JAVA 백엔드 개발자
이준경 포트폴리오

이준경

연락처 010-8223-3782

이메일 worldreaming@gmail.com

깃허브 https://github.com/ljk1782

목차

프로필

ABOUT ME

이름: 이준경

생년월일: 1998.04.21

주소: 경기도 의정부시 신곡동

전화번호: 010-8223-3782

이메일: worldreaming@gmail.com

LICENSE

정보처리기사 (2023.06.09 취득)

컴퓨터활용능력 1급 (2023.05.26 취득)

EDUCATION

React.js와 Springboot를 활용한 자바 풀스택 개발자

- 2023.09.19 ~ 2024.02.29

한경대학교

- 응용수학과 전공
- 컴퓨터공학과 복수전공
- 학점 4.0 / 4.5
- 졸업 (2017.03 ~ 2023.08)

CHANNEL

깃허브: https://github.com/ljk1782

기술 스택

Back-End

Java, Spring Framework, Spring Boot
RESTful API(JSON), MyBatis, WebSocket

Front-End

HTML5, CSS3, JavaScript(ES6),
jQuery, React, WebSocket

DB

MySQL

OS

Linux (Ubuntu 20.04 LTS)

Server

Apache Tomcat
AWS EC2

Tools

IntelliJ IDEA, VS Code
Docker, FileZilla

Collaborations

GitHub, notion, Figma

수행 프로젝트 1/2

팀 프로젝트

그룹웨어 시스템

WittyWave

프로젝트 목차

팀 프로젝트 - WittyWave

프로젝트 소개

팀의 효율성을 높이기 위한 통합 그룹웨어

개발 기간 : 2023.12.27 ~ 2024.02.28 (8주)
개발 인원 : 풀스택 6명
맡은 역할
- BackEnd & FrontEnd 개발
- AWS EC2 + Docker 를 이용한 서버 배포

