



Mise en production

Déploiement sur Mogenius

FRITZ Florian, n° étudiant : 31810971





Sommaire

PARTIE A - Back End	3
PARTIE B - NGINX	5



1.BACK END

Dans un premier 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 .` ». Le soucis est qu'on ne peut pas accéder à la base de données PostgreSQL au sein du container.

```
Dockerfile > ...
1 FROM eclipse-temurin:17-jre
2 COPY build/libs/covid-api-0.0.1-SNAPSHOT.jar /app/app.jar
3 WORKDIR /app
4 EXPOSE 8080
5 CMD ["java", "-jar", "app.jar"]
6
```

Figure 1: Dockerfile

On obtient donc une erreur légitime au moment du build. Une première idée aurait pu être d'essayer de créer un environnement postgre et le définir dans le Dockerfile mais cela ne fonctionnait pas.

On peut aussi créer un docker-compose.

```
docker-compose.yml
1 version: '3.8'
2 services:
3   db:
4     image: postgres:14.1-alpine
5     restart: always
6     environment:
7       - POSTGRES_USER=${POSTGRES_USER}
8       - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
9     ports:
10      - '5432:5432'
11     volumes:
12      - db:/var/lib/postgresql/data
13      - ./db/init.sql:/docker-entrypoint-initdb.d/create_tables.sql
14   api:
15     container_name: covid-api
16     build:
17       context: ./
```

Figure 2: docker-compose.yml

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.



```
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 3: application.yaml

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

```
gitClone  dockerBuild

[2022-12-02 16:58:33] #3 DONE 0.7s
[2022-12-02 16:58:33]
[2022-12-02 16:58:33] #5 [internal] load build context
[2022-12-02 16:58:33] #5 sha256:f1cc65ee244e94313d8a832ec63b0304a75815293f13380d9e680e3dc4d3d330
[2022-12-02 16:58:33] #5 transferring context: 2B done
[2022-12-02 16:58:33] #5 DONE 0.0s
[2022-12-02 16:58:33]
[2022-12-02 16:58:33] #4 [1/3] FROM docker.io/library/eclipse-temurin:17-jre@sha256:d198151fe5ad4256da0346568fb7842e3b7bc289f5cb7475b43ca999f617999a
[2022-12-02 16:58:33] #4 sha256:bc7f62cac76c52b836d90b8a748778ed08f7225b58bc6c16c12df54bd4ba856c
[2022-12-02 16:58:33] #4 resolve docker.io/library/eclipse-temurin:17-jre@sha256:d198151fe5ad4256da0346568fb7842e3b7bc289f5cb7475b43ca999f617999a done
[2022-12-02 16:58:33] #4 CACHED
[2022-12-02 16:58:33]
[2022-12-02 16:58:33] #6 [2/3] COPY build/libs/covid-api-0.0.1-SNAPSHOT.jar /app/app.jar
[2022-12-02 16:58:33] #6 sha256:606f100c06c5be0046e3a842bf09d6f86c6c1eb8494705dc5ba082da5d4aa607
[2022-12-02 16:58:33] #6 ERROR: failed to walk /home/build/.local/share/docker/tmp/buildkit-mount040695677/build/libs: lstat /home/build/.local/share/docker/tmp/buildkit-mount040695677/build/libs: no such file or directory
[2022-12-02 16:58:33]
[2022-12-02 16:58:33] #4 [1/3] FROM docker.io/library/eclipse-temurin:17-jre@sha256:d198151fe5ad4256da0346568fb7842e3b7bc289f5cb7475b43ca999f617999a
[2022-12-02 16:58:33] #4 sha256:bc7f62cac76c52b836d90b8a748778ed08f7225b58bc6c16c12df54bd4ba856c
[2022-12-02 16:58:33] #4 sha256:d198151fe5ad4256da0346568fb7842e3b7bc289f5cb7475b43ca999f617999a 1.70kB / 1.70kB done
[2022-12-02 16:58:33] #4 sha256:30e722f4c370e126849a9422e4935899ad55ef0252112453e3b89d11be613bad 1.16kB / 1.16kB done
[2022-12-02 16:58:33] #4 sha256:daf2e28bdf6c189944c75a481ba30d800c61640ade858622769bebb80a9f0a1af 5.65kB / 5.65kB done
[2022-12-02 16:58:33] #4 CANCELED
[2022-12-02 16:58:33] -----
[2022-12-02 16:58:33] > [2/3] COPY build/libs/covid-api-0.0.1-SNAPSHOT.jar /app/app.jar:
[2022-12-02 16:58:33] -----
[2022-12-02 16:58:33] failed to compute cache key: failed to walk /home/build/.local/share/docker/tmp/buildkit-mount040695677/build/libs: lstat /home/build/.local/share/docker/tmp/buildkit-mount040695677/build/libs: no such file or directory
[2022-12-02 16:58:33] The push refers to repository [***api-w6fju8-dev-prod-ago4g4-r40wea]
[2022-12-02 16:58:33] An image does not exist locally with the tag: ***api-w6fju8-dev-prod-ago4g4-r40wea
[2022-12-02 16:58:33] #[error]Bash wrote one or more lines to the standard error stream.
[2022-12-02 16:58:33] #[error]An image does not exist locally with the tag: ***api-w6fju8-dev-prod-ago4g4-r40wea
[2022-12-02 16:58:33] ##[section]Finishing: dockerBuild
```

Figure 4: 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 spécifiées, mais le lien à établir n'a pas été trouvé.

Type Volume Mount	Name VOLUME-MOUNT	Source pgdata-fpdvsv	Destination /var/lib/postgresql/data	
Type Change Owner	Name CHOWN	User 999	Group 999	Folder /pgdata-fpdvsv
Type Plaintext	Name ENDPOINT	Value localhost		
Type Plaintext	Name POSTGRES_USER	Value postgres		
Type Plaintext	Name POSTGRES_DB	Value covid-db		
Type Plaintext	Name PORT	Value 5432		
Type Plaintext	Name POSTGRES_PASSWORD	Value 123		

Figure 5: Variables d'environnement

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

2. NGINX

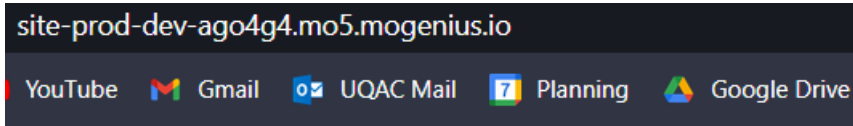
De manière plus simpliste, il est possible d'utiliser les template de mogenius accessible à la fois en interne et en externe.

Pour ajouter un service :





<https://site-prod-dev-ago4g4.mo5.mogenius.io/>



Hello World !

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

Avec un Dockerfile qui créer une image NGINX.

```
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 6: Dockerfile