

Projet de virtualisation



Table des matières

Projet de virtualisation	1
1. Introduction	3
2. Architecture globale du Projet.....	3
3. Docker Compose «docker-compose.yml»	5
4. Docker Compose «docker-compose.build.yml»	6
5. Résultat	8

1.Introduction

Le projet est une application distribuée basée sur une architecture de microservices, déployée dans des conteneurs Docker à l'aide de Docker Compose. Il vise à fournir une plateforme de vote en temps réel avec une gestion des résultats, le tout orchestré de manière modulaire pour faciliter le développement, le déploiement et la maintenance.

2.Architecture globale du Projet

a. Présentation de la structure du projet en microservices

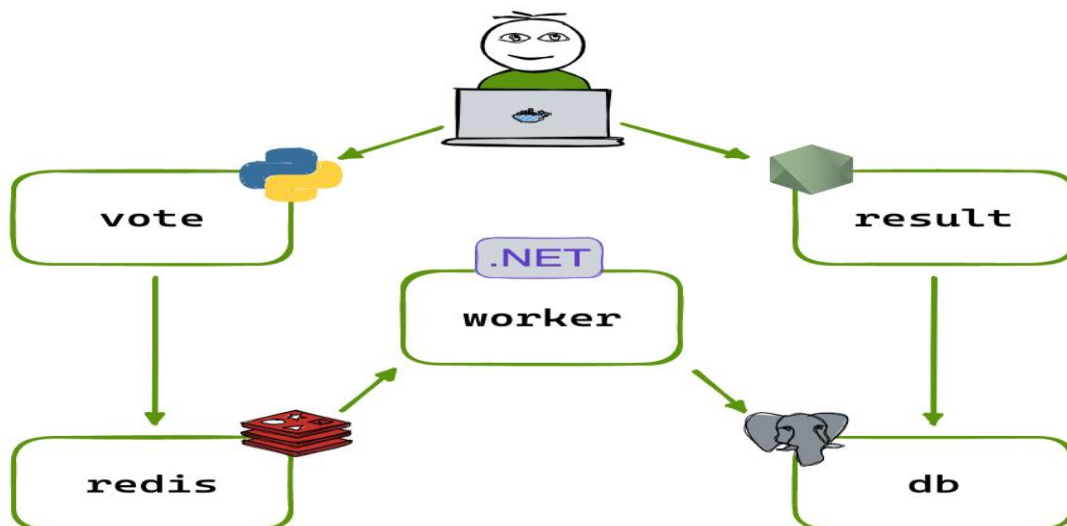


Figure 1 : Architecture globale

La structure du dossier du projet "esiea-ressources" pourrait ressembler à quelque chose de similaire à ce qui suit.

Cette structure de dossier suppose que chaque service a son propre répertoire avec son propre Dockerfile et que le fichier docker-compose.yml est à la racine du projet.

```

babacar@babacar-server:~/esiea-ressources$ tree
.
├── docker-compose.build.yml
├── docker-compose.yml
├── healthchecks
│   ├── postgres.sh
│   └── redis.sh
├── result
│   ├── docker-compose.test.yml
│   ├── Dockerfile
│   ├── package.json
│   ├── package-lock.json
│   ├── server.js
│   ├── tests
│   │   ├── Dockerfile
│   │   ├── render.js
│   │   └── tests.sh
│   └── views
│       ├── angular.min.js
│       ├── app.js
│       ├── index.html
│       ├── socket.io.js
│       ├── stylesheets
│       └── style.css
├── seed-data
│   ├── Dockerfile
│   ├── generate-votes.sh
│   └── make-data.py
├── tests
├── vote
│   ├── app.py
│   ├── Dockerfile
│   ├── requirements.txt
│   ├── static
│   │   ├── stylesheets
│   │   └── style.css
│   ├── templates
│   └── index.html
└── worker
    ├── Dockerfile
    ├── Program.cs
    └── Worker.csproj

```

13 directories, 28 files
babacar@babacar-server:~/esiea-ressources\$

Figure 2 : Structure du dossier de l'application

b. Rôle de chaque service dans l'ensemble du système

➤ Service vote :

Responsable de la gestion des votes.
 Construit à partir du répertoire ../vote/.
 Expose le port 80.
 Dépend des services redis et db.
 Appartient aux réseaux front-tier et back-tier.

