MBTI 기반 매칭 플랫폼

Moim?

3조
지원준 (팀장)
권택준
남궁찬
유부미

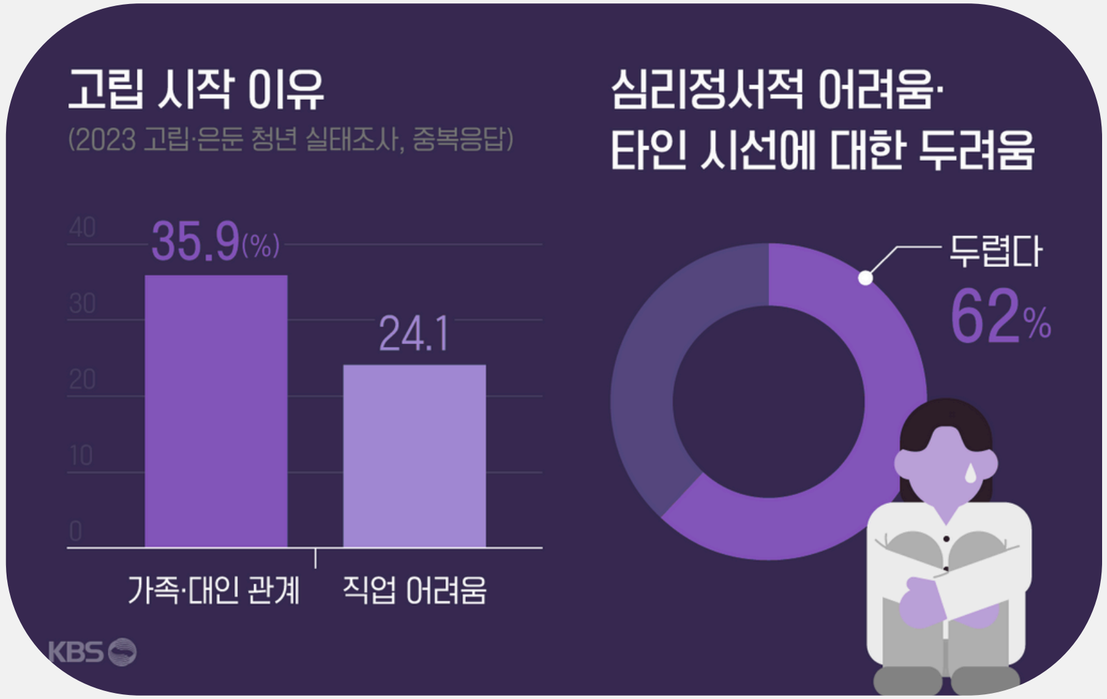
2025.07.29 ~ 2025.08.14

목차

1 프로젝트 배경

"20명 중 1명 집에서 안 나온다"
'고립·은둔 청년' 2배 증가

'대인 관계' 어려움에 고립…타인 시선·접촉 두려워



최근 20~30대 청년들 사이에서 사회적 고립감이 증가
특히 코로나 이후 사람들과 자연스럽게 어울릴 기회가 감소

2 프로젝트 개요

MBTI 성향 기반 상호작용 플랫폼

Moim?

“MBTI가 모임?”

“혼자가 아님을 느낄 수 있는 경험 제공”

“온라인에서 쉽게 소통하고, 오프라인으로 자연스럽게 연결“

3 타겟 사용자

혼자 있는 시간이 길어 외로운, 사람을 만나기 어려운 20~30대 청년

김민수(27) / 프리랜서 / 취미·학습 모임 필요

이지은(25) / 이직 준비 / 네트워킹 필요

4 프로젝트 목표

MOIM?

