Projet final Docker

Table des matières

Binôme :

Rayan KHALFOUN

Nabil HATRI

1. Résumé du projet - HumansBestFriend -:

Contexte :
Développement d'une application distribuée "HumansBestFriend", implémentée avec plusieurs conteneurs Docker sur une machine virtuelle équipée de Docker et Docker Compose.

Technologies :
Langages : Python, Node.js, .NET

Messagerie : Redis

Stockage de données : Postgres

Tâches Clés :
Fichier docker-compose.build.yml : Création pour la construction d'images à partir des Dockerfiles existants.

Service Worker : Fonctionne avec Redis et Postgres, utilisant l'image Docker .NET.

Service Vote : Liaison d'un volume au /usr/local/app du conteneur, écoute sur le port 80 (exposable sur 5002), basé sur l'image Docker Python.

Service Seed-Data : Opère dans le réseau front-tier, utilise l'image Docker Python avec apache bench.

Service Result : Port interne 80, exposé sur le port 5001, avec l'image Docker Node.js.

Base de Données Postgres : Utilisation de l'image postgres:latest.

Service Redis : Mise en place de l'architecture Redis.

Architecture de l'Application :
- Interface web Python pour le vote.
- Redis pour la collecte des votes
- Worker .NET pour traitement et stockage des votes dans Postgres.
- Interface web Node.js pour afficher les résultats en temps réel.

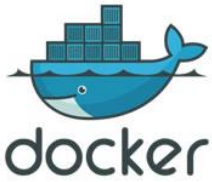
Directives Supplémentaires :
Un vote unique par navigateur client.

Documentation et soumission via un dépôt GitHub public avec un fichier SUBMISSION.md détaillé.

2. Architecture backEnd :

Nous avons commencé le projet par installer une machine virtuelle, nous avons opté pour **Virtual Box** car c'est gratuit, et assez puissant pour résoudre notre problématique.

Une fois Virtual box installé, nous avons téléchargé une image iso de **Ubuntu serveur** version 22.04.3, puis nous avons monté l'image disque et procédé à l'installation de **Docker.**



Pour travailler de manière plus claire, nous avons choisi de passer par l'invite de commande de notre machine locale, pour cela, nous nous sommes connectés à la machine virtuelle en utilisant **ssh,** voici comment nous avons procéder :

```
rayanus@mbp-de-rayan ~ % ssh user@192.168.1.27
The authenticity of host '192.168.1.27 (192.168.1.27)' can't be established.
ED25519 key fingerprint is SHA256:nNEwHzZs/NacquXGa1FvU8si2fMTYe6Tk97kNjiYHeE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.27' (ED25519) to the list of known hosts.
user@192.168.1.27's password:
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of jeu. 21 déc. 2023 13:29:30 UTC

  System load:              0.0
  Usage of /:               50.7% of 9.75GB
  Memory usage:             13%
  Swap usage:               0%
  Processes:                109
  Users logged in:          1
  IPv4 address for docker0: 172.17.0.1
  IPv4 address for enp0s3:  192.168.1.27
  IPv6 address for enp0s3:  2a01:cb04:629:8000:a00:27ff:fe5f:f76


La maintenance de sécurité étendue pour Applications n'est pas activée.

44 mises à jour peuvent être appliquées immédiatement.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Thu Dec 21 13:13:55 2023
```

Initialisation de l'environnement :

Notre machine virtuelle est désormais fonctionnelle, nous devons récupérer toute l'architecture du projet qui est disponible sur **Github,** pour cela nous utilisons la commande suivante :

```
user@vm:~/salut$ git clone https://github.com/pascalito007/ynov-resources.git
Cloning into 'ynov-resources'...
remote: Enumerating objects: 402, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 402 (delta 4), reused 20 (delta 2), pack-reused 376
Receiving objects: 100% (402/402), 11.55 MiB | 13.72 MiB/s, done.
Resolving deltas: 100% (59/59), done.
user@vm:~/salut$ cd ynov-resources/
2023/ .git/
user@vm:~/salut$ cd ynov-resources/
2023/ .git/
user@vm:~/salut$ cd ynov-resources/2023/m2/
dataeng/ webdev/
user@vm:~/salut$ cd ynov-resources/2023/m2/
dataeng/ webdev/
user@vm:~/salut$ cd ynov-resources/2023/m2/dataeng/humans-best-friend/
user@vm:~/salut/ynov-resources/2023/m2/dataeng/humans-best-friend$
```

Fichier docker-compose.build.yml :

