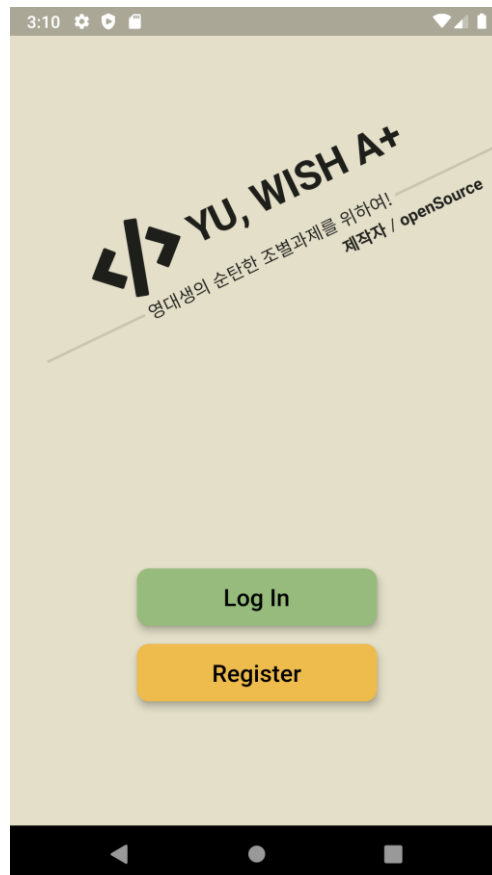


2. SDS (Software Design Specification)

YU, wish A+



Student No	Name	E-mail
21611733	박재웅 (조장)	jaewoongpark@ynu. ac. kr
21611741	손광진	skj2393@ynu. ac. kr
21611749	우재석	presls819@ynu. ac. kr
21611777	최성찬	ekdnlt8520@ynu. ac. kr
21611783	홍덕화	21611673@ynu. ac. kr

[Revision history]

Revision date	Version #	Description	Author
10/28/2020	1.01	Class Diagram 수정	박재웅, 손광진, 우재석, 최석찬 호덕화
11/06/2020	1.02	Sequence Diagram 수정	박재웅, 손광진, 우재석, 최석찬 호덕화
11/24/2020	1.03	기타 오류 수정	박재웅, 손광진, 우재석, 최석찬 호덕화
11/27/2020	1.04	일부 기능 개선	박재웅, 손광진, 우재석, 최석찬 호덕화

= Contents =

1. Introduction	4
2. Use case analysis	8
3. Class diagram	59
4. Sequence diagram	89
5. State machine diagram	119
6. User interface prototype	124
7. Implementation requirements	128
8. Glossary	129
9. References	130

1. Introduction

[Summary]

이 보고서는 'YU, wish A+' 프로젝트의 디자인 단계 보고서이며, SRS 단계의 엑셀 파일 뒤를 이은 보고서임을 밝힌다. 지난 단계들에서 'YU, wish A+'를 위한 가장 기본적이고 핵심적인 부분들을 고려했다면, 이번 SDS 단계에서는 실제 작동될 기능들과 Class 그리고 메소드에 집중한다.

[Prominete features of project]

이 어플리케이션의 주된 목적은 영남대학교 학생들의 조별 과제를 돕기 위한 것이다. 대학생들은 다양한 전공 과목과 교양 과목을 통해, 다양한 과제들을 경험하게 된다. 많은 학생들은 개인 과제를 선호하는 모습을 보이지만, 조별 과제가 필수적인 수업들이 있다. 그들은 많은 사람들과 함께 과제를 하는 것에 큰 부담을 느낄 것이다. 누군가는 특정 과목에 조별 과제가 있다는 이유로 해당 과목 수강을 포기하는 경우도 잦은 편이다.

따라서 대부분의 학생들은 해당 과제를 위해 평소 친한 동기들, 선후배들과 함께 조를 구성할 것이다. 친한 친구들과 선후배들은 서로간의 믿음과 적절한 의사소통으로 조별과제를 보다 잘 마무리할 수 있을 것이다. 하지만 모든 학생들이 평소 친한 친구들과 함께 수업을 듣지는 못한다. 누군가는 수업에서 아는 사람 없이 혼자 수강하는 학생도 있을 것이다. 어쩌면 수업에서 남은 몇 명의 학생들은 주도적으로 팀원을 구하지 못하고, 누군가는 남은 인원들과 어쩔 수 없이 조를 구성하게 될지도 모른다. (어쩌면 그 남은 인원들이 책임감이 적고, 수업에 관심이 없을지 모름에도 불구하고!)

위와 같은 경우에서 대부분의 학생들은 수동적인 조별 과제를 경험하게 된다. 조원들간에 서로 조장을 미루고, 할 일을 하지 않으며 약속을 미루는 모습은 조별 과제에 대한 부정적 경험을 줄 것이다. 조원 모두가 책임감 있게 참여한 조별 과제와는 다르게, 서로 역할과 책임을 떠넘기는 이기적인 관계에서는 얻을 수 있는 경험은 적을 것이다.

