

## Application multi-conteneurs

### Mise en situation

- Dans le cas d'une application qui répond à une montée en charge et dans le cas de plusieurs instances de chaque Microservice, il devient nécessaire d'automatiser les tâches de paramétrage, de builds des images et de run des différentes instances, de définir les dépendances entre les microservices...
- Docker-Compose est un outil permettant de définir et d'exécuter des applications Docker multi-conteneurs. Avec Docker-Compose, on utilise un fichier YAML pour configurer les services de notre application. Ensuite, avec une seule commande, nous créons et démarrons tous les services à partir de cette configuration.
- Docker Compose permet en effet d'orchestrer les conteneurs, et ainsi de simplifier les déploiements sur de multiples environnements. Docker Compose est un outil écrit en Python qui permet de décrire, dans un fichier YAML, plusieurs conteneurs comme un ensemble de services.

### Objectifs

- Créer un fichier docker-compose.yml et écrire le script docker-compose.
- Lancer l'application multi-conteneurs

## Etape 1: Ecrire le script de configuration

Dans un fichier nommé **docker-compose.yml**, ajouter le script d'automatisation. Ce script permet de faire le build, le run ... et d'autres configurations relatives aux conteneurs Docker. Ci-après un exemple de script qui fait le build des 3 microservices déjà développés dans l'atelier précédent (candidate-service, job-service et discovery-service). Il fait aussi le run de 2 instances du MS candidate, 1 du MS jobs et 1 du service de découverte eureka.

```
docker-compose.yml X
1  version: "2.2"
2  services:
3    job-service:
4      container_name: job-service
5      image: "job-service"
6      build: ./MS-job
7      ports:
8        - "8082:8082"
9      hostname: job-service
10     environment:
11       - SPRING_DATASOURCE_URL=jdbc:mysql://docker-mysql:3306/DBJobs?autoReconnect=true&useSSL=false
12       - EUREKA_CLIENT_SERVICEURL_DEFAULTZONE=http://discovery-service:8761/eureka/
13     depends_on:
14       - discovery-service
15       - docker-mysql
16
17   docker-mysql:
18     container_name: docker-mysql
19     image: "mysql:5.6"
20     environment:
21       - MYSQL_ROOT_PASSWORD=admin
22       - MYSQL_DATABASE=admin
23     ports:
24       - "3308:3306"
25
26   candidat-service:
27     container_name: candidat-service
28     image: "candidat-service"
29     build: ./MS-candidat
30     ports:
31       - "8081:8081"
32     hostname: candidat
33     environment:
34       - EUREKA_CLIENT_SERVICEURL_DEFAULTZONE=http://discovery-service:8761/eureka/
35     depends_on:
36       - discovery-service
37
38   discovery-service:
39     container_name: discovery-service
40     build: ./eureka
41     ports:
42       - "8761:8761"
43     hostname: eureka
44     image: "eureka"
45     environment:
46       - EUREKA_CLIENT_SERVICEURL_DEFAULTZONE=http://discovery-service:8761/eureka/
47
```

**port:** permet d'exposer plusieurs ports de conteneurs à une machine parente. Par exemple ports: "3000:8080".

Elle permet d'exposer le port 8080 au port 3000 de la machine parente. Nous naviguerons ici vers localhost:3000.

**build :** permet de définir le lien vers le dossier racine contenant le Dockerfile pour réaliser le build

**image :** permet de définir le nom de l'image à générer

**depends\_on :** permet de définir les autres services dont le service dépend. Nous avons supposé ici que le service candidat dépend du service job. Ceci permet de contrôler l'ordre de démarrage et d'arrêt du service.

Il existe d'autres configurations possibles, telles que volumes, links, command....  
[<https://github.com/compose-spec/compose-spec/blob/master/spec.md>] [<https://docs.docker.com/compose/>]

Il faudrait d'autre part changer dans le fichier application.properties du microservice à enregistrer localhost par le nom attribué au serveur de découverte comme suit :

```
eureka.client.server-url.default-zone=http://eureka:8761/eureka/
```

## Etape 2: Lancer l'application multi-containers

Pour lancer l'application multi-containers, il suffit d'accéder à l'emplacement du fichier docker-compose.yml et par la suite il faut lancer la commande :

```
docker-compose up
```

**NB** : Si la commande ci-dessus ne fonctionne pas, vous pouvez utiliser la commande « **docker-compose up --build** ». »,

⇒ C'est la même commande, sauf que avec le --build on demande d'exécuter les fichiers Dockerfile des micro services avant d'exécuter le fichier docker\_compose.

