

# Job11

## Job 11

---

Découverte de **portainer** .

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

Pour installer Portainer:

mkdir portainer

puis

```
sudo docker run -d -p 8000:8000 -p 9000:9000 --restart unless-stopped --name="portainer" -v /var/run/docker.sock:/var/run/docker.sock -v /portainer:/data portainer/portainer-ce
```

```
debian@debian:~/portainer$ sudo docker run -d -p 8000:8000 -p 9000:9000 --restart unless-stopped --name="portainer" -v /var/run/docker.sock:/var/run/docker.sock -v /portainer:/data portainer/portainer-ce
[sudo] Mot de passe de debian :
Unable to find image 'portainer/portainer-ce:latest' locally
latest: Pulling from portainer/portainer-ce
57654d40e0a5: Pull complete
1f476acfabd6: Pull complete
5171176db7f2: Pull complete
52e9438966a5: Pull complete
43d4775415ac: Pull complete
c1cad9f5200f: Pull complete
a5e2b359b78b: Pull complete
eb172612bcbb: Pull complete
6be7b2acffb5: Pull complete
391dff0fb880: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:4a1ceadd7f7898d9190ee0a6d22234c4323aefd80e796e84f5e57127f74370f1
Status: Downloaded newer image for portainer/portainer-ce:latest
62f84ab219a337db9fdb37b5deab8ad1200321255e9e76c922640bc09132dade
debian@debian:~/portainer$
```

On y accède à : <http://192.168.107.131:9000/>



!!

Pour ne pas refaire tous les jobs, je fais seulement refaire

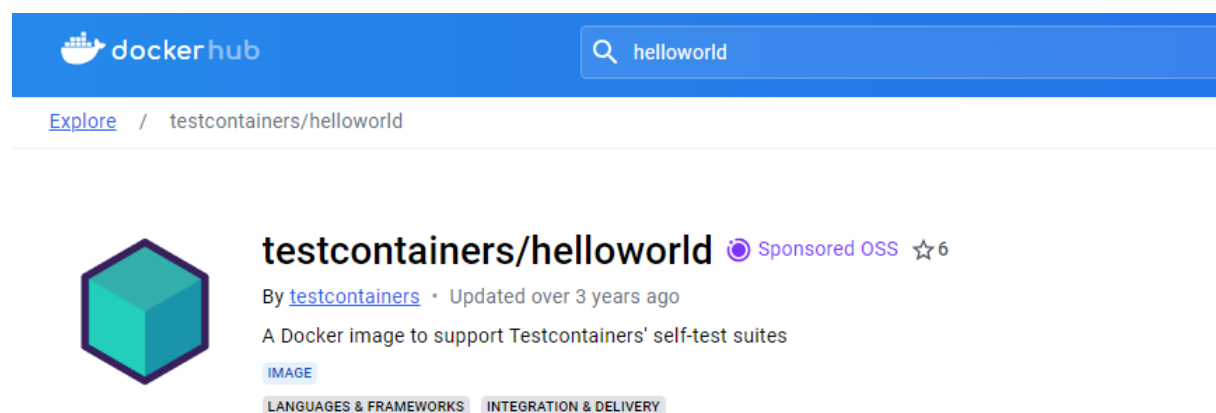
- Le Job2, qui me permet d'utiliser une image du dockerhub
- Le Job3, qui me permet de réaliser un dockerfile (build une image)
- Le Job7, qui me permet de réaliser un docker-compose (build une stack)

!!

## Job 02

Tester l'installation de docker avec le conteneur « helloworld » et se familiariser avec les commandes.

⇒ avec une image du dockerhub



The screenshot shows the Docker Hub interface. At the top is a blue header with the Docker Hub logo and a search bar containing 'helloworld'. Below the header, the breadcrumb 'Explore / testcontainers/helloworld' is visible. The main content area displays the 'testcontainers/helloworld' repository. It features a teal cube icon, the repository name, a 'Sponsored OSS' badge, and a star count of 6. Below this, it says 'By testcontainers · Updated over 3 years ago' and 'A Docker image to support Testcontainers' self-test suites'. There are two tags: 'IMAGE' and 'INTEGRATION & DELIVERY'.

