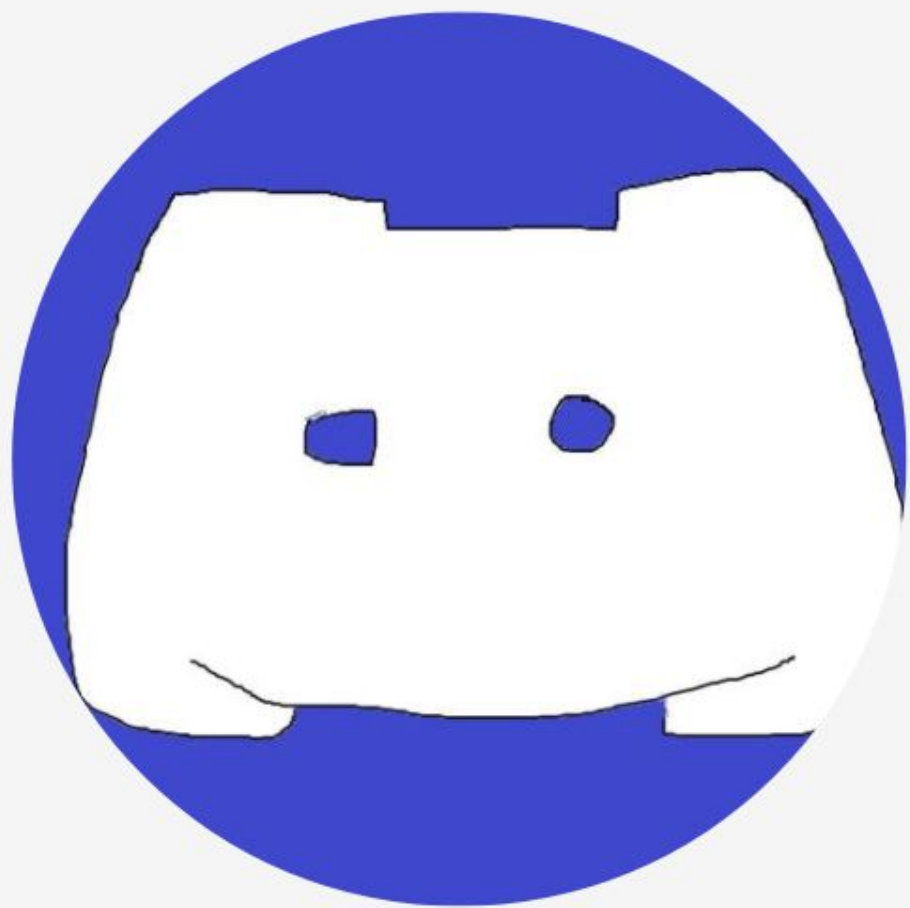


- 발표 포함사항
 - 프로젝트 목표 & 목표 달성 / 아키텍처 등 (** 프로젝트 전체를 표현하고 어필하는 항목)
 - 시연 / 결과물
 - 프로젝트 결과 시연은 "영상"으로 준비하는 것을 권장합니다. (현장 네트워크/장비 상황에 따라 원활한 진행 불가능할 수 있음)
 - 영상이 아닌 다른 방법으로 준비하는 경우 꼭! 미리 얘기해주세요.
 - 다른 방법으로 준비하더라도, 시연 영상은 같이 준비해주세요.
 - 역할 분담 / 결과물 / 개인 목표
 - 개인 단위의 상세 결과물과 어필 / 목표 달성 결과물
 - 발표 시간 관계상 발표 내용에서는 제외했지만 꼭 포함하고 싶은 부분은 **Appendix**.로 추가 ← 스토브 전환 심사 과정에서 활용 가능
 - 개인 단위의 회고
 - ex. 느낀 점, 좋았던 점 / 달성한 것, 아쉬웠던 점 / 달성하지 못한 것, 어려웠던 점, 앞으로의 목표 등
 - ** 팀 / 개인의 결과물과 목표 달성 내용 & 성장을 어필할 수 있도록 자유롭게 구성 (발표 시간 고려 필수)
 - ** 팀 단위 프로젝트에 영향이 없도록 합의 하에 산출물을 마무리 하되 발표는 개인을 어필할 수 있도록 자유롭게 구성 가능합니다.
- 발표는 반드시 "팀" 발표와 "개인" 발표가 포함되어야 합니다.
 - 최종 발표는 대표 한 분이 발표를 하는 것이 아닌, 담당할 영역에 대해서는 직접 발표하는 형태로 진행됩니다.



목차 30초

1. 팀소개 30초
2. 프로젝트 소개 30초
3. 시연 영상 3분
4. 아키텍처 3분
5. 역할분담 1분
6. 박병찬 5분
7. 김희동 5분
8. 김민지 5분
9. 김두리 5분
10. 프로젝트 달성 어필 3분
 - a. 목표 관련된 얘기
 - b. 마일스톤
 - c. 회고

34분 + a

40분 전부 가져가는 것은 비추 지루할 수 있음, 끝까지 흥미진진하게 하려면 추천

팀 소개

30초

TEAM MEMBER

Smooth하게 소개하겠습니다!



박병찬

Backend



김희동

Backend



김민지

Frontend



김두리

iOS

프로젝트 소개 (개요 느낌)

30초



스무스에서 즐겨보세요!

나만의 커뮤니티를 만들어 친구들과 소통하세요!

개인 간 다이렉트 메시지와, 커뮤니티 간 그룹 메시지를 즐겨보세요!

문자, 음성, 영상 자유로운 메시지를 전달하세요!

실시간으로 이뤄지는 메시지를 즐겨보세요!

시연 영상

3~5분

5분 동안 흥미롭게 하는 것이 key point

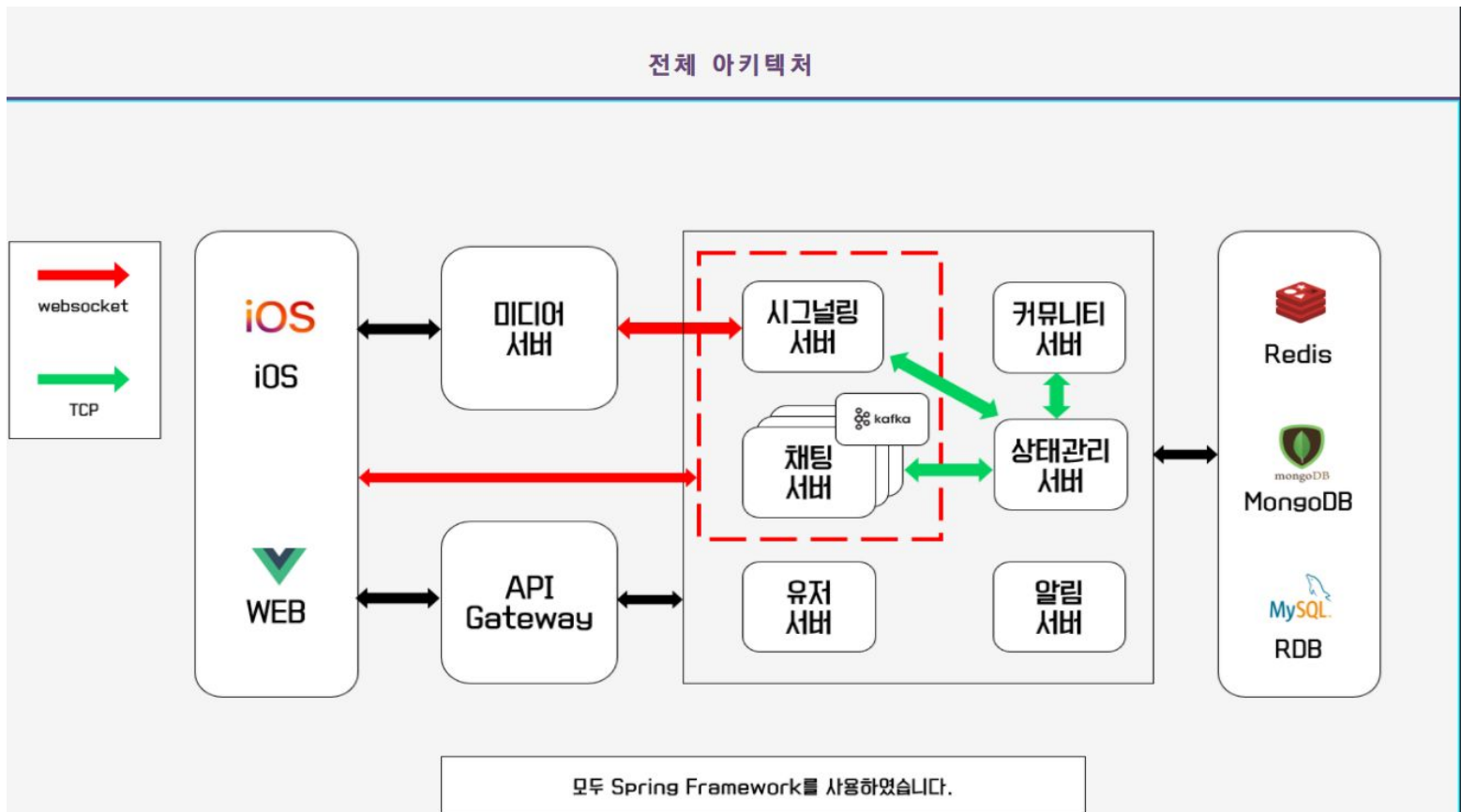
회원가입 넣돼 빨리감기, 편집 느낌으로

영화보듯 5분 틀어 놓는다는 느낌은 절대 안돼

보이스 설명 or 자막 있으면 좋다

아키텍처

3분



역할 분담

1분

업무 분담



박병찬(BE)

API Gateway
유저 서버
채팅 서버
상태관리 서버



김희동(BE)

시그널링 서버
미디어 서버
커뮤니티 서버
알림 서버



김민지(FE)

UI/UX 설계
Websocket 연동
WebRTC 연동
FCM 연동



김두리(iOS)

UI/UX 설계
Websocket 연동
WebRTC 연동
FCM 연동

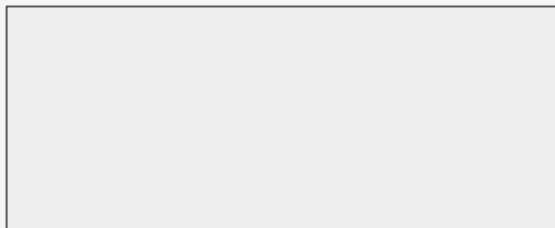
상태관리 서버 어떻게 효율적으로 연결하지?

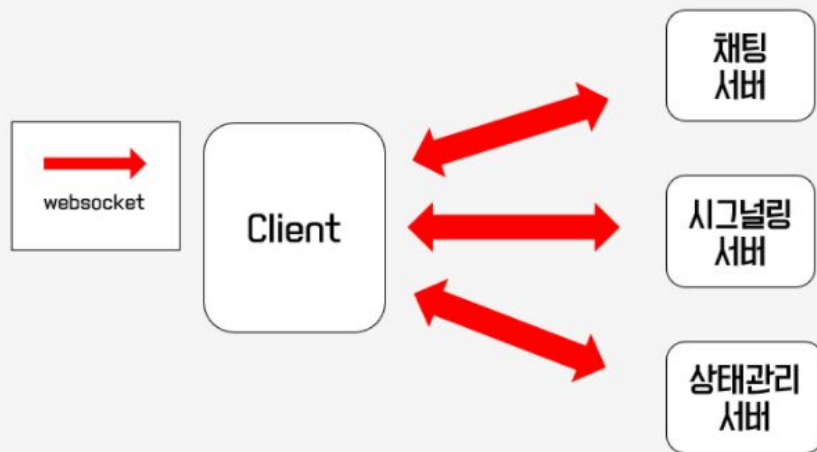
기본적인 채팅 websocket

음성,영상 채팅을 위한 WEB RTC websocket

상태관리를 위한 websocket까지?!

채팅 서버 어떻게 MSA관점으로 확장하지?





그래! 채팅 연결 하려면 웹소켓 해야지!

음.. 그래~ 음성 채팅도 웹소켓 연결 해야지..!

상태관리도.... 여기에도 웹 소켓..?

유저 한 명에 웹 소켓이 3개씩이나..???

