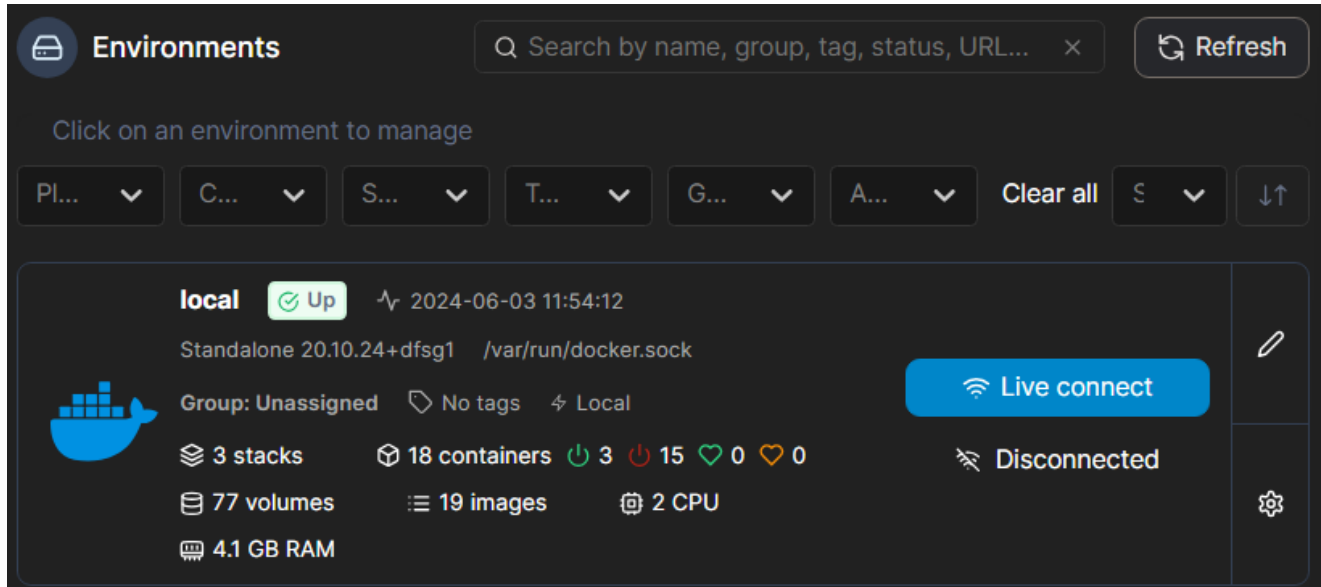


une fois notre volume créer, on va lancer notre conteneur portainer :

