UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

ENERGY UTILITY PLATFORM

DISTRIBUTED SYSTEMS

Assignment 3

Cusco Ana-Maria

2022

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Contents

1. Requirements

Develop a chat system to offer support for the clients of the energy platform if they have questions related with their energy consumption. The chat system should allow communication between the clients and the administrator of the system.

1.1. Functional requirements:

➢ The client application displays a chat box where clients can type messages.

➢ The message is sent asynchronously to the administrator, that receives the message together with the client identifier, being able to start a chat with the client.

➢ Messages can be sent back and forth between the client and the administrator during chat session.

➢ The administrator can chat with multiple clients at once.

➢ A notification is displayed for the user when the other user reads the message.

➢ A notification is displayed for the user (e.g., typing) while the user from the other end of communication types its message.

1.2. Implementation technologies:

➢Choose one of the following technologies: any RPC framework supporting data streaming (for instance: gRPC)

2. Conceptual architecture of the online platform

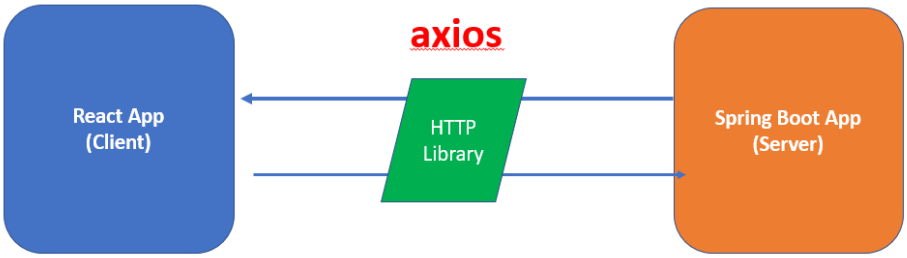
Technology Stack

- React

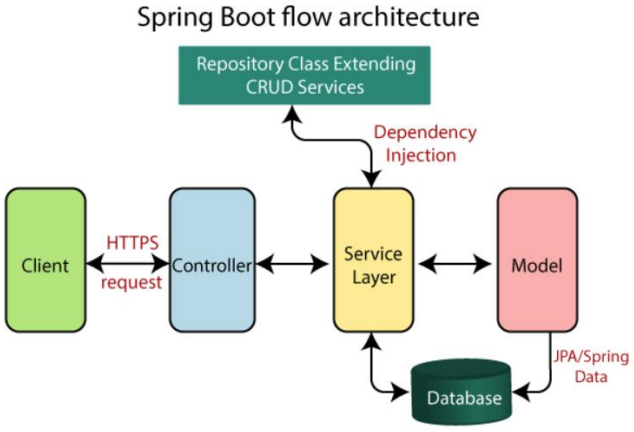
- Spring Boot
- Docker
- RabbitMQ
- WebSocket
- gRPC

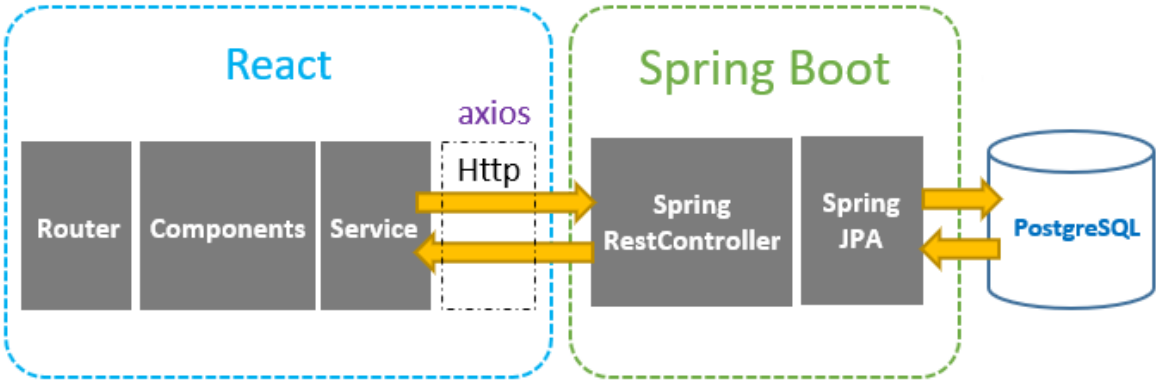
The application architecture follows the client-server model with React Client and Spring Boot for REST APIs