담당 기능
- 사내 메신저
- 캘린더
- 프로젝트 관리
- AWS EC2와 Docker를 이용한 서버 배포

프로젝트 GitHub 바로가기

팀 프로젝트 - WittyWave

사용 기술

Java 17

Spring Boot 3.2.1

Spring Data JPA

Spring Security

JWT

Web Socket

GitHub

Notion

Figma

MySQL

Docker

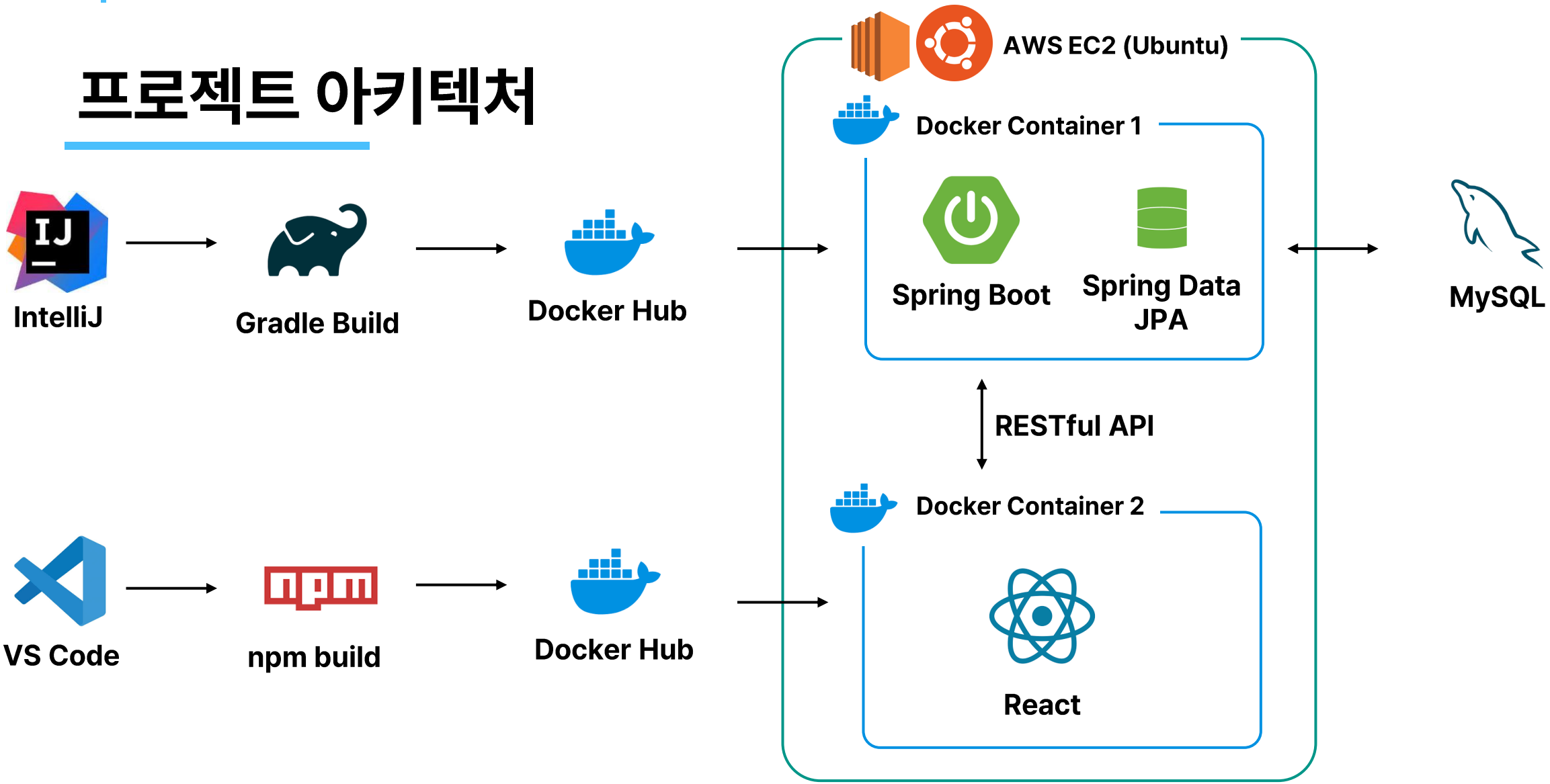
Ubuntu

AWS EC2