➤ Service result :

Gère et affiche les résultats des votes.
 Construit à partir du répertoire ../result/.
 Expose le port 80.
 Dépend des services redis et db.
 Appartient aux réseaux front-tier et back-tier.

➤ Service worker :

Effectue des tâches en arrière-plan.
 Construit à partir du répertoire ../worker/.
 Dépend des services redis et db.
 Appartient au réseau back-tier.

➤ **Service redis :**

Utilise l'image officielle redis:alpine.

Gère la persistance des données et la communication entre services.

Appartient au réseau back-tier.

➤ **Service db (PostgreSQL) :**

Utilise l'image officielle postgres:9.4.

Configuré avec un utilisateur et un mot de passe PostgreSQL.

Persiste les données à l'aide du volume db-data.

Appartient au réseau back-tier.

3. Docker Compose «docker-compose.yml»

```
version: '2'

services:
  vote:
    build: ../vote/
    ports: ["80"]
    depends_on:
      - redis
      - db
    networks:
      - front-tier
      - back-tier

  result:
    build: ../result/
    ports: ["80"]
    depends_on:
      - redis
      - db
    networks:
      - front-tier
      - back-tier

  worker:
    build: ../worker/
    depends_on:
      - redis
      - db
    networks:
      - back-tier

  redis:
    image: redis:alpine
    networks:
      - back-tier

  db:
    image: postgres:9.4
    environment:
      POSTGRES_USER: "postgres"
      POSTGRES_PASSWORD: "postgres"
    volumes:
      - "db-data:/var/lib/postgresql/data"
    networks:
      - back-tier

volumes:
  db-data:

networks:
  front-tier:
  back-tier:
```

Le fichier docker-compose.yml décrit la configuration du projet pour Docker Compose, spécifiant les services, les réseaux, et les volumes nécessaires pour l'exécution de l'application. Cette configuration permet une gestion cohérente et facile des conteneurs dans un environnement conteneurisé. Il orchestre les différents services de l'application, spécifiant leurs dépendances, réseaux, et configurations individuelles pour garantir un déploiement cohérent et fonctionnel dans un environnement conteneurisé.

Détails du docker-compose.yml :

- Version de Docker Compose :

Utilisation de la version 2 du format docker-compose.yml.

- Services :

Chaque service est défini avec sa propre configuration.

Les services vote, result, et worker sont construits à partir de répertoires spécifiques (../vote/, ../result/, ../worker/) en utilisant leurs Dockerfiles respectifs.

Les services dépendent de redis et db, ce qui assure l'ordre de démarrage approprié.

Les services vote et result appartiennent aux réseaux front-tier et back-tier, tandis que worker appartient uniquement à back-tier.

- Service redis :

Utilise l'image officielle redis:alpine.

Appartient au réseau back-tier.

Service db (PostgreSQL) :

Utilise l'image officielle postgres:9.4.

Configure un utilisateur PostgreSQL avec un mot de passe.

Utilise un volume nommé db-data pour la persistance des données.

Appartient au réseau back-tier.

- Volumes :

Le volume nommé db-data est utilisé par le service db pour la persistance des données PostgreSQL.

- Réseaux :

front-tier : Réseau pour la communication entre les services liés à l'interface utilisateur.

back-tier : Réseau pour la communication entre les services en arrière-plan et les services de stockage de données.

4.Docker Compose «docker-compose.build.yml»

```
version: '3'

services:
  worker:
    build:
      context: ../worker/
      dockerfile: Dockerfile
    network_mode: back-tier

  vote:
    build:
      context: ../vote/
      dockerfile: Dockerfile
    network_mode: front-tier

  seed-data:
    build:
      context: ../seed-data/
      dockerfile: Dockerfile
    network_mode: front-tier

  result:
    build:
      context: ../result/
      dockerfile: Dockerfile
    network_mode: back-tier
```

Détail du fichier docker-compose.yml.