상태 관리로 받아온 정보를 다른 서버에게 다시 전달 하는 구조

로그인이 곧 온라인!

로그인시 바로 채팅 서버 웹소켓에 연결해보자!



음성 채팅은 방에 입장시에만 연결이 된다!

그렇다면.. 최대 2개로 줄일 수 있다!
채팅 서버에서 상태 관리 서버로 데이터를 보내자!

어떻게 효율적으로 통신을 할까?



1. REST API

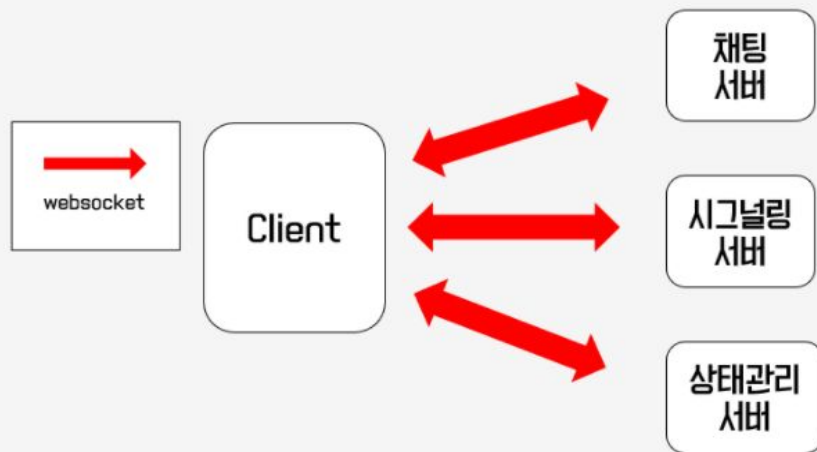


2. Web socket



3. TCP socket





그래! 채팅 연결 하려면 웹소켓 해야지!

음.. 그래~ 음성 채팅도 웹소켓 연결 해야지..!

상태관리도.... 여기에도 웹 소켓..?

유저 한 명에 웹 소켓이 3개씩이나..???

상태 관리로 받아온 정보를 다른 서버에게 다시 전달 하는 구조

김희동 개인

5분

상세 결과물 어필

개인 목표, 달성 수준

개인단위 회고(좋았던 점 , 아쉬운 점)

김희동 개인

내용은 이렇고 표지 규격에 맞게
통일되게 채워넣으면 될 듯



커뮤니티 서버

- 커뮤니티 관련 도메인 설계 및 서버 구현
- 커뮤니티, 채널, **DM CRUD**
- 음성 채널에 대한 시그널링 서버 연결 **IP** 제공
- 상태관리 서버와 연동을 통한 사용자 접속 정보 동기화

시그널링 서버 및 미디어 서버 연동

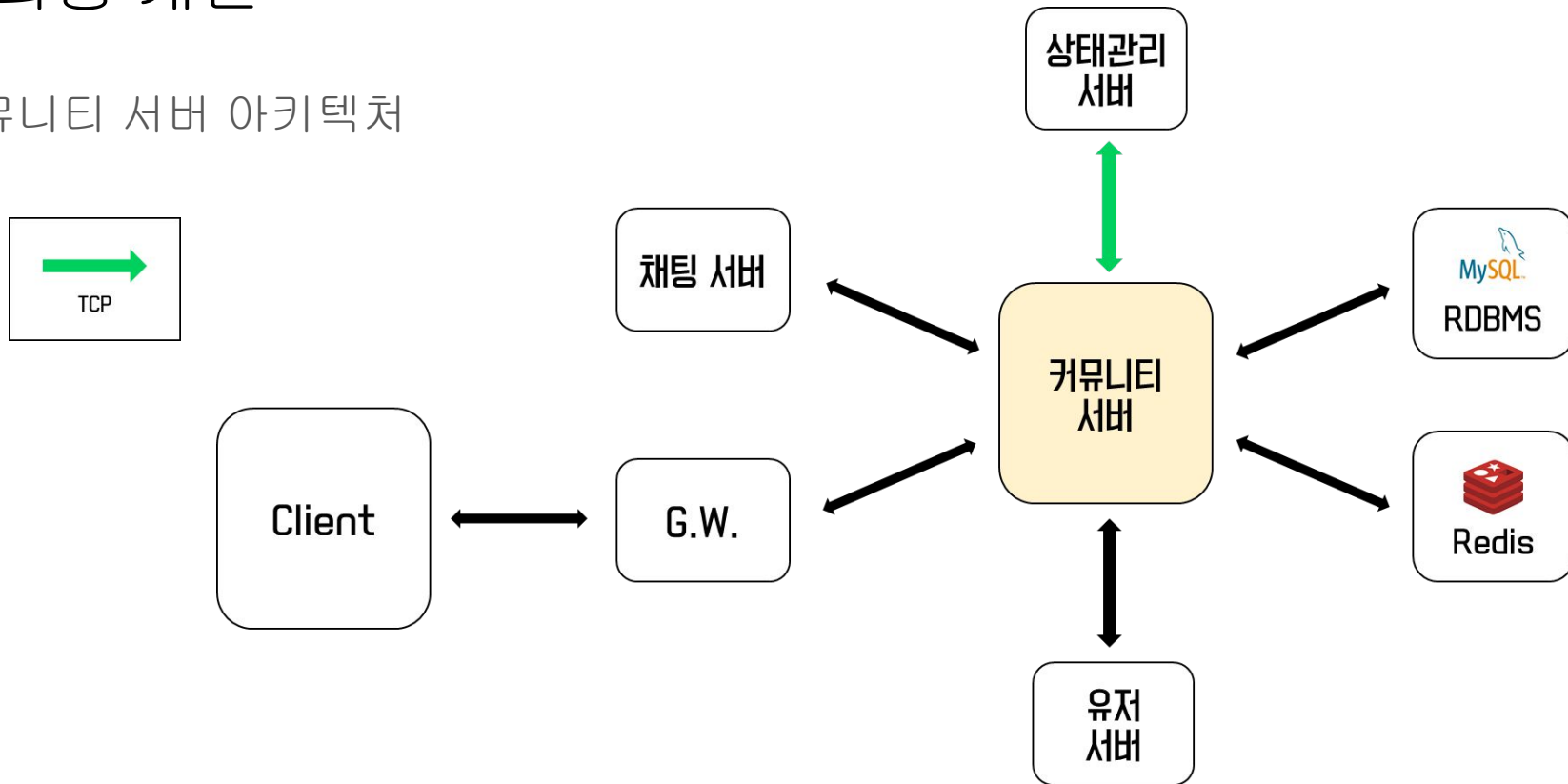
- 시그널링 서버 설계 및 개발
- 미디어 서버 인프라 구축
- 시그널링 서버 접속 정보 저장 및 상태관리 서버 연동