목표 1

성향 기반 매칭으로 대화 시작 부담 최소화

목표 2

안전한 커뮤니티 유지로 장기적 신뢰 형성

목표 3

온라인 → 오프라인으로 자연스럽게 이어지는 모임

5 주요 기능 & 팀원 역할 분담

사용자 인증 — 팀장 지원준

자체 회원가입 / 네이버 로그인
보안 / 신고 관리

MBTI 기반 — 남궁찬

MBTI 성향 테스트와
피드 게시판

모임 커뮤니티 — 유부미

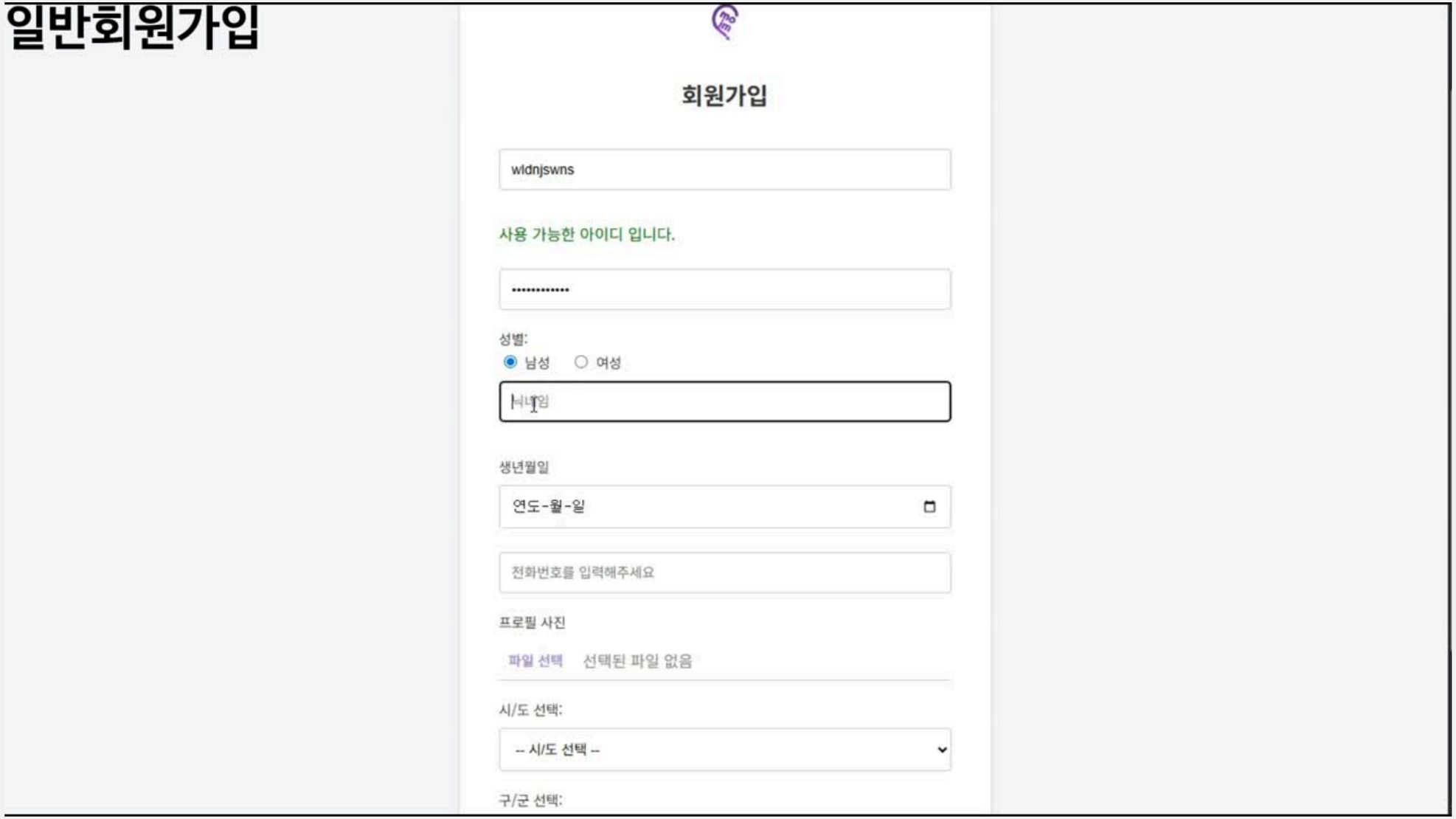
사용자 모임 및 일정 생성
모임 후기 리뷰 별점 시스템

실시간 통신 — 권택준

사용자 간 실시간 상호작용
(채팅 / 알림)