'YU, wish A+'은 영남대학교 학생들의 주도적인 조별 과제 구성을 돕는다. 또한 개인적인 프로젝트, 자격증, 공모전의 인원을 모으는 작업을 도울 것이다. 이 어플리케이션을 통해, 많은 학생들이, 조별과제의 인원에 대한 배신감과 회의감을 받지 않도록 돕는 것이 이 'YU, wish A+'의 주된 목적이다.

[Prominete features of Document]

1. Use case Diagram, Class Diagram, Sequence Diagram, State Diagram
2. User interface Prototype(Figma)
3. 시스템 요구 사항
4. 용어 설명
5. 참조

각 Diagram은 UML 표준 지침을 최대한 따른다. 하지만 Class Diagram을 작성하는 과정에서, Flutter와 Dart 언어를 선택함으로 인해 발생할 수 있는 모호한 표기가 해석의 다양성을 일으킬 수 있다. 모호한 정보들은 각 Diagram 아래에 추가적으로 표기하여, 보고서를 읽는 사람이 착각할 수 있는 부분을 최소화함을 밝힌다.

2. Use case analysis

[Use case diagram]



위의 그림은 SRS 문서에서 정의했던 'YU, wish A+' 어플리케이션의 기능들을 구분하여 작성된 Use case diagram이다. User는 어플리케이션을 통해 아래와 같은 기능들을 사용할 수 있다.

기본적인 기능으로 사용자는 로그인하기, 로그아웃, 개발자 보기, 오픈소스 보기, 회원가입하기, 내 프로필 작성하기, 내 프로필 수정, 프로필 사진 수정하기, 내 프로필 보기, 프로필 활성화/비활성화 하기, 가입자 프로필 조회, 가입자 프로필 검색하기 기능을 사용할 수 있다.

소통하기 기능으로는 쪽지 보내기, 쪽지함 확인하기, 이메일 보내기, 전화하기 기능을 사용할 수 있다.

글 열람하기 기능으로는 게시글 열람하기, 대외 활동 게시판 열람하기, 조별 게시판 열람하기 기능을 사용할 수 있으며, 게시글 상세보기, 제거하기 기능을 사용할 수 있다.

사용자는 글 작성하기 기능으로 대외활동 게시글 작성, 게시글 작성 그리고 조별 게시글 작성을 할 수 있다.

글 수정하기와 글 삭제하기 기능을 사용할 수 있는데, 대외활동 게시글을 수정하고, 게시글을 수정하고 조별 게시글을 수정할 수 있으며, 대외 활동 게시판을 삭제하고, 게시글을 삭제하며 마지막으로 조별 게시글을 삭제할 수 있는 기능이 제공된다.

[The form of Use case description]

Use case #1 : 로그인하기	
GENERAL CHARACTERISTICS	
Summary	User가 'YU, wish A+' 사용을 위해 회원인증을 하여, YU, wish A+가 제공하는 서비스를 이용할 수 있도록 한다.
Scope	YU wish A+
Level	User level
Author	박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 회원가입을 완료한 상태여야 한다.
Trigger	User가 'YU, wish A+'가 제공하는 서비스를 사용하기 위해 'YU, wish A+'의 login screen에서 이메일과 비밀번호를 입력한 후 회원 인증을 받으려고 할 때
Success Post Condition	User는 'YU, wish A+'를 사용할 수 있는 상태가 되며, 프로필 작성여부에 따라 'YU, wish A+'가 create profile screen을 출력하거나 main screen을 출력한다.
Failed Post Condition	User는 'YU, wish A+' 사용 허가를 얻지 못한다.

MAIN SUCCESS SCENARIO	
Step	Action
S	User가 'YU, wish A+'에 로그인한다.
1	이 Use case 는 User가 'YU, wish A+'가 제공하는 서비스를 사용하기 위해 login screen에서 로그인할 때 시작한다.
2	User는 로그인하기 위해 'YU, wish A+'의 welcom screen의 'Login'버튼을 click한다.
3	'YU, wish A+'는 login screen을 출력한다.
4	User는 'YU, wish A+'의 login screen에서 로그인을 할 수 있다.
5	User는 'YU, wish A+'의 login screen에서 이메일과 비밀번호를 입력하고 'Login'버튼을 click한다.
6	'YU, wish A+'는 Google Firebase Authentication System에 접근해 등록된 회원인지 확인한다.
7	'YU, wish A+'는 회원의 프로필 작성 여부에 따라 create profile screen을 출력하거나 main screen을 출력한다.
8	이 Usecase는 User가 'YU, wish A+'에 로그인이 성공하거나 실패하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
7	<p>7a. Google Firebase Authentication System에 저장되지 않은 이메일로 로그인하려는 경우 로그인에 실패한다. ...7a1. 'YU, wish A+'는 welcom screen으로 이동하고 오류 다이얼로그를 출력한다.</p> <p>7b. 비밀번호가 잘못되어 로그인에 실패한다. ...7b1. 'YU, wish A+'는 welcom screen으로 이동하고 오류 다이얼로그를 출력한다.</p>

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	회원 당 1번
<Concurrency>	제한 없음
Due Date	

Use case #2 : 회원가입
GENERAL CHARACTERISTICS

Summary	User가 YU, wish A+를 사용하기 위해 회원등록을 한다.
Scope	YU wish A+
Level	User level
Author	박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 영남대 이메일을 가지고 있어야한다.
Trigger	User가 'YU, wish A+'가 제공하는 서비스를 이용하기 위해 register screen에 등록할 이메일과 비밀번호를 입력한 후 회원등록을 하려고 할 때
Success Post Condition	'YU, wish A+'는 User가 입력한 이메일과 비밀번호를 Google Firebase Authentication System에 저장하고 User는 'YU, wish A+'가 제공하는 서비스를 이용할 수 있는 회원이 된다.
Failed Post Condition	User는 'YU, wish A+'에 회원등록을 실패하고 'YU, wish A+'가 제공하는 모든 서비스를 이용할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 'YU, wish A+'에 회원가입을 한다.
1	이 Use case 는 User가 'YU, wish A+'가 제공하는 서비스를 이용하기 위해 register screen에서 회원등록을 할 때 시작한다.
2	User는 회원가입하기 위해 'YU, wish A+'의 welcom screen의 'Register'버튼을 click한다.
3	'YU, wish A+'는 register screen을 출력한다.
4	User는 'YU, wish A+'의 register screen에서 회원등록을 할 수 있다.
5	User는 'YU, wish A+'의 register screen에서 등록할 영남대 이메일과 비밀번호를 입력하고 'Register'버튼을 click한다.
6	'YU, wish A+'는 Google Firebase Authentication System에 접근해 User가 입력한 이메일과 비밀번호를 바탕으로 회원을 추가한다.
7	'YU, wish A+'는 Google Firebase Authentication System에 회원등록을 완료한 후 welcom screen으로 이동한다.
8	이 Usecase는 User가 'YU, wish A+'에 회원등록을 성공하거나 실패하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
6	6a. User가 입력한 이메일이 영남대 이메일이 아닌 경우 회원등록에 실패한다. ...3a1. 'YU, wish A+'는 오류 다이얼로그를 출력한다.
	6b. User가 입력한 이메일이 Google Firebase Authentication System에 이미 등록된 이메일인 경우 회원등록에 실패한다. ...6b1. 'YU, wish A+'는 welcom screen으로 이동한다. ...6b2. 'YU, wish A+'는 오류 다이얼로그를 출력한다.
	6c. User가 입력한 비밀번호가 비밀번호 규정에 맞지 않으면 회원등록에 실패한다. ...6c1. 'YU, wish A+'는 welcom screen으로 이동한다. ...6c2. 'YU, wish A+'는 오류 다이얼로그를 출력한다.

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #3 : 내 프로필 작성하기	
GENERAL CHARACTERISTICS	
Summary	User는 본인의 프로필을 작성할 수 있다.
Scope	YU, wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 회원가입이 완료된 상태여야 한다.
Trigger	User가 'YU, wish A+'에 회원가입한 후 처음으로 'YU, wish A+' 사용하기 위해 로그인할 때
Success Post Condition	'YU, wish A+'은 User가 작성한 프로필 정보를 Google Firebase Database System에 추가한 후 mainScreen으로 이동한다.
Failed Post Condition	네트워크 연결 상태 이상으로 'YU, wish A+'는 Google Firebase Database System에 접근하여 프로필 추가를 못하고 User는 프로필 작성 기능 'YU, wish A+'이 제공하는 서비스를 이용할 수 없다.
MAIN SUCCESS SCENARIO	
Step	Action
S	User는 자신의 프로필을 작성한다.
1	이 Use case 는 프로필을 작성하지 않은 User가 로그인하여 'YU, wish A+'이 제공하는 서비스를 이용하려고 할 때 시작한다.
2	'YU, wish A+'는 로그인하려는 User의 프로필 작성 여부를 Google Firebase Database System에 접근하여 확인한다.
3	'YU, wish A+'는 로그인하려는 User의 프로필이 작성이 안되어있으면 createProfileScreen으로 이동한다.
4	User는 'YU, wish A+'이 이동한 createProfileScreen에서 프로필을 작성할 수 있다.
5	User는 자신의 프로필 정보를 작성한다.
6	User는 프로필 작성을 완료하기 위해 createProfileScreen의 '제출'버튼을 click한다.
7	'YU, wish A+'는 User가 작성한 프로필 정보를 Google Firebase Database System에 접근하여 추가한다.
8	'YU, wish A+'는 User가 작성한 프로필 정보를 Google Firebase Database System에 추가 후, mainScreen으로 이동한다.
9	User는 'YU, wish A+'가 제공하는 서비스를 이용할 수 있다.
10	이 Usecase는 User가 프로필 작성에 성공하거나 프로필 작성을 취소하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
5	5a. 스마트폰의 '뒤로가기'버튼을 click하면 프로필 작성이 취소된다. ...5a1. 'YU, wish A+'은 welcomScreen으로 돌아간다.
7	7a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 접근하지 못하여 필 작성에 실패한다. ...7a1. Google Firebase Database System에 User의 프로필 정보를 추가하지 않는다. 7b. User가 작성한 프로필 정보가 입력형식에 맞지 않으면 프로필 작성에 실패한다. ...7b1. 'YU, wish A+'는 에러 메시지를 출력한다. ...7b2. 프로필을 작성하는 단계로 돌아간다. (Use case #2-5) 7c. 프로필 필수 입력란이 비어있으면 프로필 작성에 실패한다. ...7c1. 'YU, wish A+'는 에러 메시지를 출력한다. ...7c2. 프로필을 작성하는 단계로 돌아간다 . (Use case #2-5)

