모바일 프로그래밍 결과 보고서

201802293 환경공학과 이진명,
202001355 무역학과 안수빈,
202002090 환경공학과 홍석희

1.프로젝트 수행 목적

1) 프로젝트 정의
- 팀 프로젝트, 스터디 등의 그룹활동 진행에 도움을 주는 애플리케이션

2) 프로젝트 배경
- 그동안 팀프로젝트를 진행하면서 어려움을 겪었던 경험을 바탕으로 대학생들이 겪을 수 있는 팀 프로젝트와 관련된 어려움을 해결해주고자 하는 목적을 가지고 있다.
- 추가적으로 자격증, 취업 준비 등의 스터디 활동도 그룹으로 진행한다는 점에서 팀 프로젝트와 공통점이 있다. 대학생을 포함한 많은 사람들이 스터디를 진행한다는 점에서 이를 관리할 수 있는 애플리케이션을 개발하고자 한다.

3) 프로젝트 목표
- 팀 프로젝트 및 스터디 활동 진행 시 발생하는 여러 문제점 (ex. 그룹원 내에서의 일정 조율 문제, 열할 분담 및 확인 문제, 활동에 미참여하는 문제, 스터디원 모집의 어려움 등)을 해결하는 기능을 구현하여 사용자들이 원활한 팀 프로젝트 및 스터디 활동을 진행할 수 있도록 도움을 주는 것이 최종 목표다.

2. 개발 및 구축 환경

개발환경			
H/W 사양		S/W 사양	
프로세서	i5, i7	윈도우	Windows 11
RAM	8.00GB	개발언어	XML, Java
시스템 종류	64bit 운영체제 x64 기반 프로세서	Database	SQLite
		API	31
구동환경			
안드로이드 기기 사양		S, Android 12.0 x86_64 Galaxy A Quantum	
모바일 웹브라우저		사용 無	

3. 프로젝트 수행 추진 체계 및 일정

1) 각 조직원의 조직도 및 역할 분담

이진명 (팀장)	안수빈	홍석희
- 자료조사 및 정리 - 발표 - 앱 개발	- 자료조사 및 정리 - 스토리보드, ppt 등 자료 제작 - 앱 개발 및 디자인	- 자료조사 및 정리 - ppt 제작 - 앱 개발

2) 프로젝트 수행 일정 (~기획서 발표 / ~중간 발표 / ~기말 발표)

'팀플앱' 만들기 프로젝트 일정

ID	작업	시작일	일 수	종료일
1	프로젝트 착수 및 기능 구성	2022-09-29	7	2022-10-05
1.1	앱 주제 설정	2022-09-29	5	2022-10-03
1.2	앱 상세 기능 구성	2022-10-03	3	2022-10-05
2	시스템 개발	2022-10-15	56	2022-12-09
2.1	개별 역할분담 및 중간고사 기간	2022-10-15	13	2022-10-27
2.2	<로그인/회원가입> 페이지 구성	2022-10-28	6	2022-11-02
2.3	<마이페이지> 구성	2022-11-03	4	2022-11-06
2.4	DB공부 및 ERD 작성	2022-11-07	6	2022-11-12
2.5	<공지/출석/스케줄> 페이지 구성(DB사용 기능 제외)	2022-10-29	12	2022-11-09
2.6	<메인 홈> 페이지 구성 및 참여 탈퇴 기능 구현	2022-11-11	16	2022-11-26
2.7	<Work/그룹정보> 페이지 구성(DB 연동)	2022-11-27	12	2022-12-08
2.8	<하단바>기능 및 <메인 홈/ 참여>페이지 구성 (DB사용 기능 제외)	2022-10-30	10	2022-11-08
2.9	<참여> 페이지 DB연동	2022-11-09	4	2022-11-12
2.10	<공지/출석/Schedule> 페이지 DB연동	2022-11-15	22	2022-12-06
2.11	<메모장> 페이지 구성 및 DB연동	2022-12-07	3	2022-12-09
2.12	앱 수정 및 디자인	2022-11-18	22	2022-12-09
3	프로젝트 발표		-	
3.1	앱 개발 계획서 작성 및 제출	2022-10-06	7	2022-10-12
3.2	계획서 발표	2022-10-13	1	2022-10-13
3.3	중간 발표 준비	2022-11-13	4	2022-11-16
3.4	중간 발표	2022-11-17	1	2022-11-17
3.5	기말 발표 준비	2022-12-08	6	2022-12-13
3.6	기말 발표	2022-12-14	1	2022-12-14

4. 프로젝트 결과 시스템 구성

1) 실제 구현된 기능

(1) 로그인 페이지
- 앱을 처음 사용하는 사용자는 회원가입 버튼을 눌러 양식 작성 후 회원가입을 할 수 있다.
- 회원가입한 정보를 입력하여 로그인 할 수 있다.
- 비밀번호 찾기 버튼을 클릭하여 아이디, 전화번호, 생년월일을 올바르게 입력하면 비밀번호를 찾을 수 있다.

(2) 메인 홈 화면
- 로그인 후 이동되는 화면으로 메인 홈 화면에는 자신이 참여되어 있는 그룹들이 표시된다.
- 그룹들은 스터디와 팀 프로젝트 2가지로 분류되어 표시되며 클릭 시 그룹 페이지로 이동한다.
- 우측 상단의 종 모양 아이콘을 클릭하면 자신이 그룹에 참여 신청한 것에 대한 결과(수락/거절)을 확인할 수 있다.

(3) 그룹 페이지
- 메인 홈 화면의 자신이 참여되어있는 그룹 우측의 > 버튼을 클릭하여 이동할 수 있다.
- 그룹 내 상세 페이지에서는 상단 배너를 통해 중요공지를 확인할 수 있다.
- 상단 배너를 통해 자신의 다가오는 일정을 바로 확인할 수 있다.
- 상단 배너를 통해 업무의 마감기한을 바로 확인할 수 있다.
- 상단 배너 아래의 5가지(공지/출석/일정/업무/그룹정보) 버튼을 이용하여 각각 해당하는 기능들을 이용할 수 있다.
- 가장 아래에는 버튼이 총 3가지가 존재하는데 그룹 내에서의 역할에 따라 사용할 수 있는 버튼이 달라진다.
- 팀장은 그룹삭제, 참여신청 리스트 확인 2가지 버튼을 이용할 수 있다.
- 참여신청 리스트 버튼은 그룹에 대한 참여신청이 없는 경우 파란색 버튼으로 표시되고 참여신청이 있는 경우 초록색으로 표시된다.
- 그룹 삭제시 그룹에 참여되어있던 모든 팀원들의 활동과 팀원들의 그룹정보(삭제되는 그룹에 대한) 역시 삭제된다.
- 참여신청에 대해 팀장은 수락 또는 거절을 할 수 있다. (참여 신청이 들어오는 경우 버튼의 글씨가 '참여 신청 확인 요청'으로 바뀌고 버튼 색 또한 바뀌어서 신청이 들어 왔는지 확인할 수 있다.)
- 팀원은 그룹탈퇴 하나의 버튼만을 이용할 수 있다.
- 그룹 탈퇴시 탈퇴한 사용자가 그룹 내에서 활동한 내용(작성한 공지, 출석 기록, 일정 등)이 모두 삭제된다.

(4) 공지사항 페이지
- 메인 홈 화면에 표시되는 그룹 페이지로 이동한 뒤 공지사항 버튼을 클릭하면 공지 페이지로 이동한다.
- 공지 유형은 '일반'과 '중요'로 나뉜다.
- 공지사항을 등록하면 공지 메인 페이지에서 카드뷰를 통해 저장된 것을 확인할 수 있고, 카드 뷰의 '>' 버튼을 클릭하면 공지 내용을 볼 수 있다.
- 이미지 버튼을 클릭하여 공지 내용을 볼 때, 그 공지 작성자인 경우 '수정', '삭제' 버튼이 보여 공지를 수정하거나 삭제할 수 있다.
- 공지 유형이 '중요'인 경우 공지 메인 페이지에서 카드뷰의 제목 색깔이 빨간색으로 표시된다.
- 또한 '중요' 공지의 경우 '팀 메인'페이지 상단 배너에서 따로 확인할 수 있다.
- 상단 배너에서는 가장 최신의 중요 공지가 표시된다.
- 또한 상단 배너의 '>' 버튼을 클릭하면 중요 공지를 모아 볼 수 있다.
- 단, 중요 공지를 모아 보는 페이지에서는 공지 수정/삭제가 불가능하다.
- 공지 수정/삭제는 오로지 '공지 메인 페이지'를 통해 들어가서 할 수 있다.
- 공지 페이지는 최신의 공지부터 차례대로 표시된다.

(5) 출석 페이지

- 출석 페이지에 들어가면 '나의 출석 현황', '팀 출석 현황', 가장 최근 출석 체크에 대한 출석 여부 박스, '출석 코드 생성/등록'을 볼 수 있다.
- '출석 코드 생성/등록'은 페이지 맨 아래의 EditText에 코드를 작성하여 생성/등록할 수 있다.
- 팀장의 경우, 출석 코드 버튼이 '생성'으로 표시되고, 출석 코드 숫자 6자리를 입력하여 코드를 생성할 수 있다. 생성된 코드 번호는 가장 최근의 출석 여부 박스에서 볼 수 있고, 이는 팀장만 확인 가능하다. 이때 박스 제목에 코드를 등록한 날짜와 시간이 표시된다. (가장 최근에 출석 코드가 생성된 날짜 표시)
- 팀원의 경우, 출석 코드 버튼이 '등록'으로 표시된다. 팀장에게 코드 6자리를 받아 입력하고 버튼을 누르면 출석 처리된다.
- 자신의 최근 출석 여부는 박스에서 확인할 수 있다.
- '팀 출석 현황'의 경우, 팀원 전체의 출석 퍼센트를 볼 수 있다. 팀장의 경우 '팀 출석 현황' 글씨 옆 출석 관리 버튼을 통해 팀원들의 출석 여부를 수정할 수 있다.
- 출석 관리 버튼을 클릭하면 '출석 관리'페이지로 연결된다. 팀장을 제외한 팀원들의 가장 최근의 출석 코드에 대한 출석 여부를 변경하는 라디오버튼을 볼 수 있다. 라디오 버튼은 '출석', '지각', '결석'으로 구성되어 있고, 라디오버튼 체크 변경 시 바로 팀원의 출석 여부가 변경된다.
- 팀장이 '출석 관리' 페이지에서 팀원의 출석 여부를 모두 확인하면 맨 밑에 '모든 출석 확인 완료' 체크를 누르고 확인 버튼을 누르면 팀장의 출석 여부가 '출석'으로 바뀐다. 또한 더 이상 팀원들의 출석 여부를 수정할 수 없게 된다. 따라서 체크 후 확인 버튼 클릭 시 경고창을 띄워 더 이상 수정할 수 없음을 알린다.
- 팀장의 출석의 경우 팀원들의 출석 여부 확인을 함으로써 출석 처리된다. 팀장이 다음 출석 코드 생성 시까지 출석을 확인하지 않았다면 팀장 출석 여부는 결석 처리된다.
- 각자의 출석률은 자신이 그룹에 참여한 이후에 생성된 출석 코드 개수 기준으로 계산된다.
- '나의 출석 현황'의 경우 출석/지각/결석의 퍼센트를 볼 수 있다. 팀 활동을 열심히 할 수 있도록 격려하기 위해 출석률에 따라 메시지를 출력한다. (출석률이 90% 이상인 경우, '팀 활동에 적극 참여 중! 고마운 사람:D', 70% 이상인 경우, '좋아요! 좀 더 적극적인 참여 기대할게요;)', 50% 이상인 경우, '적극적 참여 부탁합니다!', 50% 미만인 경우, '참여가 부진해요:(')
- 출석 페이지 입장 시 팀원의 출석/지각/결석 횟수가 토스트 메시지로 출력된다.

(6) Schedule 페이지

- 스케쥴 페이지에는 팀 일정, 팀원의 일정을 표시할 수 있다.
- Calendar 글씨 옆 +버튼을 클릭하면 일정을 작성할 수 있다.
- 일정의 경우 개인 일정은 팀원 모두가 작성할 수 있지만, 그룹 일정은 팀장만 설정 가능하다.
- 일정명, 일정 구분, 일정 날짜, 시작 시간, 종료 시간을 설정하고 등록하면 일정 등록이 완료된다.
- 일정 등록 시 같은 시간대에 등록되어있는 일정이 있으면, 겹치는 일정 중 제일 첫 번째 것의 주인과 일정 내용이 써진 경고 dialog가 생성된다. 겹치는 일정이 있어도 일정을 등록하고자 하면 dialog의 확인 버튼을 누르면 일정이 추가된다.
- 그룹 일정의 경우 일정 주인은 '그룹'으로 표시되고, 일정 유형이 개인일 경우 일정 주인의 이름이 표시된다.
- 스케쥴 페이지의 캘린더 뷰의 날짜를 선택하면 캘린더 뷰 밑에 해당 날짜의 일정이 표시된다.
- 일정 수정/삭제는 개인 일정의 경우, 일정 주인만 수정/삭제가 가능하고, 그룹 일정의 경우 팀장만 수정/삭제가 가능하다.
- 그룹일정이 새로 추가된 경우, Schedule 페이지 맨 위 상단에 추가된 일정이 표시된다
- 그룹 메인 페이지의 상단의 '다가오는 일정'에서 다가오는 자신 또는 그룹의 일정이 표시된다. 그룹 일정인 경우 빨간색으로 표시된다.

(7) Work 페이지

- Work 페이지에서 팀장은 우측 상단의 + 버튼을 클릭하여 Task를 등록할 수 있다.
- Task에는 내용을 적고 기간을 표시할 수 있다.
- Task 내에서 팀장이 ToDo를 작성함으로써 각 팀원들에게 역할을 분담해줄 수 있다.
- 팀원이 역할 수행을 완료하면 체크 표시를 하여 완료되☝다고 표시해줄 수 있다.

- ToDo는 수정할 수는 없고 삭제만 가능하다.
- Task를 진행하는 과정에서 발생하는 문제나 의견을 공유하기 위해 Comment 기능을 이용할 수 있다.
- Comment는 ToDo와 다르게 모든 팀원이 작성할 수 있다.
- Comment도 ToDo와 같이 수정은 불가능하고 삭제만 가능하다. (자신의 comment만 삭제가 가능하다.)

(8) 그룹 정보 페이지
- 그룹정보 페이지에서는 해당 그룹의 이름, 카테고리, 인원수 등의 정보를 확인할 수 있다.
- 추가적으로 그룹에 참여되어있는 팀원들의 리스트를 볼 수 있다.
- 팀장은 하단에 보이는 팀장 역할 변경 버튼을 클릭한 후 리스트에서 팀원을 클릭함으로써 팀장 역할을 다른 팀원에게 넘겨줄 수 있다.

(9) 참여 페이지
- 참여 페이지에서는 하단의 버튼을 클릭하여 직접 그룹을 생성할 수 있다.
- 그룹을 생성한 사람의 역할이 팀장이다. (팀장 역할은 팀 메인페이지의 그룹정보 페이지이서 팀장만 변경 가능)
- 다른 사용자가 만들어놓은 그룹에 참여 신청을 할 수 있다.
- 참여하는 사용자의 역할은 팀원으로 지정된다.
- 그룹은 공개 그룹과 비공개 그룹으로 나뉘며 비공개 그룹의 경우 그룹코드를 입력하여 일치하는 경우만 참여신청이 가능하다.
- 그룹은 스터디와 팀 프로젝트 2가지 종류로 나뉘며 또 다시 6개의 카테고리(토익/자격증/공모전/시험공부/팀플/기타)로 나뉜다.
- 상단의 검색창을 이용하여 그룹의 이름을 입력하여 그룹을 검색할 수 있다.
- 검색창 아래의 카테고리를 입력하면 해당 카테고리에 해당하는 그룹들만을 편리하게 검색할 수 있다.
- 그룹 중 현재 인원이 제한인원에 도달한 그룹은 참여가능한 그룹 리스트에 보여지지 않는다.
- 자신이 이미 참여중인 그룹에는 참여신청 버튼이 보이지 않는다.
- 한 명의 사용자는 최대 4개의 그룹까지 참여할 수 있다.
- 이미 4개의 그룹에 참여중인 사용자는 그룹 참여 버튼이 보이지 않는다.

(10) memo 페이지
- 하단바의 메모 페이지는 개인이 기록하고자 하는 것을 남길 수 있다. 기록한 메모는 작성한자만 볼 수 있다.
- 메모 페이지에 들어가면 상단의 '+' 버튼을 통해 메모를 추가할 수 있다.
- 제목을 작성하지 않을 경우, 메모 등록 시 '제목 없음'으로 자동 저장된다.
- 내용을 작성하지 않을 경우, 오류 메시지가 뜨며, 메모를 저장할 수 없다.
- 수정 삭제 또한 가능하다.
- 메모는 가장 최근에 등록한 것부터 표시된다. (단, 메모 표시 순서는 처음 메모를 생성한 시간에 따른 순서이다. 수정했다 해서 먼저 표시되지 않는다.)
- 메모 페이지는 하단바 구성 요소로 뒤로가기 버튼 한 번 클릭 시, 한 번 더 클릭하면 앱이 종료됨을 알리는 토스트 메시지가 뜬다. 뒤로가기 버튼 두 번 클릭 시 앱이 종료된다.

(11) 마이페이지
- 하단 바의 가장 오른쪽 버튼을 클릭하면 마이페이지로 이동한다.
- 회원정보수정 버튼을 클릭하여 자신의 정보를 수정할 수 있다. (아이디는 변경이 불가능하다.)
- 회원탈퇴 버튼을 클릭하여 탈퇴할 수 있다.
- 회원 탈퇴를 할 경우 사용자의 정보들뿐만 아니라 사용자가 앱에서 활동한 내용들(그룹 내에서 작성한 공지, 일정 등)도 모두 삭제된다.
- 자신이 참여되어있는 그룹 중에서 자신이 팀장 역할을 하고 있고 자신 외에 팀원이 있는 즉, 자신이 팀장인 2인 이상의 인원을 가지는 그룹이 있는 경우 회원 탈퇴가 불가능하다. (이 경우 그룹 페이지의 그룹정보 페이지에서 팀장 역할을 다른 팀원에게 넘겨준 후 탈퇴해야한다.)

- 그 외의 경우에는 회원탈퇴가 가능하다. (참여되어있는 그룹이 있는데 역할이 팀원인 경우, 역할이 팀장이지만 그룹에 자기자신 한명밖에 없는 경우)

2) 수정(축소) 또는 삭제된 기능

(1) 액티비티의 최소화
- 초기에 개발을 진행할 때는 모든 화면을 액티비티로만 구성했기 때문에 프로젝트가 많은 수의 액티비티로 구성되었으나 현재는 대화상자를 이용할 수 있는 부분은 대화상자로 대체하였다.

(2) 공지 확인 여부 삭제
- 사용자가 공지를 읽□는지 확인여부를 표시할 수 있는 체크박스 기능을 삭제하였다.

(3) 출석 방식 변경
- 기존의 QR코드를 이용하는 방법에서 숫자 6자의 코드를 이용하는 방법으로 변경

(4) 캘린더뷰 일정 표시 방식 변경
- 캘린더뷰에 일정을 보여주거나 아니면 일정이 있는 날짜에는 별도로 표시를 해줄 계획이□으나 커스텀 적용 과정에서 문제가 발생해서 캘린더뷰에서 날짜를 선택하면 해당 날짜의 일정들이 하단의 레이아웃에 표시되는 방식으로 표시방법을 변경하였다.

(5) 그룹 화상채팅, 보이스채팅 기능 삭제
- 시간상의 문제도 있고, 기술적으로도 부족한 부분이 많다고 생각하여 그룹 화상채팅과 보이스채팅 기능을 구현하지 않기로 결정하였다.

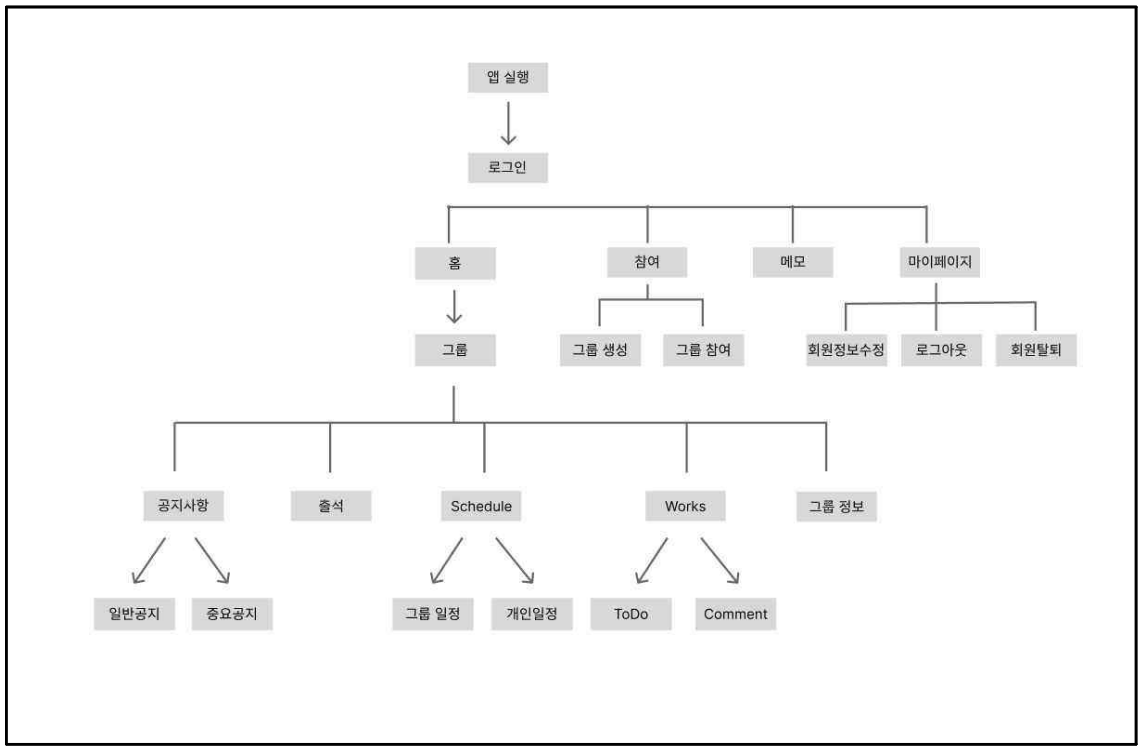
(6) 그룹채팅 삭제 후 메모장 기능 추가
- 그룹 내의 Task페이지에서의 Comment 기능이 그룹채팅과 거의 유사하기 때문에 굳이 그룹 채팅이 필요하지 않을 것으로 판단하여 그룹채팅 기능을 삭제하였다.
- 기존의 그룹채팅 페이지에는 대신 메모장 기능을 추가하였다.

(7) 그룹 내 자료 등록 및 공유 기능 삭제
- 그룹 내에서 ppt, 이미지 등의 자료를 공유하는 기능을 삭제하였다.
- txt외의 ppt 등의 파일을 어떻게 저장하고 관리해야하는지 잘 모르기 때문에 이미지 파일만이라도 저장하고 관리할 수 있도록 개발할 것을 계획하였으나 결국에는 시간적인 문제로 구현하지 못하게 되□다.

(8) 마이페이지의 내가 쓴 comment를 모아서 보여주는 기능 삭제
- 중요한 기능이 아니라고 판단하여 삭제하였다.

5. 시스템 흐름도

1) 흐름도



앱 STeam이 작동하는 간략한 흐름은 위의 그림과 같다. 회원가입을 마친 사용자는 그룹을 생성하거나 그룹에 참여할 수 있고, 그룹에 참여된 사용자는 그림에서 보이는 것과 같이 공지사항, 출석, Schedule, Works 등의 기능을 그룹 내에서 사용할 수 있다. 각 기능들에 대한 자세한 내용은 아래의 결과물에서 확인할 수 있다.

6. 시스템 구현 결과물 및 사용자 지침서

1) 로그인 및 회원가입

(1) 앱 사용이 처음인 경우

- 로그인 화면에서 회원가입 버튼을 눌러 아이디, 비밀번호 등의 개인정보를 입력한 후 회원가입을 할 수 있다.

① 아이디와 비밀번호는 6~10자 영문 숫자 조합으로 입력할 수 있다.

② 이름은 최대 10자까지 입력할 수 있다.

③ 전화번호의 경우 '-'를 제외한 숫자만 입력할 수 있다. (ex. 01012345678)

④ 이메일은 형식에 맞게 입력할 수 있다.

