Identity Management System

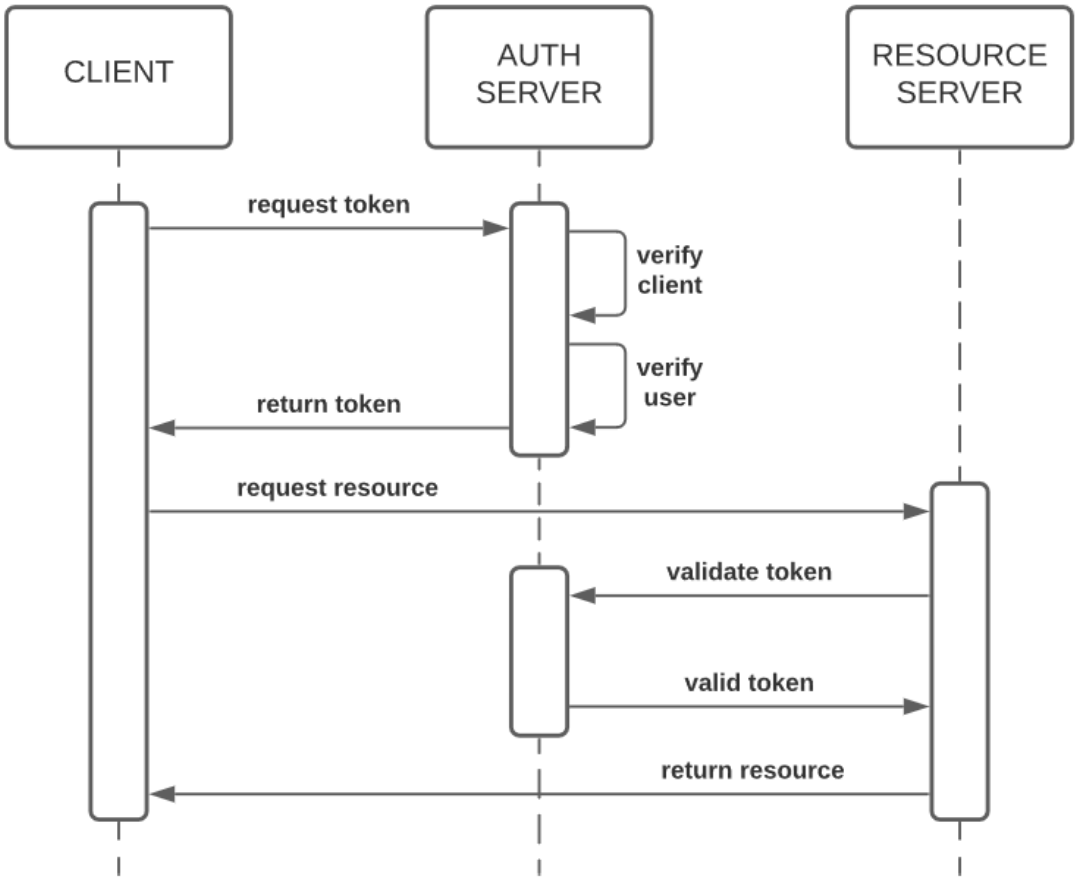
Our project is a real estate rental system based on microservice architecture in Java. To implement the identity system we used an auxiliary service called “Auth Server” and an independent microservice called “Account Service” that is responsible for all work with user profiles.

- **Auth Server**

The Auth Server service is an implementation of the "authorization server" pattern. All authorization and authentication in the system is carried out through it. In our system we use authorization protocol OAuth 2. This protocol is an authorization scheme, which allows the client to have limited access to protected resources without the need to transmit a login and password with each request. Also we used JWT (Json Web Token) open standard for creating token access based on JSON format.

To access protected resources, the user needs to send a request to the Auth Server with the correct login and password only the first time, in response the user will receive a token in which information about him is encoded. In the future, the user only needs to attach this token to requests to access resources. When other services receive requests for protected resources, they will send the token from the request to the Auth Server to verify its validity. Only microservices of the system can verify the token and get information about the user.

The working principle of Auth Server is shown in the picture below.



- **Account Service**

Account Service is a microservice responsible for all logic related to user accounts. This service implements such functionality as creating an account, obtaining information about the account, and editing it.

The internal architecture of this service is a classic three-layer architecture:

- Repository - all interaction with the database takes place through this layer;
- Service - business logic is implemented in this layer;
- Controllers - represent controllers that process requests from users;

The Account Service has access to a private Postgres database. Interaction with other services takes place via the HTTP protocol through the Spring Cloud OpenFeign framework.

Internal Architecture is shown in the picture below.