L'objectif de ce TP va d'être de créer 1 serveur WEB httpd qui va afficher le contenu d'un fichier index.html stocké sur un volume partagé et qui sera mis à jour toutes les heures par un container producteur à l'aide d'un script.

1. Tout d'abord nous allons crée un réseau part pont appelé tpweb à l'aide de la commande docker network create --driver bridge tpweb

```
[root@centos7 ~]# docker network create --driver bridge tpweb
f6d27e190af52f516dee703fad60dd53923bb9d4fcd0dcd507a25c2ebe52a751
```

2. On effectue un docker network ls pour s'assurer de la bonne création de celui-ci

```
NETWORK ID          NAME                DRIVER              SCOPE
f000f233af40        bridge              bridge              local
08ea34653346        host                host                local
0c7d63331f02        none                null                local
f6d27e190af5        tpweb               bridge              local
[root@centos7 ~]#
```

3. Maintenant que notre réseau est créé nous allons créer le volume qui sera partagé entre les deux containers. Pour cela nous utilisons la commande docker volume create tpweb-partage

4. Comme précédemment nous allons vérifier la bonne création du volume a l'aide de la commande docker volume ls

```
[root@centos7 ~]# docker volume ls
DRIVER              VOLUME NAME
local               tpweb-partage
[root@centos7 ~]#
```

5. On crée ensuite notre serveur web (web) basé sur une image httpd que l'on exposera sur le port 80 :80 et dont on montera le volume crée précédemment
   La commande est la suivante : docker run -tid --name WEB -v tpweb-partage:/home/Projet/Apache2/html_data --net tpweb -p 80:80 httpd

```
[root@centos7 ~]# docker run  -tid --name WEB -v tpweb-partage:/home/Projet/Apache2/html_data
/ --net tpweb -p 80:80 httpd
Unable to find image 'httpd:latest' locally
latest: Pulling from library/httpd
852e50cd189d: Pull complete
67d51c33d390: Pull complete
b0ad2a3b9567: Pull complete
136f1f71f30c: Pull complete
01f8ace29294: Pull complete
Digest: sha256:fddc534b7f6bb6197855be559244adb11907d569aae1283db8e6ce8bb8f6f456
Status: Downloaded newer image for httpd:latest
ae93c65de9868f8023bbe8dfaf57da5d683a5b4b422e89e2438ecc657360a554
[root@centos7 ~]#
```

6. Nous allons maintenant créer le container nginx qui faira office de producteur et qui modifiera le fichier index.html toutes les heurex : docker run -tid --name SERVNGINX -v tpweb-partage:/home/Projet/Apache2/html_data --net tpweb nginx

```
[root@centos7 ~]# docker run  -tid --name PRODUCTEUR -v tpweb-partage:/home/Projet/Apache2/ht
ml_data/ --net tpweb nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
852e50cd189d: Already exists
571d7e852307: Pull complete
addb10abd9cb: Pull complete
d20aa7ccdb77: Pull complete
8b03f1e11359: Pull complete
Digest: sha256:6b1daa9462046581ac15be20277a7c75476283f969cb3a61c8725ec38d3b01c3
Status: Downloaded newer image for nginx:latest
762bdc6468b6e3dcbd0b96377c00334cd62a27f52d7f1e6ee6e0177465721cd6
[root@centos7 ~]#
```

7. On vérifie la bonne création des containers à l'aide d'un docker ps :

```
[root@centos7 ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
              PORTS               NAMES
762bdc6468b6        nginx               "/docker-entrypoint.…"   About a minute ago   Up Abou
t a minute   80/tcp              PRODUCTEUR
ae93c65de986        httpd               "httpd-foreground"       29 minutes ago       Up 29 m
inutes       0.0.0.0:80->80/tcp  WEB
[root@centos7 ~]#
```

8. Maintenant que nos 2 containers sont créés nous allons connecter sur chacun pour installer les prérequis et les configurer.

On utilisera la commande docker exec -it NOMDUSERVER bash pour se connecter sur le container et mettre à jour les paquets, installer nano et la commande ping avec la commande suivante : apt update && apt install nano && apt install iputils-ping -y

Nous le faisons dans un premier temps sur le serveur WEB

```
root@ae93c65de986:~/html_data# apt update && apt install nano && apt install iputils-ping -yqq
Get:1 http://security.debian.org/debian-security buster/updates InRelease [65.4 kB]
Get:2 http://deb.debian.org/debian buster InRelease [121 kB]
Get:3 http://deb.debian.org/debian buster-updates InRelease [51.9 kB]
Get:4 http://security.debian.org/debian-security buster/updates/main amd64 Packages [252 kB]
Get:5 http://deb.debian.org/debian buster/main amd64 Packages [7906 kB]
Get:6 http://deb.debian.org/debian buster-updates/main amd64 Packages [7856 B]
Fetched 8405 kB in 2s (3600 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  lsb-base
Use 'apt autoremove' to remove it.
Suggested packages:
  spell
The following NEW packages will be installed:
  nano
0 upgraded, 1 newly installed, 0 to remove and 4 not upgraded.
Need to get 544 kB of archives.
After this operation, 2269 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian buster/main amd64 nano amd64 3.2-3 [544 kB]
Fetched 544 kB in 0s (5625 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package nano.
(Reading database ... 6692 files and directories currently installed.)
Preparing to unpack .../archives/nano_3.2-3_amd64.deb ...
Unpacking nano (3.2-3) ...
Setting up nano (3.2-3) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group editor) doesn't exist
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group pico) doesn't exist
The following package was automatically installed and is no longer required:
  lsb-base
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  libcap2 libcap2-bin libpam-cap
The following NEW packages will be installed:
  iputils-ping libcap2 libcap2-bin libpam-cap
0 upgraded, 4 newly installed, 0 to remove and 4 not upgraded.
Need to get 104 kB of archives.
After this operation, 319 kB of additional disk space will be used.
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libcap2:amd64.
```

