

IUT DE COLMAR

R4ROM19

ANNÉE 2022-23

---

## TP 2 - Docker-compose

---

MARTIN BAUMGAERTNER

29 mars 2023

---

## Table des matières

<b>1</b>	<b>Déployer et gérer une stack de services interconnectés</b>	<b>2</b>
1.1	Première configuration . . . . .	2
1.1.1	Question 2-3-4 . . . . .	2
1.1.2	Question 5 . . . . .	2
1.2	Configuration de services interconnectés . . . . .	4
1.2.1	Question 1-2 . . . . .	4
<b>2</b>	<b>Réaliser et publier son image Docker</b>	<b>5</b>
2.1	Créer un fichier Dockerfile . . . . .	5
2.1.1	Question a . . . . .	5
2.1.2	Question b . . . . .	5
2.1.3	Question c . . . . .	6
2.2	Dockerfile - un peu plus loin . . . . .	6
2.2.1	Question a . . . . .	6
2.2.2	Question b . . . . .	6
2.2.3	Question c . . . . .	7
2.2.4	Question d . . . . .	7
2.3	Publier son image sur un registry privé . . . . .	8
2.3.1	Question a-b-c . . . . .	8
2.3.2	Question d-e . . . . .	8
<b>3</b>	<b>Script bash pour créer et exécuter un conteneur Docker</b>	<b>9</b>
3.1	Le script . . . . .	10

---

# 1 Déployer et gérer une stack de services interconnectés

## 1.1 Première configuration

### 1.1.1 Question 2-3-4

Après avoir créé mon fichier yml et l'avoir renseigné, j'ai pu le lancer avec la commande **docker-compose -f docker-compose.yml up -d**. Je peux très bien visualiser les composants démarrés avec **docker-compose ps**, je peux obtenir le même résultat que cette commande si je fais **docker ps -all**. Voici donc ce que j'obtiens :

```
~/R4ROM19-outils-devops/TP2 on main !1 ?2 > docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
b9b1d124451f	mysql	"docker-entrypoint.s..."	10 minutes ago	Up 10 minutes	0.0.0.0:3306->3306/tcp, 33060/tcp	tp2-mysdb-1
5b7244156007	httpd	"httpd-foreground"	10 minutes ago	Up 10 minutes	0.0.0.0:80->80/tcp	tp2-mywebserver-1

FIGURE 1 – docker ps -all

Je peux aussi bien entendu visualiser les logs avec **docker-compose logs**

```
~/R4ROM19-outils-devops/TP2 on main !1 ?2 > docker-compose logs
```

```
tp2-mywebserver-1 | AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 172.18.0.3. Set the 'ServerName' directive globally to suppress this message
tp2-mywebserver-1 | [Fri Mar 24 13:23:55.035552 2023] [mpm_event:notice] [pid 1:tid 281473766776848] AH00489: Apache/2.4.56 (Ubuntu) configured -- resuming normal operations
tp2-mywebserver-1 | [Fri Mar 24 13:23:55.035754 2023] [core:notice] [pid 1:tid 281473766776848] AH00094: Command Line: 'httpd -D FOREGROUND'
tp2-mywebserver-1 | 172.18.0.1 -- [24/Mar/2023:13:24:07 +0000] "GET / HTTP/1.1" 200 45
tp2-mywebserver-1 | 172.18.0.1 -- [24/Mar/2023:13:24:08 +0000] "GET /favicon.ico HTTP/1.1" 404 196
tp2-mywebserver-1 | 172.18.0.1 -- [24/Mar/2023:13:24:09 +0000] "GET / HTTP/1.1" 200 45
tp2-mysdb-1 | 2023-03-24 13:23:55+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.32-1.el8 started.
tp2-mysdb-1 | 2023-03-24 13:23:55+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
tp2-mysdb-1 | 2023-03-24 13:23:55+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.32-1.el8 started.
tp2-mysdb-1 | 2023-03-24 13:23:55+00:00 [Note] [Entrypoint]: Initializing database files
tp2-mysdb-1 | 2023-03-24T13:23:55.958111Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a future release. Please use SET GLOBAL host_cache_size
```

FIGURE 2 – docker-compose logs

### 1.1.2 Question 5

Je me connecte au docker avec la commande **docker exec -it tp2-mysdb-1 /bin/bash**. On peut constater que j'atteris bien dans le docker en bash.

