Request-Reply Communication

Assignment 1

Distributed Systems

**Student:** Jimborean Oana-Elena

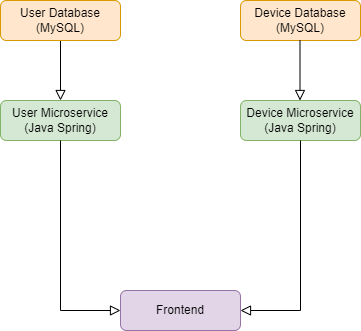
**Group:** 30443

1. **Introduction:**

I had to develop 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 (manager), and clients. The administrator can perform CRUD (Create-Read-Update-Delete) operations on user accounts (defined by ID, name, role: admin/client), smart energy metering devices (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).

1. **Architecture**

The system architecture comprises two Java Spring microservices for user and device management, each with its dedicated MySQL database. The frontend, which serves as the user interface is designed in React. When the app is started, the users can Login and are taken to their users’ page. Administrators can perform CRUD operations on user accounts and devices, while clients can view their devices. The system enforces role-based access control and communicates between the frontend and microservices for (the user with the role of user can’t perform CRUD operations).



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 as admin, you can see all the devices, delete them and edit them. The admin can also delete/edit a user, and if a user is deleted, his device is also removed from the list.

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.

1. **Databases**

**User Database:** stores user-related information, including userId, username, email, password, status and role (admin or client). It is responsible for maintaining user accounts.

**Device Database:** stores data related to smart energy metering devices. This includes attributes like deviceId, description, address, maximum hourly energy consumption, and user IDs for mapping devices to users.

1. **Deployment**

To deploy the Energy Management System, 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 microservices were containerized, alongside its database, 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.

