TRAVAUX PRATIQUES
DOCKER : REGISTRY ET GESTION DES IMAGES

I. INTRODUCTION

Dans ce TP, voir comment gérer les images que nous avons créé pour le mise à disposition dans notre infrastructure

II. GESTION DU CYCLE D'UNE IMAGE DOCKER

II.1. **gestion des tag d'une image.** les différentes version d'un image peuvent être modifiée. Par défaut, quand vous créer une image elle porte le tag **latest**.

Quand vous avez créer cette image à l'aide d'un Dockerfile ou à l'aide d'un commit d'un conteneur, vous pouvez aussi préciser un nom de repository et un tag, créant une image identifiée par

```
repository/image_name:tag
```

par défaut, le tag donné étant **latest** et il n'y a pas de repository. Pour créer un nouveau tag ou déplacer une image vers un nouveau regpository, vous pouvez utiliser la commande

```
docker image tag #repos_src/image_name_src:tag_src #repo_dest/image_name_dets:tag_dest
```

permet de créer une nouvelle image dans le repository de destination, avec un nouveau tag. Le repository est facultatif.

Par exemple, vous pouvez créer une version personnelle d'un image alpine de la façon suivante

```
docker image tag alpine:latest test/alpine_perso:1.0
```

```
nono@hikaru16:~/dockerfile_exemple/ex1$ docker tag alpine:latest test/alipne_perso:1.0
nono@hikaru16:~/dockerfile_exemple/ex1$ docker image ls
REPOSITORY          TAG       IMAGE ID       CREATED       SIZE
web                 1.0       b97dcaa2f4d4   4 days ago    142MB
alpine              latest    b2aa39c304c2   3 weeks ago   7.05MB
test/alipne_perso   1.0       b2aa39c304c2   3 weeks ago   7.05MB
nono@hikaru16:~/dockerfile_exemple/ex1$
```

FIGURE 1. exemple de création de tag

Vous pouvez vous rendre compte sur l'illustration 1 que l'ID de l'image alpine et de l'image que nous venons de créer sont les mêmes. Normal puisque nous avons simplement créer un duplicata de l'image alpine avec un nouveau repository, nom et tag. Vous allez donc créer un nouveau tag d'une de vos images existantes.

II.2. **Sauvegarde et transfert manuel des images.** Suite au précédent TP, vous avez pu créer différentes images à l'aide de Dockerfile. Je partirais du principe que vous avez créer une image nommée web. La sauvegarde manuelle peut être utilisée lorsque pour des raisons de sécurité, la machine sur laquelle vous travailler ou sur laquelle vous souhaitez déployer le service n'a pas d'accès internet. Dans ce cas, il faut récupérer l'image sur un support de transport. Pour cela nous allons faire une sauvegarde de l'image créée à l'aide de la commande

```
docker save - o path/filename #nom_image:tag
```

illustré par la figure 2

```
nono@hikaru16:~/dockerfile_exemple/ex1$ docker save -o alpine_perso.tar test/alipne_perso:1.0
nono@hikaru16:~/dockerfile_exemple/ex1$ ls
alpine_perso.tar  bonjour.txt  Dockerfile
nono@hikaru16:~/dockerfile_exemple/ex1$ ls -la alpine_perso.tar
-rw------- 1 nono nono 7347200  5 mars  16:01 alpine_perso.tar
nono@hikaru16:~/dockerfile_exemple/ex1$ ls -lah alpine_perso.tar
-rw------- 1 nono nono 7,1M  5 mars  16:01 alpine_perso.tar
nono@hikaru16:~/dockerfile_exemple/ex1$
```

FIGURE 2. exemple de sauvegarde d'une image

Vous pouvez remarquer que le fichier ainsi obtenu à la même taille que le fichier image listé par **docker image ls** Une fois ce fichier crée, vous pouvez le charge sur un support externe afin de le transmettre. Pour récupérer cette image dans votre cache local, vous allez utiliser la commande

```
4 docker load -i path/filename
```

comme illustré sur la figure 3

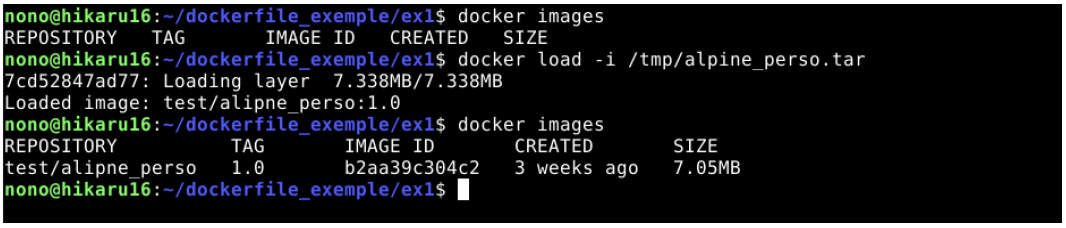

FIGURE 3. exemple de chargement d'une image à partir d'un fichier

II.3. **enregistrement sur un registry public : hub docker.** Vous pouvez sauvegarder vos images sur une registry publique. Pour cela nous allons utiliser le hub docker. vous pouvez créer un compte gratuit pour créer des repository public. Une fois la création de votre compte faite et que vous vous soyez connecté, vous allez pouvoir créer un nouveau repository. Pour cela, utiliser le bouton repository. vous arrivez sur la page illustrée figure 5