- Version de Docker Compose :

Utilisation de la version 3 du format docker-compose.yml.

Services de Construction :

Chaque service spécifie le contexte de construction et le fichier Dockerfile à utiliser.

- Service worker :

Construit à partir du ../worker/ en utilisant le Dockerfile spécifié.

Le mode réseau est configuré sur back-tier.

- Service vote :

Construit à partir du ../vote/ en utilisant le Dockerfile spécifié.

Le mode réseau est configuré sur front-tier.

- Service seed-data :

Construit à partir du ../seed-data/ en utilisant le Dockerfile spécifié.

Le mode réseau est configuré sur front-tier.

- Service result :

Construit à partir du ../result/ en utilisant le Dockerfile spécifié.

Le mode réseau est configuré sur back-tier.

Remarques :

Chaque service spécifie le contexte de construction (context) en indiquant le répertoire où se trouvent les fichiers nécessaires à la construction de l'image Docker.

Le fichier Dockerfile utilisé pour chaque service est spécifié avec le paramètre dockerfile.

Le mode réseau (network_mode) est configuré pour chaque service, indiquant à quel réseau Docker le conteneur doit se connecter. Cela est particulièrement pertinent pour l'interaction entre les services.

Ce fichier docker-compose.build.yml définit les étapes de construction des images Docker pour chaque service du projet. Il organise les différentes étapes de construction, permettant ainsi de créer les images nécessaires à partir des sources spécifiées dans chaque répertoire de service.

5. Résultat

Contenu du dossier du projet :

```
babacar@babacar-server: ~/esiea-ressources$ ls
docker-compose.build.yml  docker-compose.yml  healthchecks  result  seed-data  tests  vote  worker
```