```
~/R4ROM19-outils-devops/TP2 on main !1 ?2 > docker exec -it tp2-mysdb-1 /bin/bash
```

```
bash-4.4# ls
```

```
bin boot dev docker-entrypoint-initdb.d entrypoint.sh etc home lib lib64
```

FIGURE 3 – Connexion à la base de données

---

J'ai aussi bien évidemment accès la base de données en m'y connectant avec la commande **mysql -u root -p**

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.02 sec)
```

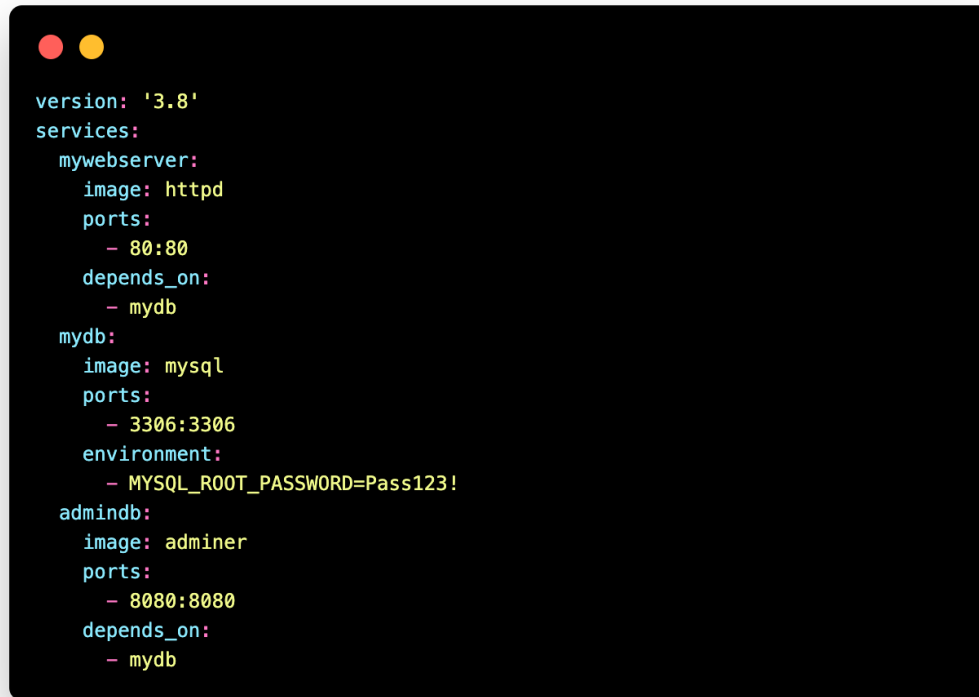
FIGURE 4 – Connexion à la base de données

---

## 1.2 Configuration de services interconnectés

### 1.2.1 Question 1-2

Je viens donc modifier mon fichier de configuration du docker **docker-compose.yaml**. de cette manière, pour y ajouter toutes les demandes de l'exercice.

