esiea

Projet Virtualisation

PANDAREDOUTIL Kevin

SAAD HAMIT Mahamat Haroun

Groupe : 48

1. Introduction

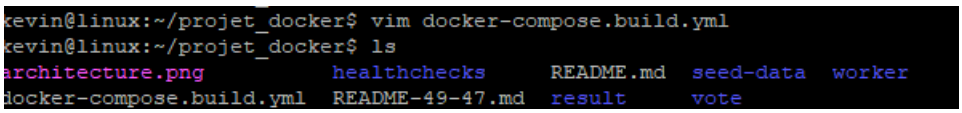
Dans le cadre de notre cursus du cycle ingénieur. Nous sommes amenés à réaliser un projet de virtualisation en utilisant docker et esxi. Le but est de déployer des conteneurs Docker pour l'installation d'une application. Ce rapport détaille les étapes nécessaires pour ce déploiement avec des captures d'écran.

2. Mise en place et déploiement des conteneurs

2.1 Création du fichier docker-compose.build.yml

Afin de commencer le projet dans de bonnes conditions nous créons un dossier "Projet docker" lesquels nous allons mettre notre projet docker.

Nous créons ensuite un fichier "docker-compose.build.yml". Ce fichier sera responsable de la construction des images d'application à partir du contenu Dockerfile fourni sur git.



En fonction des éléments données dans le git, nous remplissions ce fichier en spécifiant le nom de l'image à construire et le chemin du dockerfile associé. Voici une première capture qui montre la construction des images.

```
services:
  worker:
    build:
      context: ./worker
      dockerfile: Dockerfile

  vote:
    build:
      context: ./vote
      dockerfile: Dockerfile

  seed-data:
    build:
      context: ./seed-data
      dockerfile: Dockerfile

  result:
    build:
      context: ./result
      dockerfile: Dockerfile
```

2.2 Construction des images

Après avoir mise en place le fichier docker-compose.build.yml. Nous construisons les images via la commande suivante :

```
kevin@linux:~/projet_docker$ docker compose -f docker-compose.build.yml build
```

Nous pouvons ensuite voir les images construites :

NOTE : Cette capture a été prise à la fin du projet car nous avons rencontré quelques problèmes.