1. **sprint boot-backend (server, rabbit-mq consumer)** – To develop and expose REST API, cosume queue messages
2. **react-frontend (client)** – Consume REST API.
3. **spring-boot-producer** (rabbit-mq producer) – To produce messages in the queue



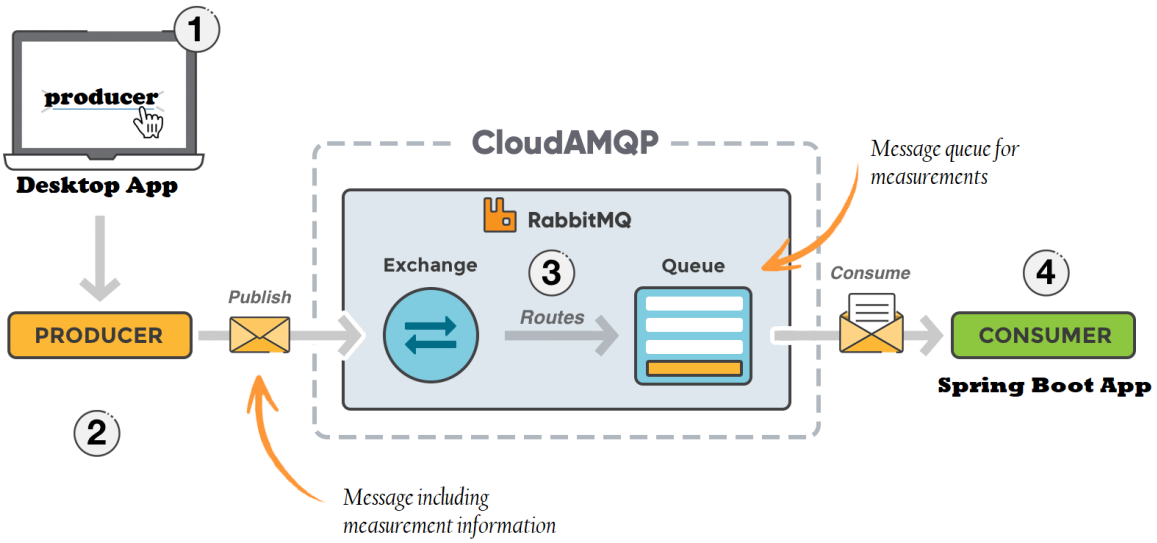
Backend part follows a layered architecture in which each layer with the layer directly below or above (hierarchical structure) it.