A screenshot of a terminal window with a dark background and light-colored text. The text represents a Docker Compose configuration file. It defines three services: 'mywebserver' (httpd), 'mydb' (mysql), and 'admindb' (adminer). 'mywebserver' depends on 'mydb'. 'admindb' also depends on 'mydb'. The configuration includes version, image, ports, and environment variables for each service.

```
version: '3.8'
services:
  mywebserver:
    image: httpd
    ports:
      - 80:80
    depends_on:
      - mydb
  mydb:
    image: mysql
    ports:
      - 3306:3306
    environment:
      - MYSQL_ROOT_PASSWORD=Pass123!
  admindb:
    image: adminer
    ports:
      - 8080:8080
    depends_on:
      - mydb
```

FIGURE 5 – docker-compose.yaml

---

## 2 Réaliser et publier son image Docker

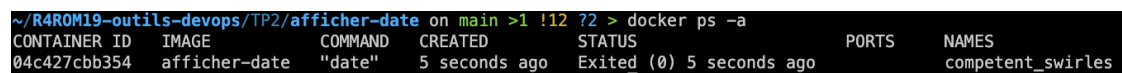
### 2.1 Créer un fichier Dockerfile

#### 2.1.1 Question a

J'ai donc créé un fichier **Dockerfile** dans lequel j'ai renseigné les commandes suivantes :

```
FROM alpine
ENTRYPOINT ["date"]
```

Puis, j'ai lancé la compilation avec **docker build -t afficher-date ..** Et pour finir je l'ai lancé avec **docker run -it afficher-date**. On peut bien évidemment vérifier sa présence : **docker ps -a**.

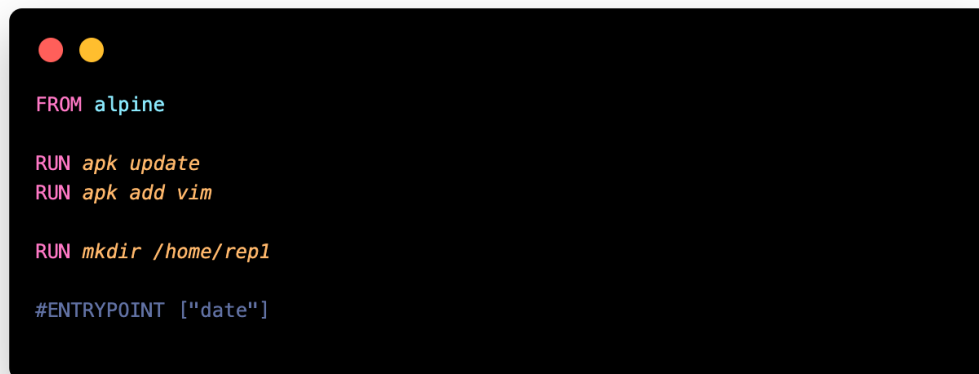


```
~/R4ROM19-outils-devops/TP2/afficher-date on main > !12 ?2 > docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS          PORTS          NAMES
04c427cbb354   afficher-date  "date"                  5 seconds ago  Exited (0) 5 seconds ago          competent_swirles
```

FIGURE 6 – docker ps -a

#### 2.1.2 Question b

À partir des instructions demandées et précisés dans le Tp voici donc le fichier **Dockerfile** final :



```
FROM alpine

RUN apk update
RUN apk add vim

RUN mkdir /home/repl

#ENTRYPOINT ["date"]
```

FIGURE 7 – Dockerfile

---

### 2.1.3 Question c

J'ai donc refait exactement les mêmes commandes que précédemment pour compiler ce nouveau fichier, et le lancer avec **docker run -it afficher-date**. Une fois dans le conteneur je peux vérifier que le dossier a bien été créé et qu'il contient bien l'éditeur vim :



