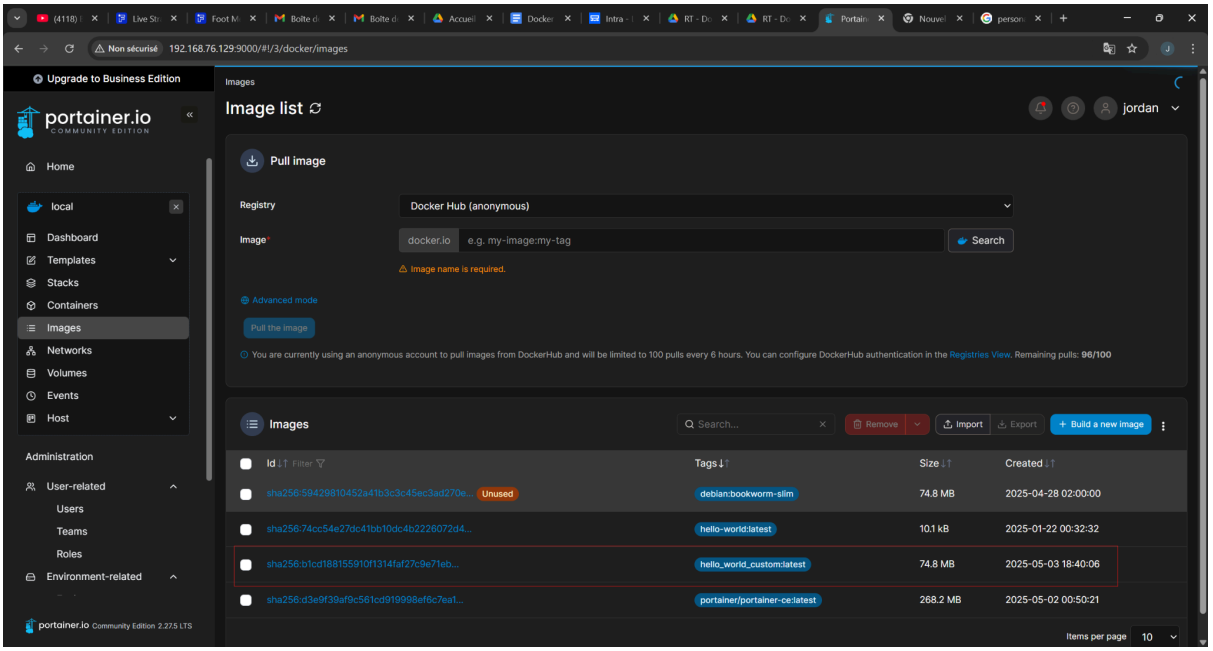
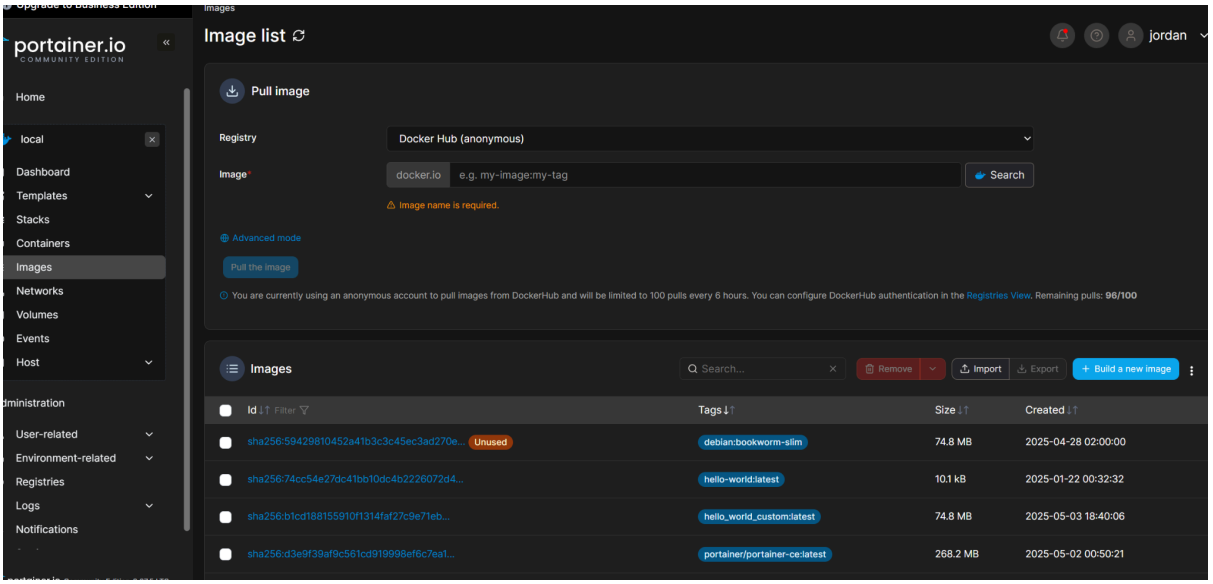


