**모바일 정보통신 종합설계 보고서**

* **과제 제목: WebSocket을 이용한 어플 개발**

팀명: 가이드

팀원: 김민성 박상철 지승우

**[목차]**

**1. 과제명**

**2. 팀원 조직**

**클라이언트 백앤드**

**3. 과제 목표**

**4. 과제 개요**

**5. 과제 설계 도면**

**6. 설계된 프로그램**

**(구성 프로그램을 정리하여 설명하고, 프로그램 코드 자체는 첨부에 기술)**

**7. 완성 과제 Demo**

**8. 과제 목표달성 성과**

**9. 결론**

**\*첨부: 프로그램 코드**

이상 첨부 프로그램 코드를 제외하고 15페이지 내외 분량

1. **과제명**

본 과제의 명칭은 “가톨릭대학교내 아나바다운동”이라 한다.

1. **팀원 조직**

김민성 -

지승우 -

박상철 –

1. **과제 목표**
   1. 최근 2년간의 코로나로 인해 학교생활을 불편하게 했을 학우들을 위해 만든 어플로, 주변인들의 도움이 필요하지만 아직 친구를 사귀지 못해서 도움이 필요한 상황을 빠르게 대처할 수 있도록 만든 어플입니다.
   2. 회원가입 로그인 게시물등록 등의 API 설계 구현

HTTP protocol을 통한 설계한 API 사용

WebSocket protocol을 통해 빠른 실시간 채팅구현

1. **과제 개요**
   1. 프로젝트 추진배경
2. 가톨릭대학교 학생들만 이용할 수 있는 학생 기반 커뮤니티 어플 활성화
3. 가톨릭대학교내에서 도움을 요청하고 싶을 때 게시글을 이용해서 많은 사람들이 도와줄 수 있음
4. 가톨릭대학교내에서 필요한 것들을 채팅으로 인해 빠르게 얻을 수 있음.

예) 필요한 물품을 가져와야 했지만 가져오지 못했을 경우, 다른 사람에게 부탁할 수 있다.

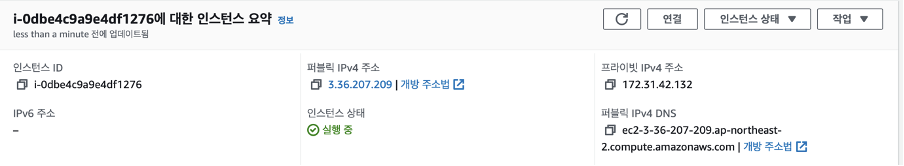
B. 프로젝트 진행기간: 2022.10.11 ~ 2022.12.04 (2개월)

C. 프로젝트 추진일정

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 단계 | 1주차 | 2주차 | 3주차 | 4주차 | 5주차 | 6주차 | 7주차 | 8주차 | 9주차 |
| 프로젝트 주제선정 및 기획 |  |  |  |  |  |  |  |  |  |
| 프로세스 기본설계 |  |  |  |  |  |  |  |  |  |
| 프로세스 상세설계 |  |  |  |  |  |  |  |  |  |
| 프로세스 통합 |  |  |  |  |  |  |  |  |  |
| 프로젝트 완료 |  |  |  |  |  |  |  |  |  |
| 프로젝트 문서화 |  |  |  |  |  |  |  |  |  |

**5. 과제 설계 도면**

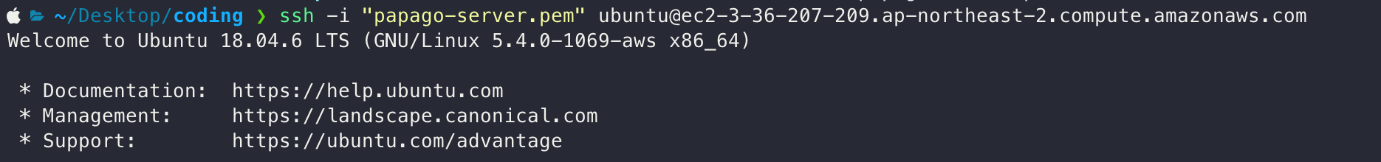
AWS (Amazon Web Services)



AWS (Amazon Web Services)는 아마존(Amazon)에서 제공하는 클라우드 서비스로, 네트워킹을 기반으로 가상 컴퓨터와 스토리지, 네트워크 인프라 등 다양한 서비스를 제공하고 있다.

AWS에서 Linux ubuntu 운영체제의 서버를 사서 서버를 만들었다.

ssh protocol을 통한 AWS 서버 접속

****

ssh protocol이란 네트워크 보안 도구 중 하나로 원격접속을 안전하게 할 수 있게 해주는 프로토콜이다. ssh protocol을 이용해 AWS서버에 쉽게 접속할 수 있다.