FIGURE 8 – vim

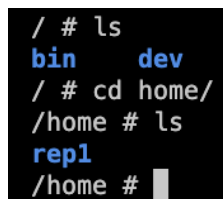


FIGURE 9 – Répertoire

## 2.2 Dockerfile - un peu plus loin

### 2.2.1 Question a

```
docker build -t mydockerfile -f mydockerfile .
```

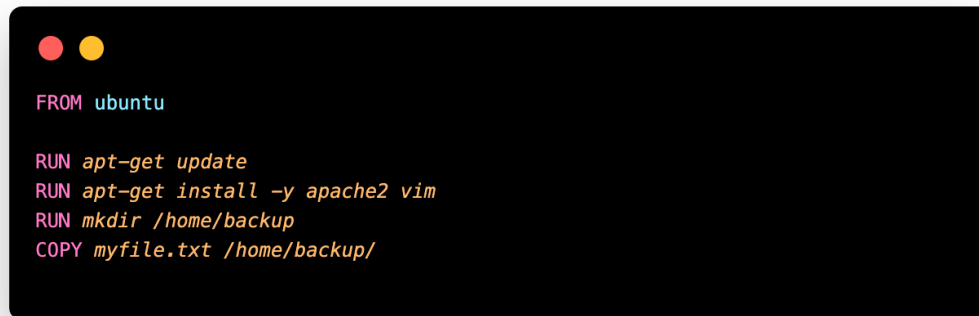
### 2.2.2 Question b

```
docker run -it mydockerfile
```

---

### 2.2.3 Question c

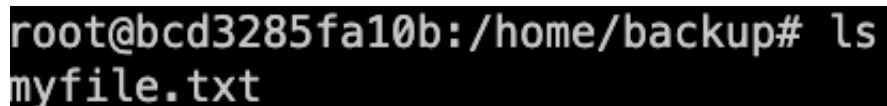
Voici donc le script utilisé pour créer le conteneur :

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The terminal displays a Dockerfile script with the following commands:

```
FROM ubuntu  
  
RUN apt-get update  
RUN apt-get install -y apache2 vim  
RUN mkdir /home/backup  
COPY myfile.txt /home/backup/
```

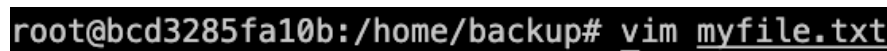
FIGURE 10 – Conteneur 2

Et donc je peux aller dans le conteneur pour vérifier la présence du fichier. Je peux aussi bien évidemment écrire dedans.

A terminal window showing the output of the 'ls' command. The prompt is 'root@bcd3285fa10b:/home/backup#' and the command is 'ls'. The output is 'myfile.txt'.

```
root@bcd3285fa10b:/home/backup# ls  
myfile.txt
```

FIGURE 11 – ls

A terminal window showing the command to open a file in vim. The prompt is 'root@bcd3285fa10b:/home/backup#' and the command is 'vim myfile.txt'.

```
root@bcd3285fa10b:/home/backup# vim myfile.txt
```

FIGURE 12 – vim myfile.txt

### 2.2.4 Question d

J'ai donc arrêté le docker et je me suis rendu compte que les modifications ne sont pas persistantes. Pour faire ceci, il faut utiliser un volume, avec la commande suivante : **docker run -it -v /Users/martinbaumgaertner/R4ROM19-outils-devops/TP2/mydockerfile :/home/backup mydockerfile**



---

## 2.3 Publier son image sur un registry privé

### 2.3.1 Question a-b-c

J'ai donc créé un dockerfile avec uniquement **FROM alpine**. Ensuite, j'ai créé l'image du dockerfile avec la commande suivante, en précisant comme tag mon prénom : **docker build -t martinbaumg/martin-baumgaertner :martin -f /Users/martinbaumgaertner/R4ROM19-outils-devops/TP2/docker/dockerfile** .

Puis, avec **1.0** : **docker build -t martinbaumg/martin-baumgaertner :1.0 -f /Users/martinbaumgaertner/R4ROM19-outils-devops/TP2/docker/dockerfile** .