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	회원당 1번
<Concurrency>	제한 없음
Due Date	

Use case #4-1 : 내 프로필 수정
GENERAL CHARACTERISTICS

Summary	User는 본인이 작성한 프로필을 수정 할 수 있다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 프로필 작성이 완료된 상태여야 한다.
Trigger	User가 'YU, wish A+'에 로그인한 후 profile screen에 출력된 프로필을 수정하려고 '프로필 수정' 버튼을 click할 때
Success Post Condition	'YU, wish A+'은 User가 수정한 프로필 정보를 Google Firebase Database System에 수정한 후 profile screen에 수정한 프로필 정보를 출력한다.
Failed Post Condition	네트워크 연결 상태 이상으로 'YU, wish A+'는 Google Firebase Database System에 접근하여 프로필 수정을 못하고 User는 프로필 수정 기능을 사용하지 못한다.

MAIN SUCCESS SCENARIO

Step	Action
S	User는 자신의 프로필을 수정한다.
1	이 Use case는 User가 프로필을 수정하려고 할 때 시작한다.
2	User는 profile screen의 '프로필 수정'버튼을 click한다.
3	'YU, wish A+'는 User가 프로필을 수정할 수 있는 형태(form)을 제공한다.
4	User는 'YU, wish A+'이 제공한 형태(form)을 통해 프로필을 수정할 수 있다.
5	User는 수정할 프로필 정보를 수정한다.
6	User는 프로필 수정작업을 완료하기 위해 'Save'버튼을 click한다.
7	'YU, wish A+'는 User가 수정한 프로필 정보를 Google Firebase Database System에 접근하여 수정한다.
8	'YU, wish A+'는 User가 수정한 프로필 정보를 Google Firebase Database System에 수정을 완료한 후, User에게 제공한 형태(form)를 회수한다.
9	'YU, wish A+'는 User가 수정한 프로필 정보로 profile screen에 프로필을 출력한다.
10	User는 수정한 프로필 정보를 profile screen에서 확인한다.
11	이 Usecase는 User가 프로필 수정에 성공하거나 프로필 수정을 취소하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
5	5a. 'YU, wish A+'이 제공한 형태(form)의 'Cancel'버튼을 click하면 프로필 수정이 취소된다. ...5a1. 'YU, wish A+'은 User에게 제공한 형태(form)를 회수한다. ...5a2. 'YU, wish A+'은 User의 기존 프로필 정보를 출력한다.
	5b. 스마트폰의 '뒤로가기'버튼을 click하면 프로필 수정이 취소된다. ...5b1. 'YU, wish A+'은 User에게 제공한 형태(form)를 회수한다. ...5b2. 'YU, wish A+'은 main screen으로 돌아간다.
7	7a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 접근하지 못하여 필 수정에 실패한다. ...7a1. Google Firebase Database System에 User가 수정하려고하는 프로필 정보로 반영도 않는다.
	7b. User가 수정하려고 하는 프로필 정보가 입력형식에 맞지 않으면 프로필 수정에 실패한다. ...7b1. 'YU, wish A+'는 에러 메시지를 출력한다. ...7b2. 프로필을 수정하는 단계로 돌아간다. (Use case #2-5)
	7c. 프로필 필수 입력란이 비어있으면 프로필 수정에 실패한다. ...7c1. 'YU, wish A+'는 에러 메시지를 출력한다. ...7c2. 프로필을 수정하는 단계로 돌아간다 . (Use case #2-5)

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #4-2 : 프로필 사진 수정하기
GENERAL CHARACTERISTICS

