Online Energy

Utility Platform Assignment 3

Student: Miu Daria

Group: 30442

Profesor: Cristina Pop

[**System Architecture Diagram**](#_v0fgnn5pk883) **3**

[**Deployment on Docker**](#_ixzjvjw6obxk) **4**

[Deploying the frontend React application](#_x7lk5s6qoo8j) 4

[Deploying the database, rabbitMQ and the Spring Boot app](#_b1sbis1w4vnb) 5

[Deploying Envoy and GRPC server](#_p3yglfniuqgy) 7

# 

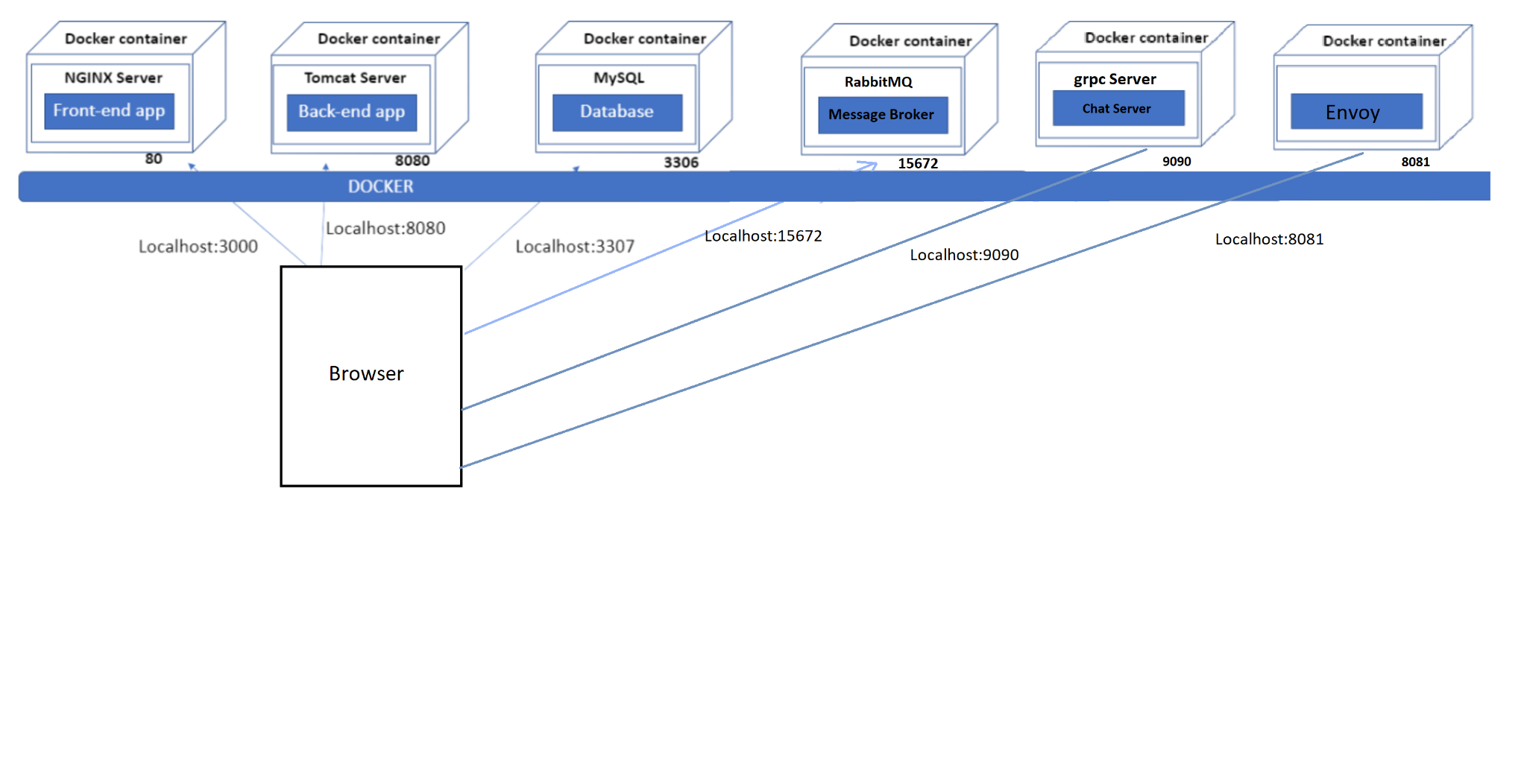
# System Architecture Diagram



# 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.

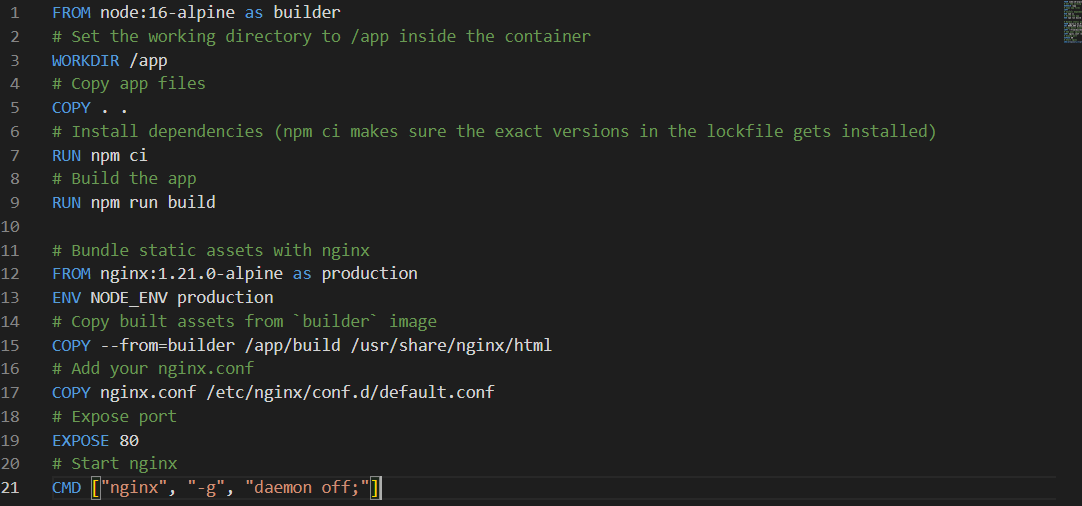
Below is a generalized diagram of the docker architecture of the deployment.