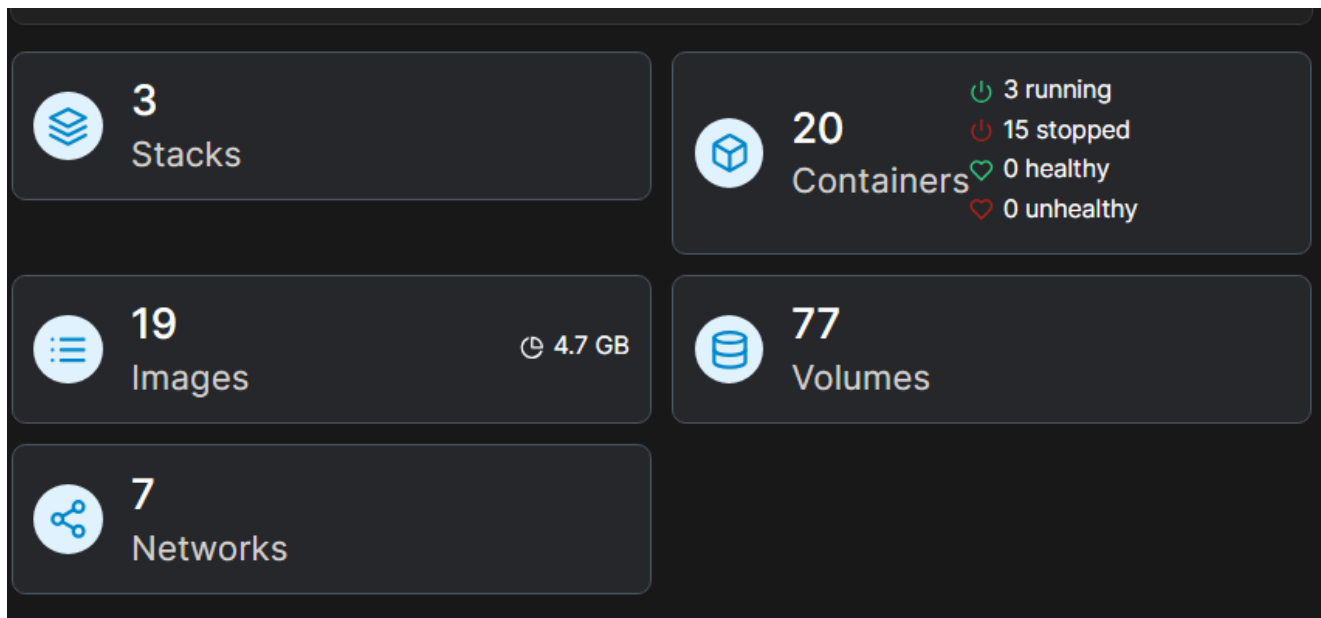
```
docker run -d -p 9000:9000 --name portainer --restart always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce
```

Job 03 :

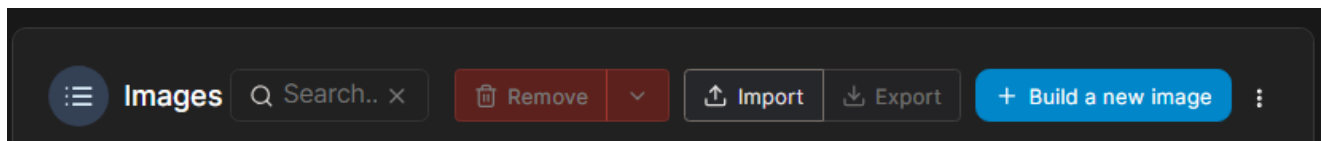
Pour créer une image avec Portainer, il faut se rendre dans le dash board, et utilisier les donnée local, une fois dans les données local, on peut créer une image :



The screenshot shows the Portainer 'Environments' dashboard. At the top, there's a search bar and a 'Refresh' button. Below, a list of environments is shown. The selected environment is 'local', which is 'Up' and was created on '2024-06-03 11:54:12'. It is a 'Standalone' environment with the host path '/var/run/docker.sock'. The environment details include: Group: Unassigned, No tags, Local. It shows 3 stacks, 18 containers (3 running, 15 stopped, 0 healthy, 0 unhealthy), 77 volumes, 19 images, and 2 CPU. The RAM usage is 4.1 GB. A 'Live connect' button is visible, and the status is 'Disconnected'.



The screenshot shows the Portainer resource overview dashboard. It displays five main resource categories: 3 Stacks, 20 Containers (3 running, 15 stopped, 0 healthy, 0 unhealthy), 19 Images (4.7 GB), 77 Volumes, and 7 Networks.



The screenshot shows the Portainer 'Images' section. It includes a search bar, a 'Remove' button, an 'Import' button, an 'Export' button, and a '+ Build a new image' button.

```
Web editor

You can get more information about Dockerfile format in the official documentation.

Define or paste the content of your Dockerfile here

1  # Utilisez l'image Debian minimale comme image de base
2  FROM debian:latest
3
4  # Ajoutez une étiquette pour indiquer l'auteur du Dockerfile
5  LABEL maintainer="votre_nom"
6
7  # Installez les packages nécessaires pour exécuter un serveur web simple (ici, on utilise netcat)
8  RUN apt-get update && apt-get install -y
9
10
11 CMD ["echo", "helloworld5"]
12
```

