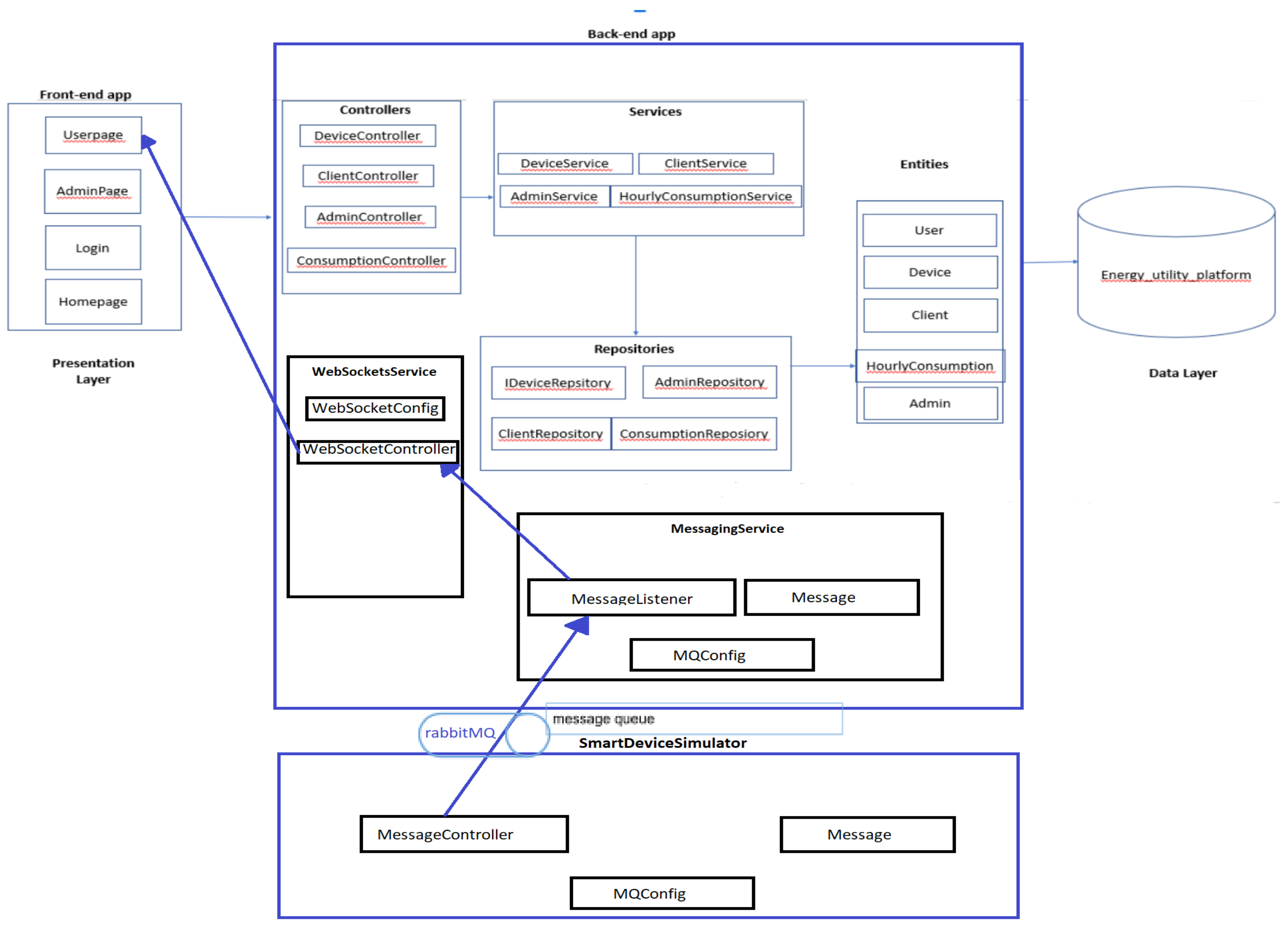
Online Energy Utility Platform Assignment 2

