DOCKER

Qu’est ce qu’est Docker ?
Docker est une plate-forme open source qui permet d'exécuter des applications dans des environnements isolés du système qui sont appelés conteneurs. Comparé à des applications ordinaires les conteneurs sont très légers en taille et en ressource consommé. Les conteneurs sont également portables, ce qui veut dire qu’on peut les déplacer de leurs environnement sans rencontrer de problème de compatibilité. Cela permet également de déployer ce même conteneur sur différentes machines.
Il utilise une architecture client-serveur pour gérer les conteneurs, offrant ainsi un moyen efficace de gérer l'ensemble du cycle de vie des applications, de leur développement à leur déploiement en production.

Pour résumer : Docker simplifie le déploiement d'applications en les isolant dans des conteneurs, ce qui facilite la gestion, la portabilité et la scalabilité des applications.

Cas d’utilisation :
Déploiement d'applications web, Isolation des environnements de développement, Développement cross-plateforme, Optimisation des ressources, Migration vers le cloud ….

FICHE DE PROCÉDURE

Prise en main Docker.

Étape 1 – Installation de docker

a- Initialisation des répertoires de docker

Add Docker's official GPG key:

$ apt-get update

$ apt-get install ca-certificates curl gnupg

$ install -m 0755 -d /etc/apt/keyrings

$ curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o /etc/apt/keyrings/docker.gpg

$ chmod a+r /etc/apt/keyrings/docker.gpg

Add the repository to Apt sources:

echo \

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

  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \

   tee /etc/apt/sources.list.d/docker.list > /dev/null

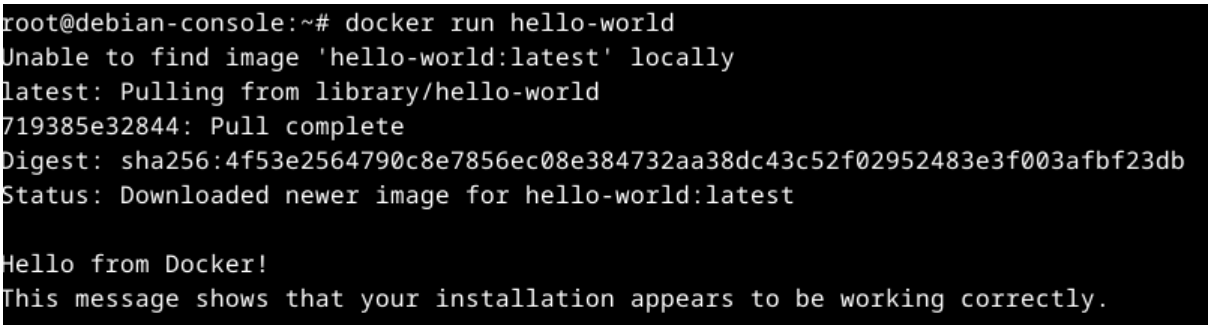
 apt-get update

b- Installation du package

$ apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

c- vérification de l'installation de docker

$ docker run hello-world



1) **Récupération de l'image du service NGINX**

$ docker pull nginx

```
root@debian-console:~# docker run --name some-nginx -v /some/content:/usr/share/nginx/html:ro -d nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
360eba32fa65: Pull complete
c5903f3678a7: Pull complete
27e923fb52d3: Pull complete
72de7d1ce3a4: Pull complete
94f34d60e454: Pull complete
e42dcfe1730b: Pull complete
907d1bb4e931: Pull complete
Digest: sha256:275c11c8996bacda0ca35e12f42e9696e80032a2b9926b79f0ac58fdecf5de2f
Status: Downloaded newer image for nginx:latest
8d6f0038333cc98f67ebe3c24b2a330d751424172773bca8582018653db28319
root@debian-console:~#
```

$

Vérification de la liste des images téléchargées : $ docker images

```
root@debian-console:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    61395b4c586d   6 days ago     187MB
nginx         <none>    f5a6b296b8a2   2 weeks ago    187MB
hello-world   latest    9c7a54a9a43c   4 months ago   13.3kB
root@debian-console:~#
```

3) **Conteneur NGINX**

Lancement d'un conteneur NGINX

Lancement du conteneur NGINX en arrière-plan avec le partage du port :

$ docker run -d -p 8080:80 nginx

```
root@debian-console:~# docker run -d -p 8080:80 nginx
7314a8d51272bf35051b7c2690eca2261d1ed18d2bc16d300d38af25b12a360d
```

