Mise en production

Déploiement sur Mogenius

FRITZ Florian, n° étudiant : 31810971

02/12/2022

Sommaire

1. INTRODUCTION

L'objectif de ce TP de Mise en Production de programme est de déployer des applications sur un serveur distant via l'outil web Mogenius. Il s'inscrit dans la suite du travail précédent dans lequel nous avions développé un build docker en local de notre application de back end (covid-api).

2. NGINX

Dans un premier temps, nous allons tester la plateforme en créant une application basique. Pour cela, nous utilisons les templates de Mogenius, qui sont enfaite des préconfigurations. Nous utiliserons une simple application NGINX qui affichera une page html et un simple hello_world.

Figure 1: Ajout d'un service Mogenius

Ce service est créé dans un environnement que nous avons appelé dev sur la plateforme et qui possède des ressources limitées au niveau du CPU et de la RAM notamment.

Mogenius demande alors un accès Github pour créer un repository contenant l'application et un dockerfile pour le build.

```
1  FROM nginxinc/nginx-unprivileged:stable-alpine
2
3  COPY html /usr/share/nginx/html
4
5  EXPOSE 8080
6
7  USER 101
8
9  CMD nginx -g 'daemon off;'
```

Figure 2: Dockerfile

Ce dockerfile crée une image Docker NGINX et place les fichiers du serveur web dans un dossier html. Nous pouvons alors utiliser Mogenius pour build et accéder en externe à l'application suivant le lien https://site-prod-dev-ago4g4.mo5.mogenius.io/

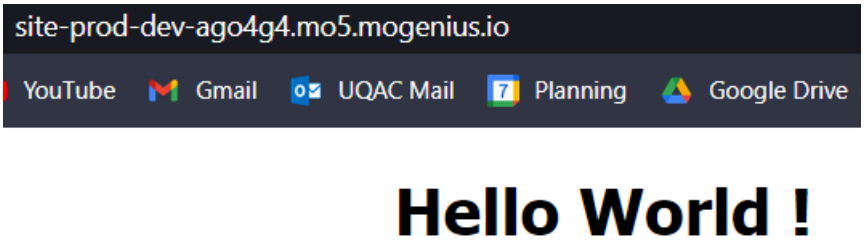


Figure 3: Hello World NGINX

Disponible sur le repository : https://github.com/Aeranduils/site

3. BACK END

Dans un second temps l'idée était de déployer un Docker de l'API backend sur Mogenius. Pour cela on build déjà un docker avec « docker build -t covid.api .». Celui-ci fonctionne en local.

```
Dockerfile > ...
1   FROM gradle:7.5.1-jdk17-alpine AS build
2   COPY --chown=gradle:gradle . /home/gradle/src
3   WORKDIR /home/gradle/src
4   RUN gradle build --no-daemon
5
6   FROM eclipse-temurin:17-jre
7   COPY build/libs/covid-api-0.0.1-SNAPSHOT.jar /app/app.jar
8   WORKDIR /app
9   EXPOSE 8080
10  CMD ["java", "-jar", "app.jar", "-XX:+UnlockExperimentalVMOptions","-Djava.security.egd=file:/dev/./urandom"]
11
```

Figure 4: Dockerfile

On peut aussi créer un docker-compose.

docker-compose.yml

```yaml
version: '3.8'
services:
  db:
    image: postgres:14.1-alpine
    restart: always
    environment:
      - POSTGRES_USER=${POSTGRES_USER}
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
    ports:
      - '5432:5432'
    volumes:
      - db-data:/var/lib/postgresql/data
  api:
    build: .
    ports:
      - "8000:8080"
    environment:
      DATABASE.HOST: dba
      DATABASE.PORT: 5432
      DATABASE.NAME: ${POSTGRES_DB}
      DATABASE.USER: ${POSTGRES_USER}
      DATABASE.PASSWORD: ${POSTGRES_PASSWORD}
    depends_on:
      - db
volumes:
  db-data:
```

Figure 5: docker-compose.yml

Puis on build avec docker compose up --build

Nous spécifions des variables d'environnement pour que le code fonctionne à la fois en local et sur le code déployé.

Par exemple, dans la configuration .yaml.

```yaml
spring:
  datasource:
    url: jdbc:postgresql://${ENDPOINT}:${PORT}/${POSTGRES_DB}
    username: ${POSTGRES_USER}
    password: ${POSTGRES_PASSWORD}
  jpa:
    database-platform: org.hibernate.dialect.PostgreSQLDialect
    open-in-view: false
    show-sql: true
    hibernate.ddl-auto: create
  liquibase:
    enabled: false
```

Figure 6: application.yaml

Cela fonctionne bien en local, malheureusement nous ne pouvons pas vérifier le déploiement puisque le docker build échoue.

