Formation Docker : Manipulations pratiques avec Java

El Hadji Gaye

Auteur	El Hadji Gaye
Pour	Formations
Date	06/11/2024
Objet	Formation Docker : Manipulations pratiques avec Java.

I) Vocabulaire

Conteneurisation: En informatique, un conteneur est une structure de données, une classe, ou un type de données abstrait, dont les instances représentent des collections d'autres objets. Autrement dit, les conteneurs sont utilisés pour stocker des objets sous une forme organisée qui suit des règles d'accès spécifiques. On peut implémenter un conteneur de différentes façons, qui conduisent à des complexités en temps et en espace différentes. On choisira donc l'implémentation selon les besoins.

Un conteneur est une enveloppe virtuelle qui permet de distribuer une application avec tous les éléments dont elle a besoin pour fonctionner : fichiers source, environnement d'exécution, librairies, outils et fichiers. Ils sont assemblés en un ensemble cohérent et prêt à être déployé sur un serveur et son système d'exploitation (OS). Contrairement à la virtualisation de serveurs et à une machine virtuelle, le conteneur n’intègre pas de noyau, il s’appuie directement sur le noyau de l'ordinateur sur lequel il est déployé.

Virtualisation : La virtualisation consiste, en informatique, à exécuter sur une machine hôte, dans un environnement isolé, des systèmes d'exploitation — on parle alors de virtualisation système — ou des applications— on parle alors de virtualisation applicative. Ces ordinateurs virtuels sont appelés serveur privé virtuel (Virtual Private Server ou VPS) ou encore environnement virtuel (Virtual Environment ou VE).

II) Les commandes Docker à connaître

1. Commande docker ps

docker ps vous affiche toutes les instances de docker qui tournent actuellement sur votre environnement. Si vous ajoutez l'option -a, alors vous verrez mêmes les containers stoppés.

docker ps -a

2. *Commande docker images*

docker images est une commande qui vous montre les images que vous avez construites, et le -a vous montre les images intermédiaires.

```
docker images -a
```

3. *Commande docker network*

docker network ls est la commande docker qui liste les différents réseaux.

```
docker network ls
```

4. *Commande docker de runtime*

docker-compose up (-d) (--build)
docker-compose stop

La docker-compose est la plus simple car vous n'avez besoin que de 2 commandes : up et stop. stop est assez explicite et stop (mais ne supprime pas) vos conteneurs, mais up nécessite plus d'explications : cela va construire vos images si elles ne le sont pas déjà, et va démarrer vos dockers.

docker build (-t NAME) PATH /URL

Si vous voulez re-build vos images, utilisez l'option --build (vous pouvez aussi utiliser la commande docker-compose build pour uniquement construire des images). L'option -d, qui signifie "detach" fait tourner les conteneurs en tâche de fond.

Avec Docker, vous avez besoin d'une commande séparée pour construire votre image, où vous pouvez spécifier le nom de votre image et vous devez spécifier le PATH ou URL selon votre contexte (cela peut être un repo git).

docker run (-d) (-p hostPort :containerPort) (--name NAME)

run crée le conteneur en utilisant l'image que vous indiquez. Vous pouvez spécifier de nombreux paramètres. Nous vous recommandons d'ajouter un nom à votre conteneur et vous pourriez avoir besoin de spécifier quelques ports à exposer. Comme pour docker-compose, le -d lance le conteneur en tâche de fond.

docker start ID /NAME

docker stop ID /NAME

Le start and stop ne devraient pas être trop compliqués à comprendre, mais il faut noter que vous pouvez "start" uniquement des conteneurs qui sont déjà arrêtés, donc déjà build avec la commande run.

docker exec -it NAME /ID "sh" /"/bin/bash"

Cette commande vous permet de lancer un shell sur votre container. Je préfère utiliser "/bin/bash" mais votre conteneur peut ne pas avoir bash d'installé, et seulement "sh" qui est plus courant (surtout sur les alpines). Si vous avez des configurations spéciales dans votre conteneur, vous aurez peut-être besoin d'utiliser des arguments supplémentaires pour vous y connecter.

5. *Commandes docker de suppression*

Ces commandes permettent de supprimer vos conteneurs et vos images. Vous en aurez probablement besoin pour libérer de l'espace disque.

docker rm ID /NAME

docker-compose rm

Le docker rm supprime seulement un conteneur alors que docker-compose rm supprime tous les conteneurs démarrés avec une commande docker-compose.

docker rmi ID /NAME

Docker rmi supprime l'image que vous passez en paramètre et récursivement toutes les images intermédiaires utilisées pour la construire.

6. *Commandes docker de logs*

Les commandes suivantes sont utiles quand vous devez débugger certains de vos conteneurs (ou, plus souvent, l'application que vous déployez à l'intérieur).

docker logs ID /NAME (-f --tail NBLINE)

Cette commande affiche les logs du container passé en paramètre. Si vous utilisez l'option -f --tail NBLINE vous pouvez suivre en live le flux de vos logs (NBLINE est le nombre de lignes que vous souhaitez afficher). Gardez à l'esprit de choisir un nombre de lignes que vous serez capable de gérer, pour ne pas être dépassé par vos logs.

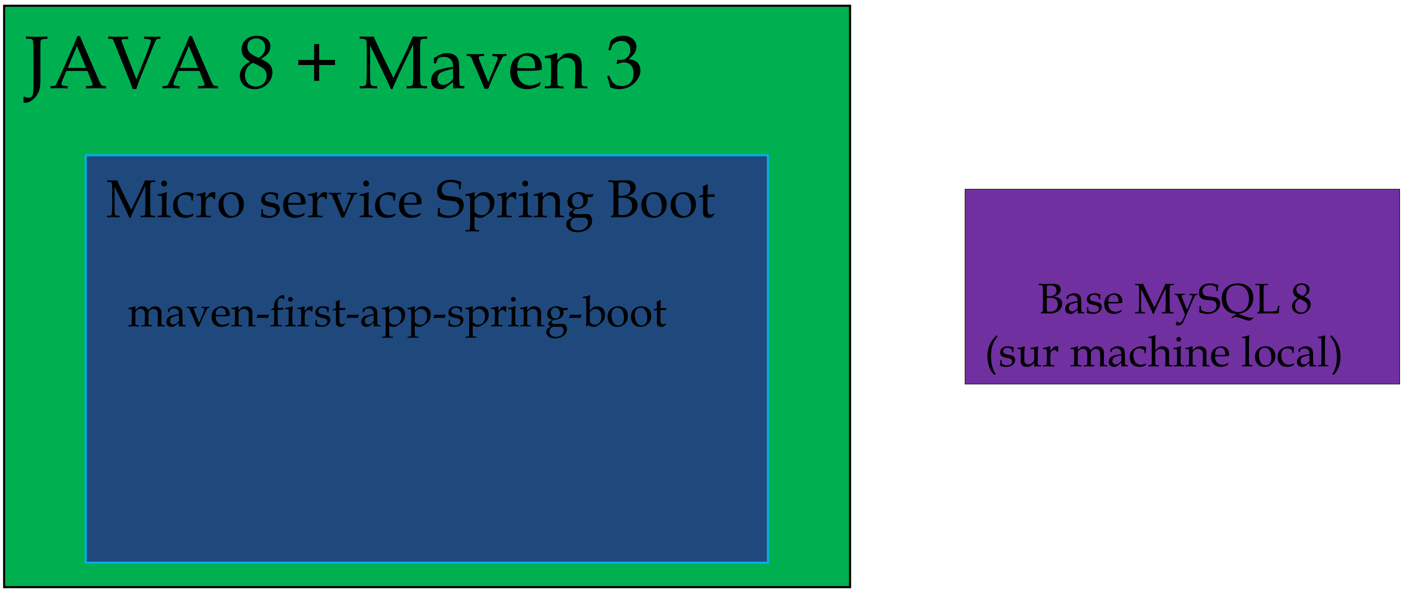
docker-compose logs (ID /NAME)

L'option (ID /NAME) avec docker-compose logs vous permet de voir les logs d'un conteneur uniquement, au lieu de voir tous les logs. L'astuce ici est que si vous n'utilisez pas l'option -d quand vous utilisez docker run ou docker-compose up vous verrez vos logs directement (mais vous aurez besoin d'arrêter le conteneur pour quitter la vue). Cela peut toujours être utile pour débugger des applications au démarrage.

III) Déployer un Micro Service Spring Boot avec Docker

1. *Version 1 : Micro Service Spring Boot sans Docker*

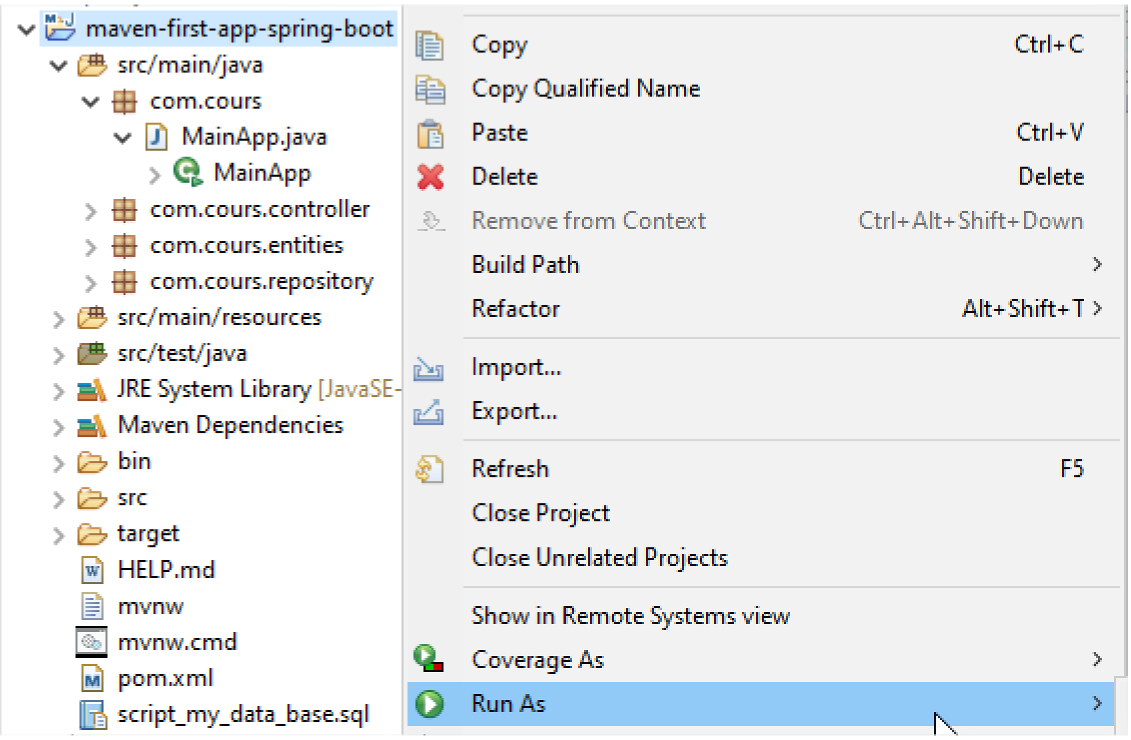
Recupérer le projet **maven-first-app-spring-boot**. Ce projet était développé sous Java 8 avec une base de donnée MySQL 8 qui était installé sur votre poste en local, l'architecture de l'application sera :

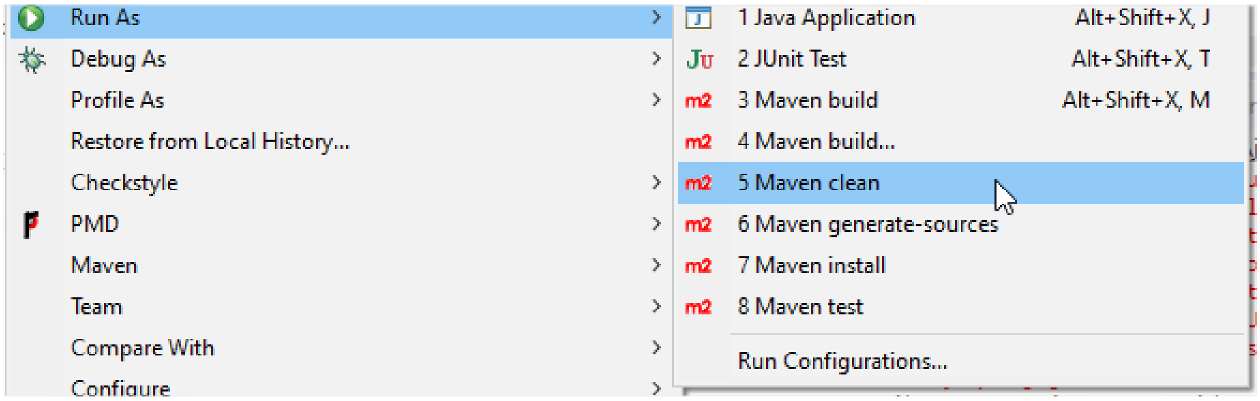

JAVA 8 + Maven 3

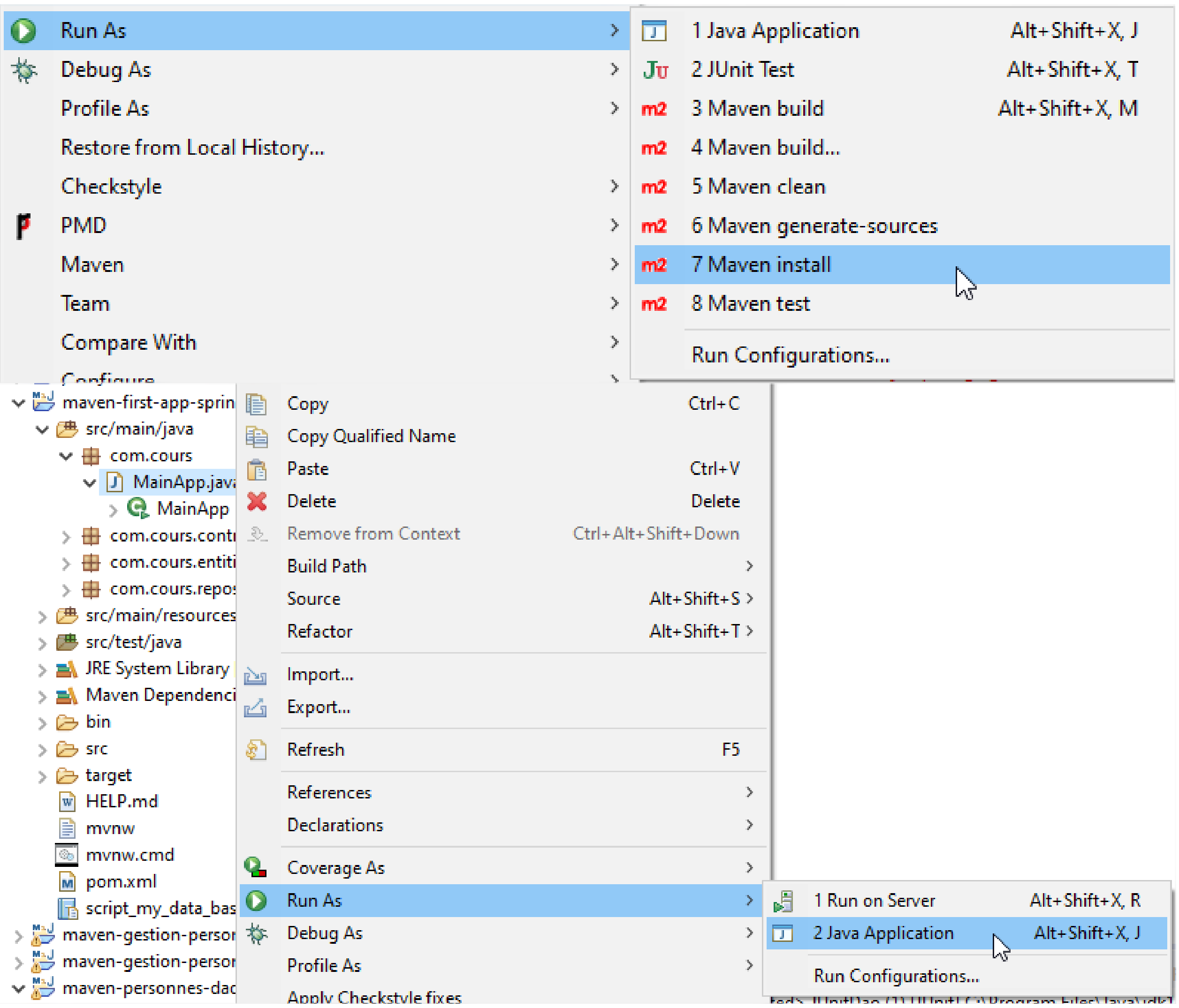
Micro service Spring Boot

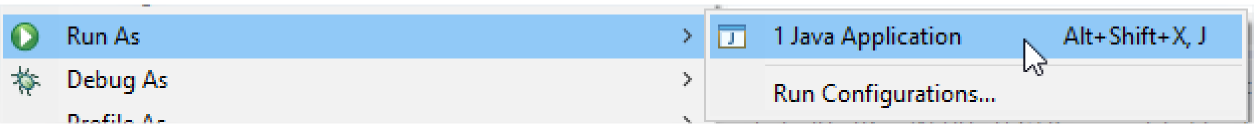
maven-first-app-spring-boot

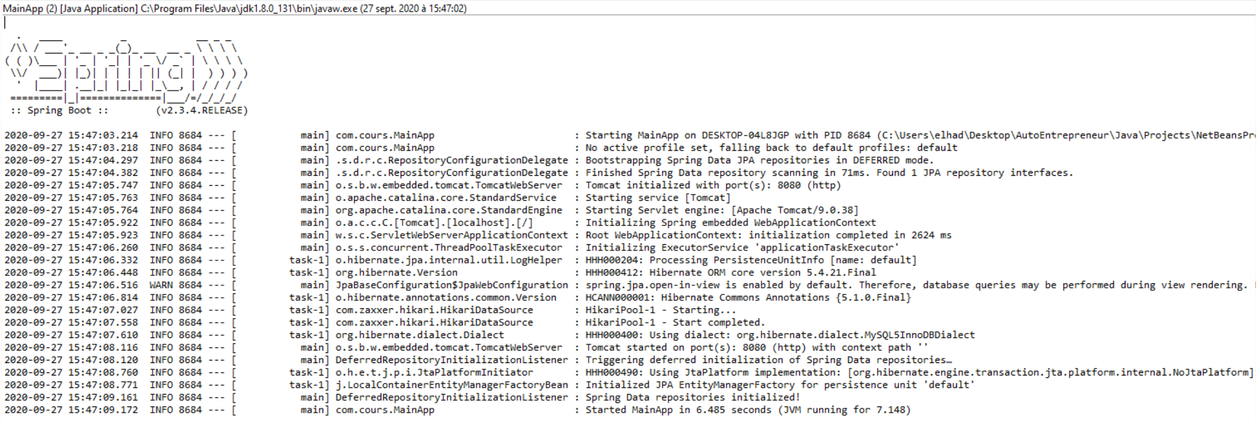
Base MySQL 8 (sur machine local)




Run As
Debug As
Profile As
Restore from Local History...
Checkstyle
PMD
Maven
Team
Compare With
Configure
1 Java Application Alt+Shift+X, J
2 JUnit Test Alt+Shift+X, T
3 Maven build Alt+Shift+X, M
4 Maven build...
5 Maven clean
6 Maven generate-sources
7 Maven install
8 Maven test
Run Configurations...


Run As
Debug As
Profile As
Restore from Local History...
Checkstyle
PMD
Maven
Team
Compare With
Configure
1 Java Application Alt+Shift+X, J
2 JUnit Test Alt+Shift+X, T
3 Maven build Alt+Shift+X, M
4 Maven build...
5 Maven clean
6 Maven generate-sources
7 Maven install
8 Maven test
Run Configurations...
maven-first-app-sprin
src/main/java
com.cours
MainApp.java
MainApp
com.cours.contr
com.cours.entiti
com.cours.repos
src/main/resources
src/test/java
JRE System Library
Maven Dependenci
bin
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
script_my_data_bas
maven-gestion-persor
maven-gestion-persor
maven-personnes-dac
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
References
Declarations
Coverage As
Run As
Debug As
Profile As
Apply Checkstyle fixes
1 Run on Server Alt+Shift+X, R
2 Java Application Alt+Shift+X, J
Run Configurations...

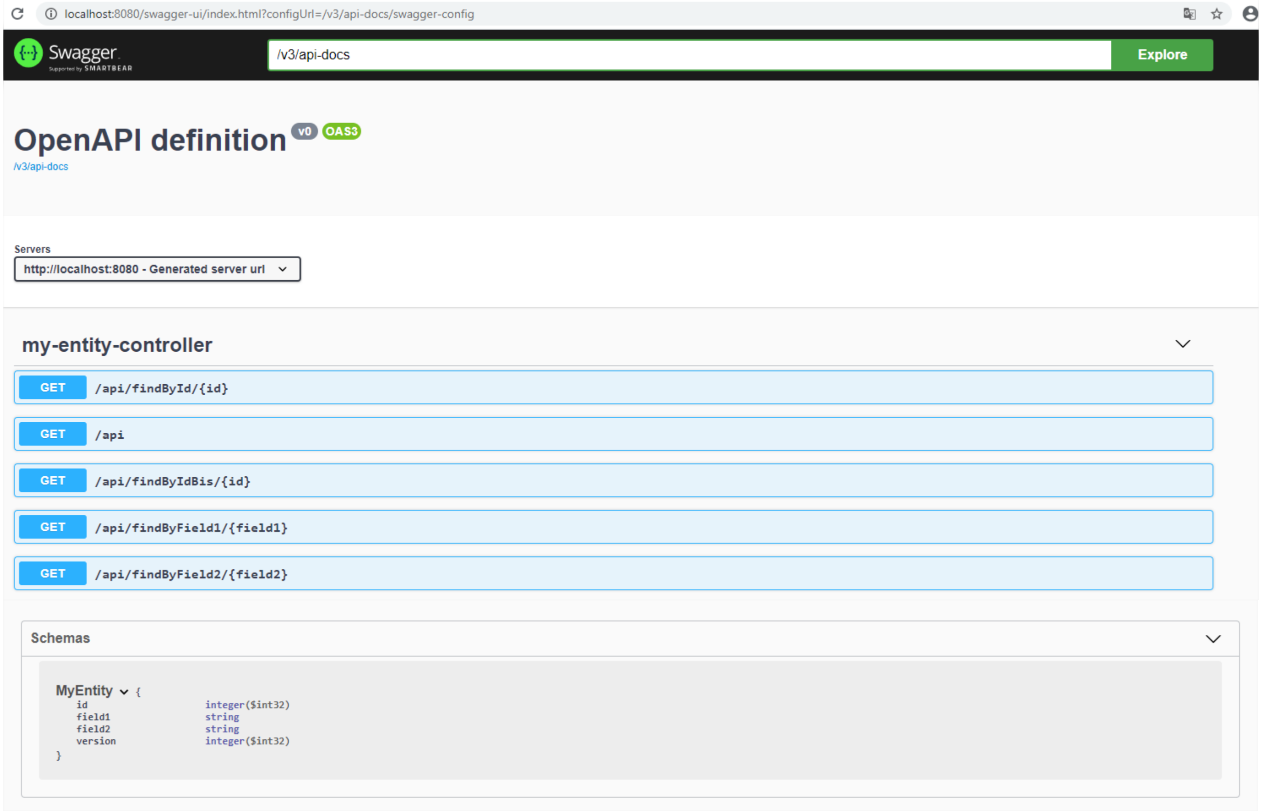

Run As
Debug As
Profile As
1 Java Application Alt+Shift+X, J
Run Configurations...



