Equipe : Yanis Hamdaoui, Wassil Fahem

Projet Docker

Etape 1 : Connexion SSH de Gitbash vers la VM

```
wassf@LAPTOP-4E65GICU MINGW64 ~
$ ssh wassil@172.17..0.1
ssh: Could not resolve hostname 172.17..0.1: Name or service not known

wassf@LAPTOP-4E65GICU MINGW64 ~
$ ssh wassil@172.17.0.1
ssh: connect to host 172.17.0.1 port 22: Connection refused

wassf@LAPTOP-4E65GICU MINGW64 ~
$ ssh wassil@192.168.1.68
The authenticity of host '192.168.1.68 (192.168.1.68)' can't be established.
ED25519 key fingerprint is SHA256:2PbBAlFZT6MvCtJdObCKeZtPRsVHVGJ3YJfKComvJ6E.
This host key is known by the following other names/addresses:
   ~/.ssh/known_hosts:1: 172.20.10.4
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.68' (ED25519) to the list of known hosts.
wassil@192.168.1.68'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 09:18:13 UTC

  System load:                 0.03857421875
  Usage of /:                  49.8% of 11.21GB
  Memory usage:                6%
  Swap usage:                  0%
  Processes:                   123
  Users logged in:             1
  IPv4 address for docker0:    172.17.0.1
  IPv4 address for enp0s3:     192.168.1.68
  IPv6 address for enp0s3:     2a02:8428:701:ec01:a00:27ff:fe9a:400b


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

42 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 09:14:51 2023
wassil@wassilvm:~$ docker ps
CONTAINER ID   IMAGE      COMMAND    CREATED    STATUS    PORTS      NAMES
wassil@wassilvm:~$ docker ps -a
```

Etape 2 : Faire un Docker pull de Redis et Postgre**s**

```
wassil@wassilvm:~$ docker pull redis
Using default tag: latest
latest: Pulling from library/redis
af107e978371: Pull complete
b031def5f2c4: Pull complete
bf7f0c8796d3: Pull complete
e3b2691a4104: Pull complete
190b4d7a237a: Pull complete
797591c7970a: Pull complete
4f4fb700ef54: Pull complete
45ce3854ac9a: Pull complete
Digest: sha256:7f26b254f1169010a2c0e160d166f0e12fb05ceab21a1efc342e43ad397a2674
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
wassil@wassilvm:~$ docker pull postgres:15-alpine
15-alpine: Pulling from library/postgres
661ff4d9561e: Pull complete
e4a3f96ea8e5: Pull complete
0c1e2e159ea1: Pull complete
26c071a8426e: Pull complete
e9a1ba05d22c: Pull complete
efc39a79d7dc: Pull complete
72124e665f9e: Pull complete
aa569f3e770e: Pull complete
86d5fe07cb37: Pull complete
Digest: sha256:e2a22801fcab638f9491039f8257e9f719ab02e8c78c6a6f2c0349505f92dc35
Status: Downloaded newer image for postgres:15-alpine
docker.io/library/postgres:15-alpine
wassil@wassilvm:~$ docker ps -a
```

Equipe : Yanis Hamdaoui, Wassil Fahem

Etape 3 : Remettre le Registry en cas d'absence de ce dernier

```
wassil@wassilvm:~$ docker run -d -p 5000:5000 --restart always --name registry r
egistry:2
Unable to find image 'registry:2' locally
2: 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:2
bb645ab65b78b26d260f2655672e8488494d02791f6aaaf13feabe257c118b0c
```

Etape 4 : Clonage du projet GitHub dans la VM

```
wassil@wassilvm:~$ git clone https://github.com/Wassil91/ynov-resources.git
Cloning into 'ynov-resources'...
remote: Enumerating objects: 402, done.
remote: Counting objects: 100% (30/30), done.
wassil@wassilvm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ vi docker-compose.yaml
```

Etape 5 : Tag des bases de données

Pour Redis :

```
wassil@wassilvm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ docker tag redis:latest localhost:5000/redis:latest
```