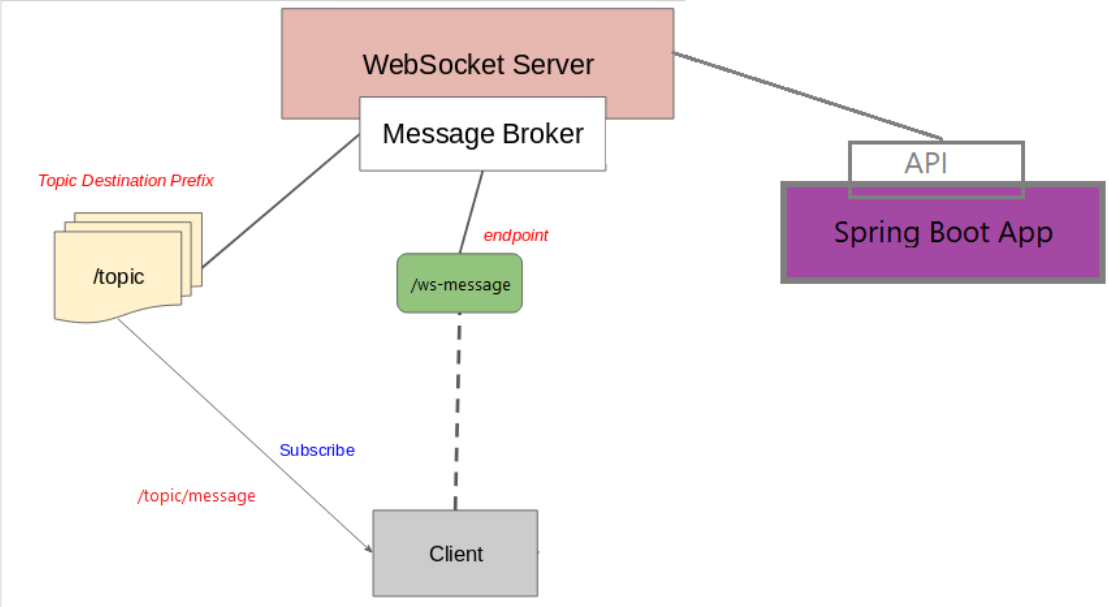


– Spring Boot exports REST Apis using Spring Web MVC & interacts with PostgreSQL Database using Spring JPA.

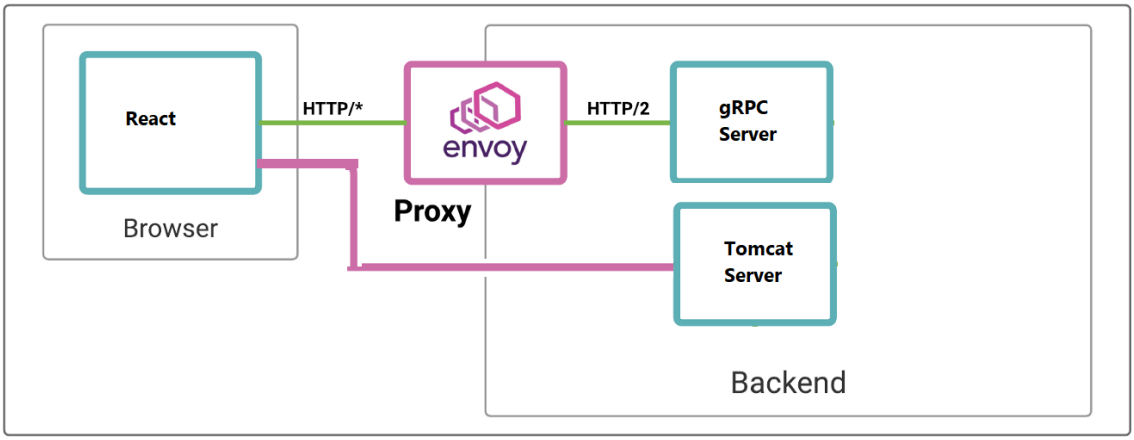
– React Client sends HTTP Requests and retrieve HTTP Responses using axios, shows data on the components. I also use React Router for navigating to pages.



The application uses CloudAMQP in order to send measurements to the queue and the Spring Boot Application will consume those messages and will alert the clients if the device consumption has exceeded the maximum consumption limit.



For displaying real time notification the application uses WebSocket. The client is notified every time a device consumtion exceeds the max limit.



This assignment was developed based on the previous 2 assignments with a new functionality integrated – the gRPC Chat Application.

This application includes the gRPC framework which define the chat service in a .proto file and generates the server code using protocol buffer compiler.

The client application uses Envoy proxy for converting HTTP1 requests to HTTP2.

Multi user communication is implemented and also a typing message is displayed whenever a user wants to send a message to another.

4. UML Deployment diagram

This applcation was deployed locally and include 5 containers:

Frontend – React client

Backend – REST Spring Server

Envoy – proxy for gRPC

Postgres- database

gRPC – gRPC server

See docker-compose.yml for more info.