## Create container

Name

e.g. myContainer

### Image configuration


Registry


Docker Hub (anonymous)

Image\*


docker.io

testcontainers/helloworld


 Advanced mode

Always pull the image 




 You are currently using an anonymous account to pull images from DockerHub and will be limited to 100 pulls every 6 hours. You can configure Docker


### Webhooks

Create a container webhook 





 Business Feature

### Network ports configuration


Publish all exposed network ports to random  
host ports 



Manual network port publishing 

 publish a new network port

### Access control

Enable access control 




#### Administrators

I want to restrict the management of this resource to administrators only



### Actions

Auto remove 



Deploy the container

## Container logs



### Log viewer settings

Auto-refresh logs ?



Wrap lines



Display timestamps



Fetch

All logs



Search

Filter...

Lines

100

Actions

Download logs

Copy

Copy selected lines

Unselect

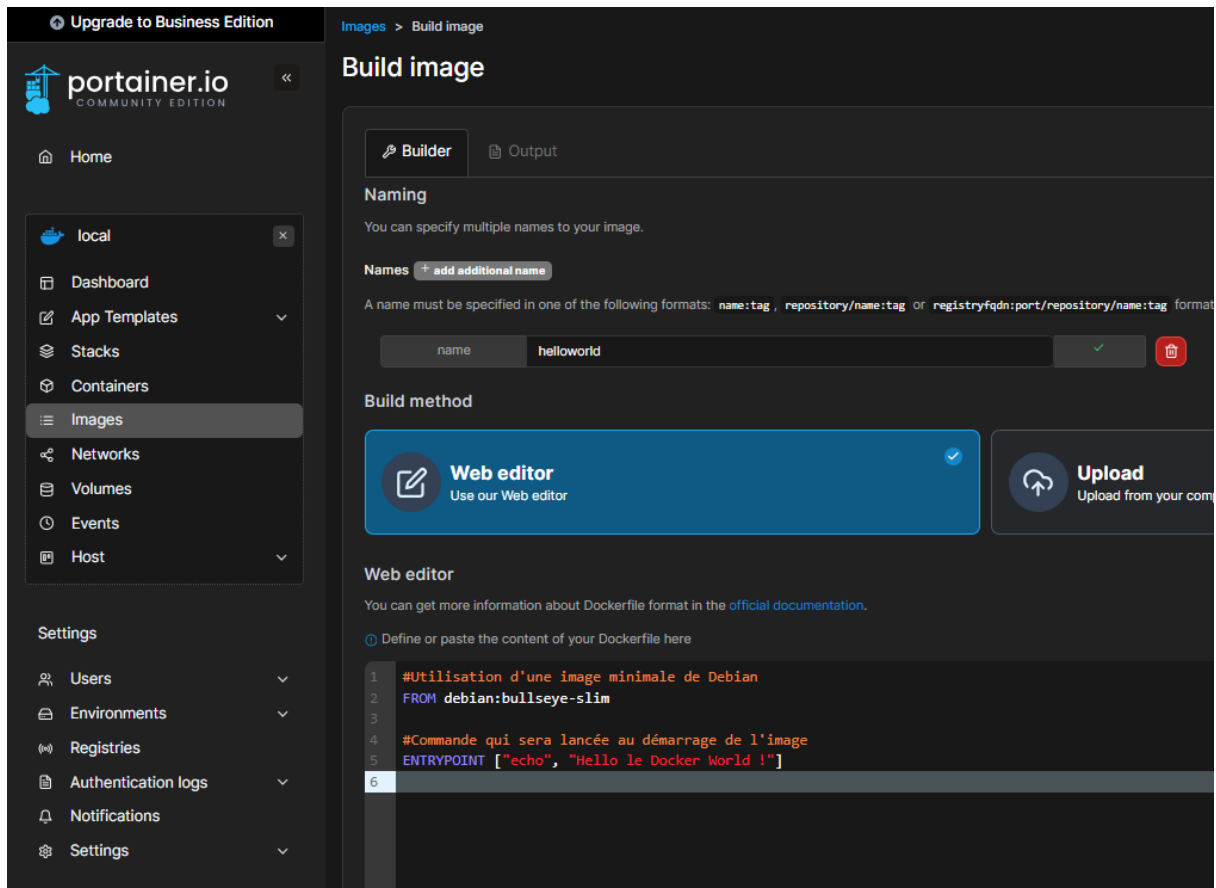
```
2024/06/02 11:39:01 DELAY_START_MSEC: 0
2024/06/02 11:39:01 Sleeping for 0 ms
2024/06/02 11:39:01 Starting server on port 8080
2024/06/02 11:39:01 Sleeping for 0 ms
2024/06/02 11:39:01 Starting server on port 8081
2024/06/02 11:39:01 Ready, listening on 8080 and 8081
```