image montrant que l’image avait bien été build

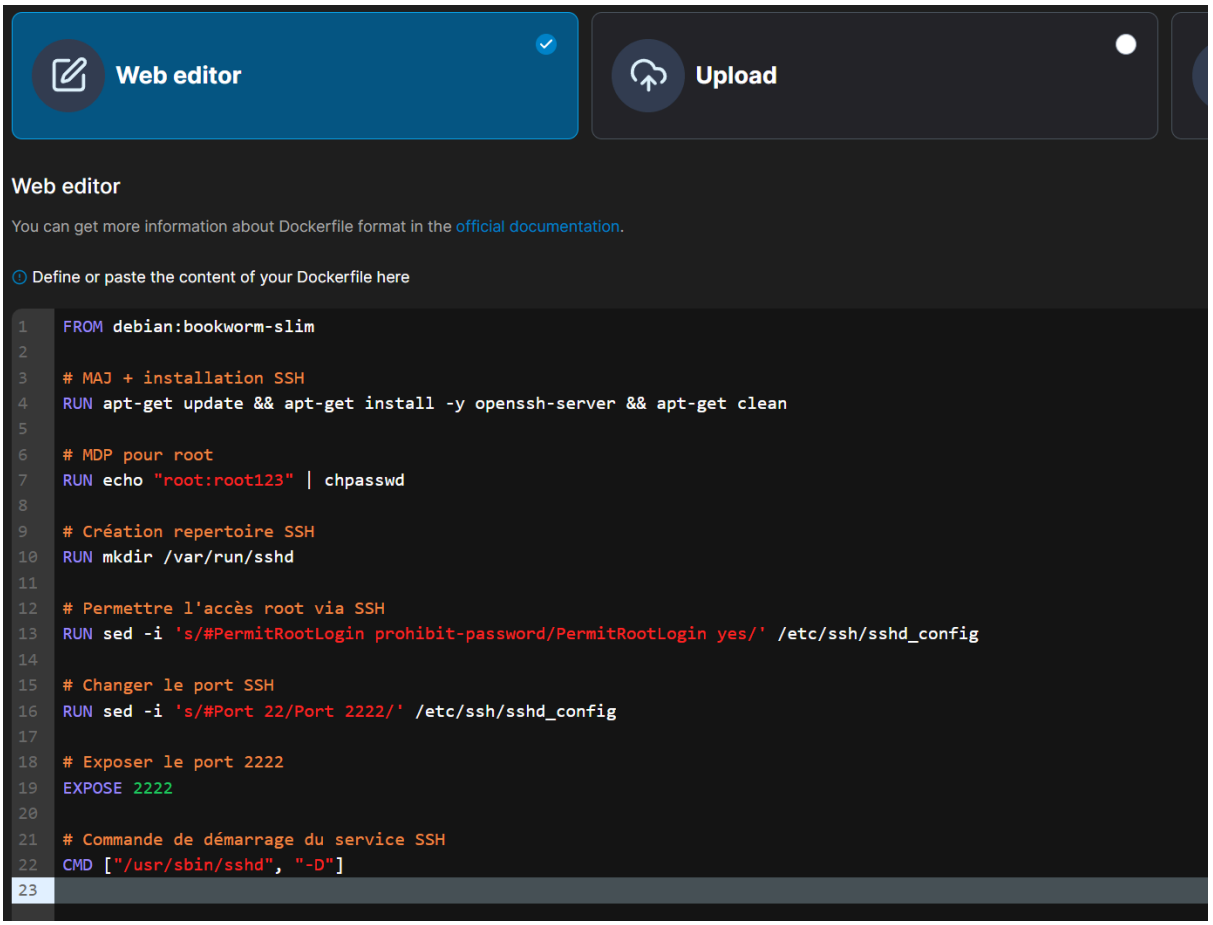


Job 04 version portainer

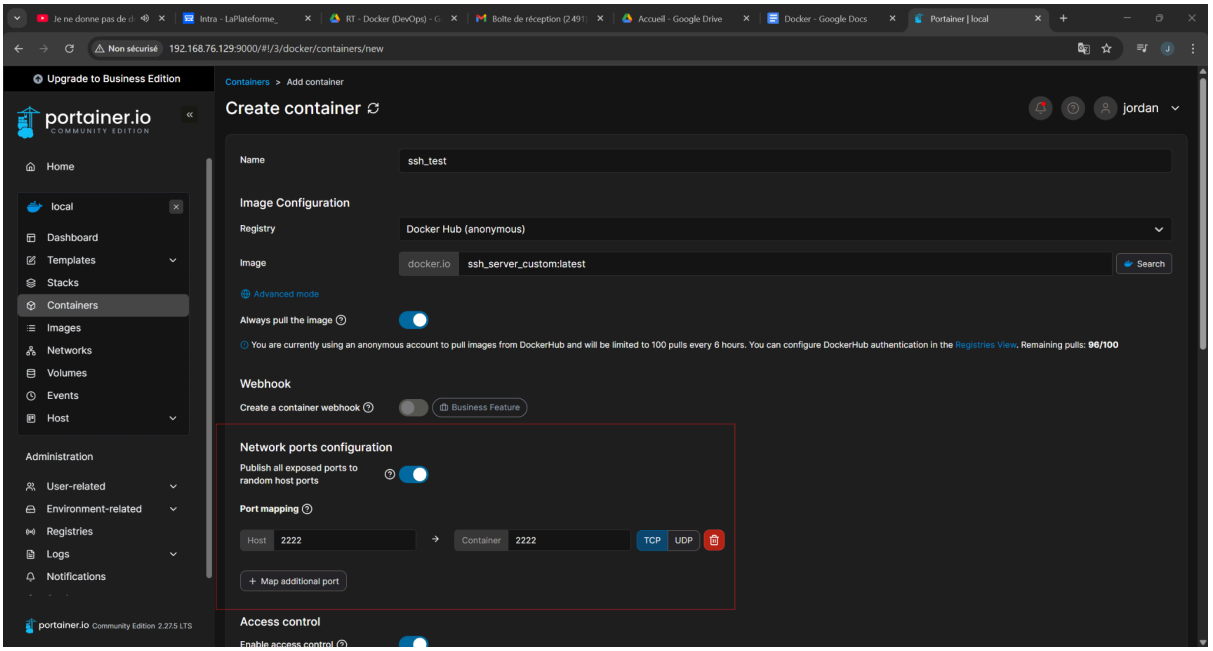
On va dans image



Ecriture du Dockerfile



Création du container



test de la connexion avec l’utilisateur root

```
Microsoft Windows [version 10.0.26100.3775]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\reina>ssh root@192.168.76.129 -p 2222
ssh: connect to host 192.168.76.129 port 2222: Connection refused

C:\Users\reina>ssh root@192.168.76.129 -p 2222
The authenticity of host '[192.168.76.129]:2222 ([192.168.76.129]:2222)' can't be established.
ED25519 key fingerprint is SHA256:v2Lr9KZ7+e2KW20S8mmnihL0KsefmA0Or8nqf9Kqpc8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? Y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '[192.168.76.129]:2222' (ED25519) to the list of known hosts.
root@192.168.76.129's password:
Linux b7648dbe5c0a 6.1.0-32-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.129-1 (2025-03-06) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@b7648dbe5c0a:~#
```

