Chat system for client support

Zaharia Tudorita

Faculty of Computer Science, Technical University of Cluj-Napoca

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.

- 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. Conceptual Architecture

In the first figure, we represent the package gRPC with classes generated by the proto compiler. The files are used by gRPC service having the annotation *@GrpcService*. Considering that the maven dependency net.devh creates the server automatically, there is no need for a class or package to take care of it.

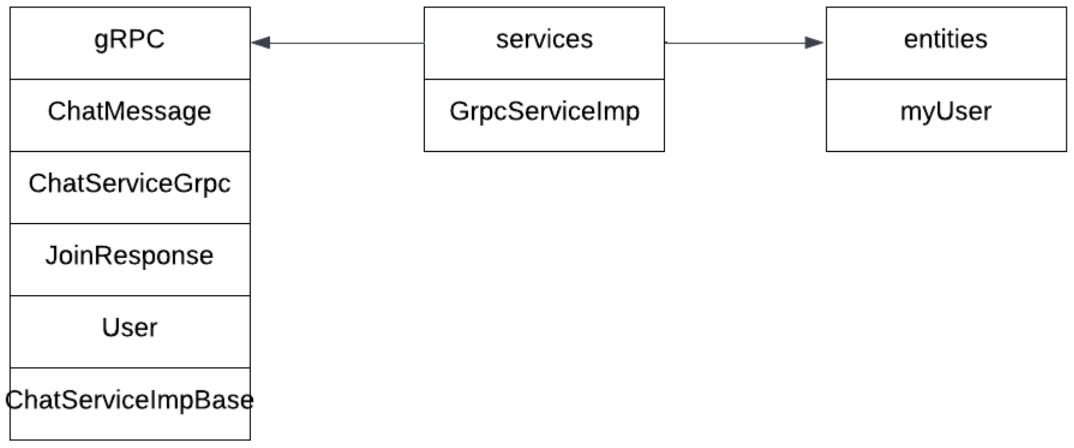


Figure 1. Backend architecture

2. Deployment Diagram

The application was deployed using docker containers. We created an image for database, frontend, backend, RabbitMQ, and Envoy Proxy. Given that gRPC communicates over HTTP/2 the nginx server will send the procedure firstly to the Envoy Proxy over HTTP/1, consequently the proxy transmits the remote call to the IP address of the internal docker container of the tomcat server with gRPC port 9090. Application was locally deployed, working on localhost with port map 3001 for the client side.