6 시연 영상

일반회원가입

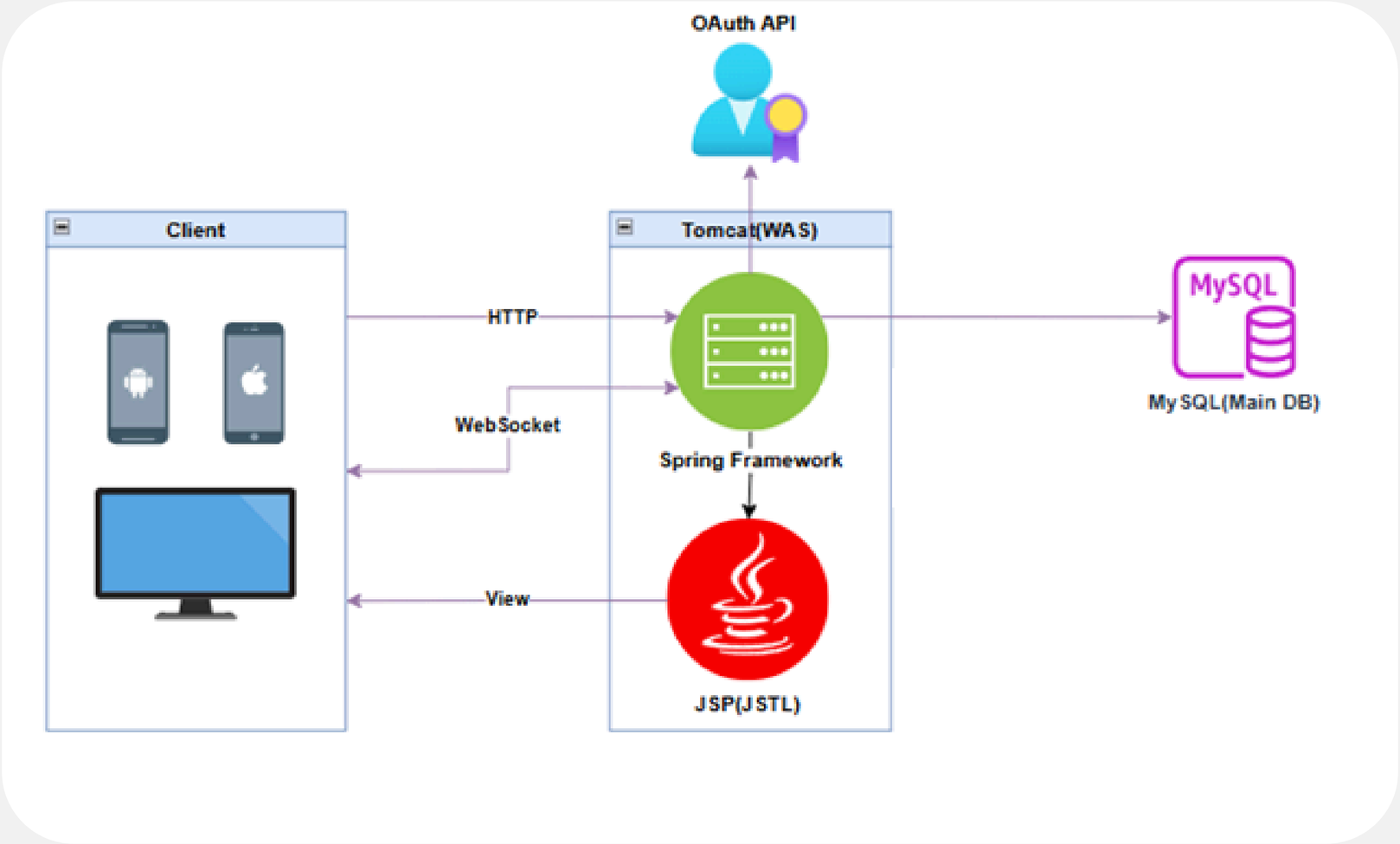


7 프로젝트 수행 절차 및 방법

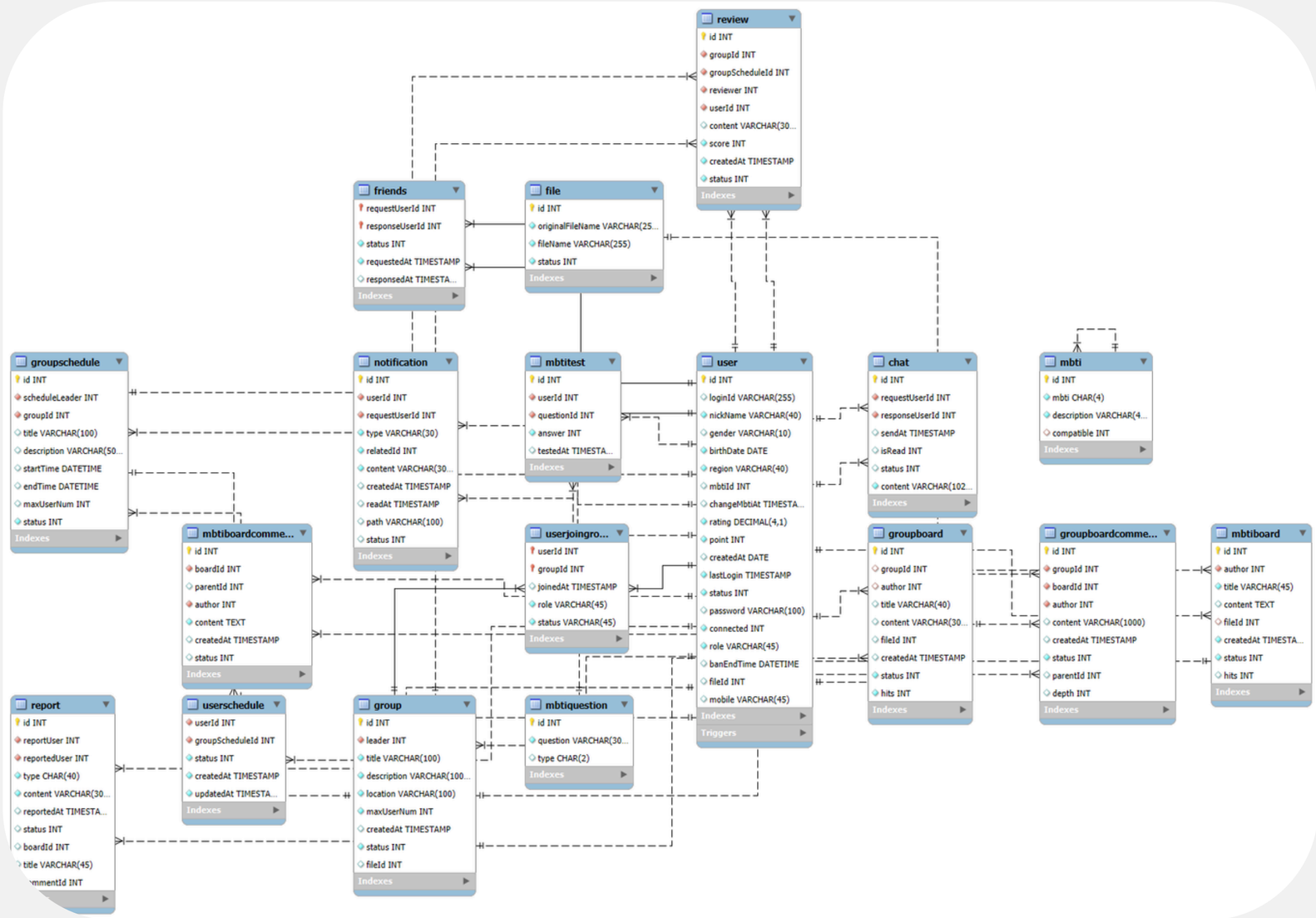
https://github.com/won-Jo0n/moim

01	프로젝트 준비 및 설계	유스케이스 다이어그램과 ERD 작성
02	개발 착수 및 협업	개발 시작, GitHub 형상관리
03	오류 검수와 서비스 구축	버그 수정 후 AWS 배포 예정

8 시스템 아키텍처


OAuth API
Client
Tomcat(WAS)
HTTP
WebSocket
Spring Framework
View
JSP(JSTL)
MySQL
My SQL(Main DB)

9 ERD


review
id INT
groupId INT
groupScheduleId INT
reviewer INT
userId INT
content VARCHAR(30...
score INT
createdAt TIMESTAMP
status INT
Indexes
friends
requestUserId INT
responseUserId INT
status INT
requestedAt TIMESTAMP
responsedAt TIMESTA...
Indexes
file
id INT
originalFileName VARCHAR(25...
fileName VARCHAR(255)
status INT
Indexes
groupschedule
id INT
scheduleLeader INT
groupId INT
title VARCHAR(100)
description VARCHAR(50...
startTime DATETIME
endTime DATETIME
maxUserNum INT
status INT
Indexes
notification
id INT
userId INT
requestUserId INT
type VARCHAR(30)
relatedId INT
content VARCHAR(30...
createdAt TIMESTAMP
readAt TIMESTAMP
path VARCHAR(100)
status INT
Indexes
mbtitest
id INT
userId INT
questionId INT
answer INT
testedAt TIMESTA...
Indexes
user
id INT
loginId VARCHAR(255)
nickName VARCHAR(40)
gender VARCHAR(10)
birthDate DATE
region VARCHAR(40)
mbtiId INT
changeMbtiAt TIMESTA...
rating DECIMAL(4,1)
point INT
createdAt DATE
lastLogin TIMESTAMP
status INT
password VARCHAR(100)
