Web sockets and security

Assignment 3

Distributed Systems

**Student:** Jimborean Oana-Elena

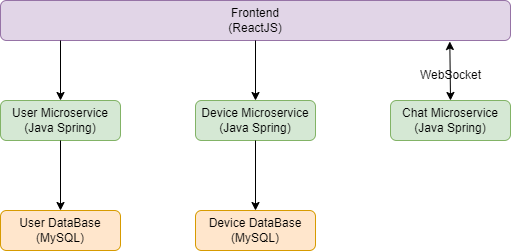
**Group:** 30443

1. **Introduction:**

I had to develop a chat microservice and an authorization component for the Energy Management System. The authorization component provides secured access of users to systems’ microservices. The chat microservice allows communication between the users and the administrator of the system, allowing them to ask questions and receive answers. The final project needs to have an authorization component: JWT based authorization. One of the services is chosen as authorization server. This service generates access tokens to the client application. The tokens will be used to access other microservices.

1. **Architecture**

The system architecture consists of a Chat Microservice (Java Spring), which uses web socket technology in order for users to communicate with the admins. The front-end application is implemented in ReactJS, being implemented in continuation of the first project. The message is sent asynchronously to the administrator, that receives the message together with the user identifier, being able to start a chat with the user. Messages can be sent back and forth between the user and the administrator during a chat session. The administrator can chat with multiple users at once. A notification is displayed for the user when the other administrator reads the message and vice versa. A notification is displayed for the user (e.g., typing) while the administrator from the other end of communication types of its message and vice versa.



1. **Frontend Components**

The frontend is developed using React, a popular JavaScript-based framework. It communicates with the backend to retrieve or send web page data. After logging in, users are redirected to pages corresponding to their roles.

After logging in, the users can click on the “Chat” button and can send messages to the admins. Every time a new user logs in, it appears to the admins on the chat page, and they can text to the new entered user.

1. **Backend Microservices**

User Management Microservice**:** responsible for managing user-related operations. It handles user registration, authentication, and CRUD operations for user accounts. Users are defined by attributes like ID, name, and role (admin or user). This microservice communicates with the User Database to store and retrieve user data.

Device Management Microservice**:** dedicated to managing smart energy devices. It handles CRUD operations on devices, such as add, get, update, and delete. Each device is defined by attributes like ID, description, address, and maximum hourly energy consumption. The Device Management Microservice communicates with the Device Database to store and retrieve device-related data.

Chat Microservice: supports message send and receive primitives. Administrator can chat with multiple clients at once. Additionally, a notification is displayed for the user when the other user reads the message. The communication between users and admins is done using web socket technology.

1. **Deployment**

To deploy the Chat Microservice, I used Docker, which provides the network and container virtualization functionality, allowing the containers to interact with each other in a secure environment and allowing certain ports and endpoints to pass through to the clients.

Backend deployment: the microservice was containerized, using Docker images which encapsulate the applications for consistent deployment.

Frontend deployment:the React frontend was containerized using Docker and a dockerfile was created to specify its environment and dependencies.