## Job 03

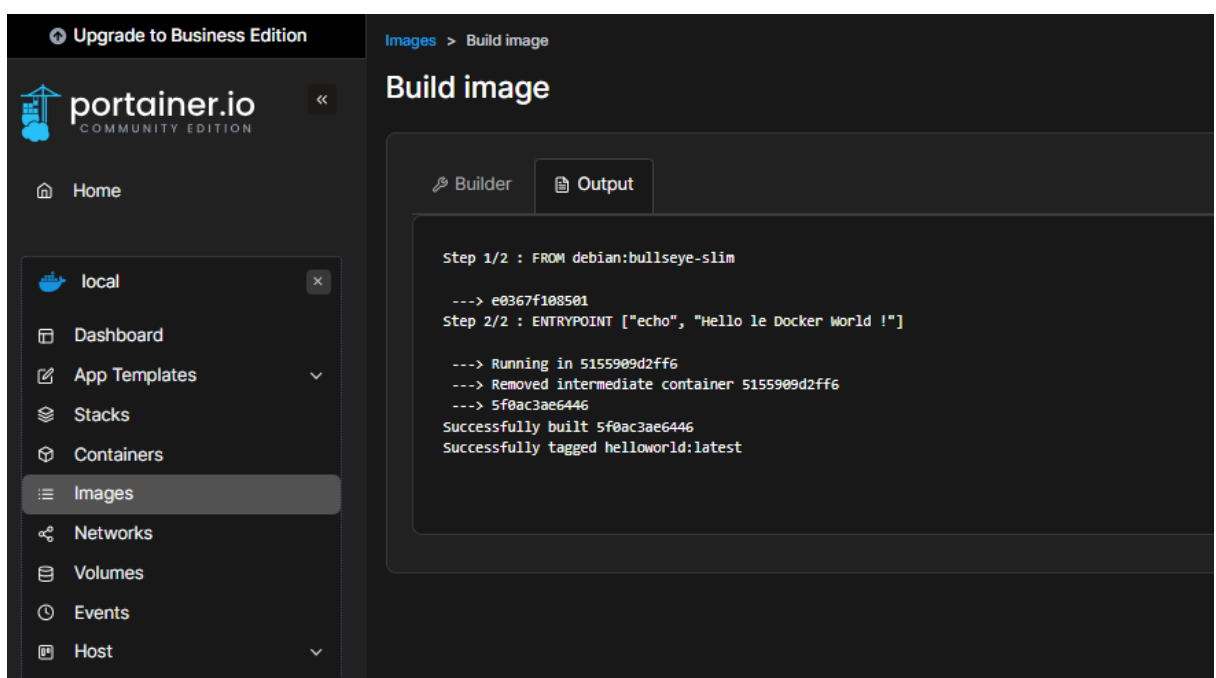
Utilisation de « Dockerfile » pour recréer le conteneur « helloworld » depuis une image Debian minimum.

⇒ comme un dockerfile

1ère étape : build une image




Voilà qui est fait :



2ème étape : on crée un conteneur

(en advanced mode)

 Advanced mode


Containers > Add container


## Create container

**Name**



**Image configuration**  
When using advanced mode, image and repository **must be** publicly available.

**Image \***

 Simple mode

**Always pull the image** 


**Webhooks**

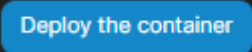
**Create a container webhook**   Business Feature

