코드리뷰 가이드

작성자 : 김희동

❑ 개요

- 본 문서는 코드 리뷰 진행 시 참고하기 위한 설명 및 가이드라인에 대한 문서입니다.
- 작성된 코드는 스마일게이트 스토브 데브캠프 2기 기간동안 '**디스코드 클론코딩**'을 주제로 진행된 프로젝트의 산출물입니다.
- 저는 백엔드 포지션으로 참여하여 아래와 같은 역할 수행 및 기능을 구현했습니다.
  1. 디스코드 내 커뮤니티 및 채널 관리를 위한 **커뮤니티 서버** 구현
  2. 화상채팅을 위한 **시그널링 서버** 구현 및 **미디어 서버** 연동
  3. **알림 서버** 구현

❑ 내용 구성

- 내용은 서비스를 기준으로 나뉘어져 있고, 구성은 다음과 같습니다.
  1. 서비스의 역할 및 주요 제공 기능 설명
  2. 핵심 기능 및 구현 포인트 설명
- 각 서비스에 세부적인 설명은 서비스별 루트 디렉토리의 readme에 첨부되어 있습니다.

❑ 관련 자료

항목	내용
팀 래퍼지토리	https://github.com/stove-smooth/sgs-smooth
커뮤니티 서버	https://github.com/stove-smooth/sgs-smooth/tree/develop/src/backend/community
시그널링 서버	https://github.com/stove-smooth/sgs-smooth/tree/develop/src/backend/signaling
알림 서버	https://github.com/stove-smooth/sgs-smooth/tree/develop/src/backend/notification

(작업은 하나의 래퍼지토리에서 브랜치 및 디렉토리를 나누어 작업을 진행했습니다.)

❑ 커뮤니티 서버

- 역할

서비스명	역할
커뮤니티 서버	디스코드 내에서 제공하는 커뮤니티 및 채널에 대한 관리 기능 담당

- 제공 기능

기능	설명
커뮤니티 관리 기능	- 커뮤니티, 채널에 대한 조회, 생성, 수정, 삭제 기능 제공 - 사용자별 커뮤니티 배치 위치 설정 기능 - 채널 순서 위치 변경 기능 제공
커뮤니티 초대 기능	초대장 생성 및 초대장을 통한 가입 기능 제공

- 핵심 기능 및 구현 포인트

1. 도메인 설계

    - 커뮤니티-카테고리-채널로 이어지는 계층적인 관계에서 상위 계층에서 퇴장 및 차단이 발생할 시 하위 계층의 명단에서도 제거되고, 같은 깊이의 계층에서 소속된 사용자의 수나 구성원이 달라질 수 있는 상황이 존재했고,

    - 접근 권한이 없는 유저의 접속을 제어하기 위한 **예외 처리**와 상위 계층에서 퇴장/차단 기능이 작동될 때 **도메인 내에 비즈니스 로직을 추가**하여 하위 계층에 저장된 구성원의 상태까지 함께 관리하는 로직을 구현했습니다.

    - 또한 사용자가 커뮤니티 및 채널의 **배치 순서를 자유롭게 이동**할 수 있는 기능을 제공하기 때문에 **최대한 적은 수의 채널이 영향을 받게 하면서** 위치를 자유롭게 변경할 수 있는 기능을 구현했습니다.

2. 사용자의 접속 상태 동기화

    - 커뮤니티에 접속한 사용자들의 목록을 제공하기 위해 상태관리 서버로부터 **사용자들의 접속 상태를 동기화**합니다.

    - 사용자가 접속하거나 종료하는 이벤트가 발생될 때 **상태관리 서버로부터 TCP 연결을 통해 해당 정보를 업데이트**합니다.

        - TCP 연결 설정 : /config/tcp

- 데이터 동기화 : /service/MessageService.java

2. 또한 **무결성을 위해 6시간 주기로 전체 사용자 접속 상태를 불러와서 동기화**합니다 (Full Query)

    - /util/UserStateUtil.java

3. 시그널링 서버에 연결해야 할 주소 제공

    - 게이트웨이를 통해 시그널링 서버의 웹소켓을 연결하는 경우 게이트웨이에 부하가 발생할 수 있고, 게이트웨이 내에 웹소켓 커넥션 수에 따른 분배 또는 음성 채널의 아이디에 따라 분배하는 로직을 포함해야 합니다.
    - 따라서 **음성 채널에 연결(시그널링 서버 연결)할 때 해당 음성 채널을 담당하는 시그널링 서버의 주소를 확인하여 클라이언트에게 제공**합니다.

