PRICHICI MARIA SANZIANA

Deploying REST API + PostgreSQL database with Docker

The goal of this tutoria lis to present who I did the following steps:

1. Dockerize Spring Boot Application (my secured REST API)

Giving the fact that I have used Maven, first 2 steps were to clean and install the project:

```
D:\Proiect SOA\Proiect_SOA>mvn clean
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for com.example:demo:jar:0.0.1-SNAPSHOT
[WARNING] 'dependencies.dependency.scope' for org.projectlombok:lombok:jar must be one of [provided, compile, runtime, test, system] but is 'annotationProcessor'. @ line 57
, column 11
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[WARNING]
[INFO]
[INFO] ------------------------< com.example:demo >------------------------
[INFO] Building demo 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- clean:3.3.2:clean (default-clean) @ demo ---
[INFO] Deleting D:\Proiect SOA\Proiect_SOA\target
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.691 s
[INFO] Finished at: 2024-02-07T15:51:19+02:00
[INFO] ------------------------------------------------------------------------
```

```
D:\Proiect SOA\Proiect_SOA>mvn install
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for com.example:demo:jar:0.0.1-SNAPSHOT
[WARNING] 'dependencies.dependency.scope' for org.projectlombok:lombok:jar must be one of [provided, compile, runtime, test, system] but is 'annotationProcessor'. @ line 57
, column 11
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[WARNING]
[INFO]
[INFO] ------------------------< com.example:demo >------------------------
[INFO] Building demo 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[WARNING] Parameter 'annotationProcessorPaths' is unknown for plugin 'spring-boot-maven-plugin:3.2.2:repackage (repackage)'
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ demo ---
[INFO] Copying 1 resource from src\main\resources to target\classes
[INFO] Copying 0 resource from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.11.0:compile (default-compile) @ demo ---
[INFO] Changes detected - recompiling the module! :source
[INFO] Compiling 12 source files with javac [debug release 17] to target\classes
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ demo ---
[INFO] skip non existing resourceDirectory D:\Proiect SOA\Proiect_SOA\src\test\resources
```

Since we'll use a container of postgreSQL and not a local stored database, it is usefull to add "-DskipTests=true" in the command line. This will prevent our build to fail as the test package cannot connect to the database yet.

Next, I created a docker-composed.yml file in order to be able to deploy the app and database at the same time:

```yaml
version: '3.2'
services:
    app:
        container_name: demoapi
        image: demoapi
        build: ./
        ports:
            - "8000:18080"
        depends_on:
            - postgresqldb
    postgresqldb:
        restart: always
        container_name: proiect_soa-postgresqldb-1
        image: postgres:latest
        ports:
            - "5432:5432"
        environment:
            - POSTGRES_PASSWORD=postgre
            - POSTGRES_USER=postgres
            - POSTGRES_DB=postgres
```

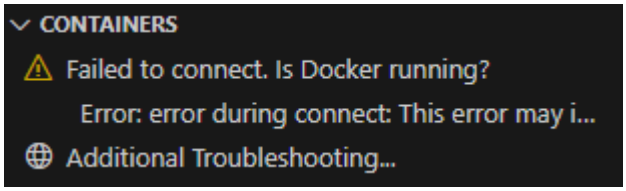
The link between the app and database it is made with the „depends_on” variable. We can see that the docker container will run on the 8000 port and it will be linked with the 18080 port (that I set for my REST API in the application.properties file).

```
#the port occupied
server.port=18080
```

In order to use the postgre container and not the localhost address, in the same application.properties file, I have modifies the connection to the database: the commented line was for when I created the app, and now I replaced the IP adress for localhost with the name of the dockerized postgre container.

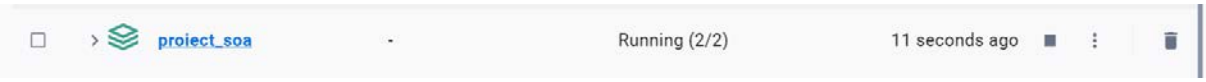
```
#spring.datasource.url = jdbc:postgresql://localhost:5432/postgres
spring.datasource.url = jdbc:postgresql://postgresqldb:5432/postgres
```

NOTE: Verify that your docker service is running in order not to have this error message:



Now, I could have built the two separately, for example:

But because I have created the composed file, I can run the command: docker-compose up, that will run both of the containers in the docker-compose file under the name proiect_soa container:



2. Nginx configurations

In the nginx.conf, I have added the following:

```
server {
    listen       18080;
    server_name  localhost;

    #charset koi8-r;

    #access_log  logs/host.access.log  main;

    location / {
    proxy_pass http://localhost:8000;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    }
```

Because my microfrontends are trying to fetch from the localhost:18080 port, I need to redirect them to the dockarized container where I now run my REST API.