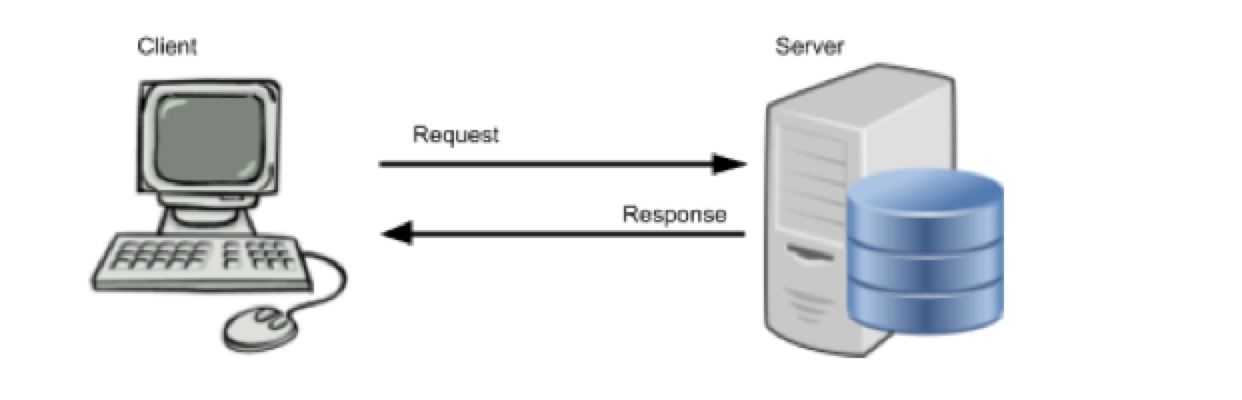
Java Spring framework + JPA를 통한 API 개발

Java Spring framework는 자바 플랫폼을 위한 오픈소스 애플리케이션 프레임워크로서 엔터프라이즈 급 애플리케이션을 개발하기 위한 모든 기능을 종합적으로 제공하는 경량화 된 솔루션이다. Java Spring은 자바 객체와 라이브러리들을 관리해주며, 톰캣과 같은 WAS 가 내장되어 있어 자바 웹 어플리케이션을 구동할 수 있다. 또한 JPA (Java Persistence API)는 자바 진영에서 ORM(Object-Relational Mapping) 기술 표준으로 사용되는 인터페이스의 모음이다. ORM이란 우리가 일반 적으로 알고 있는 애플리케이션 Class와 RDB(Relational Database)의 테이블을 매핑(연결)한다는 뜻이며, 기술적으로는 어플리케이션의 객체를 RDB 테이블에 자동으로 영속화 해주는 것이라고 보면 된다. JPA를 통해 서버 DB table에 연결하여 데이터를 CRUD(Create(생성), Read(읽기), Update(갱신), Delete(삭제))를 할 수 있다.

HTTP(Hypertext Transfer Protocol)와 WebSocket

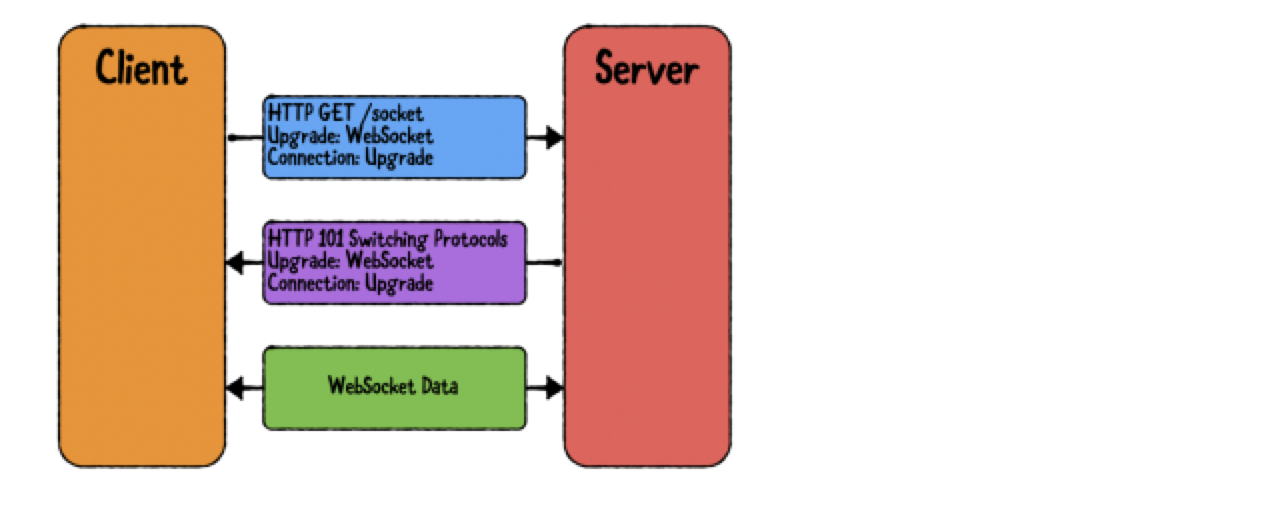
HTTP(Hypertext Transfer Protocol)는 인터넷상에서 데이터를 주고받기 위한 서버/클라이언트 모델을 따르는 프로토콜이다. 애플리케이션 레벨의 프로토콜로 TCP/IP 위에서 작동하며, HTTP는 어떤 종류의 데이터든지 전송할 수 있도록 설계돼 있다. 이러한 HTTP로 보낼 수 있는 데이터는 HTML 문서, 이미지, 동영상, 오디오, 텍스트 문서 등 여러 종류가 있고, 클라이언트가 요청을 보내면 서버가 응답하는 방식을 이용한다.

이러한 특징으로 로그인, 회원가입, 게시물 업로드, 정보창 띄우기 같은 부분의 통신을 구현하였다.



다만, 채팅을 이런 방식으로 구현하기에는 문제가 있다. HTTP는 불특정 다수를 대상으로 하는 서비스에 적합한 방식이나 요청-응답이 한 쌍을 이루는 구조로 응답이 끝나면 연결이 끊어지고, 단방향으로 통신하는 문제가 있다. 이런 문제를 해결하기 위해, 클라이언트가 서버에게 메시지 업데이트 요청하고, 서버가 클라이언트에게 최신 데이터 응답을 무한 반복하는 Polling 방식이나 클라이언트는 서버와의 연결을 유지하고, 서버는 클라이언트에게 단방향 event 메시지 보내는 SSE 방식, 그리고 이번에 사용한 Web Socket 방식 등이 있다.

채팅 프로그램에 Polling 방식을 사용하기엔 연결을 끊고 다시 연결하는 과정을 반복하는 것이 비경제적이기 때문에 채택하지 않았고, SSE 방식은 연결은 유지되지만, 서버에서의 단방향 통신에 적합하여 채택하지 않았고 Web Socket 방식을 채택하였다.



Web Socket은 서버와 클라이언트 간에 연결을 유지하고, 언제든 양방향 통신 또는 데이터 전송이 가능하도록 하는 기술이다. Web Socket은 HTTP를 기반으로 하고 있으며, TCP 방식을 이용한다. 그리고 클라이언트와 서버는 다음과 같은 순서로 연결을 한다.

1. 클라이언트는 서버에 http 프로토콜로 Handshake 요청을 함.

2. 서버는 응답 코드 101로 응답.