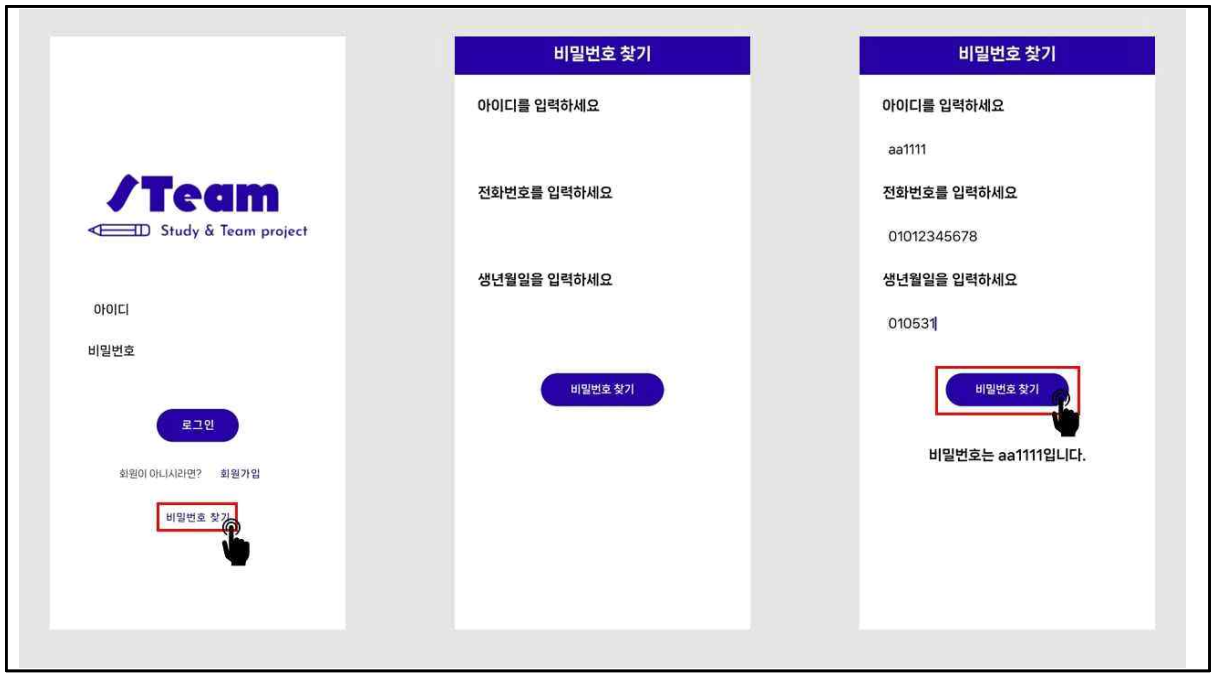
⑤ 생년월일은 6자리로 입력할 수 있다. (ex. 2001년 1월 1일 -> 010101)

- 입력받은 정보를 검사 후 DB에 저장하는 코드

```
String id = id_EditText.getText().toString().trim();
String pw = pw_EditText.getText().toString().trim();
String pw_check = pw_check_EditText.getText().toString().trim();
String name = name_EditText.getText().toString().trim();
String phone = phone_EditText.getText().toString().trim();
String email = email_EditText.getText().toString().trim();
String birth = birth_EditText.getText().toString().trim();
// 누락된 정보가 없는지 확인하는 과정
// 누락된 정보가 있는 경우 토스트 메세지 출력
if(!(id.length()>=6 && id.length()<=10)) {
    LayoutInflater inflater = getLayoutInflater();

    View layout = inflater.inflate(R.layout.toastcustom,
            (ViewGroup) findViewById(R.id.toastLayout));

    TextView text = layout.findViewById(R.id.toastText);

    Toast toast = new Toast( context: SignUpActivity.this);
    text.setText("아이디를 입력하세요");
    toast.setDuration(Toast.LENGTH_SHORT);
    toast.setView(layout);
    toast.show();
    return;
} else if (!(pw.length()>=6 && pw.length()<=10)){
```

```
sql = "SELECT NAME FROM CUSTOMER WHERE custID = '" + id + "'";
cursor = database.rawQuery(sql,  selectionArgs: null);

// 다른 사용자가 아이디를 사용중인 경우
if(cursor.getCount() != 0) {
    // 토스트 메세지 출력
    LayoutInflater inflater = getLayoutInflater();

    View layout = inflater.inflate(R.layout.toastcustom,
            (ViewGroup) findViewById(R.id.toastLayout));

    TextView text = layout.findViewById(R.id.toastText);

    Toast toast = new Toast( context: SignUpActivity.this);
    text.setText("이미 존재하는 아이디입니다.");
    toast.setDuration(Toast.LENGTH_SHORT);
    toast.setView(layout);
    toast.show();
    // 사용 가능한 아이디인 경우
} else {
    // 미리 만들어놓은 사용자 추가 함수를 이용하여 입력받은 값들의 정보를 가지
    helper.addCustomer(id, pw, name, phone, email, birth,  gl:
    // DB에 정보를 추가한 후 회원가입이 완료되었다는 토스트메세지 출력 후 다시
    LayoutInflater inflater = getLayoutInflater();

    View layout = inflater.inflate(R.layout.toastcustom,
            (ViewGroup) findViewById(R.id.toastLayout));

    TextView text = layout.findViewById(R.id.toastText);
```

(2) 회원가입은 되어 있지만 비밀번호를 잊어버렸을 경우



- 비밀번호를 잊은 경우에는 로그인 화면에서 비밀번호 찾기 기능을 이용하면 된다.
- 비밀번호 찾기 화면에서 아이디, 전화번호, 생년월일을 입력하면 해당 정보와 일치하는 사용자의 비밀번호를 알려준다.

- 입력받은 정보로 비밀번호를 검색하여 찾는 코드

```
btnFindPW.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        // EditText로 입력받은 정보들을 String으로 변환 후 공백을 제거하여 변수로 저장
        String id = edtID.getText().toString().trim();
        String phone = edtPhone.getText().toString().trim();
        String birth = edtBirth.getText().toString().trim();
        // 입력 받은 정보가 하나라도 없는 경우
        if(id.length() == 0 || phone.length() == 0 || birth.length() == 0){
            // 정보를 모두 입력하라고 토스트 메세지 출력
            LayoutInflater inflater = getLayoutInflater();

            View layout = inflater.inflate(R.layout.toastcustom,
                    (ViewGroup) findViewById(R.id.toastLayout));

            TextView text = layout.findViewById(R.id.toastText);

            Toast toast = new Toast( context passwordActivity.this);
            text.setText("정보를 모두 입력하세요");
            toast.setDuration(Toast.LENGTH_SHORT);
            toast.setView(layout);
            toast.show();

        }
        // 정보가 제대로 입력된 경우
        else{
            Cursor cursor = myDB.getPW(id, phone, birth);
            // 아이디와 전화번호, 생년월일이 일치하는 경우 비밀번호 표시
            if(cursor.getCount() != 0){
                cursor.moveToNext();
                String pw = cursor.getString( i: 0);
                txtPW.setText("비밀번호는 " + pw + "입니다.");
            }
```

(3) 회원가입이 되어 있는 경우

- 로그인 화면에서 아이디와 비밀번호를 입력하여 로그인하면 된다.
- 가장 우측 사진은 로그인 성공 시 뜨는 초기 화면이자 해당 앱의 메인 화면(홈)이다.

- 입력되었는지 확인

```
if(id.length() == 0 || pw.length() == 0) {
```

- 회원인지 확인

```
// 입력한 아이디가 존재하는지(회원인지) 검색하기 위한 sql문
sql = "SELECT Name FROM CUSTOMER WHERE custID = '" + id + "'";
cursor = database.rawQuery(sql, selectionArgs: null);
// Cursor의 개수가 1개가 아니면, 즉 가입된 아이디가 없으면
if(cursor.getCount() != 1) {
    // 토스트 메시지 출력
```

- 로그인 성공

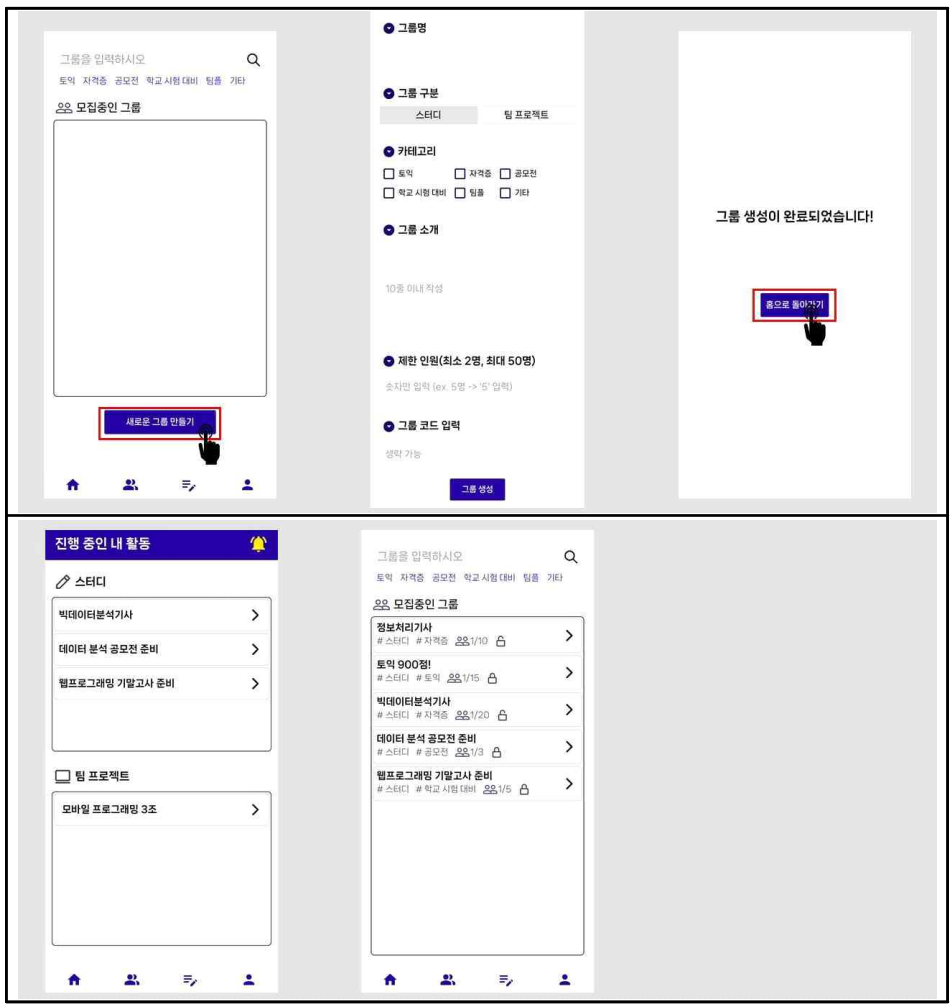
```
Intent intent = new Intent(getApplicationContext(), MainActivity.class);
// 현재 사용자의 아이디를 넘겨줌
// 앱을 사용하면서 현재 사용자와 관련된 데이터를 저장하거나 또는 불러오기 위해 아이디가 필요
intent.putExtra( name: "ID1", id);
intent.putExtra( name: "ID2", id);
// 로그인 후 메인 화면으로만 아이디를 넘겨주는 것이 아니라
// 앱이 작동하기 위해 거의 모든 액티비티에서 아이디가 필요하기 때문에
// 대부분의 Intent에 아이디를 덧붙여 넘겨줌
startActivity(intent);
finish();
```

2) '참여' 페이지 (Join.java)

(1) '참여' 페이지 개요

- 참여 페이지에서는 그룹 생성, 참여 및 신청이 가능하다.

(2) 그룹 생성하기



- 하단 바의 두 번째 버튼을 클릭하면 '참여' 페이지로 이동한다.
- 해당 페이지에는 앱 사용자들이 생성한 그룹들의 리스트를 보여준다.
- 원하는 그룹이 없다면 '새로운 그룹 만들기' 버튼을 클릭하여 원하는 그룹을 생성할 수 있다.
-그룹 생성 양식은 ① 그룹명 ② 그룹 구분 ③ 카테고리 ④ 제한 인원 ⑤ 그룹 코드 입력으로 이루어져 있다.

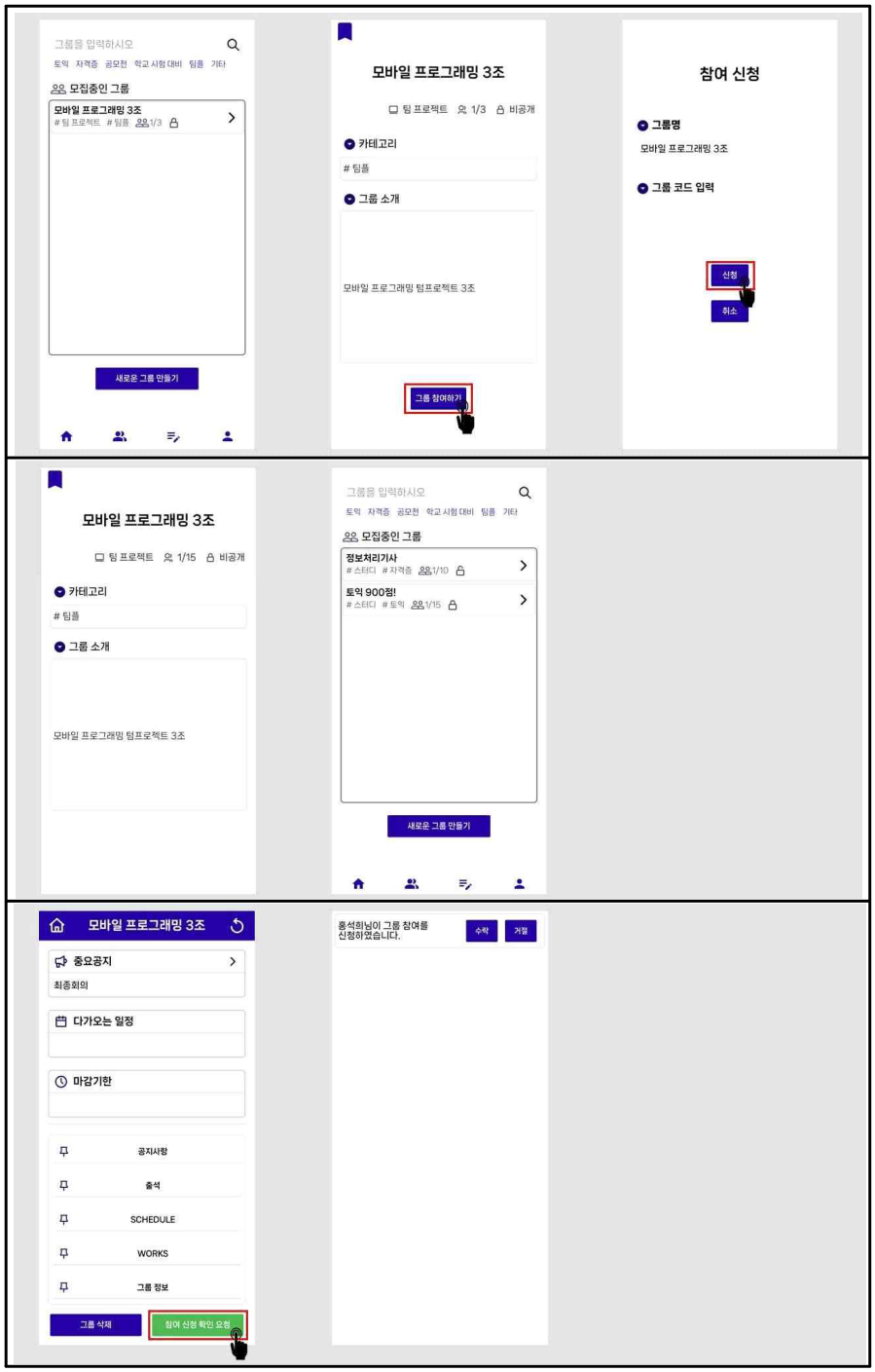
① 그룹명은 필수 입력 사항이다.
② 그룹 구분은 '스터디'와 '팀 프로젝트'로 분류되어 있다. (필수 입력 사항)
③ 카테고리는 총 6개로 이루어져 있으며, 토익/자격증/공모전/학교 시험 대비/팀플/기타로 이루어져 있다. (필수 입력 사항)
④ 그룹 소개는 10줄 이내로 작성할 수 있으며, 필수 입력 사항이다.
⑤ 제한인원은 최소 2명, 최대 50명까지 지정할 수 있으며, 숫자만 입력할 수 있다. (필수 입력 사항)
⑥ 그룹코드는 선택 입력사항이다. 그룹코드를 입력하지 않으면 공개 그룹, 그룹코드를 입력하면 비공개 그룹이 된다.

- 양식에 맞게 그룹 신청서를 작성하고 생성하면 완료 화면이 뜬다. 그러면 그룹 생성이 성공적으로 완료된다.
- 사용자 1인당 생성 및 가입할 수 있는 그룹의 개수는 총 4개이다.
- 본인이 4개의 그룹에 참여하게 되면 사용자는 참여 페이지 하단에 있던 '그룹 생성' 버튼 및 참여 버튼이 사라져 더 이상 그룹에 참여할 수 없게 된다.

- 그룹 생성하는 코드 (DB에 그룹 정보 추가하는 코드)

```
if(rStudy1.isChecked()){
    type="스터디";
}else if(rTeamplay1.isChecked()){
    type="팀 프로젝트";
}
gnum=1; // 그룹을 생성한 초기에는 자신밖에 팀원이 없기 때문에 인원수를 1로 지정
num = Integer.toString(gnum);
people = num + "/" + edt[2]; // '현재 인원수/ 제한인원'

String check=""; // 카테고리 저장하기 위한 변수
// 체크된 카테고리 내용들을 문자열로 만듦
for(int i=0; i<chkCategory.length;i++){
    if(chkCategory[i].isChecked()){
        check=check + "# "+chkCategory[i].getText().toString();
    }
}

// DB에 저장할 그룹코드
String gcode = edt[3];
if(edt[3].trim().equals("")){
    // 그룹코드가 없는 경우는 no로 지정
    gcode="no";
}
// 현재 시간을 문자열로 저장
String currentTime = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss").format(new Date());
// 데이터베이스 객체 생성
MyDatabaseHelper myDB = new MyDatabaseHelper( context: MakeGroup.this);
// DB에 데이터를 저장
// 그룹을 추가할 때 그룹 아이디 속성이 AUTO INCREMENT로 자동 증가되기 때문에
// 그룹을 생성한 사용자 속성에 어떤 그룹 아이디를 저장해야하는지 모름
// 이 자동으로 저장된 그룹 아이디를 알아내기 위한 용도로 뒤에 선언한 현재시간(그룹 생성 시간)을 이용한다.
myDB.addGroup(edt[0].trim(), edt[1], type, check, numPeople, gnum, gcode, currentTime);
```

(2) 그룹 참여 및 신청하기



- 모집 중인 그룹에서 각 그룹의 '>' 버튼을 클릭하면 해당 그룹의 정보를 확인할 수 있다.
- 해당 그룹에 참여되어 있지 않은 경우, 하단에 '그룹 참여하기' 버튼이 보이는데, 해당 버튼을 누르면 신청 가능 페이지로 이동하게 된다.

  ① 비공개 그룹: 그룹 코드 입력란에 알맞은 그룹 코드를 입력해야 신청할 수 있다.

  ② 공개 그룹: 그룹 코드 입력란이 비활성화되어 있어 신청 버튼을 누르면 바로 신청할 수 있다.

*유의점: 신청 버튼을 누른다고 해서 바로 그룹에 가입되는 것은 아니다. 해당 그룹의 팀장이 수락해야만 그룹활동이 가능하다.

* 그룹 신청을 하면 해당 그룹 팀장의 메인 페이지 하단에 초록색으로 '참여 신청 확인 요청' 버튼이 뜨는데, 해당 페이지에 들어가서 수락을 해야만 그룹 가입이 완료된다.

- 참여 페이지의 '모집 중인 그룹'에는 내가 속해 있는 그룹도 볼 수 있는데, 이때는 상세 페이지에 들어가지나 '그룹 참여하기' 버튼이 없어 신청 불가능하다. 즉, 그룹에 대한 정보만 확인할 수 있다. 거절된 경우에는 그룹 참여가 불가능하다.
- 그룹의 제한 인원에 도달한 그룹의 경우 참여 페이지의 '모집 중인 그룹' 리스트에 보이지 않는다.

- 그룹 참여신청 코드

```
btnApplyComplete1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        // 공개 그룹인 경우 (그룹코드가 없는 경우)
        if(inIntent.getStringExtra( name: "gCode").equals("no")){
            // DB에 참여신청 데이터를 저장 후 참여신청 완료 페이지로 이동
            myDB.addApply(gid,id);
            Intent intent = new Intent(getApplicationContext(), ApplyForJoinCompleted.class);
            intent.putExtra( name: "ID",id);
            startActivity(intent);
            finish();
```

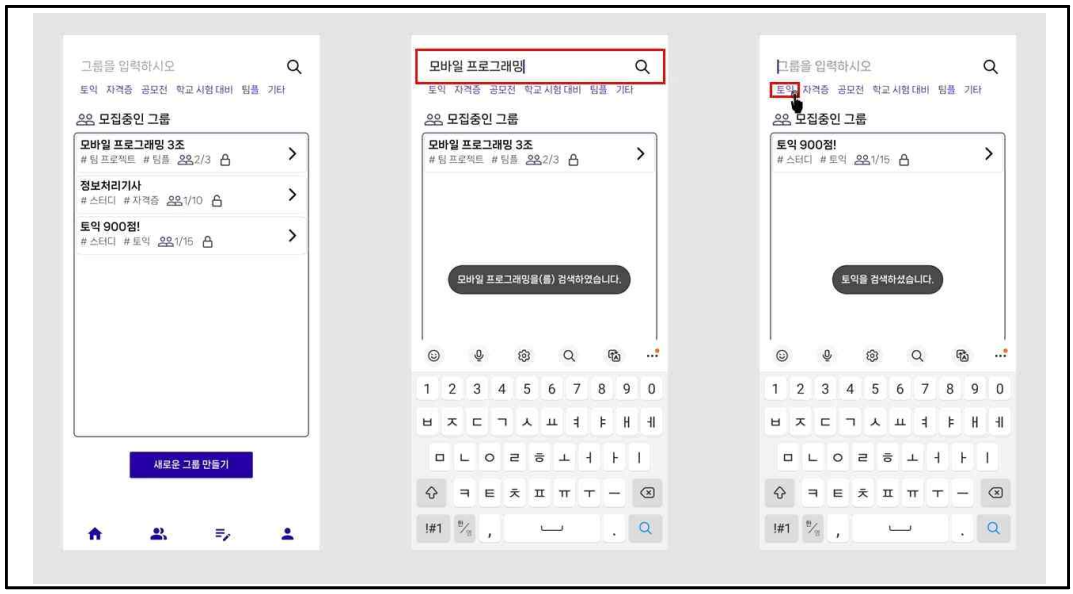
- 이미 참여중인 그룹인 경우와 4개의 그룹에 참여중이면 참여 버튼이 보이지 않도록 하는 코드

```
int mygroup = 0;
int gnum=0;
for(int k=0;k<4;k++){
    final int index;
    index = k;
    int g1234 = cursor.getInt(index);
    // 현재 그룹에 사용자가 이미 참여되어있으면
    if(gid == g1234){
        // 변수 값을 1로 변경
        mygroup=1;
    }
    // 현재 사용자가 참여되어있는 그룹이 있는 경우
    if(g1234 != -1){
        // 1씩 증가
        gnum++;
    }
}
// 만약 현재 사용자가 이미 참여중인 그룹이거나 또는 이미 4개의 그룹에 참여중인 경우
if(mygroup==1 || gnum==4){
    // 참여신청 버튼이 보이지 않도록
    btnApply1.setVisibility(View.GONE);
```

- 그룹장이 참여신청에 대해 수락하는 코드 (신청한 사용자의 빈 그룹 속성, 역할속성을 변경 해준다.)

```
Cursor ccursor = myDB.customerGroups(cid);
ccursor.moveToNext();
Cursor numcursor = myDB.getGNum(gid);
numcursor.moveToNext();

for(int i=0;i<4;i++){
    final int index;
    index = i;
    if(ccursor.getInt(index)==-1){
        if(index == 0){
            myDB.updateCustomer0(cid, gid, r: "팀원");
            myDB.modifyGNum(gid, gnum: gnum+1);
            myDB.deleteApply(cid, gid);
            myDB.addConsent(gid, cid);
            LayoutInflater inflater = getLayoutInflater();

            View layout = inflater.inflate(R.layout.toastcustom,
                    (ViewGroup) findViewById(R.id.toastLayout));

            TextView text = layout.findViewById(R.id.toastText);

            Toast toast = new Toast( context: ApplyListActivity.this);
            text.setText("수락하였습니다");
            toast.setDuration(Toast.LENGTH_SHORT);
            toast.setView(layout);
            toast.show();
            break;
```

(3) 그룹 검색하기



- 참여 페이지 상단의 검색창을 통해 그룹 검색을 할 수 있다. 입력한 키워드를 포함하고 있는 그룹들의 리스트를 제공한다.
- 검색창 아래의 카테고리를 클릭하면 해당 카테고리에 포함된 그룹들의 리스트를 제공한다.