### 2.3.2 Question d-e

J'ai donc mis le docker en ligne avec la commande suivante :  
**docker push --all-tags martinbaumg/martin-baumgaertner**

Et je vérifie que l'image est bien présente sur dockerhub ce qui est bien le cas, comme le démontre la capture d'écran suivante :

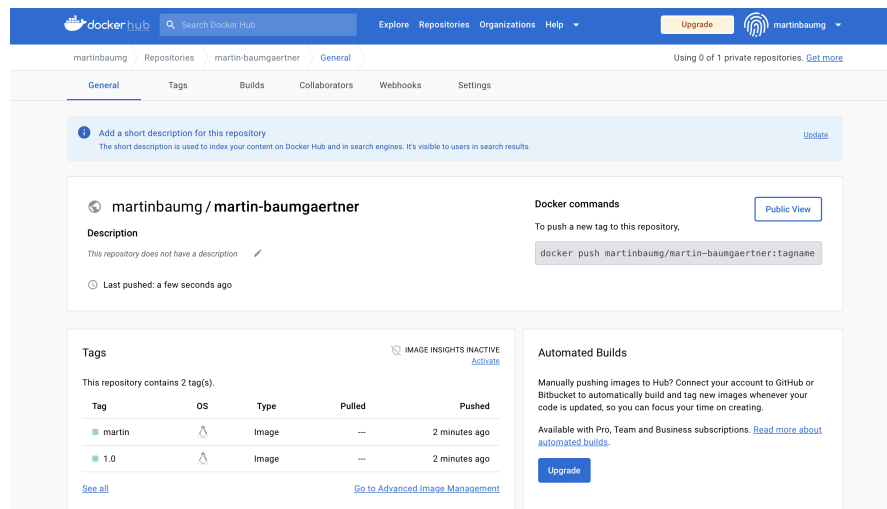


FIGURE 13 – dockerhub

---

### 3 Script bash pour créer et exécuter un conteneur Docker

J'ai donc créé le script comme demandé, que vous pourrez trouver après ces quelques lignes d'explications. Premièrement je lance bien le script avec la commande **sh run\_flask.sh**. Je peux ensuite vérifier que l'ID est le même qui s'affiche dans la console quand je me connecte au conteneur en mode interactif : Lorsque

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
d7338287bbfb	tp2_baumgaertner-martin	"python flask_app.py"	Less than a second ago	Up Less than a second	0.0.0.0:8080->8080/tcp	run_tp2_baumgaertner-martin

FIGURE 14 – ID



```
root@d7338287bbfb: /home/myapp#
```

FIGURE 15 – nom dans le conteneur

je démarre le conteneur en mode interactif avec la commande **docker exec -it run\_tp2\_baumgaertner-martin /bin/bash** et que je me connecte depuis un navigateur web, j'arrive bien sur la page suivante :