알림 서버

- **FCM** 연동
- 플랫폼 별 메세지 일괄 전송

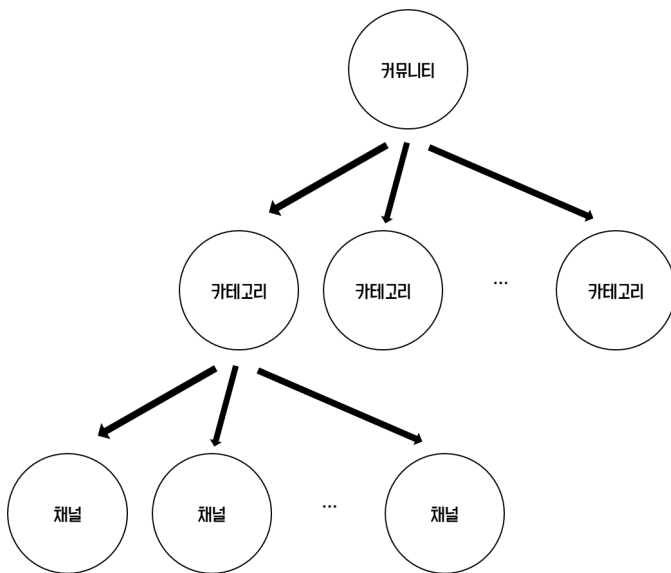
김희동 개인

커뮤니티 서버 아키텍처



김희동 개인 결과물 - 커뮤니티 서버

- 디스코드 → 커뮤니티 기반의 메신저 프로그램
- 사용자의 모든 활동은 커뮤니티 기반으로 진행

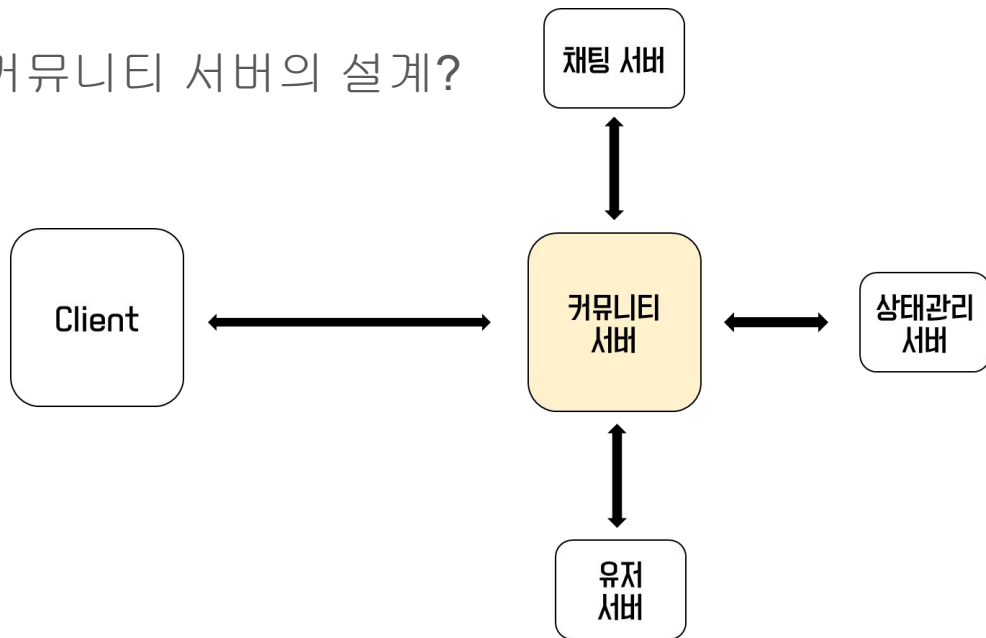


1. 상위 그룹에서 변경된 조건에 따라 하위 그룹이 영향을 받는 상황
2. 사용자의 자유로운 커스터마이징
3. 권한 별 기능 및 접근 제한

김희동 개인 결과물 - 커뮤니티 서버

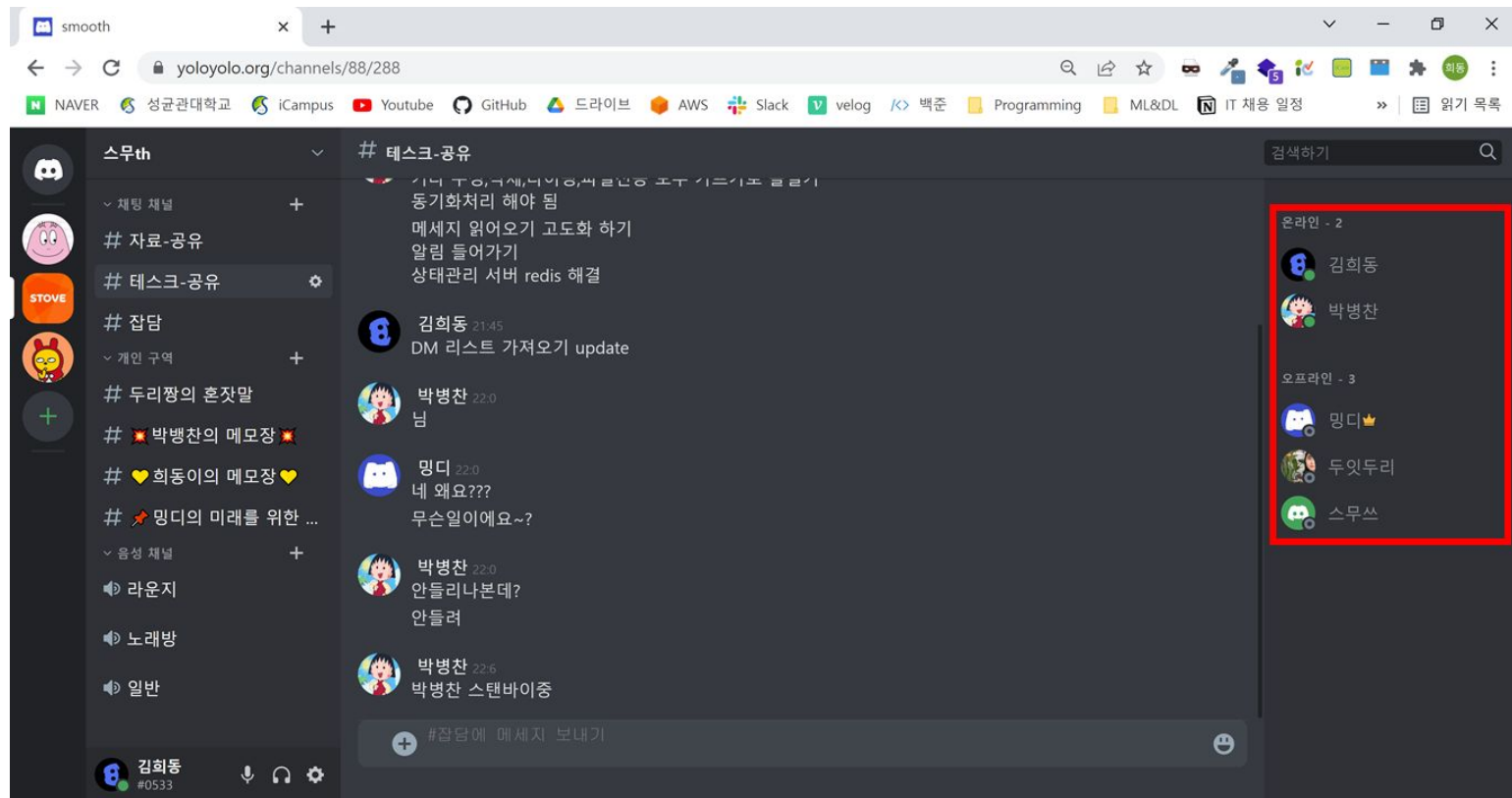
MSA 관점에서 커뮤니티 서버의 설계?

-
-



- 기능에 따라 서비스들을 분리해서 개발
- 오히려 하나의 기능을 제공하기 위해 여러 서비스에 많은 요청을 해야하는 상황 발생

김희동 개인 결과물 - 커뮤니티 서버



김희동 개인 결과물 - 커뮤니티 서버

효율적으로 요청하고 관리하자

-
-

1. 상태관리 서버와 **TCP** 연결

2. 데이터의 무결성을 위해
스케줄링을 통한 **Full Query**

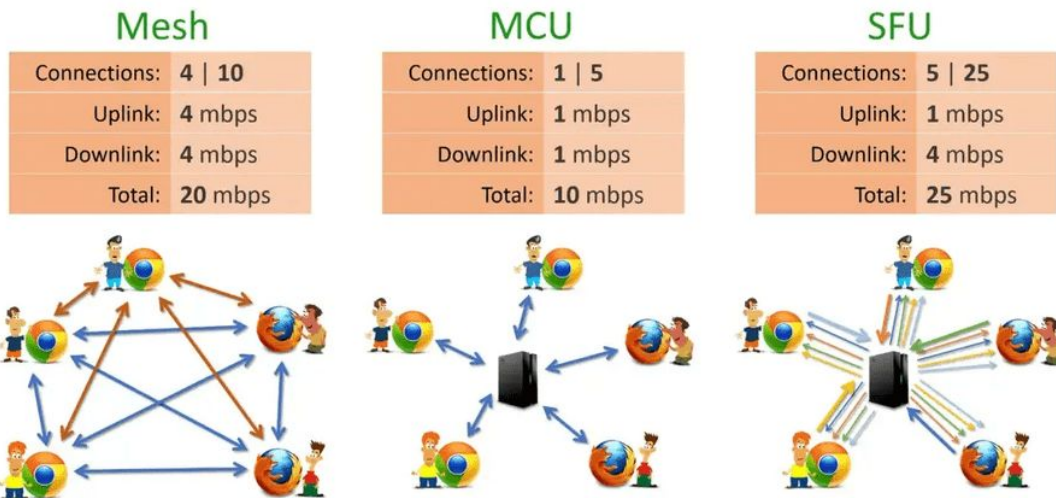


김희동 개인 결과물 - 시그널링 서버



김희동 개인 결과물 - 시그널링 서버

WebRTC 구현 방식



출처 :

<https://millo-l.github.io/WebRTC-%EA%B5%AC%ED%98%84-%EB%B0%A9%EC>

김희동 개인 결과물 - 시그널링 서버

WebRTC 구현 방식

Mesh

Connections:	4 10
Uplink:	4 mbps
Downlink:	4 mbps
Total:	20 mbps



MCU

Connections:	1 5
Uplink:	1 mbps
Downlink:	1 mbps
Total:	10 mbps



SFU

Connections:	5 25
Uplink:	1 mbps
Downlink:	4 mbps
Total:	25 mbps

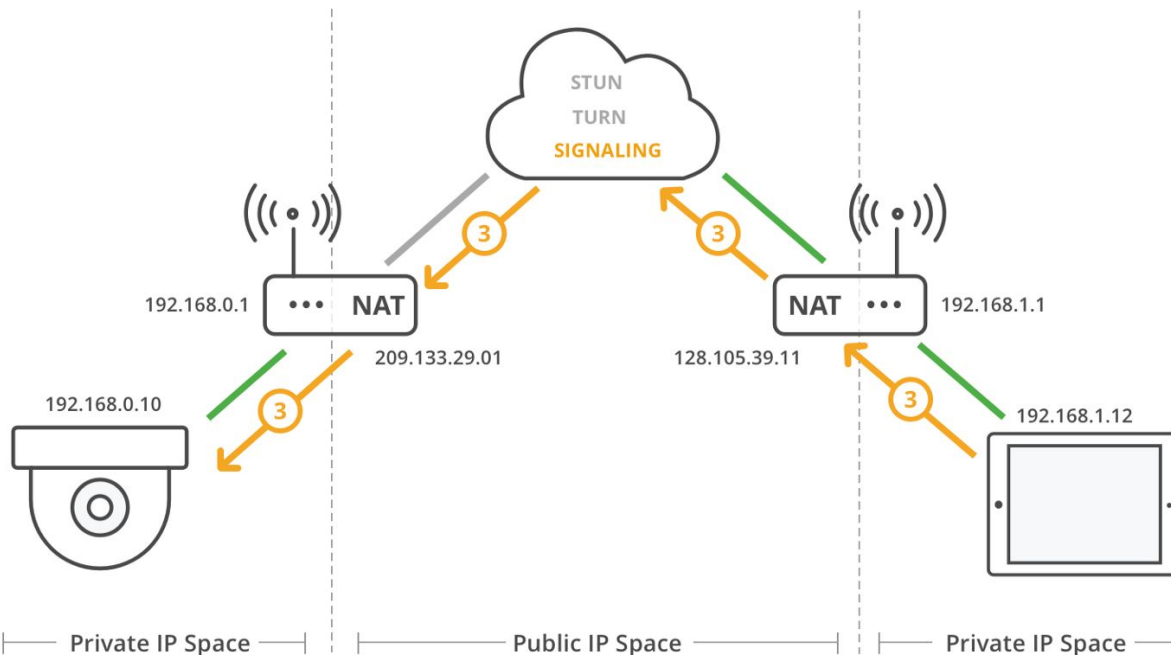


출처 :

<https://millo-l.github.io/WebRTC-%EA%B5%AC%ED%98%84-%EB%B0%A9%EC>

김희동 개인 결과물 - 시그널링 서버

STUN, TURN

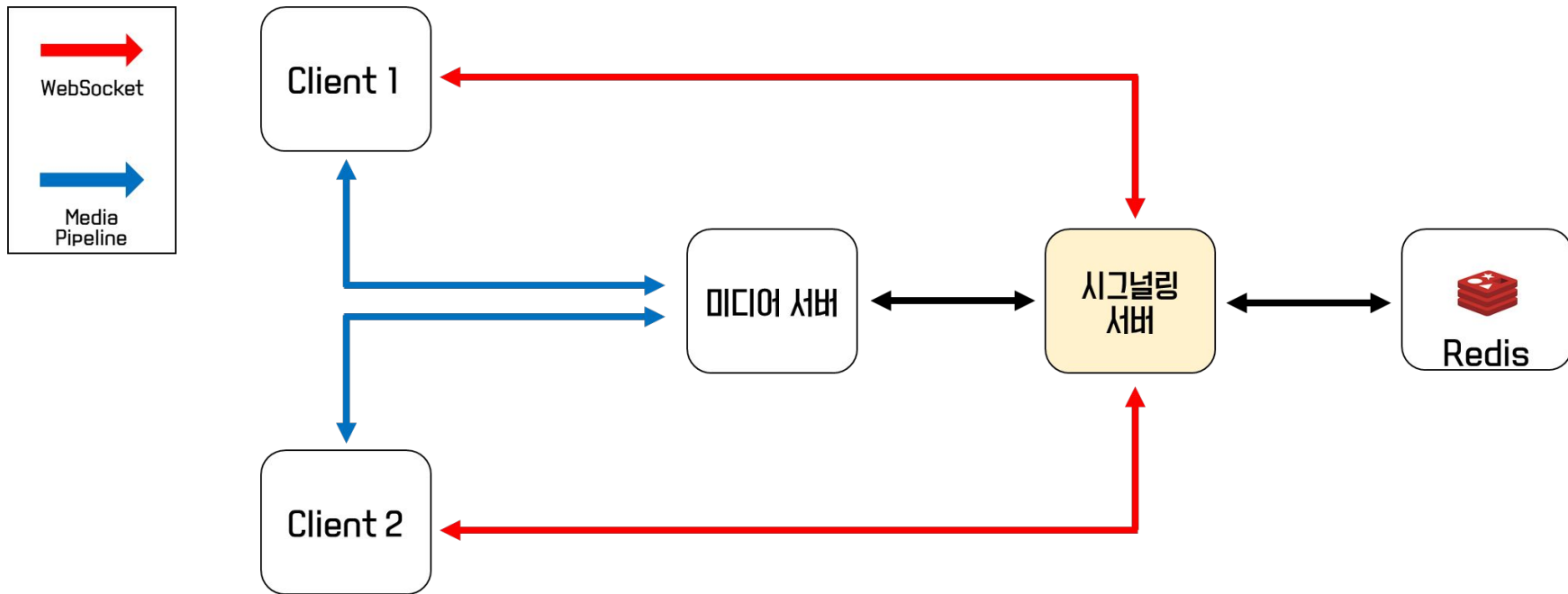


출처 :

<https://rajbharathmail.medium.com/create-stun-turn-server-in-ubuntu-18-04-aws-a59ef49c7659>

