Energy Management System

Tema 3

Student: Botis George

1. **Introduction**

This application aims to help customers and administrators communicate with each other. The program involves developing a chat system to provide support to the customers of the energy platform in case they have questions related to their energy consumption. Furthermore, this application aims to increase the security of the application by using a secret shared key throughout the microservices.

1. **Functionality**

Upon launching the application, the user will see the login page. If the user's role is that of a client, they will only see a page displaying their devices, the possibility to see the graph of electricity consumption and a button with the capacity of starting a chat with the administrator. Alternatively, if the user is an administrator, they will have functionalities for creating, deleting, updating, and viewing all customers, devices, and measurements. Upon clicking the button associated with these functions, the user will be redirected to the page dedicated to the chosen functionality. Also, the administrator can enter chats with users that text him.

1. **Technology used**

My application consists of two parts: backend and frontend.

For the backend, I used Java Spring Boot in IntelliJ IDEA. This part was not difficult for me as I have previously worked with Java during my university studies. For the frontend, I used Visual Studio Code to create the application. In this case, I opted for React and the login functionality was implemented accordingly. For the administrator's functionalities, I used React as it seemed easier to use.

As for the database, I used a relational one: MySQL.

To implement the requirements of this project, I utilized web sockets. The WebSocket protocol enables interaction between a web browser (or another client application) and a web server with lower overhead than half-duplex alternatives, such as HTTP polling. This facilitates real-time data transfer to and from the server.

1. **Security**

For security, I utilized shared secret key appraoch. If the user is not logged in, they do not have permission to access other pages of the application. Additionally, a user with the role of a client cannot access pages with administrator functionalities. The application verifies if the ID of the user attempting to access the application belongs to an administrator. If not, access is not granted to the client. A secret key is shared across the microservice which is used to encrypt and decrypt the user token at each moment a service is called or used.

* 1. **UML Diagram**

A diagram of a person with text

Description automatically generated

* 1. **Database Diagram**A screenshot of a computer

     Description automatically generated
  2. **Class Diagram**

A screenshot of a computer program

Description automatically generated

* 1. **Deployment Diagram**

1. A diagram of a computer program

   Description automatically generated