TP 1 : Docker

Après installation des outils requis, nous allons tester quelques commandes :

Commande : **docker run hello-world**

```
C:\Users\Anissa>docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:4bd78111b6914a99dbc560e6a20eab57ff6655aea4a80c50b0c5491968cbc2e6
Status: Downloaded newer image for hello-world:latest

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

Commande : **docker run -it ubuntu bash**

```
C:\Users\Anissa>docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
57c139bbda7e: Pull complete
Digest: sha256:e9569c25505f33ff72e88b2990887c9dcf230f23259da296eb814fc2b41af999
Status: Downloaded newer image for ubuntu:latest
root@cacd5e6fdc4c:/#
```

Commande : **docker images**

```
C:\Users\Anissa>docker images
REPOSITORY                    TAG       IMAGE ID       CREATED        SIZE
ubuntu                        latest    fd1d8f58e8ae   2 weeks ago    77.9MB
nginxinc/nginx-unprivileged   latest    926cafd5e393   2 months ago   187MB
hello-world                   latest    d2c94e258dcb   9 months ago   13.3kB
```

Commande : **docker ps -a**

```
C:\Users\Anissa>docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED       STATUS                     PORTS     NAMES
cacd5e6fdc4c   ubuntu        "bash"     6 hours ago   Exited (0) 48 seconds ago            crazy_diffie
62e5f827d992   hello-world   "/hello"   6 hours ago   Exited (0) 6 hours ago               objective_maxwell
```

Commande : **docker run -p 80:80 nginx**

```
C:\Users\Anissa>docker run -p 80:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
c57ee5000d61: Pull complete
9b0163235c08: Pull complete
f24a6f652778: Pull complete
9f3589a5fc50: Pull complete
f0bd99a47d4a: Pull complete
398157bc5c51: Pull complete
1ef1c1a36ec2: Pull complete
Digest: sha256:84c52dfd55c467e12ef85cad6a252c0990564f03c4850799bf41dd738738691f
Status: Downloaded newer image for nginx:latest
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/02/13 13:40:05 [notice] 1#1: using the "epoll" event method
2024/02/13 13:40:05 [notice] 1#1: nginx/1.25.3
2024/02/13 13:40:05 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/02/13 13:40:05 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2024/02/13 13:40:05 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/02/13 13:40:05 [notice] 1#1: start worker processes
2024/02/13 13:40:05 [notice] 1#1: start worker process 29
2024/02/13 13:40:05 [notice] 1#1: start worker process 30
2024/02/13 13:40:05 [notice] 1#1: start worker process 31
2024/02/13 13:40:05 [notice] 1#1: start worker process 32
```

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Commande : **docker run -p 80:80:80 -d nginx**

```
C:\Users\Anissa>docker run -p 8080:80 -d nginx
e0c812a395897b4eaca7e9ead11e43e1e9cf105959a837407e32f975022c76ac
```

J'ai modifié le port car mon port 80 est déjà occupé par la précédente commande.

5) Exécuter un serveur web dans un conteneur docker :

a) Récupérer une image :

Nous allons nous servir de l'image récupéré précédemment, la vérification de l'image a également été effectué plus haut. Nous avons crée un fichier en HTML .

index.html

C: > Users > Anissa > Desktop > Licence > Ynov > DevOps > Docker > index.html > html > body > h1

```
1   <!DOCTYPE html>
2   <html lang="fr">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <title>Document</title>
7   </head>
8   <body>
9       <h1> Bonjour</h1>
10  </body>
11  </html>
```

d) Pour des questions pratiques je vais en revanche arrêter mes conteneurs actifs. Pour se faire je vais commencer par lister les conteneurs en cours d'exécution avec la commande : **docker ps**

```
C:\Users\Anissa>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED       STATUS       PORTS                  NAMES
e0c812a39589   nginx     "/docker-entrypoint.…"   2 hours ago   Up 2 hours   0.0.0.0:8080->80/tcp   crazy_bartik
d67d0aed9bbd   nginx     "/docker-entrypoint.…"   2 hours ago   Up 2 hours   0.0.0.0:80->80/tcp     sleepy_davinci
```

Avec la commande docker stop, j'arrête mes conteneurs actifs :

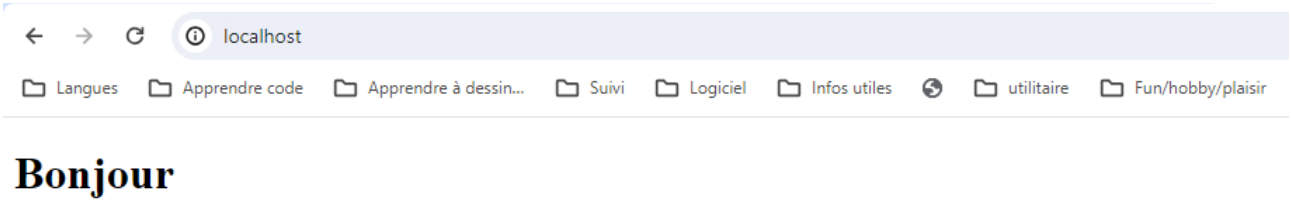
```
C:\Users\Anissa>docker stop e0c812a39589
e0c812a39589