Summary	User는 본인의 프로필의 사진을 수정할 수 있다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 프로필 작성이 완료된 상태여야 한다.
Trigger	User가 프로필 사진을 수정하기 위해 'YU, wish A+'의 profile screen에서 프로필 사진을 click할 때
Success Post Condition	'YU, wish A+'은 User가 수정한 프로필 사진을 Google Firebase Storage에 수정한 후 profile screen에 수정이 반영된 프로필 사진을 출력한다.
Failed Post Condition	네트워크 연결 상태 이상으로 'YU, wish A+'는 Google Firebase Storage에 접근하여 프로필 사진을 수정 못하고 User는 프로필 사진 수정 기능을 사용할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User는 자신의 프로필 사진을 수정한다.
1	이 Use case 는 User가 프로필 사진을 수정하기 위해 profile screen에 출력된 프로필 사진을 click 할 경우 시작한다.
2	User가 프로필 사진을 수정하기 위해 profile screen에 자신의 프로필 사진을 click한다.
3	'YU, wish A+'는 Default Photos Application을 실행한다.
4	User는 실행된 Default Photos Application에서 수정하고 싶은 사진을 click한다.
5	Default Photos Application은 User가 선택한 사진 정보를 'YU, wish A+'에 전달한다.
6	Default Photos Application은 종료하고 다시 'YU, wish A+'의 profile screen로 돌아온다.
7	'YU, wish A+'는 Default Photos Application에서 전달된 사진정보를 Google Firebase Storage에 수정한다.
8	'YU, wish A+'는 Google Firebase Storage에 수정된 사진으로 profile screen에 출력한다.

10	이 Use case는 User가 프로필 사진 수정에 성공하거나 취소하면 끝난다.
EXTENSION SCENARIOS	
Step	Branching Action
4	4a. 스마트폰의 '뒤로가기'버튼을 click하면 프로필 사진 수정이 취소된다. ...4a1. Default Photos Application은 종료하고 다시 'YU, wish A+'의 profile screen로 돌아온다.
7	7a. 네트워크 연결 상태 이상으로 Google Firebase Storage에 접근하지 못하여 프로필 사진 수정에 실패한다. ...7a1. Google Firebase Storage에 User가 선택한 사진으로 프로필 사진을 수정하지 않는다.
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #5 : 내 프로필 보기
GENERAL CHARACTERISTICS

Summary	사용자는 본인이 작성한 프로필을 볼 수 있다.
Scope	YU, wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	사용자
Preconditions	사용자는 YU, wish A+에 회원가입과 프로필 작성이 완료된 상태여야 한다.
Trigger	YU, wish A+에 로그인한 후 내 프로필을 보려고 할 때
Success Post Condition	YU, wish A+은 현재 로그인한 사용자의 프로필 정보를 Google Firebase Database System에서 가져와 프로필 스크린에 출력한다.
Failed Post Condition	YU, wish A+은 현재 로그인한 사용자의 프로필 정보를 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 네트워크가 안정될 때 까지 LoadingProgressIndicator가 돌아간다.

MAIN SUCCESS SCENARIO

Step	Action
S	YU, wish A+ 사용자는 자신의 프로필을 열람한다.
1	이 Use case 는 YU, wish A+ 사용자가 프로필을 보려고 할 때 시작한다.
2	YU, wish A+ 사용자가 메인화면의 '내 프로필'버튼을 클릭한다.
3	YU, wish A+는 현재 로그인한 사용자를 파악하여 Google Firebase Database System에서 정보를 가져오고 가져올 때 까지 LoadingProgressIndicator를 실행한다. 프로필
4	YU, wish A+는 Google Firebase Database System에서 프로필 정보를 가져오면 프로필 스크린에 가져온 프로필 정보를 출력한다,
5	이 Use case 는 프로필 출력이 성공하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
3	3a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 프로필 정보를 가져오지 못하여 프로필 출력에 실패한다. ...3a1. 네트워크가 안정될 때 까지 LoadingProgressIndicator가 돌아간다.

RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #6 : 프로필 공개 활성화/비활성화
GENERAL CHARACTERISTICS

Summary	User가 'YU, wish A+'의 주소록에서 User의 프로필이 출력될지 여부를 설정할 수 있다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 로그인한 상태여야 한다.
Trigger	User가 'YU, wish A+'의 주소록에서 User의 프로필이 출력될지 여부를 설정하려고 profile screen에서 '프로필 비공개' 스위치를 click할 때
Success Post Condition	'YU, wish A+'는 현재 로그인 중인 User의 계정의 프로필 공개 여부를 Google Firebase Database System에서 설정하고 'YU, wish A+'의 다른 User들은 비공개된 User들의 프로필을 주소록에서 조회할 수 없게 된다.
Failed Post Condition	User는 프로필 공개 여부 설정을 실패하고 'YU, wish A+'의 다른 User들은 공개된 User들의 프로필만 주소록에서 조회할 수 있다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 프로필 공개 여부를 설정한다.
1	이 Use case 는 User가 프로필 공개 여부를 설정하려고 할 때 시작한다.
2	User는 프로필 공개 여부를 설정 하기위해 'YU, wish A+'의 profile screen의 '프로필 비공개'스위치를 click한다.
3	'YU, wish A+'는 현재 사용하고 있는 계정의 프로필 공개 여부를 Google Firebase Database System에 접근하여 설정한다.
4	'YU, wish A+'는 Google Firebase Database System에 현재 사용하고 있는 계정의 프로필 공개 여부 설정 값으로 출력한다.
5	'YU, wish A+'의 다른 User들은 'YU, wish A+'의 주소록에서 공개된 프로필만 조회할 수 있다.
6	이 Usecase는 User가 프로필 공개 여부를 설정하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
------	------------------

3

3a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 프로필 공개 여부 설정에 실패한다.