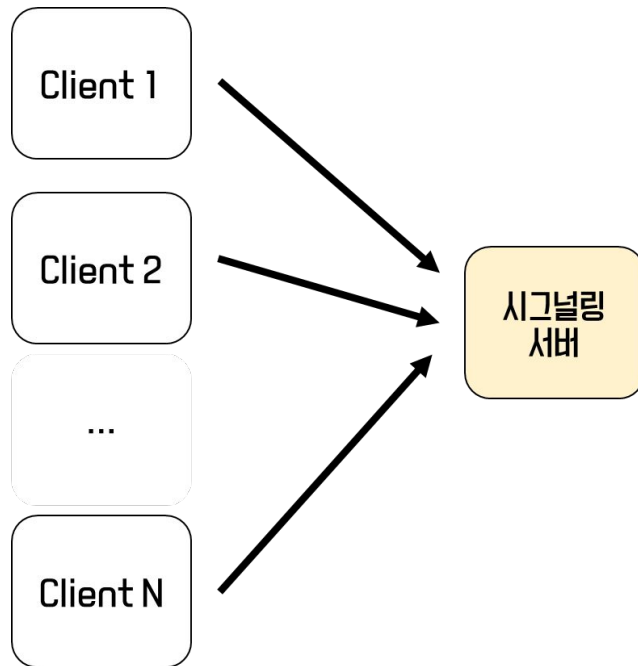
김희동 개인 결과물 - 시그널링 서버

시그널링 서버 - 아키텍처



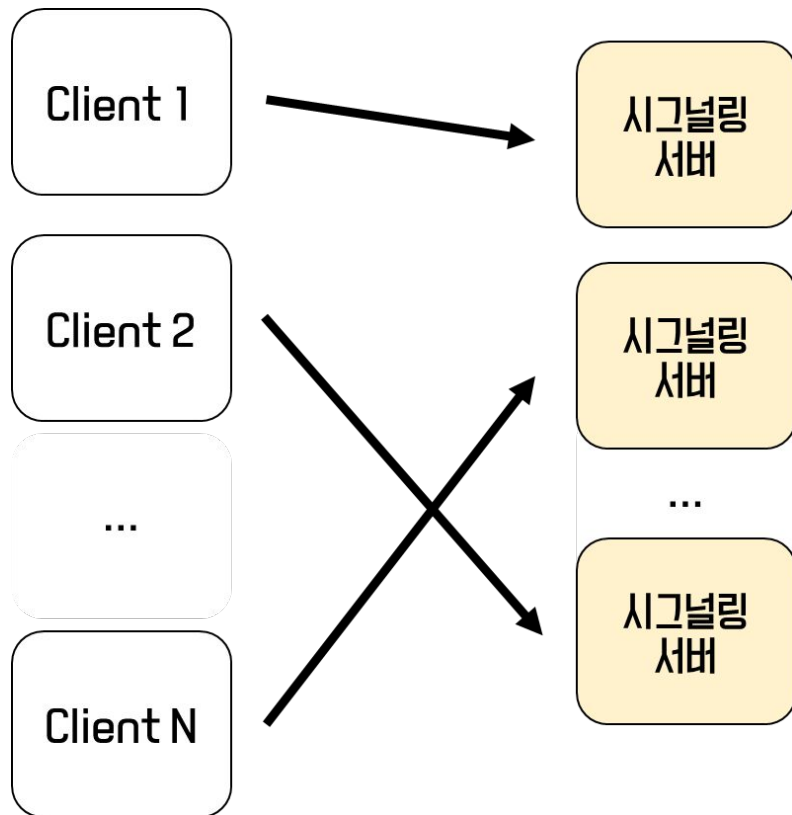
김희동 개인 결과물 - 시그널링 서버

시그널링 서버의 확장성



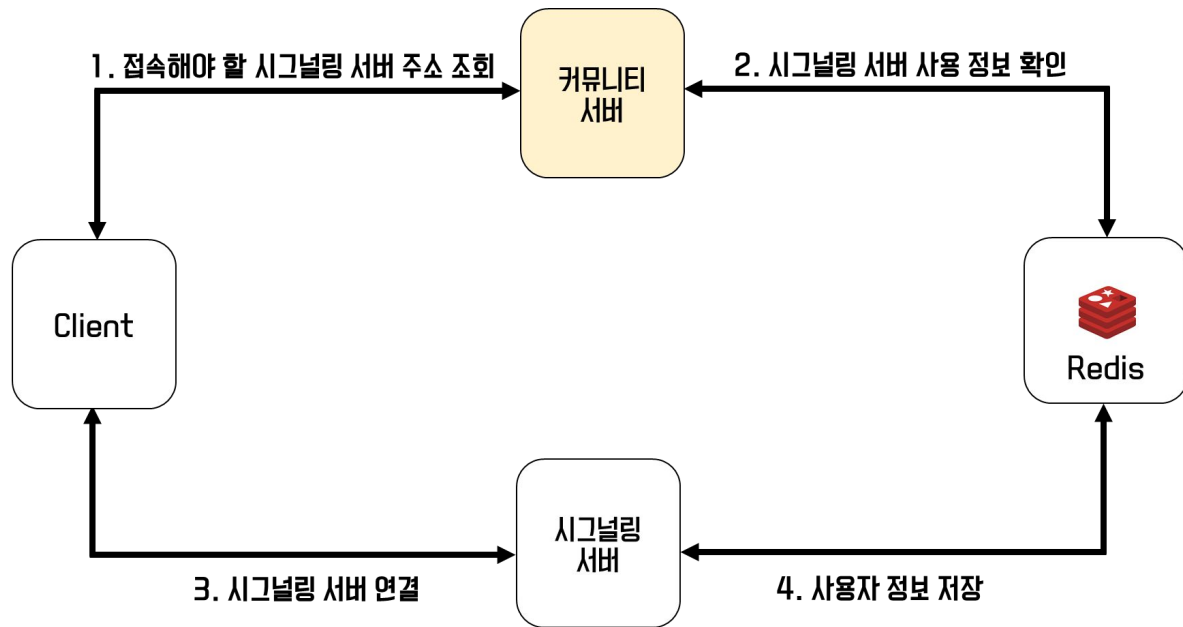
김희동 개인 결과물 - 시그널링 서버

시그널링 서버의 확장성



시그널링 서버의 확장성

김희동 개인 결과물 - 시그널링 서버



김희동 개인 목표

1. 대용량 서비스를 고려한 설계 및 개발 능력을 갖춘 상태
 - MSA 관점에서 설계를 진행
 - 서비스간 요청 및 메세지의 효율성을 고려한 설계
2. 사용한 기술 및 방법에 대해 이유를 설명할 수 있는 상태
 - WebRTC에 대한 이해
 - 사용 기술 및 선택 배경에 대한 정리
3. 성능 측정 및 기능을 개선하는 습관
 - Sonarqube를 통해 코드의 중복성을 낮추기 위해 노력
 - nGrinder를 통해 성능 측정 후 API의 부분적인 병문 형식 제거 및 여격 바번

김민지 개인

웹 프론트엔드 개발 (vue)

전체 ui 구현

커뮤니티

- 커뮤니티, 카테고리, 채널 **CRUD**,
- **drag and drop**
- 초대장 전송, 입장

DM

- 그룹 DM , 1:1 DM

채팅

- **webSocket**을 이용한 실시간 채팅

상태

- 실시간 타이핑 상태
- 친구/멤버 온라인/오프라인 상태
- 음성 채널 입장 상태

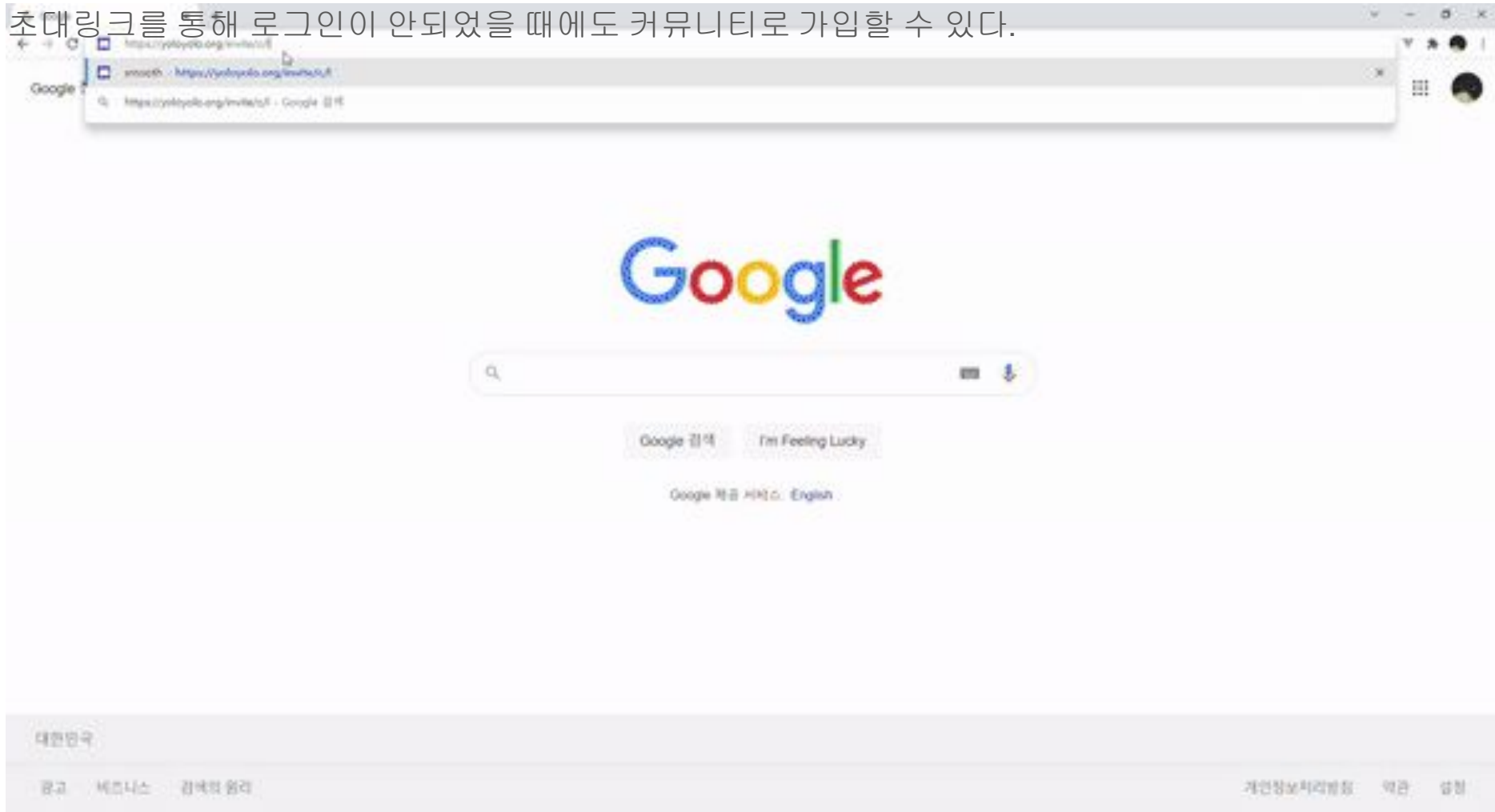
음성통화

- **webRTC**를 이용한 실시간
음성/영상 통화 구현

알림

- **FCM**을 통한 알림 수신
- 안읽은 메시지 처리

초대링크를 통해 로그인 안되었을 때에도 커뮤니티로 가입할 수 있다.



김민지 개인

WebSocket

라우트 구조를 트리 구조로 변경

로그인 후에 socket 연결이 true가 되게끔
하고, 로그인 후에는 어디서든 구독을 받을 수
있게끔 하기 위해 라우트 구조를 설정하였다.

로그인 후 socket 연결하고 socket이
연결되었을때만 채팅 페이지가 렌더링되게
하여 채팅이 문제없이 이루어진다.