C'est un fichier essentiel dans notre projet car il permet de créer toutes les images dont on a besoin pour générer nos différents containers, voici le contenu du fichier :

```yaml
version: '3'

services:
  worker:
    build:
      context: ./worker
      dockerfile: Dockerfile
    depends_on:
      redis:
        condition: service_healthy
      db:
        condition: service_healthy
    networks:
      - humansbestfriend-network

  vote:
    build:
      context: ./vote
      dockerfile: Dockerfile
    depends_on:
      redis:
        condition: service_healthy
    networks:
      - humansbestfriend-network

  seed-data:
    build:
      context: ./seed-data
      dockerfile: Dockerfile
    depends_on:
      vote:
        condition: service_healthy
    networks:
      - humansbestfriend-network

  result:
    build:
      context: ./result
      dockerfile: Dockerfile
    depends_on:
      db:
        condition: service_healthy
    networks:
      - humansbestfriend-network

  redis:
    image: "redis:latest"
    networks:
      - humansbestfriend-network

  db:
    image: "postgres:latest"
    networks:
      - humansbestfriend-network

networks:
  humansbestfriend-network:
```

Exécution de la commande du fichier build:

```
[user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker-compose -f Docker-compose.build.yml build
redis uses an image, skipping
db uses an image, skipping
Building vote
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon   12.8kB
Step 1/13 : FROM python:3.11-slim AS base
 ---> dd150e5400f1
Step 2/13 : RUN apt-get update &&     apt-get install -y --no-install-recommends curl &&     rm -rf /var/lib/apt/lists/*
 ---> Using cache
 ---> 3e404d1daa10
Step 3/13 : WORKDIR /usr/local/app
 ---> Using cache
 ---> 991b2221161e
Step 4/13 : COPY requirements.txt ./requirements.txt
 ---> Using cache
 ---> 4d8ae102efb4
Step 5/13 : RUN pip install --no-cache-dir -r requirements.txt
 ---> Using cache
```

Création du registre :

```
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker pull registry
Using default tag: latest
latest: Pulling from library/registry
c926b61bad3b: Pull complete
5501dced60f8: Pull complete
e875fe5e6b9c: Pull complete
21f4bf2f86f9: Pull complete
98513cca25bb: Pull complete
Digest: sha256:0a182cb82c93939407967d6d71d6caf11dcef0e5689c6afe2d60518e3b34ab86
Status: Downloaded newer image for registry:latest
docker.io/library/registry:latest
```

Création pour chaque image un tag associé :

```
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker tag humans-best-friend_worker localhost:5000/humans-best-friend_worker
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker tag humans-best-friend_result localhost:5000/humans-best-friend_result
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker tag humans-best-friend_seed-data localhost:5000/humans-best-friend_seed-data
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker tag humans-best-friend_vote localhost:5000/humans-best-friend_vote
```

Exécution du container Regitry :

Dans cette étape nous exécutons le container Registry avant d'y **push** nos images

```
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo  docker run -d -p 5000:5000 --restart=always --name registry registry:2
Unable to find image 'registry:2' locally
2: Pulling from library/registry
Digest: sha256:0a182cb82c93939407967d6d71d6caf11dcef0e5689c6afe2d60518e3b34ab86
Status: Downloaded newer image for registry:2
af950663f222f4beb2d1be20ab71ffff007452fa8432b3ef5abdba1d828fcc15
```

Push des images dans le Registry:

```
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker push localhost:5000/humans-best-friend_worker
Using default tag: latest
The push refers to repository [localhost:5000/humans-best-friend_worker]
e8fa2f6ac320: Pushed
4fe999b18f80: Pushed
5ee537aacfb3: Pushed
7c73e219acdd: Pushed
d47be6633206: Pushed
e6cd39d4cea4: Pushed
latest: digest: sha256:2a7b23c96717d03c232301a2ab57d560246e88a8a66c5463ecd070ea9b0a0b78 size: 1577
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker push localhost:5000/humans-best-friend_result
Using default tag: latest
The push refers to repository [localhost:5000/humans-best-friend_result]
a36c14b6e7fb: Pushed
d67b434556af: Pushed
6bda07ce6c2a: Pushed
911610dfeac5: Pushed
e5228d3c7640: Pushed
a3e13afd3574: Pushed
69ea00d1a724: Pushed
88a1fe82f870: Pushed
fa348aca89c0: Pushed
6a02db9fb904: Pushed
7292cf786aa8: Pushed
latest: digest: sha256:36f611204f69204f890670e4a7544b73e9df496c9db399462900bcdedc165a2e size: 2625
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker push localhost:5000/humans-best-friend_seed-data
Using default tag: latest
The push refers to repository [localhost:5000/humans-best-friend_seed-data]
cc7b2c24ea2f: Pushed
8b212d60ad43: Pushed
b504406e825e: Pushed
a1874f8da75f: Pushed
4cec408bacee: Pushed
a1e3c54d75a8: Pushed
661ecc6e457f: Pushed
384858ccd7ef: Pushed
7292cf786aa8: Mounted from humans-best-friend_result
latest: digest: sha256:3beeb6b79d4a7237a0516aad503540a849522fee0aec870fe00f1ca11e21f428 size: 2203
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker push localhost:5000/humans-best-friend_vote
Using default tag: latest
The push refers to repository [localhost:5000/humans-best-friend_vote]
ae6af372e835: Pushed
edeae0ca1471: Pushed
e8c62fec92f3: Pushed
87ed06bd756a: Pushed
b33eb41d87f9: Pushed
5a0cd8defe69: Pushed
82f021973527: Pushed
62ee4febd598: Pushed
384858ccd7ef: Mounted from humans-best-friend_seed-data
7292cf786aa8: Mounted from humans-best-friend_seed-data
latest: digest: sha256:03c1803a6517ff78f4270a892704be0d2c54fd3fd064d9a00f527d0a5916b755 size: 2414
```