MainApp (2) [Java Application] C:\Program Files\Java\jdk1.8.0_131\bin\javaw.exe (27 sept. 2020 à 15:47:02)

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.3.4.RELEASE)

2020-09-27 15:47:03.214  INFO 8684 --- [           main] com.cours.MainApp                        : Starting MainApp on DESKTOP-04L8JGP with PID 8684 (C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansPr
2020-09-27 15:47:03.218  INFO 8684 --- [           main] com.cours.MainApp                        : No active profile set, falling back to default profiles: default
2020-09-27 15:47:04.297  INFO 8684 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFERRED mode.
2020-09-27 15:47:04.382  INFO 8684 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 71ms. Found 1 JPA repository interfaces.
2020-09-27 15:47:05.747  INFO 8684 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2020-09-27 15:47:05.763  INFO 8684 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2020-09-27 15:47:05.764  INFO 8684 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-09-27 15:47:05.922  INFO 8684 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2020-09-27 15:47:05.923  INFO 8684 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2624 ms
2020-09-27 15:47:06.260  INFO 8684 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2020-09-27 15:47:06.332  INFO 8684 --- [         task-1] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2020-09-27 15:47:06.448  INFO 8684 --- [         task-1] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.4.21.Final
2020-09-27 15:47:06.516  WARN 8684 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering.
2020-09-27 15:47:06.814  INFO 8684 --- [         task-1] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
2020-09-27 15:47:07.027  INFO 8684 --- [         task-1] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2020-09-27 15:47:07.558  INFO 8684 --- [         task-1] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2020-09-27 15:47:07.610  INFO 8684 --- [         task-1] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.MySQL5InnoDBDialect
2020-09-27 15:47:08.116  INFO 8684 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2020-09-27 15:47:08.120  INFO 8684 --- [           main] DeferredRepositoryInitializationListener : Triggering deferred initialization of Spring Data repositories…
2020-09-27 15:47:08.760  INFO 8684 --- [         task-1] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2020-09-27 15:47:08.771  INFO 8684 --- [         task-1] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2020-09-27 15:47:09.161  INFO 8684 --- [           main] DeferredRepositoryInitializationListener : Spring Data repositories initialized!
2020-09-27 15:47:09.172  INFO 8684 --- [           main] com.cours.MainApp                        : Started MainApp in 6.485 seconds (JVM running for 7.148)
```

Lancer l'URL http://localhost:8080/swagger-ui.html



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config

OpenAPI definition v0 OAS3

/v3/api-docs

Servers

http://localhost:8080 - Generated server url

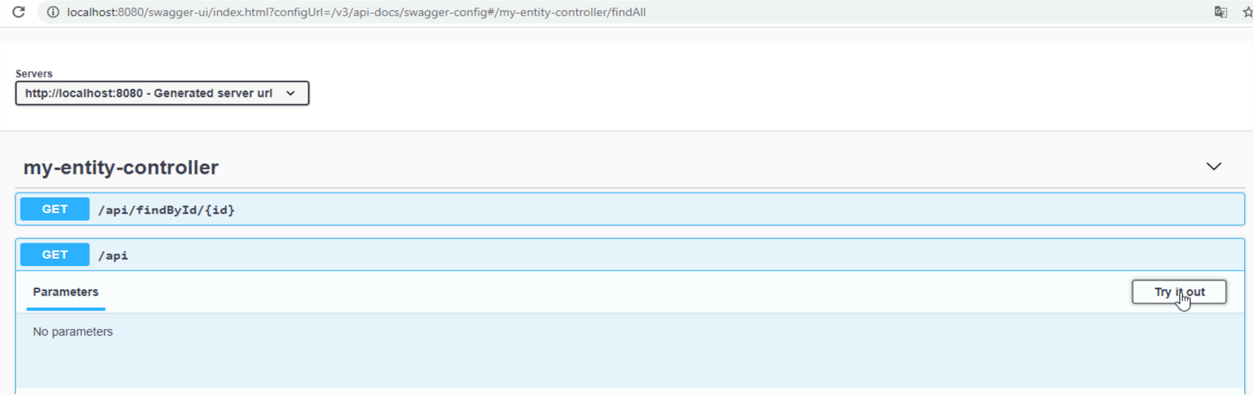
my-entity-controller

- GET /api/findById/{id}
- GET /api
- GET /api/findByIdBis/{id}
- GET /api/findByField1/{field1}
- GET /api/findByField2/{field2}

Schemas

```
MyEntity ⌄ {
    id          integer($int32)
    field1      string
    field2      string
    version     integer($int32)
}
```

En cliquant sur **GET/api** on obtient :



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/my-entity-controller/findAll

Servers

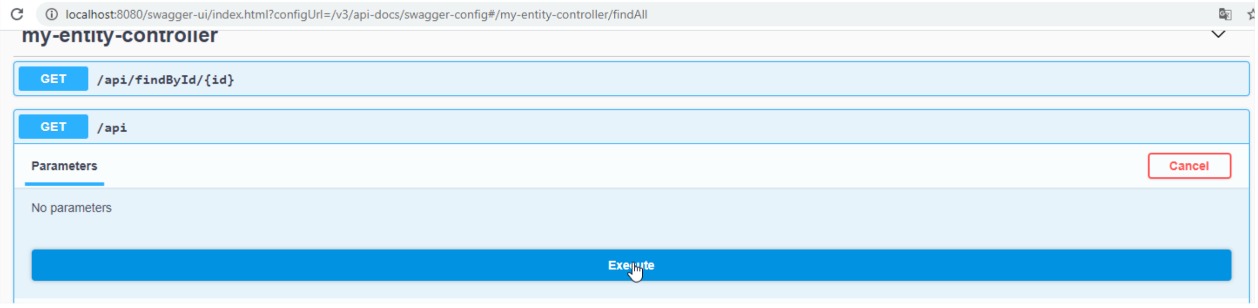
http://localhost:8080 - Generated server url

my-entity-controller

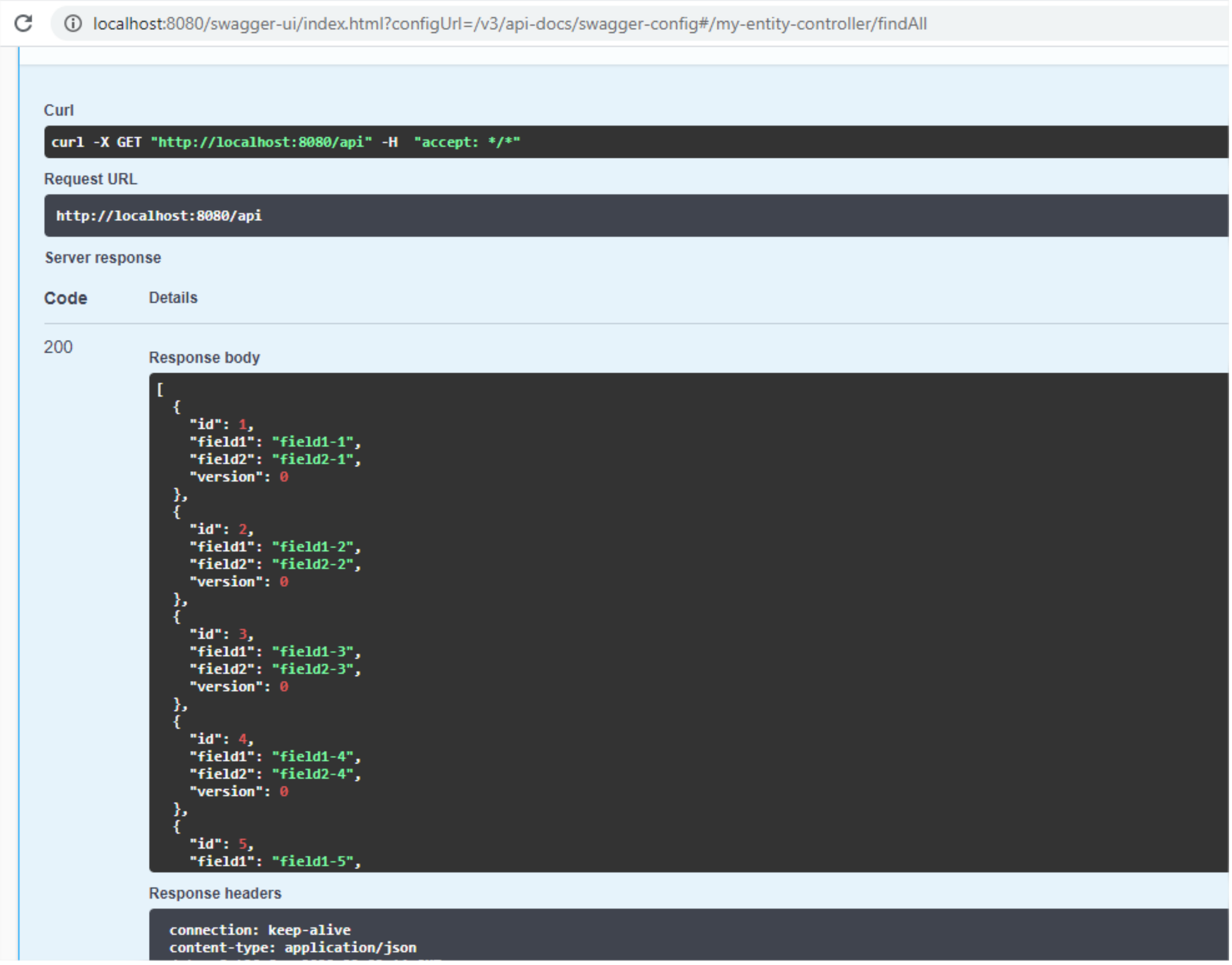
- GET /api/findById/{id}
- GET /api

Parameters

No parameters



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/my-entity-controller/findAll
my-entity-controller
GET
/api/findById/{id}
GET
/api
Parameters
Cancel
No parameters
Execute



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/my-entity-controller/findAll

Curl

```
curl -X GET "http://localhost:8080/api" -H  "accept: */*"
```

Request URL

```
http://localhost:8080/api
```

Server response

Code Details

200

Response body

```
[
  {
    "id": 1,
    "field1": "field1-1",
    "field2": "field2-1",
    "version": 0
  },
  {
    "id": 2,
    "field1": "field1-2",
    "field2": "field2-2",
    "version": 0
  },
  {
    "id": 3,
    "field1": "field1-3",
    "field2": "field2-3",
    "version": 0
  },
  {
    "id": 4,
    "field1": "field1-4",
    "field2": "field2-4",
    "version": 0
  },
  {
    "id": 5,
    "field1": "field1-5",
```

Response headers

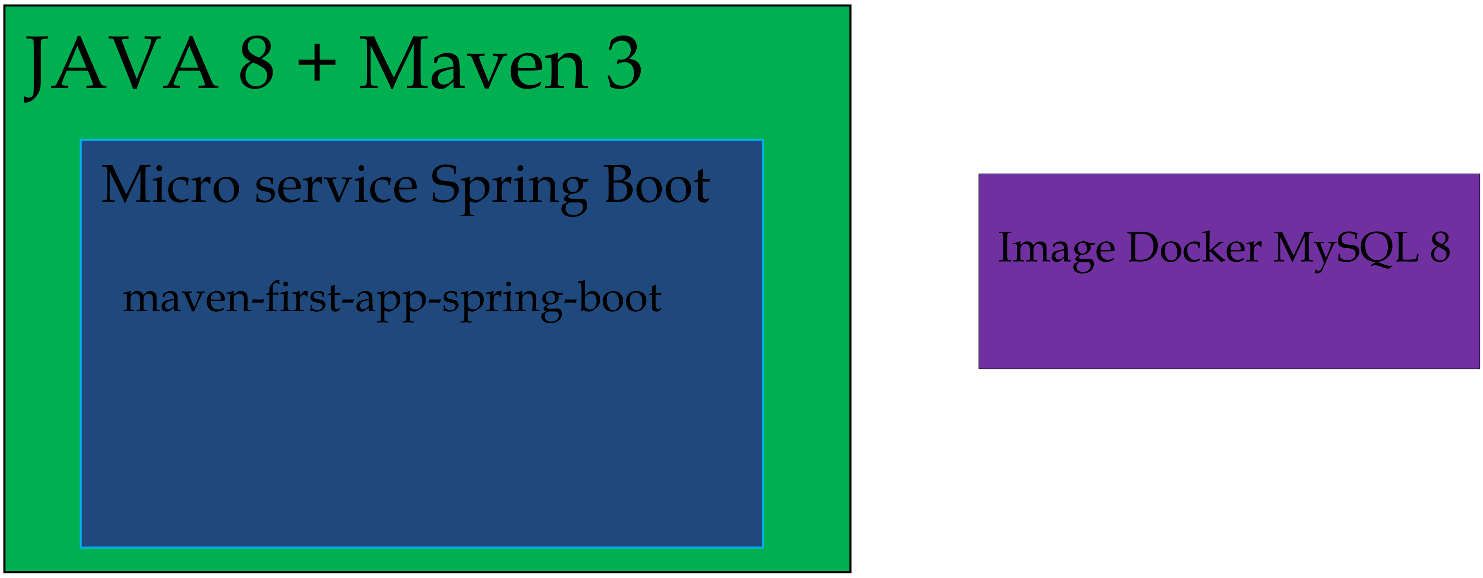
```
connection: keep-alive
content-type: application/json
```

Nous allons faire une première amélioration dans cette application en utilisant une image docker de MySQL 8.

2. *Version 2 : Micro Service Spring Boot avec une image docker MySql 8*

Nous allons améliorer notre architecture micro service en utilisant une image Docker MySQL 8 à la place d'une base de données installé phisiquement dans la machine local.

L'achitecture de l'application resemblera à :



a. Le fichier init_my_data_base.sql

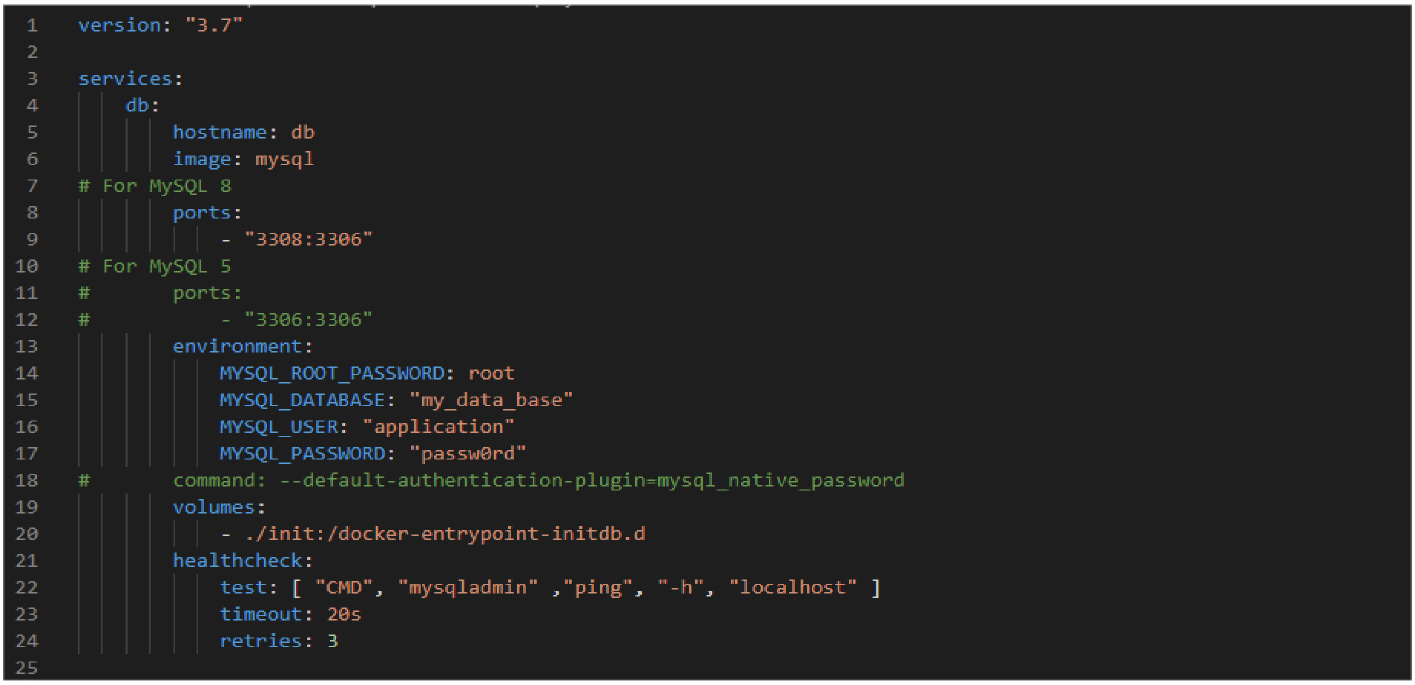
Créer le fichier **maven-first-app-spring-boot/init/init_my_data_base.sql** dont le contenu sera :

```sql
/* Base de données: my_data_base */
DROP DATABASE IF EXISTS my_data_base;
/*DROP USER IF EXISTS 'application'@'localhost';*/
CREATE DATABASE my_data_base DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER IF NOT EXISTS'application'@'localhost' IDENTIFIED BY 'passw0rd';
/* For MYSQL 8 */
GRANT ALL PRIVILEGES ON my_data_base.* TO'application'@'localhost';
/* For MYSQL 5 */
/*GRANT ALL ON my_data_base.* TO 'application'@'localhost' IDENTIFIED BY 'passw0rd';*/
USE my_data_base;

SET FOREIGN_KEY_CHECKS = 0;
DROP TABLE IF EXISTS MyEntity;

CREATE TABLE MyEntity (
   id INTEGER PRIMARY KEY AUTO_INCREMENT,
   field1 VARCHAR(100),
   field2 VARCHAR(100),
   version int(15)
 )ENGINE=InnoDB DEFAULT CHARSET=utf8;


INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-1', 'field2-1',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-2', 'field2-2',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-3', 'field2-3',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-4', 'field2-4',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-5', 'field2-5',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-6', 'field2-6',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-7', 'field2-7',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-8', 'field2-8',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-9', 'field2-9',0);
INSERT INTO MyEntity(field1,field2,version) VALUES ('field1-10', 'field2-10',0);
```

b. Le fichier docker-compose.yaml pour une image MYSQL 8

Nous allons créer le fichier **maven-gestion-personnes-spring-boot/docker-compose.yaml** qui nous permettra d'initialiser une base de donnée MySQL 8.

Ce fichier aura pour contenu :



```yaml
version: "3.7"

services:
    db:
        hostname: db
        image: mysql
# For MySQL 8
        ports:
            - "3308:3306"
# For MySQL 5
#       ports:
#           - "3306:3306"
        environment:
            MYSQL_ROOT_PASSWORD: root
            MYSQL_DATABASE: "my_data_base"
            MYSQL_USER: "application"
            MYSQL_PASSWORD: "passw0rd"
#       command: --default-authentication-plugin=mysql_native_password
        volumes:
            - ./init:/docker-entrypoint-initdb.d
        healthcheck:
            test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
            timeout: 20s
            retries: 3

```

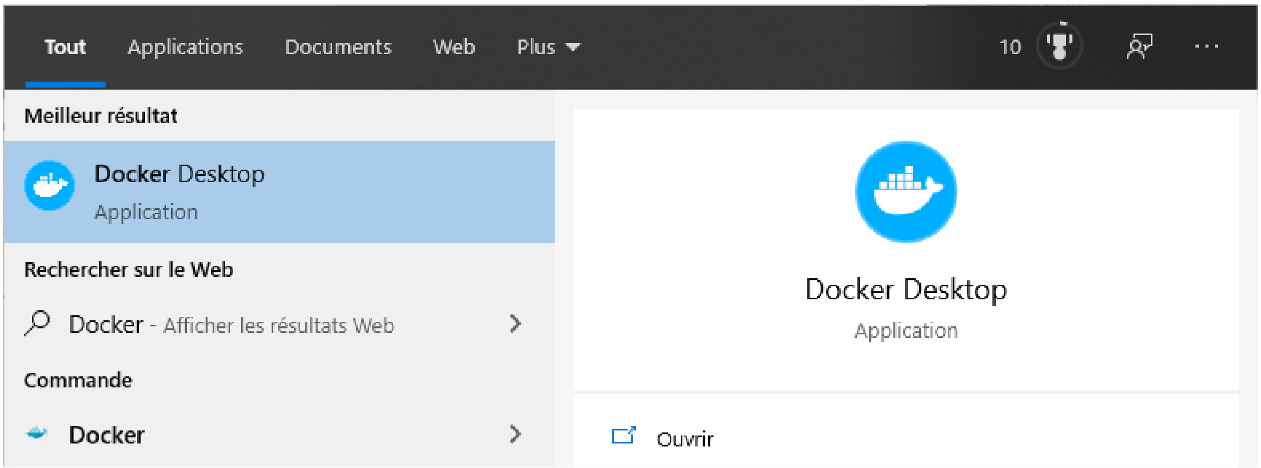
En version copiable :

