**Energy Management System**

-Distributed Systems-

This application is an Energy Management System that consists of a frontend and two microservices designed to manage users and their associated smart energy metering devices. The system can be accessed by two types of users after a login process: administrator and clients. The administrator can perform different operations on user accounts such as adding, deleting, editing and viewing. The users are defined by ID, name, role: admin/client, and the smart energy metering devices are defined by ID, description, address, maximum hourly energy consumption, and on the mapping of users to devices (each user can own one or more smart devices in different locations).

Here is a graphical diagram of the databases:

**The database for the users:**

A screenshot of a computer screen

Description automatically generated

**The database for the devices:**

**A screenshot of a computer

Description automatically generated**

The databases are created in Postgres, the two microservices for the users management and devices management are created in spring framework and the frontend part in react. To run the spring microservices, we can access it from IntelliJ Idea for example and run it. The react project can be run with the npm start command.

A screenshot of a login box

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedThe administrator can manage users, devices and the user devices. A user can have one or more devices. So the administrator can see all users who own a device, and which device. The user can see all devices, and theirs as well.

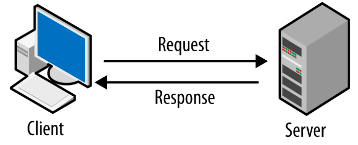
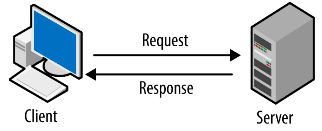
Docker

I built my images using:

• docker build -t your\_image\_name .

Started my images:

• docker-compose up -d

A diagram of a diagram

Description automatically generated

API CALLS

A diagram of a ship

Description automatically generated

FRONTEND : 3000:3000

:

DB 2 : 5432:5432

:

DB 1 : 5432:5432

:

BACKEND2 : 9090:9090

:

BACKEND1 : 8080:8080

: