Microservice structure and global Architecture description

1. Introduction

This document describes the architecture of a microservice-based system consisting of four microservices: **User Service**, **Post Service**, **Connections Service**, and **Pipeline Service**. These services work together to provide a scalable, maintainable, and loosely-coupled system where each service is independently deployable. We use **Eureka** for service discovery to enable dynamic communication between microservices.

2. Microservices

The system consists of the following microservices:

a) User Service:

- Responsible for user authentication and profile management.
- Exposes REST APIs to manage user data.

b) Post Service:

- Manages posts created by users.
- Handles CRUD operations on posts
- Communicates with User Service for user data and Pipeline Service for pipeline necessary in a post.

c) Connections Service:

- Manages user relationships - friendships and group memberships
- Handles searching and showing friendship between users and group memberships based on the provided username.
- Communicates with User Service for user data

d) Pipeline Service:

- Manages pipelines with their initial images and transformations from Picastlo
- Communicates with User Service for user data

3. Service Discovery with Eureka

Each microservice is registered with Eureka Server for dynamic discovery. **Registry Server** acts as the central point for all service registrations and allows services to discover each other without hardcoding URLs.

URL for Eureka Server: `http://localhost:8761/eureka/`

4. API Gateway

We use API Gateway to route requests from clients to the appropriate microservice.

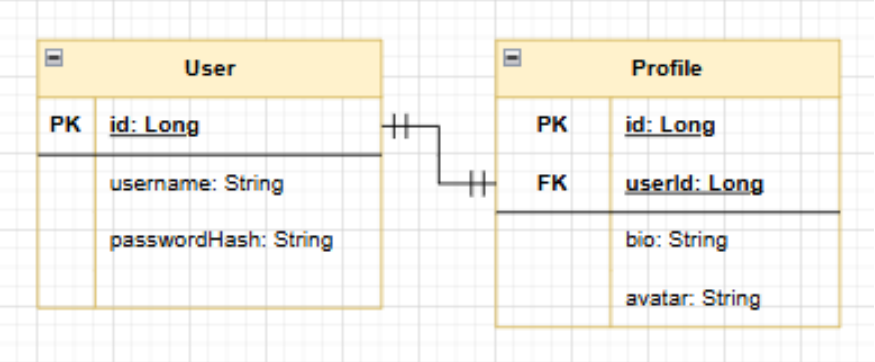
5. Security

We use Spring Security to secure each microservice. To do so, we use JWT tokens and capabilities.

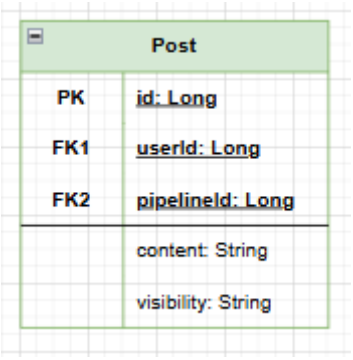
6. Database Design

Each microservice has its own database to ensure loose coupling and independence:

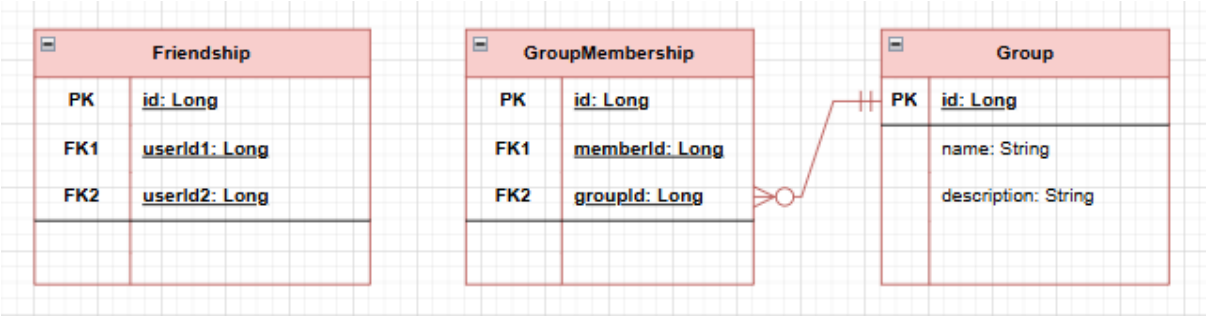
- **User Service**: Stores user details like ID, name, and hashed password and their profile information.



- **Post Service**: Stores post content, post visibility, user ID (as a foreign key) and pipeline ID (as a foreign key).



- **Connections Service**: Stores friendships (with user IDs as a foreign keys), group information (with group name and description) and group memberships (with group ID and memberID).



- **Pipeline Service**: Stores its name, image transformations, initial images, description and owner ID (as a foreign key).