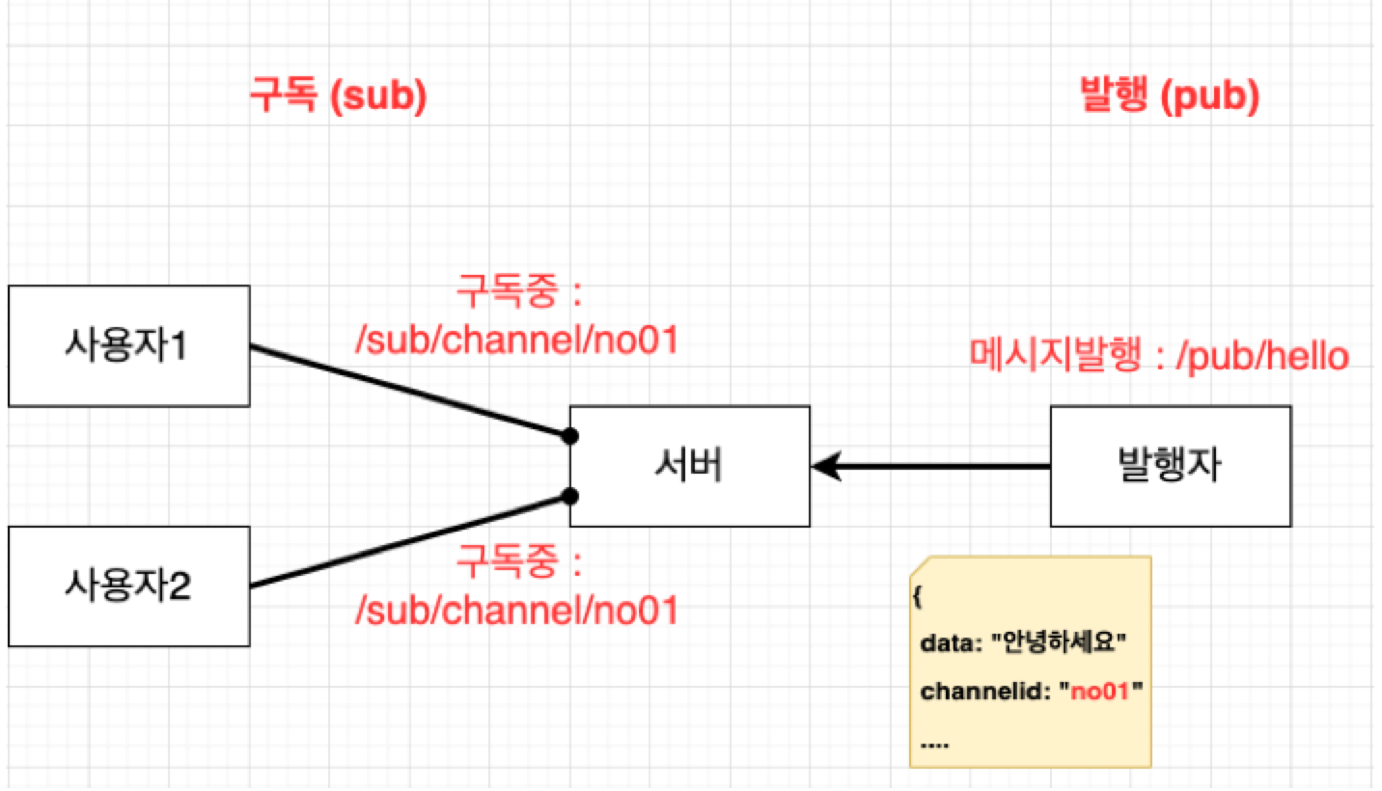
(이때, 포트는 HTTP에서 오픈한 포트를 변환하여 사용함.)

이러한 Web Socket은 주식, 게임, 메신저 등 클라이언트가 서버에게 요청하지 않아도 정보를 받을 수 있는 실시간 양방향 통신에 아주 적합하다.

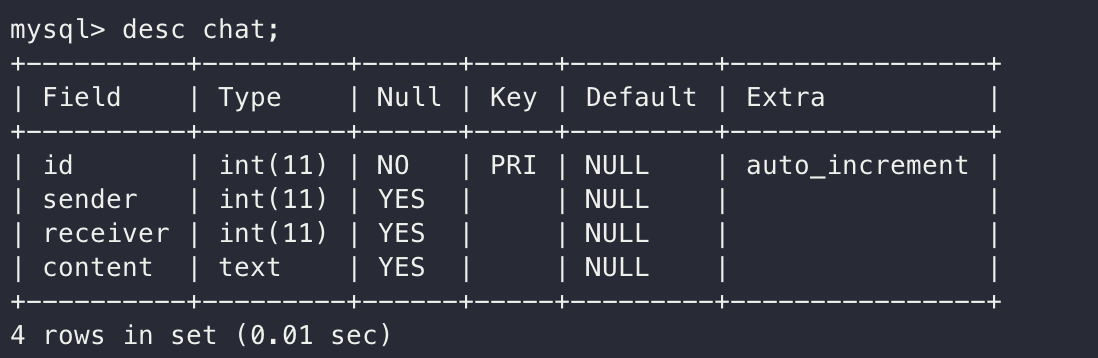
여기서 우리가 구현할 채팅 프로그램은 클라이언트가 다수이고 다중접속을 허용해야 하기 때문에 구독/발행(pub&sub) 방식을 사용하였다. 이는 다음과 같은 과정으로 수행된다.

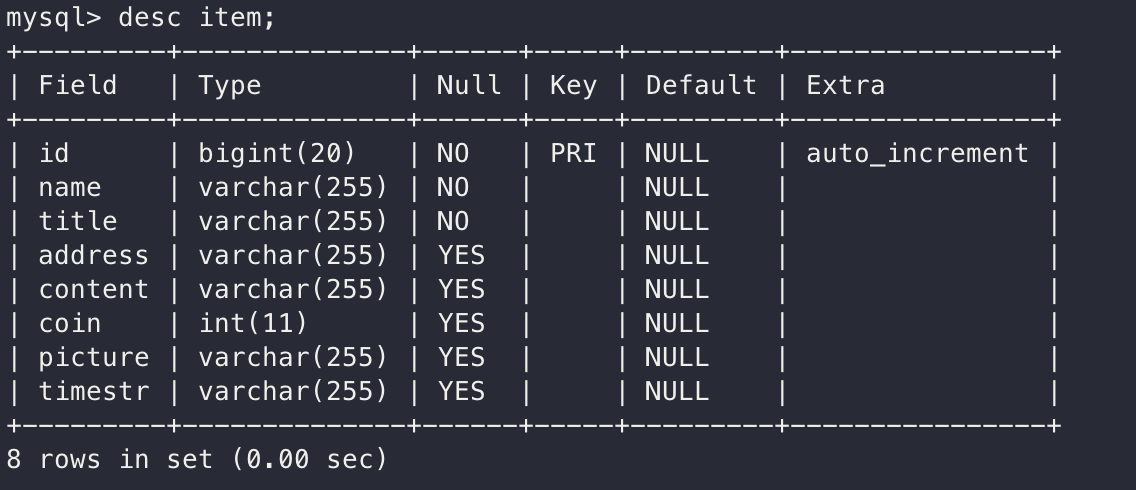
각 클라이언트는 특정 채널을 구독한다.