Récupération de la liste des conteneurs créés

$ docker ps -a

```
root@debian-console:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS
PORTS                                     NAMES
7314a8d51272   nginx          "/docker-entrypoint.…"   17 seconds ago       Up 15 seconds
0.0.0.0:8080->80/tcp, :::8080->80/tcp     gifted_shaw
2e2f5e8a6326   hello-world    "/hello"                 About a minute ago   Exited (0) About a minute ago
                                          laughing_elbakyan
3c3dc8998a78   hello-world    "/hello"                 6 days ago           Exited (0) 6 days ago
                                          condescending_clarke
8d6f0038333c   f5a6b296b8a2   "/docker-entrypoint.…"   6 days ago           Exited (255) 4 minutes ago
80/tcp                                    some-nginx
```

Vérification du statut du conteneur NGINX en cours d'exécution

$ docker ps

```
root@debian-console:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS             PORTS                                   NAMES
7314a8d51272   nginx     "/docker-entrypoint.…"   About a minute ago  Up About a minute  0.0.0.0:8080->80/tcp, :::8080->80/tcp   gifted_shaw
```

Exécution de la commande uname -a

$ docker exec -it *7314a8d51272* uname -a

7314a8d51272 = ID du conteneur

Arrêt du conteneur NGINX

Pour l'arrêter j'utilise la commande docker stop suivie de l'id du conteneur ou son nom.

$ docker stop *7314a8d51272*

```
root@debian-console:~# docker stop 7314a8d51272
7314a8d51272
```

Vérification du statut

$ docker ps

```
root@debian-console:~# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@debian-console:~#
```

Démarrage du conteneur NGINX

Pour le démarrer j'utilise la commande docker start suivie de l'id du conteneur ou son nom.

$ docker start *7314a8d51272*

```
root@debian-console:~# docker start 7314a8d51272
7314a8d51272
```

Suppression du conteneur NGINX

$ docker rm *7314a8d51272*

```
root@debian-console:~# docker rm 7314a8d51272
7314a8d51272
```

Vérification de la liste des conteneurs après suppression

$ docker ps -a

```
root@debian-console:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS                       PORTS                                   NAMES
7314a8d51272   nginx          "/docker-entrypoint.…"   45 minutes ago   Up 16 seconds                0.0.0.0:8080->80/tcp, :::8080->80/tcp   gifted_shaw
2e2f5e8a6326   hello-world    "/hello"                 47 minutes ago   Exited (0) 46 minutes ago                                            laughing_elbakyan
3c3dc8998a78   hello-world    "/hello"                 6 days ago       Exited (0) 6 days ago                                                condescending_clarke
8d6f0038333c   f5a6b296b8a2   "/docker-entrypoint.…"   6 days ago       Exited (255) 50 minutes ago  80/tcp                                  some-nginx
```

Suppression de l'image NGINX en utilisant son nom ou son ID

$ docker rmi nginx

```
root@debian-console:~# docker rmi nginx
Untagged: nginx:latest
Untagged: nginx@sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d912703568daff790ab3f755
Deleted: sha256:61395b4c586da2b9b3b7ca903ea6a448e6783dfdd7f768ff2c1a0f3360aaba99
Deleted: sha256:1c69f36a0d9b59b762eaba410fa9fd01b85140670a8d49199a7b37702cc956c0
Deleted: sha256:bac209bfab6997cccf20779ae98d5f77a66867734499ecf604a50a5826f6b8a4
Deleted: sha256:859676f4cd3004af025a02dade096ad6f9391d94d1b1a983fc6098debe435055
Deleted: sha256:cbbd97cee0d824e5e82f9a4b2e93c5eb3c66fd72a2624d5c1521dc3395bfd1e2
Deleted: sha256:0b41545d8c3de3b78778d591a8da3d9dfa5fa8807baef5edf21e5eb94ded792d
Deleted: sha256:7e87866b23143eb30086086a669b2e902368a5836446a885b2411d3feef18bef
Deleted: sha256:d310e774110ab038b30c6a5f7b7f7dd527dbe527854496bd30194b9ee6ea496e
```

Vérification de la liste des images téléchargées après la suppression de l'image

$ docker images

```
root@debian-console:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
nginx         <none>    f5a6b296b8a2   2 weeks ago     187MB
hello-world   latest    9c7a54a9a43c   4 months ago    13.3kB
```

Étape 3 – Création de vos propres images Docker

Création d'un fichier dockerfile

$ touch Dockerfile

Création d'une image

création nommée my-hello-world basée sur Debian avec un Dockerfile

$ nano Dockerfile

Écrivez ceci dans le fichier

FROM debian:latest
CMD ["echo", "Hello World!"]