- 카테고리 이용 검색

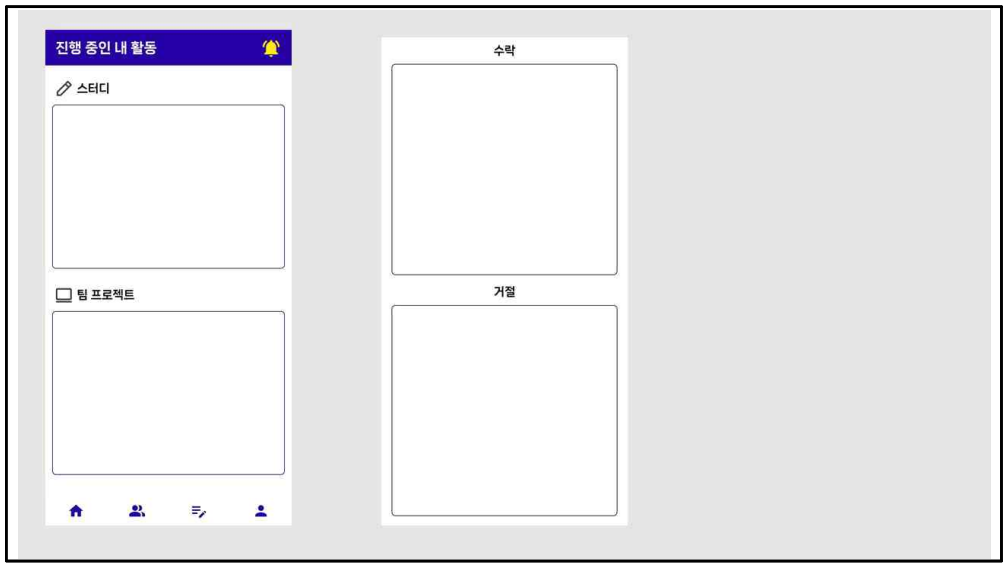
```
gCategory = gcategory.get(i);
gCapacity = gcapacity.get(i);
gNum = gnum.get(i);
gCode = gcode.get(i);
currentgroupid = gid.get(i);

// 카테고리 검색 함수에서는
// 카테고리가 포함되었는지 확인하기 위한 과정이 필요

// 여러개의 카테고리를 StringTokenizer로 각각으로 나누어 조작
int k=0;
StringTokenizer st = new StringTokenizer(gCategory,
// 여러 카테고리들 중 동일한 카테고리가 있으면 k값 1로 변경
while(st.hasMoreTokens()){
    if(category.contains(st.nextToken())){
        k=1;
        break;
    }
}

if(gCode.equals("no")){
    lockOrUnlock="공개";
}else{
    lockOrUnlock="비공개";
}
// 현재 인원수가 제한인원과 같은 그룹들은 참여할 수 없으니 보이지 않도록 하
if(gNum==gCapacity){
}
else{
    // 검색한 카테고리를 포함하는 그룹들만을 보여줌
    if(k==1){
        addCard(gName, gType, gCategory, gCapacity, gNum, lo
    }
}
```

- 검색어 이용 검색

```
gName = gname.get(i);
gType = gtype.get(i);
gContent = gcontent.get(i);
gCategory = gcategory.get(i);
gCapacity = gcapacity.get(i);
gNum = gnum.get(i);
gCode = gcode.get(i);
currentgroupid = gid.get(i);
// 그룹코드가 없으면 공개
if(gCode.equals("no")){
    lockOrUnlock="공개";
}
// 그룹코드가 있으면 공개
else{
    lockOrUnlock="비공개";
}
// 검색어가 포함된 그룹들만 보여주기 위한 변수조작
// 초기값 0으로 설정
int n=0;
// 검색어가 포함되어있으면 k=1로 변경
if(gName.contains(searchWord)){
    n=1;
}

// 현재 인원수가 제한인원과 같은 그룹들은 참여할 수 없으니 보이지
if(gNum==gCapacity){
}
// 이 경우의 그룹들만 addCard를 이용하여 layout에 추가
else{
    // k값이 1인 경우만 카드뷰를 추가
    if(n==1){
        addCard(gName, gType, gCategory, gCapacity, g
```

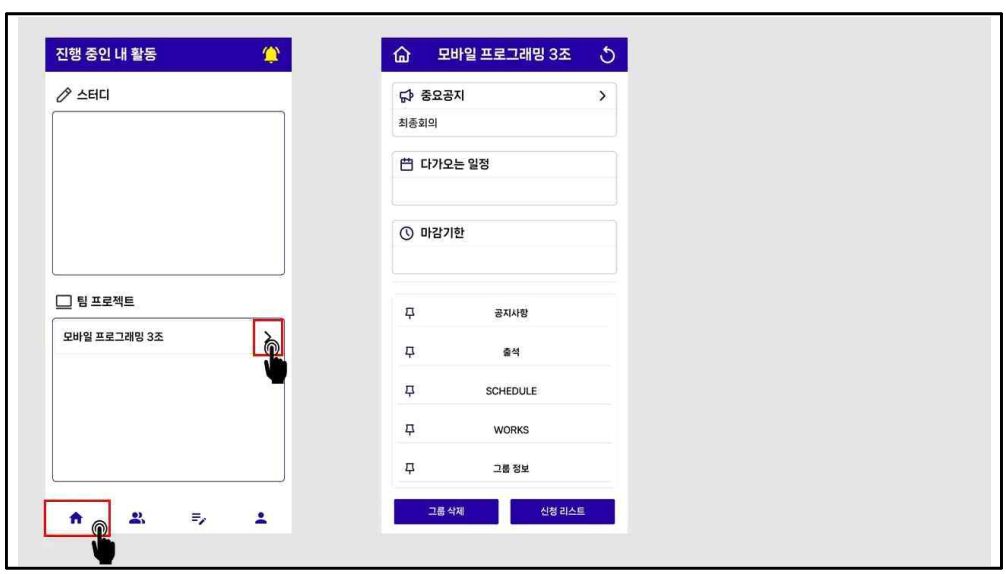
3) '홈' 화면

(1) '홈' 화면 개요



- 로그인 후 이동되는 초기 화면이자 해당 앱의 메인 화면 (홈)이다.
- '홈'에서는 본인이 참여하고 있는 그룹을 스터디/팀 프로젝트 그룹으로 분류하여 보여준다.
-우측 상단의 종 모양 아이콘을 클릭하면 자신이 그룹에 참여 신청한 것에 대한 결과를 수락과 거절로 나누어 보여준다.

(2) '메인 홈' 페이지 주요 기능 소개



-그룹 생성 및 참여가 완료되면 홈 화면에 자신의 그룹이 추가된 것을 볼 수 있다.
-진행 중인 내 활동의 각 그룹의 우측 '>' 버튼을 클릭하면 해당 그룹의 메인 페이지로 이동할 수 있다.

(3) 그룹 메인 페이지 기능 소개

- 상단에 3개의 배너가 있다.

① '중요 공지' 배너에는 공지사항 내 중요 공지로 설정한 공지사항들을 볼 수 있다. 중요 공지가 여러 개인 경우, 가장 최근의 중요 공지를 보여주고 있다.

② '다가오는 일정'은 Schedule 내에서 등록한 개인 및 그룹 일정을 보여주고 있다.

③ '마감기한' WORKS 내에서 등록한 Task의 마감기한을 보여준다.

- 상단 배너 아래에는 그룹 내의 주요 기능 5개를 이용할 수 있는 버튼이 존재한다.

① 공지사항 ② 출석 ③ SCHEDULE ④ WORKS ⑤ 그룹 정보가 존재하며 자세한 설명은 아래에서 하고자 한다.

- 팀장의 경우 하단의 ① 그룹 삭제 ② 신청 리스트 버튼을 이용할 수 있다.
- 팀원의 경우 하단의 그룹 탈퇴 버튼을 이용할 수 있다.

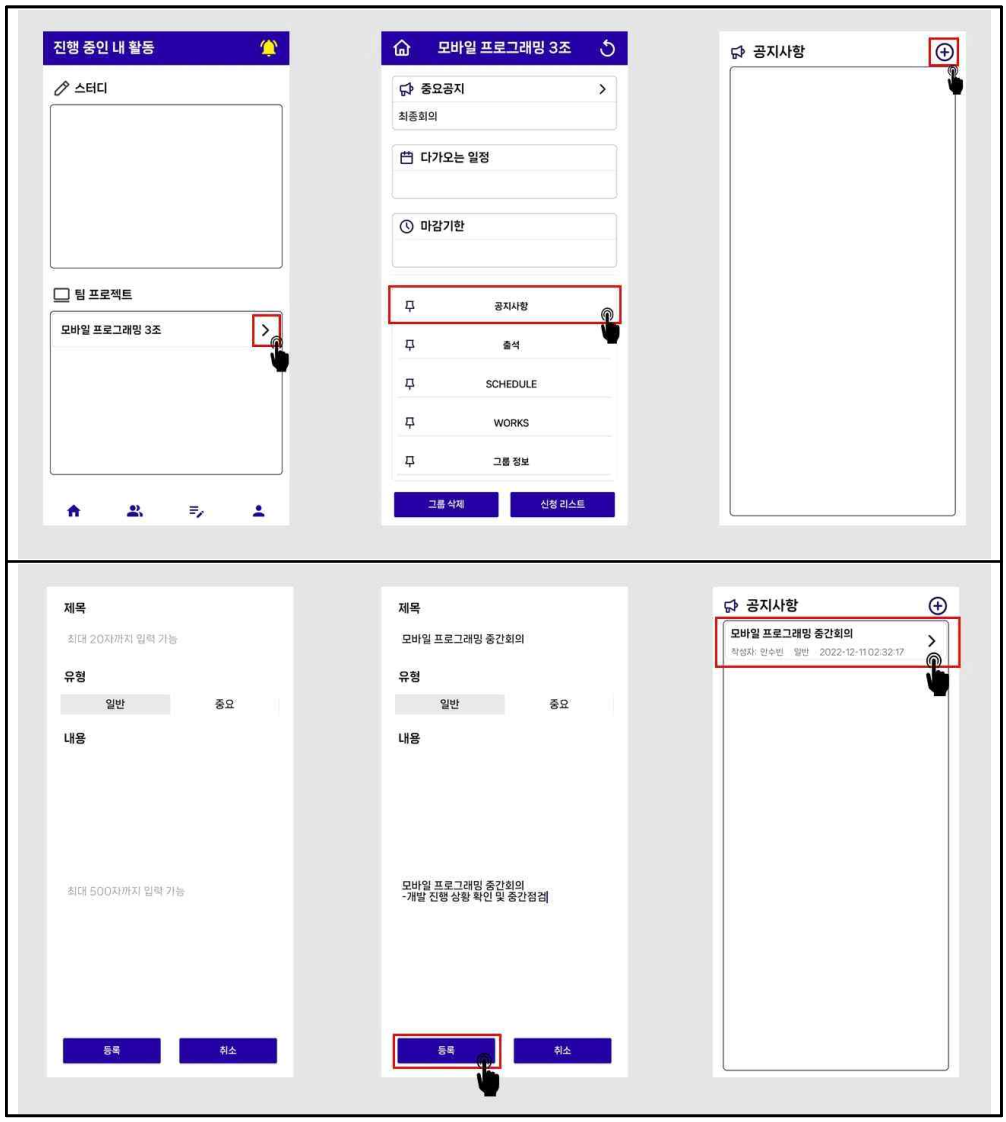
- 상단 배너의 공지 표시 코드
코드가 너무 길어서 사진은 생략하였고 간략히 설명하자면, 가장 먼저 intent로 넘겨받은 그룹의 id를 이용하여 현재 그룹의 공지, Schedule, Task 3가지 정보를 가져온다. 공지에 대해서는 가장 최근에 작성된 중요공지의 이름을 가져와서 텍스트뷰에 setText()로 적용해주고 Schedule과 Task에 대해서도 마감기한을 비교하여 제일 가까운 마감 일정을 표시해주는 내용의 코드이다.

- 우측 상단 종 모양을 누르면 나오는 페이지에서 신청 결과를 보여주는 코드

```
// 현재 사용자가 참여신청한 내역에 대한 결과(수락/거절)들을 모두 불러옴
Cursor ccursor = myDB.getConsent(id);
Cursor dcursor = myDB.getDeny(id);

if(ccursor != null){
    // 수락 내역을 하나씩 카드뷰로 만들어서 보여주는 과정으로
    // 카드뷰를 추가하는 과정들은 앞서 설명한 코드와 동일하다
    while(ccursor.moveToNext()){
        int gid = ccursor.getInt( i: 0);
        Cursor cursor = myDB.groupnames(gid);
        cursor.moveToNext();
        String txt = cursor.getString( i: 0)+" 그룹에서 참여 신청을 수락하였습니다.";

        addCardConsent(txt, gid);
    }
}
if(dcursor != null){
    // 거절 내역을 하나씩 카드뷰로 만들어서 보여주는 과정으로
    // 카드뷰를 추가하는 과정들은 앞서 설명한 코드와 동일하다.
    while(dcursor.moveToNext()){
        int gid = dcursor.getInt( i: 0);
        Cursor cursor = myDB.groupnames(gid);
        cursor.moveToNext();
        String txt = cursor.getString( i: 0)+" 그룹에서 참여 신청을 거절하였습니다.";

        addCardDeny(txt, gid);
    }
```

(3-1) '공지사항' 이용하기

① 공지사항 등록하기



-그룹 메인 페이지에서 '공지사항' 버튼을 클릭하면 공지사항 메인 페이지로 이동한다.
-공지사항 페이지의 '+' 버튼을 클릭하면 공지사항을 추가할 수 있다.
-공지사항 신청 양식은 ① 제목 ② 유형 ③ 내용으로 구성되어 있다.
② 공지사항 유형은 일반과 중요 공지사항이 분류되어 있다.
-양식에 맞게 작성한 공지사항을 '등록' 버튼을 통해 등록하면 공지사항 화면에 등록한 공지사항이 등록됐음을 확인할 수 있다.
-각 공지사항 우측의 '>' 버튼을 클릭하면 공지사항 상세 내용을 확인할 수 있다.
-공지사항 상세 페이지에서는 작성한 공지사항의 제목과 내용, 작성자, 유형, 등록 시간을 확인할 수 있다.

- 현재 그룹의 그룹 리스트 불러와서 보여주는 코드

```
myHelper = new MyDatabaseHelper( context: this);

sqlDB=myHelper.getReadableDatabase();
Cursor cursor=myHelper.noticeRead(GId);

layoutNotice = (LinearLayout) findViewById(R.id.layoutNotice);
layoutNotice.removeAllViews();

nid = new ArrayList<>();
gid = new ArrayList<>();
memberid = new ArrayList<>();
ntitle = new ArrayList<>();
ntype = new ArrayList<>();
ncontent = new ArrayList<>();
ntime = new ArrayList<>();

if(cursor!=null){

    while(cursor.moveToNext()){
        nid.add(cursor.getInt( i: 0));
        gid.add(cursor.getInt( i: 1));
        memberid.add(cursor.getString( i: 2));
        ntitle.add(cursor.getString( i: 3));
        ntype.add(cursor.getString( i: 4));
        ncontent.add(cursor.getString( i: 5));
        ntime.add(cursor.getString( i: 6));
    }

    for(int i=gid.size()-1 ;i>=0; i--){
        nId = nid.get(i);
        memberID = memberid.get(i);
        nTitle = ntitle.get(i);
        nType = ntype.get(i);
        nContent = ncontent.get(i);
        nTime = ntime.get(i);

        addCard(nId, memberID, nTitle, nType, nContent, nTime);
```

- 그룹 데이터 DB에 저장하는 코드

```
}else{
    if(noticeContent.trim().equals("")){
        LayoutInflater inflater = getLayoutInflater();

        View layout = inflater.inflate(R.layout.toastcus
                (ViewGroup) findViewById(R.id.toastLayou

        TextView text = layout.findViewById(R.id.toastTe

        Toast toast = new Toast( context: makeNoticeActivit
        text.setText("공지 내용을 입력하세요");
        toast.setDuration(Toast.LENGTH_SHORT);
        toast.setView(layout);
        toast.show();
    }else{

        if(rGroupNoticeType.getCheckedRadioButtonId()==
            noticeType="중요";
        }
        else{
            noticeType="일반";
        }

        String currentTime = new SimpleDateFormat( pattern

        //db 내용 추가
        MyDatabaseHelper myDB = new MyDatabaseHelper( co
        myDB.addNotice(GId, id, noticeTitle, noticeType,


        // 끝내기
        finish();
```

② 공지사항 수정하기

-각 공지사항의 상세 페이지에서 '수정' 버튼을 클릭하면 기존에 작성한 내용을 바탕으로 공지사항을 수정할 수 있다. 이때 제목, 유형, 내용 모두 수정할 수 있다.

-공지사항 수정 완료 후 하단의 '수정' 버튼을 클릭하면 수정이 완료된다.

- 입력받은 수정 값과 만들어놓은 메소드를 이용하여 공지 UPDATE

```
String currentTime = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss").format(new Date());

//db 내용 추가
MyDatabaseHelper myDB = new MyDatabaseHelper( context: modifyNoticeActivity.this);
myDB.updateNotice(GId, nId, noticeTitle, noticeType, noticeContent);
```

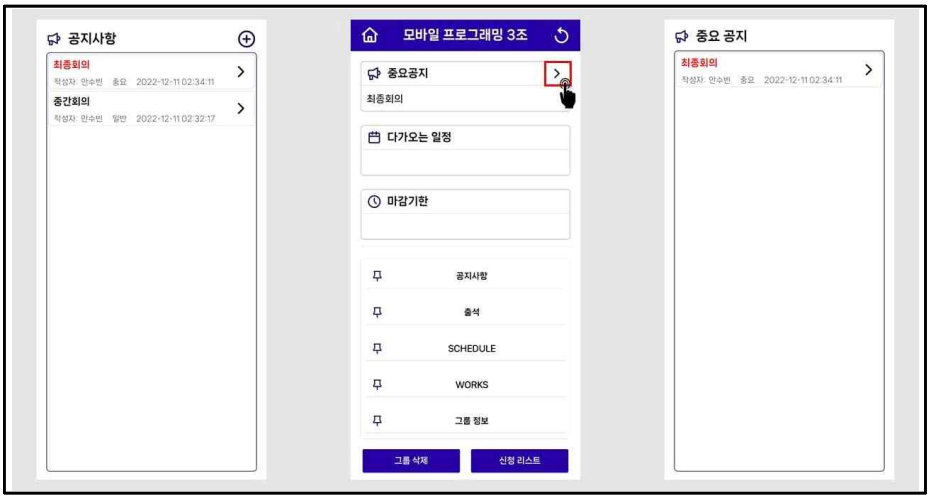
③ 공지사항 삭제하기

- 각 공지사항의 상세 페이지에서 '삭제' 버튼을 클릭하면 공지가 삭제된다.

- 만들어놓은 메소드 이용하여 공지 삭제

```
dlg.setTitle("공지를 정말 삭제하시겠습니까?");
dlg.setPositiveButton( text: "삭제", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {

        myDB.deleteNotice(GId, nId);
```

④ 중요 공지 확인하기



- 공지사항 등록 시에 유형을 '중요'로 지정한 경우, 공지사항의 메인 페이지에서 제목의 색이 빨간색으로 표

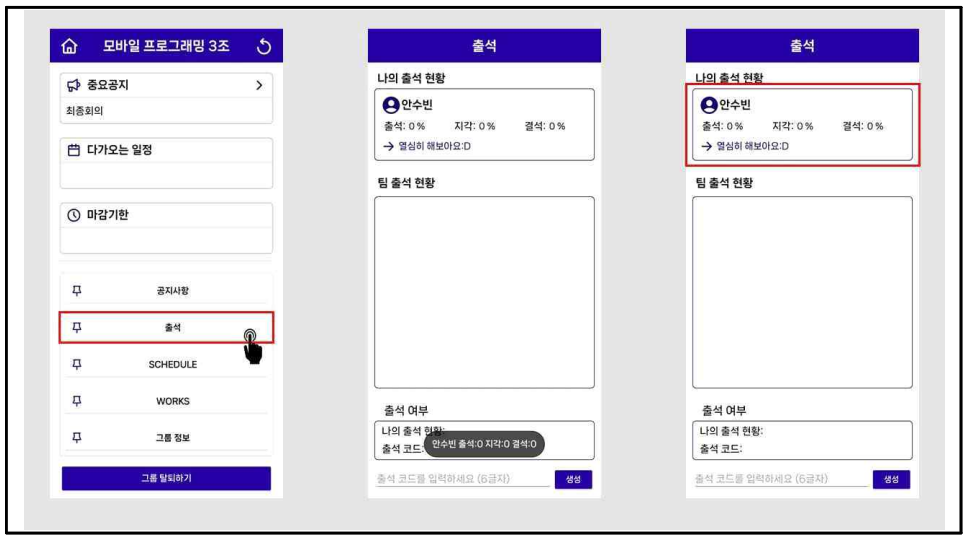
시된다.
- 중요 공지사항으로 등록한 경우, 그룹 메인 페이지에서 상단 배너의 중요 공지에도 빨간색으로 공지사항이 등록된다.
- 상단 배너의 중요 공지에서 우측의 '>' 버튼을 클릭하면 중요 공지만 따로 모아볼 수 있다.

*단, 중요 공지 페이지에서는 공지의 수정 및 삭제가 불가능하며, 공지사항의 수정 및 삭제는 오로지 공지사항 페이지에서만 가능하다.

* 공지는 최신의 공지부터 차례로 등록된다.

(3-2) '출석' 기능 이용하기

① '출석' 기능 개요



- 그룹 메인 화면에서 출석 버튼을 클릭하면 출석 메인 화면으로 이동하게 된다.
- 출석 페이지에 들어가면 처음에 하단에 일시적으로 팀원들의 출석 현황을 횟수로 보여준다. (이름 출석: 0 지각: 0 결석: 0)
- 출석 페이지에서는 ① 나의 출석 현황 ② 팀 출석 현황 ③ 출석 여부 ④ 출석 코드 생성 및 등록 기능을 이용할 수 있다.

① 나의 출석 현황에서는 나의 출석을 출석 / 지각 / 결석의 퍼센트 환산값으로 확인할 수 있다. 이때 출석률은 본인이 그룹에 참여한 이후 생성된 출석 코드 개수 기준으로 계산된다. 또한, 그룹 활동 독려를 위해 출석률에 따라 메시지가 출력된다.

  · 90%~: 팀 활동에 적극적 참여 중! 고마운 사람:D
  · 70%~: 좋아요! 좀 더 적극적인 참여 기대할게요:)
  · 50%~: 적극적 참여 부탁합니다!
  · ~49%: 참여가 부진해요:(

② 팀원들의 출석 현황 정보를 제공한다. 각 팀원의 출석률은 본인이 그룹에 참여한 이후 생성된 출석 코드 개수 기준으로 계산된다.

③ 최근 출석에 대한 나의 출석 현황을 알려준다.

④ 팀장과 팀원 기능이 분류된다.

·팀장의 경우 출석 코드를 입력하여 해당 시간의 출석 코드를 발급받을 수 있다. 따라서 버튼이 '발급' 버튼으로 되어 있다.

·팀원의 경우 출석 코드를 발급받을 수 없으며, 해당 버튼이 '생성'이 아닌 '등록'으로 되어 있다. 팀장이 발급한 출석 코드를 입력하면 출석 처리가 되는 것이다.