2. 발행자(클라이언트)는 서버를 거쳐서 특정 채널을 구독하는 클라이언트에게 메시지를 보낸다. (다른 채널을 구독한 클라이언트는 메시지를 받을 수 없고, 채팅 메신저에서 같은 채널을 구독한 사람끼리만 서로 채팅)



MySql

서버의 DB는 MySql을 통해 구현하였다. ****

****

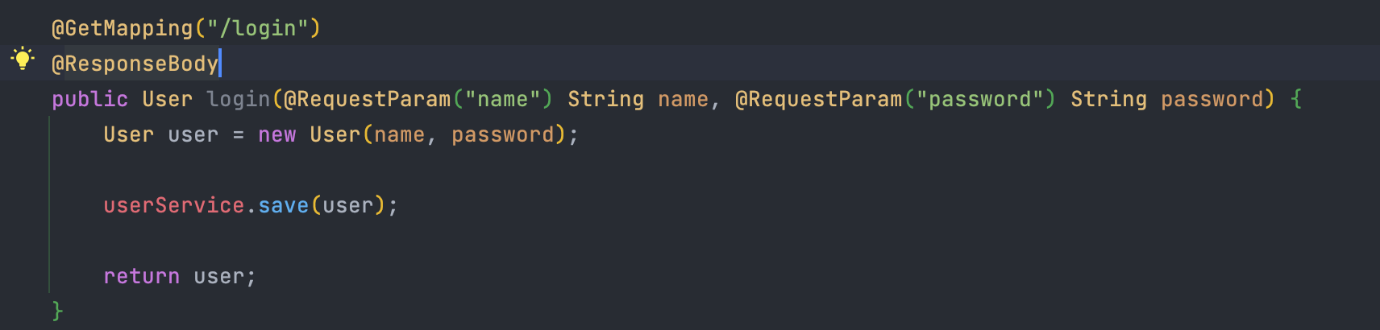
로그인 정보를 담기 위한 user table, 게시물 정보를 담기 위한 item table, 채팅 정보를 담기 위한 chat table을 만들어 데이터베이스를 설계하였다.

**6. 설계된 프로그램**

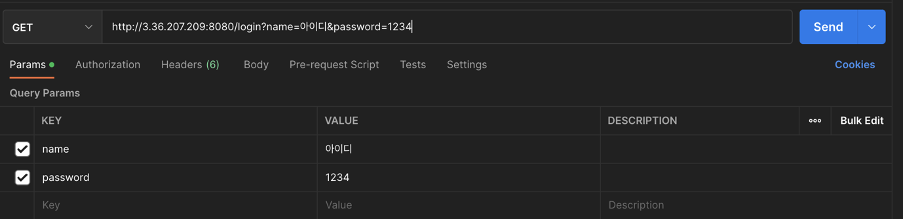
클라이언트와 서버는 서로 직접적으로 통신하기 보다는 API(Application Programming Interface)라 불리는 중간자 역할의 인터페이스를 사용한다. 예를 들어, 기상청의 소프트웨어 시스템에는 일일 기상 데이터가 들어 있다. 휴대폰의 날씨 앱은 API를 통해 이 시스템과 ‘대화’하여 휴대폰에 매일 최신 날씨 정보를 표시할 수 있다.

각각의 기능(회원가입, 로그인, 게시물등록)을 담당하는 API를 Java Spring을 통해 직접 설계하고 어플에서 특정 정보가 필요할 때 http protocol을 이용한 API요청을 통해 정보를 받아오는 방식으로 어플을 설계하였다.

**회원가입 API**



Java Spring framework를 통해 API를 간단히 설계할 수 있다. 요청 parameter로 name(id)과 password를 입력하여 name(id)가 중복되지 않은 경우 aws서버의 user table database에 저장되는 방식으로 설계를 하였다.



\***포스트맨**(Postman)은 개발자들이 API를 디자인하고 [빌드](https://ko.wikipedia.org/wiki/ìíí¸ì¨ì´_ë¹ë)하고 [테스트](https://ko.wikipedia.org/wiki/ìíí¸ì¨ì´_íì¤í¸)하고 반복하기 위한 [API](https://ko.wikipedia.org/wiki/API) 플랫폼이다.

Postman이라는 프로그램을 통해 설계한 API를 간단히 테스트해볼 수 있다.

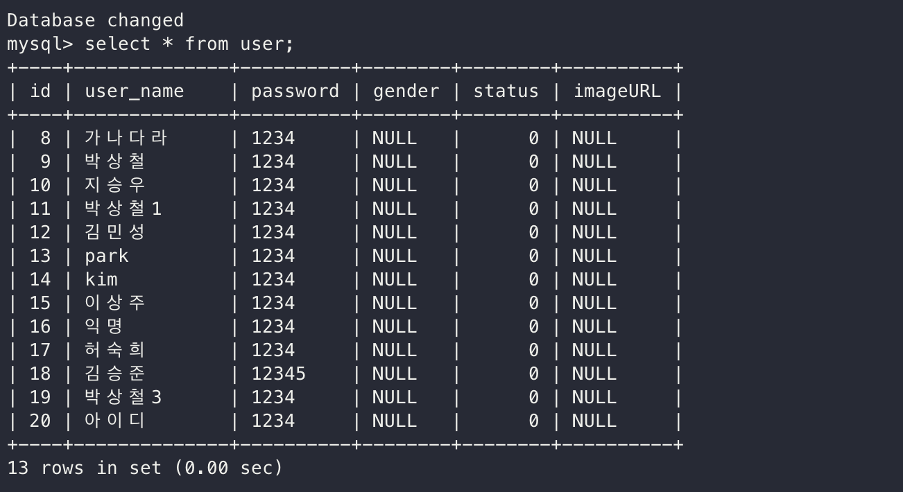
http://3.36.207.209:8080/login?name=아이디&password=1234

http protocol로 회원가입 API요청을 하였다. 여기서 3.36.207.209는 AWS 서버의 IP주소이고, 8080은 Webserver Process로 접근할 수 있는 port number이다. name 요청 parameter에 회원가입 하고 싶은 id를 입력하고 password 요청 parameter에 password를 입력하면 회원가입이 완료된다.