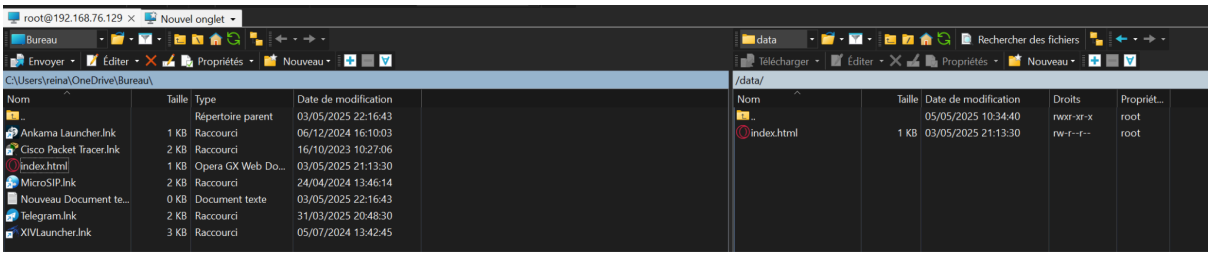
Job7 version portainer

Fichier yml dans stacks avec utilisation de l’image ssh du job 04

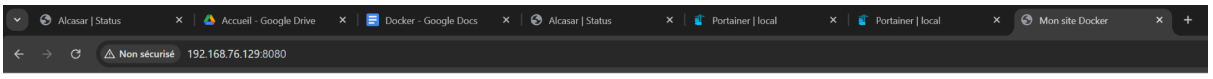
Define or paste the content of your docker compose file here

```yaml
version: "3.3"

services:
  nginx:
    image: nginx:latest
    container_name: nginx
    volumes:
      - /home/jordan/commondir:/usr/share/nginx/html
    restart: unless-stopped
    ports:
      - "8080:80"

  ftp:
    image: ssh_server_custom:latest
    container_name: ftpjordan
    volumes:
      - /home/jordan/commondir:/data
    restart: unless-stopped
    ports:
      - "2222:2222"
```

test avec WinSCP



Vérification que l’index.html de base a bien été remplacé par notre fichier transfert

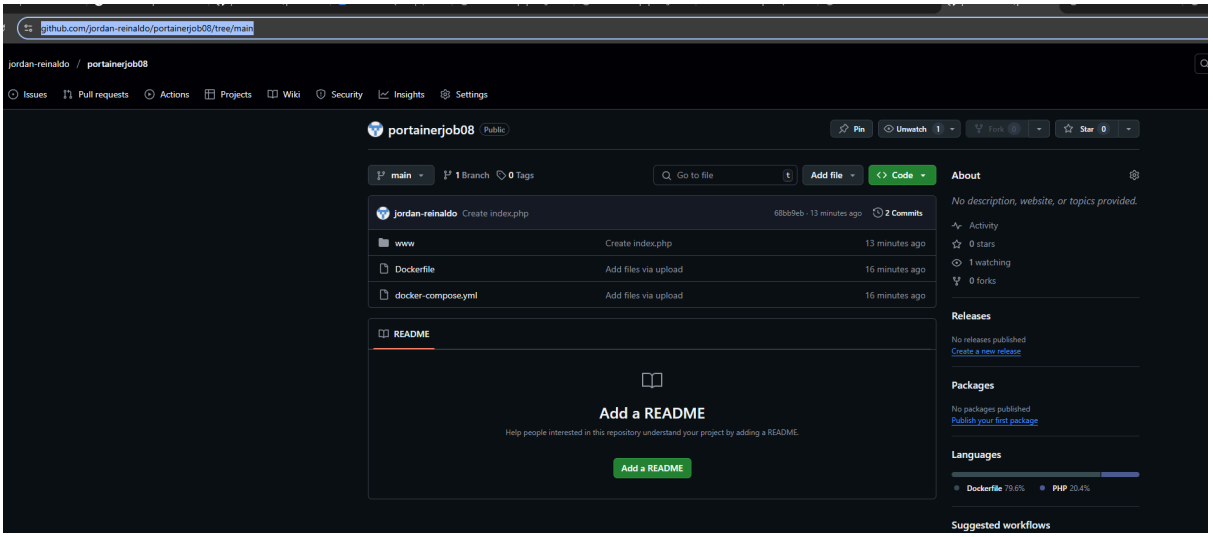


Jordan / Reinaldo ðŸ˜Ž

Job 8 version portainer

Apache version

fichier github pour le stack de portainer



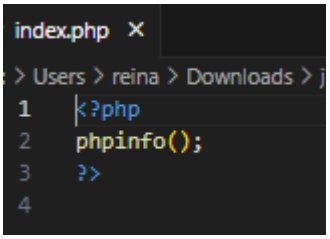
Dockerfile

```
FROM php:8.1-apache

RUN a2enmod rewrite

COPY www/ /var/www/html/

EXPOSE 80
```

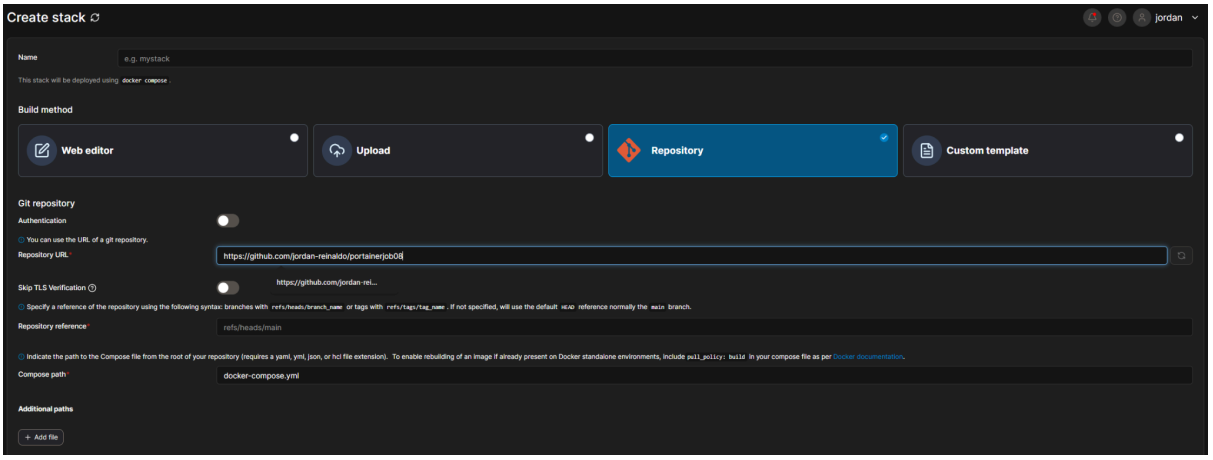
docker-compose

```
> Users > reina > Downloads > job08-apache > docker-compose.ym
1   version: '3.3'
2
3   services:
4     apache_web:
5       build: .
6       container_name: apache_job08
7       ports:
8         - "9700:80"
9       volumes:
10        - ./www:/var/www/html"
11
```