...3a1. User는 프로필 공개 여부 설정을 변경하지 못한다.

...3a2. 'YU, wish A+' 현재 프로필 공개 설정 값을 출력한다.

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #7 : 가입자 프로필 조회
GENERAL CHARACTERISTICS

Summary	User는 'YU, wish A+'를 이용하는 전체 가입자들의 프로필을 조회할 수 있다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	프로필을 조회하려는 User는 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 main screen에서 '주소록' 버튼을 click 하였을 때
Success Post Condition	'YU, wish A+'은 전체 가입자의 프로필을 리스트 형식으로 출력한다. 리스트의 아이템들은 이름, email, college, department, phone, comment 속성을 보여준다. 프로필을 비공개로 한 가입자의 경우 이름과 email 속성만 제공한다. User는 가입자들의 프로필을 확인 할 수 있다. User가 리스트 프로필을 long click 하면 자세히 보여준다. User가 리스트 프로필을 click 하면 소통하기 기능을 제공한다.
Failed Post Condition	User는 가입자 프로필 정보를 리스트 형식으로 볼 수 없다. 'YU, wish A+'은 Loading Prgress Indicator 화면을 띄운다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 가입자 프로필을 조회한다.
1	이 Use case는 'YU, wish A+'를 이용하는 사용자들을 조회하기 위해 User가 주소록 창으로 이동할 때 시작한다.
2	User가 주소록 버튼을 클릭한다.
3	'YU, wish A+'은 profile List screen으로 이동한다.
4	'YU, wish A+'은 Google Firebase Database System에 접근하여 가입자들의 정보를 이름 오름차순으로 가져온다.
5	'YU, wish A+'은 가져온 정보로 비공개와 공개 프로필을 구분하여 형태에 맞게 가입자 리스트를 구성한다.
6	'YU, wish A+'은 가입자 프로필 화면을 출력한다.
7	User는 출력된 가입자 프로필을 확인한다.
8	이 Use Case는 User가 출력된 가입자 프로필을 확인하거나 '뒤로가기' 버튼을 클릭하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
3	3a. 스마트폰의 '뒤로가기'버튼을 click하면 가입자 프로필 조회가 취소된다. ...3a1. 'YU, wish A+' 가입자 프로필 조회를 위한 작업을 중지한다. ...3a2. 'YU, wish A+' profileList screen을 닫고 main screen으로 이동한다.
4	4a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 데이터를 가져올 수 없을 시 ...4a1. 'YU, wish A+'은 Loading Prgress Indicator 화면을 띄우고 대기한다.

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #8 : 가입자 프로필 검색
GENERAL CHARACTERISTICS

Summary	User는 'YU, wish A+'를 이용하는 전체 사용자의 프로필을 조회할 때 조건을 설정하여 조회할 수 있다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	프로필을 조회하려는 User는 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 가입자 프로필 조회, profile list screen에서 검색을 기능을 제공하는 버튼을 click할 때
Success Post Condition	'YU, wish A+'은 이름, 학과 조건을 만족하는 가입자 프로필에 대해서만 리스트로 구성하여 사용자에게 출력한다. User는 가입자들의 프로필을 확인 할 수 있다. User가 리스트 프로필을 long click 하면 자세히 보여준다. User가 리스트 프로필을 click하면 소통하기 기능을 제공한다.
Failed Post Condition	User는 가입자 프로필 정보를 리스트 형식으로 볼 수 없다. 'YU, wish A+'은 Loading Progress Indicator 화면을 띄운다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 가입자 프로필 검색한다.
1	이 Use case 는 가입자 프로필을 조회할 때 조건을 적용할 때 시작된다.
2	User가 입자 프로필 검색을 제공하는 버튼을 클릭한다.
3	'YU, wish A+'는 조건을 적용할 수 있는 창을 제공한다.
4	User는 이름의 키워드 혹은 학과 선택하여 조건을 정하여 적용한다.
5	'YU, wish A+'은 넘겨 받은 검색 조건을 이용해 Google Firebase Database System 에서 가입자 정보를 가져온다.
6	'YU, wish A+'은 가져온 정보로 비공개와 공개 프로필을 구분하여 형태에 맞게 가입자 리스트를 구성한다.
7	'YU, wish A+'은 가입자 프로필 화면을 출력한다.
8	User는 출력된 가입자 프로필을 확인한다.
10	이 Use Case는 User가 출력된 가입자 프로필을 확인하거나 '뒤로가기' 버튼을 클릭하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
4	<p>4a. User가 검색 조건을 설정하지 않고 공백으로 적용할시</p> <p>...4a1. 'YU, wish A+'은 Google Firebase Database System에 접근하여 가입자들의 정보를 이름 오름차순으로 가져온다.</p> <p>...4a2. 가져온 데이터를 이용해 화면을 구성하는 단계로 간다. (Usecase ?-6)</p> <p>4b. User가 취소버튼을 누를 시</p> <p>...4b1. 'YU, wish A+'은 조건을 적용할 수 있는 창을 닫는다.</p>
5	<p>5a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 데이터를 가져올 수 없을 시</p> <p>...5a1. 'YU, wish A+'은 Loading Prgress Indicator 화면을 띄우고 대기한다.</p>
6	<p>6a. 스마트폰의 '뒤로가기'버튼을 click하면 가입자 프로필 조회가 취소된다.</p> <p>...6a1. 'YU, wish A+' 가입자 프로필 조회를 위한 작업을 중지한다.</p> <p>...6a2. 'YU, wish A+' profileList screen을 닫고 main screen으로 이동한다.</p>