9. Toujours sur le serveur WEB nous allons nous déplacer dans le répertoire /home/Projet/Apache2/html_data et créer un fichier index.html avec le contenu suivant :

```
root@ae93c65de986:~/html_data# cat index.html
First texte
root@ae93c65de986:~/html_data#
```

10. Nous allons paramétrer le serveur apache pour qu'il pointe par défaut sur ce fichier index. Pour cela on se déplace dans le répertoire suivant : cd /usr/local/apache2/conf et on va éditer le httpd.conf avec nano et rechercher la phrase Possible values for .

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/usr/local/apache2/htdocs"
<Directory "/usr/local/apache2/htdocs">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
```

11. On modifie les valeurs entre « « par le chemin de notre index.html comme ceci :

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/home/Projet/Apache2/html_data"
<Directory "/home/Projet/Apache2/html_data">
    #
    # Possible values for the Options directive are "None", "All",
```

12. Nous sortons maintenant du container web afin de vérifier que le container producteur communique bien avec le container web.
    Pour cela nous faisons un docker inspect network tpweb afin de récupérer l'adresse IP du container WEB.

```
[root@centos7 ~]# docker inspect network tpweb
[
    {
        "Name": "tpweb",
        "Id": "f6d27e190af52f516dee703fad60dd53923bb9d4fcd0dcd507a25c2ebe52a751",
        "Created": "2020-11-28T08:54:00.253573439-05:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "762bdc6468b6e3dcbd0b96377c00334cd62a27f52d7f1e6ee6e0177465721cd6": {
                "Name": "PRODUCTEUR",
                "EndpointID": "aec7e8c14309adfddae2810f2bd2ad8da80603497f17f3d48a34ff4c8ba8dc3f",
                "MacAddress": "02:42:ac:12:00:03",
                "IPv4Address": "172.18.0.3/16",
                "IPv6Address": ""
            },
            "ae93c65de9868f8023bbe8dfaf57da5d683a5b4b422e89e2438ecc657360a554": {
                "Name": "WEB",
                "EndpointID": "1398ca041a595074a82d80cf1c26c661c451d09ce8ecf322f273a85525b3b936",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]
```

Serveur WEB : 172.18.0.2
Serveur PRODUCTEUR : 172.18.0.3

13. On redémarre le serveur WEB pour prendre en compte le changement de la configuration d'apache : docker stop id du container (visible avec docker ps) puis docker start id du container
14. On se connecte sur le serveur PRODUCTEUR pour effectuer un ping vers le serveur WEB avec la commande 172.18.0.2

```
root@762bdc6468b6:/# ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2) 56(84) bytes of data.
64 bytes from 172.18.0.2: icmp_seq=1 ttl=64 time=0.149 ms
64 bytes from 172.18.0.2: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 172.18.0.2: icmp_seq=3 ttl=64 time=0.066 ms
64 bytes from 172.18.0.2: icmp_seq=4 ttl=64 time=0.113 ms
^C
--- 172.18.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 9ms
rtt min/avg/max/mdev = 0.066/0.099/0.149/0.034 ms
root@762bdc6468b6:/#
```

15. Maintenant que l'on sait que les serveurs communique bien ensemble nous allons créer un script date.sh qui va mettre à jour toutes les heures le fichier index.html crée précédemment.

On se déplace dans le répertoire projet et on crée un répertoire script, on crée un date.sh (touch date.sh) et on insère les choses suivantes :

```
root@762bdc6468b6:/home/Projet/Script# cat date.sh
#!/bin/bash
while true
do
echo "" > /home/Projet/Apache2/html_data/index.html
date >> /home/Projet/Apache2/html_data/index.html
sleep 3600
done
```

Le script va donc faire une boucle toutes les heures qui va vider le fichier index.html et réinscrire la date et l'heure du jour.

Si on fait un cat du fichier index.html

```
root@762bdc6468b6:/home/Projet/Apache2/html_data# cat index.html
First texte
```

On peut voir que le fichier n'est pas modifié, c'est normal il faut maintenant donner les droits au script à l'aide de la commande chmod a+x date.sh avant de l'exécuter avec ./date.sh (bien s'assurer d'exécuter ses commandes dans le même répertoire que le script).

16. On se connecte maintenant sur le serveur WEB, on patiente 1 heure et on effectue la commande curl « adresse ip du serveur WEB » :

```
root@ae93c65de986:/usr/local/apache2# curl 172.18.0.2

Sat Nov 28 15:54:53 UTC 2020
```

Le script est fonctionnel est le tp terminé !