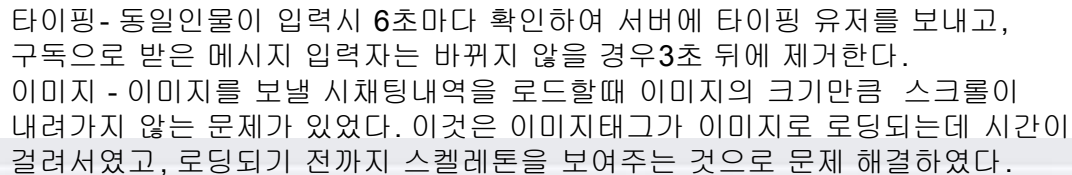
```
src > routes > index.js > ...
16 mode: "history",
17 routes: [
18   {
19     path: "/login",
20     name: "LoginPage",
21     component: LoginPage,
22   },
23   {
24     path: "/register",
25     name: "RegisterPage",
26     component: RegisterPage,
27   },
28   {
29     path: "/",
30     name: "MainPage",
31     component: MainPage,
32     children: [
33       {
34         path: "channels/@me/:id",
35         name: "privateDmPage",
36         component: PrivateDMPage,
37         meta: { auth: true },
38       },
39       {
40         path: "channels/@me",
41         name: "MyPage",
42         component: MyPage,
43         meta: { auth: true },
44       },
45     ],
46   },
47 ]
```

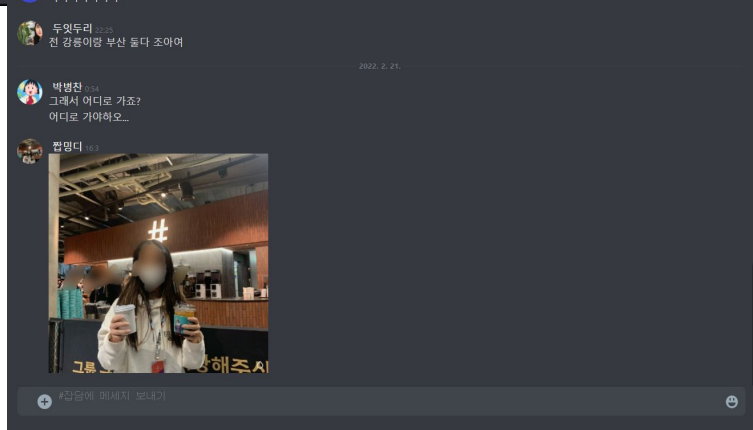
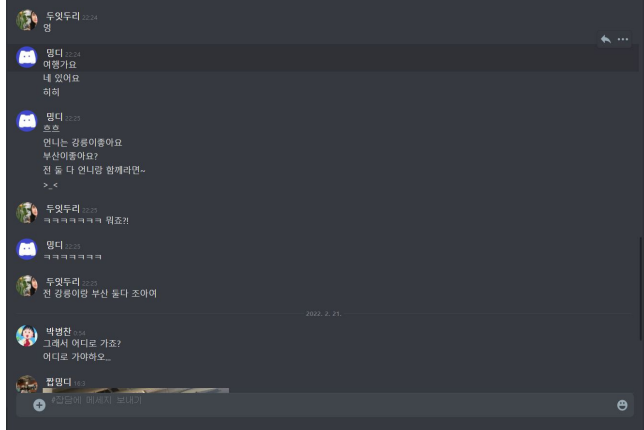
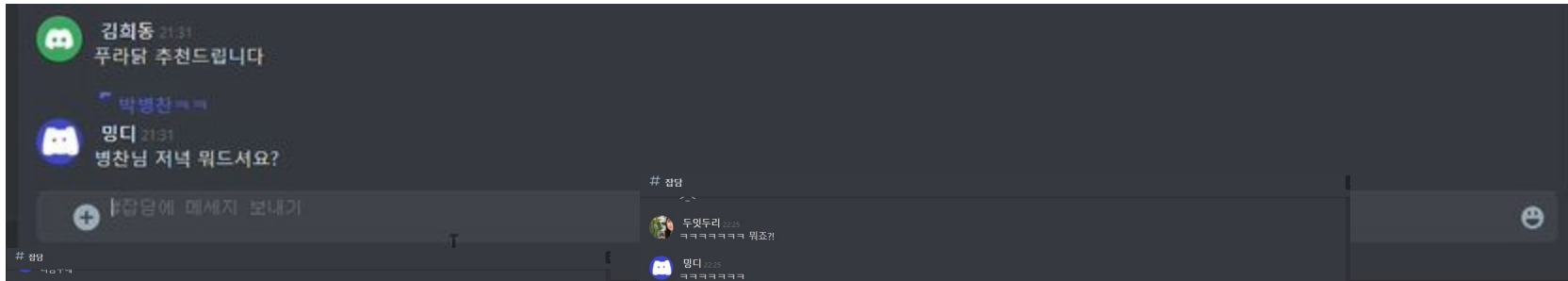
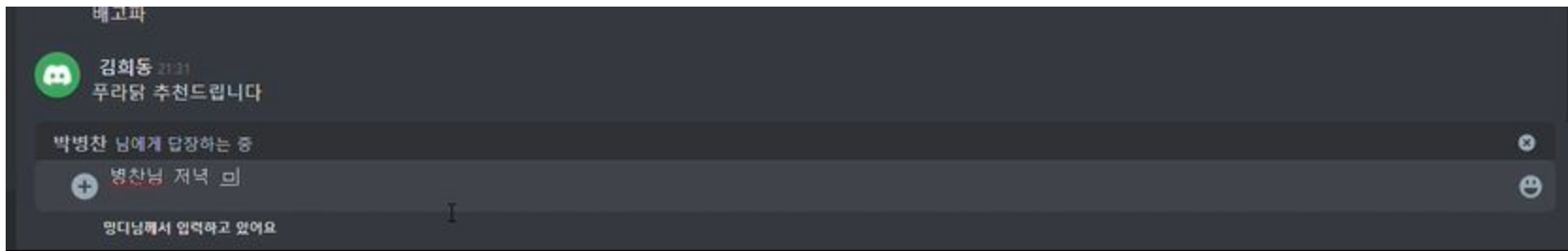
로그인 전

로그인 후

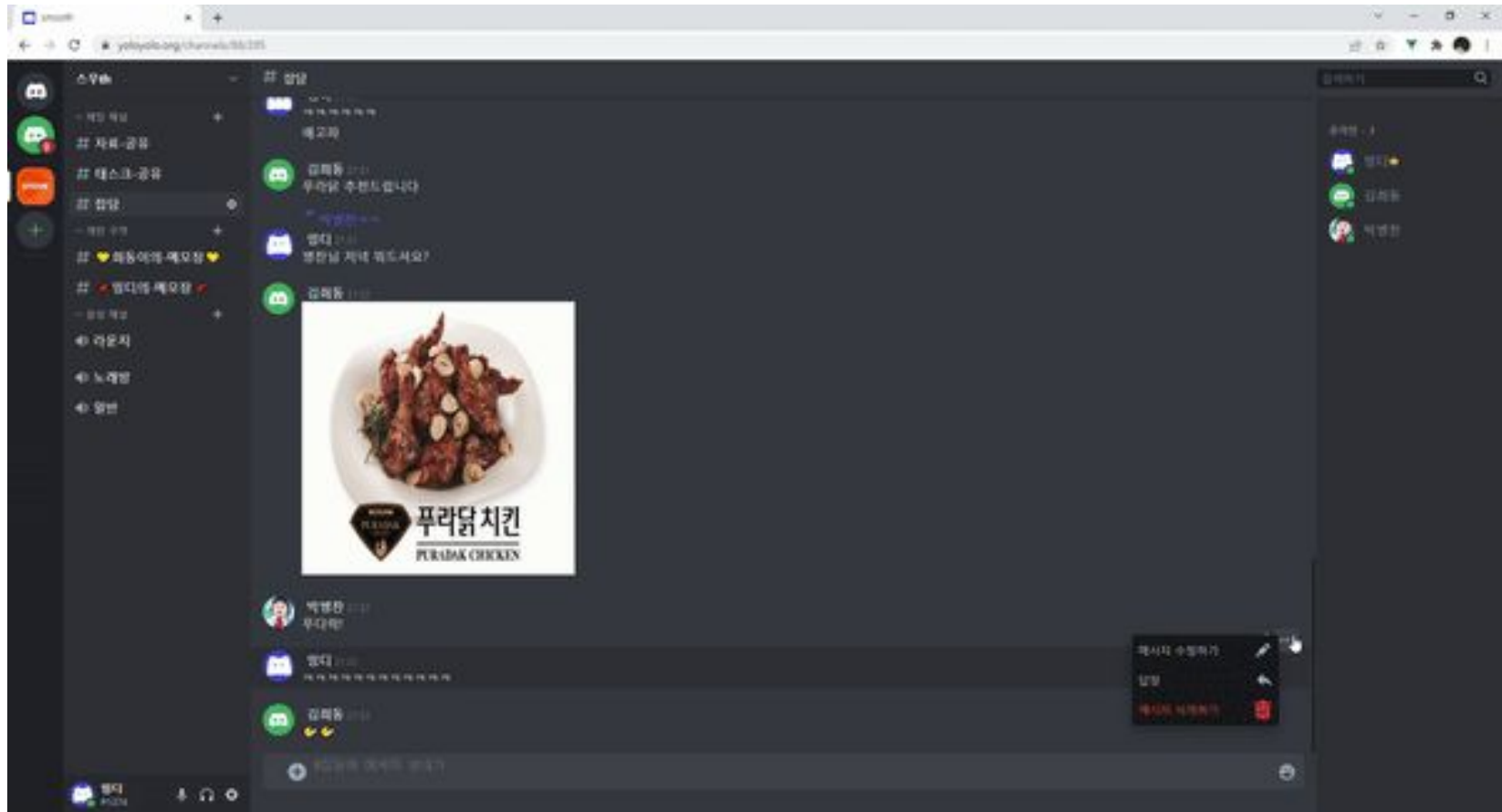
타이핑 - 동일인물이 입력시 **6초**마다 확인하여 서버에 타이핑 유저를 보내고, 구독으로 받은 메시지 입력자는 바뀌지 않을 경우**3초** 뒤에 제거한다.

이미지 - 이미지를 보낼 시 채팅내역을 로드할때 이미지의 크기만큼 스크롤이 내려가지 않는 문제가 있었다. 이것은 이미지태그가 이미지로 로딩되는데 시간이 걸려서였고, 로딩되기 전까지 스켈레톤을 보여주는 것으로 문제 해결하였다.





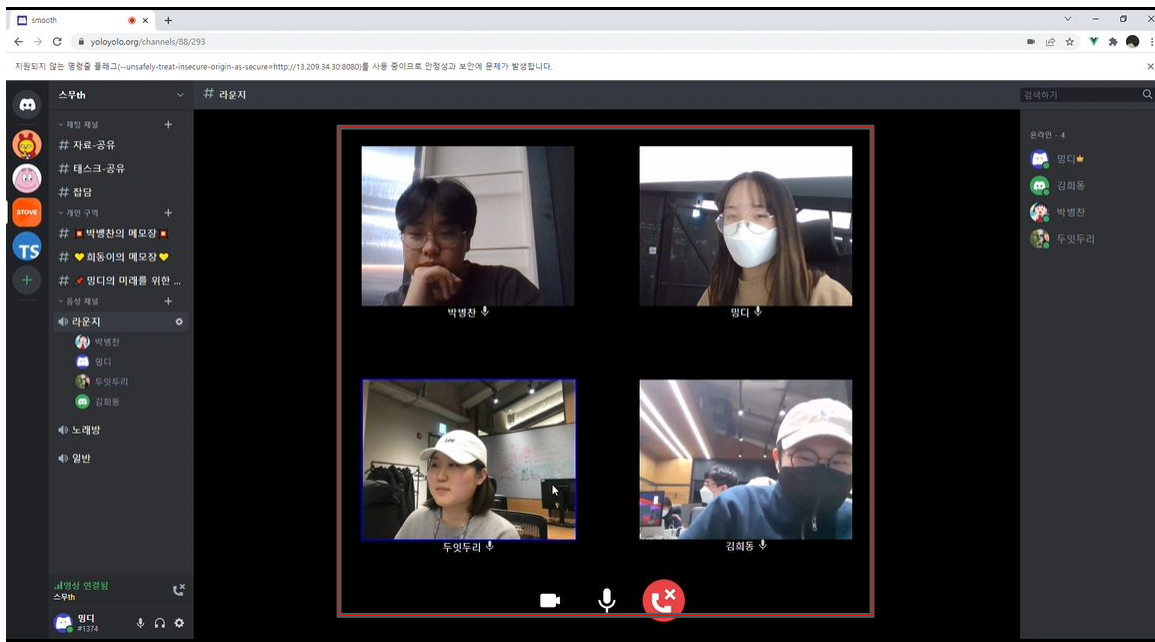
메시지 수정 및 이모지



김민지

WebRTC

1. 참가자 관리
 - 시그널링 메시지를 전송 및 수신을 하며 참가자 관리를 원활하게 하기 위하여 관련 로직을 **vuex**에 등록했다.
 - 그래서 참가자가 변할 때마다 비디오를 유연하게 보여줄 수 있다.