FileZilla

팀 프로젝트 - WittyWave

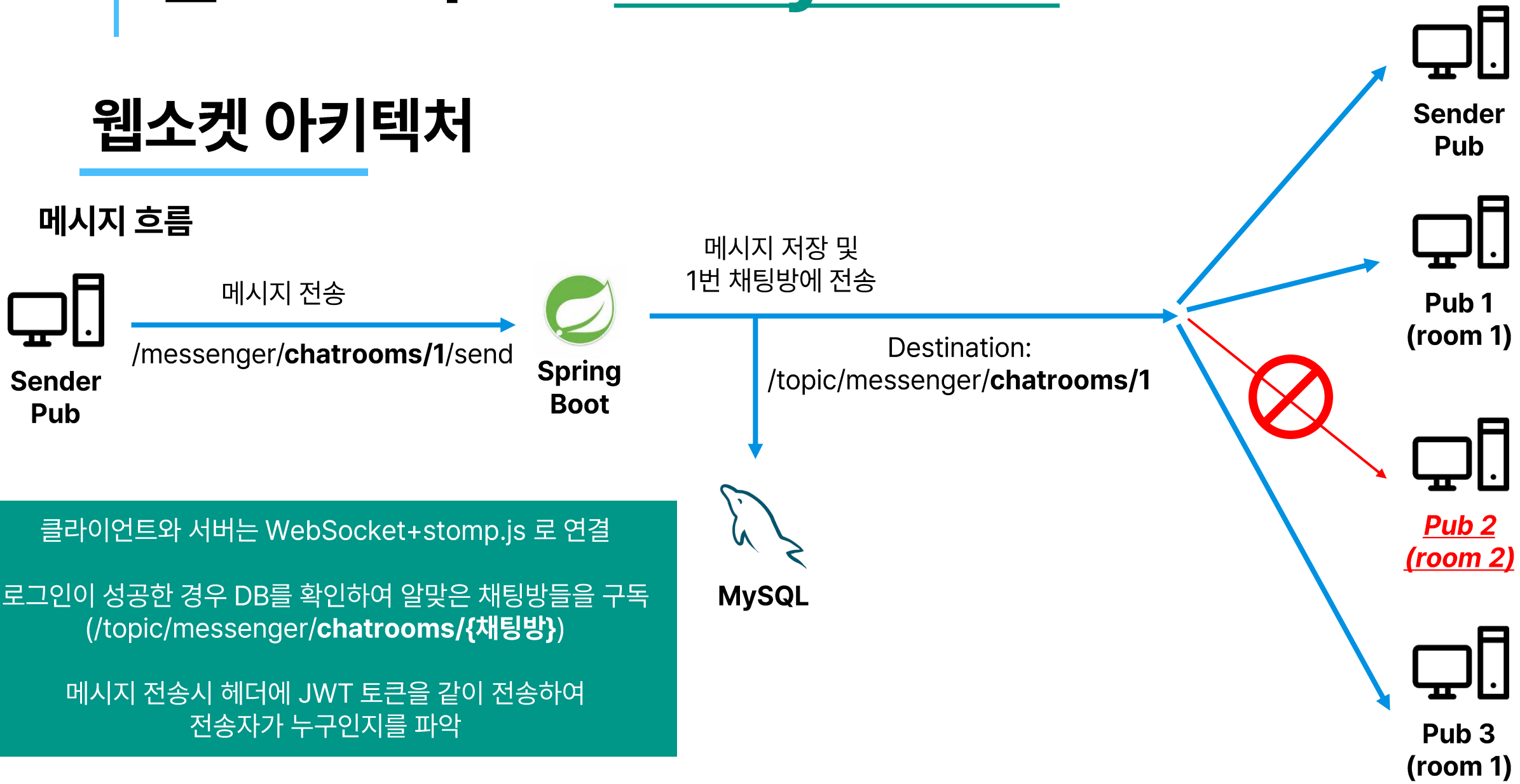
프로젝트 아키텍처



팀 프로젝트 - WittyWave

웹소켓 아키텍처

메시지 흐름



클라이언트와 서버는 WebSocket+stomp.js 로 연결

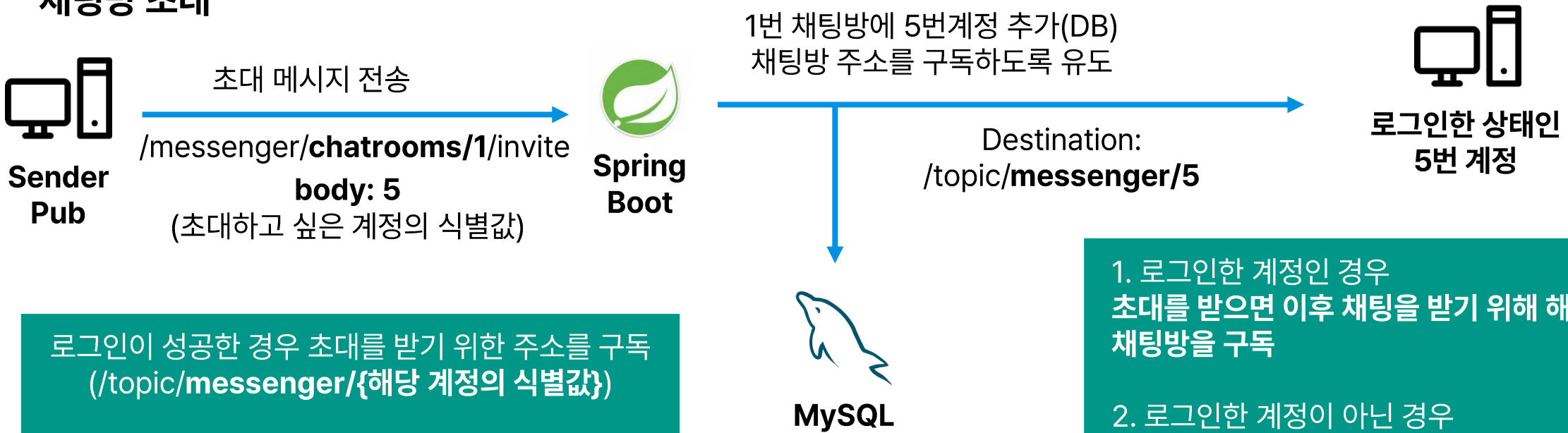
로그인이 성공한 경우 DB를 확인하여 알맞은 채팅방들을 구독
(/topic/messenger/**chatrooms/{채팅방}**)

