

On commence la séance en mettant à jours notre docker-compose.yml On ajoute pour cela :

• Un indication de build :

build: ./back-mail

• Une dépendance :

-postares

On rebuild le tout : docker compose build | en se plaçant dans la racine commune au front et au back

Afin de résoudre le problème du front qui ne déamre pas, et sans log, nous apportons quelques modifications au dockerfile et au entrypoint

On a créé dans un second temps un autre fichier yaml.

Il est plus respectueux des regles de bienfaisance que l'initial et est donc voué à rester.

On a par exemple :

- Supprimer le nommage manuel des contenaire (allias)
- Eviter la repetition des variables d'environment (via un fichier .env)
- Ajouter une condition de redemarage du conteneur selon l'état de son service (healthcheck)

docker-compose -f docker-compose 2.yml down --volume: pour supprimer tous les conteneurs dudock-compose ainsi que le volume !

Maintenant que nos image sont bonne, nous allons les publier!

Pour cela, nous commençons par les renommer : docker tag 921c20b7806e public.ecr.aws/u1k0q0r7/back:juliette Puis on les push : docker push public.ecr.aws/u1k0q0r7/back:juliette

On peut ensuite récuperer l'image d'un de nos copains : docker pull public.ecr.aws/u1k0q0r7/back:axel

Pour l'utiliser, on appelle dans un docker-compose3.yml les images d'Axel avant de up ce yaml : docker-compose -f docker-compose3.yml up -d

backend: image: public.ecr.aws/u1k0q0r7/back:axel build: ./back

## Pour faire un bilan :

Docker	file	Backend	

FROM maven:3.8-openjdk-18 as builder

WORKDIR usr/src/app

COPY

RUN mvn clean package -DskipTests=true

FROM openjdk:18-jdk

**EXPOSE 8080** 

WORKDIR usr/src/app

COPY --from=builder usr/src/app/target/fs-back-\*.jar project.jar

CMD ["java", "-jar", "project.jar"]

## **Dockerfile Frontend:**

FROM node:16 AS builder

WORKDIR /usr/src/app

COPY

RUN npm install && npm run build

FROM nginx

**EXPOSE 80** 

ENV BACKEND\_URL\_ENV\_VAR=localhost

COPY nginx/ /etc/nginx/

COPY sed.sh /docker-entrypoint.d/sed.sh RUN chmod +x /docker-entrypoint.d/sed.sh

COPY --from=builder /usr/src/app/dist /usr/share/nginx/html

## Docker-compose Projet :

```
volumes:
 postgres:
networks:
backend:
services:
postgres:
  image: postgres:14
  volumes:
  - postgres:/var/lib/postgresql/data:rw
  networks:
   backend:
    aliases:
  postgres 
env_file:
   - .env
  healthcheck:
   test: ["CMD", "pg_isready", "-d", "$POSTGRES_DB", "-U", "$POSTGRES_USER"]
   interval: 5s
  timeout: 5s
retries: 2
 backend:
  image: public.ecr.aws/u1k0q0r7/back:axel build: ./back
  networks:
  backend:
    aliases:
     - backend
  env_file:
   - .env
  ports:
- 8080:8080
  depends_on:
   postgres:
    condition: service_healthy
  healthcheck:
test: ["CMD", "curl", "localhost:8080/restaurants"]
interval: 5s
   timeout: 5s
   retries: 2
 frontend:
  image: public.ecr.aws/u1k0q0r7/front:axel build: ./front
  environment:
   - BACKEND_URL_ENV_VAR=http://localhost:8080
  ports:
- 4200:80
  depends_on:
  backend:
    condition: service_healthy
  healthcheck:
   test: ["CMD", "curl", "localhost:80"]
   interval: 5s
   timeout: 5s
```

retries: 2