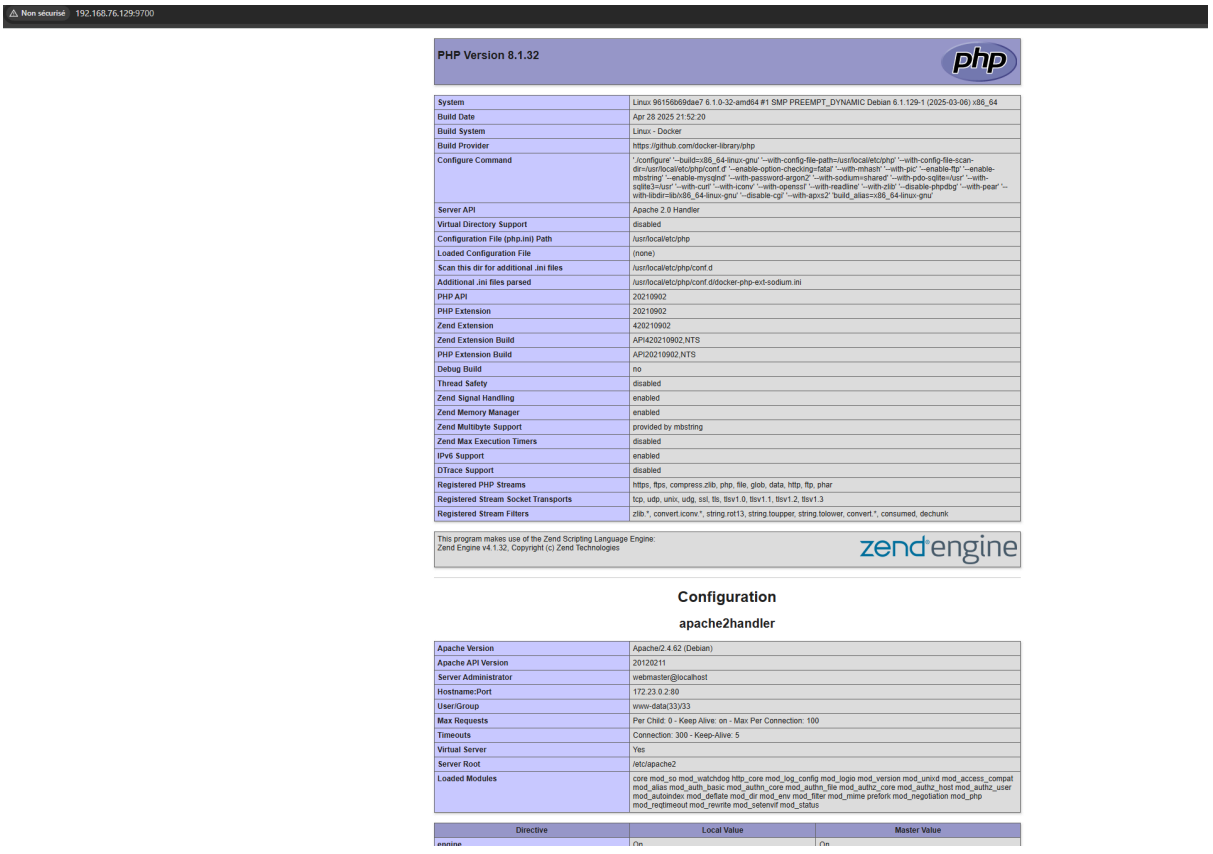
index.php



On utilise le repo github où l'on a placé nos fichiers pour faire notre stack

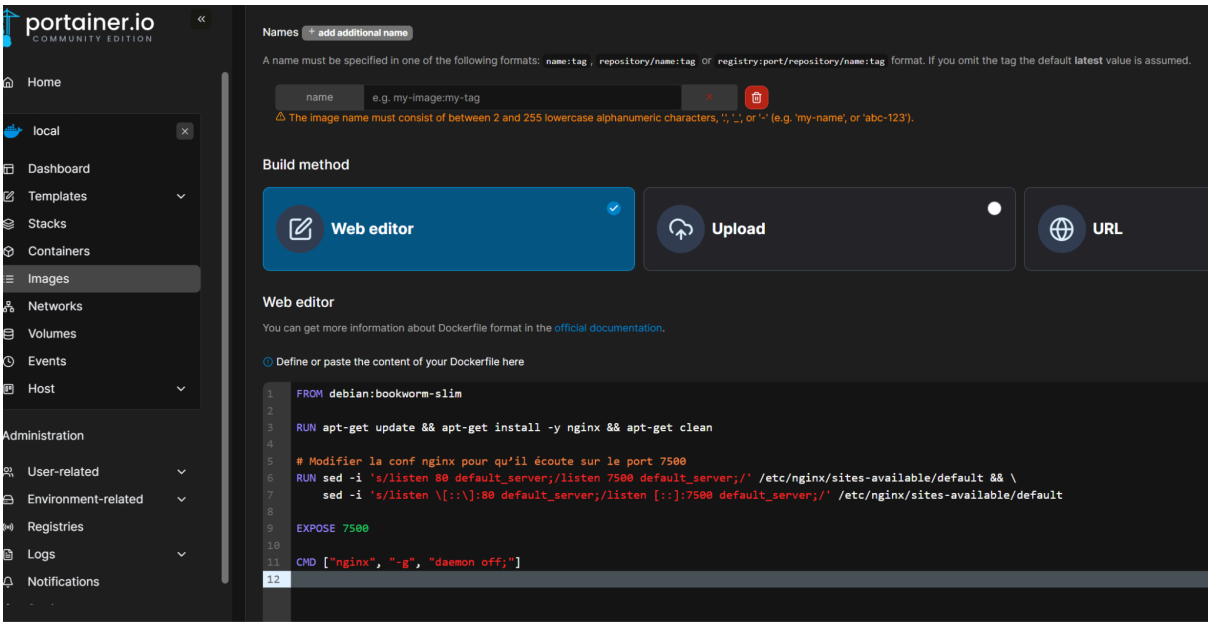


Test



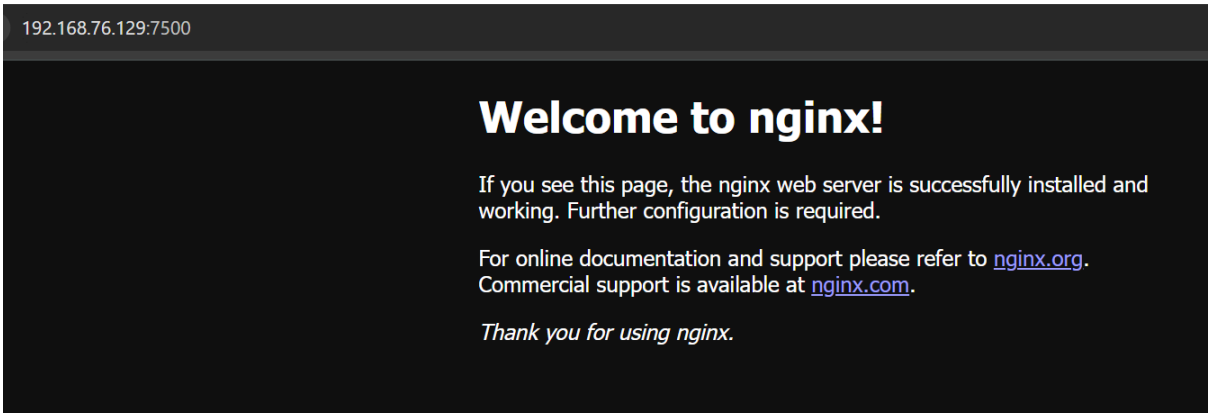
Nginx version :

Création de l’image



container créer

test

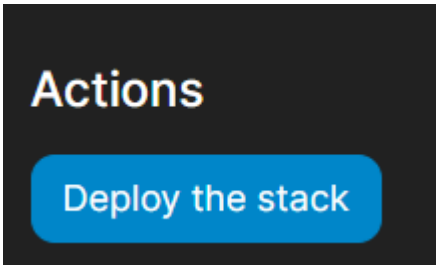


Job 9 version portainer

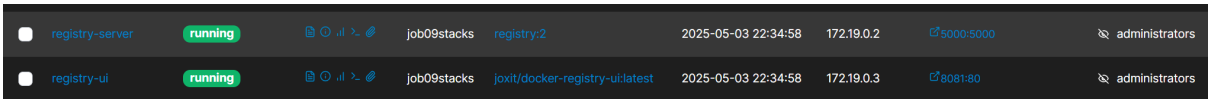
On va dans stacks puis dans web editor on écrit notre docker-compose.yml

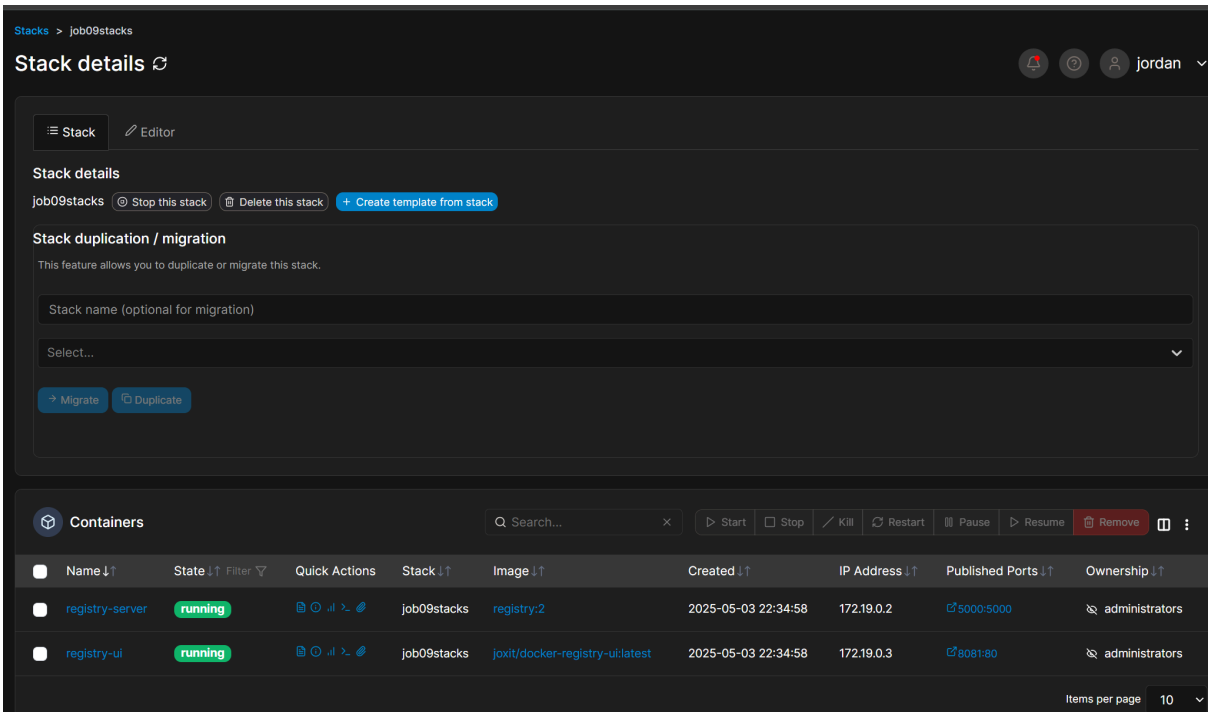
```yaml
version: '3.3'

services:
  registry:
    image: registry:2
    container_name: registry-server
    ports:
      - "5000:5000"
    volumes:
      - ./data:/var/lib/registry
    environment:
      REGISTRY_STORAGE_DELETE_ENABLED: "true"
      REGISTRY_HTTP_HEADERS_Access-Control-Allow-Origin: '["http://192.168.76.129:8081"]'
      REGISTRY_HTTP_HEADERS_Access-Control-Allow-Methods: '["GET", "OPTIONS", "PUT", "DELETE"]'
      REGISTRY_HTTP_HEADERS_Access-Control-Allow-Headers: '["Authorization", "Accept"]'

  registry-ui:
    image: joxit/docker-registry-ui:latest
    container_name: registry-ui
    ports:
      - "8081:80"
    depends_on:
      - registry
    environment:
      REGISTRY_TITLE: "Registry Local Privé"
      SINGLE_REGISTRY: "true"
      REGISTRY_NAME: "LocalRegistry"
      DELETE_IMAGES: "true"
      REGISTRY_SECURED: "false"
      REGISTRY_URL: "http://192.168.76.129:5000"
```