**Network ports configuration**

3ème étape : on déploie le conteneur

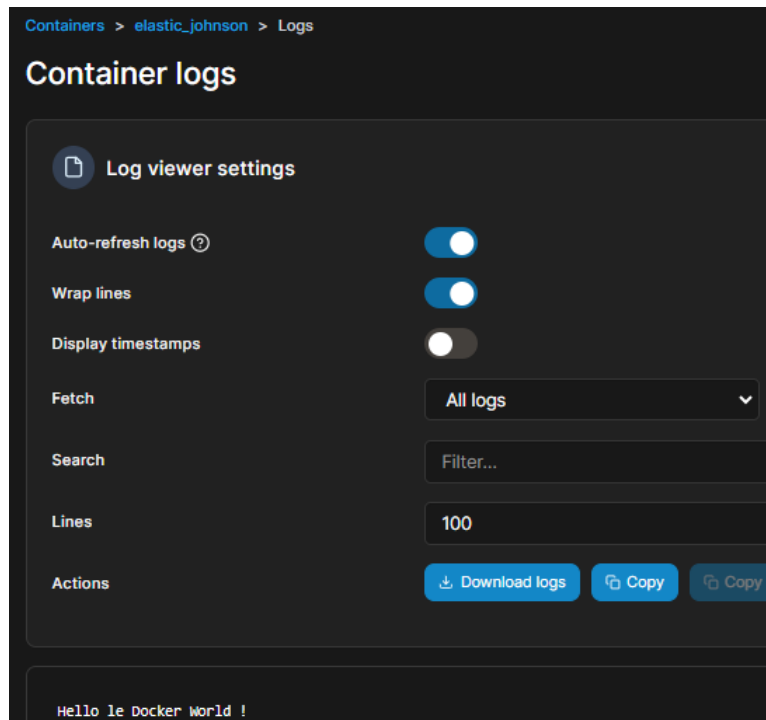
**Actions**

**Auto remove** 



4ème étape : check : on le start puis on va voir les logs

Containers					
<div> <div>Q Search...</div> <div>Start</div> </div>					
Name ↓↑	State ↓↑ Filter ▾	Quick Actions	Stack ↓↑	Image ↓↑	Created ↓↑
elastic_johnson	exited	<div> <div></div> <div></div> </div>	-	helloworld:latest	2024-06-02 12:57:53



## 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.


⇒ comme un docker-compose










On crée le volume "job7"

Volumes

## Volume list



 **Volumes**

<input type="checkbox"/> Name   Filter 	Stack  	Driver  
<input type="checkbox"/> job7 <span>Unused</span>	-	local

On crée une stack "nginx-ftp"  
(en indentant bien)


Stacks > nginx-ftp

## Stack details

 Stack  Editor

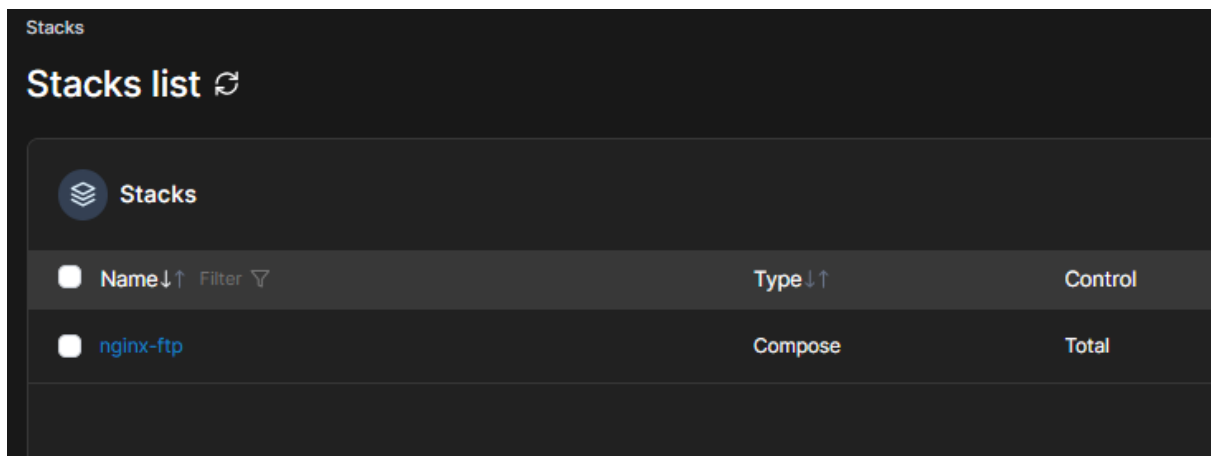
This stack will be deployed using `docker compose`.

