ARIJ (JIRA 협업 툴)

AJIR (6조)

contact: tnsdyd6933@naver.com

발표 목차

1. 프로젝트 소개
2. 시연 영상
3. 조금 난이도가 있었던..
4. 팀원 소개 및 소감

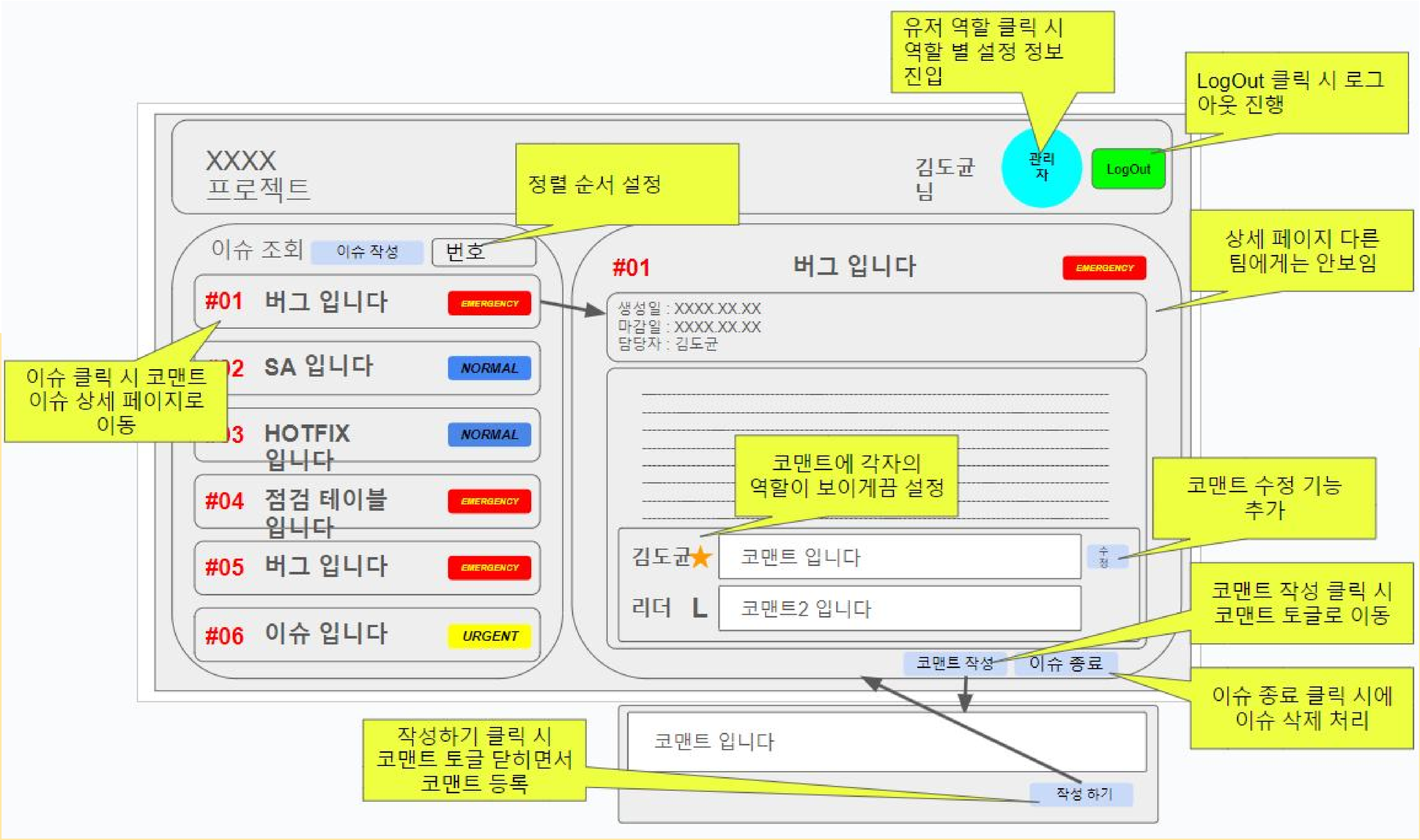
프로젝트 소개

1. 프로젝트 발의
2. 프로젝트 기획
   - 유저 권한
   - 기술 선택

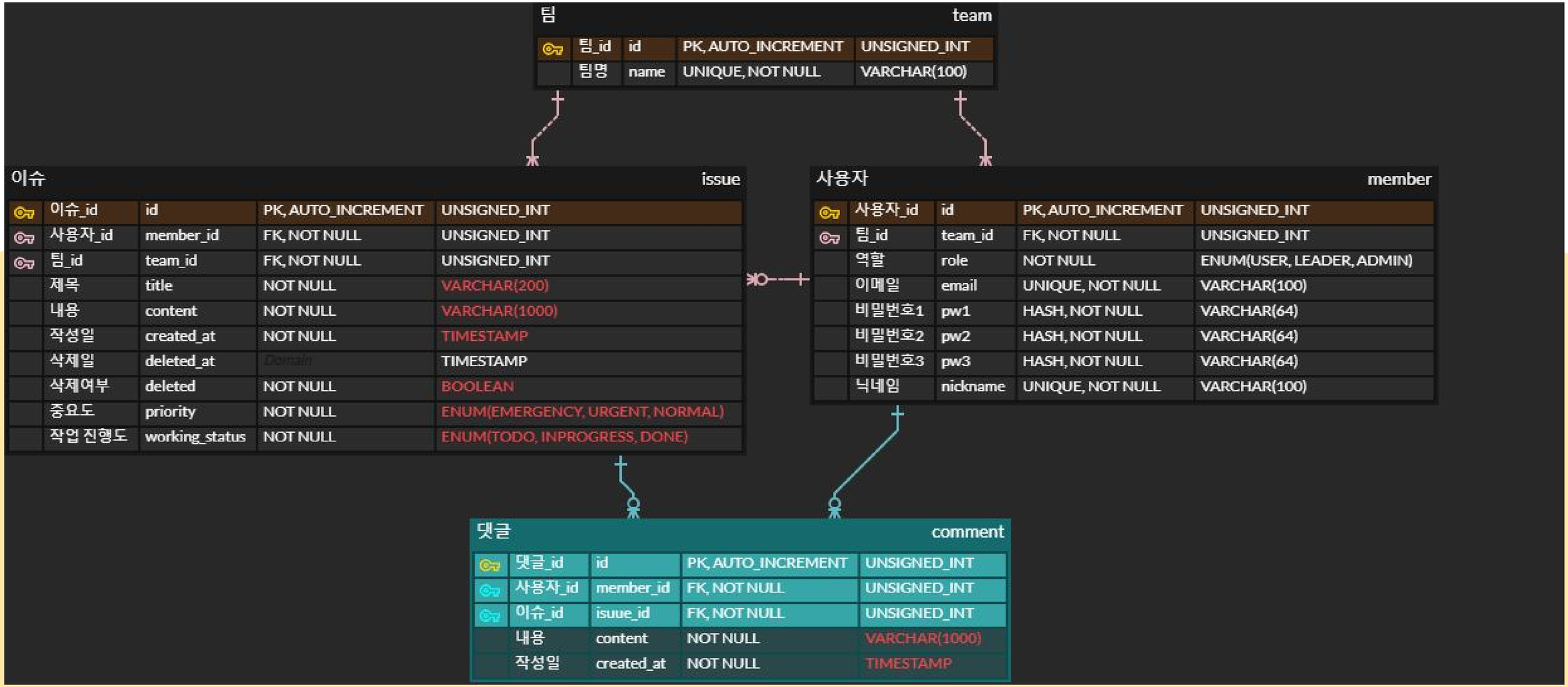
프로젝트 발의

ARIJ (JIRA 협업툴)

- 개발 이슈 공유
- 팀 구성 (Leader)
- 코멘트 소통
- 중요도 지정
- 작업 상태 지정



프로젝트 기획 DBMS


팀
team
팀_id
id
PK, AUTO_INCREMENT
UNSIGNED_INT
팀명
name
UNIQUE, NOT NULL
VARCHAR(100)
이슈
issue
이슈_id
id
PK, AUTO_INCREMENT
UNSIGNED_INT
사용자_id
member_id
FK, NOT NULL
UNSIGNED_INT
팀_id
team_id
FK, NOT NULL
UNSIGNED_INT
제목
title
NOT NULL
VARCHAR(200)
내용
content
NOT NULL
VARCHAR(1000)
작성일
created_at
NOT NULL
TIMESTAMP
삭제일
deleted_at
Domain