```
version: "3.7"

services:
  db:
    hostname: db
    image: mysql
# For MySQL 8
    ports:
      - "3308:3306"
# For MySQL 5
#      ports:
#        - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: "my_data_base"
      MYSQL_USER: "application"
      MYSQL_PASSWORD: "passw0rd"
#     command: --default-authentication-plugin=mysql_native_password
    volumes:
      - ./init:/docker-entrypoint-initdb.d
    healthcheck:
      test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
      timeout: 20s
      retries: 3
```

Ligne 9 : on redirige les données de l'image Docker MySQL 8 du port **3306** de Docker vers le port **3308** de notre machine local.

Attention : ceci est un fichier yaml donc il faut respecter les tabulations.

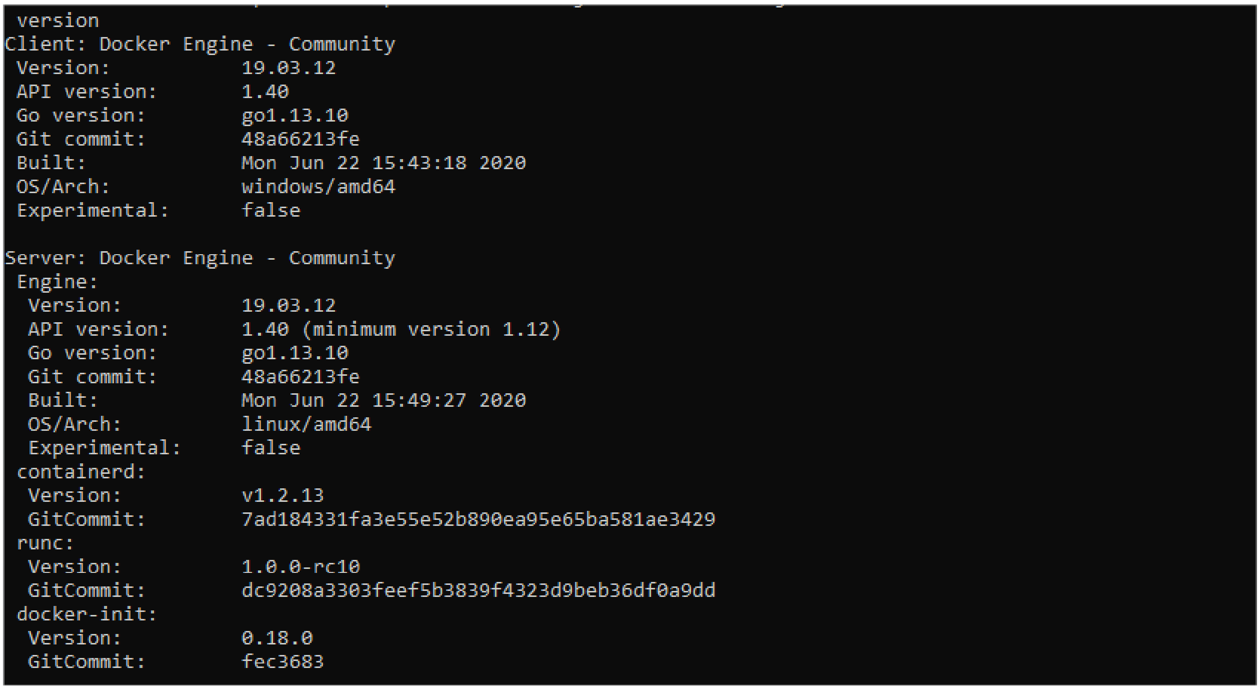
Lancer votre **Docker Desktop** :



Après avoir lancer Docker laisser lui 30 secondes pour qu'il puisse terminer de démarrer.

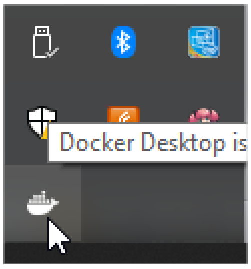
Lancer la commande : **docker version**

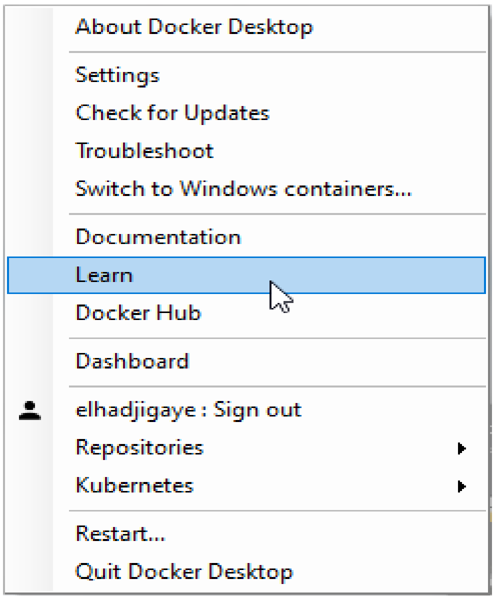
Verifier avec votre invité de commande que votre client et votre serveur Docker sont bien démarés :

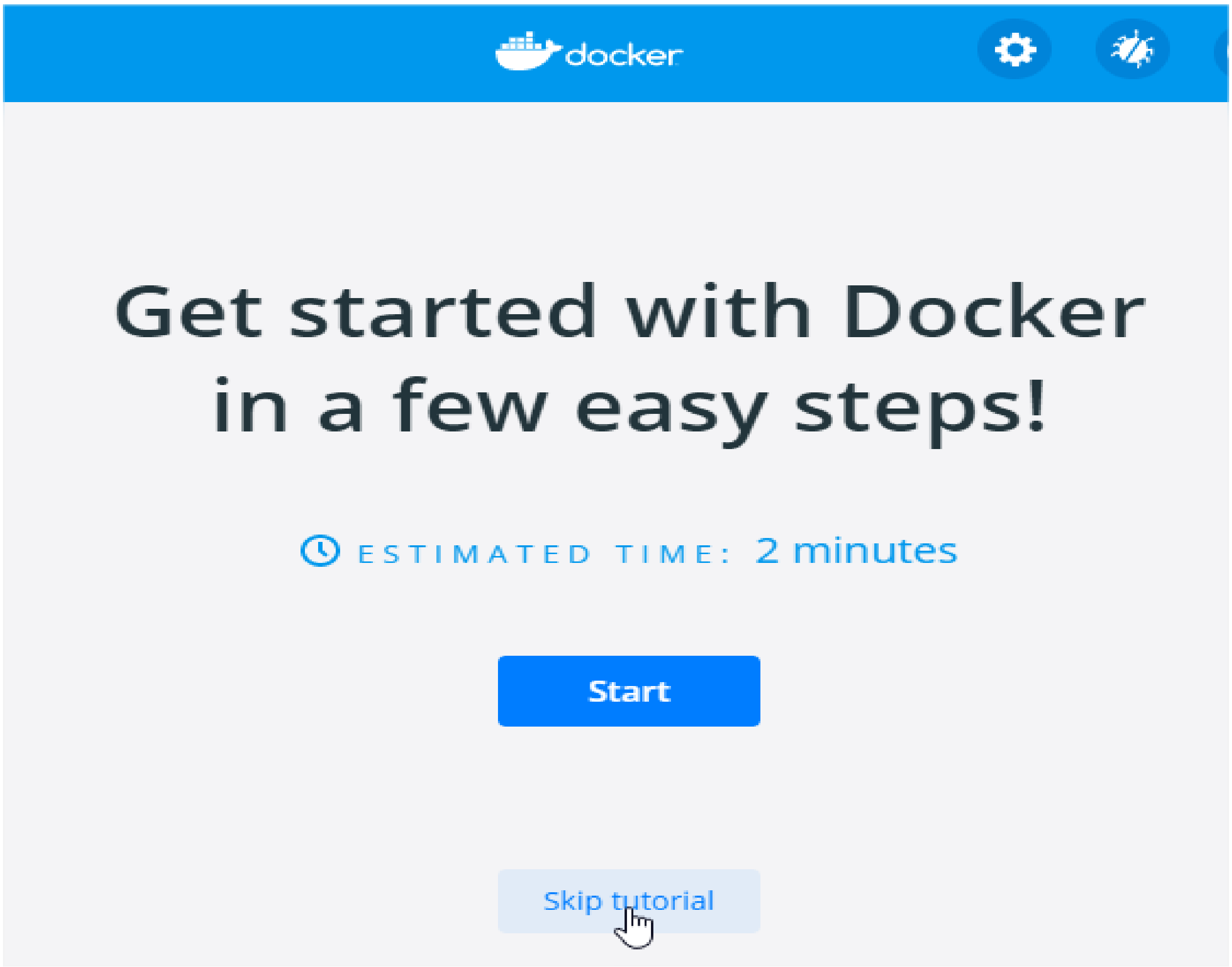


```
 version
Client: Docker Engine - Community
 Version:           19.03.12
 API version:       1.40
 Go version:        go1.13.10
 Git commit:        48a66213fe
 Built:             Mon Jun 22 15:43:18 2020
 OS/Arch:           windows/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.12
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.10
  Git commit:       48a66213fe
  Built:            Mon Jun 22 15:49:27 2020
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          v1.2.13
  GitCommit:        7ad184331fa3e55e52b890ea95e65ba581ae3429
 runc:
  Version:          1.0.0-rc10
  GitCommit:        dc9208a3303feef5b3839f4323d9beb36df0a9dd
 docker-init:
  Version:          0.18.0
  GitCommit:        fec3683
```

Lancer l'interface Docker qui permet de voir la liste des images Docker.

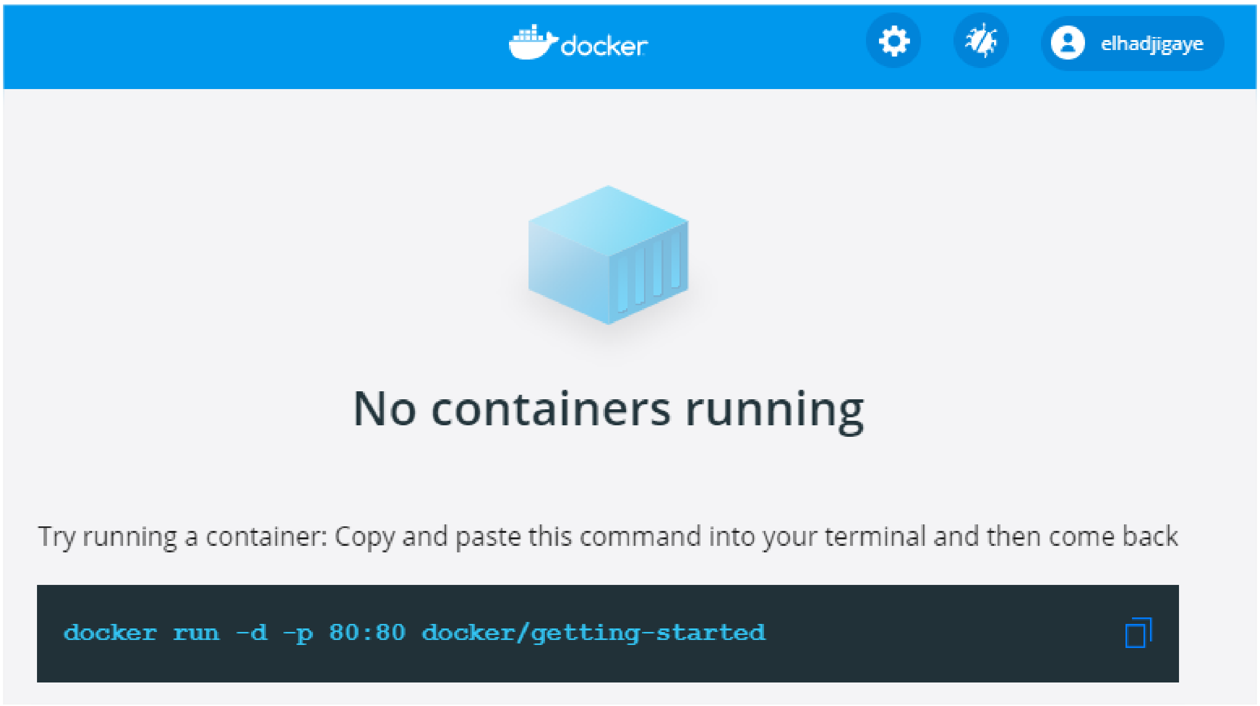






Get started with Docker in a few easy steps!

ESTIMATED TIME: 2 minutes



No containers running

Try running a container: Copy and paste this command into your terminal and then come back

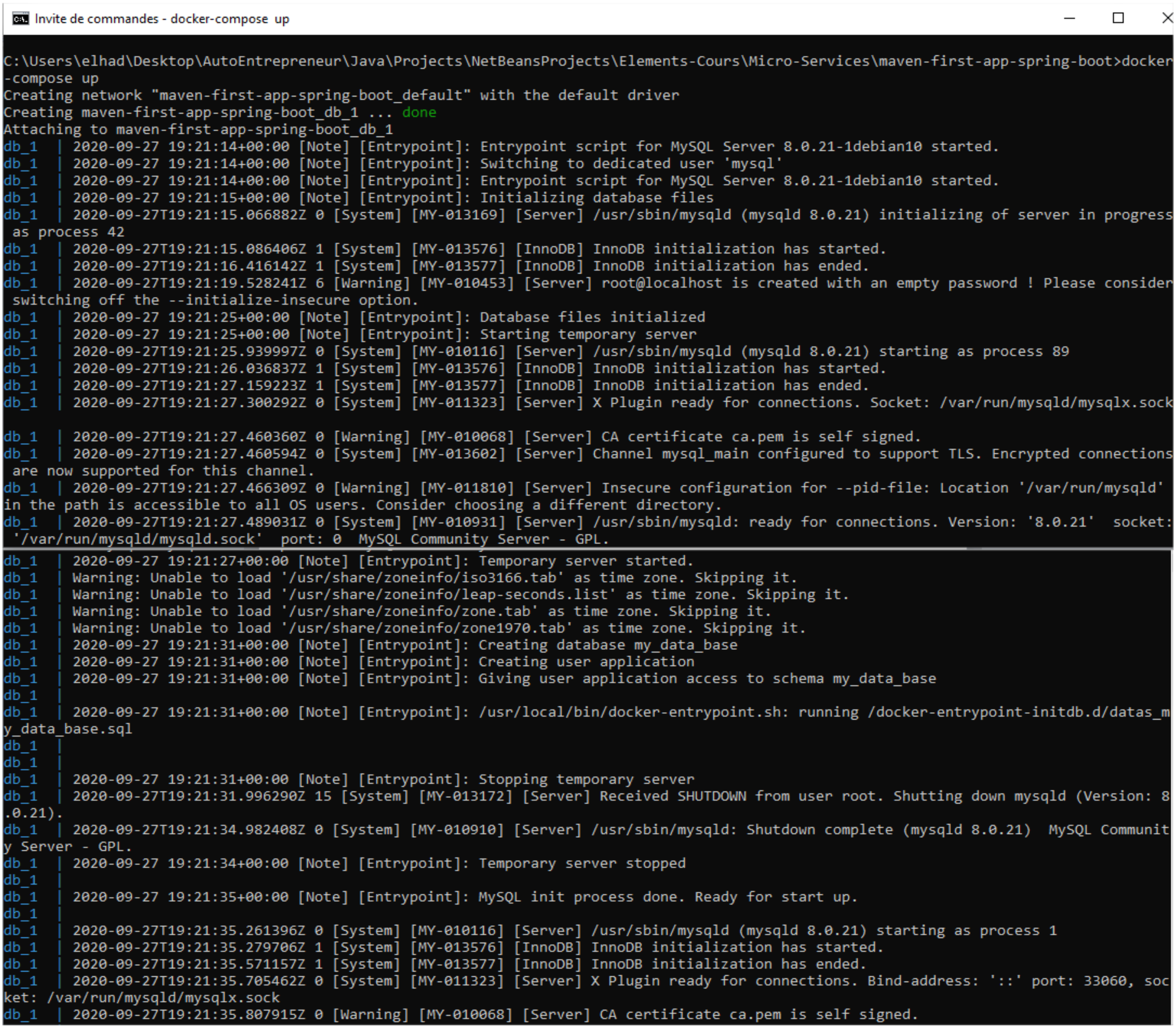
```
docker run -d -p 80:80 docker/getting-started
```

Lancer ensuite les commandes suivantes :

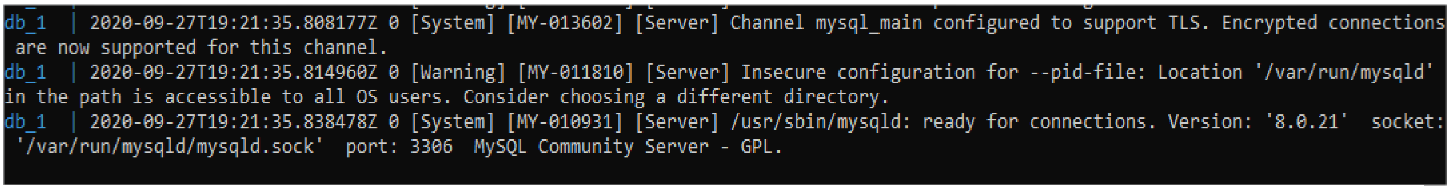
```
cd maven-first-app-spring-boot
docker-compose build
docker-compose up
```

En spécifiant le fichier docker compose à utiliser, on obtient :

```
docker-compose –f docker-compose-mysql-8.yaml build
docker-compose –f docker-compose-mysql-8.yaml up
```

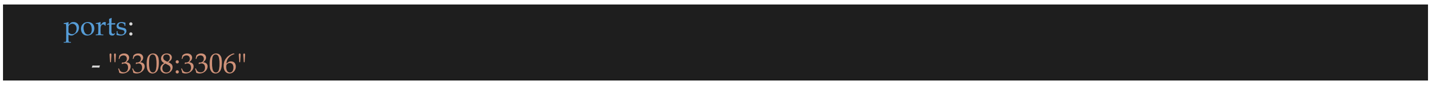


```
C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansProjects\Elements-Cours\Micro-Services\maven-first-app-spring-boot>docker
-compose up
Creating network "maven-first-app-spring-boot_default" with the default driver
Creating maven-first-app-spring-boot_db_1 ... done
Attaching to maven-first-app-spring-boot_db_1
db_1  | 2020-09-27 19:21:14+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1  | 2020-09-27 19:21:14+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
db_1  | 2020-09-27 19:21:14+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1  | 2020-09-27 19:21:15+00:00 [Note] [Entrypoint]: Initializing database files
db_1  | 2020-09-27T19:21:15.066882Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.21) initializing of server in progress
 as process 42
db_1  | 2020-09-27T19:21:15.086406Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
db_1  | 2020-09-27T19:21:16.416142Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
db_1  | 2020-09-27T19:21:19.528241Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty password ! Please consider
 switching off the --initialize-insecure option.
db_1  | 2020-09-27 19:21:25+00:00 [Note] [Entrypoint]: Database files initialized
db_1  | 2020-09-27 19:21:25+00:00 [Note] [Entrypoint]: Starting temporary server
db_1  | 2020-09-27T19:21:25.939997Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.21) starting as process 89
db_1  | 2020-09-27T19:21:26.036837Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
db_1  | 2020-09-27T19:21:27.159223Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
db_1  | 2020-09-27T19:21:27.300292Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Socket: /var/run/mysqld/mysqlx.sock

db_1  | 2020-09-27T19:21:27.460360Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
db_1  | 2020-09-27T19:21:27.460594Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections
 are now supported for this channel.
db_1  | 2020-09-27T19:21:27.466309Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file: Location '/var/run/mysqld'
in the path is accessible to all OS users. Consider choosing a different directory.
db_1  | 2020-09-27T19:21:27.489031Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '8.0.21'  socket:
 '/var/run/mysqld/mysqld.sock'  port: 0  MySQL Community Server - GPL.
db_1  | 2020-09-27 19:21:27+00:00 [Note] [Entrypoint]: Temporary server started.
db_1  | Warning: Unable to load '/usr/share/zoneinfo/iso3166.tab' as time zone. Skipping it.
db_1  | Warning: Unable to load '/usr/share/zoneinfo/leap-seconds.list' as time zone. Skipping it.
db_1  | Warning: Unable to load '/usr/share/zoneinfo/zone.tab' as time zone. Skipping it.
db_1  | Warning: Unable to load '/usr/share/zoneinfo/zone1970.tab' as time zone. Skipping it.
db_1  | 2020-09-27 19:21:31+00:00 [Note] [Entrypoint]: Creating database my_data_base
db_1  | 2020-09-27 19:21:31+00:00 [Note] [Entrypoint]: Creating user application
db_1  | 2020-09-27 19:21:31+00:00 [Note] [Entrypoint]: Giving user application access to schema my_data_base
db_1  |
db_1  | 2020-09-27 19:21:31+00:00 [Note] [Entrypoint]: /usr/local/bin/docker-entrypoint.sh: running /docker-entrypoint-initdb.d/datas_m
y_data_base.sql
db_1  |
db_1  |
db_1  | 2020-09-27 19:21:31+00:00 [Note] [Entrypoint]: Stopping temporary server
db_1  | 2020-09-27T19:21:31.996290Z 15 [System] [MY-013172] [Server] Received SHUTDOWN from user root. Shutting down mysqld (Version: 8
.0.21).
db_1  | 2020-09-27T19:21:34.982408Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: Shutdown complete (mysqld 8.0.21)  MySQL Communit
y Server - GPL.
db_1  | 2020-09-27 19:21:34+00:00 [Note] [Entrypoint]: Temporary server stopped
db_1  |
db_1  | 2020-09-27 19:21:35+00:00 [Note] [Entrypoint]: MySQL init process done. Ready for start up.
db_1  |
db_1  | 2020-09-27T19:21:35.261396Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.21) starting as process 1
db_1  | 2020-09-27T19:21:35.279706Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
db_1  | 2020-09-27T19:21:35.571157Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
db_1  | 2020-09-27T19:21:35.705462Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, soc
ket: /var/run/mysqld/mysqlx.sock
db_1  | 2020-09-27T19:21:35.807915Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
```



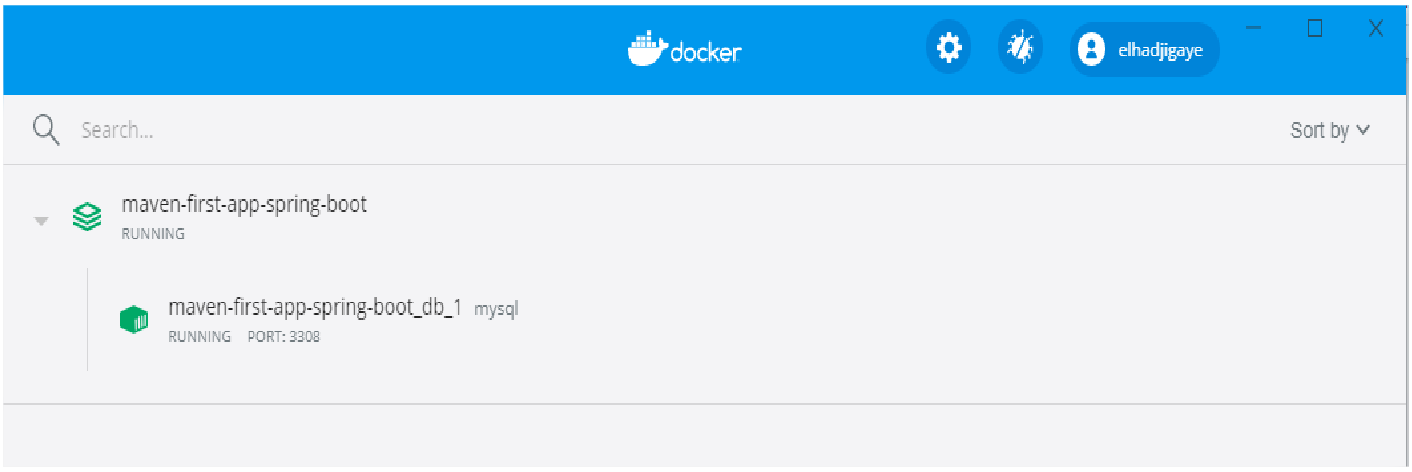
```
db_1  | 2020-09-27T19:21:35.808177Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections
 are now supported for this channel.
db_1  | 2020-09-27T19:21:35.814960Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file: Location '/var/run/mysqld'
in the path is accessible to all OS users. Consider choosing a different directory.
db_1  | 2020-09-27T19:21:35.838478Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '8.0.21'  socket:
 '/var/run/mysqld/mysqld.sock'  port: 3306  MySQL Community Server - GPL.
```

Nous voyons d’après les logs que l’image Docker de MySQL a été démaré sur le port 3306. Dans l’application nous écoutons sur le port 3308 c’est pour cela que nous avons dans notre fichier de configuration Docker :