C:\Users\Anissa>docker stop d67d0aed9bbd
d67d0aed9bbd
```

Je vérifie à nouveau mes conteneurs actifs pour m'assurer qu'ils ont bien été arrêtés :

```
C:\Users\Anissa>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

A présent je démarre un conteneur qui sert la page HTML crée précédemment à l'aide d'un volume :

```
C:\Users\Anissa>docker run -p 80:80 -v C:\Users\Anissa\Desktop\Licence\Ynov\DevOps\Docker:/usr/share/nginx/html -d nginx
0ffa835f3b1f70091295af95dbf4f41ef086342c1289e06369a156cca41eece3
```



e) A présent nous allons supprimer ce conteneur et arriver au même résultat grâce à la commande

docker cp qui permets de copier des fichiers ou des répertoires entre le système de fichiers local et un conteneur Docker en cours d'exécution.

Dans un premier temps je vais afficher la liste de tout mes conteneurs avec la commande **docker container ls -a** :

```
C:\Users\Anissa>docker container ls -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                      PORTS                NAMES
0ffa835f3b1f   nginx         "/docker-entrypoint.…"   10 minutes ago   Up 10 minutes               0.0.0.0:80->80/tcp   unruffled_leakey
438e33b68b76   nginx         "/docker-entrypoint.…"   12 minutes ago   Created                     0.0.0.0:80->80/tcp   wizardly_faraday
bbad5085202d   nginx         "/docker-entrypoint.…"   26 minutes ago   Created                                          jolly_hoover
c60c218d55ff   nginx         "/docker-entrypoint.…"   44 minutes ago   Created                                          amazing_hellman
e0c812a39589   nginx         "/docker-entrypoint.…"   2 hours ago      Exited (0) 22 minutes ago                        crazy_bartik
5bb646e168dc   nginx         "/docker-entrypoint.…"   2 hours ago      Created                                          inspiring_lamport
d67d0aed9bbd   nginx         "/docker-entrypoint.…"   2 hours ago      Exited (0) 21 minutes ago                        sleepy_davinci
b27851418754   nginx         "/docker-entrypoint.…"   3 hours ago      Exited (0) 3 hours ago                           romantic_shamir
cacd5e6fdc4c   ubuntu        "bash"                   9 hours ago      Exited (0) 3 hours ago                           crazy_diffie
62e5f827d992   hello-world   "/hello"                 9 hours ago      Exited (0) 9 hours ago                           objective_maxwell
```

Ensuite nous allons tout simplement tous les supprimer avec la commande **docker rm** :

```
C:\Users\Anissa>docker container ls -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                      PORTS                NAMES
0ffa835f3b1f   nginx         "/docker-entrypoint.…"   10 minutes ago   Up 10 minutes               0.0.0.0:80->80/tcp   unruffled_leakey
438e33b68b76   nginx         "/docker-entrypoint.…"   12 minutes ago   Created                     0.0.0.0:80->80/tcp   wizardly_faraday
bbad5085202d   nginx         "/docker-entrypoint.…"   26 minutes ago   Created                                          jolly_hoover
c60c218d55ff   nginx         "/docker-entrypoint.…"   44 minutes ago   Created                                          amazing_hellman
e0c812a39589   nginx         "/docker-entrypoint.…"   2 hours ago      Exited (0) 22 minutes ago                        crazy_bartik
5bb646e168dc   nginx         "/docker-entrypoint.…"   2 hours ago      Created                                          inspiring_lamport
d67d0aed9bbd   nginx         "/docker-entrypoint.…"   2 hours ago      Exited (0) 21 minutes ago                        sleepy_davinci
b27851418754   nginx         "/docker-entrypoint.…"   3 hours ago      Exited (0) 3 hours ago                           romantic_shamir
cacd5e6fdc4c   ubuntu        "bash"                   9 hours ago      Exited (0) 3 hours ago                           crazy_diffie
62e5f827d992   hello-world   "/hello"                 9 hours ago      Exited (0) 9 hours ago                           objective_maxwell

C:\Users\Anissa>docker rm 0ffa835f3b1f 438e33b68b76 bbad5085202d c60c218d55ff e0c812a39589 5bb646e168dc d67d0aed9bbd b27851418754 cacd5e6fdc4c 62e5f827d992
438e33b68b76
bbad5085202d
c60c218d55ff
e0c812a39589
5bb646e168dc
d67d0aed9bbd
b27851418754
cacd5e6fdc4c
62e5f827d992
Error response from daemon: You cannot remove a running container 0ffa835f3b1f70091295af95dbf4f41ef086342c1289e06369a156cca41eece3. Stop the container before attempting
 removal or force remove
```

Oup’s ! Il semble que j’ai oublié d’arrêter mon précédent conteneur !