(i) name(id)가 중복이 아닌 경우



이미 회원 가입된 name(id)이 없이 회원가입이 정상적으로 잘 된 경우 user 객체를 json형식으로 반환한다.



또한 aws서버의 MySQL user table에 저장된다.



어플 에서는 volley 라이브러리를 통해서 http요청을 보내고 json객체가 정상적으로 반환되면 ‘가입되었습니다.’ 라는 메세지가 뜨면서 가입이 완료되게 되고 첫 화면으로 돌아가게 된다.

(ii) name(id)가 중복인 경우

로그인을 구현할 때 당연히 id는 고유한 값이여야 하고 중복이 허락되지 않는다. 그렇기 때문에 중복인 경우 "이미 존재하는 회원입니다." 라는 메세지와 함께 오류를 띄우도록 Java Spring 서버에 구현을 하였다.

@Override

public void save(User theUser) {

Optional<User> result = Optional.ofNullable(userRepository.findByUserName(theUser.getUserName()));

result.ifPresent(m -> {

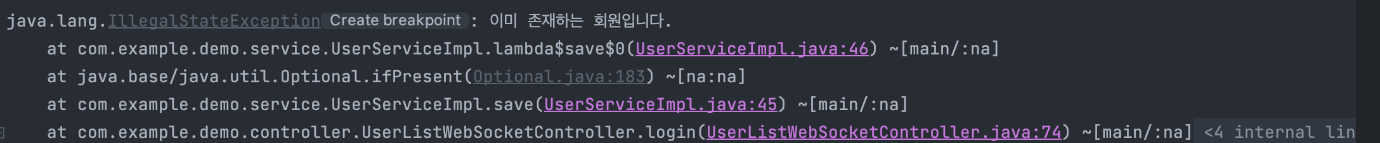
throw new IllegalStateException("이미 존재하는 회원입니다.");

});

userRepository.save(theUser);

}

http://3.36.207.209:8080/login?name=박상철&password=1234



이미 회원가입된 name(id)이 DB에 있는 경우 웹서버에서 에러를 띄운다는 것을 알 수 있다.



어플 에서는 ‘이미 가입된 아이디입니다.’ 라는 토스트 메세지를 띄우고 회원가입이 실패하게 된다.

**로그인 API**

다음은 로그인을 하기 위한 API이다. 로그인API도 회원가입API와 유사하게 설계하였다. name과 password를 http 요청을 보내면 user Database에서 name과 password를 확인한 후 정보가 정확히 일치하면 로그인 user의 객체를 json형식으로 반환하고, 일치하지 않는 경우 null값을 반환하도록 설계하였다.

@GetMapping("/login\_check")

@ResponseBody

public User login\_check(@RequestParam("name") String name, @RequestParam("password") String password) {

if (userService.login\_check(name, password)){

return userService.findByUserName(name);

}

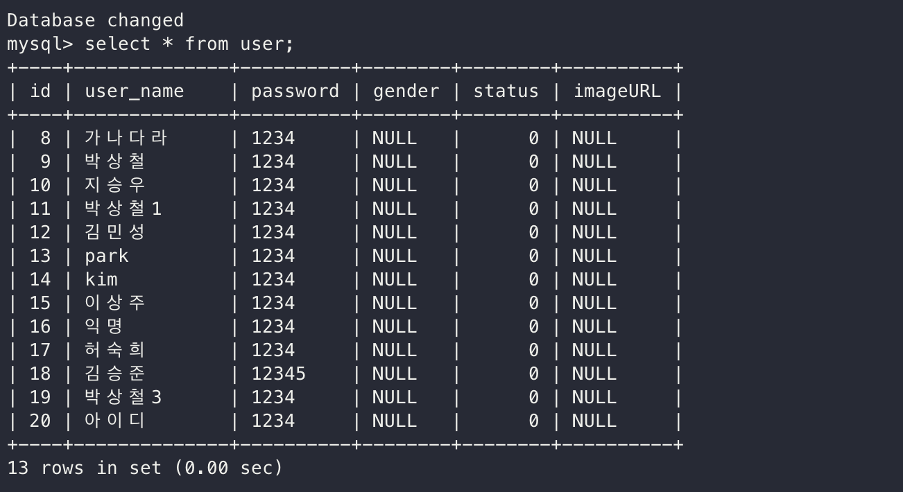
else {

return null;

}

}

(i) 로그인이 실패한 경우



http://3.36.207.209:8080/login\_check?name=아이디&password=12345



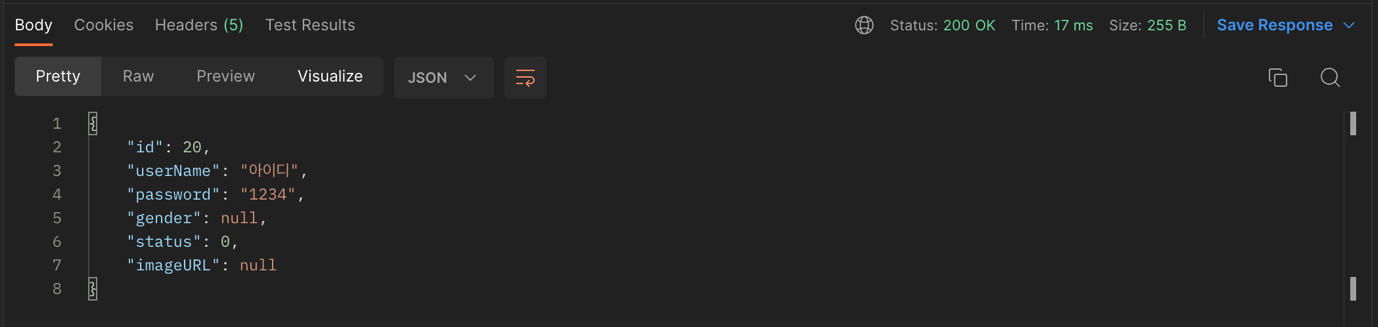
위의 db에 ‘박상철’ 이라는 name은 있지만 password가 1234이므로 위의 로그인 정보는 db와 일치하지 않는다. 이럴 경우에 null값을 반환하게 된다.



어플에서는 로그인 버튼을 누르면 아이디와 비밀번호를 요청parameter로 http요청을 보내고 반환 값이 null이면 ‘아이디나 비밀번호가 틀렸습니다.’ 라는 토스트 메세지를 띄우고 로그인이 실패하게 된다.