```
    ports:
      - "3308:3306"
```

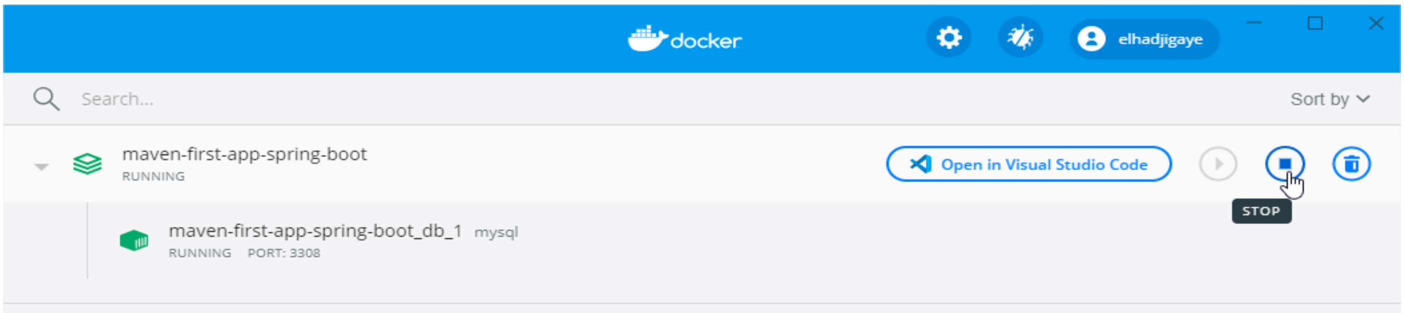
En allant voir sur le Docker Desktop on voit l’image qu’on vient de créer.



maven-first-app-spring-boot
RUNNING

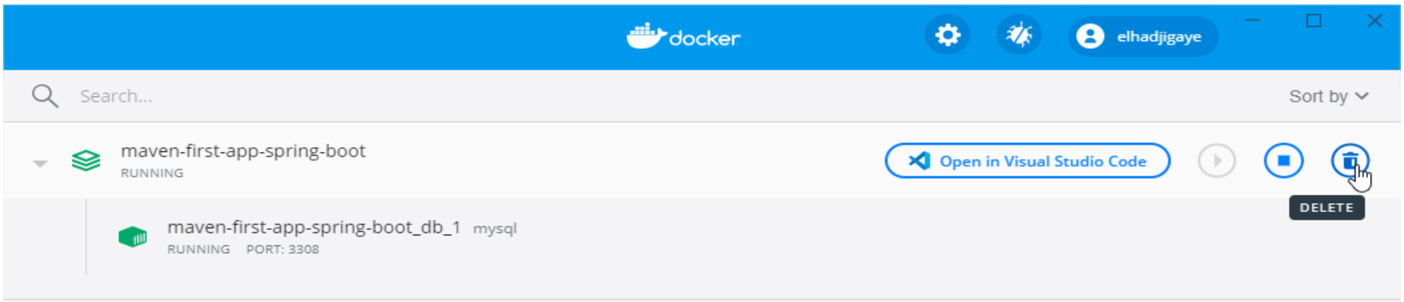
maven-first-app-spring-boot_db_1 mysql
RUNNING PORT: 3308

Il est possible d’arreter le conteneur ou même de le supprimer.



maven-first-app-spring-boot
RUNNING

maven-first-app-spring-boot_db_1 mysql
RUNNING PORT: 3308

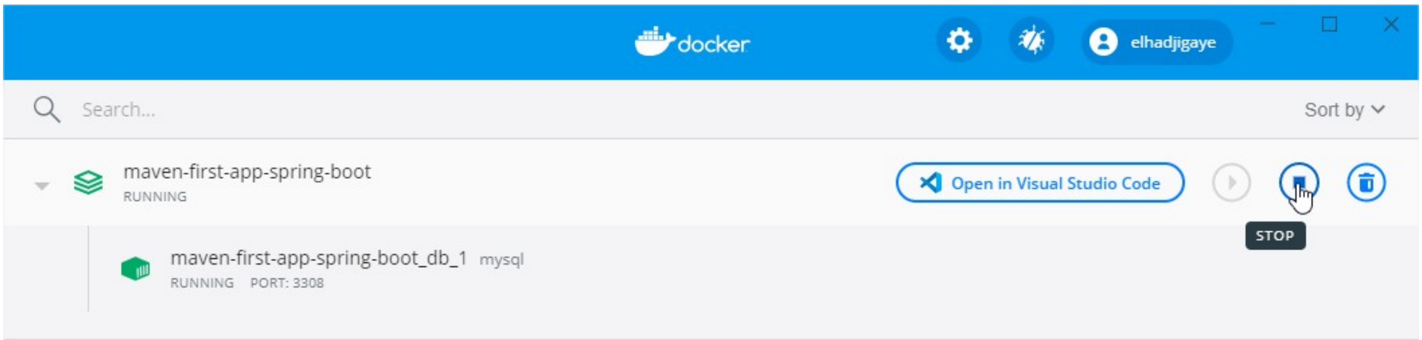


maven-first-app-spring-boot
RUNNING

maven-first-app-spring-boot_db_1 mysql
RUNNING PORT: 3308

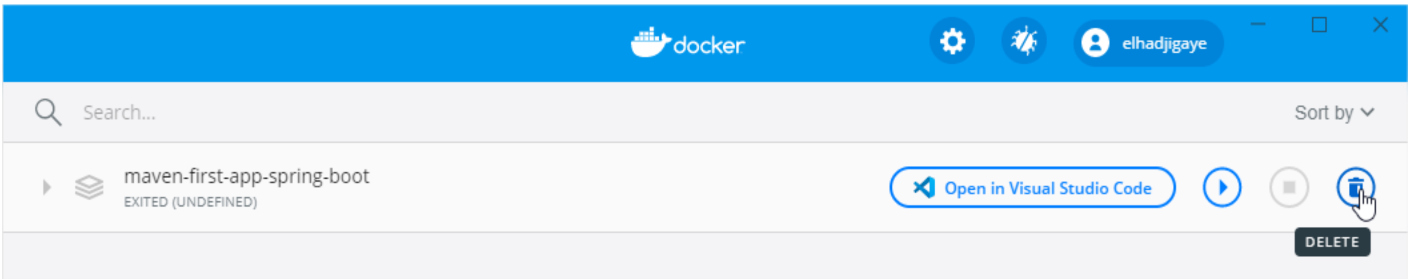
c. Le fichier docker-compose.yaml pour une image MYSQL XXX

Arreter et supprimer le précédent conteneur :

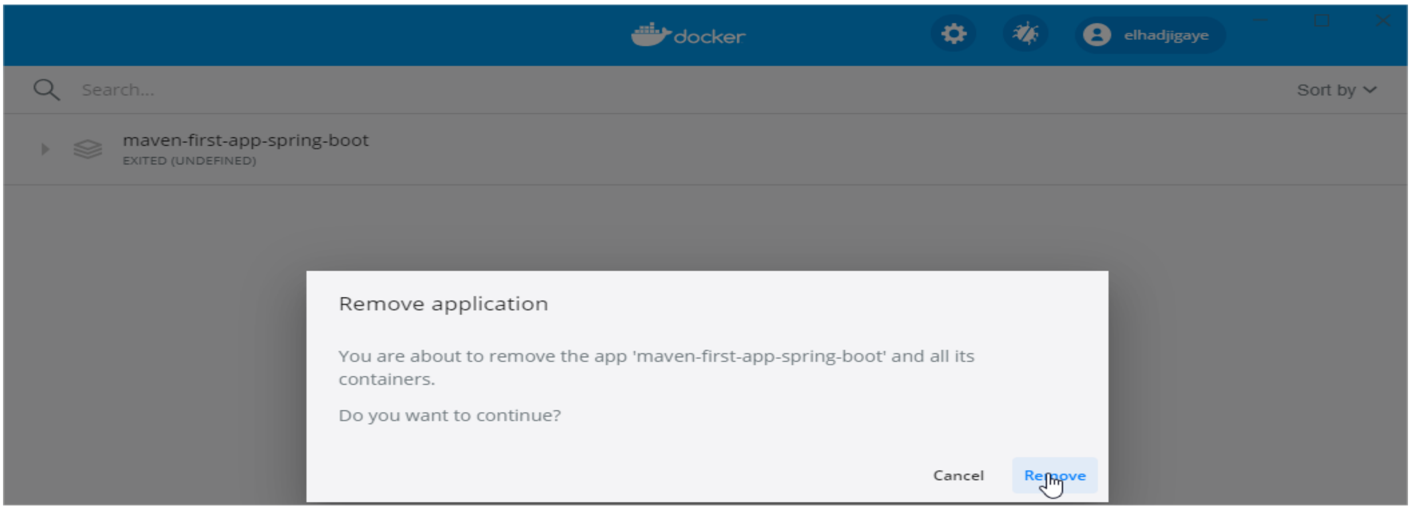


maven-first-app-spring-boot
RUNNING

maven-first-app-spring-boot_db_1 mysql
RUNNING PORT: 3308



maven-first-app-spring-boot
EXITED (UNDEFINED)



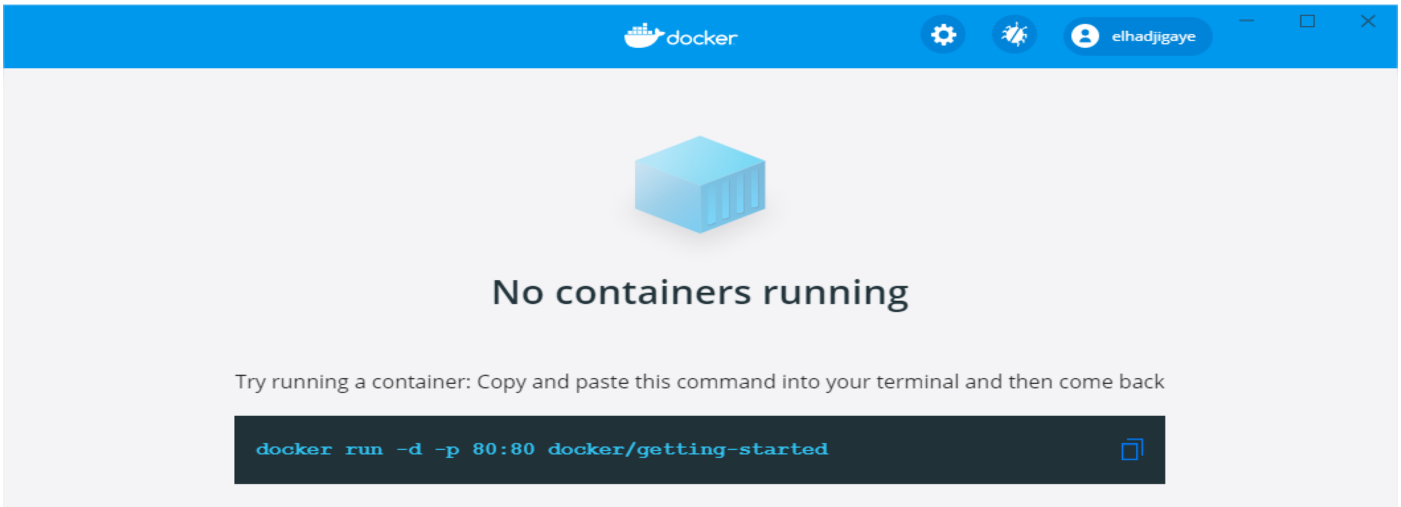
maven-first-app-spring-boot
EXITED (UNDEFINED)

Remove application

You are about to remove the app 'maven-first-app-spring-boot' and all its containers.

Do you want to continue?

Cancel Remove



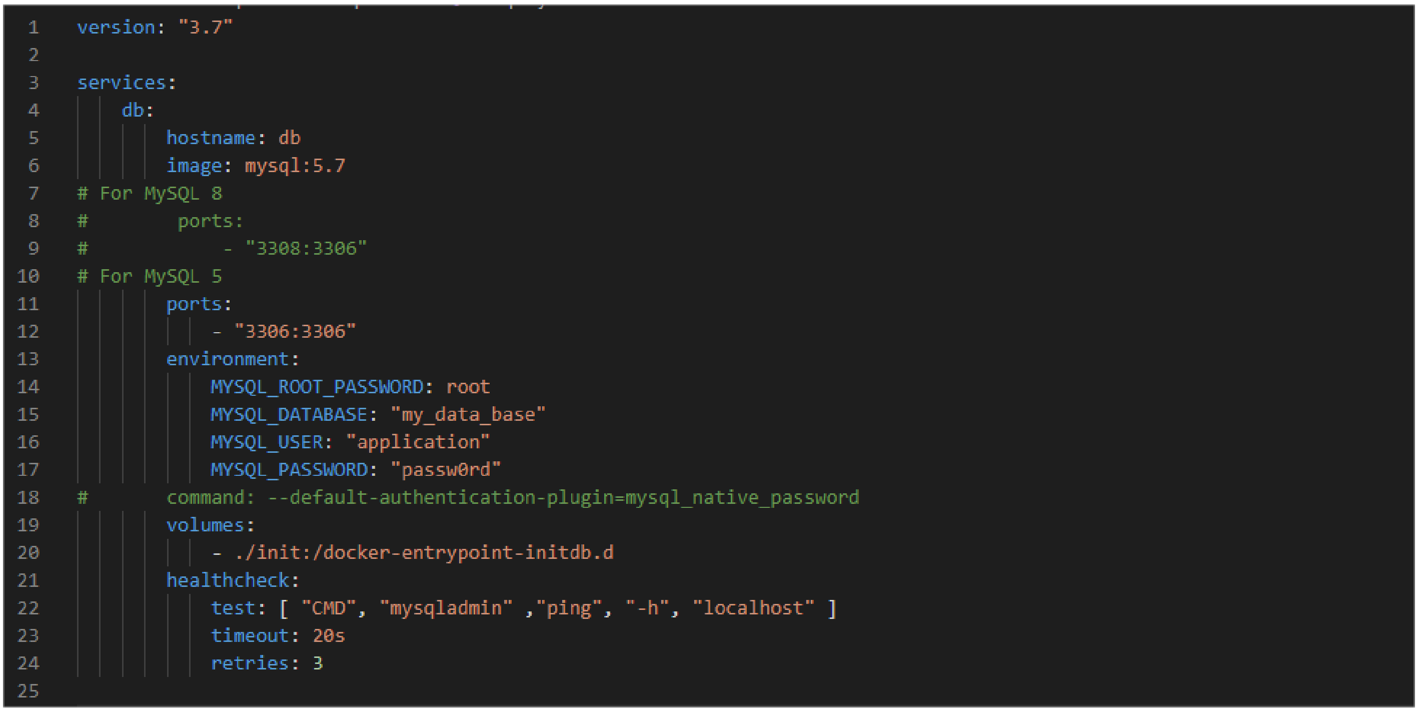
No containers running

Try running a container: Copy and paste this command into your terminal and then come back

```
docker run -d -p 80:80 docker/getting-started
```

Modifier le fichier **maven-gestion-personnes-spring-boot/docker-compose.yaml** pour modifier la version du MySQL pour une version **5.7**.

Le fichier **docker-compose.yaml** aura donc pour contenu :



```yaml
version: "3.7"

services:
    db:
        hostname: db
        image: mysql:5.7
# For MySQL 8
#         ports:
#             - "3308:3306"
# For MySQL 5
        ports:
            - "3306:3306"
        environment:
            MYSQL_ROOT_PASSWORD: root
            MYSQL_DATABASE: "my_data_base"
            MYSQL_USER: "application"
            MYSQL_PASSWORD: "passw0rd"
#       command: --default-authentication-plugin=mysql_native_password
        volumes:
            - ./init:/docker-entrypoint-initdb.d
        healthcheck:
            test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
            timeout: 20s
            retries: 3

```

En version copiable :

```
version: "3.7"

services:
  db:
    hostname: db
    image: mysql:5.7
# For MySQL 8
#      ports:
#        - "3308:3306"
# For MySQL 5
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: "my_data_base"
      MYSQL_USER: "application"
      MYSQL_PASSWORD: "passw0rd"
#     command: --default-authentication-plugin=mysql_native_password
    volumes:
      - ./init:/docker-entrypoint-initdb.d
    healthcheck:
      test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
      timeout: 20s
      retries: 3
```

Ligne 12 : on redirige les données de l'image Docker MySQL 5.7 du port **3306** de Docker vers le port **3306** de notre machine local.
Le fichier **pom.xml** aura la dépendance MySQL suivante :

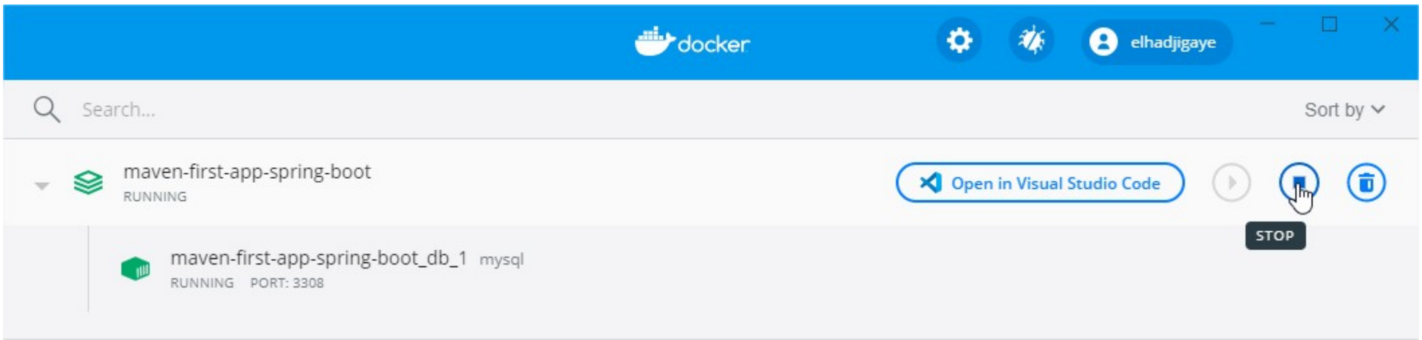
```
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.47</version>
</dependency>
```

Le fichier **application.properties** devient :