Une fois notre image créée, on peut se rendre sur notre machine Debian et voir que l'image a bien été créée :

```
raph@DebianXDocker:~$ images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
xampp_ftp            latest       017764100123      17 hours ago     394MB
xampp_php            latest       eae88b5c0d64      17 hours ago     443MB
xampp_nginx          latest       ec7ce2478230      17 hours ago     188MB
job8-portainer       latest       a7c7701c591a      18 hours ago     188MB
ssh-portainer        latest       c2704a0d297c      19 hours ago     185MB
hello-word-portainer latest       69067b560cf0      19 hours ago     136MB
mariadb              latest       f44a87143f65      4 days ago       412MB
ssh                  latest       eefafcc44002      2 weeks ago      185MB
debian               latest       5027089adc4c      2 weeks ago      117MB
helloworld2          latest       2068bc6b0013      2 weeks ago      136MB
docker               latest       1feaad25659a      3 weeks ago      365MB
nginx                latest       e784f4560448      4 weeks ago      188MB
portainer/portainer-ce latest       a3f85c245ec3      6 weeks ago      293MB
delfer/alpine-ftp-server latest       7255f93ca5a4      4 months ago     8.78MB
registry             2           d6b2c32a0f14      8 months ago     25.4MB
phpmyadmin/phpmyadmin latest       933569f3a9f6      10 months ago    562MB
hello-world          latest       d2c94e258dcb      13 months ago    13.3kB
fauria/vsftpd        latest       9bfb39139661      16 months ago    394MB
php                  7.4-fpm     38f2b691dcb8      18 months ago    443MB
raph@DebianXDocker:~$ S
```

on peut aussi lancer un conteneur pour vérifier que notre image fonctionne correctement :

```
raph@DebianXDocker:~$ sudo docker run hello-word-portainer
helloworld
```

Job 04

On va maintenant suivre les mêmes étapes que précédemment, pour créer une image utilisant Debian, avec un service SSH :

```

FROM debian:latest

RUN apt-get update && apt-get install -y openssh-server

RUN mkdir /var/run/ssh

# Set root password for SSH access (change 'your_password' to your desired password)

RUN echo 'root:root123' | chpasswd

0 RUN sed -i 's/#Port 22/Port 8000/' /etc/ssh/sshd_config
1
2 RUN sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config
3
4 RUN sed 's@session\s*required\s*pam_loginuid.so@session optional pam_loginuid.so@g' -i /etc/pam.d/ssh
5
6 EXPOSE 8000
7
8 CMD ["/usr/sbin/sshd", "-D"]

```

Une fois notre image correctement créer, on peut lancer notre conteneur avec cette commande :

```
sudo docker run -d -p 8000:8000 --name ssh-portainer ssh-portainer
```

on veut voir que notre service est bien up et qu'il fonctionne correctement sur le port voulu

```

raph@DebianX:~/ssh2$ sudo docker run -d -p 8000:8000 --name ssh-portainer ssh-portainer
5ddb76f0ec080a0f6e1fa9115a0d7a8fd3485c34a02d2d0d5fe09d3c8baa345f
raph@DebianX:~/ssh2$ ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
5ddb76f0ec08   ssh-portainer                       "/usr/sbin/sshd -D"     4 seconds ago Up 3 seconds  0.0.0.0:8000->8000/tcp, :::8000->8000/tcp  ssh-portainer
3b06b4325167   delier/alpine-ftp-server:latest     "/sbin/tini -- /bin/..." 18 hours ago  Up 9 minutes  8000/tcp, 9443/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp  job7-portainer-ftp-1
30e479360c3b   delier/alpine-ftp-server:latest     "/sbin/tini -- /bin/..." 18 hours ago  Up 9 minutes  8000/tcp, 9443/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp  job7-ftp_1
bac47525834c   portainer/portainer-ce              "/portainer"            19 hours ago  Up 9 minutes  9000/tcp, 9443/tcp, 0.0.0.0:9000->9000/tcp, :::9000->9000/tcp  portainer
raph@DebianX:~/ssh2$