connected INT
role VARCHAR(45)
banEndTime DATETIME
fileId INT
mobile VARCHAR(45)
Indexes
Triggers
chat
id INT
requestUserId INT
responseUserId INT
sendAt TIMESTAMP
isRead INT
status INT
content VARCHAR(102...
Indexes
mbti
id INT
mbti CHAR(4)
description VARCHAR(4...
compatible INT
Indexes
mbtiboardcomme...
id INT
boardId INT
parentId INT
author INT
content TEXT
createdAt TIMESTAMP
status INT
Indexes
userjoingro...
userId INT
groupId INT
joinedAt TIMESTAMP
role VARCHAR(45)
status VARCHAR(45)
Indexes
groupboard
id INT
groupId INT
author INT
title VARCHAR(40)
content VARCHAR(30...
fileId INT
createdAt TIMESTAMP
status INT
hits INT
Indexes
groupboardcomme...
id INT
groupId INT
boardId INT
author INT
content VARCHAR(1000)
createdAt TIMESTAMP
status INT
parentId INT
depth INT
Indexes
mbtiboard
id INT
author INT
title VARCHAR(45)
content TEXT
fileId INT
createdAt TIMESTA...
status INT
hits INT
Indexes
report
id INT
reportUser INT
reportedUser INT
type CHAR(40)
content VARCHAR(30...
reportedAt TIMESTA...
status INT
boardId INT
title VARCHAR(45)
mmentId INT
userschedule
userId INT
groupScheduleId INT
status INT
createdAt TIMESTAMP
updatedAt TIMESTA...
Indexes
group
id INT
leader INT
title VARCHAR(100)
description VARCHAR(100...
location VARCHAR(100)
maxUserNum INT
createdAt TIMESTAMP
status INT
fileId INT
Indexes
mbtiquestion
id INT
question VARCHAR(30...
type CHAR(2)
Indexes