Il faut avoir le résultat ci-dessous, qui confirment que tous les conteneurs ont été créés.

```
=> => naming to docker.io/library/candidat-service 0.0s
[+] Running 5/5
✔ Network workspace-spring-tool-suite-4-4240release_2_default Created 0.1s
✔ Container docker-mysql Created 0.1s
✔ Container discovery-service Created 0.1s
✔ Container candidat-service Created 0.1s
✔ Container job-service Created 0.1s
Attaching to candidat-service, discovery-service, docker-mysql, job-service
docker-mysql | 2024-10-12 21:48:09+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.6.51-1debian9 started.
docker-mysql | 2024-10-12 21:48:09+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
docker-mysql | 2024-10-12 21:48:09+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 5.6.51-1debian9 started.
docker-mysql | 2024-10-12 21:48:09+00:00 [Note] [Entrypoint]: Initializing database files
docker-mysql | 2024-10-12 21:48:09 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit_defaults_f...
```

Vous pouvez aussi vérifier que tous les images ont été créer sur docker desktop.

<input type="checkbox"/>	Name	Tag	Status	Created	Size	Actions
<input type="checkbox"/>	<a href="#">candidat-service</a> 9e24167f4e06	latest	<a href="#">In use</a>	17 minutes ag	589.87 MB	
<input type="checkbox"/>	<a href="#">job-service</a> 12b97188b09b	latest	<a href="#">In use</a>	17 minutes ag	526.05 MB	
<input type="checkbox"/>	<a href="#">eureka</a> bad216aa9f06	latest	<a href="#">In use</a>	17 minutes ag	528.2 MB	
<input type="checkbox"/>	<a href="#">demo</a> 976a5a3a1623	latest	Unused	23 days ago	535.74 MB	
<input type="checkbox"/>	<a href="#">mysql</a> dd3b2a5dcb48	5.6	<a href="#">In use</a>	3 years ago	302.52 MB	

Pour appeler les différents services lancés, consulter les liens :

<http://localhost:8761/>  
<http://localhost:8082/candidats/>  
<http://localhost:8081/jobs>

Maintenant pour exécuter le fichier docker-compose.yml directement sur docker hub il faut modifier le fichier docker-compose.yml en ajoutant le repository et la version pour les images.

Puis lancer les deux commandes

1- docker compose build : pour préparer les images.