        - /service/ChannelService.java

- URL : https://github.com/stove-smooth/sgs-smooth/tree/develop/src/backend/community

❑ 시그널링 서버 및 미디어 서버

- 역할

서비스명	역할
시그널링 서버	WebRTC 화상채팅을 제공하기 위해 클라이언트간 정보를 전달, 사용자의 엔트포인트를 미디어 서버로 연결해주는 기능을 담당
미디어 서버	클라이언트의 커넥션을 줄이기 위해 미디어 서버를 연결하여 통신을 제공

- 제공 기능

기능	설명
클라이언트 정보 전송	- WebRTC를 위해 필요한 사용자들의 SDP 및 ICE Candidate 정보 전달
방 관리 기능	- 웹소켓 설정을 통해 방 입장 및 퇴장을 관리하고, 이벤트가 발생했을 때 해당 방의 사용자에게 웹소켓을 통한 메시지 전송 및 상태관리 서버로 TCP 연결을 통해 해당 정보 전송 - 입장한 방에 따라 미디어 서버 파이프라인 연결

- 핵심 기능 및 구현 포인트

1. 웹소켓 메시지 핸들링

   - 방 입장, 정보 전송, 퇴장, 설정 변경에 대한 **메시지를 핸들링**하고 방에 입장 및 퇴장시킵니다.

     - 메시지 핸들러 : /service/MesageHandler.java

     - 메시지 타입 정의 : https://github.com/stove-smooth/sgs-smooth/wiki/시그널링-서버-메세지-타입

2. 방 정보 및 현재 웹소켓 연결 정보 외부 Redis에 저장

   - 시그널링 서버가 확장될 때 음성 채널 정보 및 접속 정보를 메모리에 저장하는 경우 서버별 접속 상태를 확인할 수 없기 때문에 외부 Redis에 값을 저장했습니다.

   - 또한 음성 채널에 접근할 때 채널의 아이디에 따라, 처음 입장하는 경우 현재 서버 별 연결 수에 따라 시그널링 서버의 주소를 제공해주기 때문에 외부에 값을 추가적으로 저장하고 있습니다.

3. 시그널링 서버 연결 방법

   - 초기 설계 시 시그널링 서버가 확장될 때를 고려하여 내부의 메모리를 활용하지 않고 redis와 같은 외부 캐시 메모리에 저장하여 공유하고자 했습니다.

   - 하지만 kurento에서 제공하는 데모 샘플과 kurento client를 활용하는 과정에서 채널에 대한 입장 정보를 담은 메모리를 저장하지 않고 발생되는 이벤트(ICE candidate가 발생될 때 이벤트)를 처리하는 것에 어려움이 있었습니다.

   - 따라서 **접속하는 음성 채널을 기준으로 같은 시그널링 서버에 연결하는 로직으로 구현했고, 음성 채널이 존재하지 않는 경우 가장 연결 수가 적은 시그널링 서버에 연결**하게 했습니다.

   - 시그널링 서버 내에서 담당하는 음성 채널이 존재하고, 해당 채널의 사용자 접속 정보를 내부 메모리로 관리하면서 이벤트를 처리하고, 외부에 값을 저장하는 형태입니다. (**상태관리 서버에 TCP 연결을 통해 정보 전송**)

- URL : https://github.com/stove-smooth/sgs-smooth/tree/develop/src/backend/signaling

❑ 알림 서버

- 역할

서비스명	역할
알림 서버	클라이언트에게 푸시 메시지를 전송하는 기능을 제공

- 제공 기능

기능	설명
푸시 메시지 전송	- 푸시 메시지 전송

- 핵심 기능 및 구현 포인트

1. 푸시 메시지 전송
    - 채팅 서버와 상태관리 서버를 통해 푸시 메시지를 전송해야 할 유저들의 아이디와 함께 메시지의 내용을 전달받습니다.
    - 유저 아이디를 기준으로 유저 서버에서 디바이스 토큰 및 타입(ex. Web, iOS)을 조회하고, 메세지 내에서 전송해야 할 플랫폼을 기준으로 그룹화합니다.
    - 이후 플랫폼별로 그룹화된 메시지를 일괄 전송합니다.

- URL : https://github.com/stove-smooth/sgs-smooth/tree/develop/src/backend/notification