(ii) 로그인이 성공한 경우

http://3.36.207.209:8080/login\_check?name=아이디&password=1234



이번에는 로그인 정보를 정확하게 담아 http 요청을 했다면 요청에 맞는 user 객체가 json형식으로 반환되게 된다.



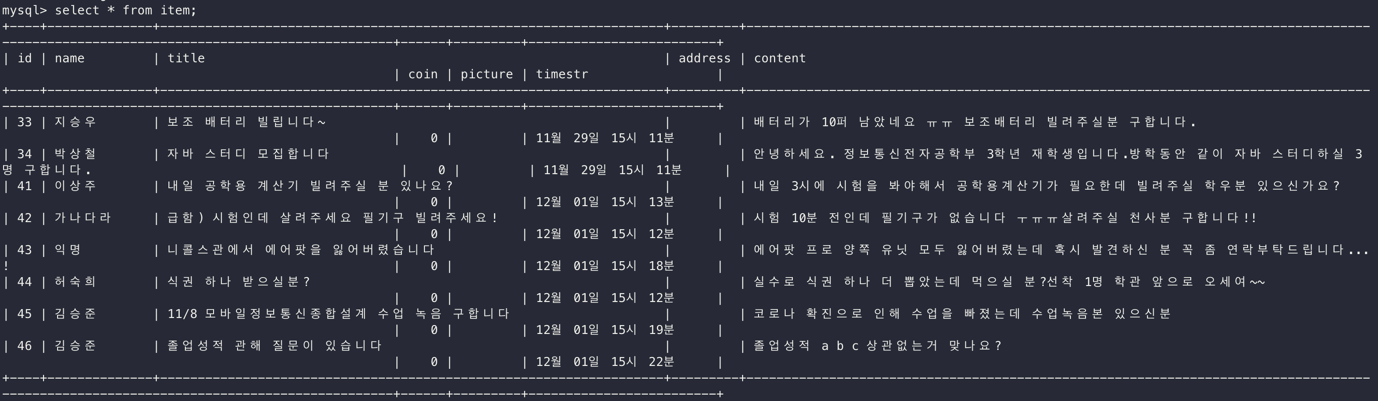
어플에서는 로그인 정보를 정확하게 담아 http요청을 보냈다면 ‘사용자님 로그인에 성공하셨습니다.’ 라는 토스트메세지를 띄우고 게시판 액티비티로 화면전환 되게 된다.

**게시물 작성 API**



이 어플은 게시물 작성을 하면 item table DB에 게시글이 저장되고, 저장된 item table DB를 가져와서 리싸이클러뷰 형태로 게시판에 표시되는 형식이다.

http://3.36.207.209:8080/item/add?name=김승준&title=졸업성적 관해 질문이 있습니다&content=~~~



이런식으로 http 요청을 보내면 서버 DB item table에 저장되게 된다.

**게시판 API**

게시판에는 item table에 저장되어 있는 데이터들을 가져와서 리싸이클러뷰 형태로 띄워야 한다.

그러기 위해서 서버에 http 요청을 보내서 item table에 있는 모든 데이터를 json형태로 받아와야 한다.

@GetMapping("/item/all")

@ResponseBody

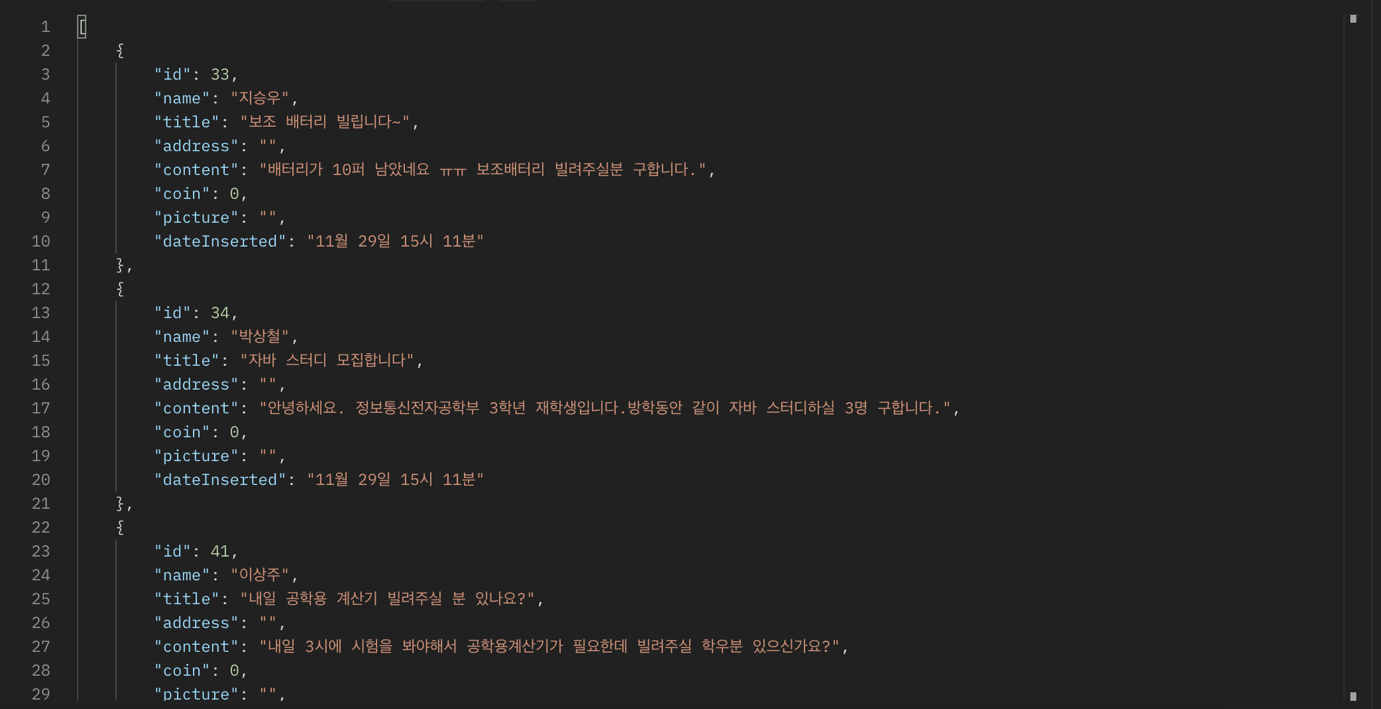
public List<Item> itemFindAll(){

return itemService.findItems();

}

http://서버ip:8080/item/all 을 입력하면 json형식으로 모든 Item을 반환하도록 java spring 서버를 통해 구현해 두었다.

http://3.36.207.209:8080/item/all



item/all로 http요청을 보내 모든 데이터를 가져오고 이 데이터를 리싸이클러뷰 형태로 띄우기만 하면된다.