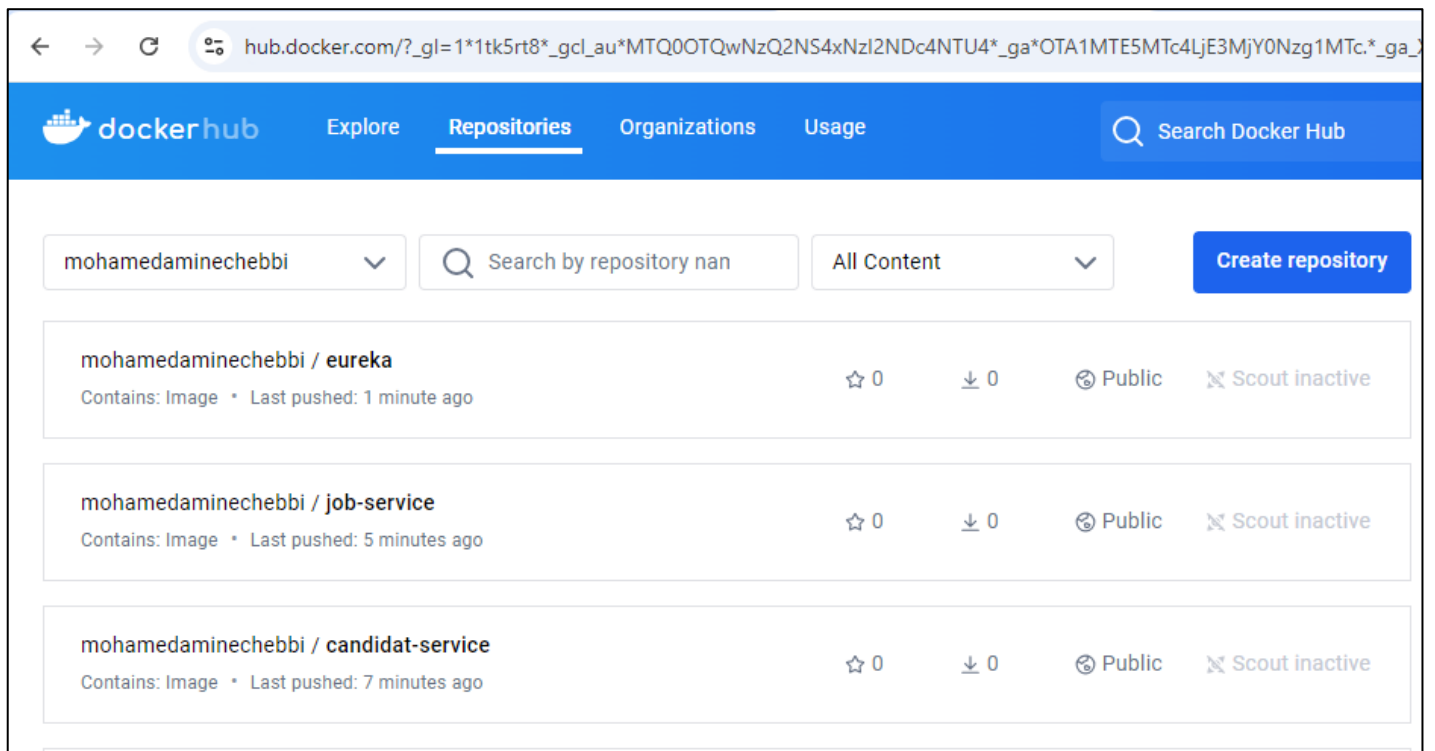
```
C:\Users\Mohamed Amine\Documents\workspace-spring-tool-suite-4-4.24.0.RELEASE_2>docker-compose build
time="2024-10-12T23:24:45+01:00" level=warning msg="C:\\Users\\Mohamed Amine\\Documents\\workspace-spring-tool-suite-4-4.24.0.RELEASE_2\\d
ocker-compose.yml: 'version' is obsolete"
[+] Building 8.0s (20/20) FINISHED
=> [discovery-service internal] load build definition from dockerfile
=> => transferring dockerfile: 142B
=> [discovery-service internal] load metadata for docker.io/library/openjdk:17
=> [discovery-service auth] library/openjdk:pull token for registry-1.docker.io
=> [discovery-service internal] load .dockerignore
=> => transferring context: 2B
=> [discovery-service internal] load build context
=> => transferring context: 56.76MB
=> [discovery-service 1/2] FROM docker.io/library/openjdk:17@sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaeaf9f87662e7b3f38cb7
=> CACHED [discovery-service 2/2] ADD target/eureka.jar eureka.jar
=> [discovery-service] exporting to image
=> => exporting layers
=> => writing image sha256:bad216aa9f06341d5c9b578fcd09f08e33c3b1e2ce0410b2070f0d530484f079
=> => naming to docker.io/mohamedaminechebbi/eureka:latest
=> [job-service internal] load build definition from dockerfile
=> => transferring dockerfile: 147B
```

## 2- docker compose push

```
C:\Users\Mohamed Amine\Documents\workspace-spring-tool-suite-4-4.24.0.RELEASE_2>docker-compose push
time="2024-10-12T23:24:56+01:00" level=warning msg="C:\\Users\\Mohamed Amine\\Documents\\workspace-spring-tool-suite-4-4.24.0.RELEASE_2\\d
ocker-compose.yml: 'version' is obsolete"
[+] Pushing 1/21
[+] Pushing 1/21Skipped 0.0s
  ✓docker-mysql Skipped 0.0s
[+] Pushing 1/21edaminechebbi/candidat-service:latest: 804364b895f9 Preparing 1.6s
  ✓docker-mysql Skipped 0.0s
[+] Pushing 1/21edaminechebbi/candidat-service:latest: 804364b895f9 Preparing 1.7s
  ✓docker-mysql Skipped 0.0s
[+] Pushing 1/21edaminechebbi/candidat-service:latest: 804364b895f9 Preparing 1.8s
  ✓docker-mysql Skipped 0.0s
- Pushing mohamedaminechebbi/candidat-service:latest: 804364b895f9 Pushing [> 524.8
[+] Pushing 1/21edaminechebbi/candidat-service:latest: 6b5aaff44254 Preparing 1.8s
  ✓docker-mysql Skipped 0.0s
```

**NB** : Il est possible que l'exécution de la deuxième commande soit plus longue que d'habitude !

En fin vous pouvez vérifier la création des images sur docker hub.



The screenshot shows the Docker Hub interface for the user 'mohamedaminechebbi'. The page displays three repositories: 'eureka', 'job-service', and 'candidat-service'. Each repository entry includes the repository name, a star icon with a count of 0, a download icon with a count of 0, the visibility status 'Public', and a 'Scout inactive' status. The 'eureka' repository was last pushed 1 minute ago, 'job-service' was last pushed 5 minutes ago, and 'candidat-service' was last pushed 7 minutes ago. The interface includes a search bar, a 'Create repository' button, and a navigation menu with 'Explore', 'Repositories', 'Organizations', and 'Usage'.