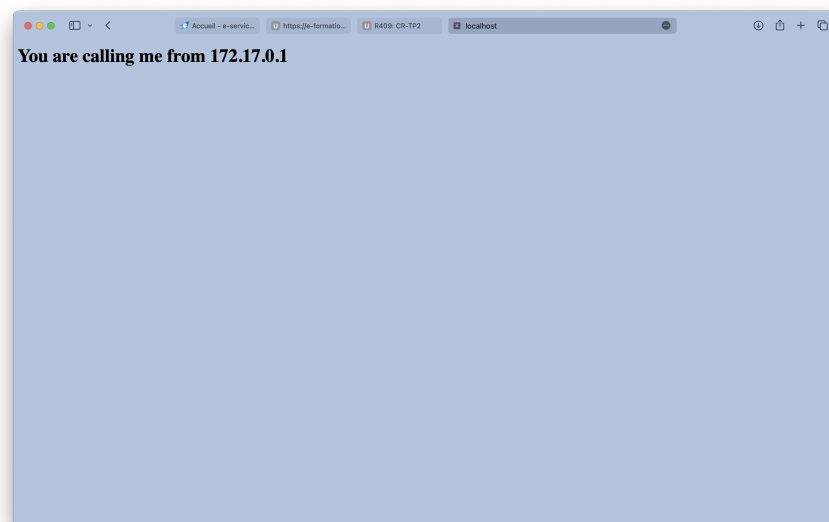
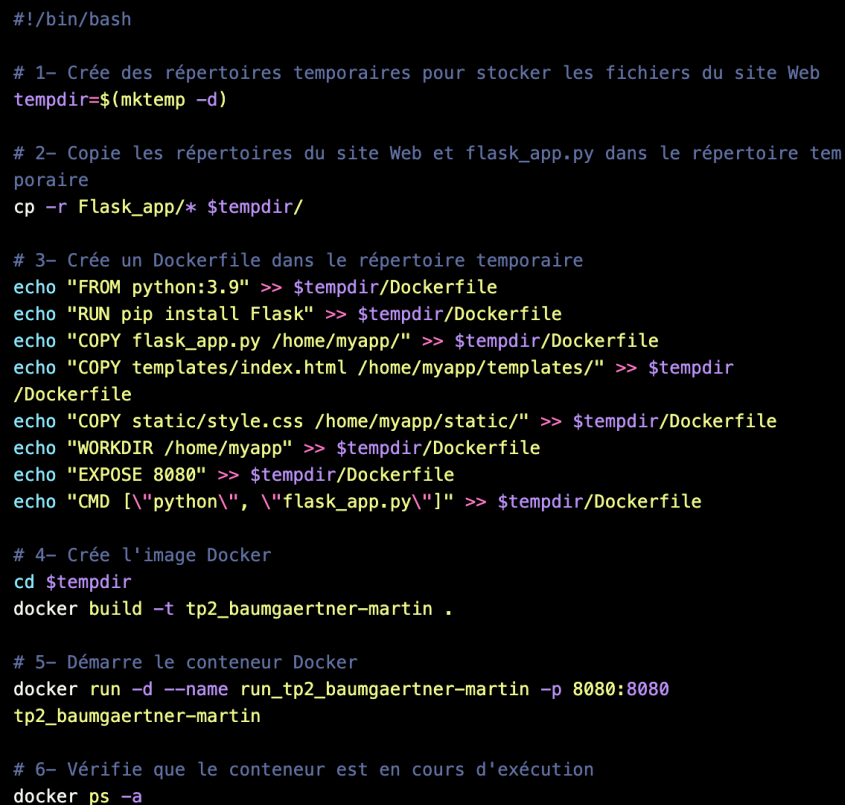


FIGURE 16 – page flask

---

### 3.1 Le script

Voici le script qui a été utilisé :



```
#!/bin/bash

# 1- Crée des répertoires temporaires pour stocker les fichiers du site Web
tempdir=$(mktemp -d)

# 2- Copie les répertoires du site Web et flask_app.py dans le répertoire temporaire
cp -r Flask_app/* $tempdir/

# 3- Crée un Dockerfile dans le répertoire temporaire
echo "FROM python:3.9" >> $tempdir/Dockerfile
echo "RUN pip install Flask" >> $tempdir/Dockerfile
echo "COPY flask_app.py /home/myapp/" >> $tempdir/Dockerfile
echo "COPY templates/index.html /home/myapp/templates/" >> $tempdir/Dockerfile
echo "COPY static/style.css /home/myapp/static/" >> $tempdir/Dockerfile
echo "WORKDIR /home/myapp" >> $tempdir/Dockerfile
echo "EXPOSE 8080" >> $tempdir/Dockerfile
echo "CMD [\"python\", \"flask_app.py\"]" >> $tempdir/Dockerfile

# 4- Crée l'image Docker
cd $tempdir
docker build -t tp2_baumgaertner-martin .

# 5- Démarre le conteneur Docker
docker run -d --name run_tp2_baumgaertner-martin -p 8080:8080 tp2_baumgaertner-martin

# 6- Vérifie que le conteneur est en cours d'exécution
docker ps -a
```

FIGURE 17 – script final