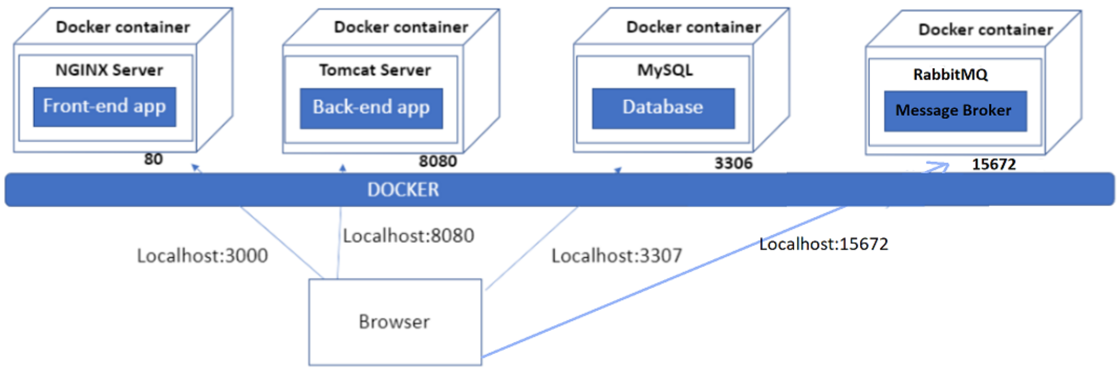
Student: Miu Daria
Group: 30442
Profesor: Cristina Pop

System Architecture Diagram



Deployment on Docker

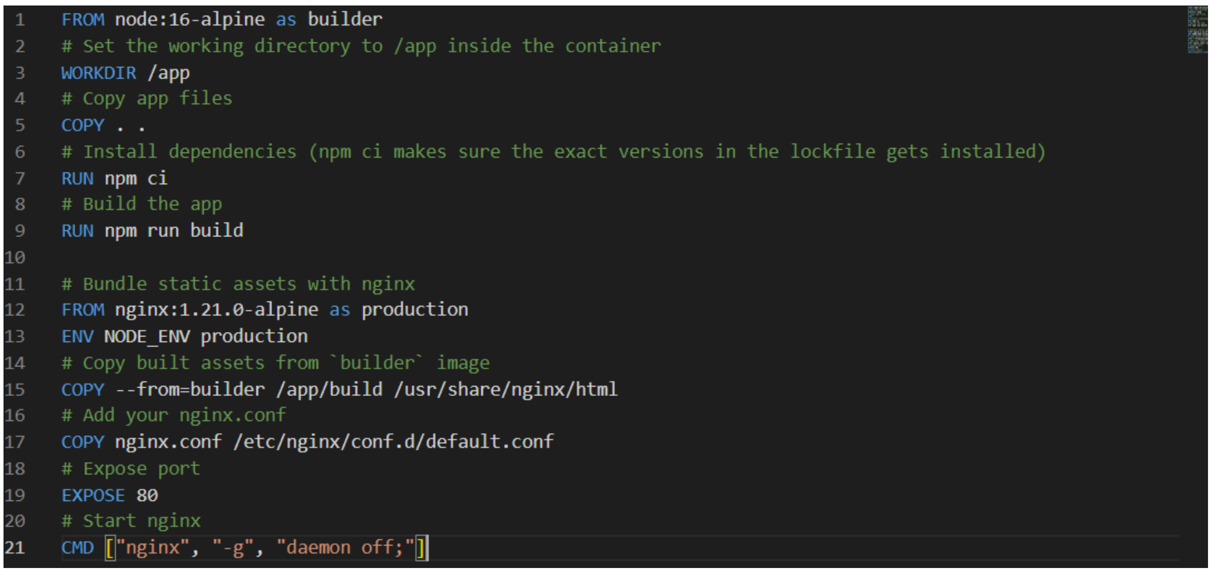
All the components of the application are deployed locally on docker containers. Docker is a container-based technology where containers are running as processes in the user space of the operating system. The React frontend part is deployed in one container and the Mysql database and the Spring Boot Java application are together in another container.
Below is a generalized diagram of the docker architecture of the deployment.