Figure 1 : Contenu du dossier du projet

a. Construction des images & lancement des conteneurs

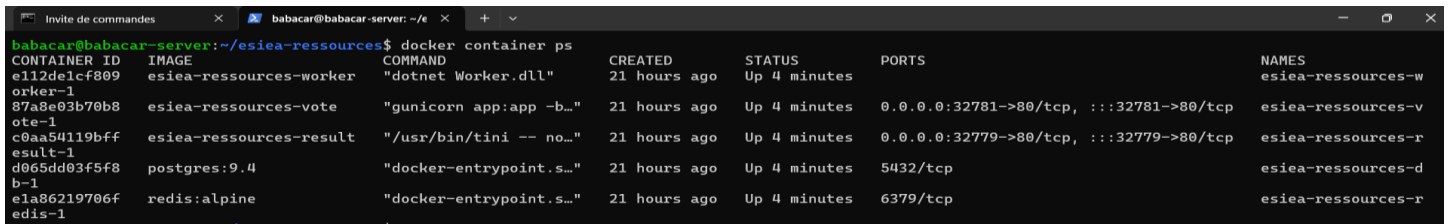
```
docker-compose up -d
```

Cette commande construira les images pour tous les services spécifiés dans le fichier `docker-compose.yml`.

```
babacar@babacar-server: ~/esiea-ressources$ docker-compose up
[+] Running 7/7
✓ Network esiea-ressources_back-tier Created 0.3s
✓ Network esiea-ressources_front-tier Created 0.2s
✓ Container esiea-ressources-redis-1 Created 0.2s
✓ Container esiea-ressources-db-1 Created 0.2s
✓ Container esiea-ressources-worker-1 Created 0.1s
✓ Container esiea-ressources-vote-1 Created 0.1s
✓ Container esiea-ressources-result-1 Created 0.2s
Attaching to esiea-ressources-db-1, esiea-ressources-redis-1, esiea-ressources-result-1, esiea-ressources-vote-1, esiea-ressources-worker-1
esiea-ressources-redis-1 | 1:C 18 Dec 2023 12:32:18.356 # WARNING Memory overcommit must be enabled! Without it, a background save or replication may fail
esiea-ressources-redis-1 | under low memory condition. Being disabled, it can also cause failures without low memory condition, see https://github.com/jemalloc/jemalloc/issues/1328.
esiea-ressources-redis-1 | To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
esiea-ressources-redis-1 | 1:C 18 Dec 2023 12:32:18.358 * o000o000o000o Redis is starting o000o000o000o
esiea-ressources-redis-1 | 1:C 18 Dec 2023 12:32:18.358 * Redis version=7.2.3, bits=64, commit=00000000, modified=0, pid=1, just started
esiea-ressources-redis-1 | 1:C 18 Dec 2023 12:32:18.358 # Warning: no config file specified, using the default config. In order to specify a config file use
esiea-ressources-redis-1 | se redis-server /path/to/redis.conf
esiea-ressources-redis-1 | 1:M 18 Dec 2023 12:32:18.362 * Increased maximum number of open files to 10032 (it was originally set to 1024).
esiea-ressources-redis-1 | 1:M 18 Dec 2023 12:32:18.362 * monotonic clock: POSIX clock_gettime
esiea-ressources-redis-1 | 1:M 18 Dec 2023 12:32:18.370 * Running mode=standalone, port=6379.
esiea-ressources-redis-1 | 1:M 18 Dec 2023 12:32:18.381 * Server initialized
esiea-ressources-redis-1 | 1:M 18 Dec 2023 12:32:18.382 * Ready to accept connections tcp
esiea-ressources-db-1 | PostgreSQL Database directory appears to contain a database; Skipping initialization
esiea-ressources-db-1 | LOG: database system was shut down at 2023-12-18 11:28:41 UTC
esiea-ressources-db-1 | LOG: MultiXact member wraparound protections are now enabled
esiea-ressources-db-1 | LOG: database system is ready to accept connections
esiea-ressources-db-1 | LOG: autovacuum launcher started
esiea-ressources-result-1 | Mon, 18 Dec 2023 12:32:22 GMT body-parser deprecated undefined extended: provide extended option at server.js:67:17
esiea-ressources-result-1 | App running on port 80
esiea-ressources-result-1 | Connected to db
esiea-ressources-vote-1 | [2023-12-18 12:32:22 +0000] [1] [INFO] Starting gunicorn 21.2.0
esiea-ressources-vote-1 | [2023-12-18 12:32:22 +0000] [1] [INFO] Listening at: http://0.0.0.0:80 (1)
esiea-ressources-vote-1 | [2023-12-18 12:32:22 +0000] [1] [INFO] Using worker: sync
esiea-ressources-vote-1 | [2023-12-18 12:32:22 +0000] [7] [INFO] Booting worker with pid: 7
esiea-ressources-vote-1 | [2023-12-18 12:32:22 +0000] [8] [INFO] Booting worker with pid: 8
esiea-ressources-vote-1 | [2023-12-18 12:32:23 +0000] [9] [INFO] Booting worker with pid: 9
esiea-ressources-vote-1 | [2023-12-18 12:32:23 +0000] [10] [INFO] Booting worker with pid: 10
esiea-ressources-worker-1 | Connected to db
esiea-ressources-worker-1 | Found redis at 172.25.0.2
esiea-ressources-worker-1 | Connecting to redis
```