TIMESTAMP
삭제여부
deleted
NOT NULL
BOOLEAN
중요도
priority
NOT NULL
ENUM(EMERGENCY, URGENT, NORMAL)
작업 진행도
working_status
NOT NULL
ENUM(TODO, INPROGRESS, DONE)
사용자
member
사용자_id
id
PK, AUTO_INCREMENT
UNSIGNED_INT
팀_id
team_id
FK, NOT NULL
UNSIGNED_INT
역할
role
NOT NULL
ENUM(USER, LEADER, ADMIN)
이메일
email
UNIQUE, NOT NULL
VARCHAR(100)
비밀번호1
pw1
HASH, NOT NULL
VARCHAR(64)
비밀번호2
pw2
HASH, NOT NULL
VARCHAR(64)
비밀번호3
pw3
HASH, NOT NULL
VARCHAR(64)
닉네임
nickname
UNIQUE, NOT NULL
VARCHAR(100)
댓글
comment
댓글_id
id
PK, AUTO_INCREMENT
UNSIGNED_INT
사용자_id
member_id
FK, NOT NULL
UNSIGNED_INT
이슈_id
isuue_id
FK, NOT NULL
UNSIGNED_INT
내용
content
NOT NULL
VARCHAR(1000)
작성일
created_at
NOT NULL
TIMESTAMP