Maintenant il faut construire l'image avec

$ docker build -t my-hello-world .

```
root@debian-console:~# docker build -t my-hello-world .
[+] Building 4.8s (5/5) FINISHED                                                docker:default
 => [internal] load build definition from Dockerfile                                      0.2s
 => => transferring dockerfile: 85B                                                       0.0s
 => [internal] load .dockerignore                                                         0.2s
 => => transferring context: 2B                                                           0.0s
 => [internal] load metadata for docker.io/library/debian:latest                          1.6s
 => [1/1] FROM docker.io/library/debian:latest@sha256:eaace54a93d7b69c7c52bb8ddf9b3fcba0c106a4  3.0s
 => => resolve docker.io/library/debian:latest@sha256:eaace54a93d7b69c7c52bb8ddf9b3fcba0c106a4  0.1s
 => => sha256:eaace54a93d7b69c7c52bb8ddf9b3fcba0c106a497bc1fdbb89a6299cf945c63 1.85kB / 1.85kB  0.0s
 => => sha256:8a6e23e1b192b30eff14036a92e9ecdb551a1a10aa8535728b0c13d14d8c9462 529B / 529B      0.0s
 => => sha256:2657a4a0a6d5e8b3515004185275768f115a64a833de40125bb3f6b0b8cc598b 1.46kB / 1.46kB  0.0s
 => => sha256:167b8a53ca4504bc6aa3182e336fa96f4ef76875d158c1933d3e2fa19c57e0 49.56MB / 49.56MB  1.1s
 => => extracting sha256:167b8a53ca4504bc6aa3182e336fa96f4ef76875d158c1933d3e2fa19c57e0c3       1.6s
 => exporting to image                                                                    0.0s
 => => exporting layers                                                                   0.0s
 => => writing image sha256:5715a814c120581169f46911fcd1c06cb093e03808717ed87b4ed0bcf0cff3a9  0.0s
 => => naming to docker.io/library/my-hello-world                                         0.0s
```

Vérification du fonctionnement du conteneur my-hello-world

$ docker run my-hello-world

```
root@debian-console:~# docker run my-hello-world
Hello World!
```

Création d'un fichier html dans un répertoire de travail

$ cd /home/etu1/

$ mkdir tp (dossier que je vais utiliser)

$ touch index.html

Pour ajouter du code html :

$ nano index.html

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>TP</title>
  <link rel="stylesheet" href="style.css">
  <script src="script.js"></script>
</head>
<body>
  <h1>CECI EST UN TEST</h1>
</body>
</html>
```

Vérification du fichier html dans le répertoire de travail

$ ls /home/etu1/tp

Vérification de son contenu :

$ cat /home/etu1/tp/index.html

```
root@debian-console:~# ls /home/etu1/tp/
index.html
root@debian-console:~# cat /home/etu1/tp/index.html
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>TP</title>
  <link rel="stylesheet" href="style.css">
  <script src="script.js"></script>
</head>
<body>
  <h1>CECI EST UN TEST</h1>
</body>
</html>
```

Construction d'une image my-nginx basée sur NGINX, en incluant le fichier HTML créé :

Je me déplace dans le répertoir du fichier html créé
$ cd /home/etu1/tp

Je créer un fichier Dockerfile et ajoute le texte ci-dessous:

FROM nginx:latest
COPY index.html /usr/share/nginx/html

Construisez l'image avec la commande :
$ docker build -t my-nginx .

Lancement d'un conteneur basé sur l'image my-nginx

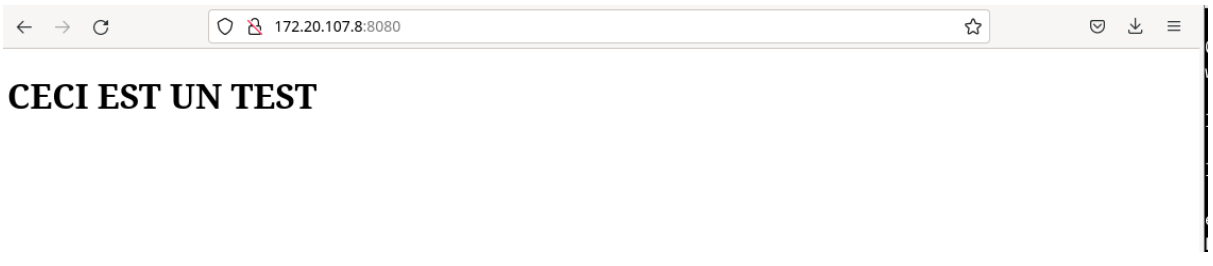
$docker run -d -p 8080:80 my-nginx

Vérification du lancement du conteneur

$docker ps

Vérification de l'affichage de la page dans un navigateur web

Pour faire je tape l'adresse IP de ma machine avec le port 8080 dans le navigateur



Étape 4 – Automatisation du déploiement de vos conteneurs

Création du fichier de configuration Docker Compose

$ touch docker-compose.yml
$ nano docker-compose.yml
ecrivez ceci à l'intérieur