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #9 : 게시글 열람
GENERAL CHARACTERISTICS

Summary	User는 다른 사용자들이 작성한 게시글을 리스트 형식으로 열람할 수 있다. 게시글은 제목, 작성자의 학과, email과 작성일을 노출한다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	게시글을 열람하는 User는 'YU, wish A+'에 회원 가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 게시글을 열람하기 위해 main screen에서 '게시글 보기' 버튼을 click 하였을 때
Success Post Condition	'YU, wish A+'은 게시글을 리스트로 구성하여 사용자에게 출력한다. User는 작성일 내림차순으로 정렬된 게시글들을 확인할 수 있다. User가 자신의 게시글을 long click 하면 수정할 수 있다. User가 다른 사람의 게시글을 long click 하면 소통하기 기능 사용할 수 있다.
Failed Post Condition	User는 게시글 정보를 리스트 형식으로 열람할 수 없다. 'YU, wish A+'은 Loading Prgress Indicator 화면을 띄운다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 게시글을 열람한다.
1	이 Use case 는 User가 게시글을 열람하기 위해 main screen에서 게시글 열람하기를 click할 경우 시작된다.
2	User가 게시글 열람하기 버튼을 click한다.
3	'YU, wish A+'은 Google Firebase Database System에 게시글 정보를 요청하여 작성일 내림차순으로 가져온다.
4	'YU, wish A+'은 가져온 게시글 정보를 이용하여 가입자 리스트를 구성하여 출력한다.
5	User는 출력된 게시글 리스트들을 확인한다.
6	이 Use Case는 User가 출력된 게시글 리스트를 확인하거나 '뒤로가기' 버튼을 클릭하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
3	3a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 데이터를 가져올 수 없을 시 ...3a1. 'YU, wish A+'은 Loading Progress Indicator 화면을 띄우고 대기한다.
4	4a. User가 '뒤로가기'버튼을 click할 시 ...4a1. 'YU, wish A+'은 게시글 열람을 하기 위한 작업을 중지한다. ...4a2. 'YU, wish A+'은 post list screen 을 닫고 main screen으로 이동한다.

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #10 : 게시글 작성
GENERAL CHARACTERISTICS

Summary	User는 게시글 열람 때 보여지는 게시글을 작성할 수 있다. 제목과 내용을 작성할 수 있다. 작성시 학과 정보와 작성일은 자동적으로 추가된다.
Scope	YU wish A+
Level	User level
Author	손광진
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	게시글을 작성하려는 User는 'YU, wish A+'에 회원 가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 게시글을 작성하기 위해 'YU, wish A+'의 post list screen에서 작성하기 버튼을 click할 때
Success Post Condition	작성한 게시글의 정보가 Google Firebase Database System에 추가된다. User가 리스트를 새로고침 할 경우, 작성한 게시글이 추가된 것을 확인 할 수 있다.
Failed Post Condition	네트워크 연결 상태 이상으로 게시글의 정보가 Google Firebase Database System에 추가되지 않는다. User가 리스트를 새로고침 하여도, 작성한 게시글이 추가된 것을 확인 할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 게시글을 작성한다.
1	이 Use case 는 User가 게시글을 작성하기 위해 게시글 작성 버튼을 click할 경우 시작된다.
2	User가 게시글을 작성하기 위해 게시글 작성 버튼을 click 한다.
3	'YU, wish A+'는 게시글 작성을 위한 창을 띄어준다.
4	User는 제목과 내용을 입력하고 작성 버튼을 click 한다.
5	'YU, wish A+'은 Google Firebase Database System에 접근하여 데이터를 추가를 요청한다.
6	'YU, wish A+'은 입력창을 닫고 post list screen을 출력한다.
7	User는 새로고침을 하여 등록된 게시글을 확인한다.
8	이 Use case는 User가 게시글 작성에 성공하거나 뒤로가기 버튼을 통해 취소하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
4	<p>4a. 뒤로가기'버튼을 click하여 게시글을 작성하지 않을 시 ...4a1. 'YU, wish A+'은 게시글 작성을 위한 창을 닫는다. ...4a2. 'YU, wish A+'은 post list screen으로 이동한다.</p> <p>4b. User가 제목과 내용을 공백으로 하고 작성버튼을 누를 시 ...4b. 1. "제목과 내용을 작성하라는 메시지를 출력한다" (Use case ?-4)</p>
5	<p>5a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 데이터를 추가하지 못할 시 ...5a1. User는 작성한 자신의 게시글을 확인할 수 없다.</p>