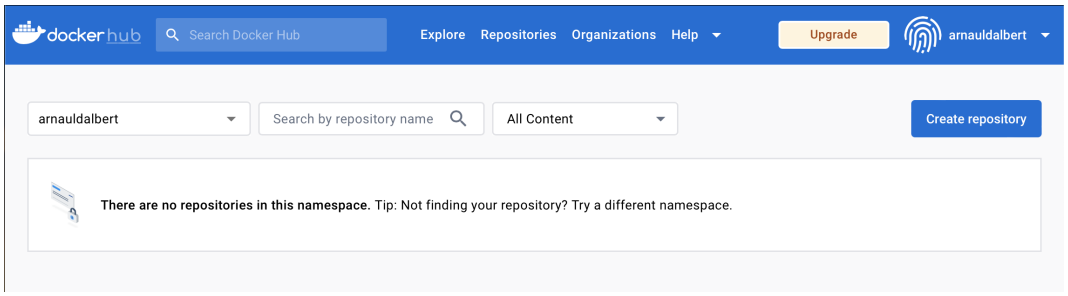

FIGURE 4. page des repository personnel sur le hub docker

En utilisant le bouton create repository, vous atteignez la page de création de vos repository illustrée figure **??** . Avec un compte gratuit vous ne pouvez avoir qu'un repository privé et autant de repository publics que souhaité. Créer votre premier repository, nommé test