Vérification des images dans le Regitry avec CURL :

```
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ curl localhost:5000/v2/_catalog
{"repositories":["humans-best-friend_result","humans-best-friend_seed-data","humans-best-friend_vote","humans-best-friend_worker"]}
```

Fichier docker-compose.yml :

Voici le contenu de notre fichier compose :

```yaml
version: "3"

services:
  vote:
    image: localhost:5000/humans-best-friend_vote
    depends_on:
      redis:
        condition: service_healthy
    ports:
      - "5001:80"
    networks:
      - humansbestfriend-network

  seed-data:
    image: localhost:5000/humans-best-friend_seed-data
    depends_on:
      - vote
    restart: "no"
    networks:
      - humansbestfriend-network

  result:
    image: localhost:5000/humans-best-friend_result
    depends_on:
      db:
        condition: service_healthy
    ports:
      - "5002:80"
    networks:
      - humansbestfriend-network


  worker:
    image: localhost:5000/humans-best-friend_worker
    depends_on:
      redis:
        condition: service_healthy
      db:
        condition: service_healthy
    networks:
      - humansbestfriend-network

  redis:
    image: redis:alpine
    volumes:
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/redis.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

  db:
    image: postgres:15-alpine
    environment:
      POSTGRES_USER: "postgres"
      POSTGRES_PASSWORD: "postgres"
    volumes:
      - "db-data:/var/lib/postgresql/data"
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/postgres.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

volumes:
  db-data:

networks:
  humansbestfriend-network:
```

Exécution du docker compose :

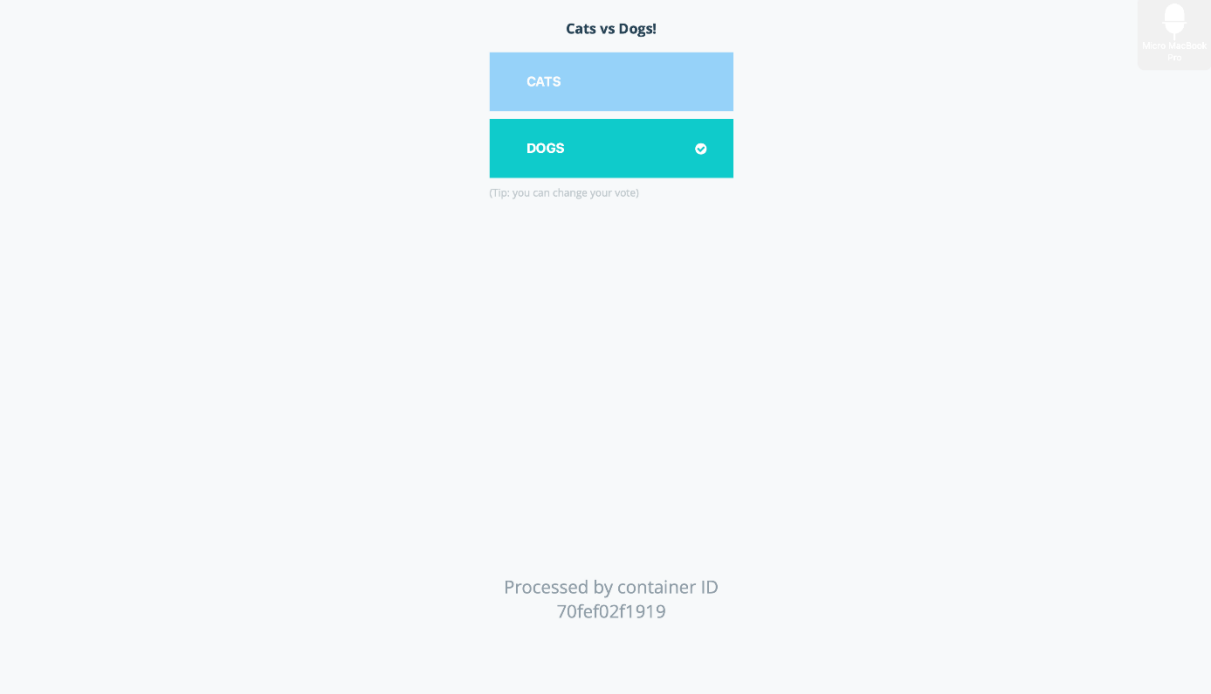
Nous lançons chacun de nos containers avec les paramètres prédéfinis :

```
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ sudo docker-compose up -d
Creating humans-best-friend_redis_1 ... done
Creating humans-best-friend_db_1    ... done
Creating humans-best-friend_result_1    ... done
Creating humans-best-friend_vote_1   ... done
Creating humans-best-friend_worker_1 ... done
Creating humans-best-friend_seed-data_1 ... done
user@vm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ ip a
```