10 프로젝트 수행 경과

1. 사용자 비밀번호 암호화 및 인증

적용 기술 : Spring Security

적용 목적 : 안전한 비밀번호 관리와 인증 절차를 통해 보안 강화

주요 코드

```
@PostMapping("/join")  wonJoon +1
public String join(@ModelAttribute UserDTO userDTO, @RequestParam("command") String command,
                   @RequestParam("city") String city, @RequestParam("county") String county,
                   @RequestParam(value="profile", required = false)MultipartFile profile) throws IOException {

    String region = city + " " + county;
    userDTO.setRegion(region);

    if(!profile.isEmpty()){
        int fileId = fileUtil.fileSave(profile);
        userDTO.setFileId(fileId);
    }else{
        userDTO.setFileId(0);
    }

    if(command.equals("OAuthJoin")){
        oAuthService.OAuthJoin(userDTO);
    }else{
        String encodedPassword = passwordEncoder.encode(userDTO.getPassword());
        userDTO.setPassword(encodedPassword);

        int result = userService.join(userDTO);
    }

    return "redirect:/";
}
```

```
@Service  2 usages  WON_JOON₩jee52
@RequiredArgsConstructor
public class CustomerUserDetailService implements UserDetailsService {
    private final UserRepository userRepository;


    @Override  1 usage  WON_JOON₩jee52
    public UserDetails loadUserByUsername(String loginId) throws UsernameNotFoundException {
        UserDTO user = userRepository.getUserByLoginId(loginId);  // 아이디로 사용자 검색



        if (user == null) {
            throw new UsernameNotFoundException("사용자를 찾을 수 없습니다: " + loginId);
        }
        // 1. status가 0이고, ban_end_time이 현재 시간보다 미래라면
        if (user.getStatus() == 0 && user.getBanEndTime() != null && user.getBanEndTime().isAfter(LocalDateTime.now())) {
            // 사용자에게 보낼 메시지 (예외 메시지)
            DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
            String formattedDate = user.getBanEndTime().format(formatter);

            String message = String.format("정지된 계정입니다. 제재 해제 날짜: %s", formattedDate);
            throw new DisabledException(message); // 로그인 거부
        }

        // 2. status가 0이지만, ban_end_time이 현재 시간보다 과거라면
        //    (즉, 제재 기간이 만료되었을 경우)
        if (user.getStatus() == 0 && user.getBanEndTime() != null && user.getBanEndTime().isBefore(LocalDateTime.now())) {
            // 사용자 상태를 정상으로 되돌리는 로직
            user.setStatus(1);
            user.setBanEndTime(null);
            userRepository.updateUserStatus(user);
        }

        return new CustomerUserDetails(user);
    }
}
```

```
public class CustomerUserDetails implements UserDetails, Serializable {  8 usages  WON
    private static final long serialVersionUID = 1L;  no usages
    private final UserDTO userDTO; // UserDTO 객체를 포함  7 usages

    public CustomerUserDetails(UserDTO userDTO) {  1 usage  WON_JOON₩jee52
        this.userDTO = userDTO;
    }

    // 실제 로그인된 사용자 정보를 가져올 때 사용
    public UserDTO getUserDTO() {  2 usages  WON_JOON₩jee52
        return userDTO;
    }

    @Override  1 usage  WON_JOON₩jee52
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return Collections.singletonList(new SimpleGrantedAuthority(userDTO.getRole()))
    }
```