JSONArray jsonItem = new JSONArray(response);

for (int i = jsonItem.length() - 1; i >= 0 ; i--) {

Gson gson = new Gson();

MainData mainData = gson.fromJson(jsonItem.get(i).toString(), MainData.class);

items.add(mainData);

}

받아온 모든 json형태의 item을 gson형태로 변경하였다. gson은 json형태로 받아온 데이터를 java 객체로 변환해준다. gson형태로 바꿔 adapter에 데이터를 넣어주면 리싸이클러뷰에 들어가게 된다.

## WebSocket을 이용한 채팅 구현

@MessageMapping("/userList/status/listen")

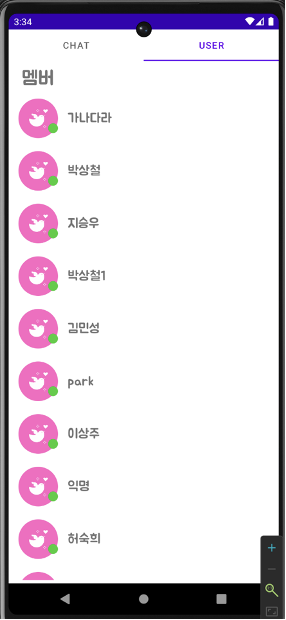
@SendTo("/topics/userList")

public List<User> subscribeToUserList() {

return userService.findAll();

}

Websocket protocol을 통해 모든 유저의 객체를 List에 담아서 보내줄 수 있다.



모든 유저의 리스트를 가져오고 리스트를 클릭하면 채팅을 할 수 있다.

@MessageMapping("/userList/chat/{idString}/listen")

@SendTo("/topics/chatList/{idString}")

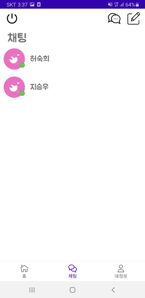
public List<User> subscribeToChatUserList(@DestinationVariable String idString) {

int id = Integer.parseInt(idString);

return chatService.findConnectUser(id);

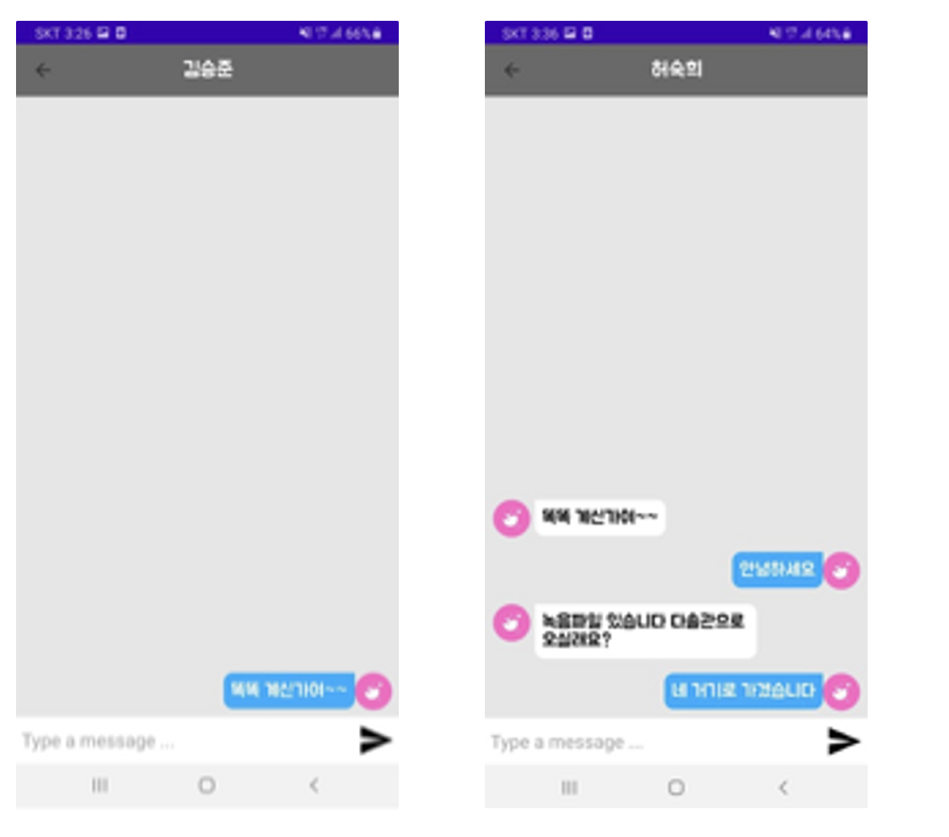
}

이번에는 WebSocket protocol을 통해 나와 한번이라도 채팅을 한 사람의 리스트를 가져온다. 만약 ‘허숙희’라는 사람이 나에게 채팅을 보냈다면 채팅 리스트에 ‘허숙희’가 추가되고 채팅이 왔다는 것을 알 수 있다.