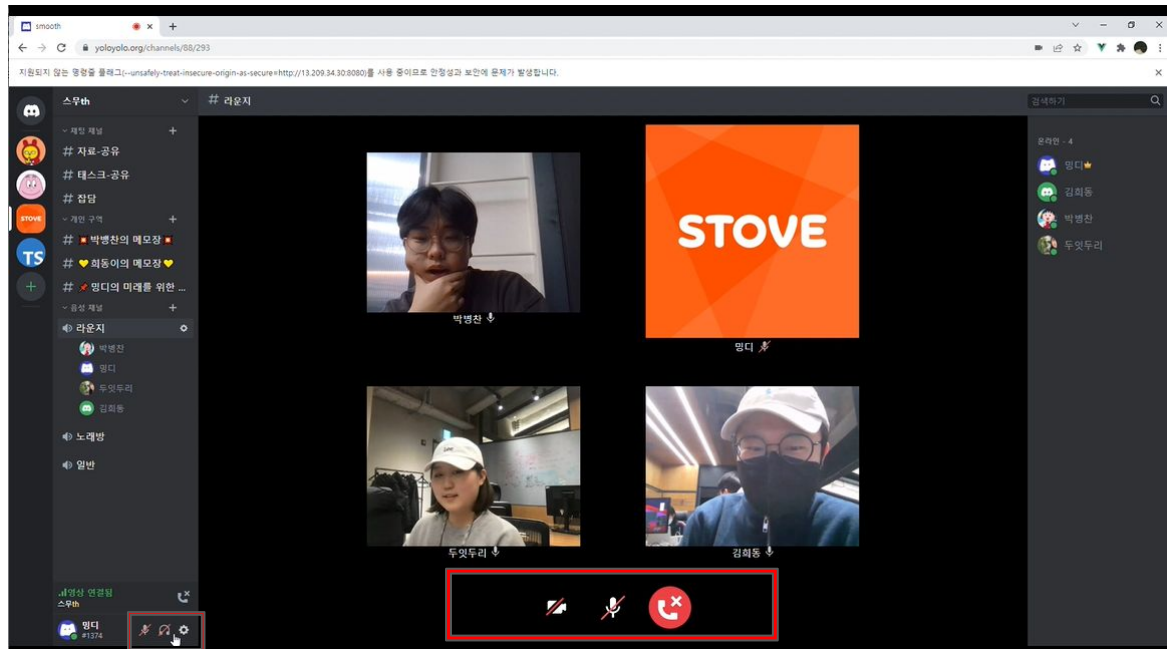
김민지

WebRTC

2. 마이크, 카메라 관리

다른 참가자의 마이크와 카메라 상태를 관리하기 위해 시그널링 메시지를 전송 및 수신한다.

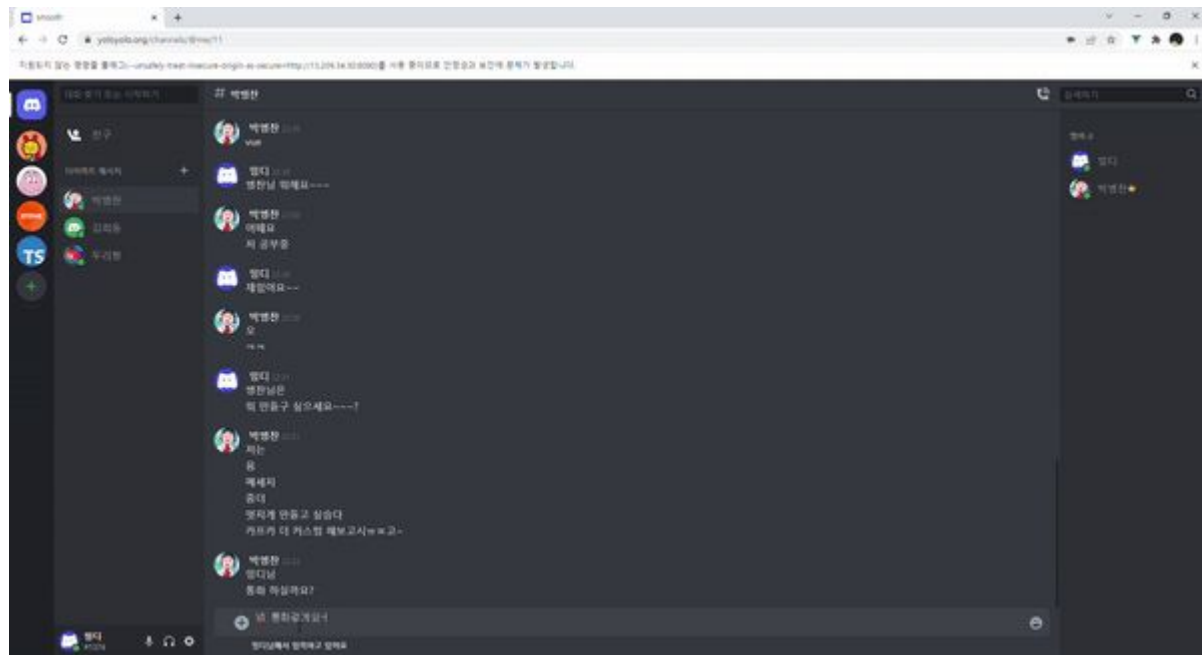
그것을 토대로 vuex에서 관리한다.



김민지

WebRTC음성을 감지하여 사용자에게 보여준다

3.



김민지 개인목표

상태관리 라이브러리? 안써봤어요..!

소규모 프로젝트, 간단한 **CRUD** 기능 구현 경험

김민지 개인 목표		
소규모 프로젝트 경험이 부족해요	클린 코드를 작성하는 개발자 되기!!	코드 개선하기! 코드 리뷰하기!!
대용량 서비스 프로젝트에 참여할 기회가 없다. 소규모 프로젝트만 참여했다.	대용량 서비스 프로젝트 개발 능력을 갖춘다.	1주일에 한 번 자가 감시하여 코드를 더 개선할 수 있는지 고민한다.
간단한 기능만 개발해서 코드 작성 경험이 많지 않은 상태이다.	1년 뒤에 백도 빠르게 이해할 수 있는 클린코드 작성 능력을 갖춘다.	2주일에 1번 주기로 자제어, [백엔드] 남게 요행한다. 코드리뷰하여 목표가 잘 지켜지고 있는지 점검한다.
현재 상태	목표	상세 계획

1. 대용량 서비스 개발을 위한 다양한 기능 사용에 대한 도전 경험을 갖추고
싶다!

vue? vuex? 공부해보자.

webSocket? 실시간 채팅? 실시간 상태 관리?

webRTC?

FCM?

김민지 개인목표

2. 대용량 서비스 속에서도 코드를 개선하여 좋은 코드를 작성하고 싶다!

피드백 수용

- 라우트 가드와 인터셉터에 대한 명확한 파일분리
- 라우트 구조를 계층 구조로 변환
 - watch를 통한 주소 변화 감지
 - socket disconnect 위치 destroyed 등록.
 - 스켈레톤 UI를 통한 스크롤 문제 해결

주기적인 코드 개선

- vuex 구조 개편 (기능별로 별도의 파일로 분리)
- 공통 로직 분리
- 화면 렌더링 문제 개선(if, prop, :key) 이용
- socket subscribe 로직 개선

김두리

1. iOS 전체 앱 개발
 - a. 로그인/회원가입
 - b. 친구 기능
 - c. 커뮤니티 **CRUD**
 - d. 채널 **CRUD**
 - e. 커뮤니티, 채널 순서 바꾸기
 - f. 초대 기능
 - g. 프로필 편집
2. 채팅
 - a. **stomp** 프로토콜 이용
 - b. 실시간 **Websocket** 통신
3. **WebRTC** SDK 기반 미디어 서버 통신
4. 상태 관리
5. **FCM** 연동

김두리

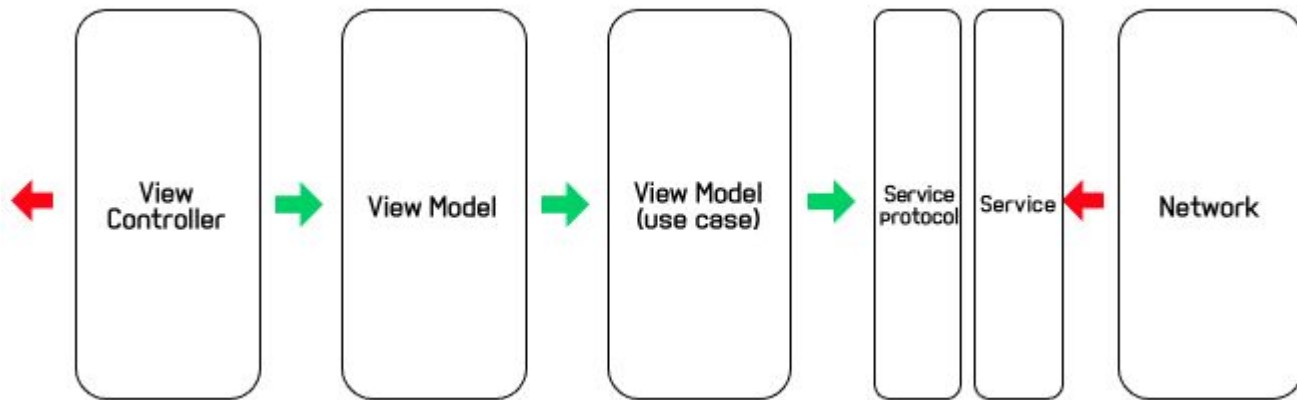
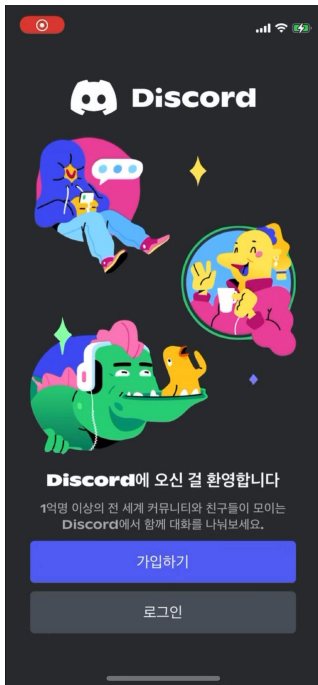
아키텍처에 대한 고민

- [?] 우리 프로젝트는
 - 실시간 데이터 처리
 - 채팅
 - 상태
 - 중복되는 비즈니스 로직이 존재
 - 채널/카테고리 별 **CRUD**
 - 커뮤니티/개인정보 수정
 - etc ...

김두리 개인 - 1

1. 유지보수 하기 쉬운 설계에 대한 통찰력을 갖춘 상태
 - 적용 패턴 : MVVM
 - MVVM ? 내 프로젝트에서는 어떻게 적용했는데?
 - a. View/ViewController
 - i. bind
 - b. ViewModel
 - i. input
 - ii. output
 - c. Model

김두리 개인 - 아키텍처 설계 : MVVM



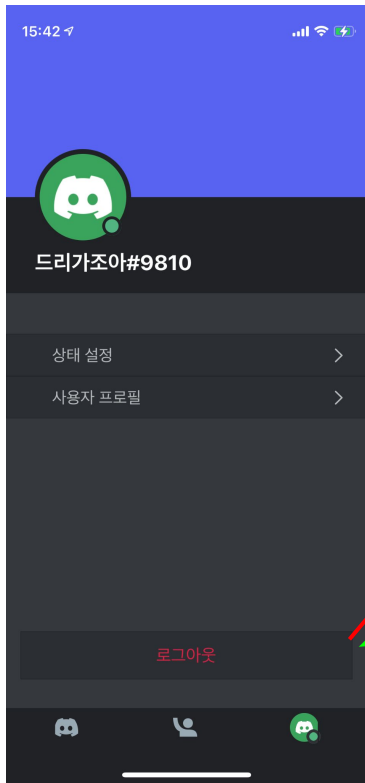
김두리 개인 - 1

스킵 될 수도 있음

1. MVVM

- 뷰는 ui를 그리는데만 집중하고, 뷰 모델은 비즈니스 로직에만 집중!
 - output 값 기반으로, 화면에 어떻게 그려질지 뷰 모델만 따로 테스트 가능해짐

But, ViewModel <-> data layer의 의존도 증가



```
struct Input {  
    let fetch = PublishSubject<Void>()  
    let tapLogoutButton = PublishSubject<Void>()  
}
```

```
struct Output {  
    let user = PublishRelay<User>()  
    let goToLoginHome = PublishRelay<Void>()  
}  
  
struct Model {  
    var user: User?  
}
```

김두리 개인 - 1

스킵 될 수도 있음

1. MVVM

- 네트워크 요청 값에 따라 뷰 모델(domain layer) 이 더 **tastable**해야 함
- 데이터에 대한 실제 구현은 **Data Layer(Service)**에 있어서, 의존성의 방향을 내부로 향하도록 유지하면서도 서비스에 요청을 보내도록
 - vm이 호출하는 프로토콜의 구현체를 네트워크를 사용하지 않아도, 단독으로 테스트 가능해짐



```
protocol UserServiceProtocol {
    // MARK: POST
    func signIn(email: String, password: String, deviceToken: String, _ completion: @escaping (SignIn?, MoyaError?) -> Void)
}

struct UserService: Networkable, UserServiceProtocol {
    typealias Target = UserTarget

    func signIn(email: String, password: String, deviceToken: String, _ completion: @escaping (SignIn?, MoyaError?) -> Void) {
    }
}
```

김두리 개인 - 1

1. MVVM

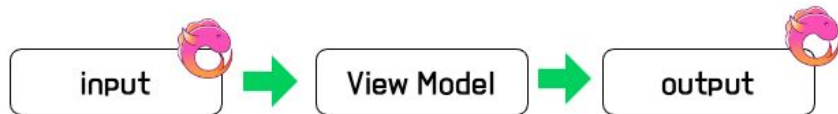
- data binding

- mvvm : ui를 view model에서 분리하기 위한 목적

그러면 ui 업데이트가 발생 했을 때, 어떻게 처리하는지?