Figure 2 : Construction des images et lancement des conteneurs

b. Vérification du lancement des conteneurs



```
babacar@babacar-server:~/esiea-ressources$ docker container ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
e112de1cf809	esiea-ressources-worker	"dotnet Worker.dll"	21 hours ago	Up 4 minutes		esiea-ressources-w
87a8e03b70b8	esiea-ressources-vote	"gunicorn app:app -b..."	21 hours ago	Up 4 minutes	0.0.0.0:32781->80/tcp, :::32781->80/tcp	esiea-ressources-v
c0aa54119bff	esiea-ressources-result	"/usr/bin/tini -- no..."	21 hours ago	Up 4 minutes	0.0.0.0:32779->80/tcp, :::32779->80/tcp	esiea-ressources-r
d065dd03f5f8	postgres:9.4	"docker-entrypoint.s..."	21 hours ago	Up 4 minutes	5432/tcp	esiea-ressources-d
e1a86219706f	redis:alpine	"docker-entrypoint.s..."	21 hours ago	Up 4 minutes	6379/tcp	esiea-ressources-r

Figure 3 : Statu des conteneurs (services)

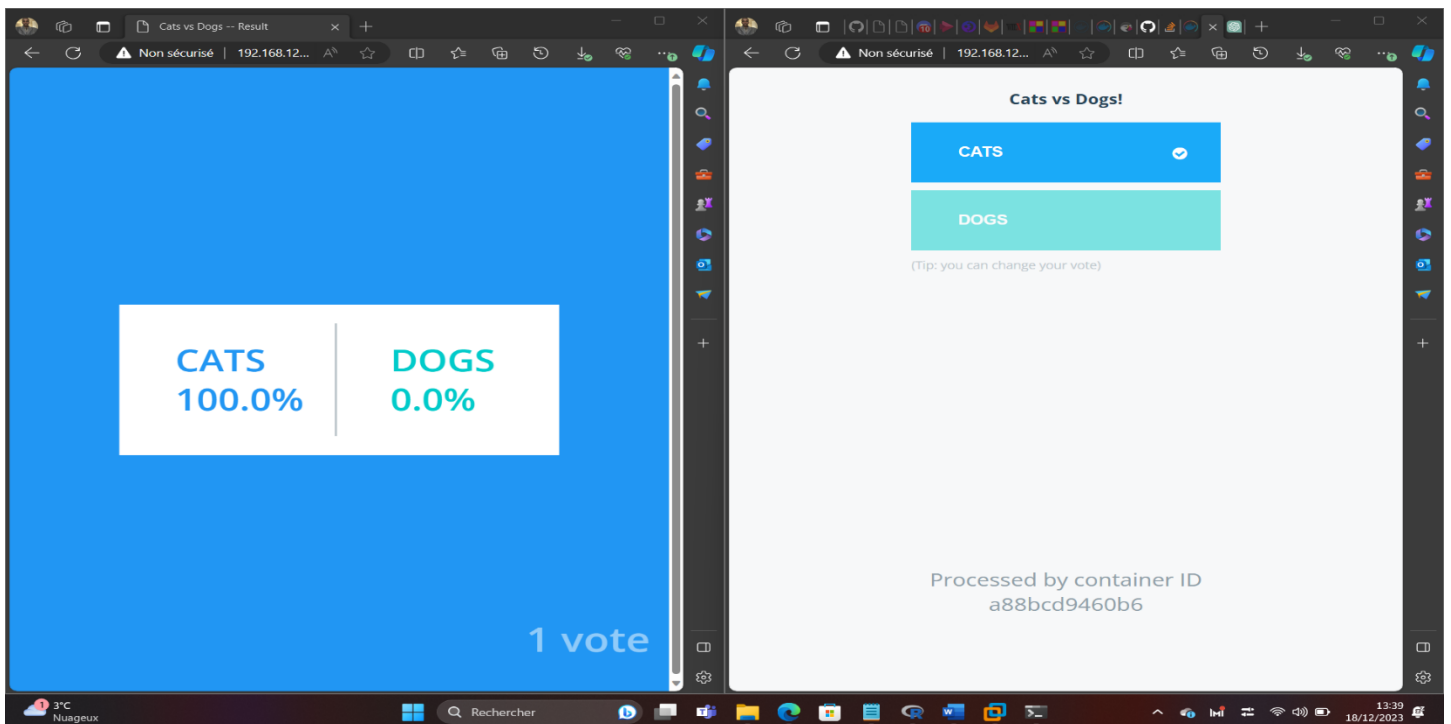


Figure 4: les deux écrans de l'application

6. Conclusion

Le projet réussit à mettre en place une infrastructure moderne et modulaire pour une plateforme de vote en temps réel. Bien que des défis aient été rencontrés, les réalisations clés démontrent la capacité du

projet à répondre à son objectif principal. Les perspectives d'amélioration soulignent des domaines spécifiques où des ajustements pourraient être apportés pour une amélioration continue.