## 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



nginx.conf

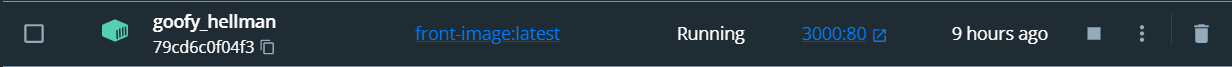


After the files were created, I run the following commands:

- docker build . -t image-front

- docker run -p 3000:80 -d image-front

And the container from the image started running as seen below



## 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



docker-compose.yml

version: "3.4"

services:

api:

image: "containerregistrydariamiu30442.azurecr.io/backend-image-a2:latest"

domainname: "backend-a2"

ports:

- 8080:8080

environment:

SPRING\_RABBITMQ\_HOST: rabbitmq

DB\_IP: demo-db

RABBIT\_IP: demo-rabbit

DB\_PORT: 3306

DB\_USER: root

DB\_PASSWORD: cara12345

DB\_DBNAME: energy\_utility\_platform

deploy:

resources:

reservations:

cpus: '1'

memory: 2G

db:

image: "containerregistrydariamiu30442.azurecr.io/db:latest"

environment:

MYSQL\_DATABASE: energy\_utility\_platform

MYSQL\_ROOT\_PASSWORD: cara12345

MYSQL\_HOST\_AUTH\_METHOD: trust

domainname: "backend-a2"

ports:

- 3306:3306

deploy:

resources:

reservations:

cpus: '1'

memory: 2G

rabbitmq:

image: "containerregistrydariamiu30442.azurecr.io/rabbitmq:latest"

domainname: "backend-a2"

ports:

- 15672:15672

- 5672:5672

deploy:

resources:

reservations:

cpus: '1'

memory: 2G

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:

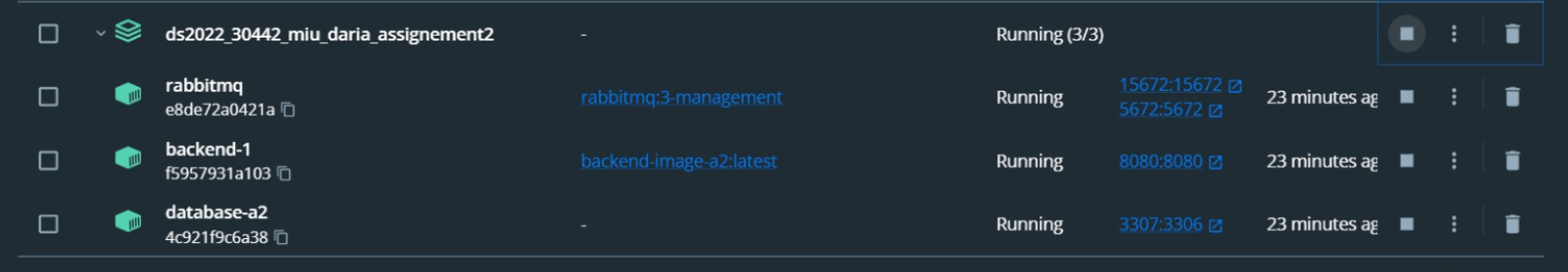


The next step was running the following commands in the terminal:

-docker build . -t backend-image

-docker-compose up -d

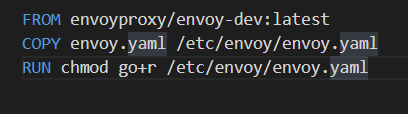
And finally, the database, rabbitmq and the backend image can be seen running in the docker container as below:



## Deploying Envoy and GRPC server

First I created the Dockerfile and the envoy.yaml files:

Dockerfile



envoy.yaml

To build the image:

- docker build -t grpc-web-react .

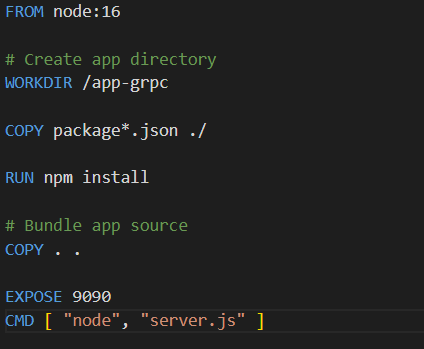
To run the docker:

-docker run -d --name grpc-web-react -p 8080:8080 -p 9901:9901 grpc-web-react

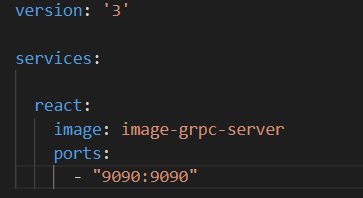


Then for the grpc server part the following files were used:

Dockerfile



docker-compose.yml



To build the image:

- docker build -t image-grpc-server.

To run the docker:

-docker compose up -d