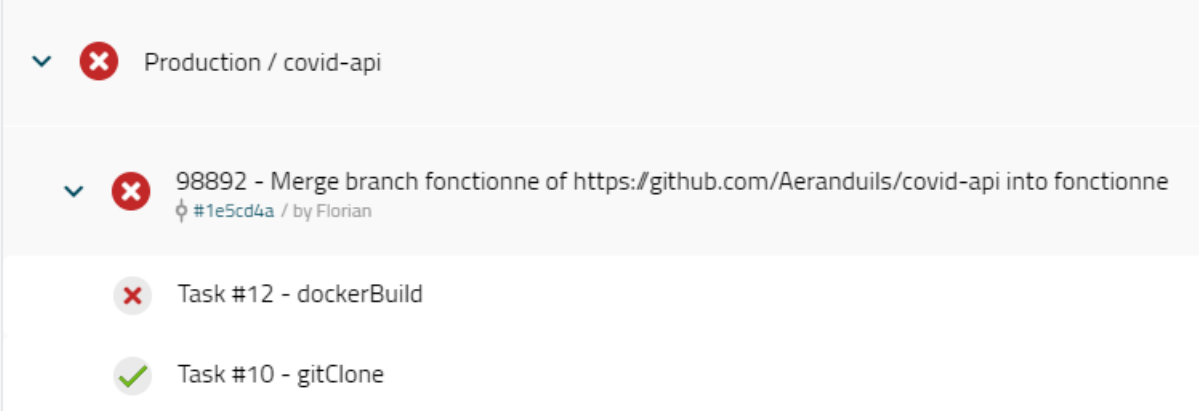
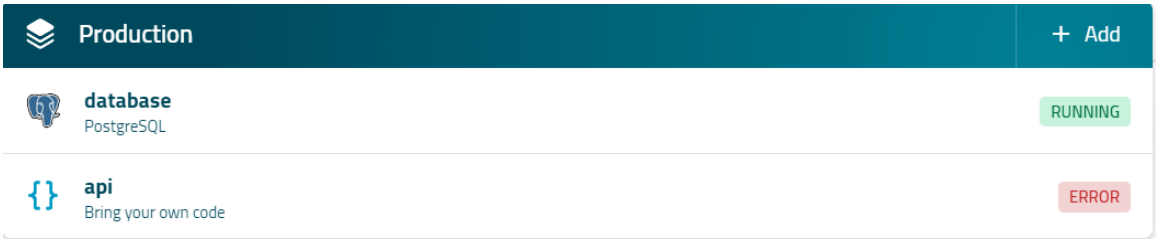


Figure 7: Code erreur Mogenius

Nous avons également créé un deuxième service, pour la base de données postgresSQL. Il nous faut pouvoir établir un lien entre ce service et le service de l'api.



Les variables d'environnements ont également été spécifié, mais le lien à établir n'a pas été trouvé.

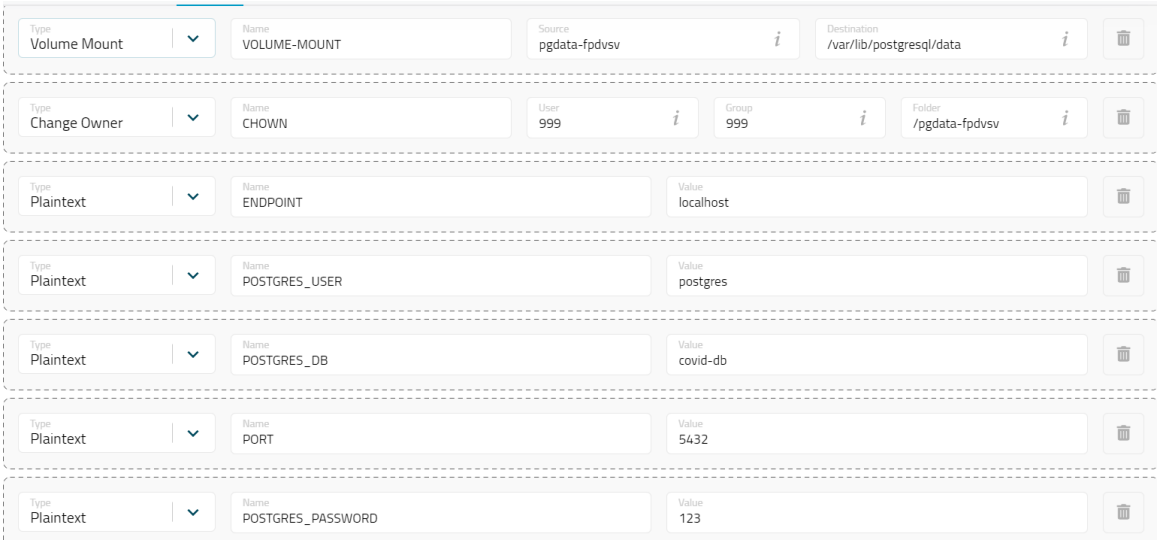


Figure 8: Variables d'environnement

Il aurait fallu spécifié le port 8080 comme le port à écouté.

Le dépôt de l'api utilisé est disponible à l'adresse : https://github.com/Aeranduils/covid-api.