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

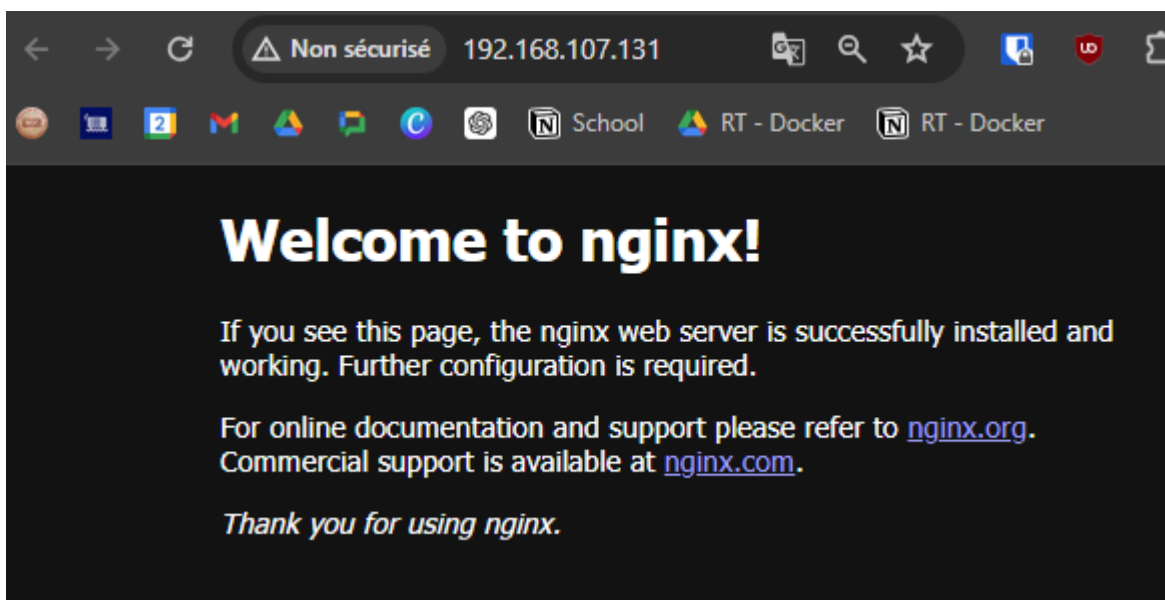
 Define or paste the content of your docker compose file here

```
1 version: '3.8'
2 services:
3   nginx:
4     image: nginx
5     ports:
6       - '80:80'
7     volumes:
8       - 'job7:/usr/share/nginx/html'
9   ftp:
10    image: fauria/vsftpd
11    ports:
12      - '2020:20'
13      - '2121:21'
14    environment:
15      FTP_USER: user
16      FTP_PASS: password
17      PASV_ADDRESS: 192.168.107.131
18    volumes:
19      - 'job7:/home/vsftpd/user'
20    restart: always
21 volumes:
22   job7:
```

Puis on la déploie



On accède bien à la page web, donc serveur nginx fonctionnel :



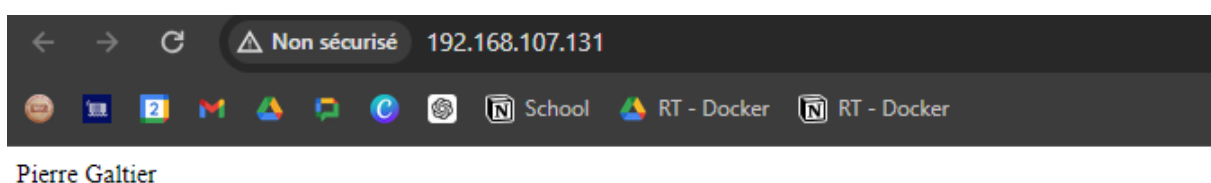
Transfert avec FileZilla fonction toujours pas donc j'ouvre une session ftp depuis le terminal de mon poste et j'envoie la page index.html :

```
Invite de commandes - ftp
Microsoft Windows [version 10.0.22631.3593]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\pgalt>ftp
ftp> open 192.168.107.131 2121
Connecté à 192.168.107.131.
220 (vsFTPd 3.0.2)
200 Always in UTF8 mode.
Utilisateur (192.168.107.131:(none)) : user
331 Please specify the password.
Mot de passe :

230 Login successful.
ftp> cd /usr/share/nginx/html
550 Failed to change directory.
ftp> put C:/Users/pgalt/Desktop/index.html
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
ftp : 231 octets envoyés en 0.00 secondes à 231.00 Ko/s.
ftp> |
```

On actualise la page :



⇒ ça fonctionne