- 출석율 계산 후 출석률에 맞는 문구 보여주는 코드

```
if(cursor!=null){
    while(cursor.moveToNext()){
        participantID.add(cursor.getString( i: 0));
        participantName.add(cursor.getString( i: 2));
    }

    for(int i=0; i<participantID.size(); i++){
        String pID = participantID.get(i);
        String pName = participantName.get(i);

        int attendNum=0; // 출석수
        int lateNum=0; // 지각수
        int absenceNum=0; //결석수

        float attendRate; // 출석률
        float lateRate; // 지각률
        float absenceRate; // 결석률

        String AttendRate, LateRate, AbsenceRate;

        myHelper = new MyDatabaseHelper( context: this);
        sqlDB=myHelper.getReadableDatabase();

        int totalNum=0; // 총 결석 체크 수 (사람마다 그룹 참가 타임이 다를

        // aID 저장할 곳
        Cursor cursorMatchAId;
        ArrayList <Integer> attendID = new ArrayList<>();
        int AttendID;

        // 해당 aID 수를 구하기 위한
        Cursor cursorAIdNum;
        ArrayList <Integer> aIdCount = new ArrayList<>();

        // AttendStatus에서 gid, custid가 맞는 aid 가져오기 열넘버3
        cursorMatchAId = myHelper.AttendanceStatusRate(gId, pID);
        if(cursorMatchAId!=null){
            while(cursorMatchAId.moveToNext()){
if(cursorMatchAId!=null){
    while(cursorMatchAId.moveToNext()){
        attendID.add(cursorMatchAId.getInt( i: 3));
    }

    for(int k=0; k<attendID.size(); k++){
        AttendID = attendID.get(k);

        // 가져온 aid를 AttendanceRead2에 넣어서 aid 개수 가져오기. 열 넘
        cursorAIdNum = myHelper.AttendanceRead2(gId, AttendID);
```

```
if(cursorAIdNum!=null){
    while(cursorAIdNum.moveToNext()){
        aIdCount.add(cursorAIdNum.getInt( i: 0));
    }

    totalNum = aIdCount.size(); // 총 출석 체크 수 (사람마다 그룹 참가 타임이 다를 수
ArrayList <String> attendListM = new ArrayList<>();
String attendStatus;

Cursor cursorRate = myHelper.AttendanceStatusRate(gId, pID);
if(cursorRate!=null){
    while(cursorRate.moveToNext()){
        attendListM.add(cursorRate.getString( i: 4)); // 출석 여부 가져오기
    }


    for(int j=0; j<attendListM.size(); j++){
        attendStatus = attendListM.get(j); // 출석, 지각, 결석 횟수

        if(attendStatus.trim().equals("출석")){
            attendNum+=1;
        }else if(attendStatus.trim().equals("지각")){
            lateNum+=1;
        }else if(attendStatus.trim().equals("결석")){
            absenceNum+=1;
        }
    }

    LayoutInflater inflater = getLayoutInflater();

    View layout = inflater.inflate(R.layout.toastcustom,
            (ViewGroup) findViewById(R.id.toastLayout));

    TextView text = layout.findViewById(R.id.toastText);

    Toast toast = new Toast( context: AttendanceMain.this);
    text.setText(pName+" 출석:"+attendNum+" 지각:"+lateNum+" 결석:"+absenceNum);
    toast.setDuration(Toast.LENGTH_SHORT);
    toast.setView(layout);
    toast.show();
if(totalNum!=0){
    attendRate = (float)attendNum/totalNum*100;
    lateRate = (float)lateNum/totalNum*100;
    absenceRate = (float)absenceNum/totalNum*100;

    AttendRate = String.format("%.1f", attendRate); // 출석률
    LateRate = String.format("%.1f", lateRate); // 지각률
    AbsenceRate = String.format("%.1f", absenceRate); // 결석률

    addCardMemberAttend(pName, AttendRate, LateRate, AbsenceRate);
```

② 출석 코드 발급 받기 (팀장)

- 출석 코드는 출석 페이지의 하단에 6자리 숫자를 입력하여 발급할 수 있다.
- 출석 코드는 팀장만 발급할 수 있으며, 팀원의 경우 출석 코드 발급은 불가능하고 등록만 가능하다.
- 출석 코드를 발급하면 '출석 여부' 부분에 날짜와 시간이 표시되며, 해당 부분에서 본인의 출석 현황과 출석 코드를 확인할 수 있다. (가장 최신의 출석만 보여준다)
- 출석 코드를 발급하면 '팀 출석 현황'에서 팀원 전체의 출석률을 출석 / 지각 / 결석으로 나누어 비율로 환산한 수치를 보여준다. 단, 발급 후 기본값은 결석이다. 이는 출석 코드 입력 및 확인을 통해 출석으로 변경할 수 있다.

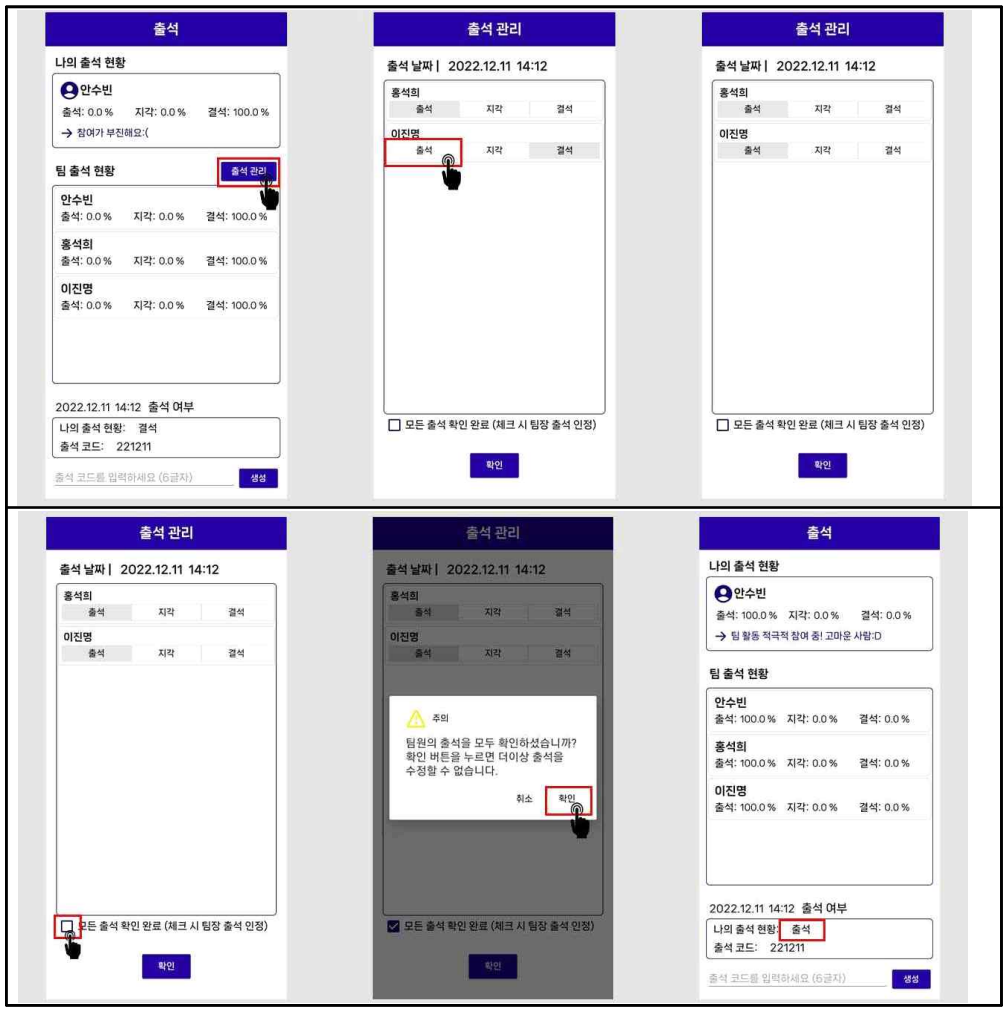
- 출석코드 생성 후 DB 저장 및 출석여부 테이블에 팀원들의 출석내역 생성하는 코드

```
if(btnCodeType == 1){
    // 팀장일 때
    intCode = Integer.parseInt(stringCode);

    String currentTime = new SimpleDateFormat( pattern: "yyyy.MM.dd  HH:mm").format(new Date());

    myHelper.addAttendance(intCode, gId, id, currentTime,  attendCHECK: 0);  // attendCHECK: 팀장이 팀원의 출석을 모두 확인했는

    Cursor cursor = myHelper.getGroupParticipant(gId);
    Cursor cursor2 = myHelper.AttendanceRead(gId);

    // 현재 그룹에 참여중인 멤버의 아이디와 이름을 저장할 ArrayList
    ArrayList <String> participantID = new ArrayList<>();
    ArrayList <String> participantName = new ArrayList<>();

    // 출석ID를 저장할 ArrayList
    ArrayList <Integer> aid = new ArrayList<>();

    if(cursor2!=null){
        while (cursor2.moveToNext()){
            aid.add(cursor2.getInt( i: 0));
        }

        int aId = aid.get(aid.size()-1); // 현재 등록하는 출석의 출석ID 저장

        if(cursor!=null){
            while(cursor.moveToNext()){
                participantID.add(cursor.getString( i: 0));
                participantName.add(cursor.getString( i: 2));
            }

            for(int i=0; i<participantID.size(); i++){
                String pID = participantID.get(i);

                myHelper.addAttendaceStatus(gId, pID, aId,  aSTATUS: "결석");
```

③ 출석 코드 등록하기 (팀원)

-팀원의 경우 팀장과 달리 출석 코드를 발급받을 수 없기 때문에 출석 페이지의 하단 부분에 출석 코드를 등록하는 기능을 사용할 수 있다.
- 팀장이 발급한 출석 코드를 입력한 후 등록 버튼을 누르면 출석 처리가 되며, 초기의 결석이 출석으로 변경됨을 확인할 수 있다.

④ 출석 관리하기 (팀장)



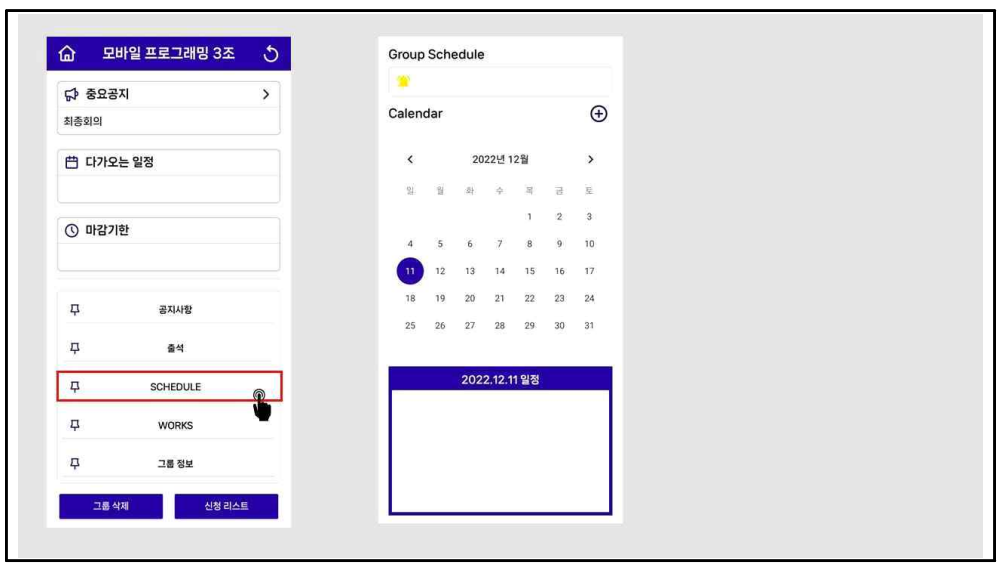
- 팀장의 경우 팀 출석 현황에서 '출석 관리' 버튼을 통해 출석 관리 페이지로 이동할 수 있다.
- 출석 관리 페이지에서는 팀장을 제외한 팀원들의 최근 출석에 대한 출석 여부를 변경할 수 있다.
-기존의 출석 여부를 변경하고 싶으면 버튼을 누르면 되고, 버튼을 누르게 되면 바로 팀원의 출석 현황이 변경된다.
- 팀장이 출석 관리 페이지에서 팀원의 출석 여부를 모두 확인하면 하단의 '모든 출석 확인 완료'를 체크하면 된다. 체크 후 뜨는 화면에서 확인 버튼을 누르면 팀장의 출석 현황 또한 '출석'으로 변경된다. 즉, 확인과 동시에 팀장의 출석이 등록되는 것이다. 만일 팀장이 다음 출석 코드 생성 시까지 출석을 확인하지 않는다면 팀장은 결석 처리된다.
- 출석 확인 이후에는 팀장 및 팀원의 출석 현황은 수정 불가능하다.

- 팀장이 팀원출석 확인 완료 및 팀장 출석하는 코드 (이후 출석 변경 불가)

```
btnAttendOK.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(chkAttend.isChecked()){
            AlertDialog.Builder dlg = new AlertDialog.Builder( context: AttendanceManageme
            dlg.setTitle("주의");
            dlg.setMessage("팀원의 출석을 모두 확인하셨습니까?\n확인 버튼을 누르면 더이상 출석을 수정할
            dlg.setIcon(R.drawable.warning);
            dlg.setPositiveButton( text: "확인", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Cursor cursor0 = myHelper.AttendanceRead(gId);

                    // 출석ID를 저장할 ArrayList
                    aid = new ArrayList<>();


                    if(cursor0!=null){
                        while (cursor0.moveToNext()){
                            aid.add(cursor0.getInt( i: 0));
                        }

                        aId2 = aid.get(aid.size()-1); // 출석 마지막 행의 출석ID

                        myHelper.updateAttendanceStatus(gId, aId2, id, aSTATUS: "출석");
                        myHelper.updateAttendance(gId, aId2, attendCHECK: 1); // 팀원출석 확
                        finish();
                        finish();
                    }

                }
            });
            dlg.setNegativeButton( text: "취소", listener: null);
            dlg.show();
        }else{
            finish();
```

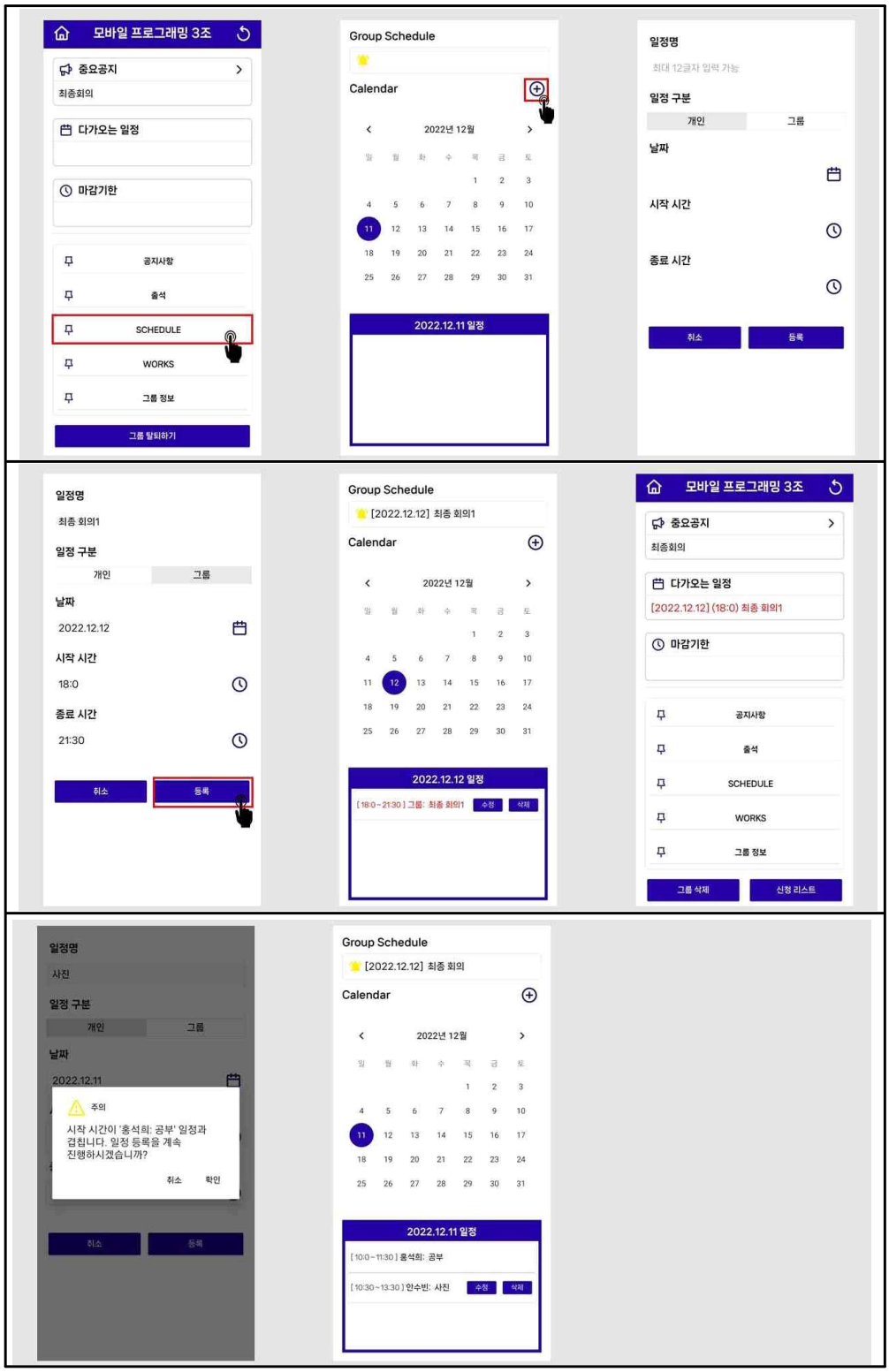
(3-3) 'SCHEDULE' 기능 이용하기

① SCHEDULE 기능 개요



- 그룹 메인 화면에서 'SCHEDULE' 버튼을 클릭하면 SCHEDULE 메인 화면으로 이동한다.
- SCHEDULE 페이지에서는 ① Group Schedule ② Calendar ③ 일정 리스트를 제공한다.
  ① Group Schedule의 경우, 그룹 일정에 대한 알림을 표시한다.
  ② Calendar의 경우 개인 및 그룹 일정을 추가할 수 있는 기능을 제공한다.
  ③ 일정 리스트는 Calendar에서 등록한 개인 및 그룹 일정을 날짜별로 보여준다.

② 일정 추가하기



- SCHEDULE 메인 페이지의 Calendar 옆 '+' 버튼을 클릭하면 일정을 추가할 수 있다.
-일정 추가 양식으로는 ① 일정명 ② 일정 구분 (유형) ③ 날짜 ④ 시작 시간 ⑤ 종료 시간을 설정할 수 있다.
① 일정명은 필수 입력사항이며 최대 12자 입력할 수 있다.
② 일정 구분은 개인과 그룹으로 분류되어 있다. 개인 일정의 경우 팀장과 팀원 모두 등록할 수 있지만, 그룹 일정의 경우 팀장만 등록할 수 있다.
- 양식에 맞게 정보를 입력한 후 하단의 '등록' 버튼을 클릭하면 일정 등록이 완료된다.
- 등록 후에는 SCHEDULE 페이지의 일정 리스트에 등록한 일정이 추가됐음을 확인할 수 있다.
- 일정의 종류가 그룹 일정인 경우, 일정 리스트에 일정 주인이 그룹이라고 표시된다. 개인 일정의 경우에는 등록한 이용자의 이름으로 표시된다.
- Calendar에서 각 날짜를 클릭하면 해당 날짜의 일정을 보여준다.
- 그룹 일정 추가 시 SCHEDULE 페이지 상단의 Group Schedule 알림창에 최신의 그룹 일정을 보여준다.
- 일정 추가 시 그룹 메인 페이지에는 다가오는 일정을 개인 / 그룹 구분 없이 최신의 것을 보여준다.
그룹 일정의 경우 빨간색으로, 개인 일정의 경우 검은색 글씨로 등록된다.
- 만일 일정 등록 시에 같은 시간대에 등록된 일정이 있다면 겹치는 일정 중 제일 첫 번째 것의 주인과 일정명이 적힌 경고창이 뜬다. 경고창이 떠도 일정을 등록하고자 하면 '확인' 버튼을 눌러 등록하면 정상적으로 일정이 등록된다.

- 겹치는 일정 검사 후 일정 등록하는 코드 (겹치는 일정 검사 코드는 너무 길어서 생략했습니다.)

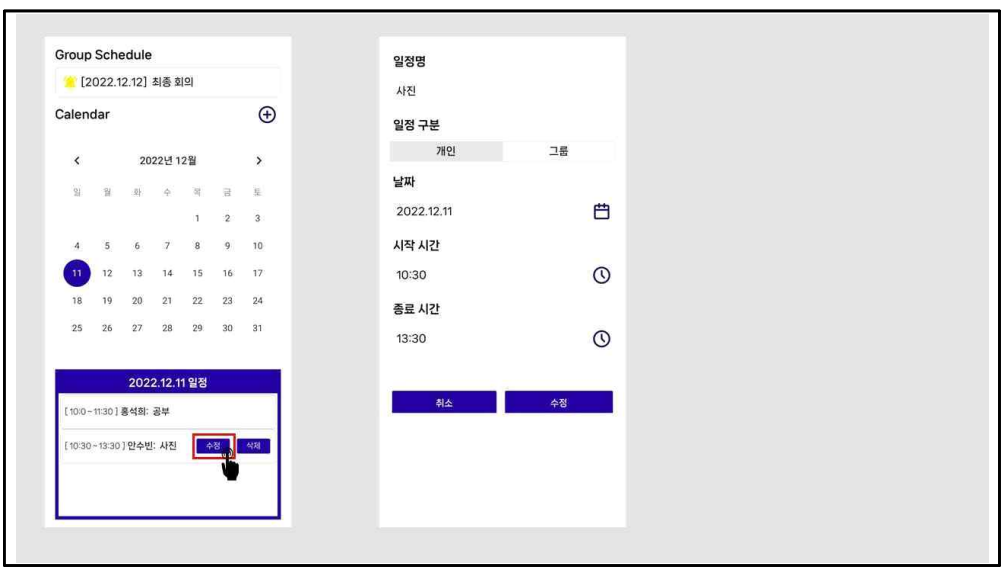
```
switch(plusType){
    case 0:
        plusScheduleComplete(id, date, startTime, endTime, gId);
        break;

    case 1: // 설정한 시작시간이 이미 있는 일정과 겹치는 경우
        AlertDialog.Builder dlg = new AlertDialog.Builder( context plusScheduleActivity.this, R.style.AlertDialogTheme);
        dlg.setTitle("주의");
        dlg.setIcon(R.drawable.warning);
        dlg.setMessage("시작 시간이 '" + sOwner + ": " + sTitle + "' 일정과 겹칩니다. 일정 등록을 계속 진행하시겠습니까?");
        dlg.setPositiveButton( text "확인", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                plusScheduleComplete(id, date, startTime, endTime, gId);
                finish();
            }
        });
        dlg.setNegativeButton( text "취소",  listener null);
        dlg.show();
        break;

    case 2:// 설정한 종료시간이 이미 있는 일정과 겹치는 경우
        AlertDialog.Builder dlg2 = new AlertDialog.Builder( context plusScheduleActivity.this, R.style.AlertDialogTheme);
        dlg2.setTitle("주의");
        dlg2.setIcon(R.drawable.warning);
        dlg2.setMessage("종료 시간이 '" + sOwner + ": " + sTitle + "' 일정과 겹칩니다. 일정 등록을 계속 진행하시겠습니까?");
        dlg2.setPositiveButton( text "확인", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                plusScheduleComplete(id, date, startTime, endTime, gId);
                finish();
            }
        });
        dlg2.setNegativeButton( text "취소",  listener null);
        dlg2.show();
        break;
```

- 캘린더뷰 날짜 클릭시 해당 날짜 일정 보여주는 코드

```
sCalendarView1.setOnDateChangeListener(new CalendarView.OnDateChangeListener() {
    @Override
    public void onSelectedDayChange(@NonNull CalendarView view, int year, int month, int dayOfMonth) {
        selectYear = year;
        selectMonth = month+1;
        selectDay = dayOfMonth;
        tvWhatDate1.setText(selectYear+"."+selectMonth+"."+selectDay+" 일정");

        layoutDateSchedule1.removeAllViews();

        if(cursor!=null){

            for(int i=0;i<gid.size(); i++){

                sId = sid.get(i);
                gId = gid.get(i);
                memberId = memberid.get(i);
                sName = sname.get(i);
                sType = stype.get(i);
                sDate = sdate.get(i);
                sStartTime = sStarttime.get(i);
                sEndTime = sEndtime.get(i);

                // 캘린더 뷰에서 선택한 날짜에 대한 일정만 보여줌. 기본은 현재 날짜 스케줄로..!
                if(sDate.trim().equals(selectYear+"."+selectMonth+"."+selectDay)){
                    addCard(sId, memberId, sName, sType, sDate, sStartTime, sEndTime);
                }
```

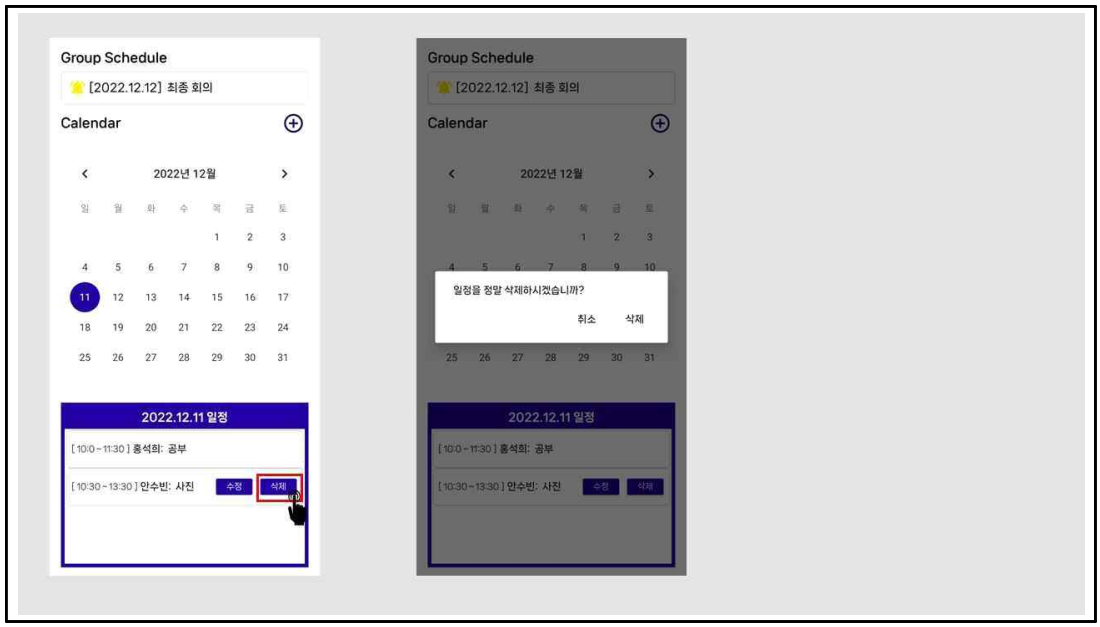
- 수정, 삭제 기능 역시 Schedule을 새로 등록하는 것과 거의 유사함
(Query문으로 만들어놓은 메소드를 이용)

③ 일정 수정하기



- 일정의 수정은 일정 리스트에서 일정의 주인만 할 수 있다. 그룹 일정의 경우는 팀장만 수정할 수 있다.