Nous allons effectuer les mêmes étapes plus haut : **docker ps ,** pour afficher l’id du conteneur actif

```
C:\Users\Anissa>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                NAMES
0ffa835f3b1f   nginx     "/docker-entrypoint.…"   22 minutes ago   Up 22 minutes   0.0.0.0:80->80/tcp   unruffled_leakey

C:\Users\Anissa>docker stop 0ffa835f3b1f
0ffa835f3b1f
```

Nous l’avons stoppé, il nous reste plus qu’à le supprimer et vérifier qu’il ne reste plus de container :

```
C:\Users\Anissa>docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                       PORTS     NAMES
0ffa835f3b1f   nginx     "/docker-entrypoint.…"   23 minutes ago   Exited (0) 18 seconds ago              unruffled_leakey

C:\Users\Anissa>docker rm 0ffa835f3b1f
0ffa835f3b1f

C:\Users\Anissa>docker container ls -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Nous allons créer un nouveau conteneur vide :

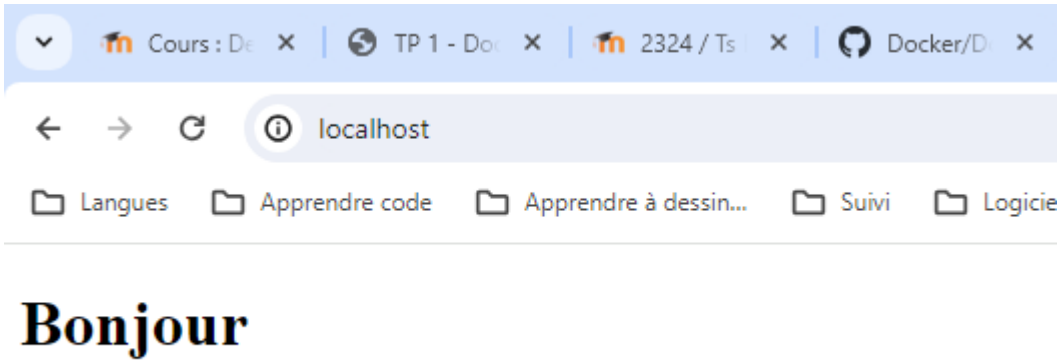
```
C:\Users\Anissa>docker create -p 80:80 nginx
8f128f609cc562492519939a091c27a5dedabc19f6d30c63f722a83d17662d34
```

Ensuite nous allons copier notre fichier index.html dans ce conteneur :