On déploie les containers

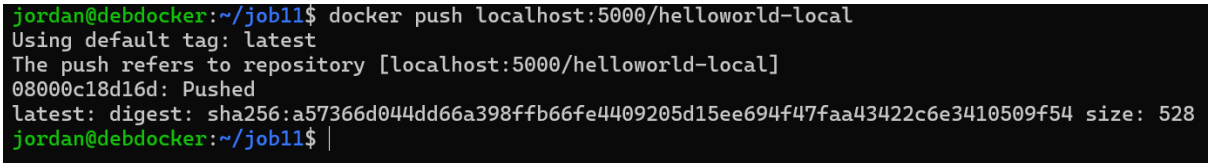


les deux containers ont bien été déployés

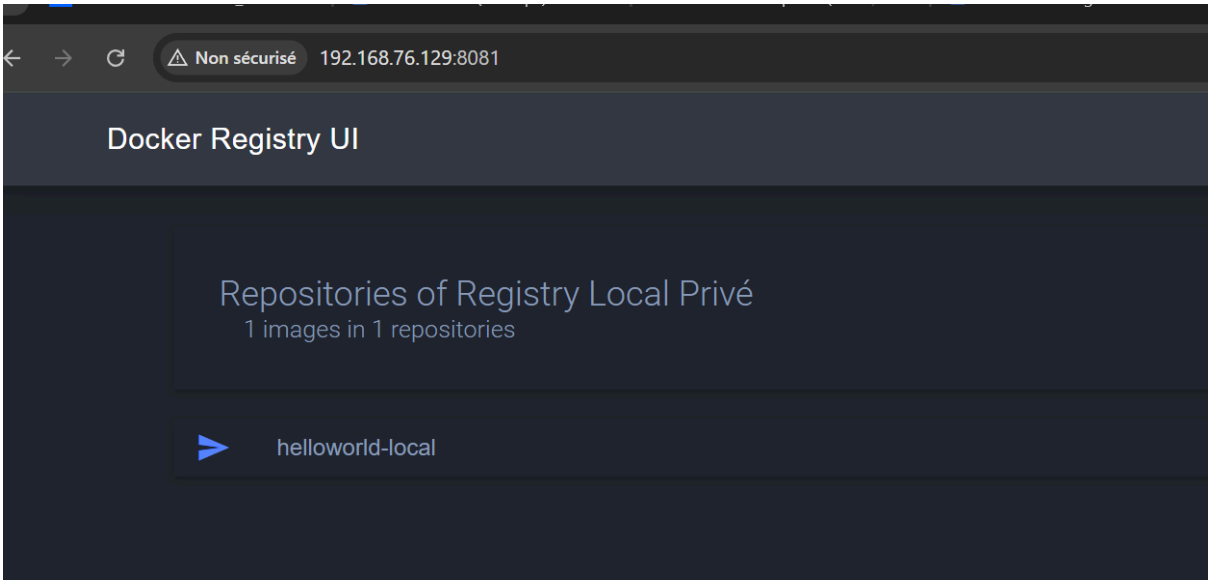




on tag puis on push une image en CLI pour voir si notre registry fonctionne



test



Pour aller plus loin

On complique un peu le système, il va falloir faire l'équivalent de XAMPP
Avec un fichier Dockerfile et un fichier yml faire :
➔ un serveur nginx (ou apache) avec PHP
➔ un serveur MariaDB/MySQL (avec phpMyAdmin)
➔ un serveur FTP
➔ un volume pour stocker l'ensemble des données communes aux différents serveurs
Tester le système.

Fichier docker-compose.yml avec serveur web, mariadb, phpmyadmin et ftp

```
 GNU nano 7.2                                                  docker-compose.yml
version: '3.3'

services:
  web:
    build: .
    container_name: web_php
    volumes:
      - ./www:/var/www/html
    ports:
      - "9500:80"
    depends_on:
      - db

  db:
    image: mariadb
    container_name: mariadb
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: xampp_db
    volumes:
      - db_data:/var/lib/mysql

  phpmyadmin:
    image: phpmyadmin
    container_name: phpmyadmin
    restart: always
    ports:
      - "9501:80"
    environment:
      PMA_HOST: db

  ftp:
    image: fauria/vsftpd
    container_name: ftp_server
    restart: always
    ports:
      - "9510:21"
      - "21100-21110:21100-21110"
    volumes:
      - ./www:/home/vsftpd
    environment:
```

```
    environment:
      - FTP_USER=user
      - FTP_PASS=pass
      - PASV_ENABLE=YES
      - PASV_MIN_PORT=21100
      - PASV_MAX_PORT=21110
      - PASV_ADDRESS=192.168.76.129

volumes:
  db_data:
```