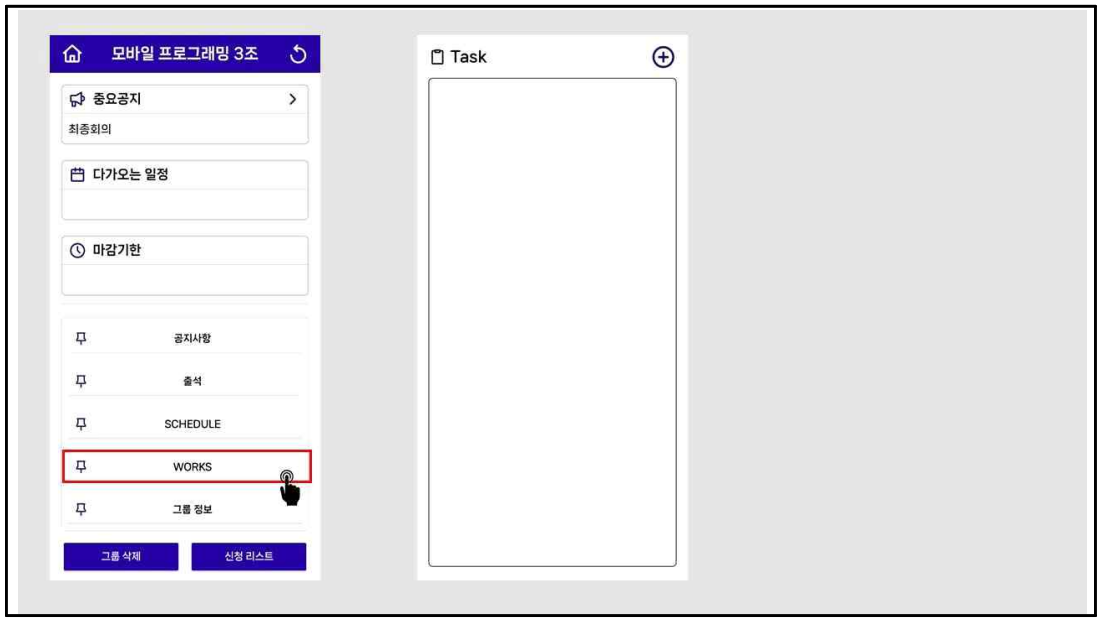
④ 일정 삭제하기



- 일정의 삭제는 일정 리스트에서 일정의 주인만 할 수 있다. 그룹 일정의 경우는 팀장만 삭제할 수 있다.

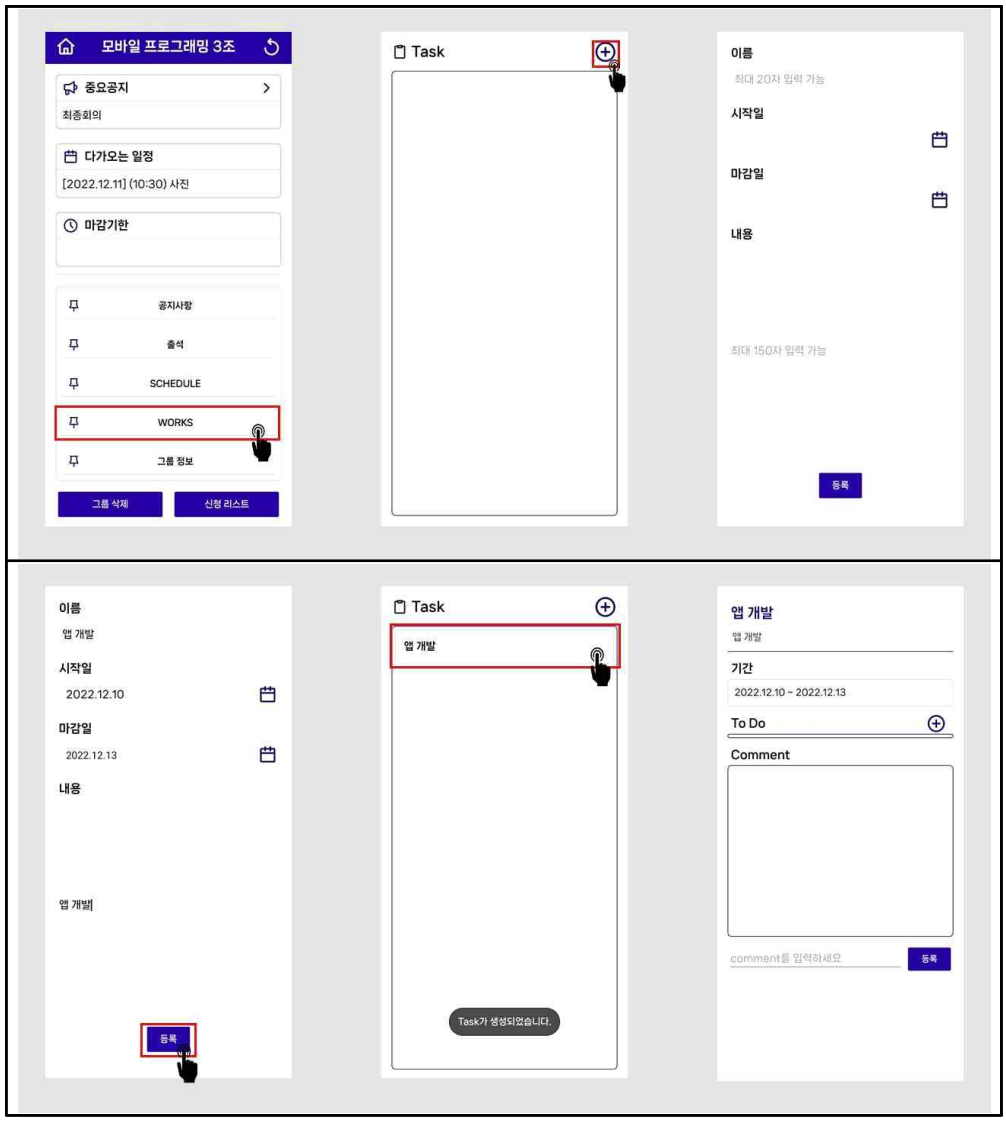
(3-4) 'WORKS' 페이지 이용하기

① 'WORKS' 개요



- 그룹 메인 페이지에서 'WORKS' 버튼을 클릭하면 'WORKS' 페이지로 이동한다.
- WORKS 페이지에서는 Task 등록이 가능하다.

② Task 등록하기



- WORKS 페이지 우측 상단의 '+' 버튼을 통해 Task를 등록할 수 있다.
- Task 등록 양식은 ① Task 이름 ② Task 시작일 ③ Task 마감일 ④ Task 내용으로 되어 있다.
- Task 등록 후에는 WORKS 페이지에 Task가 등록됐음을 확인할 수 있다.
- 각 Task를 클릭하면 해당 Task의 상세 페이지에 들어갈 수 있다.
- Task 상세 페이지에서는 Task의 이름, 내용, 기간을 보여주며 추가로 ① ToDo ② Comment를 등록할 수 있다.

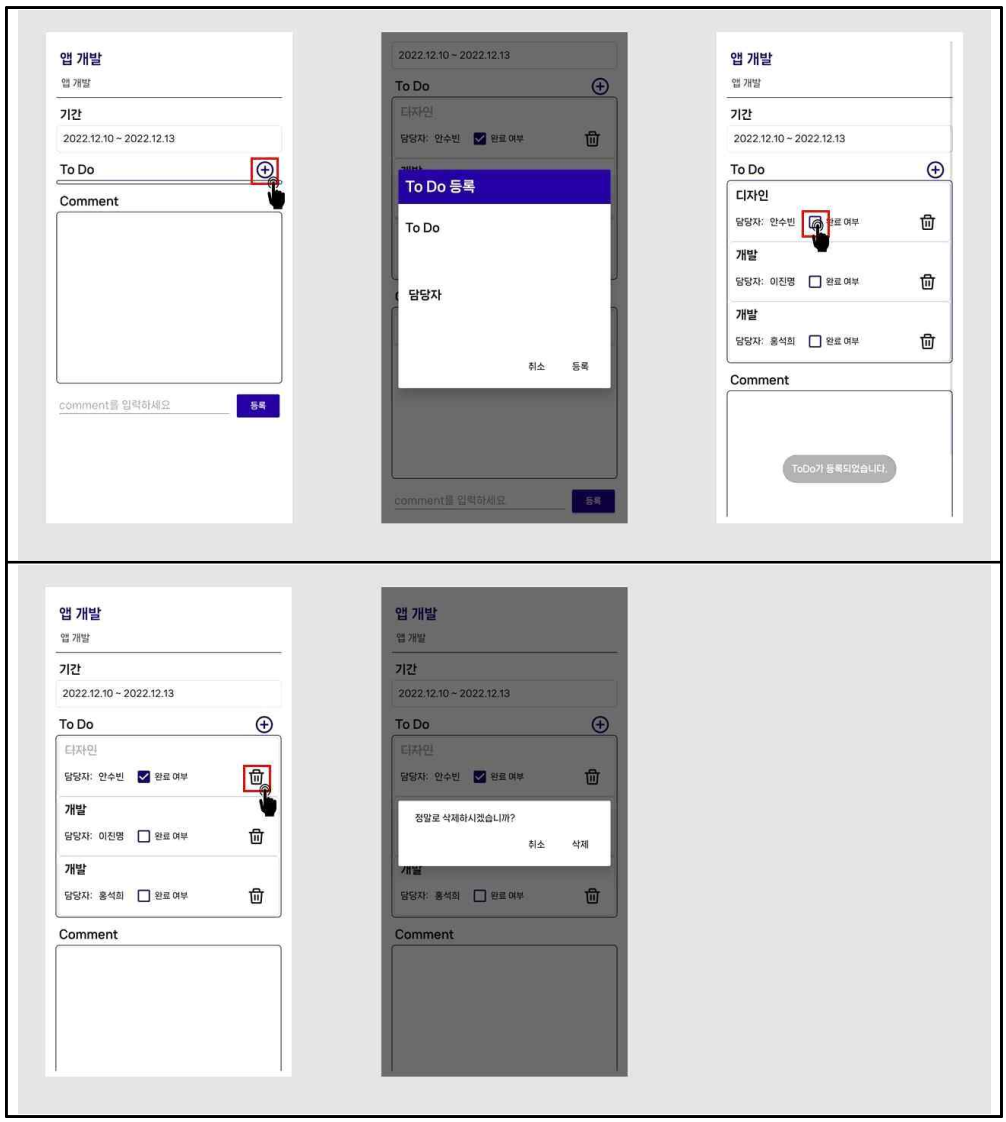
- Task 리스트 동적으로 보여주는 코드 (ToDo, Comment 역시 동일한 방법으로 보여줌)

```
Cursor taskCursors = myDB.getTasks(gid);
// 가져온 모든 Task들의 정보들을 하나씩 가져와서
while(taskCursors.moveToNext()){
    int taskId = taskCursors.getInt( i: 0);
    String taskTitle = taskCursors.getString( i: 1);
    String taskContent = taskCursors.getString( i: 2);
    String sdate = taskCursors.getString( i: 3);
    String edate = taskCursors.getString( i: 4);
    // 만들어놓은 함수로 카드뷰로 만들어서 하나씩 추가
    addCardTask(taskId, taskTitle, taskContent, sdate, edate);
```

- Task 만들기 (DB에 추가), ToDo, Comment 역시 동일한 방법으로 생성

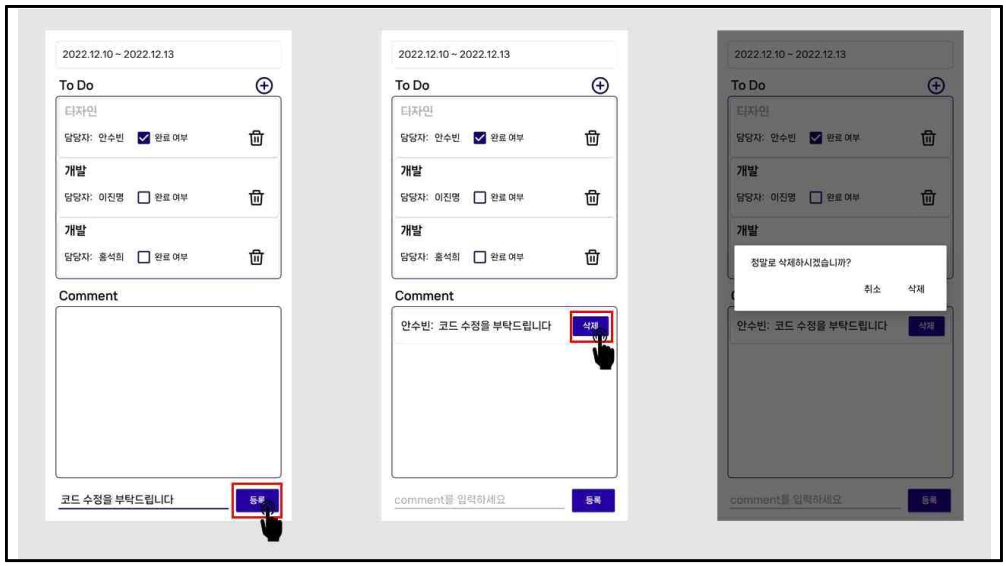
```
}else{
    Intent outIntent = new Intent(getApplicationContext(), workActivity.class);
    // Task 테이블에 정보 저장
    myDB.addTask(gid, id, name, content, sdate, edate);
```

③ ToDo 이용하기



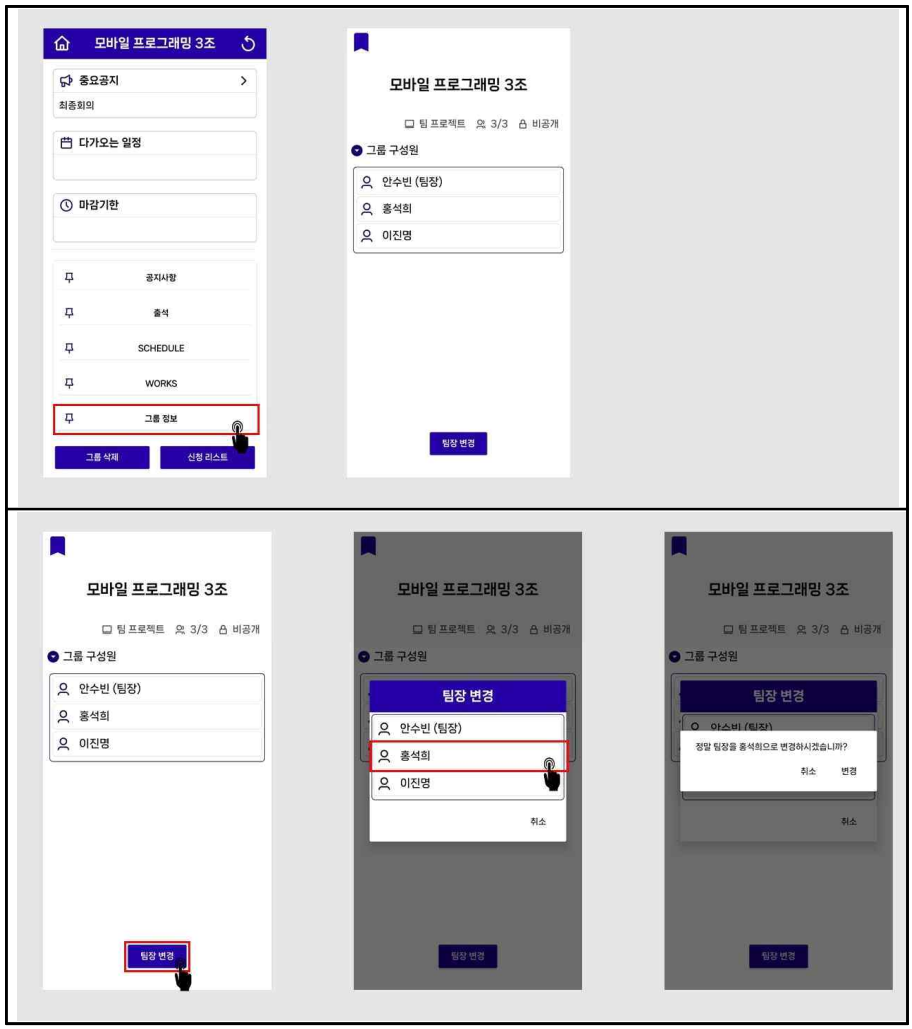
- ToDo의 경우 팀장만 이용한 기능이며, ToDo를 등록하여 팀원들에게 역할을 분담해줄 수 있다.
- '+' 버튼을 클릭하여 ToDo를 등록할 수 있다.
- ToDo 등록 양식은 ① ToDo의 이름 ② ToDo 담당자로 구성되어 있다.
- ToDo 등록 이후 역할을 수행했을 경우 완료 여부를 체크하여 업무를 완료했음을 표시할 수 있다.
- ToDo는 수정이 불가능하고 삭제만 가능하다.

④ Comment 이용하기



- ToDo와 달리 Comment는 팀장과 팀원 모두 이용할 수 있는 기능이다.
- 하단에 Comment를 입력하여 '등록' 버튼을 클릭하면 Comment가 등록된다.
- Comment도 ToDo와 마찬가지로 수정은 불가능하고 삭제만 가능하다.

(3-5) '그룹 정보' 보기



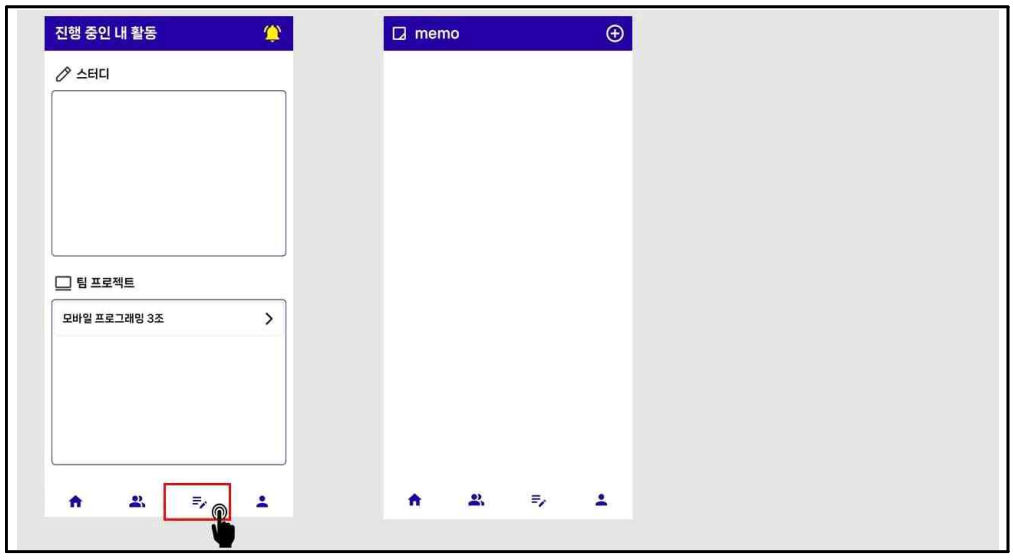
- 그룹 메인 페이지에서 '그룹 정보' 버튼을 클릭하면 '그룹 정보' 페이지로 이동한다.
- 그룹 정보 페이지에서는 해당 그룹의 이름, 카테고리, 인원수, 공개 여부와 그룹 구성원에 대한 정보를 제공한다.
- 하단의 '팀장 변경' 버튼을 클릭하면 해당 그룹 팀원들의 리스트를 보여준다. 원하는 팀원을 클릭함으로써 해당 팀원의 역할을 팀장으로 변경할 수 있다.