10 프로젝트 수행 경과

2. 사용자 간편 로그인 기능

적용 기술 : Naver OAuth API

적용 목적 : 서드파티 로그인을 통한 사용자 편의성 향상

주요 코드

```
@GetMapping("/oauthLogin")  no usages  wonJoon *
public String naverLogin(HttpSession session) throws UnsupportedEncodingException {
    String clientId = "";
    String redirectURI = java.net.URLEncoder.encode( s: "http://localhost:8080/naver/callback",  enc: "UTF-8");
    String state = UUID.randomUUID().toString();
    session.setAttribute( s: "naver_state", state);

    String apiURL = "https://nid.naver.com/oauth2.0/authorize?response_type=code"
            + "&client_id=" + clientId
            + "&redirect_uri=" + redirectURI
            + "&state=" + state;
    return "redirect:" + apiURL;
}
```

```
@RequestMapping(value = "/naver/callback", method = RequestMethod.GET)  no usages  wonJoon +1 *
public String naverCallback(@RequestParam String code,
                            @RequestParam String state,
                            HttpSession session, Model model) throws Exception {
    String sessionState = (String) session.getAttribute( s: "naver_state");
    if (!state.equals(sessionState)) {
        throw new IllegalStateException("잘못된 state 값입니다.");
    }

    // 1. Access Token 요청
    String clientId = "";
    String clientSecret = "";
    String redirectURI = URLEncoder.encode( s: "http://localhost:8080/naver/callback",  enc: "UTF-8");

    String tokenURL = "https://nid.naver.com/oauth2.0/token?grant_type=authorization_code"
            + "&client_id=" + clientId
            + "&client_secret=" + clientSecret
            + "&code=" + code
            + "&state=" + state;

    URL url = new URL(tokenURL);
    HttpURLConnection con = (HttpURLConnection)url.openConnection();
    con.setRequestMethod("GET");

    BufferedReader br = new BufferedReader(new InputStreamReader(con.getInputStream()));
    String line, response = "";
    while ((line = br.readLine()) != null) {
        response += line;
    }
    br.close();

    // JSON 파싱
    JSONObject json = new JSONObject(response);
    String accessToken = json.getString( key: "access_token");

    // 2. 사용자 정보 요청
    URL infoUrl = new URL( spec: "https://openapi.naver.com/v1/nid/me");
    HttpURLConnection infoCon = (HttpURLConnection) infoUrl.openConnection();
    infoCon.setRequestMethod("GET");
    infoCon.setRequestProperty("Authorization", "Bearer " + accessToken);
```

```
BufferedReader infoBr = new BufferedReader(new InputStreamReader(infoCon.getInputStream()));
String infoLine, userInfo = "";
while ((infoLine = infoBr.readLine()) != null) {
    userInfo += infoLine;
}
infoBr.close();

JSONObject userJson = new JSONObject(userInfo);
JSONObject responseObj = userJson.getJSONObject( key: "response");
String naverId = (String) responseObj.get("id");
Map<String, Object> oAuthData = new HashMap<>();

for(String key : responseObj.keySet()){
    oAuthData.put(key, responseObj.get(key));
    System.out.println(key + ": " + responseObj.get(key));
}

try {
    UserDetails userDetails = customerUserDetailService.loadUserByUsername(naverId);

    // SecurityContext에 인증 정보 설정
    Authentication authentication = new UsernamePasswordAuthenticationToken(
            userDetails,  credentials: null, userDetails.getAuthorities()
    );
    SecurityContextHolder.getContext().setAuthentication(authentication);

    return "redirect:/home"; // 로그인 성공 후 홈 페이지로 리다이렉트
} catch (UsernameNotFoundException e) {
    // 사용자 정보가 DB에 없는 경우 회원가입 페이지로 이동
    // 네이버에서 가져온 정보를 회원가입 폼에 미리 채워넣을 수 있습니다.
    model.addAttribute( attributeName: "OAuthData", oAuthData);
    return "forward:/user/join";
} catch (DisabledException e) {
    // 정지된 계정일 경우 처리
    model.addAttribute( attributeName: "errorMessage", e.getMessage());
    return "redirect:/"; // 로그인 폼으로 돌아가 에러 메시지 표시
}
```