Deploying the frontend React application

The first step I did was creating the Dockerfile and the nginx.conf file in the root folder of my react project. After that, I wrote the contents of the files as below:

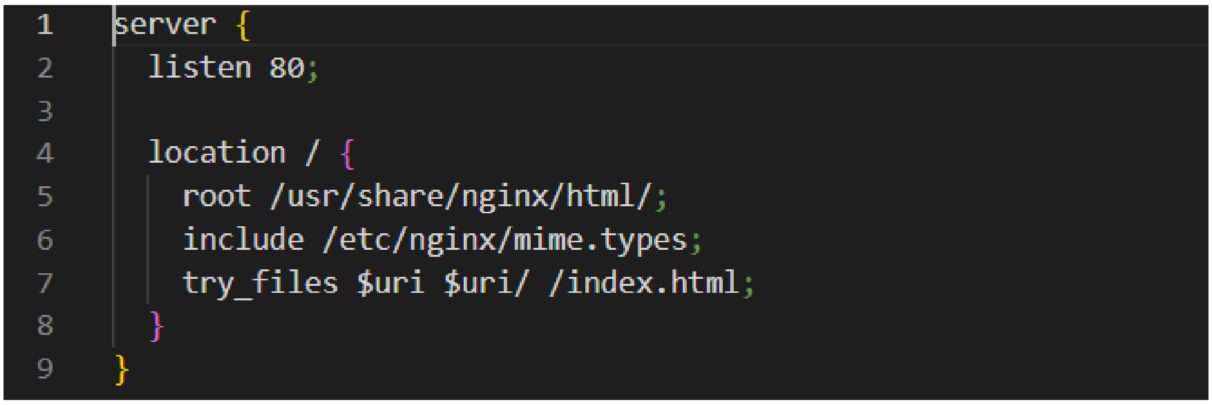
Dokerfile



```
FROM node:16-alpine as builder
# Set the working directory to /app inside the container
WORKDIR /app
# Copy app files
COPY . .
# Install dependencies (npm ci makes sure the exact versions in the lockfile gets installed)
RUN npm ci
# Build the app
RUN npm run build

# Bundle static assets with nginx
FROM nginx:1.21.0-alpine as production
ENV NODE_ENV production
# Copy built assets from `builder` image
COPY --from=builder /app/build /usr/share/nginx/html
# Add your nginx.conf
COPY nginx.conf /etc/nginx/conf.d/default.conf
# Expose port
EXPOSE 80
# Start nginx
CMD ["nginx", "-g", "daemon off;"]
```

nginx.conf



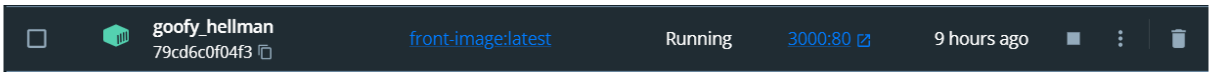
```
server {
  listen 80;

  location / {
    root /usr/share/nginx/html/;
    include /etc/nginx/mime.types;
    try_files $uri $uri/ /index.html;
  }
}
```

After the files were created, I run the following commands:

- docker build . -t image-front
- docker run -p 3000:80 -d image-front

And the container from the image started running as seen below

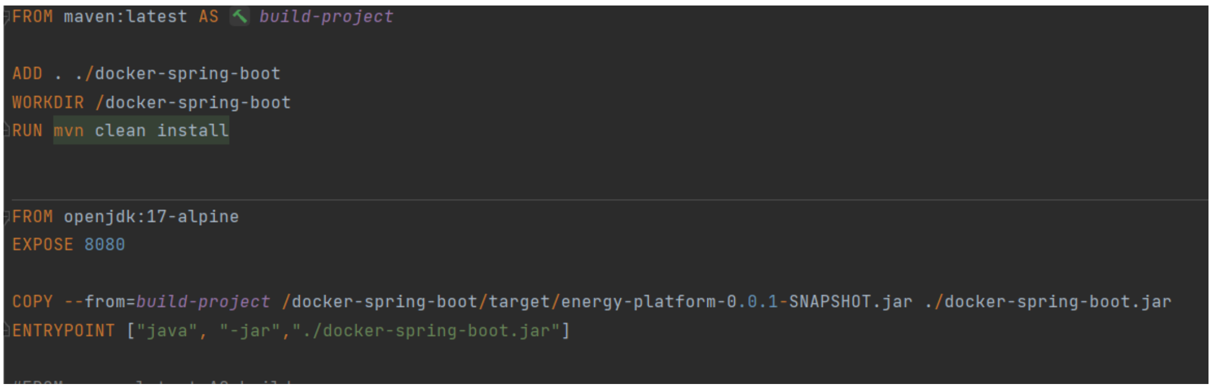


```
goofy_hellman    front-image:latest    Running    3000:80    9 hours ago
79cd6c0f04f3
```