- 팀원의 역할 팀장으로 바꿔주는 코드

```
dlg1.setPositiveButton( text: "변경", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        // 팀장만 팀장 변경 버튼이 보이도록 했으니 현재 사용자의 ID인 id가
        // 그룹 팀장의 id임
        Cursor gcursor1 = myDB.customerGroups(id);
        gcursor1.moveToNext();
        // 팀장의 그룹속성을 하나씩 확인해보면서
        // 현재 그룹에 대한 역할을 팀장에서 팀원으로 변경
        for(int j=0;j<4;j++){
            if(gcursor1.getInt(j)==gid){
                // 현재 그룹 팀장의 역할을 팀원으로 변경
                if(j==0){
                    myDB.modifyCustomerRole1(id, c: "팀원");
                    break;
                }
                else if(j==1){
                    myDB.modifyCustomerRole2(id, c: "팀원");
                    break;
                }
                else if(j==2){
                    myDB.modifyCustomerRole3(id, c: "팀원");
                    break;
                }
                else if(j==3){
                    myDB.modifyCustomerRole4(id, c: "팀원");
                    break;
```

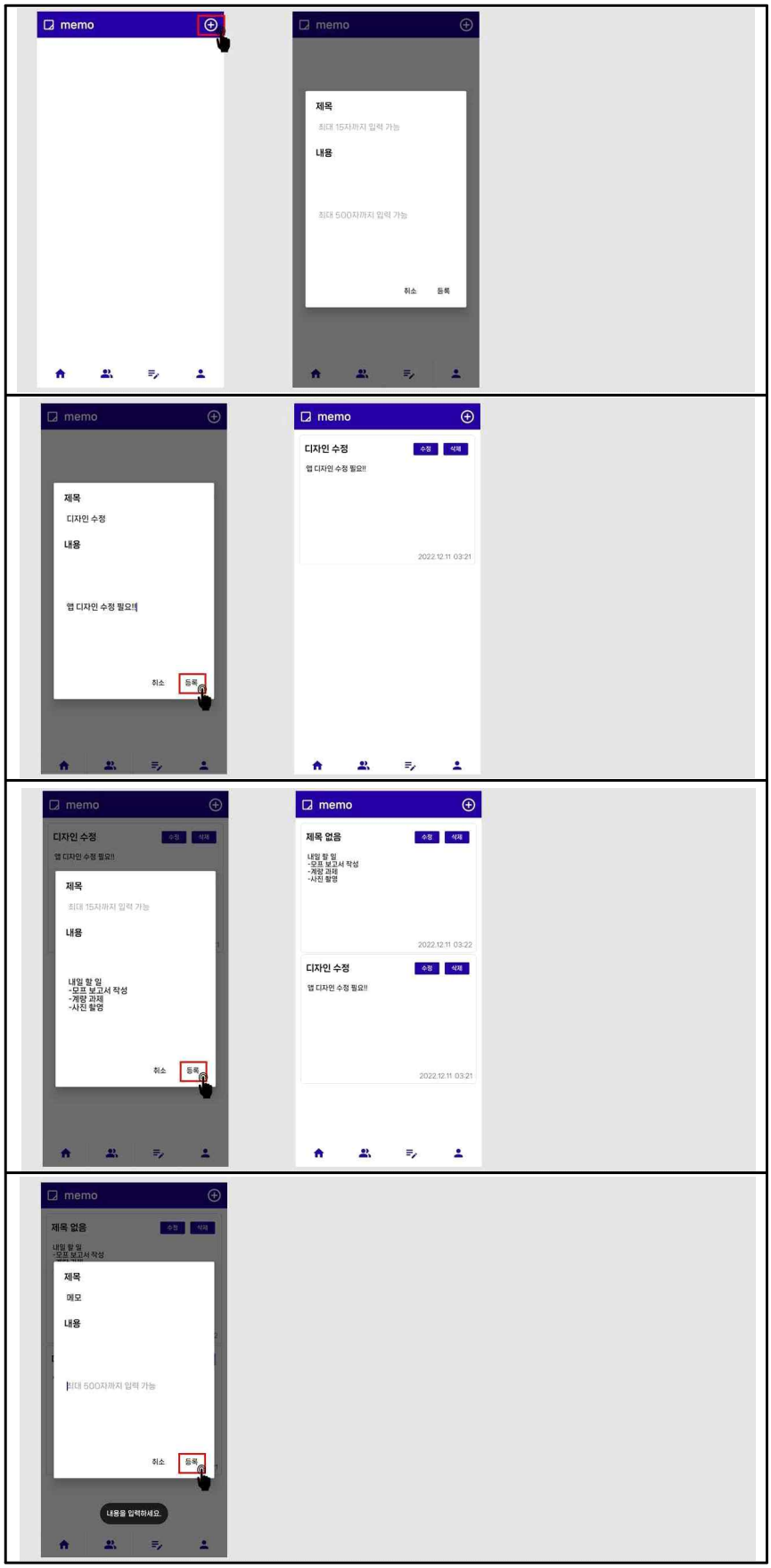
4) '메모' 기능 이용하기

(1) '메모' 기능 개요



- 왼쪽 기준 하단바의 세 번째 버튼을 클릭하면 메모장 기능을 이용할 수 있다.
- 메모 기능의 경우 개인이 사용하는 것으로, 그룹과 상관없이 작성자 개인만 열람할 수 있다.

(2) '메모' 추가하기


memo
제목
최대 15자까지 입력 가능
내용
최대 500자까지 입력 가능
취소
등록
디자인 수정
수정
삭제
앱 디자인 수정 필요!!
2022.12.11 03:21
제목 없음
내일 할 일
-모프 보고서 작성
-계량 과제
-사진 촬영
2022.12.11 03:22
내일 할 일
-모프 보고서 작성
-계량 과제
-사진 촬영
메모
내용을 입력하세요.

- 우측 상단의 '+' 버튼을 클릭하면 메모를 추가할 수 있다.
- 메모 입력 양식은 ① 제목 ② 내용으로 구성되어 있다.
- 메모를 등록하면 메모 페이지에 등록됨을 확인할 수 있다.
- 메모는 수정 및 삭제가 가능하다.
- 추가적으로 메모 작성 유형 2가지에 관해 설명하고자 한다.

① 제목 없이 내용만 입력하는 경우: 메모의 내용은 자동으로 '제목 없음'으로 설정된다.

② 내용 없이 제목만 입력하는 경우: 오류 메시지가 뜨며 메모를 저장할 수 없다.

- 메모 추가하는 코드

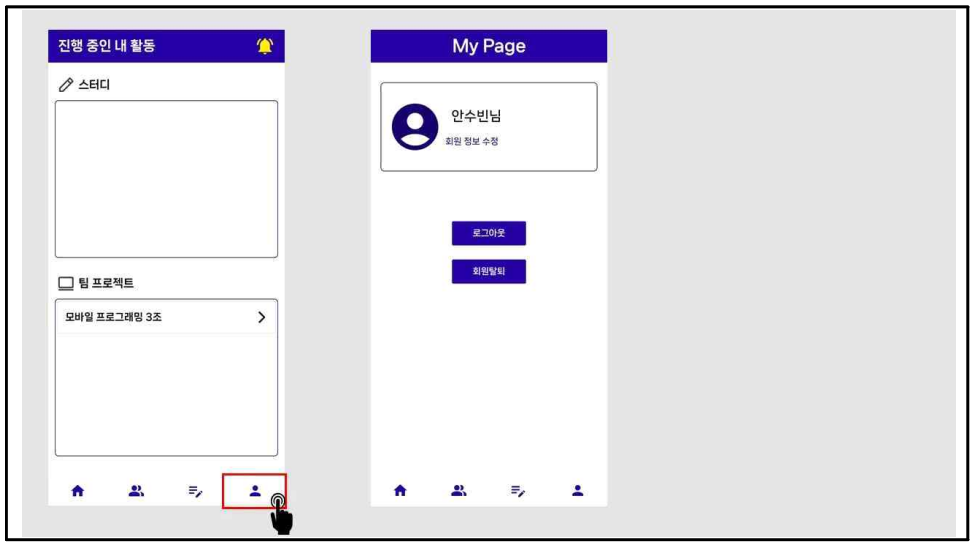
```
myHelper.addMemo(id,Title, Content, currentTime);
// dialog 종료하면서 memo 페이지도 종료 후 다시 열기
Intent intent = new Intent(getApplicationContext(),Memo.class);
intent.putExtra( name: "ID1", id);
startActivity(intent);
finish();
```

- 메모 보여주는 코드

```
Cursor cursorMemo = myHelper.MemoRead(id); // db 읽어오기 위한

if(cursorMemo!=null){
    while(cursorMemo.moveToNext()){
        // 메모 아이디, 고객 아이디, 메모 제목, 메모 내용, 메모 날짜
        memoid.add(cursorMemo.getInt( i: 0));
        custid.add(cursorMemo.getString( i: 1));
        memotitle.add(cursorMemo.getString( i: 2));
        memocontent.add(cursorMemo.getString( i: 3));
        memodate.add(cursorMemo.getString( i: 4));
    }
    for(int i=memoid.size()-1; i>=0; i--){ // 최신 메모가 위로 올라가게 함
        memoId = memoid.get(i);
        custId = custid.get(i);
        memoTitle = memotitle.get(i);
        memoContent = memocontent.get(i);
        memoDate = memodate.get(i);

        if(custId.trim().equals(id)){ // 자신의 메모만 보여줌
            addCard(memoId, memoTitle, memoContent, memoDate); // 메모 layout에 추가하기
```

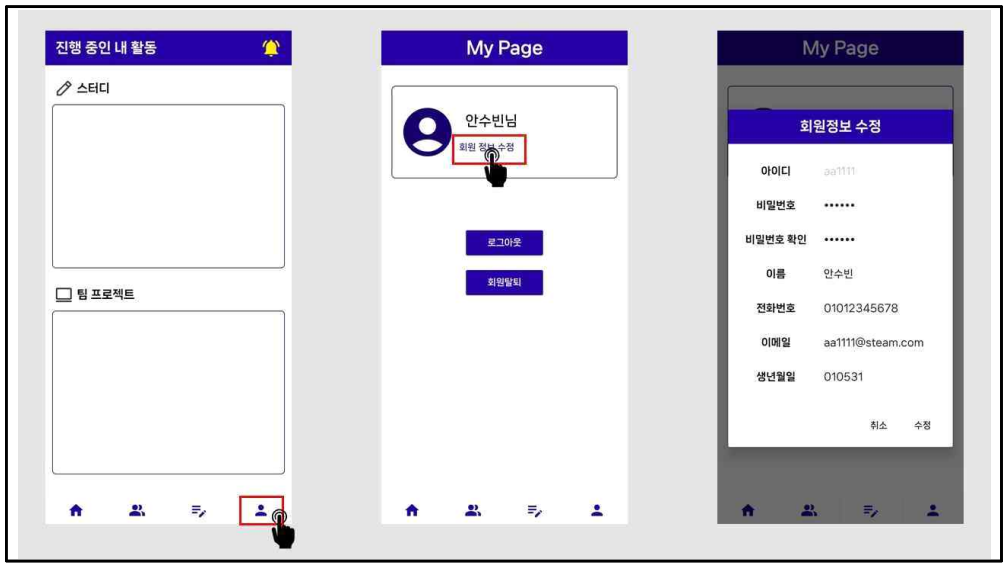
5) '마이페이지' 이용하기

(1) 마이페이지 개요



- 하단바의 가장 우측 버튼을 클릭하면 마이페이지로 이동한다.
- 마이페이지에서는 회원 정보 수정, 로그아웃, 회원 탈퇴가 가능하다.

(2) 회원정보 수정

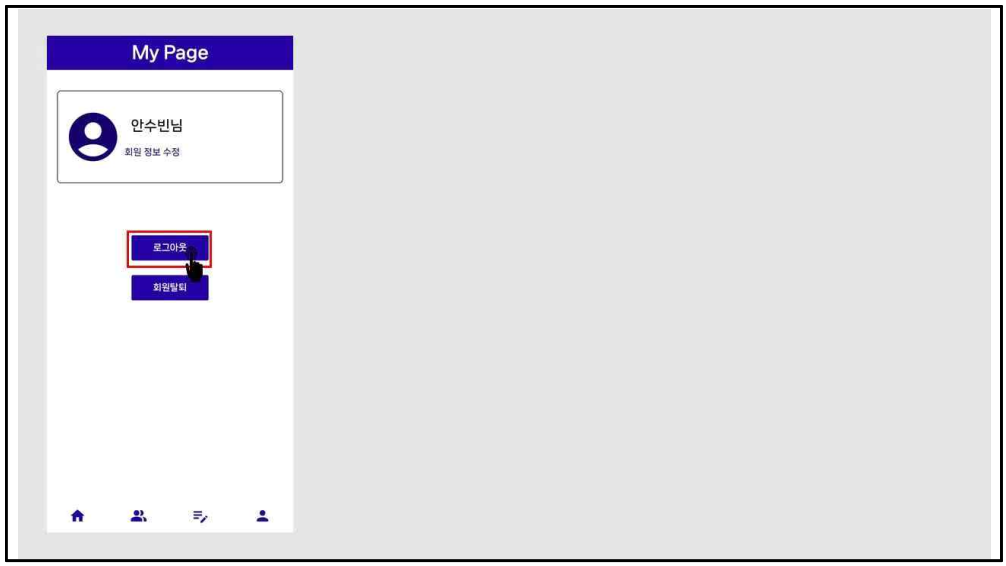


- '회원 정보 수정' 버튼을 통해 아이디를 제외한 나머지 정보를 수정할 수 있다.

- 정보 수정하는 메소드

```
// 만들어놓은 UPDATE문을 사용하는 함수를 이용하여 수정
myDB.modifyCustomer(id, pw, name, phone, email, birth);
```

(3) 로그아웃

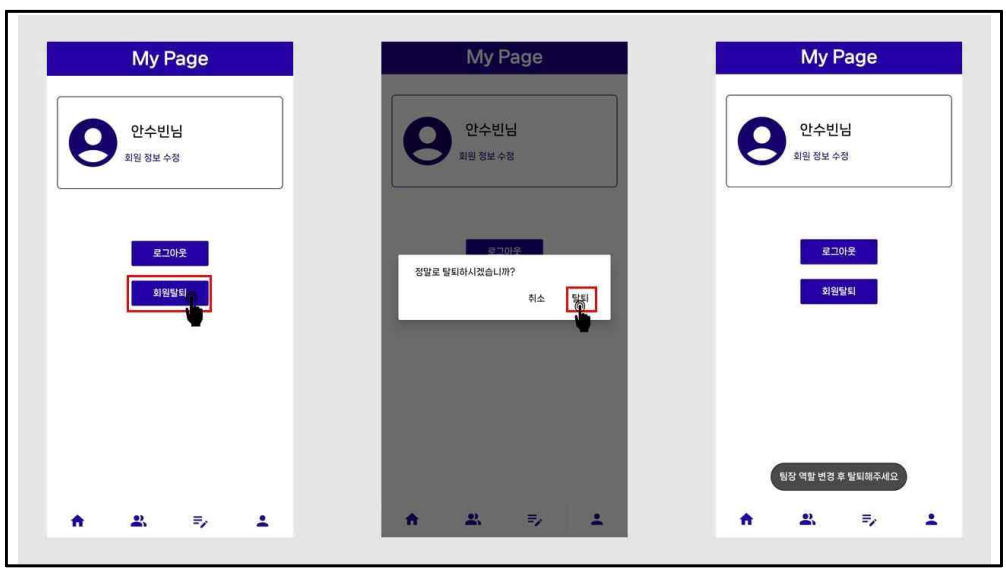


- 마이페이지의 '로그아웃' 버튼을 클릭하여 로그아웃을 할 수 있다. 로그아웃 후에는 초기화면인 로그인 페이지로 이동하게 된다.

- 로그아웃 코드

```
btnLogOut.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        // 로그인 페이지로 이동하면서 기존에 사용한 모든 액티비티들을 삭제
        Intent intent = new Intent( packageContext Mypage.this, LoginActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK); //
        startActivity(intent);
```

(4) 회원탈퇴



- 마이페이지에서 '회원 탈퇴' 버튼을 클릭하면 회원 탈퇴를 할 수 있다.
- 회원 탈퇴 시에는 사용자의 정보가 삭제될 뿐만 아니라 사용자의 활동 내역 (그룹 내 작성 공지, 일정 등)도 함께 삭제된다.

-본인이 팀장인 그룹 또는 본인을 포함한 2인 이상의 그룹원이 존재하는 그룹이 존재할 경우, 회원 탈퇴가 불가능하다. 이러한 경우 아래의 방법을 사용하여 회원 탈퇴를 진행할 수 있다.

① 그룹 페이지의 그룹 정보 페이지에서 팀장 역할을 다른 팀원에게 넘겨준 이후에 탈퇴할 수 있다.

* 즉, 참여 그룹은 있으나 본인이 팀원인 경우, 혹은 팀장이지만 1인 그룹인 경우에만 즉시 회원 탈퇴가 가능하다.

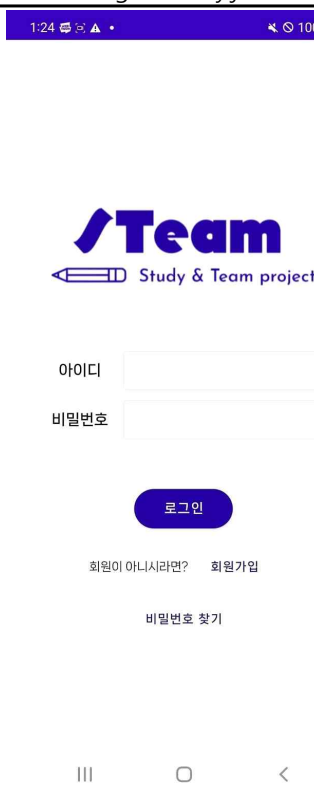
- 회원탈퇴 코드 (회원과 관련된 모든 활동 정보들을 같이 삭제하여 DB에 불필요한 정보 제거)

```
myDB.deleteGroup(customerGroupId1);
myDB.deleteGroupTask(customerGroupId1);
myDB.deleteGroupToDo(customerGroupId1);
myDB.deleteApplyForDropOut(id);
myDB.deleteConsentForDropOut(id);
myDB.deleteDenyForDropOut(id);
myDB.deleteNoticeForDropOut(id);
myDB.deleteAttendanceStatusForDropOut(id);
myDB.deleteAttendanceForDropOut(id);
myDB.deleteCommentForDropOut(id);
myDB.deleteScheduleForDropOut(id);
myDB.deleteMemoForDropOut(id);
myDB.modifyGNum(customerGroupId1, gnum: gnum1-1);
```

6) 각 페이지별 Java 파일

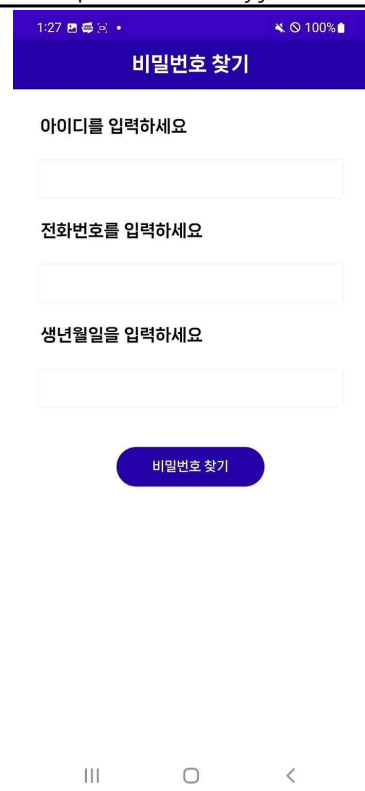
앱의 사용방법만 작성하기에도 내용이 너무 많고 코드가 많기도 해서 각 액티비티별 핵심코드를 모두 추가하지는 못하였다. 아래 표시된 Java 파일의 코드를 확인하시면 각 액티비티의 기능들을 더욱 쉽게 이해하실 수 있을 것으로 생각됩니다.