메시지 전송시 헤더에 JWT 토큰을 같이 전송하여
전송자가 누구인지를 파악

팀 프로젝트 - WittyWave

웹소켓 아키텍처

채팅방 초대



로그인이 성공한 경우 초대를 받기 위한 주소를 구독
(/topic/**messenger/{해당 계정의 식별값}**)

초대 메시지 전송시 헤더에 JWT 토큰,
body에 초대하고 싶은 계정의 식별값을 전달

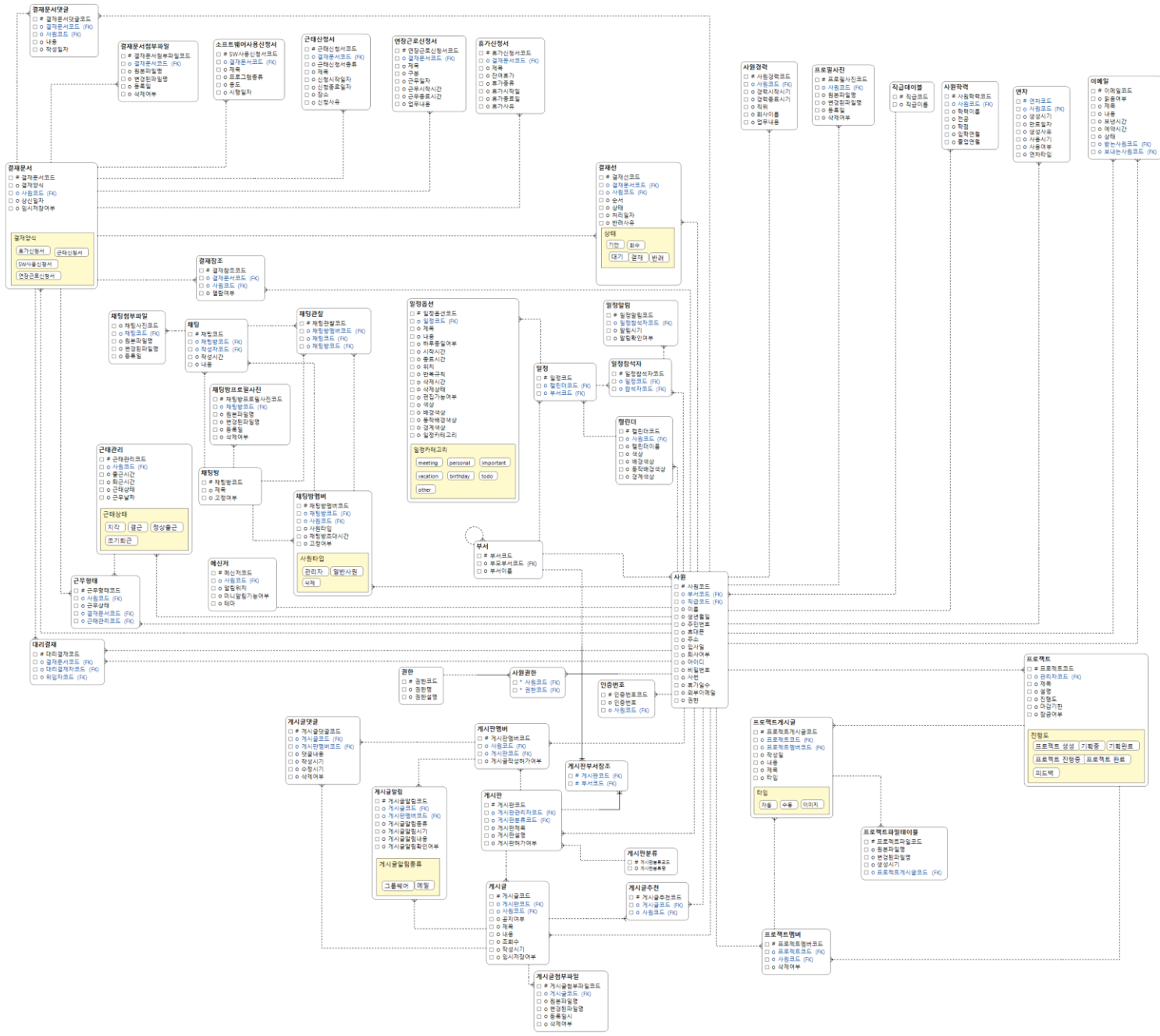
1. 로그인한 계정인 경우
초대를 받으면 이후 채팅을 받기 위해 해당 채팅방을 구독