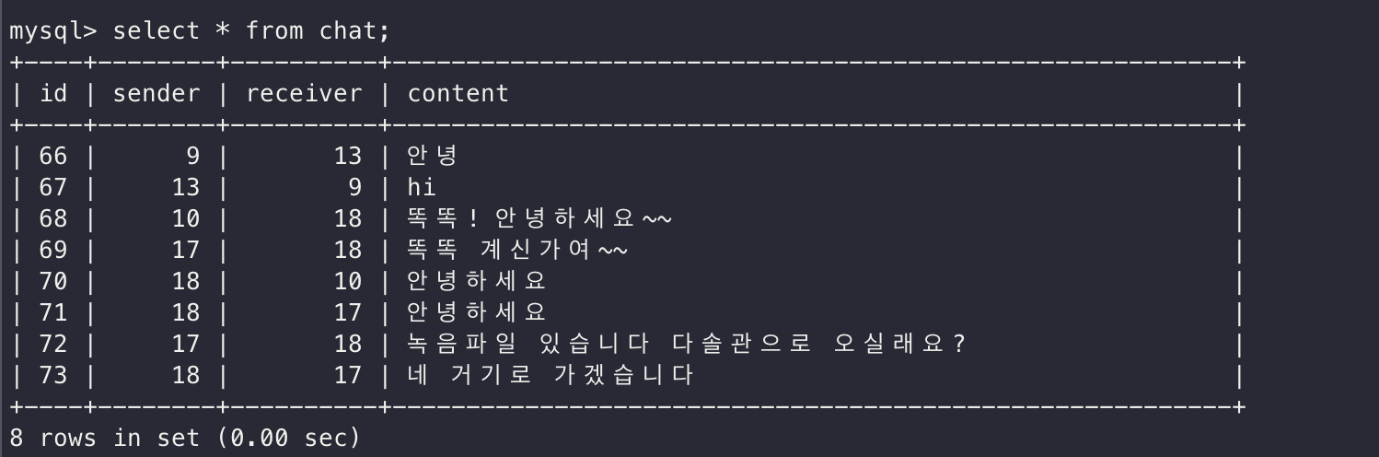


### 채팅

리스트를 클릭하면 그 사람과 채팅을 할 수 있게 되고 채팅 정보는 chat table에 저장되게 된다.



김승준이라는 유저와 허숙희라는 유저는 WebSocket protocol을 통해 아주 빠르게 실시간 채팅을 할 수 있다.

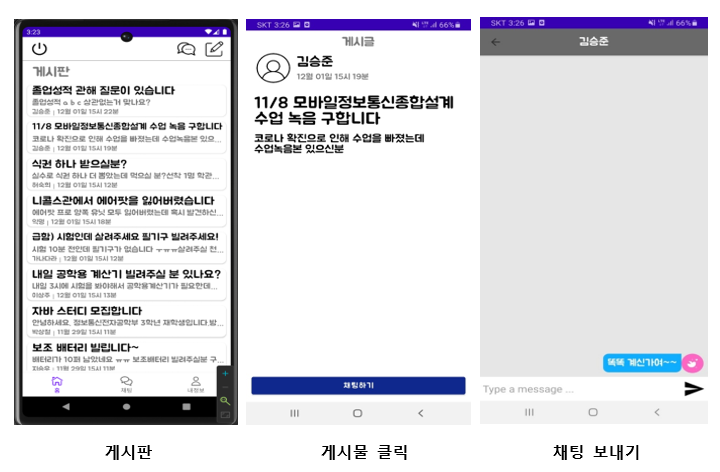


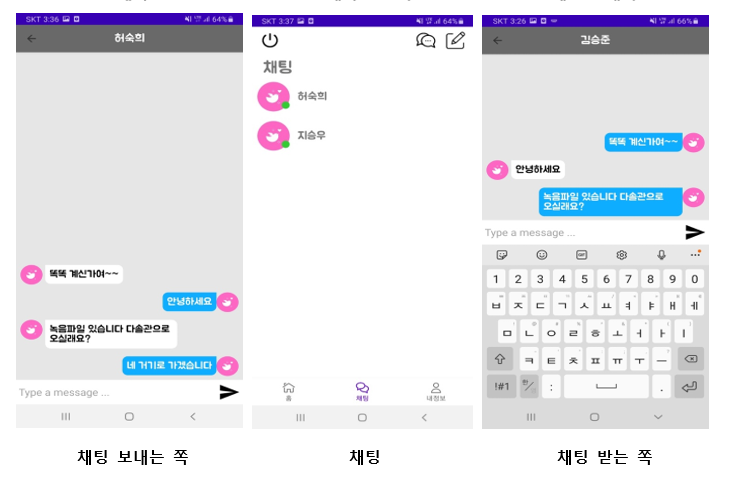
서버의 DB의 chat table에 잘 저장되어 있다는 것을 확인할 수 있다.

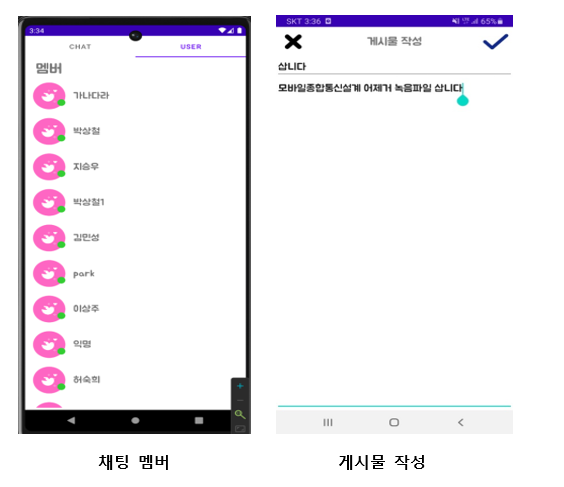
**(구성 프로그램을 정리하여 설명하고, 프로그램 코드 자체는 첨부에 기술)**

**7. 완성 과제 Demo**

****

****

****

****

**8. 과제 목표달성 성과**

HTTP와 WebSocket을 이해하여, WebSocket으로 빠른 실시간 채팅구현으로 어플 개발

MySQL을 통해 서버DB 구현

**9. 결론**

결론적으로는 처음에 구상한데로 잘 실행되었다. 이 과목이 모바일정보통신종합설계인 점을 고려해서 이와 관련된 통신분야에서 끌어올릴 수 있는 대표적인 구성들을 잘 설계한 것 같다. 로그인과 회원가입은 문제없이 생성이 되고 게시글을 이용해 채팅을 할 수 있는 형태까지 잘 만들어졌다. 다만 아쉬운 점을 뽑자면 계획한 큰 구성들(로그인, 회원가입, 게시글, 채팅 등)을 멘토링을 이용해서 퀄리티를 조금 더 향상시킬 수 있지 않았을까 라는 아쉬움이 들었지만, 9주라는 시간안에 학우들끼리 모여서 만들었을 때 어느 정도까지 만들 수 있고 어떤 퀄리티까지 낼 수 있을지 확인하고 싶었다. 이메일을 통한 학교 인증으로 가톨릭대학교 학생들만 이용할 수 있게 하면 좋았겠지만 9주라는 시간안에 서버 구축, 채팅 구현 및 디자인, 세밀한 기능구현 등 다양한 구성들을 설계하고 나니 어느새 발표인 게 아쉬울 따름이었다**.**

**\*첨부: 프로그램 코드**

이상 첨부 프로그램 코드를 제외하고 15페이지 내외 분량