Dockerfile

```
 GNU nano 7.2                                                Dockerfile
FROM php:8.1-apache

RUN a2enmod rewrite

COPY www/ /var/www/html/

EXPOSE 80
```

on fait un dossier www puis on crée un fichier index.php

```
jordan@debdocker:~/projet-xampp$ cd www
jordan@debdocker:~/projet-xampp/www$ ls
index.php  user
```

```
 GNU nano 7.2                                                index.php
<?php
phpinfo();
?>
```

Création de l’image et lancement des containers

Docker-compose build –no-cache

```
jordan@debdocker:~/projet-xampp$ docker-compose build --no-cache
db uses an image, skipping
phpmyadmin uses an image, skipping
ftp uses an image, skipping
Building web
Sending build context to Docker daemon  6.144kB
Step 1/5 : FROM php:8.1-fpm
8.1-fpm: Pulling from library/php
254e724d7786: Already exists
c65ab0511c0c: Pull complete
396ae0811dc1: Pull complete
08d68830cf18: Pull complete
dbd45828ee1d: Pull complete
66e5f13148ec: Pull complete
6f9c6ce5a07a: Pull complete
49fc91a0979c: Pull complete
f1f4ab0a9742: Pull complete
4f4fb700ef54: Pull complete
5280579f701f: Pull complete
Digest: sha256:10ea2d74309746fe72ded79a994dd28e3f6fcdc3aedc411b1d7ef2094006ae34
Status: Downloaded newer image for php:8.1-fpm
 ---> d776f258a86d
Step 2/5 : RUN apt-get update && apt-get install -y     nginx     supervisor     && apt-get clean
 ---> Running in ea8fc7fdff09
```

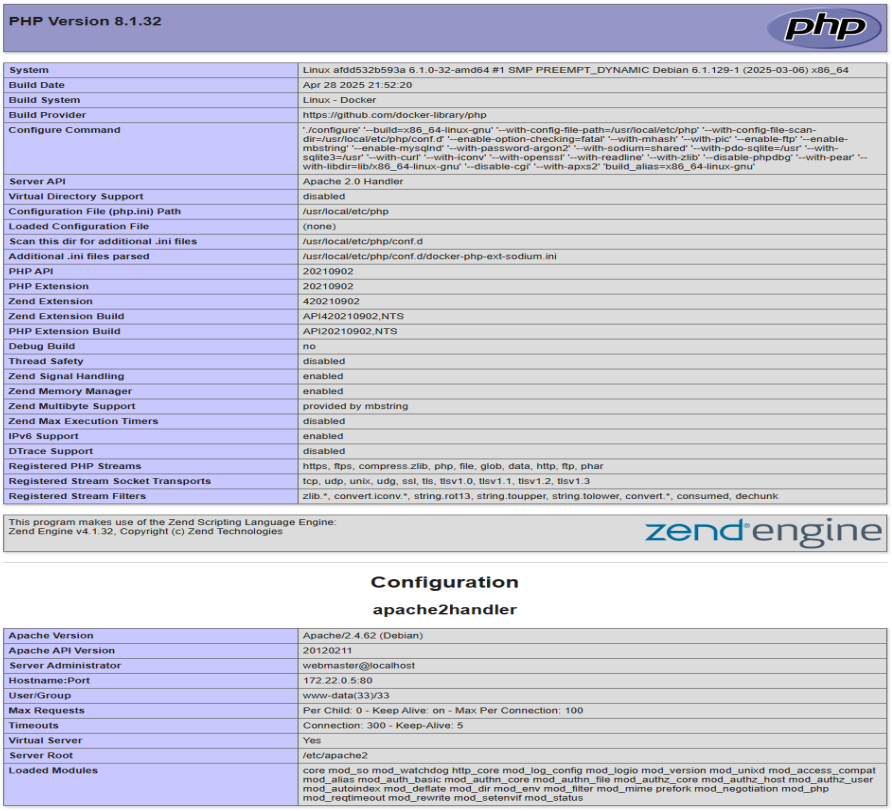
docker-compose up-d

```
jordan@debdocker:~/projet-xampp$ docker-compose up -d
Creating network "projet-xampp_default" with the default driver
Creating volume "projet-xampp_db_data" with default driver
Pulling db (mariadb:)...
latest: Pulling from library/mariadb
2726e237d1a3: Pull complete
0b86886c6aaa: Download complete
2b221cf763a8: Download complete
5e4180757702: Download complete
43028b9f5f8e: Download complete
bbef7eafa75b: Downloading [==========>                                        ]  13.95MB/69.84MB
ab732728101f: Waiting
0c9f57c1bb30: Waiting
```

```
Creating ftp_server ... done
Creating mariadb    ... done
Creating phpmyadmin ... done
Creating web_php    ... done
jordan@debdocker:~/projet-xampp
```

Test

Apache fonctionnel



PHP Version 8.1.32

System	Linux afdd532b593a 6.1.0-32-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.129-1 (2025-03-06) x86_64
Build Date	Apr 28 2025 21:52:20
Build System	Linux - Docker
Build Provider	https://github.com/docker-library/php
Configure Command	'./configure' '--build=x86_64-linux-gnu' '--with-config-file-path=/usr/local/etc/php' '--with-config-file-scan-dir=/usr/local/etc/php/conf.d' '--enable-option-checking=fatal' '--with-mhash' '--with-pic' '--enable-ftp' '--enable-mbstring' '--enable-mysqlnd' '--with-password-argon2' '--with-sodium=shared' '--with-pdo-sqlite=/usr' '--with-sqlite3=/usr' '--with-curl' '--with-iconv' '--with-openssl' '--with-readline' '--with-zlib' '--disable-phpdbg' '--with-pear' '--with-libdir=lib/x86_64-linux-gnu' '--disable-cgi' '--with-apxs2' 'build_alias=x86_64-linux-gnu'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/etc/php
Loaded Configuration File	(none)
Scan this dir for additional .ini files	/usr/local/etc/php/conf.d
Additional .ini files parsed	/usr/local/etc/php/conf.d/docker-php-ext-sodium.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
Zend Max Execution Timers	disabled
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk

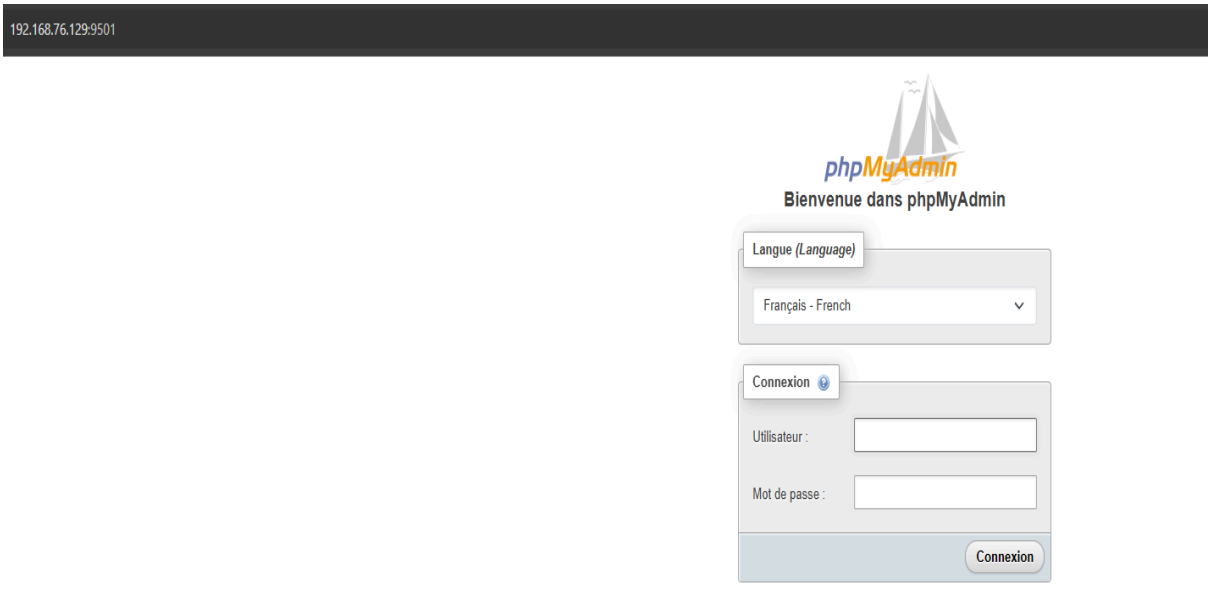
This program makes use of the Zend Scripting Language Engine:
Zend Engine v4.1.32, Copyright (c) Zend Technologies

Configuration

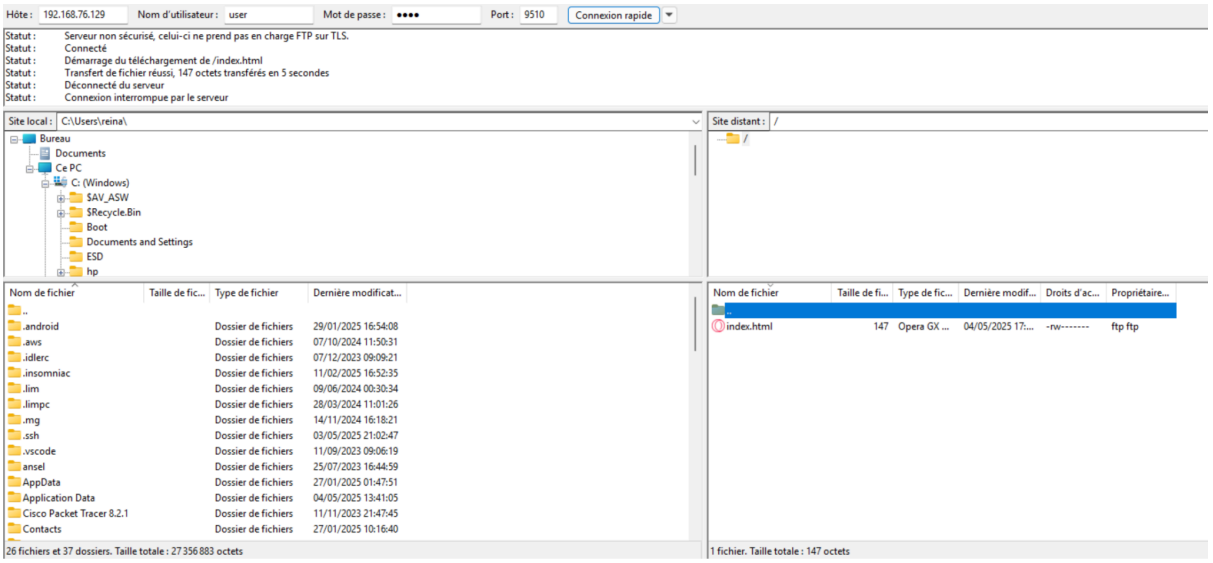
apache2handler

Apache Version	Apache/2.4.62 (Debian)
Apache API Version	20120211
Server Administrator	webmaster@localhost
Hostname:Port	172.22.0.5:80
User/Group	www-data(33)/33
Max Requests	Per Child: 0 - Keep Alive: on - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 5
Virtual Server	Yes
Server Root	/etc/apache2
Loaded Modules	core mod_so mod_watchdog http_core mod_log_config mod_logio mod_version mod_unixd mod_access_compat mod_alias mod_auth_basic mod_authn_core mod_authn_file mod_authz_core mod_authz_host mod_authz_user mod_autoindex mod_deflate mod_dir mod_env mod_filter mod_mime prefork mod_negotiation mod_php mod_reqtimeout mod_rewrite mod_setenvif mod_status

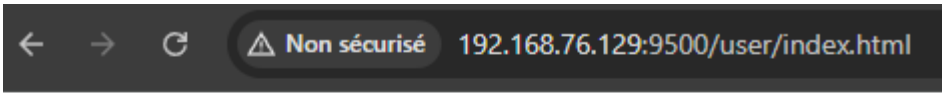
PhpMyAdmin fonctionnel



Test de transfert de fichier ftp



on peut voir que cela a fonctionné vu que nous sommes sur



Jordan / Reinaldo ðŸ˜Ž