```
C:\Users\Anissa>docker cp C:\Users\Anissa\Desktop\Licence\Ynov\DevOps\Docker\index.html 8f128f609cc5:/usr/share/nginx/html/
Successfully copied 2.05kB to 8f128f609cc5:/usr/share/nginx/html/
```

Puis on démarre notre container :

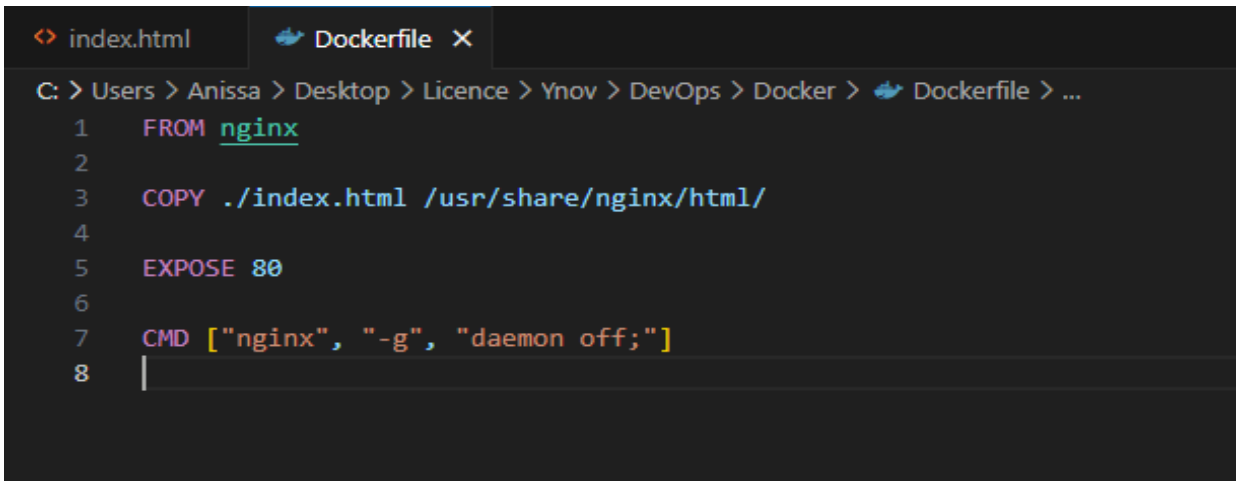
```
C:\Users\Anissa>docker start nostalgic_murdock
nostalgic_murdock
```



Le résultat est bien celui attendu.

6) Builder une image

a) Grâce au Dockerfile, nous allons créer une image :



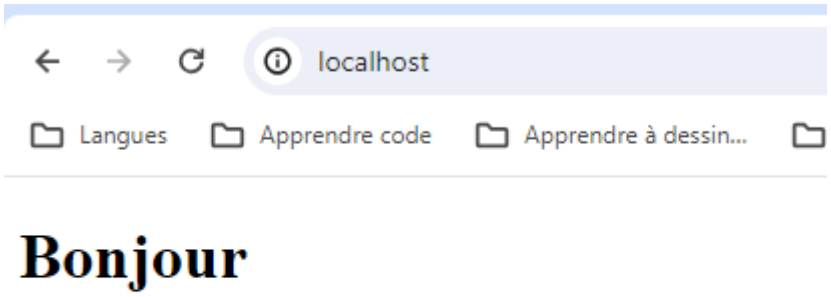
On se rend sur le dossier où se trouve dans notre Dockerfile via la commande cd puis on build notre image :

```
C:\Users\Anissa\Desktop\Licence\Ynov\DevOps\Docker>docker buildx build -t firstimage C:\Users\Anissa\Desktop\Licence\Ynov\DevOps\Docker
[+] Building 0.9s (7/7) FINISHED                                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                                       0.1s
 => => transferring dockerfile: 144B                                                                                       0.0s
 => [internal] load .dockerignore                                                                                          0.1s
 => => transferring context: 2B                                                                                            0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                            0.0s
 => [internal] load build context                                                                                          0.2s
 => => transferring context: 271B                                                                                          0.0s
 => [1/2] FROM docker.io/library/nginx                                                                                     0.4s
 => [2/2] COPY ./index.html /usr/share/nginx/html/                                                                         0.1s
 => exporting to image                                                                                                     0.1s
 => => exporting layers                                                                                                    0.1s
 => => writing image sha256:ad0575d7ec6a3a68593ecfd19d6b5b1bb3b10d99d50a3b1d9031d07bbd673a29                               0.0s
 => => naming to docker.io/library/firstimage                                                                              0.0s
```

b) Après l’avoir construite nous allons exécuter la commande docker run pour l’afficher :

```
C:\Users\Anissa\Desktop\Licence\Ynov\DevOps\Docker>docker run -p 80:80 firstimage
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/02/13 21:00:12 [notice] 1#1: using the "epoll" event method
2024/02/13 21:00:12 [notice] 1#1: nginx/1.25.3
2024/02/13 21:00:12 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/02/13 21:00:12 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2024/02/13 21:00:12 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/02/13 21:00:12 [notice] 1#1: start worker processes
2024/02/13 21:00:12 [notice] 1#1: start worker process 29
2024/02/13 21:00:12 [notice] 1#1: start worker process 30
2024/02/13 21:00:12 [notice] 1#1: start worker process 31
2024/02/13 21:00:12 [notice] 1#1: start worker process 32
```

Résultat :



c) Rappelons les faits :

Dans la procédure de la question 5 nous avons récupéré une image puis nous avons exécuté un container qui contenait notre image. L’avantage de ce process est la rapidité et la facilité d’utilisation. Notre image n’a pas besoin d’être construite, c’est l’idéal pour des cas simple sans besoin de personnaliser l’environnement.

Pour la question 6 nous avons crée un Dockerfile où nous avons écrit nos instructions pour construire notre image Docker. Les avantages de cette méthode est la reproductibilité, on peut

partager notre Dockerfile ainsi qu’avoir un contrôle total sur l’environnement et la configuration de notre image .

En conclusion, le choix entre utiliser un Dockerfile ou une image existante dépend de nos besoins spécifiques pour notre application. Le Dockerfile peut être plus complexe et si des modifications y sont apportées on aura besoin de reconstruire entièrement notre image.