```
# For MYSQL 8
#spring.datasource.url=jdbc:mysql://localhost:3308/my_data_base?serverTimezone=UTC
#spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
# For MYSQL 5
spring.datasource.url=jdbc:mysql://localhost:3306/my_data_base?useSSL=false
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.username=application
spring.datasource.password=passw0rd
# For Hibernate version < 5.3.1.Final
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
# For Hibernate version > 5.3.1.Final
#spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
spring.jpa.hibernate.ddl-auto=update
springdoc.swagger-ui.path=/swagger-ui.html
# Manage Error Message
server.error.include-message= always
```

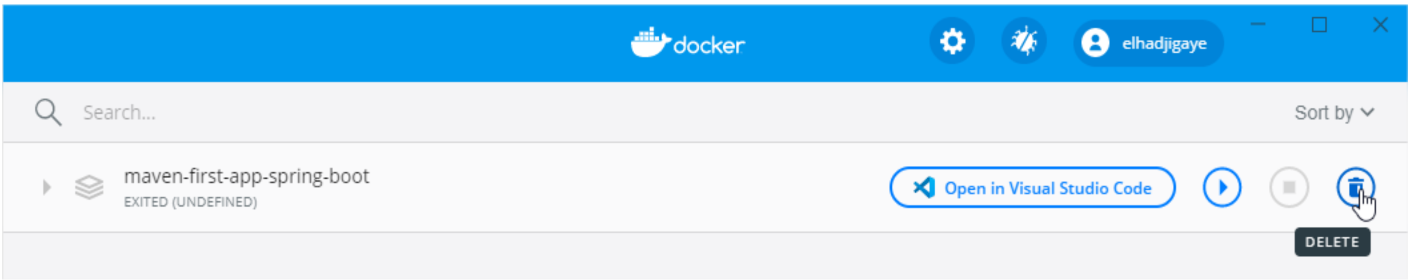
d. Le fichier docker-compose.yaml pour une image MYSQL 8 avec phpmyadmin

Arreter et supprimer le précédent conteneur :

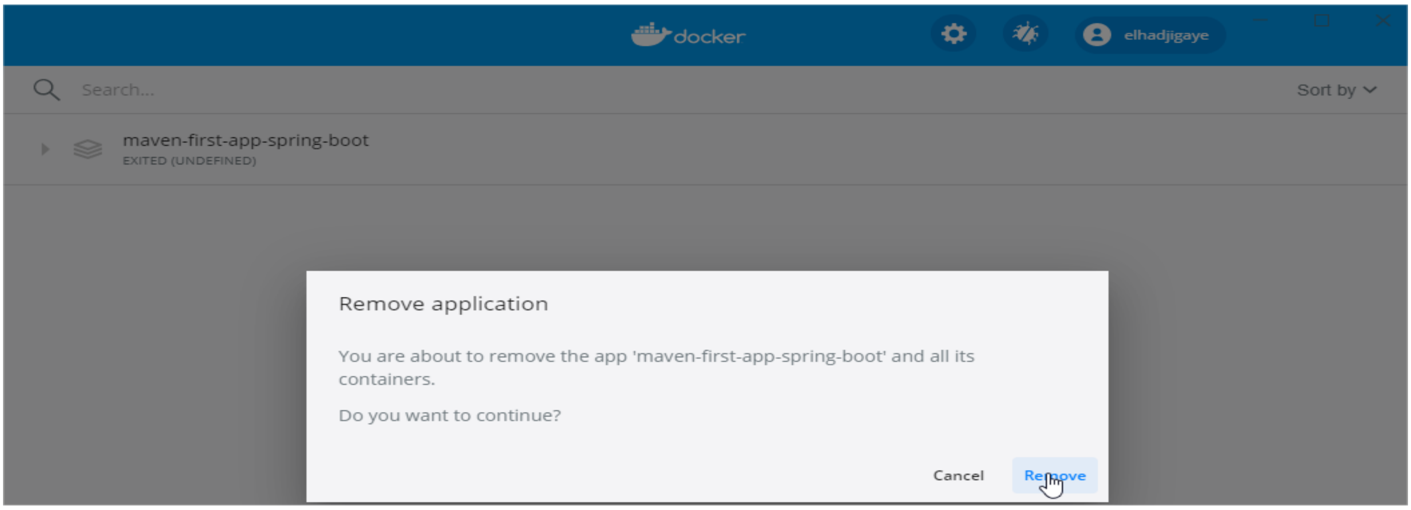


maven-first-app-spring-boot
RUNNING

maven-first-app-spring-boot_db_1 mysql
RUNNING PORT: 3308



maven-first-app-spring-boot
EXITED (UNDEFINED)

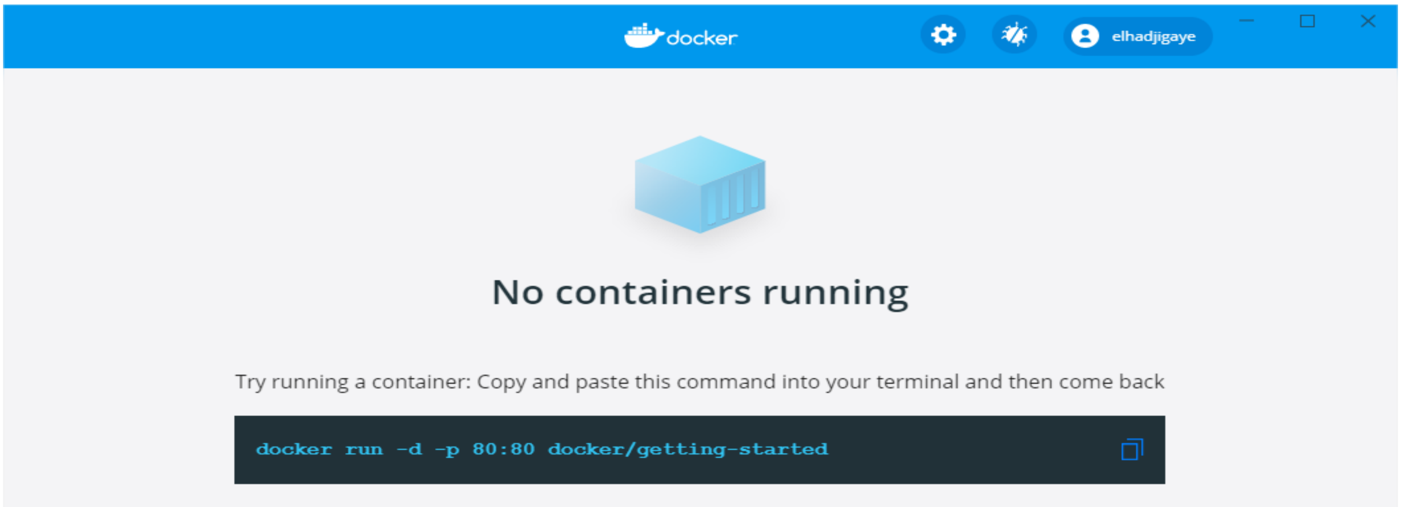


maven-first-app-spring-boot
EXITED (UNDEFINED)

Remove application

You are about to remove the app 'maven-first-app-spring-boot' and all its containers.

Do you want to continue?



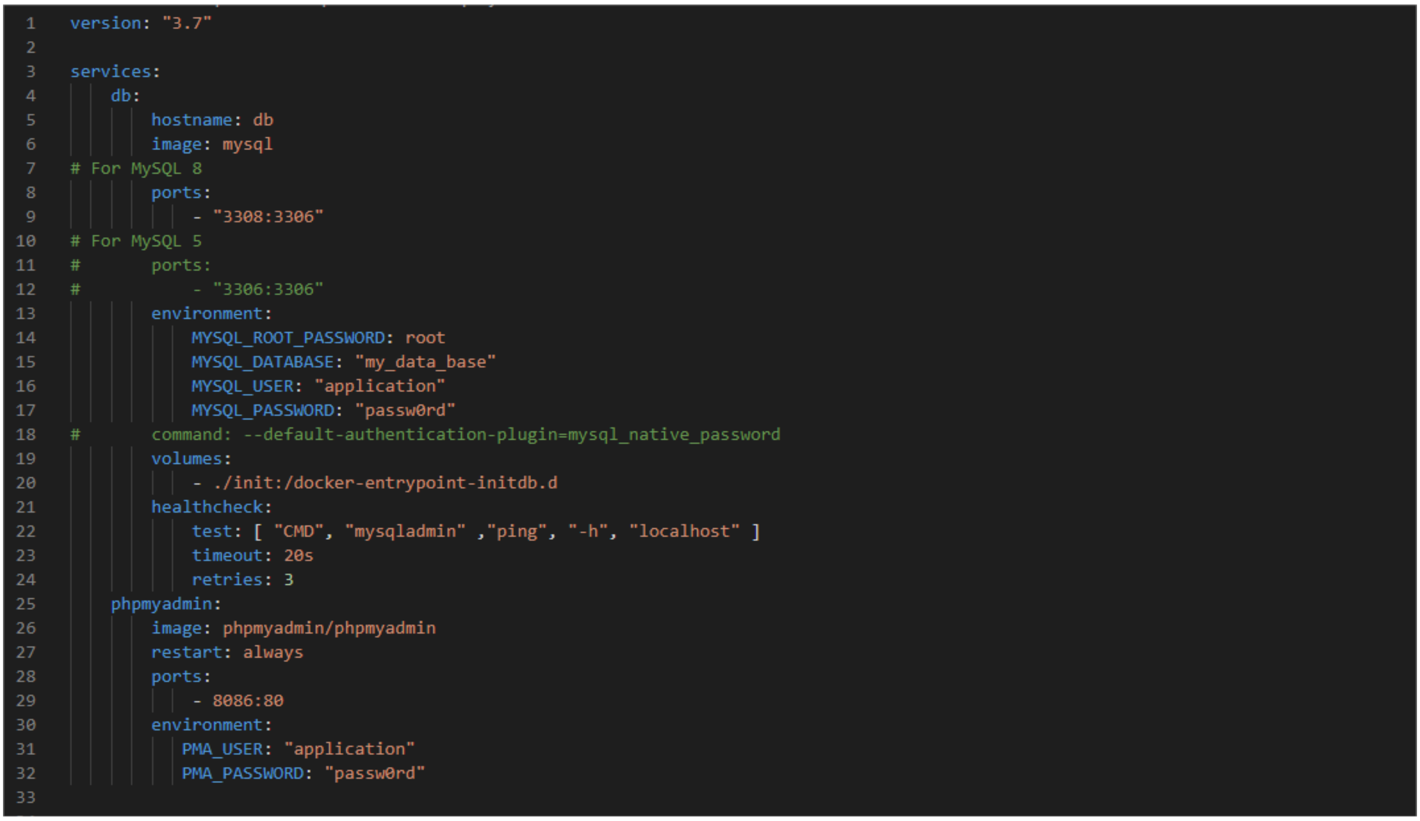
No containers running

Try running a container: Copy and paste this command into your terminal and then come back

```
docker run -d -p 80:80 docker/getting-started
```

Modifier le fichier **maven-gestion-personnes-spring-boot/docker-compose.yaml** pour ajouter phpmyadmin à votre base de donnée MySQL 8.

Ce fichier aura pour contenu :



```yaml
version: "3.7"

services:
    db:
        hostname: db
        image: mysql
# For MySQL 8
        ports:
            - "3308:3306"
# For MySQL 5
#       ports:
#           - "3306:3306"
        environment:
            MYSQL_ROOT_PASSWORD: root
            MYSQL_DATABASE: "my_data_base"
            MYSQL_USER: "application"
            MYSQL_PASSWORD: "passw0rd"
#       command: --default-authentication-plugin=mysql_native_password
        volumes:
            - ./init:/docker-entrypoint-initdb.d
        healthcheck:
            test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
            timeout: 20s
            retries: 3
    phpmyadmin:
        image: phpmyadmin/phpmyadmin
        restart: always
        ports:
            - 8086:80
        environment:
           PMA_USER: "application"
           PMA_PASSWORD: "passw0rd"

```

En version copiable :

```
version: "3.7"

services:
  db:
    hostname: db
    image: mysql
# For MySQL 8
    ports:
      - "3308:3306"
# For MySQL 5
#    ports:
#      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: "my_data_base"
      MYSQL_USER: "application"
      MYSQL_PASSWORD: "passw0rd"
#    command: --default-authentication-plugin=mysql_native_password
    volumes:
      - ./init:/docker-entrypoint-initdb.d
    healthcheck:
      test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
      timeout: 20s
      retries: 3
  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    restart: always
    ports:
      - 8086:80
    environment:
     PMA_USER: "application"
     PMA_PASSWORD: "passw0rd"
```

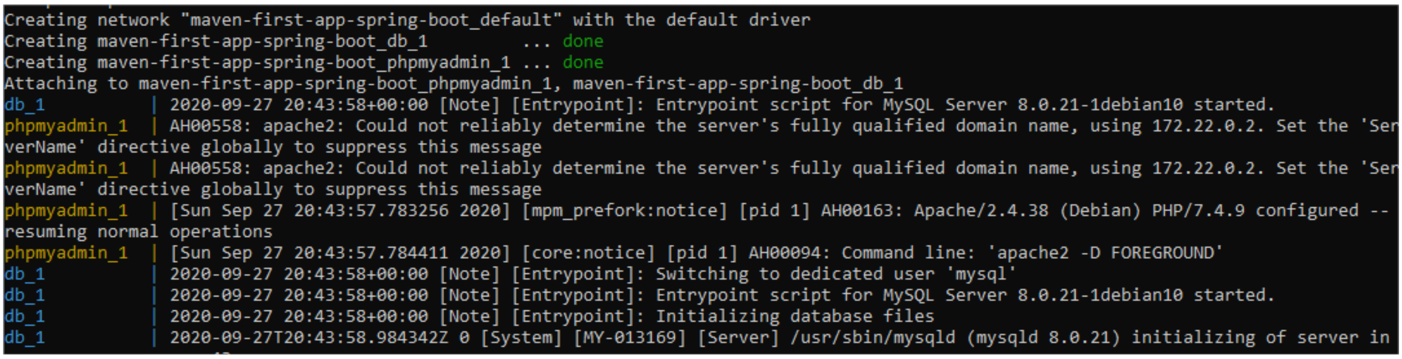
Ligne 9 : on redirige les données de l'image Docker MySQL 8 du port **3306** de Docker vers le port **3308** de notre machine local.
Ligne 29 : on redirige les données de l'image Docker **phpmyadmin** du port **80** de Docker vers le port **8086** de notre machine local.

Lancer ensuite les commandes suivantes :

```
cd maven-first-app-spring-boot
docker-compose build
docker-compose up
```

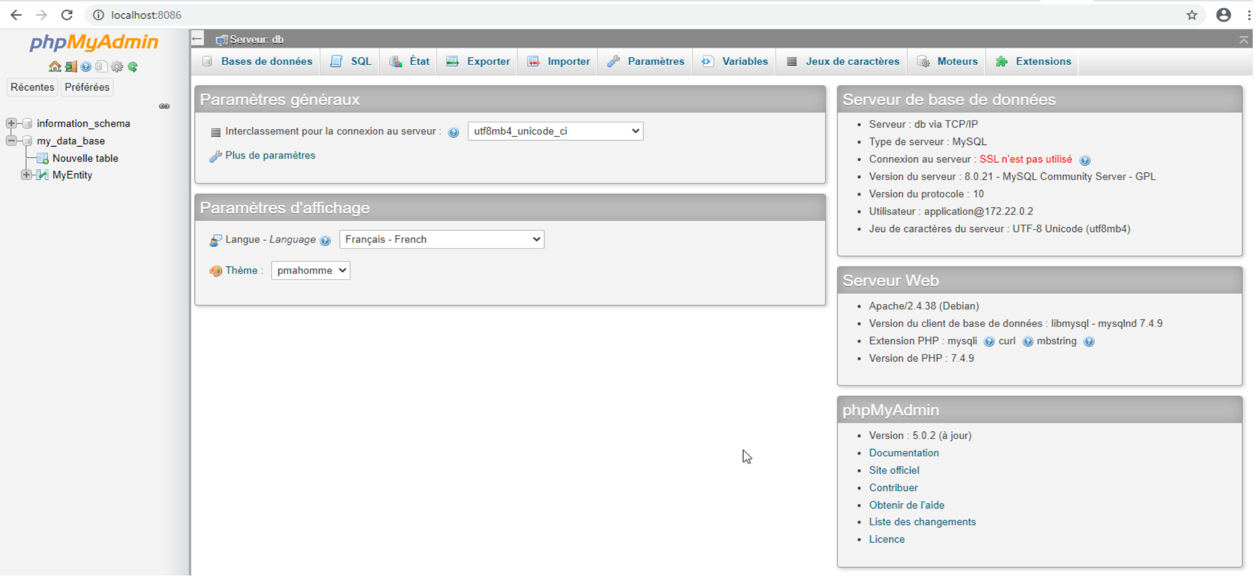
En spécifiant le fichier docker compose à utiliser, on obtient :

```
docker-compose –f docker-compose-with-phpmyadmin.yaml build
docker-compose –f docker-compose-with-phpmyadmin.yaml up
```



```
Creating network "maven-first-app-spring-boot_default" with the default driver
Creating maven-first-app-spring-boot_db_1         ... done
Creating maven-first-app-spring-boot_phpmyadmin_1 ... done
Attaching to maven-first-app-spring-boot_phpmyadmin_1, maven-first-app-spring-boot_db_1
db_1            | 2020-09-27 20:43:58+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
phpmyadmin_1    | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.22.0.2. Set the 'Ser
verName' directive globally to suppress this message
phpmyadmin_1    | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.22.0.2. Set the 'Ser
verName' directive globally to suppress this message
phpmyadmin_1    | [Sun Sep 27 20:43:57.783256 2020] [mpm_prefork:notice] [pid 1] AH00163: Apache/2.4.38 (Debian) PHP/7.4.9 configured --
resuming normal operations
phpmyadmin_1    | [Sun Sep 27 20:43:57.784411 2020] [core:notice] [pid 1] AH00094: Command line: 'apache2 -D FOREGROUND'
db_1            | 2020-09-27 20:43:58+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
db_1            | 2020-09-27 20:43:58+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1            | 2020-09-27 20:43:58+00:00 [Note] [Entrypoint]: Initializing database files
db_1            | 2020-09-27T20:43:58.984342Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.21) initializing of server in
```

Lancer l'URL de phpmyadmin sur http://localhost:8086/



Paramètres généraux

Interclassement pour la connexion au serveur : utf8mb4_unicode_ci

Plus de paramètres

Paramètres d'affichage

Langue - *Language* : Français - French

Thème : pmahomme

Serveur de base de données

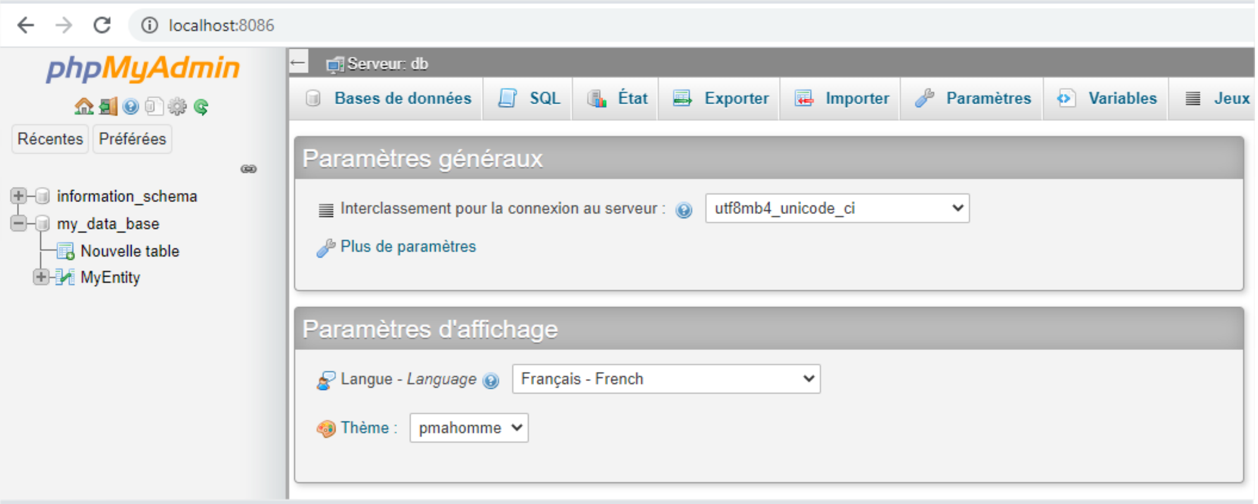
- Serveur : db via TCP/IP
- Type de serveur : MySQL
- Connexion au serveur : SSL n'est pas utilisé
- Version du serveur : 8.0.21 - MySQL Community Server - GPL
- Version du protocole : 10
- Utilisateur : application@172.22.0.2
- Jeu de caractères du serveur : UTF-8 Unicode (utf8mb4)

Serveur Web

- Apache/2.4.38 (Debian)
- Version du client de base de données : libmysql - mysqlnd 7.4.9
- Extension PHP : mysqli curl mbstring
- Version de PHP : 7.4.9

phpMyAdmin

- Version : 5.0.2 (à jour)
- Documentation
- Site officiel
- Contribuer
- Obtenir de l'aide
- Liste des changements
- Licence



Paramètres généraux

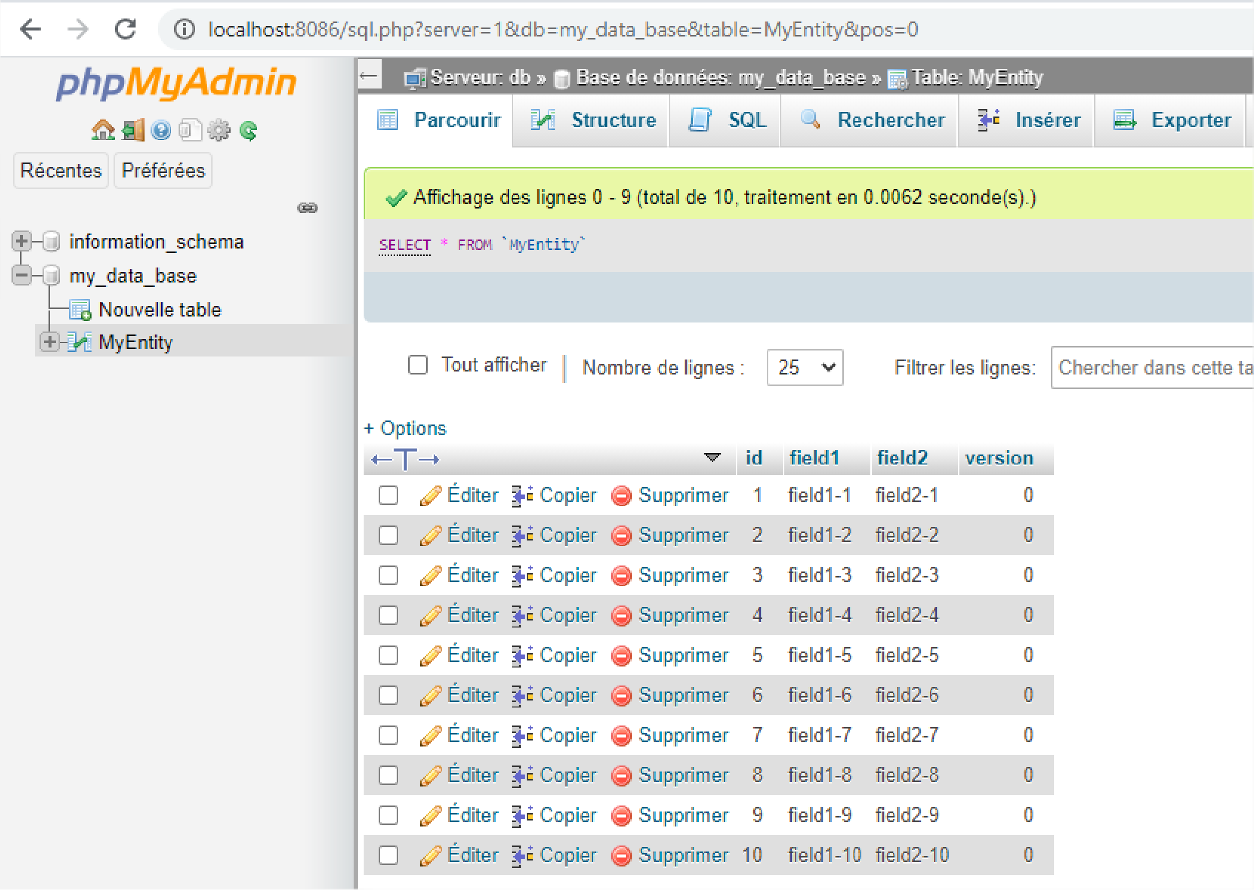
Interclassement pour la connexion au serveur : utf8mb4_unicode_ci

Plus de paramètres

Paramètres d'affichage

Langue - *Language* Français - French

Thème : pmahomme


localhost:8086/sql.php?server=1&db=my_data_base&table=MyEntity&pos=0
phpMyAdmin
Récentes
Préférées
information_schema
my_data_base
Nouvelle table
MyEntity
Serveur: db » Base de données: my_data_base » Table: MyEntity
Parcourir
Structure
SQL
Rechercher
Insérer
Exporter

✓ Affichage des lignes 0 - 9 (total de 10, traitement en 0.0062 seconde(s).)