프로젝트 기획 API 설계 & 기능 명세

TeamController

팀 생성	POST	201	409
팀 전체 조회	GET	200	
팀 단건 조회	GET	200	
팀 수정	PUT	200	
팀 삭제	DELETE	204	401
팀원 초대	PATCH	200	401/409
팀원 초대 (관리자 전용)	PATCH	200	401/409
팀원 추방	DELETE	204	401/404

IssueController

이슈 생성	POST	201	409
이슈 전체 조회	GET	200	
이슈 단건 조회	GET	200	401
이슈 수정	PUT	200	401/409
이슈 수정(중요도)	PATCH	200	
이슈 수정(작업 상태)	PATCH	200	
이슈 삭제	DELETE	204	404

MemberController

회원가입	POST	201	400/409
로그인	POST	200	400/401
회원 탈퇴	DELETE	204	401
프로필 수정	PATCH	200	409
프로필 수정(비밀번호)	PATCH	204	400/409

CommentController

댓글 생성	POST	201
댓글 수정	PUT	200
댓글 삭제	DELETE	204

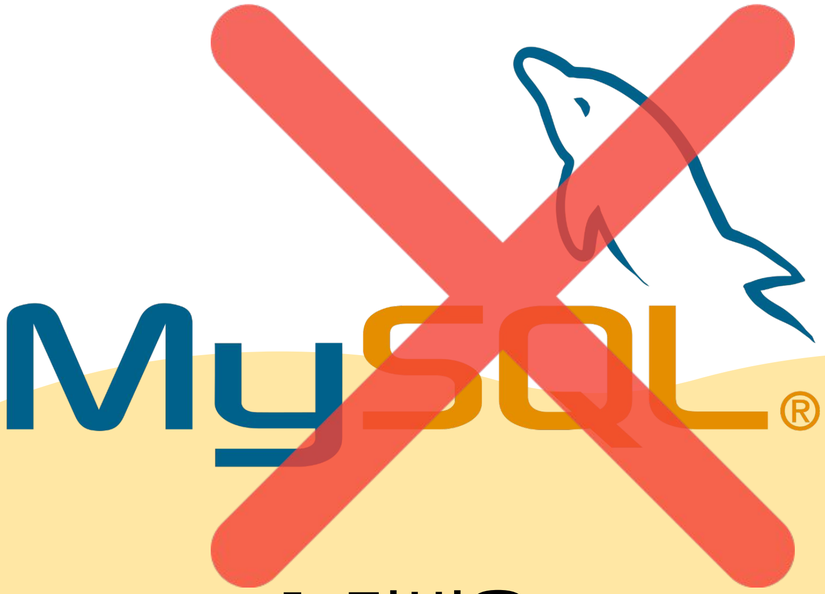
프로젝트 기획 유저 권한
(USER, LEADER, ADMIN)

ADMIN
- Full Access
  - 전체 팀 목록 조회

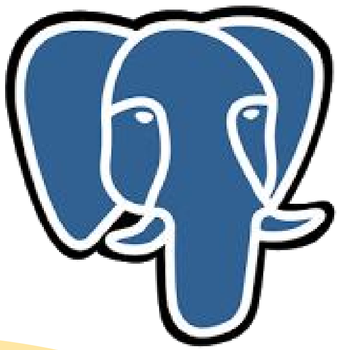
Leader
- Team Access
- Issue
  - 본인 팀의 Issue CRUD 가능 (User와 동일)
- Comment
  - 본인이 속한 팀의 본인이 작성한 Comment는 CRUD 가능
  - 본인이 속한 팀의 팀원이 작성한 Comment는 RD만 가능
- Team
  - 팀이 없는 User를 본인이 속한 팀에 초대 가능
  - 본인의 팀에 속한 User를 방출 가능
  - 본인이 속한 Team Table 관련 RUD 가능