10 프로젝트 수행 경과

3. 실시간 사용자 간 통신

적용 기술 : WebSocket + STOMP 프로토콜

적용 목적 : 지연 없는 양방향 실시간 메시지 송수신 구현

주요 코드

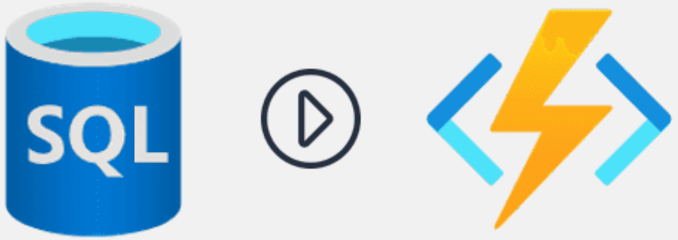
```java
@Controller  0개의 사용위치  xorwns
@RequiredArgsConstructor
public class WebSocketController {
    private final SimpMessagingTemplate messagingTemplate;
    private final NotificationService notificationService;
    private final FriendsService friendsService;
    private final ChatService chatService;
    private final UserService userService;
    private final Map<String, String> sessionToUser = new ConcurrentHashMap<>();  4개 사용 위치
    private final Map<String, Integer> sessionCount = new ConcurrentHashMap<>();  5개 사용 위치
    private final Queue<String> matchingQueue = new ConcurrentLinkedQueue<>();  5개 사용 위치
    private final Object matchingLock = new Object();  1개 사용 위치

    @MessageMapping("/notification")  xorwns
    public void notification(@Header("type") String type, @Payload Map<String, Object> data, Principal principal){
        String userId = principal.getName();
        if(type.equals("READ_NOTIFICATION")) {
            int notificationId = (int)data.get("notificationId");
            notificationService.readNotification(Integer.parseInt(userId), notificationId);
            messagingTemplate.convertAndSendToUser(userId,  destination: "/queue/main", Map.of( k1: "notificationId", notificationId), Map.of( k1: "type", type));
        }else if(type.equals("ACCEPT_FRIEND")){
            FriendsDTO friendsDTO = new FriendsDTO();
            friendsDTO.setRequestUserId((int)data.get("requestUserId"));
            friendsDTO.setResponseUserId(Integer.parseInt(userId));
            friendsDTO.setStatus(1);
            friendsService.updateFriend(friendsDTO);
        }
    }
```

```javascript
const stompClient = new StompJs.Client({
  webSocketFactory: () => new SockJS("/ws-stomp"),
});
stompClient.onWebSocketError = (error) => {
  console.error("WEBSOCKET ERROR", error);
};
stompClient.onStompError = (frame) => {
    console.error("STOMP ERROR", frame);
  //console.error("Broker reported error: " + frame.headers["message"]);
  //console.error("Additional details: " + frame.body);
};
stompClient.onDisconnect = (frame) => {
   console.error("DISCONNECT", frame);
};
stompClient.onConnect = (frame) => {
  stompClient.subscribe("/user/queue/main", (msg) => {
    const data = JSON.parse(msg.body);
    let messageItem;
    switch (msg.headers.type) {
      case "FRIEND_ONLINE":
        $(`div[data-user-id="${data.sender}"] .online-indicator`).addClass("online");
        break;
```

10 프로젝트 수행 경과

4. 소프트 딜리트



적용 기술 : 데이터베이스 트리거 + Spring Scheduler

적용 목적 : 데이터 복구 가능성을 보장하고, 기록 보존을 통해 악의적 행위 추적 및 사이버 범죄 예방

주요 코드

Name	Event	Table	Timing	Created	SQL Mode	Definer	Client Character...	Connec
update_board_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_chat_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_feedboard_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_feedboardcomment...	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_friends_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_group_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_groupboard_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_groupboardcomme...	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_groupschedule_sta...	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_mbtiboard_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_mbtiboardcommen...	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_notification_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_report_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb
trg_update_review_status	UPDATE	user	AFTER	2025-08-12 22:3...	ONLY_FULL_GR...	root@localhost	utf8mb4	utf8mb

```xml
<update id="delete" parameterType="int">
    UPDATE user SET status = -1 WHERE id = #{id};
</update>
```

```java
@Component  no usages  wonJoon
@EnableScheduling
@RequiredArgsConstructor
public class MyScheduler {
    private final MySchedulerService mySchedulerService;

    @Scheduled(cron = "0 0 4 * * ?")  no usages  wonJoon
    public void dailyCheck() {
        // 탈퇴한지 6개월이 지난 사용자 데이터 삭제
        mySchedulerService.deleteOldUserData();

        // 정지 일자가 지난 사용자를 정지 해제
        mySchedulerService.unblockUser();

        // 채팅 데이터 3개월 지나면 삭제
        mySchedulerService.deleteOldChatData();
    }
}
```

```xml
<mapper namespace="Scheduler">
    <delete id="deleteOldUserData">
        <![CDATA[
        DELETE FROM user
        WHERE lastLogin < DATE_SUB(NOW(), INTERVAL 6 MONTH);
        ]]>
    </delete>

    <update id="unblockUser">
        <![CDATA[
        UPDATE user SET status = 1 , banEndTime = null
        WHERE banEndTime <= NOW();
        ]]>
    </update>

    <delete id="deleteOldChatData">
        <![CDATA[
        DELETE FROM chat
        WHERE sendAt < DATE_SUB(NOW(), INTERVAL 3 MONTH);
        ]]>
    </delete>
</mapper>
```

11 마케팅 전략 & 기대 효과

SNS 숏폼 마케팅

- 인스타, 틱톡에서 모임후기
- 유튜브에 매칭 브이로그 콘텐츠 제작

오프라인 지역 제휴 네트워크

- 지역 상권 및 소상공인과 직접 파트너십 체결
- EX) 스터디룸, 공방, 피트니스 센터 등

친구 초대 리워드

- 초대한 친구가 첫 모임 참여시 플랫폼 포인트 지급
- 포인트는 제휴 가게에서 현금처럼 사용가능

- 바이럴 확산 속도 향상
- 플랫폼 브랜딩과 '모임 후기' 노출을 통해 신뢰도 향상
- 젊은 타겟층 집중 공략
- 규모 예산으로도 높은 노출 가능.