```

JOB 07

Pour créer un volumes qui interrégisse entre eux, sur portainer, on va se rendre dans stack :

3
Stacks

21
Containers

4 running
15 stopped

0 healthy
0 unhealthy

19
Images

4.7 GB

77
Volumes

7
Networks

Une fois Dans stack, on fait add a new stack, et on rentre la configuration de notre fichier YML :

Web editor

You can get more information about Compose file format in the [official documentation](#).

Define or paste the content of your docker compose file here

```
1  #la version utiliser
2  version: '3'
3
4  # définit les différent service que docker-compose va gérer
5  services:
6
7  #-----NGINX-----
8  #on définit notre premier service nginx
9  nginx:
10 #on utilise la dernière image de nginx
11 image: nginx:latest
12 #on spécifie les ports que l'on va utiliser
13 ports:
14 - "8080:80"
15 #on définit notre volume ou l'on va stocker notre fichier html
16 volumes:
17 - shared-data:/usr/share/nginx/html
18
19
20 #-----FTP-----
21 #on passe maintenant à la configuration de notre service FTP
22 ftp:
```

voici la configuration complète :

```
#la version utiliser
version: '3'

# définit les différent service que docker-compose va gérer
services:

#-----NGINX-----
#on définit notre premier service nginx
nginx:
#on utilise la dernière image de nginx
image: nginx:latest
#on spécifie les ports que l'on va utiliser
ports:
- "8080:80"
#on définit notre volume ou l'on va stocker notre fichier html
volumes:
- shared-data:/usr/share/nginx/html

#-----FTP-----
#on passe maintenant à la configuration de notre service FTP
ftp:
#on installer l'image delfer/alpine-frp-server
image: 'delfer/alpine-ftp-server:latest'
#on passe en mode network host
network_mode: host
#on ne définit pas de port car cela pose de problème avec l'image qu'on utilise
# ports:
# - "8765:21"
#on met en place notre volume partagé sur notre conteneur FTP
volumes:
- shared-data:/ftp/raph
#on définit notre utilisateur et son mot de pass
environment:
- USERS=one|123456789

#cette options permet de restartt le service si jamais il se stop quand notre conteneur est up
```

```
restart: always
```

```
#on définit le volumes partagé entre nos deux conteneur
```

```
volumes:
```

```
shared-data:
```

Une fois notre fichier YML, correctement configuré, on peut lancer notre service directement via Portainer :

Containers									
Q Search...									
▶ Start □ Stop ⌛ Kill ↺ Restart Pause ▷ Resume 🗑 Remove ⌵									
✓ Name ↓↑	State ↓↑ Filter ▾	Quick Actions	Stack ↓↑	Image ↓↑	Created ↓↑	IP Address ↓↑	Published Ports ↓↑	Ownership ↓↑	
✓ job7_ftp_1	exited	🔍 ⌛	job7	delfer/alpine-ftp-server:latest	2024-06-02 18:02:20	-	-	🔑 administrators	
✓ job7_nginx_1	exited	🔍 ⌛	job7	nginx:latest	2024-06-02 17:46:44	-	-	🔑 administrators	
2 item(s) selected							Items per page	10	▼

et on clique sur start

JOB 08

On va maintenant créer une image NGINX sur Portainer :

```
1  # Utiliser l'image nginx de base
2  FROM nginx
3
4  # Copier un fichier de configuration personnalisé pour nginx
5  COPY index.html /usr/share/nginx/html
6
7  # Exposer le port 80 pour le trafic HTTP
8  EXPOSE 80
9
10 # on lance notre service NGINX
11 CMD ["nginx", "-g", "daemon off;"]
```

une fois notre image créée, on peut la lancer et vérifier qu'elle fonctionne directement, soit sur notre machine soit directement sur Portainer.