2. 로그인한 계정이 아닌 경우
마찬가지 데이터베이스에는 내 계정의 채팅방 목록이 저장되어 있기 때문에 로그인 한 순간 적절한 채팅방을 구독

WittyWave

ERD (DB 설계)

원본이미지로 확인하기 <= 클릭



팀 프로젝트 - WittyWave

주요 담당 기능 – 채팅 전송

(MessengerWebsocketController)

```
@MessageMapping("/messenger/chatrooms/{chatroomCode}/send")
public void sendChat(
        @DestinationVariable String chatroomCode,
        @Payload SendDTO send,
        SimpMessageHeaderAccessor accessor) {
    String token = accessor.getFirstNativeHeader("Authorization");
    TokenUtils tokenUtils = new TokenUtils();
    if (token != null) {
        send.setEmployeeCode(tokenUtils.getUserEmployeeCode(token));
    } else {
        throw new TokenException("허가되지 않은 채팅 전송");
    }
    ChatDTO chatDTO = messengerService.sendChat(Long.parseLong(chatroomCode), send);

    String destination = "/topic/messenger/chatrooms/" + chatroomCode;
    messagingTemplate.convertAndSend(destination, chatDTO);
}
```

해당 MessageMapping 주소로 클라이언트는 메시지를 전송한다.
해당 주소는 채팅방마다 다른 이름을 가진다.
({chatroomCode}를 이용)
또한, @Payload 에는 전달받은 채팅정보가 있다.

적절한 계정의 요청인 경우 채팅을 DB에 저장하기 위해 Service의 메서드를 실행하고, destination 주소에 채팅정보를 전달한다.

현재 백엔드 서버는 모든 채팅 메시지를 단독으로 처리하고 있다. 이는 대규모 실시간 채팅이 발생하는 경우, 트래픽이 과도하게 증가할 위험이 있다.

만약 NoSQL 데이터베이스 서버를 별도로 구축한다면, 현재의 규모에서는 눈에 띄는 이점이 없을 수 있지만, 대규모 데이터와 트래픽을 처리해야 하는 상황에서는 매우 효과적인 대안이 될 것 같다.

팀 프로젝트 - WittyWave

주요 담당 기능 – 프로젝트 게시판 이미지 업로드 (1/2)

(ProjectService)

```
public ProjectPostFileDTO uploadImage(MultipartFile file) {
    try {
        LocalDateTime now = LocalDateTime.now();
        String imageName = UUID.randomUUID().toString().replace("-", "");
        String replaceFileName = null;
        try {
            replaceFileName = FileUploadUtils.saveFile(IMAGE_DIR, imageName, file);
        } catch (IOException e) {
            FileUploadUtils.deleteFile(IMAGE_DIR, replaceFileName);
            throw new DataUpdateException("이미지 업로드 실패");
        }
        ProjectPostFileDTO projectPostFileDTO = new ProjectPostFileDTO();
        projectPostFileDTO.setProjectPostFileOgFile(imageName);
        projectPostFileDTO.setProjectPostFileChangedFile(replaceFileName);
        projectPostFileDTO.setProjectPostFileCreationDate(Date.from(now.atZone(ZoneId.systemDefault()).toInstant()));
        return projectPostFileDTO;
    } catch (Exception e) {
        throw new DataUpdateException("이미지 업로드 실패");
    }
}
```

팀 프로젝트 - WittyWave

주요 담당 기능 – 프로젝트 게시판 이미지 업로드 (2/2)

게시판의 이미지는 클라이언트에서 업로드 해달라고 요청하면 일단 해당 이미지를 저장한다.
이름의 경우 중복이 될 가능성이 있기 때문에 UUID를 사용하여 랜덤 파일명을 생성하여 파일명 충돌을 방지한다.