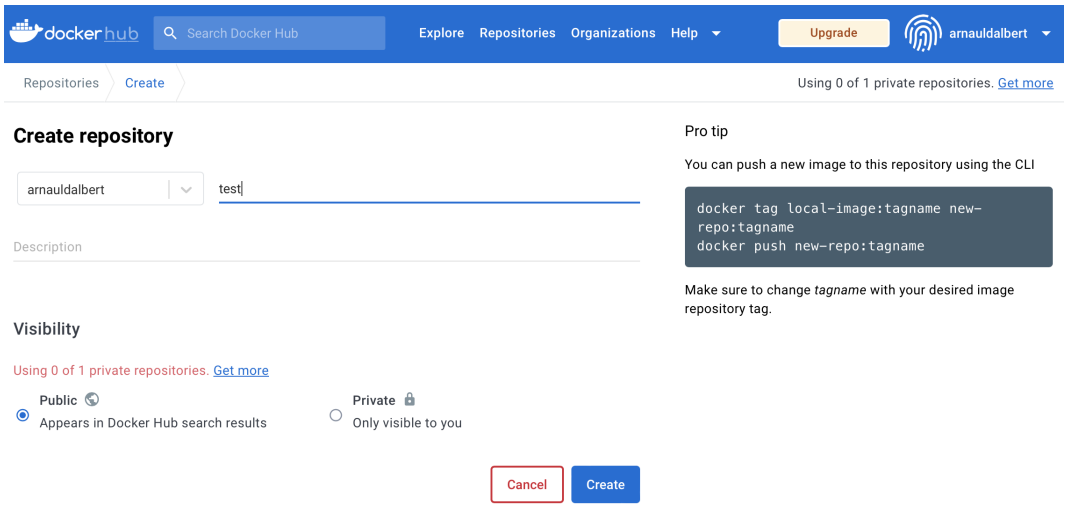

FIGURE 5. page de création

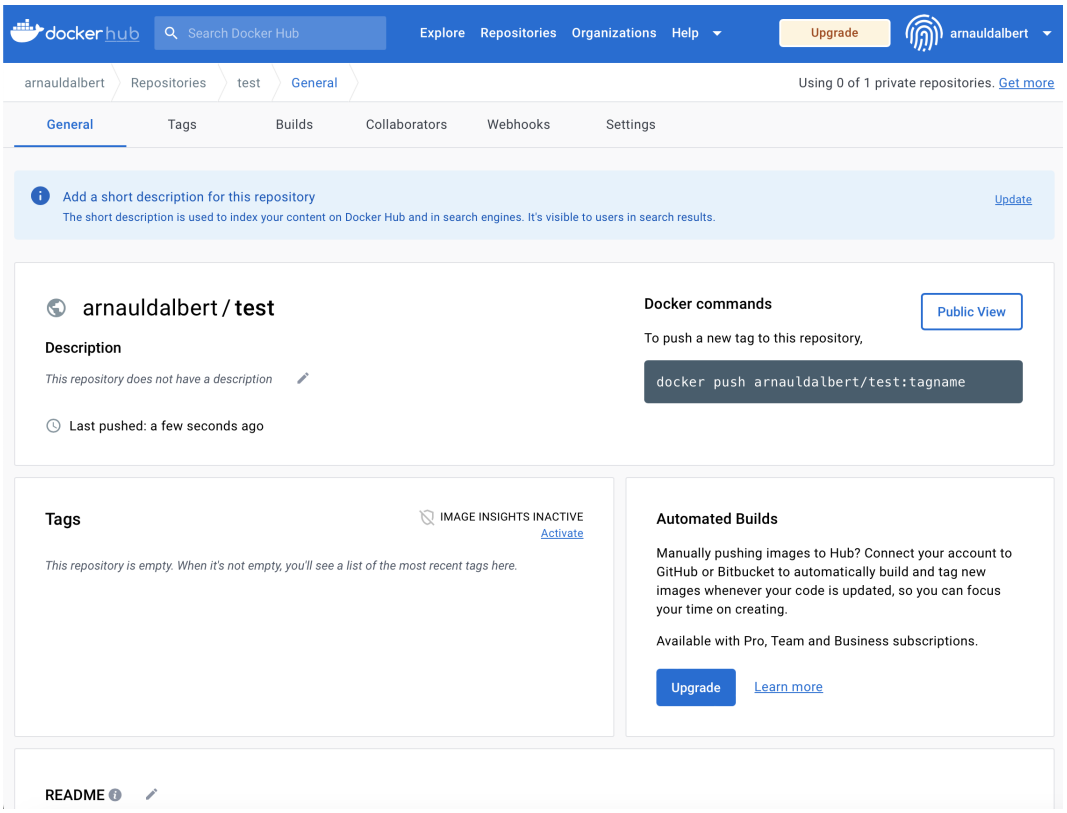

FIGURE 6. page du repository test

Une fois créer, vous arrivez sur la page de détail du repository en question illustrée figure 6

A présent vous allez pouvoir déposer de nouvelle image dans ce repository. Pour cela vous allez utlisez la commande

```
5 docker login -u #username [server / rien]
```

Par défaut c'est sur le hub.docker.com que vous allez vous connecter. Dans ce cadre, vous pourrez aussi utiliser un token pour vous connectez, plutôt que votre mot de passe. Vous suivrez la procédure indiquée par la documentation docker (https ://docs.docker.com/docker-hub/access-tokens/). Pour vous connecter sur un autre serveur, il faudra le préciser à la fin de la commande **docker login**.

Une fois votre compte connecté, vous pourrez pousser une image vers votre repository avec la commande.

```
6 docker push compte/repository/image_name:tag
```

Dans le cas de l'exemple, cette commande sera

```
7 docker push arnauldalbert/test:1.0
```

référençant l'image *arnauldalbert/test :1.0* apparaissant dans le liste des images illustrée sur la figure 7

FIGURE 7. page du repository test

II.4. **utilisation d'un registry local.** Dans le cadre d'un développement interne à l'entreprise, nous pouvons choisir de ne pas publier, même de façon privé sur un registry public. Il est alors intéressant de créer un registry local sur lequel stocker les images différentes version d'image dans la logique d'un chaine d'intégration et de développement continu (CI/CD).

La construction d'un registry local est basé sur une image docker, l'image registry. Le conteneur démarré avec cette image, doit exposer son port 5000 sur le port 5000 de l'hôte. Nous allons aussi spécifier que ce conteneur doit toujours être actif et nous allons utiliser un volume pour persister les données des images

```
docker pull registry
docker run -d -p 5000:5000 --restart=always --volume /data/:/var/lib/registry:rw --name
    myregistry registry
```

Le registry n'étant pas sécurisé, il faut préciser que nous allons utiliser un registry non sécurisé, sinon, la commande **docker push** ne fonctionnera pas car elle utilise https pour connecter le registry. Pour indiquer à docker un registry non sécurisé, il faut suivre ajouter dans le fichier /etc/docker/daemon.json (à créer s'il n'existe pas) les lignes suivantes

```
{
  "insecure-registries" : ["http://hostname:5000"]
}
```

où il faut remplacer le hostname par l'adresse IP de la machine si aucun nom d'hôte n'est défini. Il faut ensuite redémarrer le daemon docker.

```
sudo systemctl restart docker
```

et ainsi vous pouvez à présent créer des tag pour le registry local puis pousser les images.

```
docker tag #image_src hostname:5000/image_name:tag
docker push http://hostname:5000/image_name:tag
```

III. conclusion

Nous venons de voir comment mettre en place des systèmes de sauvegarde des images développées. Ces solutions constitueront une brique des chaines d'intégration et de développement continus