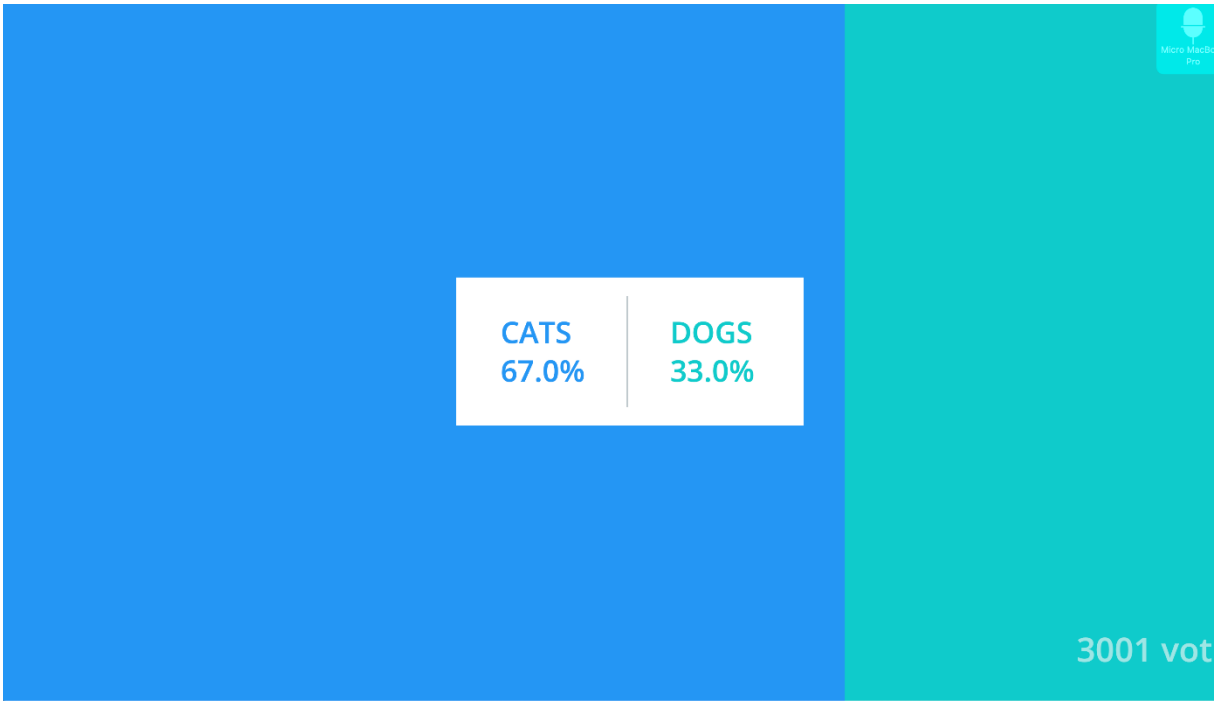
3. FrontEnd :

Voici ci-dessous une capture d'écran de notre navigateur, afin de vérifier le bon fonctionnement de notre projet :

a. Interface pour voter : 192.168.3.38 :5002 :



b. Interface pour vérifier le résultat des votes : 192.168.3.38 :5001



4. Démonstration de la Communication entre Conteneurs via IPs en utilisant PING