LoginActivity.java

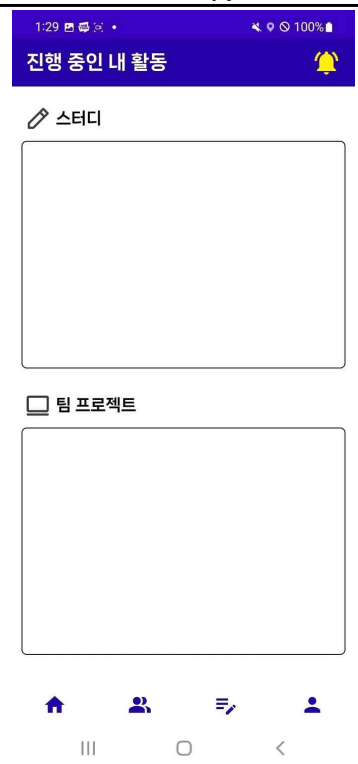


SignUpActivity.java

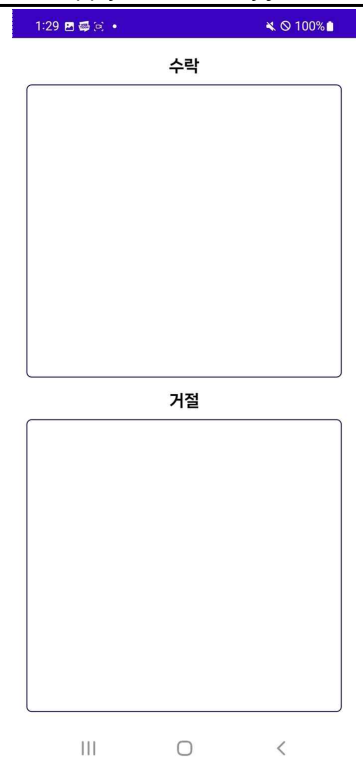
passwordActivity.java



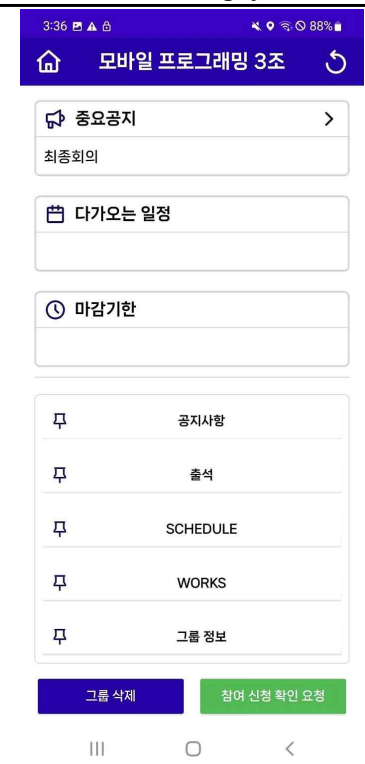
MainActivity.java



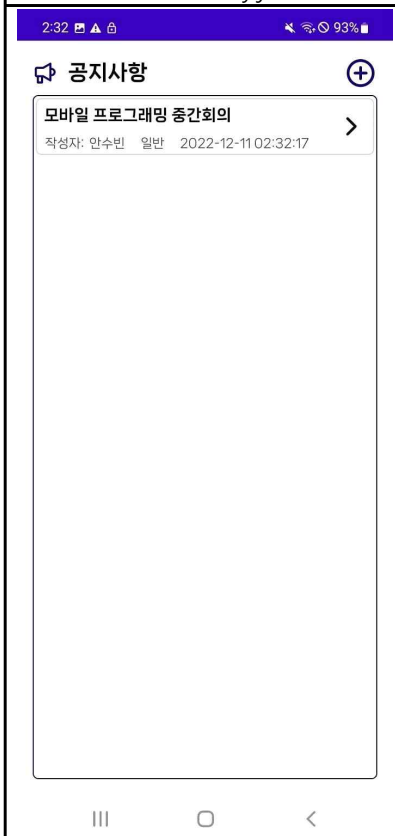
ApplyResultActivity.java



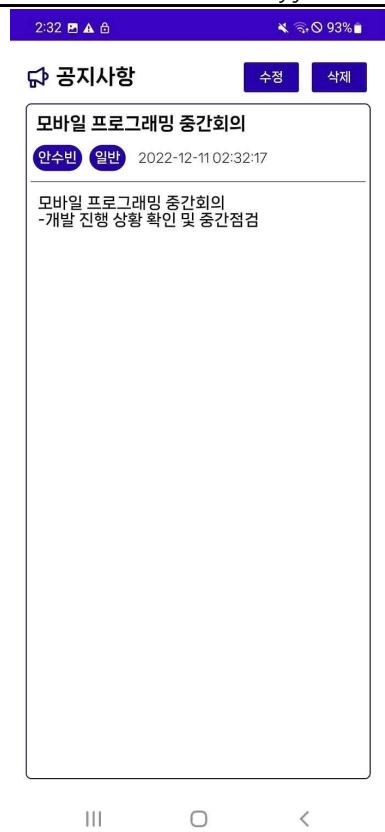
TeamMainPage.java



noticeActivity.java

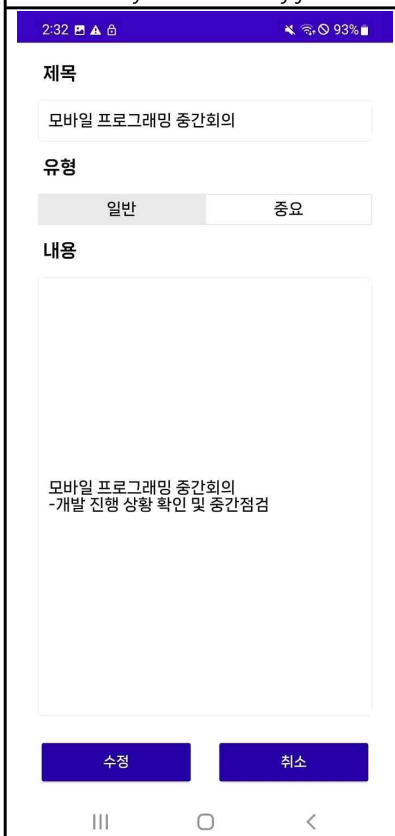


noticeContentActivity.java

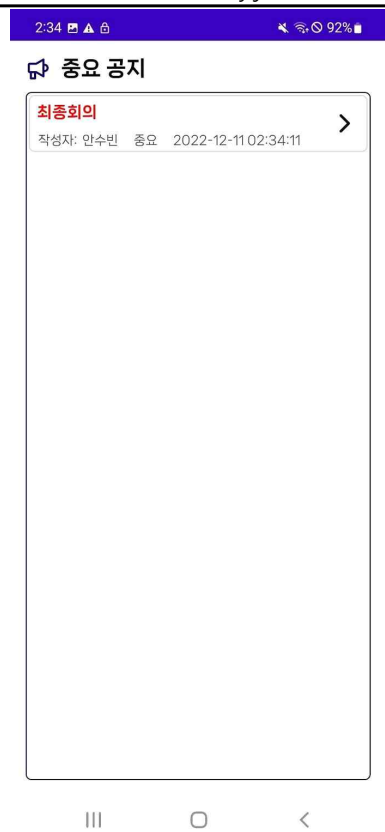


makeNoticeActivity.java

modifyNoticeActivity.java

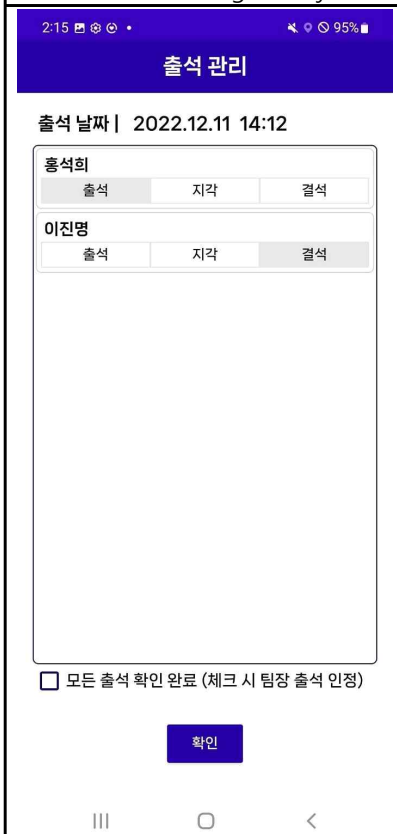


lnoticeActivity.java

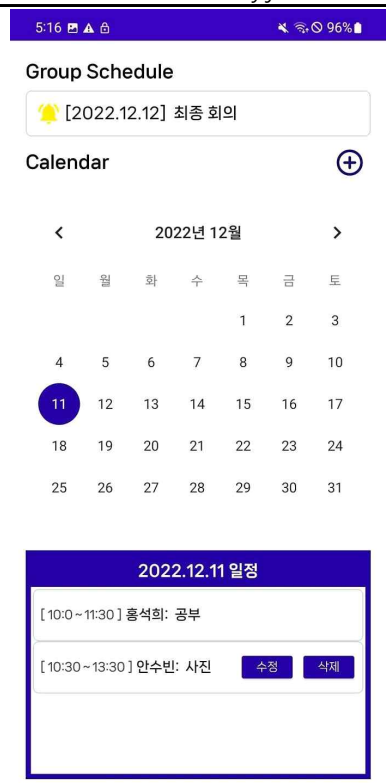


AttendanceMain.java

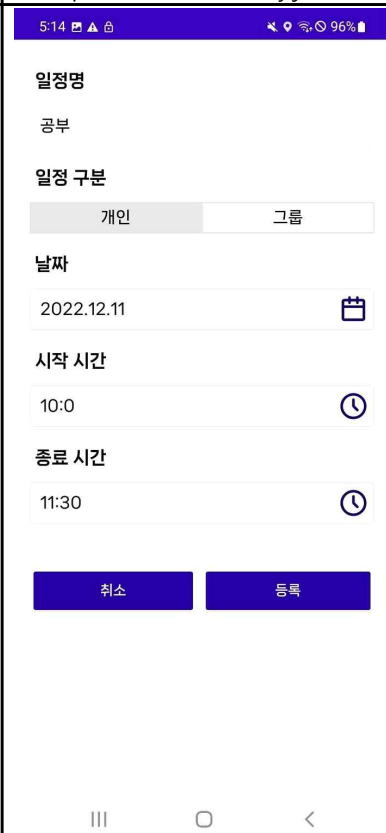
AttendanceManagement.java



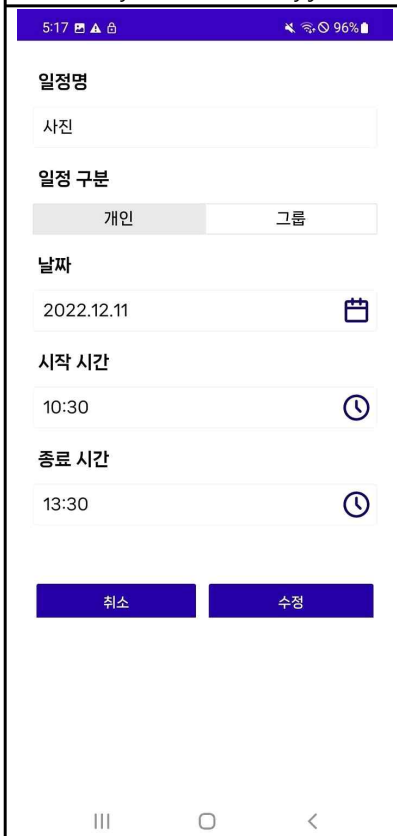
ScheduleActivity.java



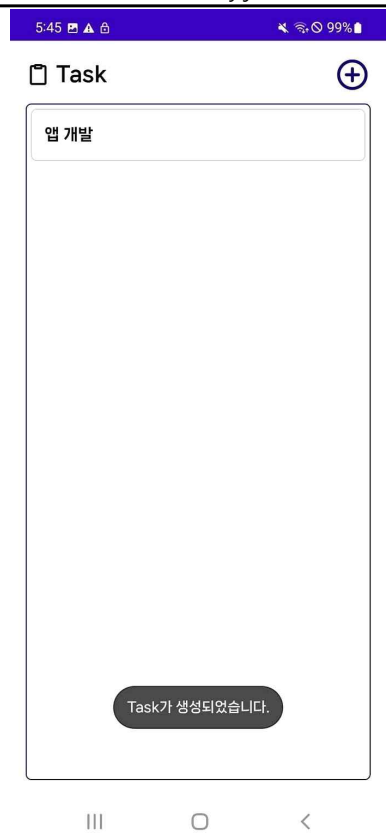
plusScheduleActivity.java



modifyScehduleActivity.java



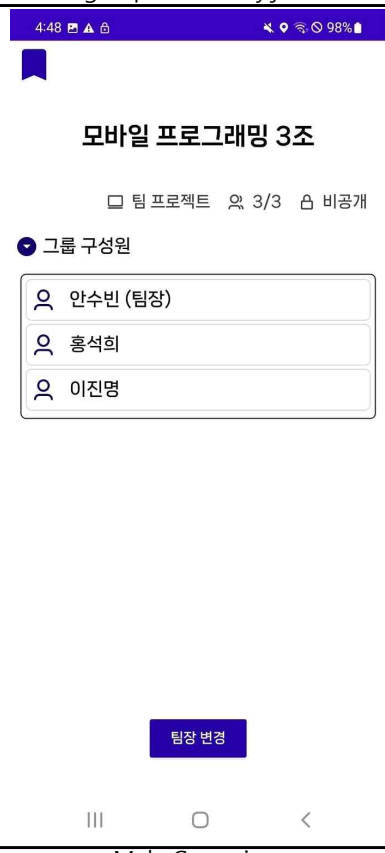
workActivity.java



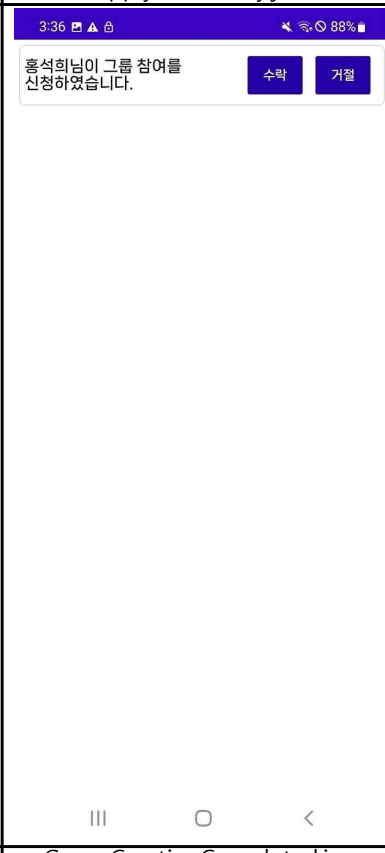
taskActivity.java

makeTaskActivity.java

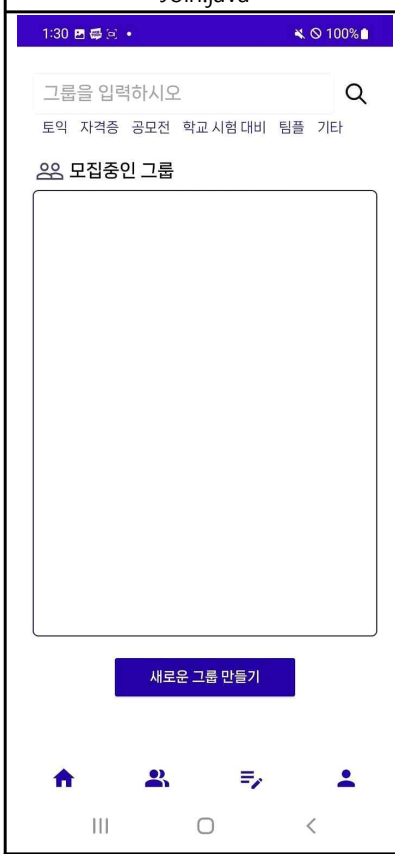
groupInfoActivity.java



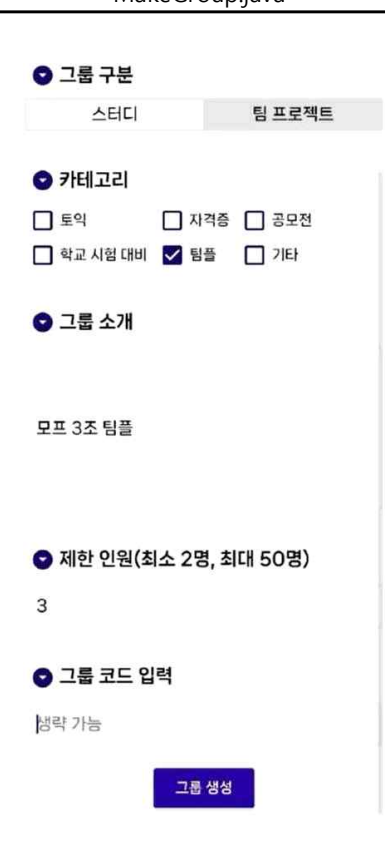
ApplyListActivity.java



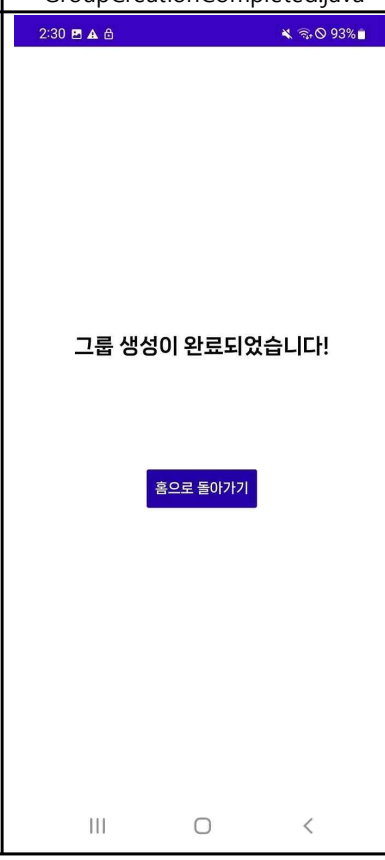
Join.java



MakeGroup.java



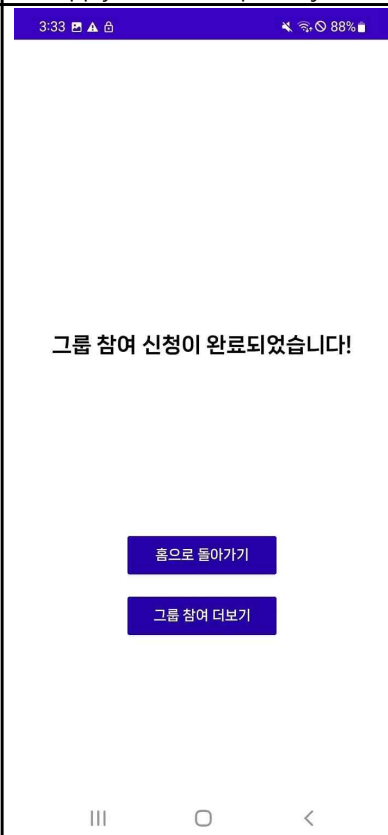
GroupCreationCompleted.java



GroupInformation.java

ApplyForJoin.java

ApplyForJoinCompleted.java



Memo.java

Mypage.java

나머지 class들

SplashActivity.java

- 앱 처음 시작 스플래시 화면

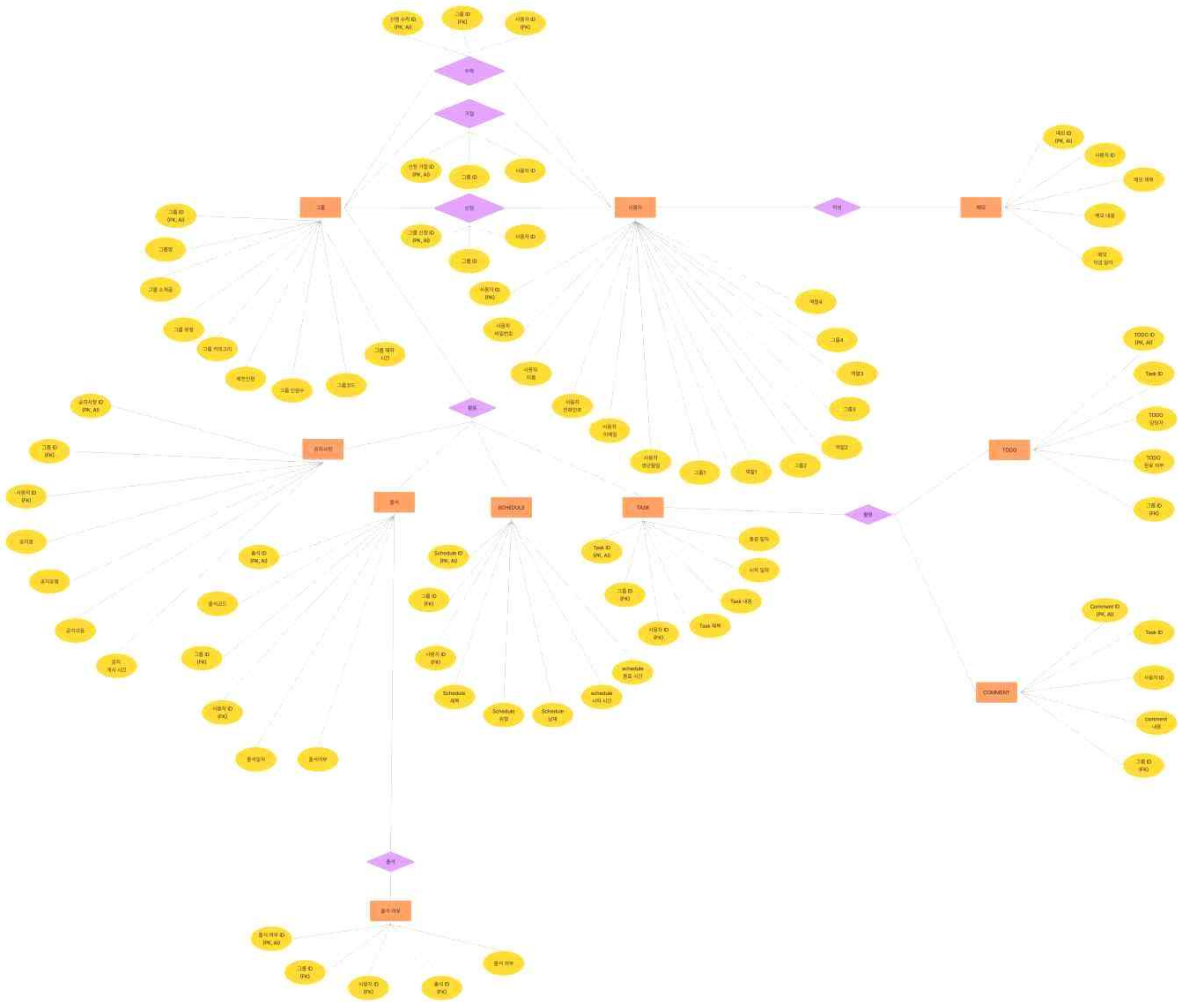
startCalendarActivity.java

- Task 시작일 마감일 선택 페이지

7. DB 구성도

1) ERD

(1) 한글 버전



- 위의 그림과 아래의 설명을 통해 그룹 테이블과 사용자 테이블을 제외한 모든 테이블에 그룹 ID, 사용자 ID 속성이 존재하는 것을 알 수가 있다. 앱의 기능면에서 반드시 그룹 ID나 사용자 ID가 반드시 필요하지 않은 경우(그룹별로 분류되지 않고 Task별로 분류되는 ToDo나 Comment 등)도 있는데 이렇게 그룹 ID와 사용자 ID 값을 포함시킨 이유는 그룹이 삭제되거나 그룹에 탈퇴할 경우 삭제된 그룹의, 탈퇴한 사용자의 모든 활동내용(공지, 출석, 일정, Comment 등)을 삭제하기 위해서이다. 활동내용 데이터가 남아있어도 문제가 되지는 않지만 데이터베이스의 용량 낭비문제로 모두 삭제하도록 구현하였다.

① 그룹 테이블
- 그룹 ID 속성 (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 그룹들을 구분해주는 속성이다.
- 그룹명 (TEXT NOT NULL) : 그룹의 이름 저장하는 속성이다.
- 그룹 소개글 (TEXT NOT NULL) : 그룹 소개 내용을 저장하는 속성이다.
- 그룹 유형 (TEXT NOT NULL) : 팀 프로젝트 / 스터디 2가지 값을 가지는 속성으로 그룹의 유형을 구분해주는 속성이다.
- 그룹 카테고리 (TEXT NOT NULL) : 그룹별로 6가지의 카테고리가 존재하는데 토익, 자격증, 공모전, 팀플, 학교 시험 공부, 기타 6가지의 카테고리가 존재한다. 그룹의 카테고리 값을 저장하는 속성으로 2개 이상의 카

테고리가 선택된 경우는 '# 카테고리1 # 카테고리2' 이런식으로 카테고리가 연결된다.
- 제한인원 (INTEGER NOT NULL) : 그룹의 제한인원(최대인원) 값을 저장하는 속성이다.
- 그룹 인원수 (INTEGER NOT NULL) : 그룹의 현재 인원을 저장하는 속성으로 그룹을 처음 생성했을 때는 값이 1이고, 인원이 추가될 때마다 1씩 증가하며 제한인원과 같아지면 더 이상 증가할 수 없다.
- 그룹코드 (TEXT NOT NULL) : 그룹에 자신이 원하는 팀원들만이 참여할 수 있도록 하는 용도로 사용되는 그룹코드 속성이다. 숫자나 문자 상관없이 입력할 수 있고, 그룹코드를 지정해놓은 그룹에 참여하기 위해서는 올바른 그룹코드를 입력해야한다.
- 그룹 생성 시간 (TEXT NOT NULL) : 그룹의 생성 시간을 저장하는 속성이다. 현재 사용자가 그룹에 참여(또는 생성)하는 방법이 사용자 테이블에 G1 ~ G4 각 4개의 속성을 두고, 각 속성에 참여 또는 생성할 그룹의 아이디를 입력해줌으로써 사용자가 그룹에 참여된다. 그렇다면 그룹을 생성하는 경우 먼저 그룹을 생성하고 생성한 그룹의 아이디 값을 사용자의 G1 ~ G4 속성에 입력해주어야한다. 그런데 생성된 그룹의 아이디 값은 AUTO INCREMENT이기 때문에 어떤 값인지 알 수 없다. 그렇기 때문에 생성 시간을 따로 만들어서 그룹을 생성한 후 생성 시간에 만들어진 그룹의 아이디를 가져온 후 이 그룹 아이디 값을 그룹을 생성한 사용자의 속성에 입력해주는 방식으로 진행이 된다.

② 사용자 테이블
- 사용자 ID 속성 (TEXT PRIMARY KEY) : 사용자들을 구분해주는 속성이다.
- 사용자 비밀번호 속성 (TEXT NOT NULL) : 로그인 시 사용자가 맞는지 확인하기 위한 값으로 사용되는 속성이다.
- 사용자 이름 (TEXT NOT NULL) : 사용자의 이름을 저장하는 속성이다.
- 사용자 전화번호 (TEXT NOT NULL) : 사용자의 전화번호를 저장하는 속성이다.
- 사용자 이메일 (TEXT NOT NULL) : 사용자의 이메일을 저장하는 속성이다.
- 사용자 생년월일 (TEXT NOT NULL) : 사용자의 생년월일을 저장하는 속성으로 비밀번호를 찾기 위해서 ID속성, 전화번호 속성과 함께 사용된다.
- 그룹1 ~ 그룹4 속성 (TEXT NOT NULL) : 사용자가 참여되어있는 그룹들의 아이디 값을 저장하는 속성으로, 참여된 그룹이 없는 경우는 –1의 값을 가진다.
- 역할1 ~ 역할4 속성 (TEXT NOT NULL) : 사용자가 참여되어있는 그룹에서의 역할을 저장하는 속성으로, 참여된 그룹이 없는 경우는 'x' 값을 가지고 그룹을 직접 생성한 경우는 '팀장' 값을 가진다. 그룹에 참여하는 경우에는 '팀원' 값을 가지며 '팀장'역할은 팀장이 변경할 수 있다.

③ (참여)신청 테이블
- 그룹 신청 ID 속성 (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 신청 내역 데이터들을 구분하는 속성이다.
- 그룹 ID 속성 (INTEGER NOT NULL) : 사용자가 참여신청을 한 그룹의 아이디 값을 저장하는 속성이다.
- 사용자 ID 속성 (TEXT NOT NULL) : 그룹에 참여신청한 사용자의 ID 값을 저장하는 속성이다.
- 참여신청 테이블의 구성을 보면 사용자가 참여신청한 그룹의 아이디 값을 가지고 있으므로 해당 그룹의 팀장에게 참여신청 내역을 보여줄 수 있다. 참여신청에 대해 팀장이 수락한 경우는 신청한 사용자의 ID 값을 이용해 사용자의 G1 ~ G4 속성 중 하나의 속성에 현재 그룹의 아이디 값을 입력해줌으로써 그룹에 참여시킬 수 있고, 역할 역시 '팀원'으로 변경시켜주면 된다.
- 수락한 경우는 수락 데이터가 추가되고, 거절한 경우는 거절 데이터가 추가된다.

④ 수락 테이블
- 신청 수락 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 신청 내역에 대한 수락 정보들을 구분하는 속성이다.
- 그룹 ID (INTEGER NOT NULL) : 수락한 그룹이 어떤 그룹인지를 저장하는 속성이다.
- 사용자 ID (TEXT NOT NULL) : 신청한 사용자의 ID를 저장하는 속성이다.
- 사용자의 ID속성을 이용하면 각 사용자들이 신청한 내역에 대한 수락 내역들을 불러와서 보여줄 수 있다.

⑤ 거절 테이블
- 신청 수락 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 신청 내역에 대한 거절 정보들을 구분하는

속성이다.
- 그룹 ID (INTEGER NOT NULL) : 거절한 그룹이 어떤 그룹인지를 저장하는 속성이다.
- 사용자 ID (TEXT NOT NULL) : 신청한 사용자의 ID를 저장하는 속성이다.
- 사용자의 ID속성을 이용하면 각 사용자들이 신청한 내역에 대한 거절 내역들을 불러와서 보여줄 수 있다.

⑥ 공지사항 테이블
- 공지 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 공지의 정보를 구분하는 속성이다.
- 그룹 ID (INTEGER NOT NULL) : 공지가 작성된 그룹 ID 값을 저장하는 속성이다.
- 멤버 ID (TEXT NOT NULL) : 공지를 작성한 사용자의 ID 값을 저장하는 속성이다.
- 공지 제목 (TEXT NOT NULL) : 공지의 제목을 저장하는 속성이다.
- 공지 타입 (TEXT NOT NULL) : 공지의 타입(일반/중요)을 저장하는 속성이다.
- 공지 내용 (TEXT NOT NULL) : 공지의 내용을 저장하는 속성이다.
- 공지 시간 (TEXT NOT NULL) : 공지 작성 시간을 저장하는 속성이다.