```
SELECT * FROM `MyEntity`
```

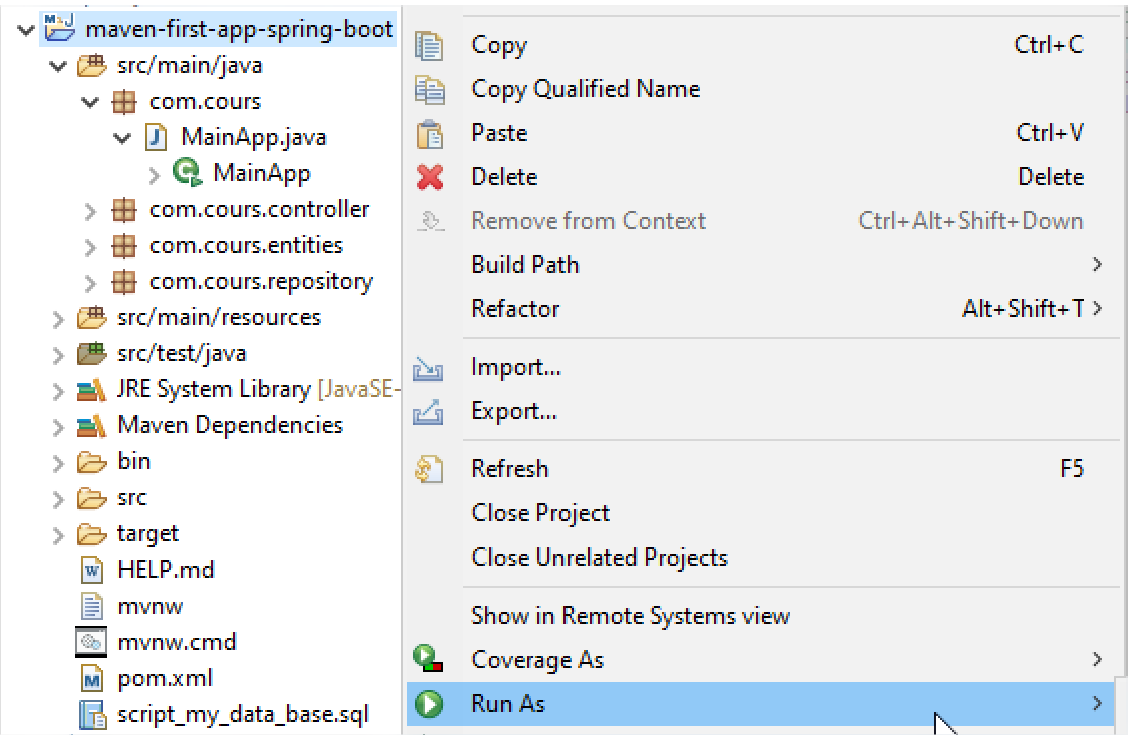
□ Tout afficher | Nombre de lignes : 25 Filtrer les lignes: Chercher dans cette ta

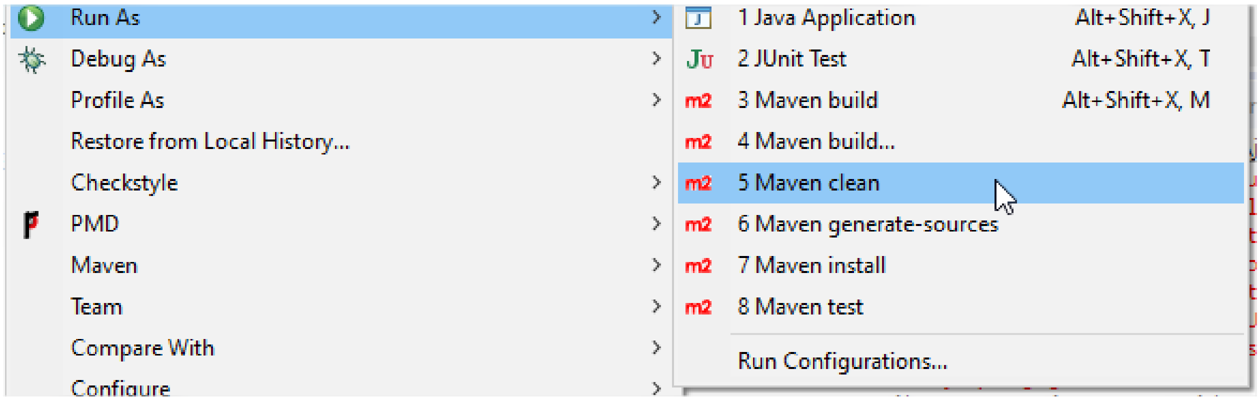
+ Options

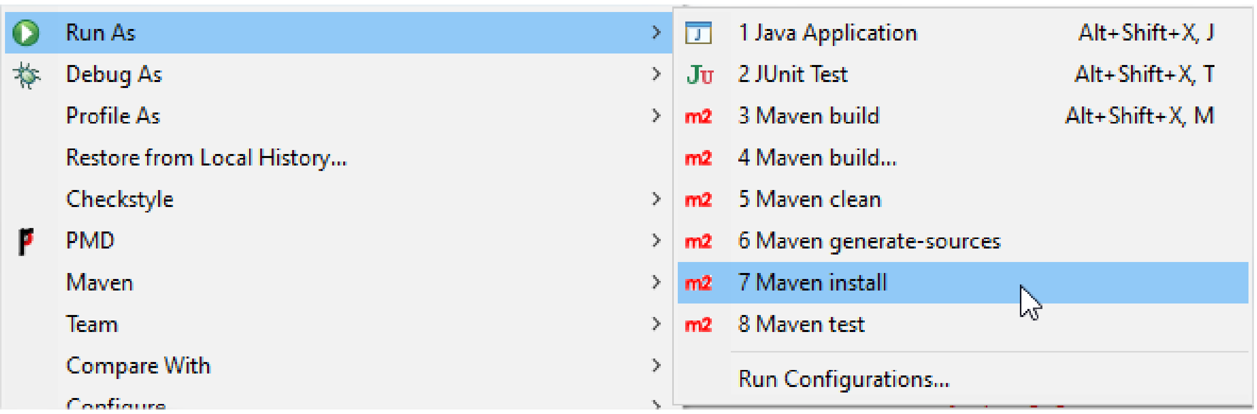
	id	field1	field2	version
□ Éditer Copier Supprimer	1	field1-1	field2-1	0
□ Éditer Copier Supprimer	2	field1-2	field2-2	0
□ Éditer Copier Supprimer	3	field1-3	field2-3	0
□ Éditer Copier Supprimer	4	field1-4	field2-4	0
□ Éditer Copier Supprimer	5	field1-5	field2-5	0
□ Éditer Copier Supprimer	6	field1-6	field2-6	0
□ Éditer Copier Supprimer	7	field1-7	field2-7	0
□ Éditer Copier Supprimer	8	field1-8	field2-8	0
□ Éditer Copier Supprimer	9	field1-9	field2-9	0
□ Éditer Copier Supprimer	10	field1-10	field2-10	0

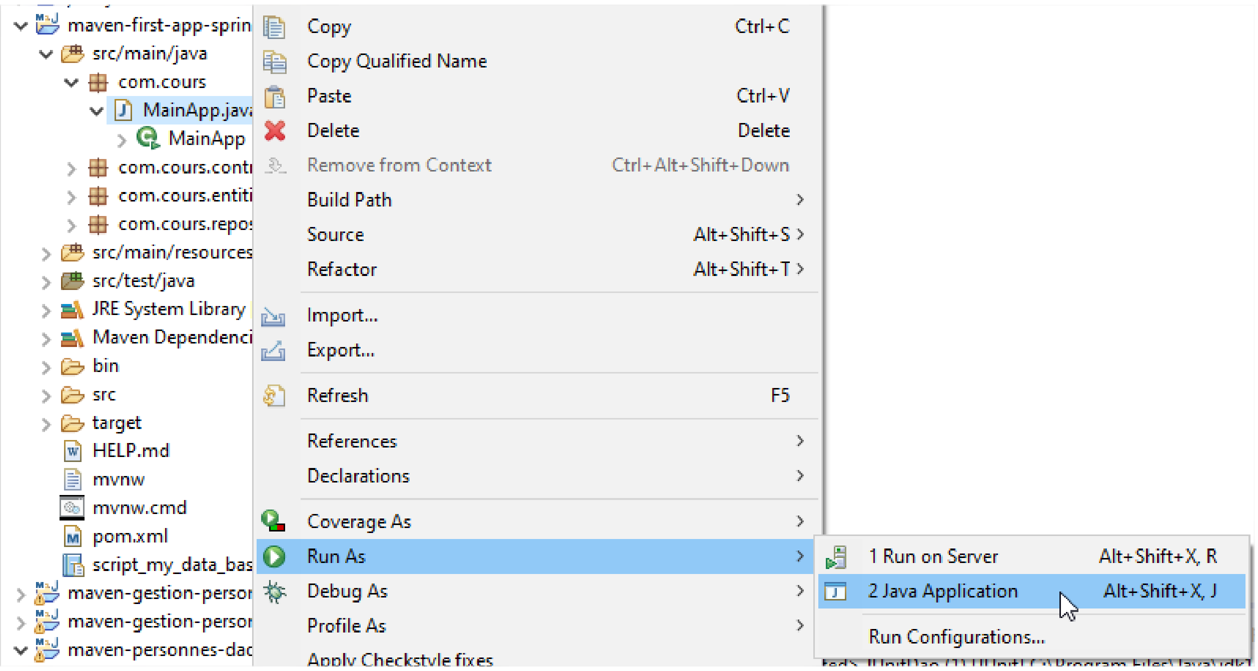
e. Lancement de l’application avec Eclipse

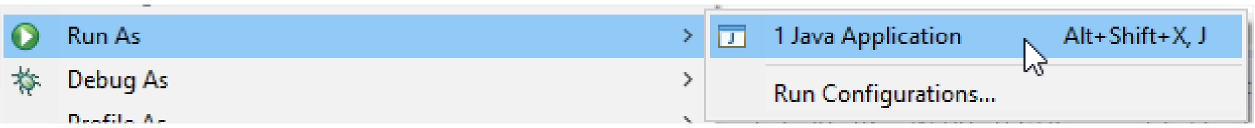
Une fois que votre image Docker de base de données est lancée, lancer votre Micro Service.

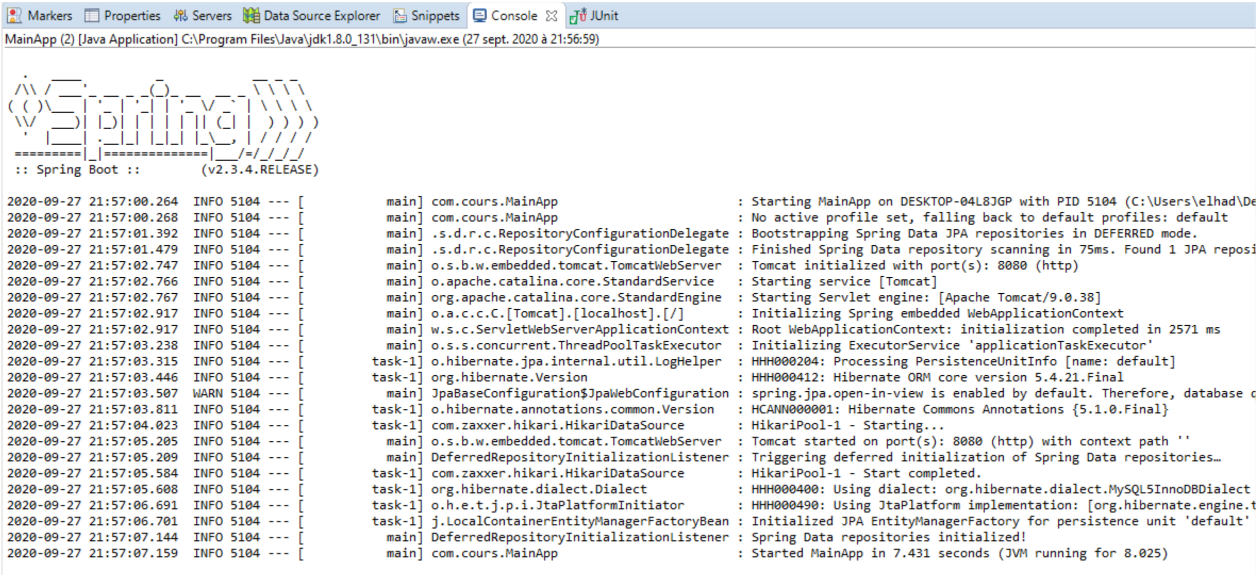








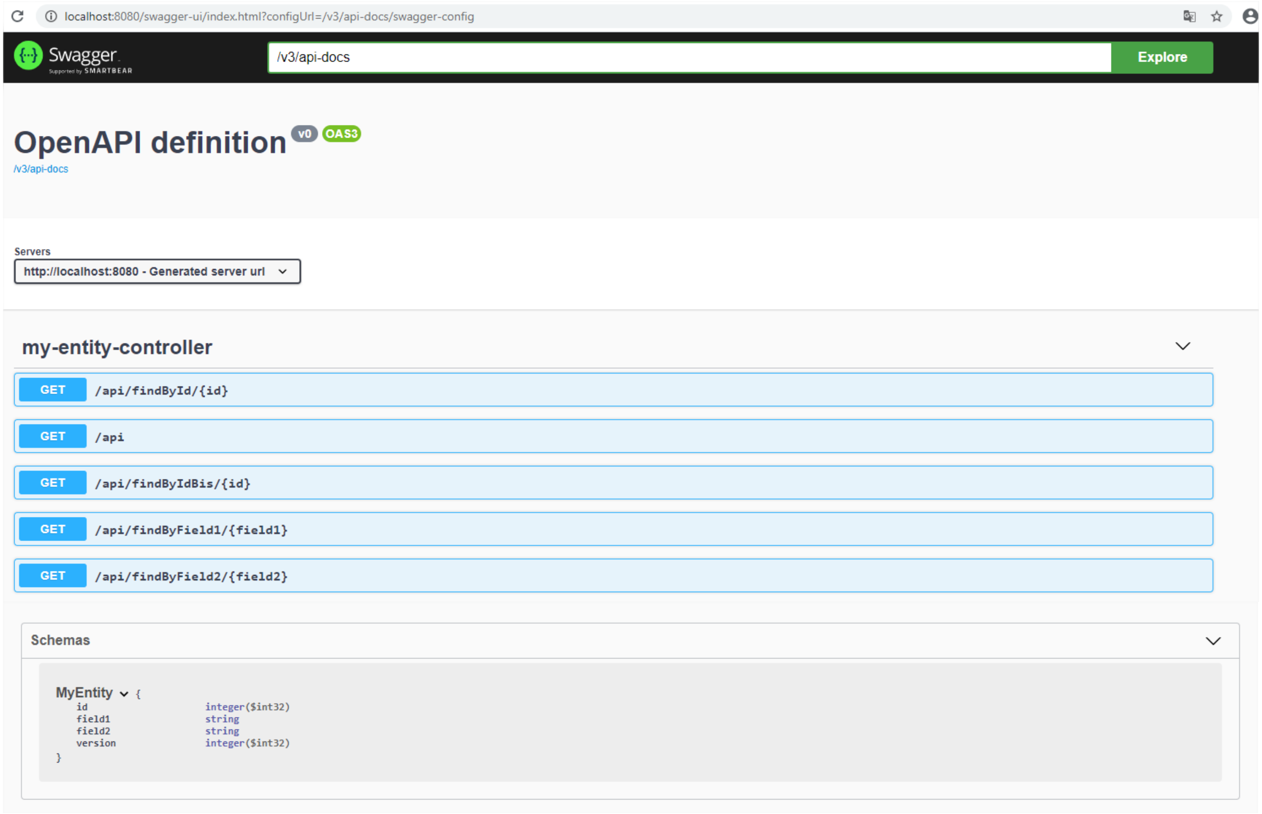




MainApp (2) [Java Application] C:\Program Files\Java\jdk1.8.0_131\bin\javaw.exe (27 sept. 2020 à 21:56:59)

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.3.4.RELEASE)

2020-09-27 21:57:00.264  INFO 5104 --- [           main] com.cours.MainApp                        : Starting MainApp on DESKTOP-04L8JGP with PID 5104 (C:\Users\elhad\De
2020-09-27 21:57:00.268  INFO 5104 --- [           main] com.cours.MainApp                        : No active profile set, falling back to default profiles: default
2020-09-27 21:57:01.392  INFO 5104 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFERRED mode.
2020-09-27 21:57:01.479  INFO 5104 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 75ms. Found 1 JPA reposi
2020-09-27 21:57:02.747  INFO 5104 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2020-09-27 21:57:02.766  INFO 5104 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2020-09-27 21:57:02.767  INFO 5104 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-09-27 21:57:02.917  INFO 5104 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2020-09-27 21:57:02.917  INFO 5104 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2571 ms
2020-09-27 21:57:03.238  INFO 5104 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2020-09-27 21:57:03.315  INFO 5104 --- [         task-1] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2020-09-27 21:57:03.446  INFO 5104 --- [         task-1] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.4.21.Final
2020-09-27 21:57:03.507  WARN 5104 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database c
2020-09-27 21:57:03.811  INFO 5104 --- [         task-1] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
2020-09-27 21:57:04.023  INFO 5104 --- [         task-1] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2020-09-27 21:57:05.205  INFO 5104 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2020-09-27 21:57:05.209  INFO 5104 --- [           main] DeferredRepositoryInitializationListener : Triggering deferred initialization of Spring Data repositories…
2020-09-27 21:57:05.584  INFO 5104 --- [         task-1] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2020-09-27 21:57:05.608  INFO 5104 --- [         task-1] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.MySQL5InnoDBDialect
2020-09-27 21:57:06.691  INFO 5104 --- [         task-1] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.t
2020-09-27 21:57:06.701  INFO 5104 --- [         task-1] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2020-09-27 21:57:07.144  INFO 5104 --- [           main] DeferredRepositoryInitializationListener : Spring Data repositories initialized!
2020-09-27 21:57:07.159  INFO 5104 --- [           main] com.cours.MainApp                        : Started MainApp in 7.431 seconds (JVM running for 8.025)
```

Lancer l'URL http://localhost:8080/swagger-ui.html



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config

/v3/api-docs

OpenAPI definition v0 OAS3

/v3/api-docs

Servers

http://localhost:8080 - Generated server url

my-entity-controller

GET /api/findById/{id}

GET /api

GET /api/findByIdBis/{id}

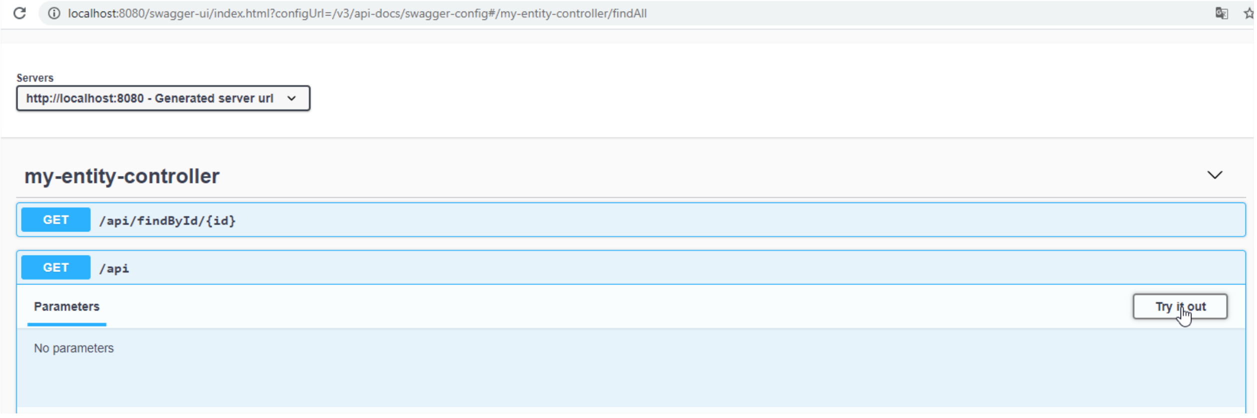
GET /api/findByField1/{field1}

GET /api/findByField2/{field2}

Schemas

```
MyEntity ⌄ {
    id              integer($int32)
    field1          string
    field2          string
    version         integer($int32)
}
```

En cliquant sur **GET/api** on obtient :



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/my-entity-controller/findAll

Servers

http://localhost:8080 - Generated server url

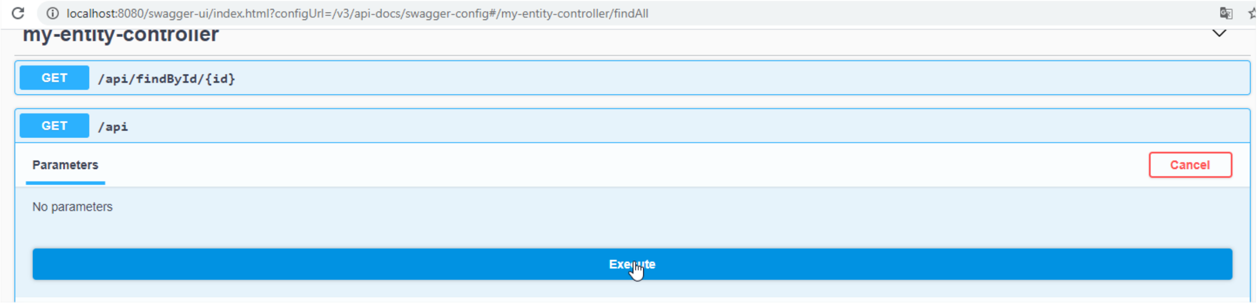
my-entity-controller

GET /api/findById/{id}

GET /api

Parameters

No parameters



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/my-entity-controller/findAll

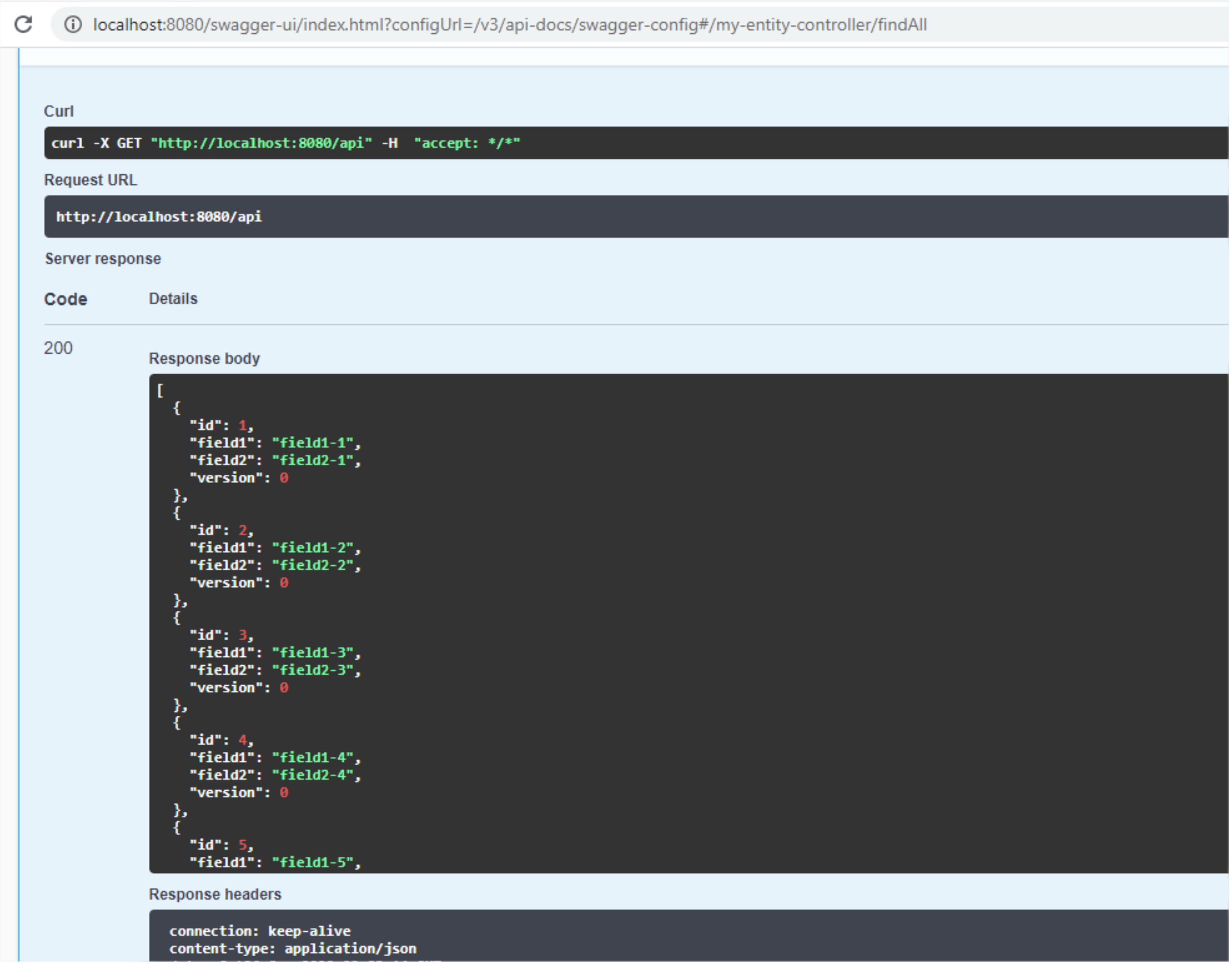
my-entity-controller

GET /api/findById/{id}

GET /api

Parameters

No parameters



localhost:8080/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config#/my-entity-controller/findAll