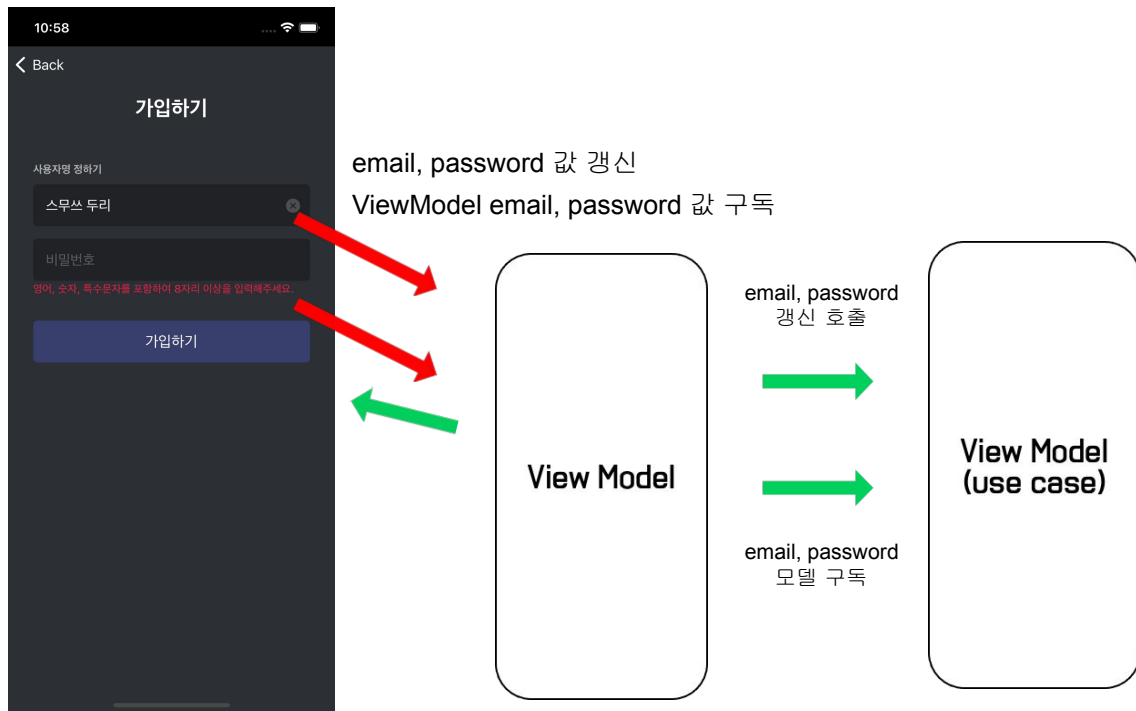
- Reactive 프로그래밍으로 해결!

- input / output 구조로 모든 이벤트를 정의하고, 방출



김두리 개인 - 1

- data binding



김두리 개인 - 1

2. Coordinator 패턴 도입

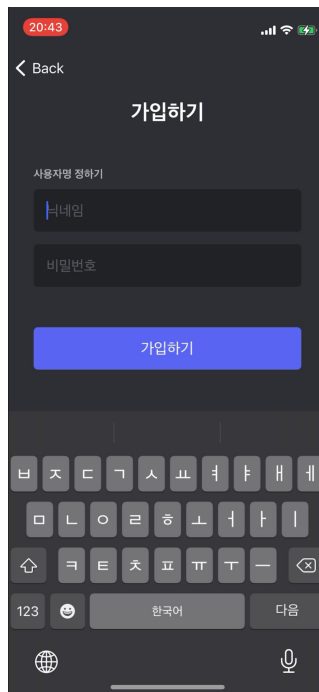
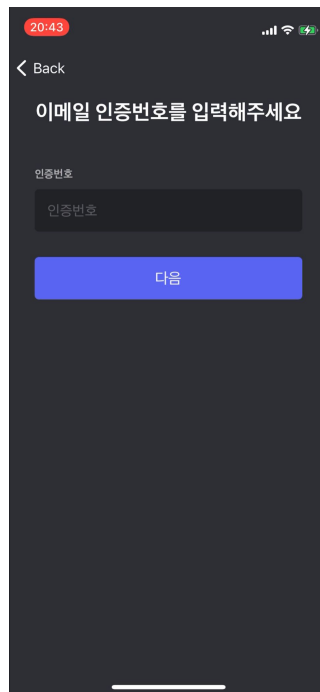
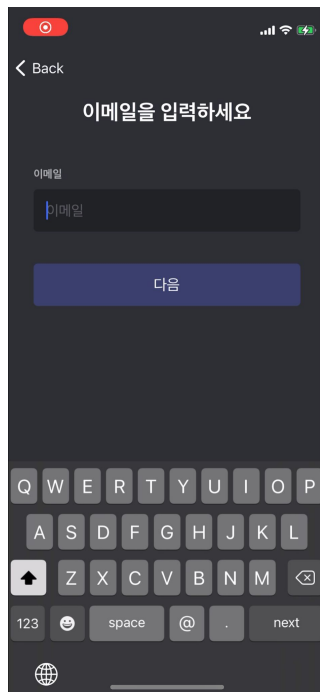
MVVM 형태로 계층 분리..

RxSwift 개념

Rx operator 학습..

바쁘다 바빠..!

회원가입 시 각 단계에서 설정된 값을 토대로

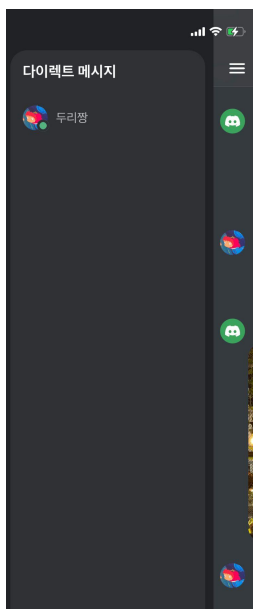


김두리 개인 - 1

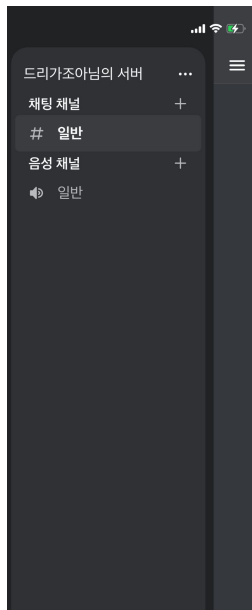
2. Coordinator (계층)



커뮤니티



direct room 정보

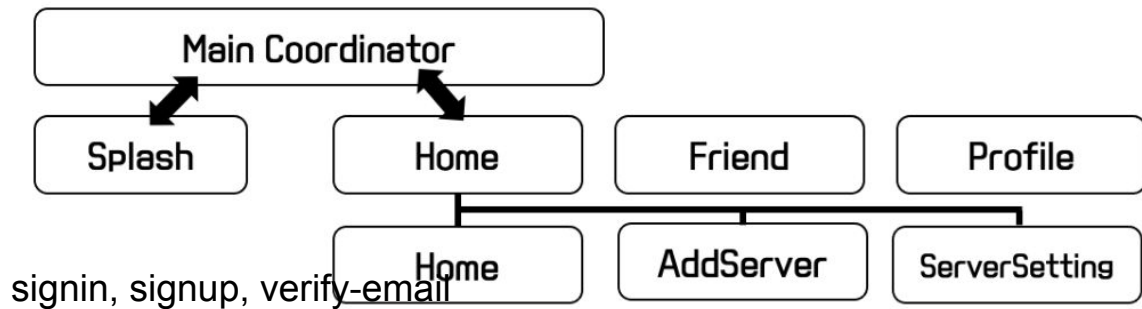


community의 channel 정보

- 채팅 화면을 보여주기 위해 커뮤니티/채널 정보 값이 상황에 맞게 전달 되어야 함
- 전달되는 값에 따라 HomeViewController에서 transition 처리
- 따라서 2개의 ViewController가 Home에 포함되어 있어야 함

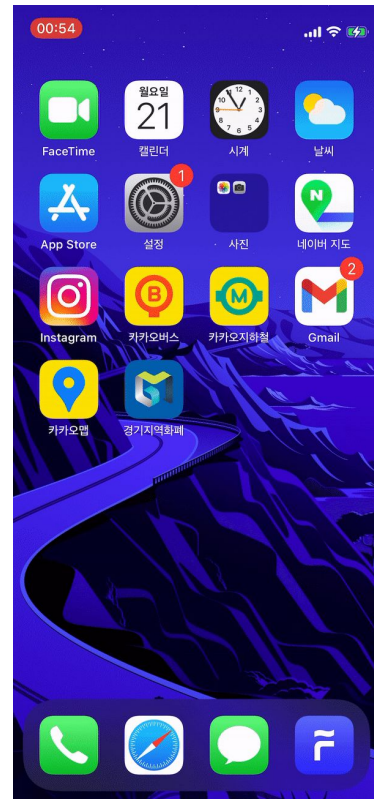
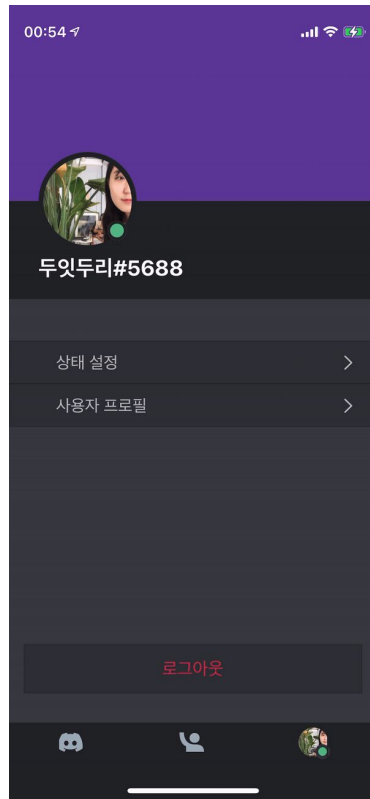
김두리 개인 - 1

2. Coordinator (계층)



김두리 개인 - 1

1. Coordinator



김두리 개인 - 2

2. RxSwift를 이해하고 상황에 맞는 비동기 코드 작성

- RxSwift?
 - Reactive extension + Swift의 합성어로, 비동기 프로그래밍을 관찰 가능한 흐름으로 지원하는 API
 - '한 가지 일을 처리하는 동시에 다른 일도 함께 처리하는 것'을 의미

In project.