⑦ 출석 테이블
- 출석 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 출석의 정보를 구분하는 속성이다.
- 출석 코드 (INTEGER NOT NULL) : 팀장이 생성한 출석 코드를 저장할 속성이다.
- 그룹 ID (INTEGER NOT NULL) : 출석 코드가 생성된 그룹 ID 값을 저장하는 속성이다.
- 멤버 ID (TEXT NOT NULL) : 출석 코드를 생성한 팀장의 ID 값을 저장하는 속성이다.
  (해당 팀장에게 출석 여부 수정 권한을 주기 위함)
- 날짜 (TEXT NOT NULL) : 출석 코드를 생성한 날짜 및 시간을 저장하는 속성이다.
- 출석 확인 여부 (INTEGER NOT NULL) : 팀장이 팀원들의 출석을 모두 확인했는지 구분하는 속성이다.
  (미확인(기본값) 시 값은 0, 확인 시 1)

⑧ 출석 여부 테이블
- 출석 여부 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 사용자의 출석 여부 정보를 구분하는 속성이다.
- 그룹 ID (INTEGER NOT NULL) : 출석 여부를 판단하고자 하는 그룹의 ID 값을 저장하는 속성이다.
- 멤버 ID (TEXT NOT NULL) : 출석 여부 해당자의 ID 값을 저장하는 속성이다.
- 출석 ID (INTEGER NOT NULL) : 생성된 출석의 ID를 저장하는 속성이다.
- 출석 여부 (TEXT NOT NULL) : 사용자들의 출석 상태를 저장하는 속성이다. (출석/지각/결석(기본값))

⑨ SCHEDULE 테이블
- 일정 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 일정의 정보를 구분하는 속성이다.
- 그룹 ID (INTEGER NOT NULL) : 일정이 생성된 그룹 ID 값을 저장하는 속성이다.
- 멤버 ID (TEXT NOT NULL) : 일정을 생성한 멤버 ID 값을 저장하는 속성이다.
- 일정 이름 (TEXT NOT NULL) : 일정 이름을 저장하는 속성이다.
- 일정 종료 (TEXT NOT NULL) : 일정 종류(개인/그룹)를 저장하는 속성이다.
- 일정 날짜 (TEXT NOT NULL) : 일정의 날짜를 저장하는 속성이다.
- 시작 시간 (TEXT NOT NULL) : 일정의 시작 시간을 저장하는 속성이다.
- 종료 시간 (TEXT NOT NULL) : 일정의 종료 시간을 저장하는 속성이다.

⑩ TASK 테이블
- Task ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 Task들을 구분하는 속성이다.
- 그룹 ID (INTEGER NOT NULL) : TASK들을 그룹별로 구분하여 검색하기 위해 사용되는 속성이다.
- 사용자 ID (TEXT NOT NULL) : TASK는 그룹의 팀장만이 생성할 수 있기 때문에 팀장의 ID값이 저장될 것이다. 이 사용자 ID 속성은 팀장인 회원이 회원탈퇴 또는 그룹탈퇴를 하는 경우 그동안 작성했던 TASK들을 모두 삭제하기 위한 용도로 사용된다.

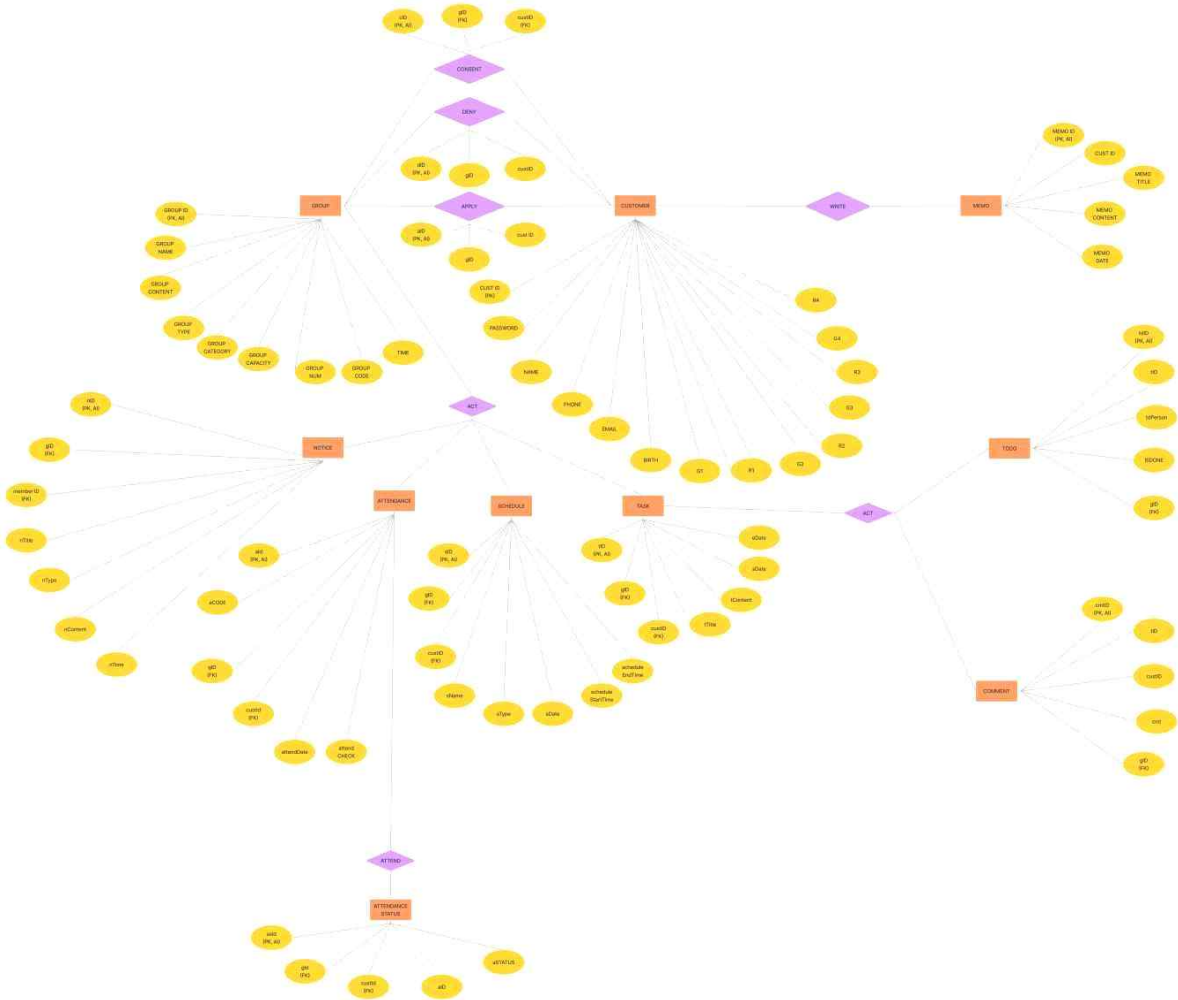
- Task 제목 (TEXT NOT NULL) : Task의 제목을 저장하는 속성이다.
- Task 내용 (TEXT NOT NULL) : Task의 내용을 저장하는 속성이다.
- 시작일자 (TEXT NOT NULL) : Task의 시작 일자를 저장하는 속성이다.
- 종료일자 (TEXT NOT NULL) : Task의 종료 일자를 저장하는 속성으로, 그룹 메인 페이지의 상단 배너에 가장 가까운 Task의 마감일을 표시해주어 그룹원들이 잊지 않고 확인할 수 있도록 해주는 기능도 한다.

⑪ ToDO 테이블
- ToDo ID (PRIMARY KEY AUTO INCREMENT) : 각 ToDo들을 구분하는 속성이다.
- Task ID (INTEGER NOT NULL) : ToDo들을 Task별로 구분하여 검색하기 위한 속성이다.
- ToDO 담당자 (TEXT NOT NULL) : ToDo의 담당자를 저장하는 속성이다.
- ToDo 완료 여부 (INTEGER NOT NULL) : 0과 1의 값을 가지며 초기 값은 0이다. 0은 아직 ToDo가 완료되지 않았다는 의미를 가지며, 1은 ToDo가 완료되☝다는 의미를 가진다.
- 그룹 ID (INTEGER NOT NULL) : 그룹이 삭제되는 경우에 해당 그룹의 ToDo 데이터들도 모두 삭제하기 위한 속성이다.

⑫ Comment 테이블
- Comment ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 Comment들을 구분해주는 속성이다.
- Task ID (INTEGER NOT NULL) : Comment들을 Task별로 구분하여 검색하기 위한 속성이다.
- 사용자 ID (TEXT NOT NULL) : Comment를 작성한 사용자가 누군지 알기 위해 사용되는 속성이다.
- Comment 내용 (TEXT NOT NULL) : Comment 내용을 저장하는 속성이다.
- 그룹 ID (INTENGER NOT NULL) : 그룹이 삭제되는 경우에 해당 그룹의 Comment 데이터들을 모두 삭제하기 위해 사용되는 속성이다.

⑬ Memo 테이블
- 메모 ID (INTEGER PRIMARY KEY AUTO INCREMENT) : 각 메모를 구분하는 속성이다.
- 사용자 ID (TEXT NOT NULL) : 메모를 작성한 사용자의 ID를 저장하는 속성이다.
- 메모 제목 (TEXT NOT NULL) : 메모의 제목을 저장하는 속성이다.
- 메모 내용 (TEXT NOT NULL) : 메모의 내용을 저장하는 속성이다.
- 메모 날짜 (TEXT NOT NULL) : 메모 작성 날짜를 저장하는 속성이다.

(2) 변수 코드명 버전 ERD



- 위의 ERD를 안드로이드 스튜디오 프로젝트에서 사용한 변수명으로 나타낸 ERD이다.

8. 결론

1) 기능 및 효과

개발한 앱 STeam을 이용하는 사용자들은 팀 프로젝트나 스터디 그룹을 직접 생성하거나 생성되어있는 그룹에 참여할 수 있다. 사용자들은 그룹 내에서 지원되는 공지사항, 출석, Schedule, Work 기능을 이용함으로써 그룹 활동이 원활히 진행되도록 도움을 받을 수 있다.

사용자들이 그룹 내에서의 중요한 공지를 놓치지 않고 확인할 수 있도록 공지의 타입을 중요공지와 일반공지로 나누☝으며, 중요공지는 그룹 메인페이지의 가장 상단에 표시되도록 하였다.

출석 기능은 팀원들이 그룹활동에 제대로 참여하지 않는 것을 막기 위해 구현되☝으며, 팀 프로젝트 같은 경우 제대로 참여하지 않은 팀원의 출석 내역을 교수님께 증거로 제출하여 평가에 반영되게 할 수 있다.

그룹활동의 경우 팀원들 간의 일정 조율이 가장 중요한 문제인데, 그룹 일정(그룹 회의 등)을 등록할 때 팀원들의 일정이 해당 날짜에 있는 경우 경고 메세지를 표시해줌으로써 모든 팀원이 참여할 수 있는 날짜에 그룹 일정이 등록될 수 있도록 구현되어있다.

일정조율 다음으로 중요하다고 생각하는 기능은 업무분담으로 담당 페이지는 Work 페이지이다.

Work페이지 내에서는 팀장이 Task(업무)를 등록할 수 있고, Task가 완료되기 위해서는 모든 팀원들이 역할을 분담하여 수행해야 하는데 역할분담을 도와주는 기능이 ToDo리스트 기능이다. 팀장은 ToDo 기능을 이용하여 팀원들에게 각자의 업무를 할당해줄 수 있으며 업무가 완료되☝다고 판단하는 경우에는 체크박스의 체크표시를 해줄 수 있다. Task를 진행하는데에 있어서 궁금증이나 공유해야할 의견이 있으면 comment 기능을 이용하여 팀원들과 소통이 가능하다.

앱 STeam을 이용함으로써 특히 대학생들이 팀 프로젝트나 스터디활동을 원활하고 체계적으로 진행할 수 있을 것으로 기대된다.

2) 계획서와 일치하지 않은 부분과 개발 완료되지 않은 부분

(1) 기기들 사이의 연결 문제

Android Studio기기에 내장되어있는 데이터베이스인 SQLite를 사용했다는 점에서 외부 기기와 상호작용이 되지 않는다는 문제가 있다. 이에 대해서는 교수님에게 문의한 결과 Proxy Layer 등을 이용하여 중계하는 방법을 사용하거나 데이터베이스를 MySQL등의 외장 대형 서버를 이용하여 해결할 수 있을 것으로 생각된다. 그렇지만 한 기기 내에서 여러 회원정보를 생성하고 이 회원들 사이에서 그룹을 생성하고 그룹 내에서 활동을 해본 결과 구현한 모든 기능이 정상적으로 작동한다. 결국 사용하는 데이터베이스의 종류가 다를 뿐이지 데이터베이스에 정보를 저장하고 불러오는 방법과 적절한 Query문을 사용하여 상황에 맞게 데이터를 처리하는 방법을 모두 올바르게 이해하였다고 생각한다.

(2) 자료(ppt, 이미지 등)저장 및 관리기능

팀 프로젝트나 스터디 활동을 하는 과정에서 필요한 문서나 이미지 등의 자료를 등록하고 공유할 수 있도록 기능을 구현하는 것이 초기 계획이☝으나 PPT 등의 문서를 관리하고 공유하는 방법을 찾기 어려워서 일단 구현을 보류하였다. SQLite에서 BLOB형태로 이미지를 저장할 수 있도록 지원해주고, 수업시간에 txt파일을 관리하는 방법은 배웠기 때문에 이미지와 txt파일에 대해서만이라도 저장 및 공유할 수 있도록 구현하는 방향으로 계획을 수정하였으나 결국에는 시간상 문제로 구현하지 못했다.

(3) Schedule을 보여주는 방식의 변경

등록된 일정이 있는 날짜의 경우는 별도로 표시해주는 방향으로 구현을 계획하였으나 커스텀 연결과정에서 종속성 문제가 발생하여 구현하지 못했다. 이 부분에 대해서는 캘린더뷰의 날짜를 클릭하였을 때 해당 날짜에 등록된 일정이 있는 경우 아래의 레이아웃에 일정들을 보여주고, 없는 경우 빈 레이아웃을 보여주는 방식으로만 일정을 표시하기로 결정하였다.

9. 향후 수행과제 및 보완사항

1) 기기 내장 데이터베이스인 SQLite의 한계 해결

결론 내용에서 설명하였듯 기기에 내장된 데이터베이스인 SQLite를 이용하여 개발했기 때문에 외부에 서버가 따로 존재하는 것이 아니라 다른 기기와의 데이터 공유가 되지 않는다. 이 부분을 해결하기 위해서는 사용하는 데이터베이스를 외부 대형 데이터베이스(My SQL 등)으로 변경하거나 교수님께서 알려주신 Proxy Layer등을 이용하여 중계해주는 방법을 사용하여 해결할 수 있을 것으로 생각된다. 앱 STeam은 사용자들간의 상호작용이 있어야만 존재할 수 있는 앱이기 때문에 이 부분을 가장 우선적으로 보완해야한다고 생각된다.

2) 자료 공유 및 관리 기능 추가

자료 공유 및 관리 기능은 저희 조가 유일하게 아예 구현하지 못한 기능이다. 그룹 활동을 진행하는 데에 있어서 자료(ppt, pdf, 문서 등)나 이미지 등을 공유하는 기능이 있다면 크게 도움을 줄 수 있다. 아래의 중간발표 이후 출처6번과 8번을 보시면 아시겠지만 SQLite를 이용하여 이미지를 저장하는 방법, 내부 저장소와 외부 저장소를 이용하여 파일을 관리하는 방법을 찾아보았다. 그렇지만 3명의 인원으로 다른 과목들을 공부하며 나머지 기능들을 완성하기에도 시간이 굉장이 빠듯하였기 때문에 결국 앱의 기능에서 삭제하게 되었다. 위의 여러 기기들 사이의 연결문제를 제외하고 가장 우선적으로 보완되어야하는 기능으로 판단된다.

3) 사용자와 그룹사이의 참여 관계 변경

현재는 사용자 테이블에 4개의 그룹 아이디 속성(G1~G4)과 4개의 역할 속성(R1~R4)를 두어 총 8개의 속성을 이용하여 사용자와 그룹 사이의 참여 관계를 관리하기 때문에 한 사용자는 최대 4개의 그룹까지만 참여할 수가 있다. 처음에 그룹 테이블과 사용자 테이블 2개의 테이블만 가지고 참여관계를 구현하려고 하였기 때문에 이와 같이 참여 가능한 그룹의 수에 제한이 생긴 것이다. 이후에 출석 기능을 구현할 때는 출석을 하나의 테이블이 아닌 2개의 테이블(출석 테이블과 출석여부 테이블)로 나누어 구현하였는데, 이렇게 하면 데이터를 더욱 효과적으로 관리할 수 있다는 것을 알게되었다. 그룹과 사용자의 참여 관계도 이와 같이 그룹 테이블과 사용자 테이블 외에 사용자의 참여 그룹과 역할을 관리하는 하나의 테이블을 더 추가하여 관리한다면 사용자가 참여할 수 있는 그룹 수의 제한을 없앨 수 있다. 그렇지만 일단은 한 사람이 4개 이상의 팀 프로젝트나 스터디를 수행하는 경우는 흔하지 않을 것으로 생각되기 때문에 사용자들의 요청이 있는 경우 데이터베이스에 관리 테이블을 하나 추가하여 참여 관계를 개선하면 될 것으로 보인다.

4) 캘린더뷰 커스텀 연결 문제 해결

캘린더뷰에 팀원의 일정이 있는 날짜에는 별도로 표시(색상의 변경 또는 아이콘 표시 등)를 해주기 위해 커스텀 연결을 시도해보았으나 종속성 문제가 발생하여 적용하지 못하였다. 커스텀 연결 이후에 적용하는 방법까지는 모두 찾아놓은 상태라 종속성 문제만 해결한다면 적용이 가능하였지만, 시간상 문제로 완전히 구현하지 못했다. 이 부분을 해결한다면 팀원들의 일정이 언제 있는지, 어느 날짜에 그룹 일정을 잡으면 좋을지 등을 더욱 쉽게 알 수가 있을 것이다. 그렇지만 이 기능이 없다고 크게 문제가 되지는 않을 것으로 판단되기 때문에 당장 중요한 기능들을 먼저 보충한 후에 보충해야할 기능이다.

10. 참고문헌 및 사이트

[~중간발표]

1) 서적

·우재남, 박길식. (2022). Android Studio를 이용한 안드로이드 프로그래밍. 한빛아카데미

·Adam Stroud. (2017). 안드로이드 데이터베이스. (오세봉, 김기환 역). 에이콘(원서출판 2017)

2) 웹 사이트

·Google, "Cloud Firestore 시작하기", Firebase, 2022.11.01.,
https://firebase.google.com/docs/firestore/quickstart?authuser=0#java.

·Android, "사용자 데이터 및 ID", Android Developers, 2022.01.28.,
https://developer.android.com/guide/user-data?hl=ko, https://developer.android.com

3) YouTube

·Bored Developer, 안드로이드 + 파이어베이스 SNS 앱 만들기 part2 [회원가입 화면 구현], 2019.04.07.,
https://www.youtube.com/watch?v=AeDQjsmy_L8&list=PLQAhd9lzs3ycyGhmQhXaKNYcCQn_28Ems&index=2

·Bored Developer, 안드로이드 + 파이어베이스 SNS 앱 만들기 part3 [로그인 화면 구현], 2019.04.08.,
https://www.youtube.com/watch?v=79OvqVryMd4&list=PLQAhd9lzs3ycyGhmQhXaKNYcCQn_28Ems&index=3

·Bored Developer, 안드로이드 + 파이어베이스 SNS 앱 만들기 part4 [비밀번호 재 설정 화면 및 회원 정보 추가 화면], 2019.04.08.,
https://www.youtube.com/watch?v=PFXsrG0Y36g&list=PLQAhd9lzs3ycyGhmQhXaKNYcCQn_28Ems&index=4

·Bored Developer, 안드로이드 + 파이어베이스 SNS 앱 만들기 part5 [회원 정보 추가 화면 DB 연결], 2019.04.08.,
https://www.youtube.com/watch?v=J6qckg6Vdpc&list=PLQAhd9lzs3ycyGhmQhXaKNYcCQn_28Ems&index=5

·hongdroid 홍드로이드, 안드로이드(Android Java) 강의, 2019.,
https://www.youtube.com/watch?v=UNKlX9J6m-A&list=PLC51MBz7PMyyyR2l4gGBMFMMUfYmBkZxm

·Code Vedanam, Dynamic Views | Programmatically add, remove and control view items |, 2021.04.20.,
https://www.youtube.com/watch?v=LjQmxg8xwGQ

4) 블로그

·김샤린, "[안드로이드] SQLite 로그인 기능", 네이버 블로그, 2017.04.21.,
https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=k_sharin&logNo=220988976110

·teamnova, "[JAVA][Android]안드로이드 스튜디오 QR코드 스캔하기", 네이버 블로그, 2021.09.19.,
https://stickode.tistory.com/233

·윤명준, "안드로이드 구분선 그리기의 모든 것", CRACKER Blog, 2018.03.16.,
https://blog.cracker9.io/2018/03/16/android_line/

·Siadaddy, "[Android] Layout 테두리 설정 및 모서리 둥글게 만들기", tistory, 2019.09.24.,
https://siadaddy-cordinglife.tistory.com/18

·aries574, "[안드로이드] 액티비티(Activity)에서 프래그먼트(Fragment) 데이터 전달하는 방법", tistory, 2022.04.11., https://aries574.tistory.com/287

·Wally Dev, "windowSoftInputMode", 네이버 블로그, 2020.03.17.,
https://blog.naver.com/tjck016/221858373105

·길상, "안드로이드 – EditText 입력 완료 후 엔터(Enter) 이벤트 처리, 네이버 블로그,
setOnEditorActionListener, 2019.06.26.,
https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=jogilsang&logNo=221571271854

[중간 발표 이후]

1.쌓인 스택 없애기 (화면 쌓이는 문제 해결)
https://www.masterqna.com/android/96459/%EC%8A%A4%ED%83%9D%EC%97%90-%EC%8C%93%EC%9D%B8-%EC%95%A1%ED%8B%B0%EB%B9%84%ED%8B%B0%EB%A5%BC-%EB%AA%A8%EB%91%90-%EC%A2%85%EB%A3%8C%ED%95%98%EB%8A%94-%EB%B0%A9%EB%B2%95%EC%9D%80-%EB%AC%B4%EC%97%87%EC%9D%B8%EA%B0%80%EC%9A%94-%EC%BD%94%ED%8B%80%EB%A6%B0
2.뒤로가기 두 번 시 종료
https://winaire.tistory.com/14
3.다이얼로그 버튼 클릭 시 종료 막는 방법
https://fullstatck.tistory.com/13
4.datepicker dialog
https://hroad.tistory.com/10
5. 예외처리 오류발생 해결 (CursorIndexOutOfException)
https://itecnote.com/tecnote/android-cursorindexoutofboundsexception-index-0-requested-with-a-size-of-0/
6.SQLite 이미지 저장방법 (찾아만 놓고 시간이 부족하여 구현 못함)
https://wikidocs.net/15881
7.자바 현재 시간 날짜 구하기
https://202psj.tistory.com/1504
8. 파일 저장방법 (내부저장소/외부저장소) - 역시 기술적, 시간적 문제로 구현 못함
https://javapp.tistory.com/126
9.취소선
https://appsnuri.tistory.com/490
10. 키보드 enter나 키보드 내 검색 버튼 기능 주기
https://blog.naver.com/PostView.nhn?blogId=jogilsang&logNo=221571271854&categoryNo=80&parentCategoryNo=0&viewDate=&currentPage=1&postListTopCurrentPage=1&from=postView
11.키보드 자판에 검색 버튼 나오게 하는 법
https://naver.me/GWrJYfZ2
12. split 함수 안 될 때
https://ddioniii.tistory.com/m/5
13.AlertDialog 커스텀
https://beinone.tistory.com/1
https://onepinetwopine.tistory.com/201
14.RadioButton 커스텀
https://curryyou.tistory.com/395
15. 캘린더 커스텀
https://aries574.tistory.com/295
https://cpcp127.tistory.com/21
https://gameprograming.tistory.com/124
https://onlyfor-me-blog.tistory.com/437
16.스플래시 화면
https://developer.android.com/about/versions/12/features/splash-screen?hl=ko
http://sjava.net/2022/06/android-12-splash-screen/
https://stickode.tistory.com/280
17. 타임피커 커스텀
https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=websearch&logNo=220624662259
https://soir1984.tistory.com/m/31

18.토스트메시지 커스텀

https://aries574.tistory.com/78

https://jaejong.tistory.com/5

19.폰트 적용

https://stickode.tistory.com/51

https://bongcando.tistory.com/12