**Energy Consumption Application**

Presecan Alexandru

Group 30442

1. **General description**

The energy consumption management application is designed to keep track of user devices and their energy consumption. The energy consumption is measured each hour for every active device and will be stored in the database. The users will be able to visualize for each of their devices, the consumption during a specified day. The administrators will be able to see all the users and all the devices and will have the right to perform CRUD operations on them. A user-device mapping feature is also available for admins in order to easily assign devices to users. The values for energy consumption are simulated by a message producer and sent to the backend through RabbitMQ. Websockets are used to send alerts to clients if their devices exceeded the maximum hourly energy consumption. A chat system is also available for clients to communicate with the admin. The chat system uses a grpc service to allow fast data transmission and a bidirectional stream for messages.

1. **Architecture**

The application is composed of 6 interconnected components: database, backend, frontend, message producer, message broker, and message consumer.

The backend and the grpc service are using https and certificates for a secure connection. The backend and the websockets are secured with a JWT token. A bad request will be received if the user tries to call an endpoint and does not have the required permission (is not logged in or tries to use an admin endpoint as a client). The frontend is also secured with angular guards so a client will not be able to access an admin url.

* **Database**

The database used is PostgresSQL and contains 3 relational tables: user, device, and energy consumption.

* **Backend**

The backend is created using Asp.NET and uses a layered architecture having the following components: controllers (presentation layer), services (business logic layer), data accessors (data access layer), and model classes.

For generating the database and creating migrations from the class models, I used the Entity Framework.

For security, a JWT library is used. The user will receive a JWT token when authenticating, which will be required in order to use any other backend endpoints. The endpoints will behave differently, depending on the token used, for example, creating an admin account will only be available for users with admin privileges.

Websockets are used to send alerts to clients if their devices exceeded the maximum hourly energy consumption. For this, the SignalR library is used.

**Diagram

Description automatically generated**

**Backend Layered Architecture Diagram**

* **Frontend**

The frontend is created using Angular and uses the common component conventions used by the framework. Other components used were services and models in order to communicate with the backend.

For creating the graphs displaying the energy consumption of a device, the devextreme library is used.

For storing the current user data and token in cookies, the ngx-cookie-service is used.

For listening to websocket messages, the signalR library is used.

**Diagram

Description automatically generated with medium confidence**

**Application Architecture Diagram**

* **Message Producer**

The message producer is a C# console application that reads a csv file with energy consumption values, creates json messages for specific devices, and sends them to the message broker.

* **Message Broker**

The message broker is a RabbitMQ instance that acts as a broker between the message producer and the message consumer, keeping track of messages and queues.

* **Message Consumer**

The message consumer is integrated with the backend, so it will listen for messages from the broker and add the measurements to the database

* **Grpc Service**

The grpc service uses .Net Grpc Web services and allows the clients to communicate with the admin. The service uses bidirectional streams for sending messages and other notifications (another user is typing, and another user has seen the messages). The clients can only communicate with the admin, and the admin can communicate with all the connected clients.

1. **Database design**

The database contains 4 tables: one used for Asp.NET Entity Framework migrations, and 3 relational tables which are used for the Energy Consumption Application. These tables are user, device, and energy consumption. Between the user and the device, we have a one-to-many relationship, and, between the device and the energy consumption, we also have a one-to-many relationship.

A picture containing box and whisker chart

Description automatically generated

**Database Diagram**

1. **UML deployment diagram**

**Diagram

Description automatically generated**Each of the 4 components runs on a different container under the same docker repository.

**Deployment Diagram**

**Graphical user interface, application

Description automatically generated**

**The containers running in docker**