User
- Team Issue Access
- Issue
  - 본인 팀의 Issue CRUD 가능
- Comment
  - 본인이 속한 팀의 본인이 작성한 Comment만 CRUD 가능
- Team
  - 팀 생성 가능(팀 생성시 권한 Leader로 승격)

프로젝트 기획 기술 선택



1. 가벼움
2. 읽기 최적화 DB
3. 레퍼런스 많음



PostgreSQL

1. MySql에 비해 더 좋아진 R/W 성능
2. 비교적 레퍼런스 적음

프로젝트 기획 기술 선택

supabase

1. 쉬운 DB 조작
2. DB Pool 제한
3. 성능/속도 이슈 심각

VS

127.0.0.1

localhost

1. DB 커스터마이징이 자유로움
2. SQL 쿼리문 실행속도가 빠름
3. 컴퓨터 끄면 아무것도 못함

(안그러면 됨ㅋㅋ)

시연 영상

https://www.youtube.com/watch?v=byN59cJxvIg

조금 난이도가 있었던..

1. @AuthenticationPrincipal
2. JWT role에 관하여
3. Dummy Team
4. JPAReopsitory 자동 맵핑
5. 비밀번호 3회 기록
6. 정규식

토큰에서 정보 찾아내기??

어디서? 어떻게?? 🤔

컨트롤러에서 헤더를 직접 받아서
토큰 정보를 찾아내자

```
@RequestHeader httpsHeaders: HttpHeaders
ResponseEntity<CommentCreateResponse> {
 val token: String = httpsHeaders["Authorization"]?.get(0) ?: throw TokenException("No Token Found")
 val userId: Long = jwtPlugin.validateToken(token.substring("Bearer ".length)).getOrNull()?.payload!!.subject.toLong()
```

토큰의 페이로드에 담긴 정보

어? 근데 우리는
스프링 시큐리티를 써서
필터에서 검증하는데?

```
val jwt = request.getBearerToken()

if (jwt != null) {
    jwtPlugin.validateToken(jwt)
        .onSuccess {
            val userId = it.payload.subject.toLong()
            val role = it.payload.get("role", String::class.java)
            val email = it.payload.get("email", String::class.java)

            val principal = UserPrincipal(
                id = userId,
                email = email,
                roles = setOf(role)
            )
            val authentication = JwtAuthenticationToken(
                principal = principal,
                details = WebAuthenticationDetailsSource().buildDetails(request)
            )
            SecurityContextHolder.getContext().authentication = authentication
        }
}
```

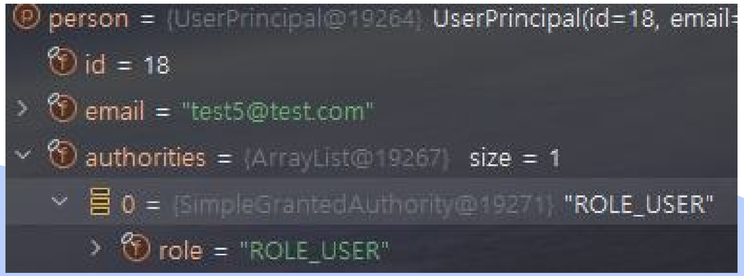
@AuthenticationPrincipal

시큐리티 컨텍스트에서 정보를 가지고 오자!

필터에서 UserPrincipal 정보를 시큐리티 컨텍스트 홀더에 넣어둠

```
SecurityContextHolder.getContext().authentication = authentication
```

UserPrincipal에 담긴 정보



컨트롤러에 아래와 같이 추가한다.

```
@AuthenticationPrincipal person: UserPrincipal?,
```

가져온 정보로 DB에서 유저 찾기

```
val member: Member =
    memberRepository.findByIdOrNull(person.id) ?: throw ModelNotFoundException("맴버", person.id.toString())
```

JWT Role

USER -> LEADER

①User 로그인

②현재 User의 Role
JWT 생성 및 전달

③User 의 Role 변경

④제출된 토큰의 Role과
실제 Role이 상이함

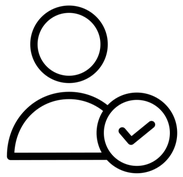
Role 변경 후 토큰 재발급 강제

vs

JWT Role 사용 안하기

DUMMY TEAM

Team 생성 기본 Process



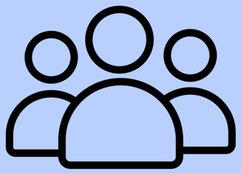
1. 아무 팀도 아닌 유저가 팀을 만들면 리더로 권한 승격

아무 팀도 아닌 새로운 유저는 어떤 소속이지??