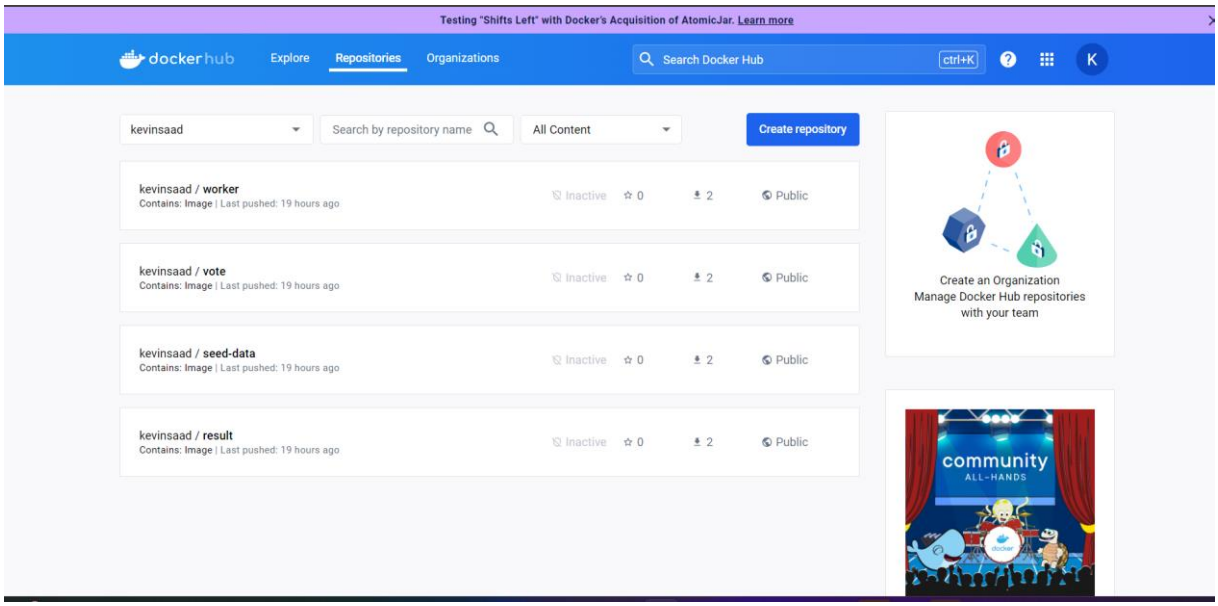
```
kevin@linux:~/projet_docker$ docker images
REPOSITORY                TAG         IMAGE ID       CREATED        SIZE
kevinsaad/seed-data       latest      ec5f56fc199d   40 hours ago   129MB
projet_docker-seed-data   latest      ec5f56fc199d   40 hours ago   129MB
localhost:5000/seed-data  latest      ec5f56fc199d   40 hours ago   129MB
kevinsaad/worker          latest      2f4bd48d6d25   2 days ago     194MB
projet_docker-worker      latest      2f4bd48d6d25   2 days ago     194MB
localhost:5000/worker     latest      2f4bd48d6d25   2 days ago     194MB
kevinsaad/result          latest      9c71b9797b52   2 days ago     224MB
projet_docker-result      latest      9c71b9797b52   2 days ago     224MB
localhost:5000/result     latest      9c71b9797b52   2 days ago     224MB
kevinsaad/vote            latest      8fc288d5f6c4   2 days ago     154MB
projet_docker-vote        latest      8fc288d5f6c4   2 days ago     154MB
localhost:5000/vote       latest      8fc288d5f6c4   2 days ago     154MB
postgres                  15-alpine   d492c6ccb0d1   2 weeks ago    240MB
adminer                   latest      8485d1424e61   2 weeks ago    250MB
redis                     latest      e40e2763392d   4 weeks ago    138MB
registry                  2           909c3ff012b7   5 weeks ago    25.4MB
```

2.3 Publication des images sur docker hub

Après création des images nous les publions sur docker hub. Voici un exemple avec l'image vote.

```
kevin@linux:~/projet_docker$ docker tag projet_docker-seed-data:latest kevinsaad/seed-data:latest
kevin@linux:~/projet_docker$ docker push kevinsaad/seed-data:latest
The push refers to repository [docker.io/kevinsaad/seed-data]
54a94610d345: Pushed
6072e2dcebc6: Pushed
6f126d59d8ab: Pushed
7595aaf67536: Pushed
4cec408bacee: Mounted from library/python
a1e3c54d75a8: Pushed
661ecc6e457f: Mounted from library/python
384858ccd7ef: Pushed
7292cf786aa8: Mounted from kevinsaad/result
latest: digest: sha256:67a68a059afada44b4816f042db09103e5786d028f97ad2d4186aacec2d317f3 size: 2203
kevin@linux:~/projet_docker$
```

Nous nous connectons à docker hub et nous voyons que les images ont été bien téléchargés.