Curl

```
curl -X GET "http://localhost:8080/api" -H  "accept: */*"
```

Request URL

```
http://localhost:8080/api
```

Server response

Code Details

200

Response body

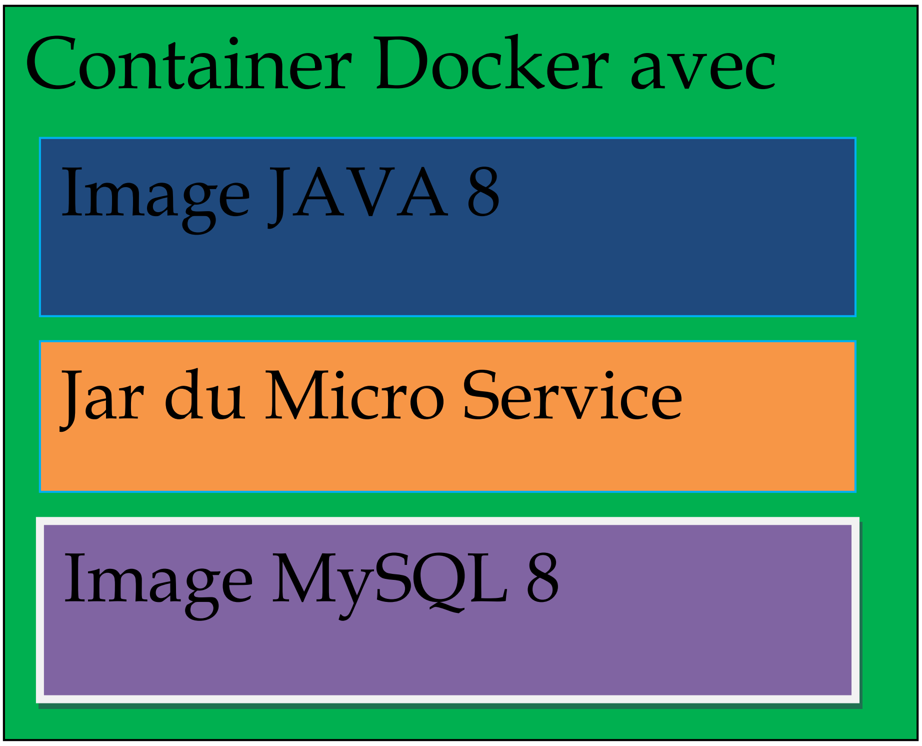
```
[
  {
    "id": 1,
    "field1": "field1-1",
    "field2": "field2-1",
    "version": 0
  },
  {
    "id": 2,
    "field1": "field1-2",
    "field2": "field2-2",
    "version": 0
  },
  {
    "id": 3,
    "field1": "field1-3",
    "field2": "field2-3",
    "version": 0
  },
  {
    "id": 4,
    "field1": "field1-4",
    "field2": "field2-4",
    "version": 0
  },
  {
    "id": 5,
    "field1": "field1-5",
```

Response headers

```
connection: keep-alive
content-type: application/json
```

3. *Version 3 : Image Docker contenant une image JDK 8 alpine (avec Jar du Micro Service Spring Boot) + une autre image Docker MySQL 8*

L'architecture de l'application ressemblera cette fois ci à :

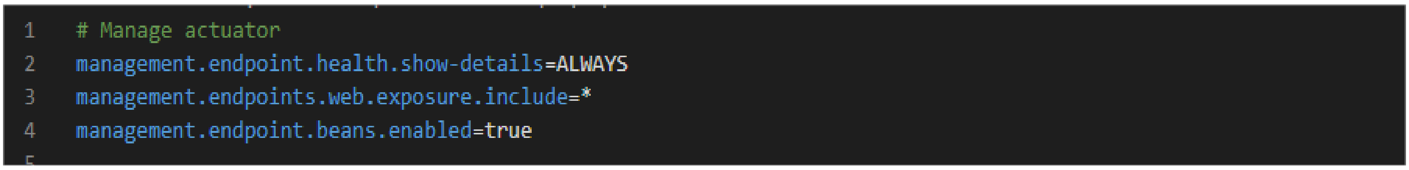


Nous allons introduire la notion de l'état ou santé du serveur à travers **actuator/metrics, actuator/health et actuator/beans**.

Mettre à jour votre fichier **pom.xml** en ajoutant la dépendance :

```xml
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

Mettre à jour le fichier **application.properties** en ajoutant :

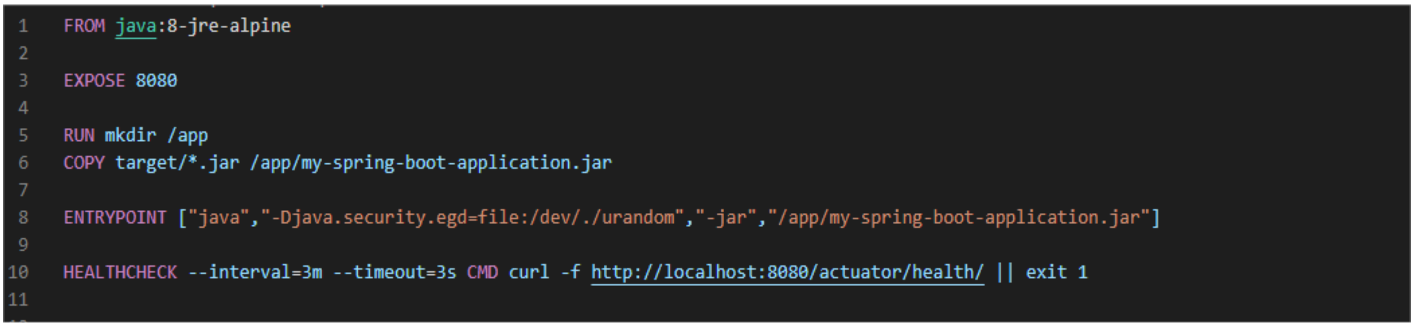


```
# Manage actuator
management.endpoint.health.show-details=ALWAYS
management.endpoints.web.exposure.include=*
management.endpoint.beans.enabled=true
```

En version copiable :

```
# Manage actuator
management.endpoint.health.show-details=ALWAYS
management.endpoints.web.exposure.include=*
management.endpoint.beans.enabled=true
```

Créer le fichier **maven-first-app-spring-boot/Dockerfile** dont le contenu sera :

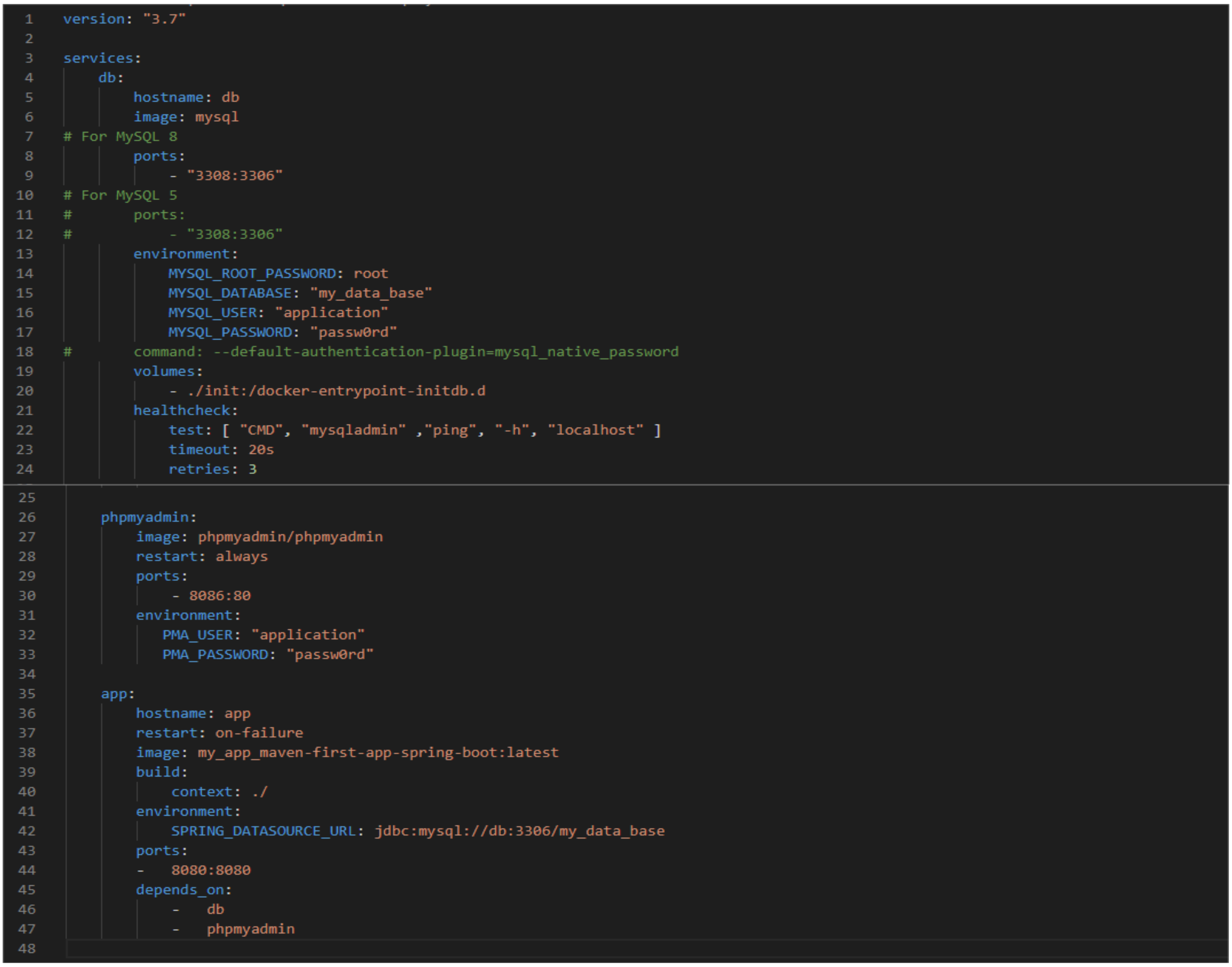


```
FROM java:8-jre-alpine

EXPOSE 8080

RUN mkdir /app
COPY target/*.jar /app/my-spring-boot-application.jar

ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app/my-spring-boot-application.jar"]

HEALTHCHECK --interval=3m --timeout=3s CMD curl -f http://localhost:8080/actuator/health/ || exit 1

```

En version copiable :

```
FROM java:8-jre-alpine

EXPOSE 8080

RUN mkdir /app
COPY target/*.jar /app/my-spring-boot-application.jar

ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app/my-spring-boot-application.jar"]

HEALTHCHECK --interval=3m --timeout=3s CMD curl -f http://localhost:8080/actuator/health/ || exit 1
```

le fichier **maven-gestion-personnes-spring-boot/docker-compose.yaml :**



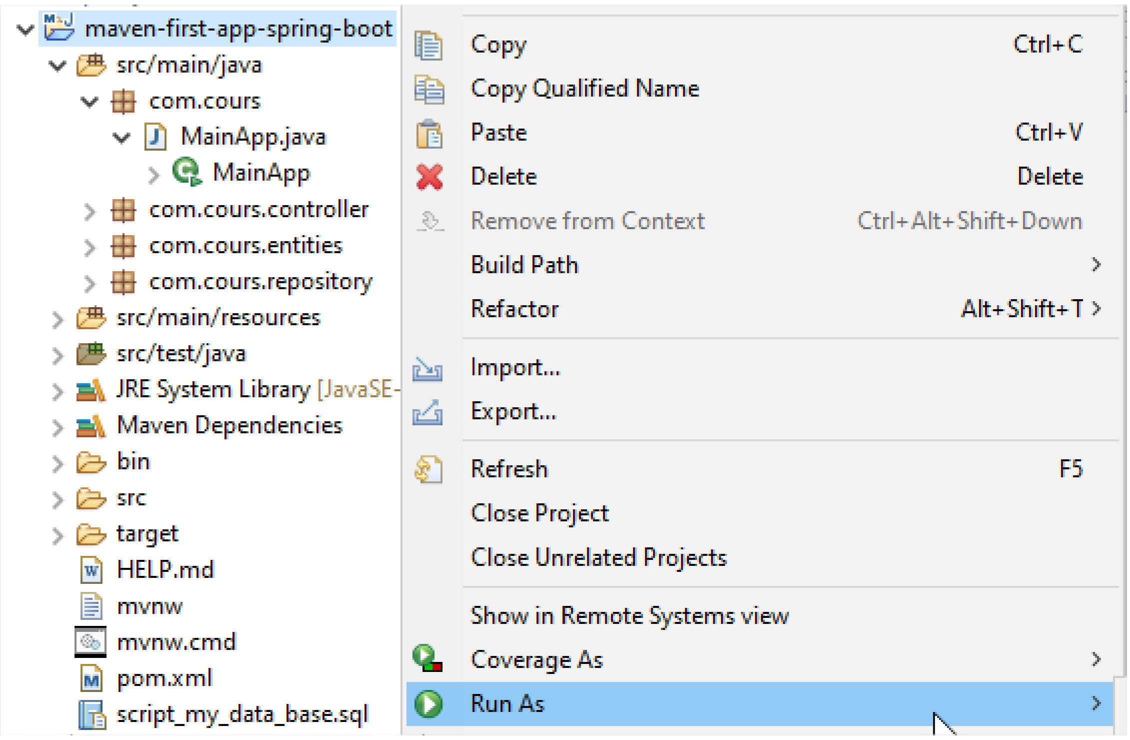
```yaml
version: "3.7"

services:
    db:
        hostname: db
        image: mysql
# For MySQL 8
        ports:
            - "3308:3306"
# For MySQL 5
#       ports:
#           - "3308:3306"
        environment:
            MYSQL_ROOT_PASSWORD: root
            MYSQL_DATABASE: "my_data_base"
            MYSQL_USER: "application"
            MYSQL_PASSWORD: "passw0rd"
#       command: --default-authentication-plugin=mysql_native_password
        volumes:
            - ./init:/docker-entrypoint-initdb.d
        healthcheck:
            test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
            timeout: 20s
            retries: 3

    phpmyadmin:
        image: phpmyadmin/phpmyadmin
        restart: always
        ports:
            - 8086:80
        environment:
            PMA_USER: "application"
            PMA_PASSWORD: "passw0rd"

    app:
        hostname: app
        restart: on-failure
        image: my_app_maven-first-app-spring-boot:latest
        build:
            context: ./
        environment:
            SPRING_DATASOURCE_URL: jdbc:mysql://db:3306/my_data_base
        ports:
        -   8080:8080
        depends_on:
            -   db
            -   phpmyadmin

```

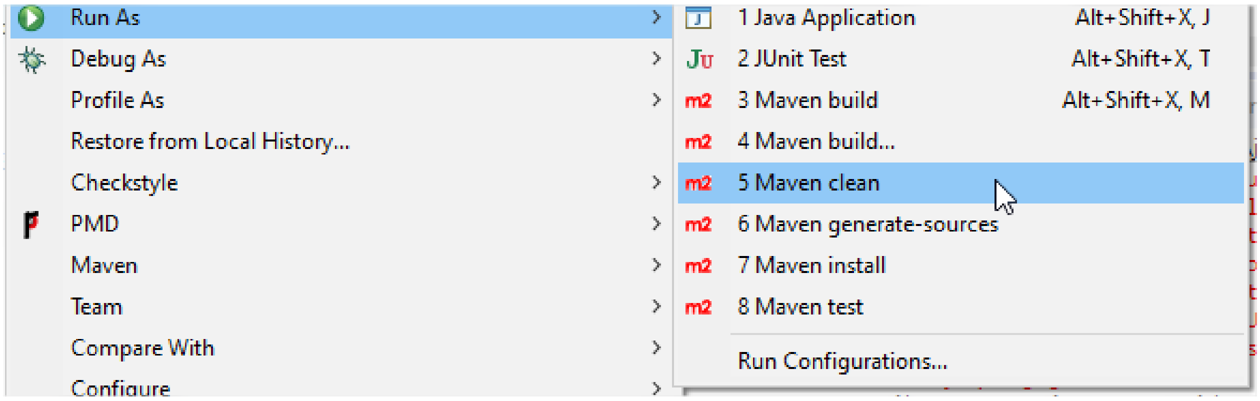
En version copiable :

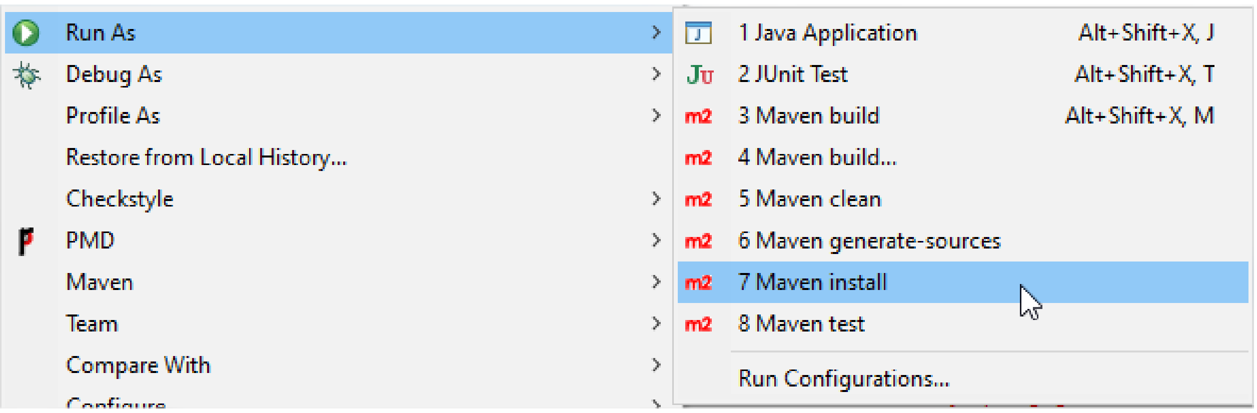
```
version: "3.7"

services:
  db:
    hostname: db
    image: mysql
# For MySQL 8
    ports:
      - "3308:3306"
# For MySQL 5
#      ports:
#          - "3308:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: "my_data_base"
      MYSQL_USER: "application"
      MYSQL_PASSWORD: "passw0rd"
#      command: --default-authentication-plugin=mysql_native_password
    volumes:
      - ./init:/docker-entrypoint-initdb.d
    healthcheck:
      test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
      timeout: 20s
      retries: 3

  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    restart: always
    ports:
      - 8086:80
    environment:
     PMA_USER: "application"
     PMA_PASSWORD: "passw0rd"

  app:
    hostname: app
    restart: on-failure
    image: my_app_maven-first-app-spring-boot:latest
    build:
      context: ./
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://db:3306/my_data_base
    ports:
    -   8080:8080
    depends_on:
      -   db
      -   phpmyadmin
```

Ligne 4-24 : configuration de l'image Docker **MySql 8**.
Ligne 26-33 : configuration de l'image Docker **phpmyadmin**.
Ligne 35-47 : configuration de l'image Docker du **Micro Service Spring Boot**..

Générer le jar **maven-first-app-spring-boot/target/maven-first-app-spring-boot-0.0.1-SNAPSHOT.jar** avec Eclipse ou un autre programme. Ce jar deviendra **my-spring-boot-application.jar** dans l'execution du fichier **maven-first-app-spring-boot/Dockerfile**.

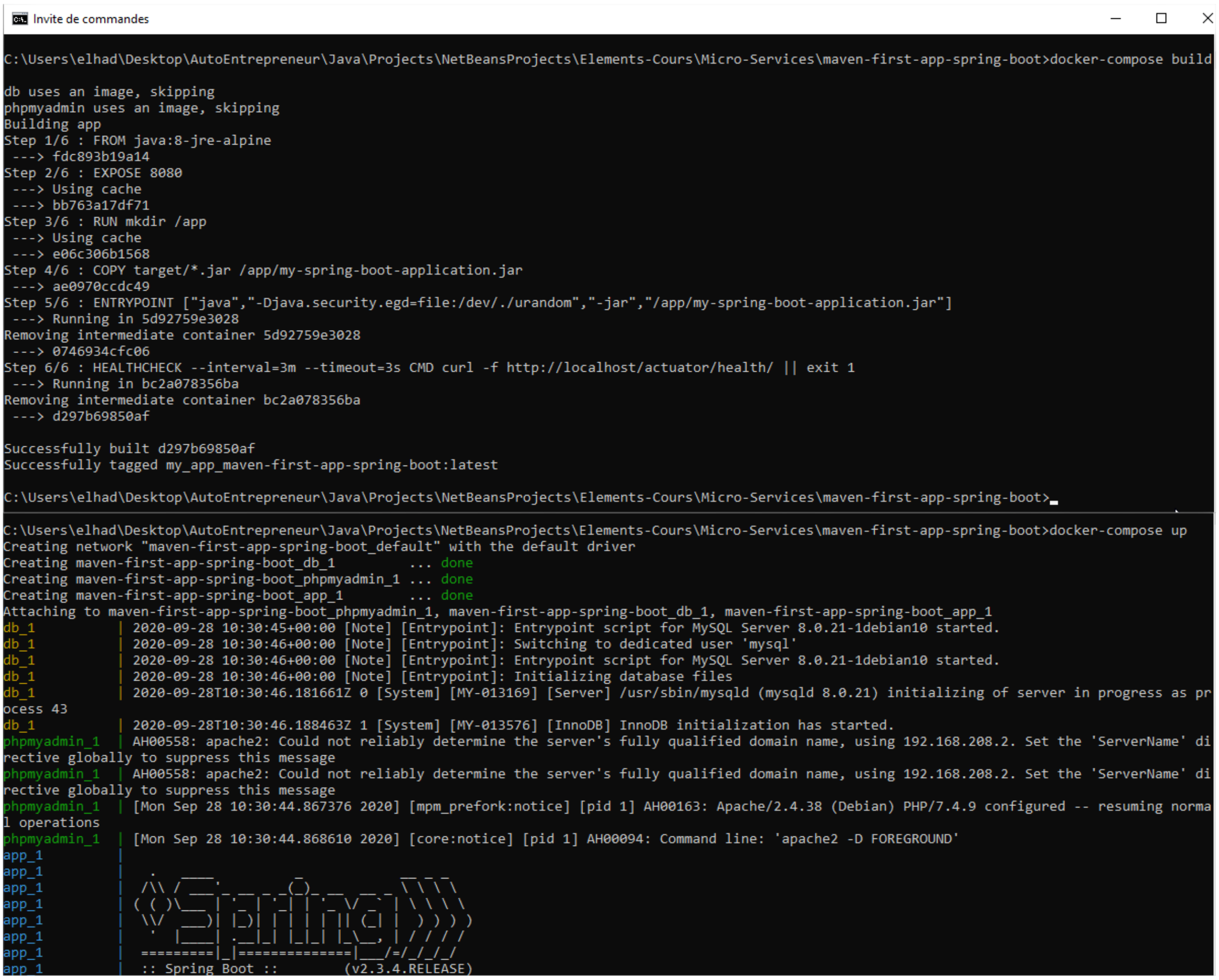






Lancer ensuite les commandes suivantes :

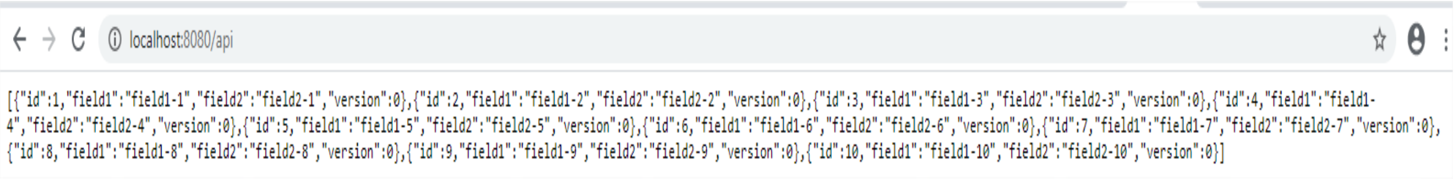
```
cd maven-first-app-spring-boot
docker-compose build
docker-compose up
```