파일명이 생성됐다면 해당 파일명 자체를 다시 클라이언트로 반환한다. 클라이언트는 이 파일명을 이용하여 설정했던 backend 주소를 통해 이미지를 출력한다. (예시: http://123.456.789.000/web-images/{랜덤파일명})

이후, 실제 파일명을 데이터베이스에 저장하는 것은 해당 이미지들이 포함된 게시글이 업로드 되는 경우이다.
이유는 해당 파일이 실제로 게시된 글에 있는 내용인지 아닌지에 따라 데이터 정리를 할지 말지 선택할 수 있기 때문이다.

이미지 파일의 저장과 실제 DB에 파일명이 저장되는 것은 시기의 차이가 있다.

1. **이미지 파일의 저장은 이미지 업로드 직후 바로 저장된다.**
2. **실제 DB에 파일명과 시간정보가 저장되는건 게시글이 업로드된 시기에 DB에 저장된다.**

위와 같이 할 경우, 나중에 DB에는 정보가 없으나 파일 자체가 저장되어 있는 경우가 발생한다.
이를 주기적인 클린업 작업으로 스케줄링하면 저장공간 최적화에 효과적일 것 같다.

팀 프로젝트 - WittyWave

회고 및 배운 점

성공적인 WebSocket 구현

WebSocket의 경우 상당히 애먹었다. 하지만 우선적으로 Demo 버전을 개발했고 이를 성공했기 때문에 이후 실제 프로젝트에 적용시키는 것은 크게 무리가 없었다고 생각한다.
(WebSocket 데모 – Backend << 바로가기)
(WebSocket 데모 – Frontend << 바로가기)
특히 너무나도 기분이 좋았던 이유는 흔하게 써왔던 API 주소를 이용한 단방향 연결 스트림이 아닌,
지금까지 사용한적 없던 연결을 지속하는 양방향 연결 스트림을 구현하고 이를 성공했다는 점이다.
물론 더 개선할 여지가 남아있지만, 앞으로 새로운 것을 공부할 때 이 경험을 기억한다면,
두려움 없이 즐거움만 가득할 것이라고 생각한다.

AWS EC2 와 Docker를 이용하여 서버 배포를 수행하다

이전까지 프로젝트에서는 로컬서버에서 실행하는 방식을 택했지만,
Front-end 프로젝트와 Back-end 프로젝트를 빌드해보고, 이 빌드 된 파일들을 Docker Hub에 올린 다음
AWS EC2로 각각 서버 배포를 해보면서 어느정도 서버 배포의 과정을 알 수 있었다.
이번 시도가 첫 시도였기 때문에 아직 문서를 여러 번 보면서 작업해야 하고,
현업 수준과 매우 다를 수 있지만, 한편으로는 앞으로 개발할 프로그램을
외부의 많은 사람들에게 보여줄 수 있다는 점이 이번 프로젝트에서 얻어간 제일 큰 성취라고 생각한다.

팀 프로젝트 - WittyWave

회고 및 배운 점

부족했던 커밋 컨벤션

이번 프로젝트를 통해 커밋 컨벤션의 중요성을 얻어갔다고 생각한다.
우리 팀원 모두 커밋메시지를 간단한 한 줄로 작성하는 경향이 있었는데,
이는 나중에 오류가 발생했을 경우 히스토리를 추적하는데 어려움을 겪게 됐다.
협업 과정에서의 커밋 메시지는 프로젝트 이해도를 높이고, 빠른 문제 해결에 기여하는 핵심 요소임을 인지하게 됐다.
앞으로 프로젝트를 시작하기 전과 진행하는 동안, 다음과 같은 점을 고려하고자 합니다.

1. 커밋 메시지를 위한 명확한 가이드라인 수립
2. 커밋 메시지의 질을 검토하고 지적하는 정기적인 리뷰
3. 커밋 메시지가 가이드라인을 준수하고 있는지 자동으로 검증하는 커밋 훅 구현

이미 진행된 프로젝트에 커밋 컨벤션을 적용하는 것은 불가능 하지만,
이 경험은 앞으로의 프로젝트에서 같은 실수를 하지 않을 교훈이 될 것이다.

수행 프로젝트 2/2 (작성예정)

팀 프로젝트

그룹웨어 시스템

WittyWave

프로젝트 목차

마무리

문제 해결을 즐기는
새로운 정보에 거리낌 없는

신입 백엔드 개발자 **이준경**이었습니다.
감사합니다.

연락처 010-8223-3782
이메일 worldreaming@gmail.com
깃허브 https://github.com/ljk1782