- 비즈니스 로직의 input/output 구조 관리를 Reactive하게 처리
- ex. RxDataSource

김두리 개인 - 2

- 총 4가지의 cellType
- [point] 데이터에 따라 cellType에 맞는 cell을 binding
- [point2] data 변화에 따른 UI update
 - 서버 데이터
 - 유저 액션
 - 후처리

```
self.input.tapServer
    .bind { serverInfo in
        self.output.selectedChannel.accept(nil)

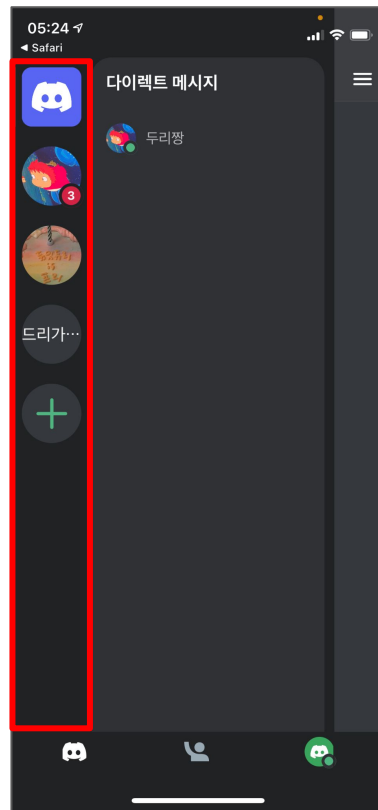
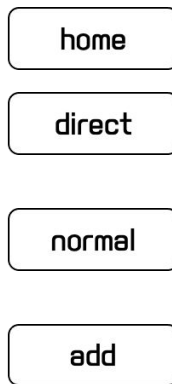
        switch serverInfo.1 {
        case .home: // MARK: 다이렉트 메시지 홈 + 다이렉트 메시지 함
            self.fetchDirect()
            self.model.selectedServerIndex = IndexPath(row: 0, section: 0)
            self.output.selectedServer.accept(self.model.selectedServerIndex)

        case .direct(let server): // MARK: 서버 리스트 (미수신 다렉 메시지)
            self.model.selectedServerIndex = serverInfo.0
            self.output.selectedServer.accept(self.model.selectedServerIndex)
            self.fetchDirectById(roomId: server.id)

        case .normal(let server): // MARK: 서버 리스트 (서버)
            self.model.selectedServerIndex = serverInfo.0
            self.output.selectedServer.accept(self.model.selectedServerIndex)

            self.fetchChannel(server: server)
            self.fetchMemebr(server: server)

        case .add: // MARK: 서버 추가 버튼
            self.model.selectedServerIndex = IndexPath(row: 0, section: 2)
            self.output.selectedServer.accept(serverInfo.0)
            self.output.goToAddServer.accept({})
        }
    }.disposed(by: disposeBag)
```



김두리 개인 목표

1. 유지보수 하기 쉬운 설계에 대한 통찰력을 갖춘 상태

[설계 단]

- [x] 주요 기능 단위 개발 시 컴포넌트 다이어그램을 먼저 설계함으로서 적절한 설계인가에 대해 점검한다.
- [x] 디자인 패턴의 경우, 선택한 디자인 패턴과 다른 디자인 패턴을 비교하여 프로젝트에 적절한 설계패턴임을 설명할 수 있다.

[코드 단]

- [x] 프로토크콜 지향 프로그래밍을 통해 다양한 상황에서 유연하게 대처할 수 있는 코드를 작성한다.
- [x] 스토리보드 사용을 지양하여, 재사용 가능한 UI 개발한다.
- [x] 자주 사용하는 컴포넌트는 미리 정의하여 사용한다.
- [x] 각 모듈(또는 기능)별로 폴더링을 구성하여, 프로젝트 폴더 구조를 직관적으로 이해할 수 있게 구성한다.

2. RxSwift를 이해하고 상황에 맞는 비동기 코드를 작성한다.

- 60% 달성, Menu <-> Home <-> Chatting ViewController간 데이터 전달(delegate)을 RxStream을 사용하여 개선 필요

마일스톤

<web>

마일스톤 1

1. 로그인/회원가입 (달성)
2. 친구 관리 (달성)
3. 커뮤니티 편집
4. 커뮤니티 채팅
5. 실시간 통화 기본 기능

마일스톤2

1. 실시간 통화 추가 기능
2. 커뮤니티 초대
3. DM
4. 채팅, 음성, 친구 상태 관리
5. 실시간 메시지 알람

<iOS>

마일스톤 1

1. 로그인/회원가입 (달성)
2. 친구 관리 (달성)
3. 커뮤니티 편집
4. 커뮤니티 채팅
5. WebRTC 기본 이해

마일스톤2

1. 커뮤니티 초대
2. DM 채팅
3. kurento media server 통신
4. 실시간 메시지 알람

프로젝트 목표 달성 어필

3분

web rtc 완료

채팅 완료

상태 관리 완료

1년 뒤에 봐도 부끄럽지 않은 코드를 작성하자!

다른 사람들이 이 프로젝트를 잡아도 금방 개발할 수 있을 만큼 쉽게 이해할 수 있는 코드이다.

이 코드를 작성하기 위해서는 해당 아키텍처의 전체적인 흐름을 읽을 수 있어야 한다.

모든 기술 스택에는 '왜?'를 생각하는 습관을 들이자!

왜 쓰는지 아는 것은 그 기술 스택이 어떻게 작동하는지 아는 상태이다.

그 기술 스택을 작동 원리부터 공부하여 제대로 파악하는 것이 중요하다.

회고

박병찬

김희동

김두리

- 좋았던 점
 - 프로젝트에 적절한 아키텍처가 무엇인지 고민을 통해, 나만의 소신이 생김
 - 좋은 패턴이나 기술에 대한 맹신 보다는, 프로젝트에 적절한 기술이 무엇인가에 대해 고려하는 시각이 생김
 - 코드의 재사용성을 높이기 위해 다양한 시도를 할 수 있었음
- 아쉬운 점
 - Rx 러닝커브로 인해, 프로젝트 기간 내 다양한 시도를 못해본 점
- 김민지
- 좋았던 점 : 새로운 도전을 할 수 있는 ‘힘’을 길렀다. 상태관리 라이브러리를 통해 데이터를 효과적으로 관리할 수 있었다. 서비스가 어떻게 만들어지는지 흐름을 경험할 수 있었다.
- 아쉬운 점 : 처음 도전하고 싶었던 **typescript**는 사용하지 못했다.

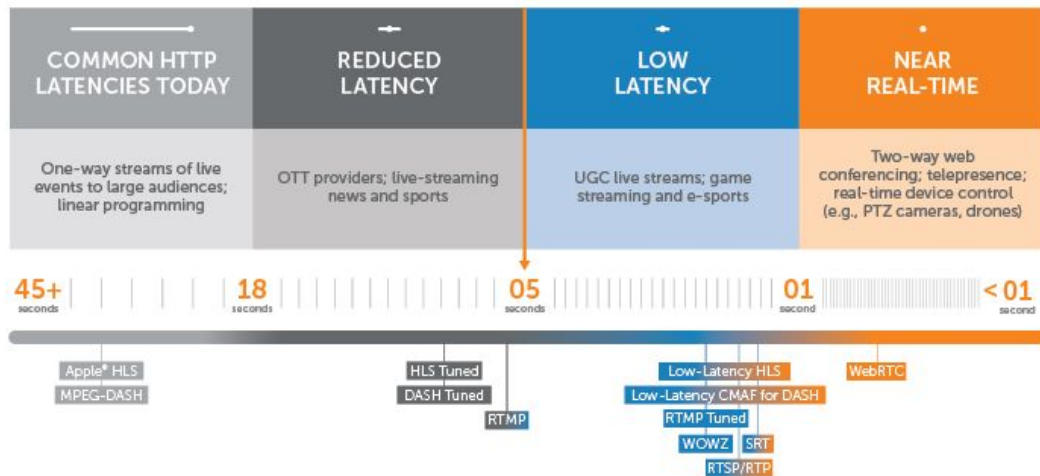
Appendix

-

김희동 개인

왜 WebRTC?

- 실시간 영상 채팅 서비스에서 중요한 것은 낮은 지연 시간(latency)이다.
- 화상 회의를 경험해보면 지연 시간이 크면 클
- 따라서 실시간 영상 채팅 서비스에서 중요한
- latency가 2초 이상이면 사용자가 불편함을 느낀다
- 따라서 latency가 낮은 프로토콜을 활용하자



김희동 개인

http / https 이슈

김희동 개인

ec2로 배포 시 브라우저에서 웹소켓세션이 끊어지는 현상 - sockjs heartbeat - socket.io와 비교

김희동 개인

웹뷰 관련 - video 태그 내 playsInline - firebase sdk 포함된 경우 실행되지 않는 현상 -> firebase sdk 미포함 버전
빌드 후 서브 도메인에서 모바일 버전 배포

김희동 개인

클린 코드 및 코드 분석 - sonarqube

성능테스트 - ngrinder - jmeter - webrtc-internels

김민지

#스크롤 이슈

처음 채팅방 입장 후 메시지 내역 수신시, 이미지가 포함되어 있으면 스크롤바가 맨 밑으로 내려가지 않는 현상.

=> 채팅 데이터를 받더라도, 이미지는 별도 네트워크 통신이 추가로 발생한다는 코드 리뷰를 받고, 이미지 로딩이 완료될 때까지 스켈레톤 이미지를 미리 보여줌으로써 문제해결

김두리 개인

- WebRTC 적용기 (회고)

[step]


1. kurento client (for iOS) 라이브러리 시도 및 적용
2. (pure) WebRTC(for iOS) 라이브러리 시도 및 적용
3. WKWebView 적용

김두리 개인

- WebRTC 적용기 (회고)

1. kurento client (for iOS) 라이브러리 시도 및 적용 (<https://github.com/nubomediaTI/Kurento-iOS>)

- [왜?] 선택하게 되었는지

- toolBox 형태의 SDK가 파일로서 배포 및 관리 되어 지고 있음 -> 자유로운 커스텀에 대한 기대 
- other client(web)과의 function 및 로직 공유 할 수 있을 것이라는 기대
- SFU 방식 호환 (다중 피어 연결 및 관리를 PeerClient 개념에서 지원)

- 실패 및 시도 철회

- object-c로 배포 됨 -> 우리 미디어서버에 맞게 커스텀 하기 위한 난이도 및 리소스 증가
- kms 바탕으로 되어 있어서, 시그널링 메시지 등 커스텀해야 하는 point 증가 및 관리의 어려움
- 기존 pod lib install 시 잔버그 존재 및 결정적으로 프로젝트 dependency path를 못 잡는 이슈 발생(해결 못함)
- 16년도 이후 관리되지 않는 오픈소스

→ [step 1]에 대한 결론 : kurento client 라이브러리도 **WebRTC** 기술 기반으로 되어 있고, 메커니즘을 이해했으니 직접 우리 서버에 맞는 SDK를 구성해보자!

김두리 개인

- WebRTC 적용기 (회고)

2. (pure) WebRTC(for iOS) 라이브러리 시도 및 적용 (<https://github.com/stasel/WebRTC-iOS>)

- 실제 작업 진행한 demo code : <https://github.com/stove-smooth/WebRTC-iOS>
- [왜?] 선택하게 되었는지
 - 강력한 Demo app (WebRTC 기술 구현을 하기 위해 확인 해야 할 요소들(ice candidate, sdp ..)을 interface에서 쉽게 확인 가능
 - swift로 되어 있는 WebRTC 관련 라이브러리 중 github Start 1위
- 실패 및 시도 철회
 - 다중 피어를 관리하는 것까지 성공하였지만, 각 Peer별 Video track을 보여줄 때 UI update 이슈 발생 (해결 못함)
 - 여러 사용자를 보여줘야 하기 때문에 collection / binding 구조 채택
 - collection view cell에 바인딩 되는 것까지 확인하였지만, 최초 1번만 rendering -> 이후 멈춤 현상 발생

→ [step 2]에 대한 결론 : Media Track 또는 Stream에 대한 이해가 필요한 상태라고 판단. 기능 구현을 위해 프로젝트 일정 상 WebView 선택이 불가피 하다고 내부 협의

김두리 개인

- WebRTC 적용기 (회고)

3. WKWebView 적용

- [왜?] 선택하게 되었는지
 - 여러 **WebView**가 있지만, 자유로운 커스텀 가능
 - **web <-> iOS** 간 자유로운 커뮤니케이션 가능
- 실패 및 시도 철회
-

→ [step 3]에 대한 결론 및 : **WebRTC**를 웹뷰로 연결 하고자 할 때 **WebView**에게 기대한 것들을 확실하게 **frontend** 단에서 협의가 이뤄져야함