Create Team

Null??, DUMMY TEAM??
팀원들과 협의 후

DUMMY TEAM 에서 팀원의 정보를 가져온다



1 번 DUMMY TEAM 생성으로 합의

2. 팀 Id 를 부여 받음과 동시에 팀원 초대가 가능해진다

JPA Repository 자동 맵핑

예상 Class 구상도

JpaRepository와 같이 상속

```
interface CustomTeamRepository {
```

QueryDSL과 같이 상속

```
interface TeamRepository: JpaRepository<Team, Long>, CustomTeamRepository {
```

```
@Repository  Ppajingae *
class QueryDslTeamRepositoryImpl: QueryDslSupport(), CustomTeamRepository
```

여만 Service 에서 사용

```
@Service  Ppajingae +1 *
@Transactional
class TeamService(
    private val teamRepository: TeamRepository,
    private val memberRepository: MemberRepository,
) {
```

Service에서 Bean 을 찾지 못했다고??

```
Error creating bean with name 'memberService' defined in file
```

???

JPA Repository 자동 맵핑

원인은??

Spring Data JPA

```
interface TeamRepository: JpaRepository<Team, Long>, CustomTeamRepository {
```

JPA 에서 인식을 못함

```
@Repository  Ppajingae *
class QueryDslTeamRepositoryImpl
```

네이밍 일치로 인식

```
@Repository  Ppajingae *
class TeamRepositoryImpl:
```

비밀번호 🔓 3회 기록

① 처음 구상은 DB에 이전 비밀번호들을 저장하는 Table을 하나 더 만들어 구현

연관관계의 복잡성과 특별한 테이블의 필요 여부가 없음에 따라 수정

② Member Model에 비밀번호 1, 2, 3을 저장하는 Column을 만들어서 저장.

```
@Column(nullable = false, length = 64, name = "pw1")
var password: String? = null,

@Column(nullable = false, length = 64, name = "pw2")
var password2: String? = null,

@Column(nullable = false, length = 64, name = "pw3")
var password3: String? = null,
```

③ 비밀번호 수정 Service에서 최근 3회의 비밀번호 비교 예외 발생 / 비밀번호 수정 후 최근 비밀번호 갱신

```
if(request.oldPw == request.newPw){
    throw PasswordRecordException()
}
if(bCryptPasswordEncoder.matches(request.newPw, member.password2)){
    throw PasswordRecordException()
    }
if(bCryptPasswordEncoder.matches(request.newPw, member.password3)){
    throw PasswordRecordException()
}

member.password3= member.password2
member.password2 = member.password
member.password =
    bCryptPasswordEncoder.encode(
        request.newPw
    )
```

정규식~~(이 형!! 말투 쫌!!) (개발자들이 꺼려하는 까칠한 규식이 형)~~

1. 최소 한글자 이상 (전방탐색)

```
(?=.*[A-Z])
```

```
(?=.*\d)
```

2. 허용 글자 범위

```
[A-Za-z\d!@#$%^&*()_+\-=\[\]{};':"\\|,.<>/?]+
```

3. 최종식

```
(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[!@#$%^&*()_+\-=\[\]{};':"\\|,.<>/?])[A-Za-z\d!@#$%^&*()_+\-=\[\]{};':"\\|,.<>/?]+
```

팀원 소개 및 소감

팀원 소개

이순용

ENFJ

tnsdyd6933@naver.com

https://github.com/ddalkyTokky

팀장

DBMS 엔티티

Member Domain 설계

~~실패하면 반역!~~

김도균

INTP

wkmkyj991112@gmail.com

https://github.com/Ppajingae

와이어프레임 설계

Team Domain 설계

한동헌

INTP

dongheon827@naver.com

https://github.com/dongheon0827

API 리스트 설계

Issue Domain 설계

~~매수된 3조 수문장~~

최민수

ISTP

hifumialice@gmail.com

https://github.com/HifumiAlice

API 리스트 설계

Comment Domain 설계

QA

~~성공하면 혁명!!~~

소감 한마디

한동헌

개념 원리에 대해 여러가지 알아가는

좋은 주차가 되었다

이순용

권한 설정은 생각보다 간단하지

않고, 반복 코드가 많다!!

스파르타야 코드먹자 코코코코코코 코코 코코 코코코코 코코코

김도균

여러 의미에서 팀원들에게 많은

도움을 받았습니다 감사합니다

최민수

어랏..? 왜 되는거지?

충돌 해결.. 잼있다!

감사합니다!!

AJIR (6조)

tnsdyd6933@naver.com