Deploying the database, rabbitMQ and the Spring Boot app

The first step I did was to create the Dockerfile and the docker-compose.yml file in the root folder of my Spring Boot application. The contents of the files can be seen below.

Dockerfile



```
FROM maven:latest AS build-project

ADD . ./docker-spring-boot
WORKDIR /docker-spring-boot
RUN mvn clean install


FROM openjdk:17-alpine
EXPOSE 8080

COPY --from=build-project /docker-spring-boot/target/energy-platform-0.0.1-SNAPSHOT.jar ./docker-spring-boot.jar
ENTRYPOINT ["java", "-jar","./docker-spring-boot.jar"]
```

docker-compose.yml

```
version: '3'
services:
 database:
   image: mysql
   restart: always
   environment:
     - MYSQL_DATABASE=energy_utility_platform
     - MYSQL_ROOT_PASSWORD=cara12345
     - MYSQL_HOST_AUTH_METHOD=trust
   ports:
     - "3307:3306"
   container_name: database-a2
 rabbitmq:
   image: rabbitmq:management
   restart: always
   container_name: rabbitmq
   ports:
     - "5672:5672"
     - "15672:15672"
 backend:
   image: backend-image-a2
   restart: always
   ports:
     - "8080:8080"
   environment:
     - SPRING_RABBITMQ_HOST=rabbitmq
     - DB_IP=demo-db
     - RABBIT_IP=demo-rabbit
     - DB_PORT=3307
     - DB_USER=root
     - DB_PASSWORD=cara12345
     - DB_DBNAME=energy_utility_platform
   depends_on:
     - rabbitmq
     - database
```

Then I run the maven component to generate the jar of the application(the jar that is included in the Dockerfile). After I changed the data source connection in the app properties file from the local one to the following:

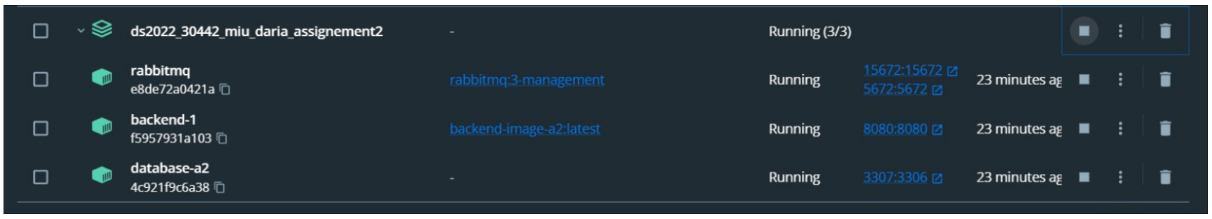
```
spring.datasource.url=jdbc:mysql://host.docker.internal:3306/energy_utility_platform
```

The next step was running the following commands in the terminal:

-docker build . -t backend-image

-docker-compose up -d

And finally, the database, rabbitmq and the backend image can be seen running in the docker container as below:



ds2022_30442_miu_daria_assignement2	-	Running (3/3)		
rabbitmq e8de72a0421a	rabbitmq:3-management	Running	15672:15672 5672:5672	23 minutes ag
backend-1 f5957931a103	backend-image-a2:latest	Running	8080:8080	23 minutes ag
database-a2 4c921f9c6a38	-	Running	3307:3306	23 minutes ag