```
version: '3'
services:
  my-nginx:
    image: my-nginx
    ports:
      - "8080:80"
```

Récupération des images spécifiées dans le fichier de configuration sans démarrer les conteneurs :

$docker compose pull

```
root@debian-console:~# docker compose pull
[+] Pulling 1/1
 ✔ my-nginx Pulled                                                1.0s
```

Lancement du conteneur en 1er plan en utilisant Docker Compose

$docker-compose up

```
root@debian-console:~# docker-compose up
Starting root_my-nginx_1 ... done
Attaching to root_my-nginx_1
my-nginx_1  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
my-nginx_1  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
my-nginx_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
my-nginx_1  | 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
my-nginx_1  | 10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
my-nginx_1  | /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
my-nginx_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
my-nginx_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
my-nginx_1  | /docker-entrypoint.sh: Configuration complete; ready for start up
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: using the "epoll" event method
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: nginx/1.25.2
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: OS: Linux 6.1.0-11-amd64
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: start worker processes
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: start worker process 29
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: start worker process 30
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: start worker process 31
```

Vérification que la page NGINX s'affiche depuis le navigateur

Je fais cela en utilisant l'adresse IP de ma VM suivie du port 8080.

Stopper le conteneur

Control + c dans le terminal

Lancement du conteneur au second plan en utilisant Docker Compose

$docker-compose up -d

```
root@debian-console:~# docker-compose up -d
Starting root_my-nginx_1 ... done
```

Récupération des logs en temps réel du conteneur lancé en arrière-plan

$docker-compose logs -f

```
root@debian-console:~# docker-compose logs -f
Attaching to root_my-nginx_1
my-nginx_1  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configurati
my-nginx_1  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
my-nginx_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
my-nginx_1  | 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
my-nginx_1  | 10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.con
my-nginx_1  | /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
my-nginx_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
my-nginx_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
my-nginx_1  | /docker-entrypoint.sh: Configuration complete; ready for start up
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: using the "epoll" event method
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: nginx/1.25.2
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: OS: Linux 6.1.0-11-amd64
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: start worker processes
my-nginx_1  | 2023/09/27 11:54:59 [notice] 1#1: start worker process 29
```

Arrêt du conteneur avec Docker Compose :

$docker-compose down

```
root@debian-console:~# docker-compose down
Stopping root_my-nginx_1 ... done
Removing root_my-nginx_1 ... done
Removing network root_default
```

Création d'un fichier de configuration permettant l'installation de wordpress + PostgreSQL

1. Création d'un fichier de configuration d'un docker.
   $ mkdir /home/etu1/docker
   $ cd /home/etu1/docker
   $ touch /home/etu1/docker/ docker-compose.yml
   $ nano /home/etu1/docker/ docker-compose.yml

   Écrivez ceci dans le fichier de configuration

```
version: '3'
services:
  db:
    image: postgres:latest
    volumes:
      - db-data:/var/lib/postgresql/data
    environment:
      POSTGRES_PASSWORD: btssio1
      POSTGRES_DB: wordpress
  wordpress:
    image: wordpress:latest
    ports:
      - "8080:80"
    volumes:
      - wp-data:/var/www/html
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: etu1
      WORDPRESS_DB_PASSWORD: btssio1
      WORDPRESS_DB_NAME: wordpress
volumes:
  db-data:
  wp-data:
```

2. Déployer la configuration
   $ docker-compose -up -d

```
root@debian-console:/home/etu1/docker# docker-compose up -d
Creating network "docker_default" with the default driver
Creating docker_db_1        ... done
Creating docker_wordpress_1 ... done
root@debian-console:/home/etu1/docker#
```

3. Stopper le docker
   $ docker-compose down