- 플랫폼 내 모임 참여 → 오프라인 교류 증가
- 제휴 가게 이용으로 지역 상권 매출 상승
- 지역 경제 활성화로 이어지는 구조 확립
- 상권·사용자·모임 주최자가 서로 연결되는 구조 → 지역 커뮤니티 구축

- 네트워크 기반 유입 가속화
- 사용자 유지율 상승
- 초대 과정에서 자연스럽게 홍보 효과 발생

12 업그레이드 방안

매칭 알고리즘 고도화

- 관심사 태그·활동 빈도·참여 후 평점까지 반영
- 개인화 추천(AI) 적용

수익 모델 다각화

- 포인트·리워드 기반 결제 시스템
- 프리미엄 구독: 매칭 우선·검색 범위 확장
- 체험형 패키지 판매
- 유료 사용자 툴

플랫폼 확장

- 모바일 앱(iOS/Android) 출시
- 위치 기반 즉시 매칭 기능 추가

- 맞춤형 매칭 정확도 향상
- 데이터 축적으로 추천 시스템, 마케팅 타겟팅에 활용 가능

- 다양한 수익원 확보
- 지역 상권과의 동반 성장
- 플랫폼 내 소비 선순환

- 사용자 접근성 극대화
- 타겟 시장 확대
- 브랜드 경쟁력 강화

13 느낀 점

[팀장 지원준] 초반 프로젝트를 기획하는 단계에서 구현 해야하는 기능들이 많이 생겨 기한 안에 마무리를 할 수 있을까 하는 생각이 들었었다. 하지만 각자 맡은 역할을 완벽하게 수행 해주었던 팀원들 덕분에 계획했던 예정대로 잘 마무리를 할 수 있었다. 프로젝트를 진행하며 예상치 못한 오류와 기능 변경이 있었지만, 문제를 분석하고 해결하는 능력이 향상되었다. DB 설계와 Git 협업 과정에서 유연성과 소통의 중요성을 배웠으며, 이번 경험을 바탕으로 더 완성도 높은 개발을 하고자 한다.

[팀원 권택준] 이번 프로젝트에서 저는 실시간 통신 구조와 동작 방식에 대한 이해가 부족해 많은 어려움을 겪었습니다. 하지만 팀원들의 도움을 받아 다양한 상황을 테스트하고, 서버·클라이언트 간 통신 흐름과 메시지 처리 방식을 개선하는 과정을 거치며 점차 문제 해결 방법을 익혀 나갈 수 있었습니다. 이 과정에서 이벤트 기반 프로그래밍, 비동기 처리, 네트워크 구조에 대한 이해가 한층 깊어졌고, 문제를 분석하고 원인을 파악하는 능력도 향상되었습니다. 무엇보다 팀과 함께 문제를 해결하며 기능을 완성해 나가는 과정에서 협업의 가치와 성취감을 크게 느낄 수 있었습니다.

[팀원 남궁찬] 이번 프로젝트를 진행하면서 다양한 오류를 마주했고, 처음에는 원인조차 파악하기 어려웠다. 하지만 오류를 하나씩 해결해 나가며 점점 패턴을 익히게 됐다. 비슷한 문제가 다시 발생했을 때는 이전보다 훨씬 빠르고 정확하게 대응할 수 있었다. 이 과정에서 단순한 기능 구현을 넘어 문제 해결 능력이 크게 향상됐다. 결국 오류 수정 경험이 프로젝트 완성도를 높이는 데 큰 밑거름이 되었다.

[팀원 유부미] 이번 프로젝트는 1차 때보다 규모가 커지고 DB를 활용하면서 구현해야 할 기능이 두 배 이상 늘어, 그 과정에서 많은 오류를 마주했습니다. 특히 DB 설계와 다양한 기능 연동 과정에서 잦은 오류가 발생해 수정과 테스트를 수없이 반복해야 했습니다. 그럼에도 불구하고, 팀원들이 서로를 믿고 격려하며 함께 문제를 해결해 준 덕분에 끝까지 무사히 마무리할 수 있었습니다. 이번 프로젝트를 통해 어려움 앞에서 당황하기보다 차분히 원인을 분석하고 해결책을 찾아가는 법을 배웠고, 덕분에 다음 프로젝트에서는 더 여유 있고 완성도 높게 임할 수 있을 것 같습니다.

Q & A

THANK YOU

감사합니다