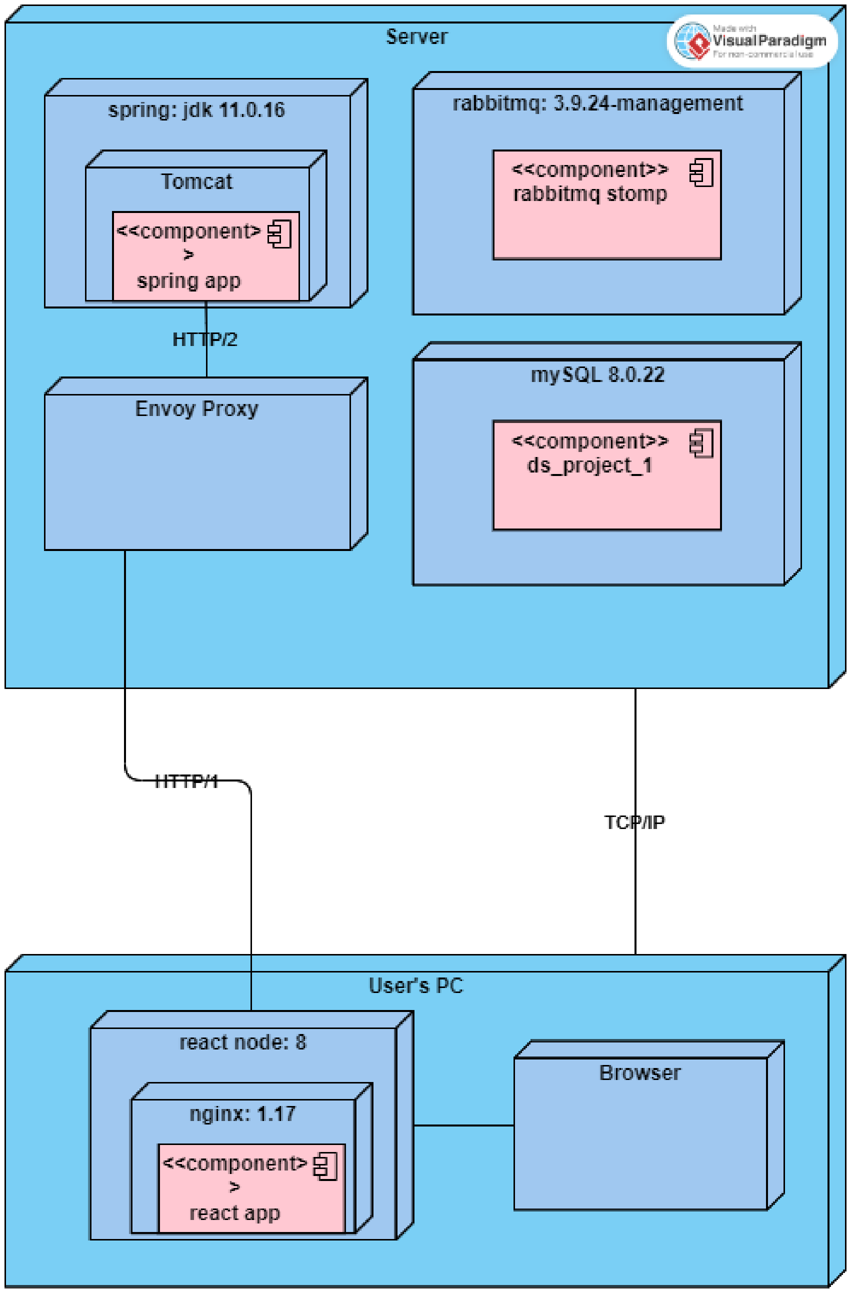
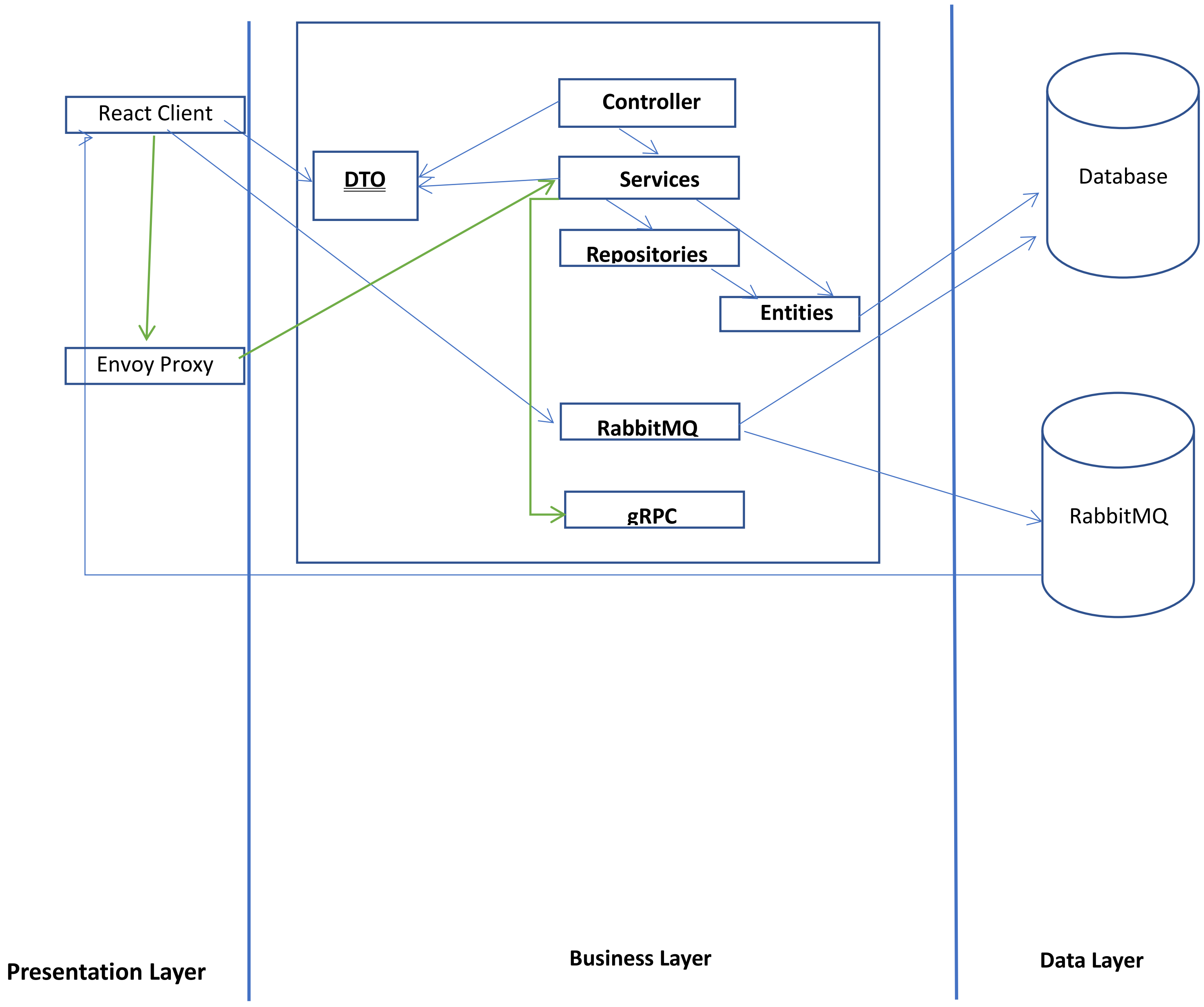


Figure 2. Deployment UML

3. Conceptual Diagram

As mentioned before, the client interacts firstly with Envoy Proxy, then the service is called from the backend server in the *service* package. The *service* will make use of **gRPC** package with interfaces generated by the proto compiler. Frontend and backend communicate using remote procedure call to create an asynchronous chat.


React Client
Envoy Proxy
Controller
DTO
Services
Repositories
Entities
RabbitMQ
gRPC
Database
RabbitMQ
Presentation Layer
Business Layer
Data Layer