```
C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansProjects\Elements-Cours\Micro-Services\maven-first-app-spring-boot>docker-compose build

db uses an image, skipping
phpmyadmin uses an image, skipping
Building app
Step 1/6 : FROM java:8-jre-alpine
 ---> fdc893b19a14
Step 2/6 : EXPOSE 8080
 ---> Using cache
 ---> bb763a17df71
Step 3/6 : RUN mkdir /app
 ---> Using cache
 ---> e06c306b1568
Step 4/6 : COPY target/*.jar /app/my-spring-boot-application.jar
 ---> ae0970ccdc49
Step 5/6 : ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app/my-spring-boot-application.jar"]
 ---> Running in 5d92759e3028
Removing intermediate container 5d92759e3028
 ---> 0746934cfc06
Step 6/6 : HEALTHCHECK --interval=3m --timeout=3s CMD curl -f http://localhost/actuator/health/ || exit 1
 ---> Running in bc2a078356ba
Removing intermediate container bc2a078356ba
 ---> d297b69850af

Successfully built d297b69850af
Successfully tagged my_app_maven-first-app-spring-boot:latest

C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansProjects\Elements-Cours\Micro-Services\maven-first-app-spring-boot>_
```

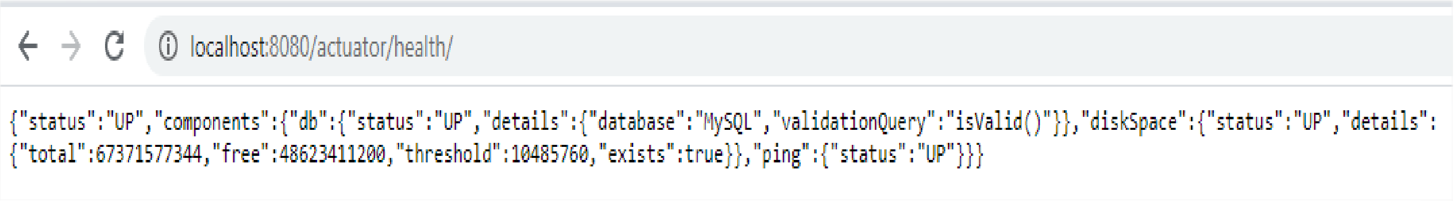
```
C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansProjects\Elements-Cours\Micro-Services\maven-first-app-spring-boot>docker-compose up
Creating network "maven-first-app-spring-boot_default" with the default driver
Creating maven-first-app-spring-boot_db_1         ... done
Creating maven-first-app-spring-boot_phpmyadmin_1 ... done
Creating maven-first-app-spring-boot_app_1        ... done
Attaching to maven-first-app-spring-boot_phpmyadmin_1, maven-first-app-spring-boot_db_1, maven-first-app-spring-boot_app_1
db_1           | 2020-09-28 10:30:45+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1           | 2020-09-28 10:30:46+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
db_1           | 2020-09-28 10:30:46+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1           | 2020-09-28 10:30:46+00:00 [Note] [Entrypoint]: Initializing database files
db_1           | 2020-09-28T10:30:46.181661Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.21) initializing of server in progress as pr
ocess 43
db_1           | 2020-09-28T10:30:46.188463Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
phpmyadmin_1   | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 192.168.208.2. Set the 'ServerName' di
rective globally to suppress this message
phpmyadmin_1   | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 192.168.208.2. Set the 'ServerName' di
rective globally to suppress this message
phpmyadmin_1   | [Mon Sep 28 10:30:44.867376 2020] [mpm_prefork:notice] [pid 1] AH00163: Apache/2.4.38 (Debian) PHP/7.4.9 configured -- resuming norma
l operations
phpmyadmin_1   | [Mon Sep 28 10:30:44.868610 2020] [core:notice] [pid 1] AH00094: Command line: 'apache2 -D FOREGROUND'
app_1          |
app_1          |   .   ____          _            __ _ _
app_1          |  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
app_1          | ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
app_1          |  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
app_1          |   '  |____| .__|_| |_|_| |_\__, | / / / /
app_1          |  =========|_|==============|___/=/_/_/_/
app_1          |  :: Spring Boot ::        (v2.3.4.RELEASE)
```

On obtient sur http://localhost:8080/api :



```
[{"id":1,"field1":"field1-1","field2":"field2-1","version":0},{"id":2,"field1":"field1-2","field2":"field2-2","version":0},{"id":3,"field1":"field1-3","field2":"field2-3","version":0},{"id":4,"field1":"field1-
4","field2":"field2-4","version":0},{"id":5,"field1":"field1-5","field2":"field2-5","version":0},{"id":6,"field1":"field1-6","field2":"field2-6","version":0},{"id":7,"field1":"field1-7","field2":"field2-7","version":0},
{"id":8,"field1":"field1-8","field2":"field2-8","version":0},{"id":9,"field1":"field1-9","field2":"field2-9","version":0},{"id":10,"field1":"field1-10","field2":"field2-10","version":0}]
```

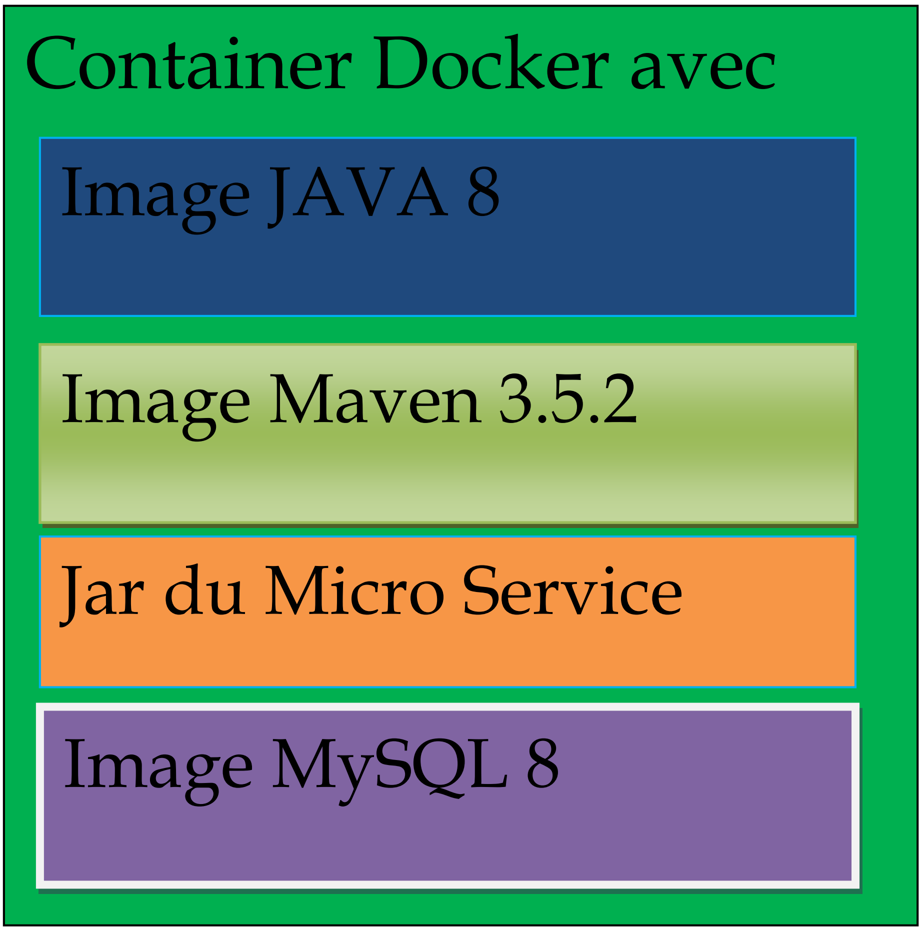
Verifions l'état de santé de notre Micro Service Spring Boot : http://localhost:8080/actuator/health/



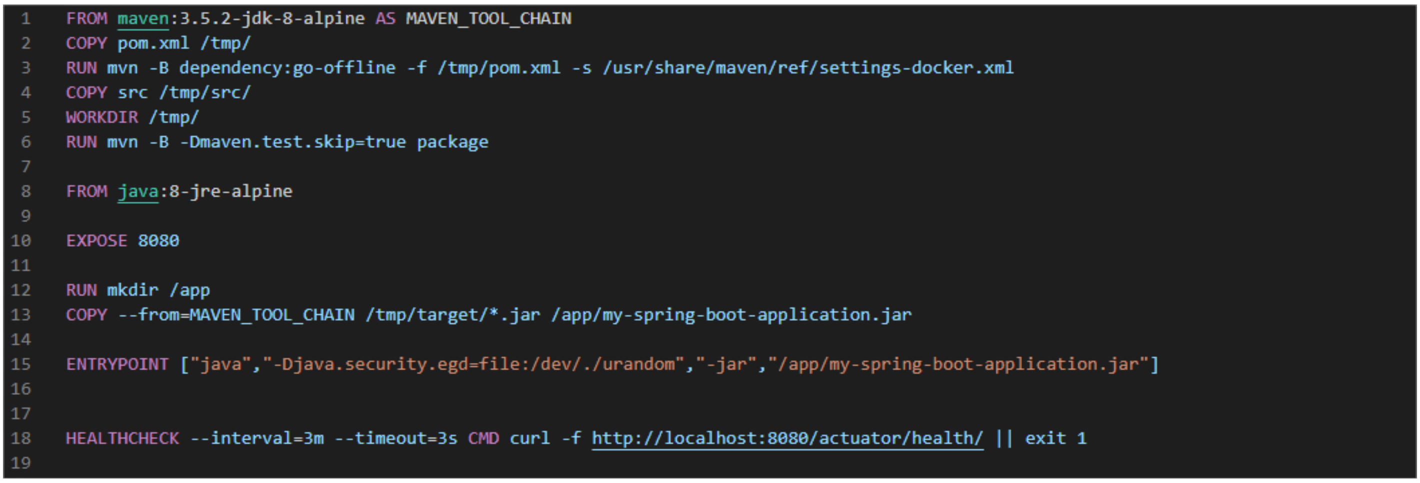
```
{"status":"UP","components":{"db":{"status":"UP","details":{"database":"MySQL","validationQuery":"isValid()"}},"diskSpace":{"status":"UP","details":
{"total":67371577344,"free":48623411200,"threshold":10485760,"exists":true}},"ping":{"status":"UP"}}}
```

4. *Version 4 : Image Docker contenant une image JDK 8 alpine (avec Maven 3.5.2 + le Jar du Micro Service Spring Boot) + image Docker MySQL 8*

L'architecture de l'application ressemblera à :



Le fichier **maven-first-app-spring-boot/Dockerfile** devient donc :



```
FROM maven:3.5.2-jdk-8-alpine AS MAVEN_TOOL_CHAIN
COPY pom.xml /tmp/
RUN mvn -B dependency:go-offline -f /tmp/pom.xml -s /usr/share/maven/ref/settings-docker.xml
COPY src /tmp/src/
WORKDIR /tmp/
RUN mvn -B -Dmaven.test.skip=true package

FROM java:8-jre-alpine

EXPOSE 8080

RUN mkdir /app
COPY --from=MAVEN_TOOL_CHAIN /tmp/target/*.jar /app/my-spring-boot-application.jar

ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app/my-spring-boot-application.jar"]


HEALTHCHECK --interval=3m --timeout=3s CMD curl -f http://localhost:8080/actuator/health/ || exit 1

```

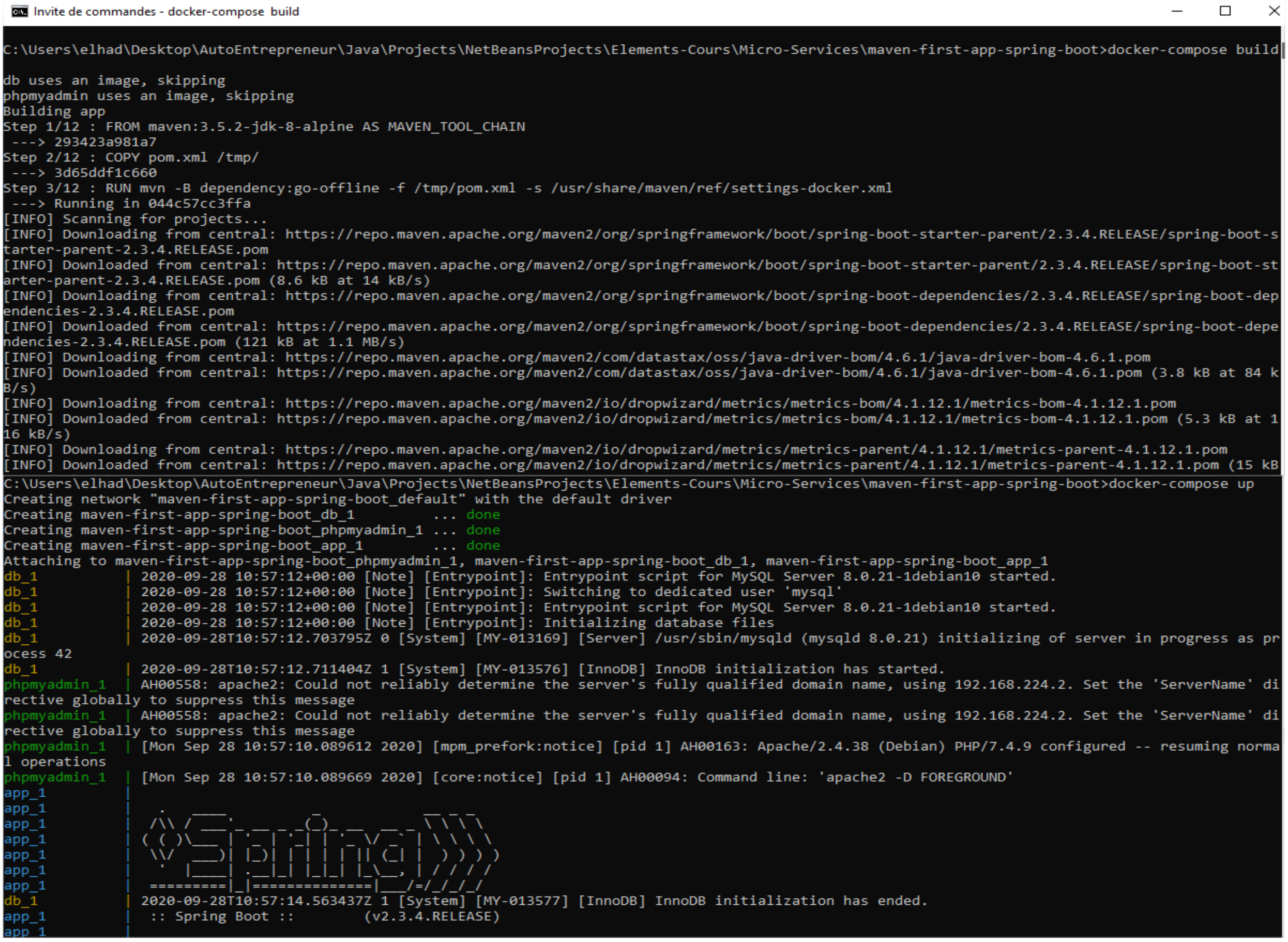
En version copiable :

```
FROM maven:3.5.2-jdk-8-alpine AS MAVEN_TOOL_CHAIN
COPY pom.xml /tmp/
RUN mvn -B dependency:go-offline -f /tmp/pom.xml -s /usr/share/maven/ref/settings-docker.xml
COPY src /tmp/src/
WORKDIR /tmp/
RUN mvn -B -Dmaven.test.skip=true package

FROM java:8-jre-alpine

EXPOSE 8080

RUN mkdir /app
COPY --from=MAVEN_TOOL_CHAIN /tmp/target/*.jar /app/my-spring-boot-application.jar

ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/app/my-spring-boot-application.jar"]


HEALTHCHECK --interval=3m --timeout=3s CMD curl -f http://localhost:8080/actuator/health/ || exit 1
```

Ligne 3 : Utilisation de Maven pour generer le projet jar du Micro Service à partir du **pom.xml**.

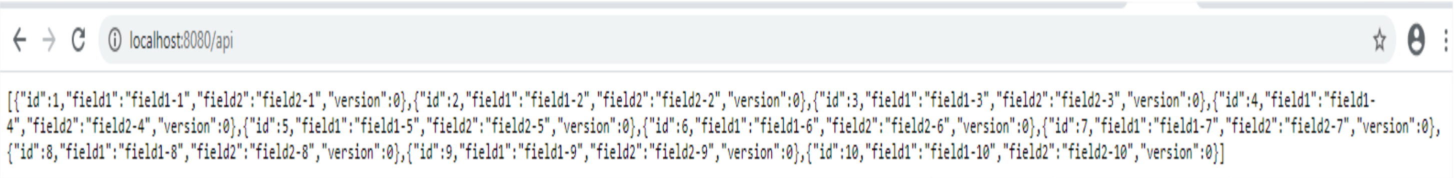
Lancer ensuite les commandes suivantes :

cd maven-first-app-spring-boot
docker-compose build
docker-compose up



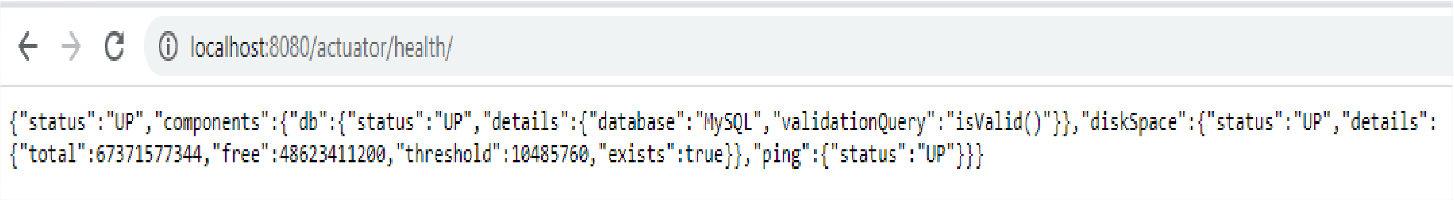
```
C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansProjects\Elements-Cours\Micro-Services\maven-first-app-spring-boot>docker-compose build

db uses an image, skipping
phpmyadmin uses an image, skipping
Building app
Step 1/12 : FROM maven:3.5.2-jdk-8-alpine AS MAVEN_TOOL_CHAIN
 ---> 293423a981a7
Step 2/12 : COPY pom.xml /tmp/
 ---> 3d65ddf1c660
Step 3/12 : RUN mvn -B dependency:go-offline -f /tmp/pom.xml -s /usr/share/maven/ref/settings-docker.xml
 ---> Running in 044c57cc3ffa
[INFO] Scanning for projects...
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-starter-parent/2.3.4.RELEASE/spring-boot-s
tarter-parent-2.3.4.RELEASE.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-starter-parent/2.3.4.RELEASE/spring-boot-st
arter-parent-2.3.4.RELEASE.pom (8.6 kB at 14 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-dependencies/2.3.4.RELEASE/spring-boot-dep
endencies-2.3.4.RELEASE.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-dependencies/2.3.4.RELEASE/spring-boot-depe
ndencies-2.3.4.RELEASE.pom (121 kB at 1.1 MB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/com/datastax/oss/java-driver-bom/4.6.1/java-driver-bom-4.6.1.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/datastax/oss/java-driver-bom/4.6.1/java-driver-bom-4.6.1.pom (3.8 kB at 84 k
B/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/io/dropwizard/metrics/metrics-bom/4.1.12.1/metrics-bom-4.1.12.1.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/io/dropwizard/metrics/metrics-bom/4.1.12.1/metrics-bom-4.1.12.1.pom (5.3 kB at 1
16 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/io/dropwizard/metrics/metrics-parent/4.1.12.1/metrics-parent-4.1.12.1.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/io/dropwizard/metrics/metrics-parent/4.1.12.1/metrics-parent-4.1.12.1.pom (15 kB
C:\Users\elhad\Desktop\AutoEntrepreneur\Java\Projects\NetBeansProjects\Elements-Cours\Micro-Services\maven-first-app-spring-boot>docker-compose up
Creating network "maven-first-app-spring-boot_default" with the default driver
Creating maven-first-app-spring-boot_db_1         ... done
Creating maven-first-app-spring-boot_phpmyadmin_1 ... done
Creating maven-first-app-spring-boot_app_1        ... done
Attaching to maven-first-app-spring-boot_phpmyadmin_1, maven-first-app-spring-boot_db_1, maven-first-app-spring-boot_app_1
db_1            | 2020-09-28 10:57:12+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1            | 2020-09-28 10:57:12+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
db_1            | 2020-09-28 10:57:12+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
db_1            | 2020-09-28 10:57:12+00:00 [Note] [Entrypoint]: Initializing database files
db_1            | 2020-09-28T10:57:12.703795Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.21) initializing of server in progress as pr
ocess 42
db_1            | 2020-09-28T10:57:12.711404Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
phpmyadmin_1    | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 192.168.224.2. Set the 'ServerName' di
rective globally to suppress this message
phpmyadmin_1    | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 192.168.224.2. Set the 'ServerName' di
rective globally to suppress this message
phpmyadmin_1    | [Mon Sep 28 10:57:10.089612 2020] [mpm_prefork:notice] [pid 1] AH00163: Apache/2.4.38 (Debian) PHP/7.4.9 configured -- resuming norma
l operations
phpmyadmin_1    | [Mon Sep 28 10:57:10.089669 2020] [core:notice] [pid 1] AH00094: Command line: 'apache2 -D FOREGROUND'
app_1           |
app_1           |   .   ____          _            __ _ _
app_1           |  /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
app_1           | ( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
app_1           |  \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
app_1           |   '  |____| .__|_| |_|_| |_\__, | / / / /
app_1           |  =========|_|==============|___/=/_/_/_/
db_1            | 2020-09-28T10:57:14.563437Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
app_1           |  :: Spring Boot ::        (v2.3.4.RELEASE)
app_1           |
```

On obtient sur http://localhost:8080/api :



```
[{"id":1,"field1":"field1-1","field2":"field2-1","version":0},{"id":2,"field1":"field1-2","field2":"field2-2","version":0},{"id":3,"field1":"field1-3","field2":"field2-3","version":0},{"id":4,"field1":"field1-
4","field2":"field2-4","version":0},{"id":5,"field1":"field1-5","field2":"field2-5","version":0},{"id":6,"field1":"field1-6","field2":"field2-6","version":0},{"id":7,"field1":"field1-7","field2":"field2-7","version":0},
{"id":8,"field1":"field1-8","field2":"field2-8","version":0},{"id":9,"field1":"field1-9","field2":"field2-9","version":0},{"id":10,"field1":"field1-10","field2":"field2-10","version":0}]
```

Verifions l'état de santé de notre Micro Service Spring Boot :
http://localhost:8080/actuator/health/



```
{"status":"UP","components":{"db":{"status":"UP","details":{"database":"MySQL","validationQuery":"isValid()"}},"diskSpace":{"status":"UP","details":
{"total":67371577344,"free":48623411200,"threshold":10485760,"exists":true}},"ping":{"status":"UP"}}}
```