Pour Postgres :

```
wassil@wassilvm:~/ynov-resources/2023/m2/dataeng/humans-best-friend$ docker tag postgres:15-alpine localhost:5000/postgres:15-alpine
```

Etape 6 : Création du Docker-compose.build.yml

Pour créer le fichier : vi docker-compose.build.yml

Equipe : Yanis Hamdaoui, Wassil Fahem

```yaml
services:
  vote:
    build:
      context: ./vote
      target: dev
    depends_on:
      redis:
        condition: service_healthy
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost"]
      interval: 15s
      timeout: 5s
      retries: 3
      start_period: 10s
    volumes:
     - ./vote:/usr/local/app
    ports:
      - "5005:80"
    networks:
      - humansbestfriend-network

  result:
    build: ./result
    # use nodemon rather than node for local dev
    entrypoint: nodemon --inspect=0.0.0.0 server.js
    depends_on:
      db:
        condition: service_healthy
    volumes:
      - ./result:/usr/local/app
    ports:
      - "5004:80"
      - "127.0.0.1:9229:9229"
    networks:
      - humansbestfriend-network

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

  redis:
    image: localhost:5000/redis:latest
    volumes:
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/redis.sh
      interval: "5s"
    networks:
      - humansbestfriend-network
```

3

```yaml
    networks:
      - humansbestfriend-network

  redis:
    image: localhost:5000/redis:latest
    volumes:
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/redis.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

  db:
    image: localhost:5000/postgres:15-alpine
    environment:
      POSTGRES_USER: "postgres_user"
      POSTGRES_PASSWORD: "postgres_pwd"
    volumes:
      - "db-data:/var/lib/postgresql/data"
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/postgres.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

  # this service runs once to seed the database with votes
  # it won't run unless you specify the "seed" profile
  # docker compose --profile seed up -d
  seed:
    build: ./seed-data
    profiles: ["seed"]
    depends_on:
      vote:
        condition: service_healthy
    networks:
      - humansbestfriend-network
    restart: "no"

volumes:
  db-data:

networks:
  humansbestfriend-network:
```
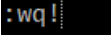
Equipe : Yanis Hamdaoui, Wassil Fahem

Etape 7 : Création du docker-compose.yml

Pour créer ce fichier : docker-compose.yml

```
wassil@wassilvm: ~/ynov-resources/2023/m2/dataeng/humans-best-friend
services:
  vote:
    image: humans-best-friend-vote
    depends_on:
      redis:
        condition: service_healthy
    ports:
      - "5005:80"
    networks:
      - humansbestfriend-network

  result:
    image: humans-best-friend-result
    depends_on:
      db:
        condition: service_healthy
    ports:
      - "5004:80"
    networks:
      - humansbestfriend-network

  worker:
    image: humans-best-friend-worker
    depends_on:
      redis:
        condition: service_healthy
      db:
        condition: service_healthy
    networks:
      - humansbestfriend-network

  redis:
    image: localhost:5000/redis:latest
    volumes:
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/redis.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

  db:
    image: localhost:5000/postgres:15-alpine
    environment:
      POSTGRES_USER: "postgres_user"
      POSTGRES_PASSWORD: "postgres_pwd"
    volumes:
      - "db-data:/var/lib/postgresql/data"
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/postgres.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

volumes:
  db-data:
```

```
      - db-data:/var/lib/postgresql/data
      - "./healthchecks:/healthchecks"
    healthcheck:
      test: /healthchecks/postgres.sh
      interval: "5s"
    networks:
      - humansbestfriend-network

volumes:
  db-data:

networks:
  humansbestfriend-network:
```


PS : `:wq!` permet de sauvegarder et de quitter sur VI

Equipe : Yanis Hamdaoui, Wassil Fahem

Etape 8 : lancement du docker build

Grace à la commande : docker compose -f docker-compose.build.yml up -d

Etape 9 : lancement du docker-compose.yml

Equipe : Yanis Hamdaoui, Wassil Fahem

Etape 10 : Check des containers démarré sous forme de liste avec :

Docker ps

Pour les containers qui sont stoppé mais pour voir ce qui ont été créé : Docker ps -a

Docker start « nom du container »: pour démarrer un container stoppé