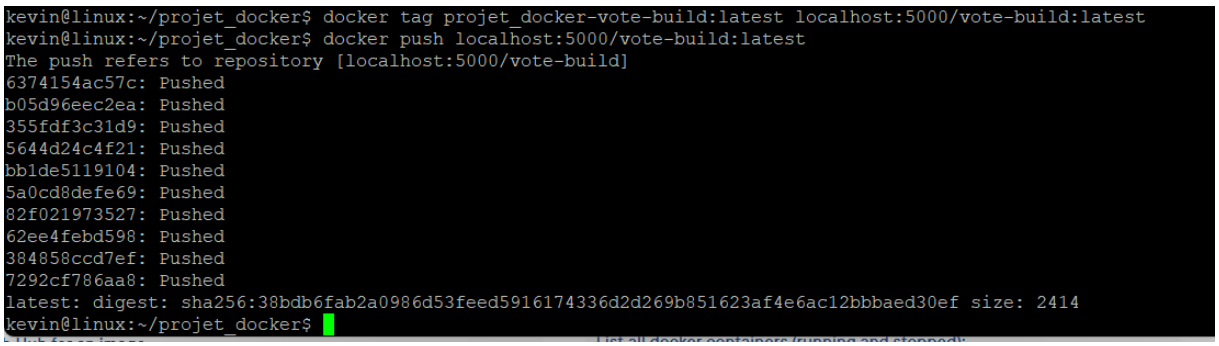


2.4 Publication des images sur un registre privé

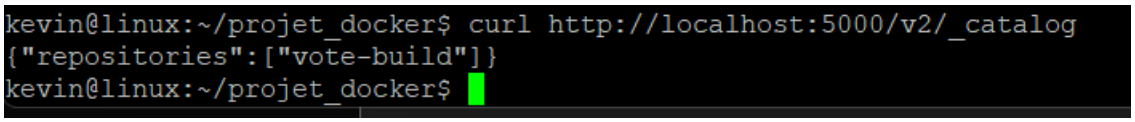
Démarrage du registre privé

```
kevin@linux:~/projet_docker$ docker run -d -p 5000:5000 --restart=always --name registry registry:2
```

D’abord nous faisons un tag de l’image puis nous faisons un push vers le chemin de notre registre privé. Dans notre cas c’est le localhost.



Vérification si l’image a été bien publié dans le registre privé :



2.5 Creation du fichier docker-compose.yml

Ce fichier est utilisé pour définir l'ensemble complet de services, de configurations réseau, de volumes, et d'autres paramètres nécessaires pour lancer et exécuter les conteneurs de notre application. Voici un extrait de notre fichier docker-compose.yml. Nous avons bien changé le réseau en “cats-or-dogs-network“.

kevin@linux: ~/projet_docker

```yaml
version: '3.1'

services:
  worker:
    image: localhost:5000/worker:latest
    depends_on:
      - redis
      - db
    networks:
      - cats-or-dogs-network
    environment:
      POSTGRES_USER: admin
      POSTGRES_PASSWORD: test

  vote:
    image: localhost:5000/vote:latest
    volumes:
      - ./vote:/usr/local/app
    ports:
      - "5002:80"
    networks:
      - cats-or-dogs-network
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost"]
      interval: 15s
      timeout: 5s
      retries: 3
      start_period: 10s

  seed-data:
    image: localhost:5000/seed-data:latest
    depends_on:
      - vote
    restart: "no"
    networks:
```

2.6 Déploiement des conteneurs

Après avoir créé le fichier docker-compose.yml, nous effectuons le déploiement des conteneurs.

```
kevin@linux:~/projet_docker$ docker compose up -d
[+] Running 1/1
 ✔ Network projet_docker_cats-or-dogs-network  Created
[+] Running 9/9et_docker_default              Creating
 ✔ Network projet_docker_cats-or-dogs-network  Created
 ✔ Network projet_docker_default               Created
 ✔ Volume "projet_docker_db-data"              Created
 ✔ Container projet_docker-redis-1             Started
 ✔ Container projet_docker-vote-1              Started
 ✔ Container projet_docker-db-1                Started
 ✔ Container projet_docker-result-1            Started
 ✔ Container projet_docker-worker-1            Started
 ✔ Container projet_docker-seed-data-1         Started
```

3. Conclusion

Ce projet illustre un exemple basique d'application distribuée avec Docker, mettant en en pratique les aspects liés à la construction d'images, à la configuration des services, et à l'utilisation de différents réseaux. Il fournit une base pour comprendre comment Docker peut être utilisé pour mettre en place des applications complexes à l'aide de conteneurs.