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #11-1 : 게시글 수정
GENERAL CHARACTERISTICS

Summary	User는 자신이 작성한 게시글을 수정할 수 있다. 제목과 내용을 수정할 수 있다. 작성일은 수정한 시간으로 변경된다.
Scope	YU wish A+
Level	User level
Author	손광진
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	프로필을 조회하려는 User는 'YU, wish A+'에 회원 가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 자신의 게시글을 long click할 때
Success Post Condition	수정한 정보로 Google Firebase Database System 데이터가 변경된다. User가 리스트를 새로고침 할 경우, 수정한 게시글이 추가된 것과 기존 게시물이 사라진 것을 확인할 수 있다.
Failed Post Condition	네트워크 연결 상태 이상으로 'Google Firebase Database System' 데이터가 변경되지 않는다. User가 리스트를 새로고침 하여도, 수정된 것을 확인할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 작성한 게시글을 수정한다.
1	이 Use case 는 User가 게시글을 수정하기 위해 post list screen에 출력된 자신의 게시글을 long click 할 경우 시작한다.
2	User가 자신의 게시글을 long click한다.
3	'YU, wish A+'은 수정을 위해 기존의 제목과 내용이 입력되어 있는 입력창을 제공한다.
4	User는 제목과 내용을 수정 후 수정 버튼을 click한다.
5	'YU, wish A+'은 Google Firebase Database System에 접근하여 데이터 수정을 요청한다.
6	'YU, wish A+'은 입력창을 닫고 post list screen을 출력한다.
7	User는 새로고침을 하여 수정된 게시글을 확인한다.
8	이 Use case는 User가 게시글 수정에 성공하거나 뒤로가기 버튼을 통해 취소하면 끝난다.

EXTENSION SCENARIOS	
Step	Branching Action
4	4a. 뒤로가기'버튼을 click하여 게시글을 작성하지 않을 시 ...4a1. 'YU, wish A+'은 게시글 작성을 위한 창을 닫는다. ...4a2. 'YU, wish A+'은 post list screen으로 이동한다.
	4b. User가 제목과 내용을 공백으로 하고 작성버튼을 누를 시 ...4b. 1. "제목과 내용을 작성하라는 메시지를 출력한다" (Use case ?-4)
5	5a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 데이터를 변경하지 못할 시 ...5a1. User는 변경된 자신의 게시글을 확인할 수 없다.

RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #11-2 : 게시물 삭제
GENERAL CHARACTERISTICS

Summary	User가 'YU, wish A+'의 게시판에 자신이 작성한 게시글을 삭제한다.
Scope	YU wish A+
Level	User level
Author	우재석
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 로그인한 상태여야하고 User가 작성한 게시글이 'YU, wish A+'의 게시판에 게시되어 있어야한다.
Trigger	User가 'YU, wish A+'의 게시판에서 자신의 게시글을 삭제하려고 할 때
Success Post Condition	'YU, wish A+'는 User가 삭제하려는 User의 게시글을 Google Firebase Database System에서 삭제하고 삭제된 게시글은 다른 User들이 조회할 수 없는 상태가 된다.
Failed Post Condition	User는 자신이 작성한 게시글을 삭제할 수 없는 상태가 된다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 자신이 작성한 게시글을 삭제한다.
1	이 Use case 는 User가 자신이 작성한 게시글을 삭제하려고 할 때 시작한다.
2	User는 작성한 게시글을 삭제하기 위해 'YU, wish A+'의 postlist screen으로 이동한다.
3	User postlist screen에 출력된 게시글 중 삭제하려는 자신의 게시글을 long click한다.
4	'YU, wish A+'는 User가 작성한 게시글인지 확인한다.
5	'YU, wish A+'의 User가 작성한 게시글인지 확인되면 post update delete screen을 출력한다.
6	User는 post update delete screen의 'delete' 아이콘을 click한다.
7	'YU, wish A+'는 Google Firebase Database System에 접근하여 게시글 데이터를 삭제한다.
8	'YU, wish A+' Google Firebase Database System에서 게시글을 삭제한 후, postlist screen으로 돌아온다.
9	'YU, wish A+'의 User들은 삭제된 게시글을 조회할 수 없게 된다.

10	이 Usecase는 User가 자신이 작성한 게시글을 삭제하면 끝난다.	
EXTENSION SCENARIOS		
Step	Branching Action	
5	5a. User가 작성한 게시글이 아니면 게시글 삭제에 실패한다. ...5a1. ‘YU, wish A+’는 사용자와 소통하기’ dialogue를 출력한다.	
7	7a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 게시글 삭제가 실패한다. ...7a1. ‘YU, wish A+’는 게시글 삭제를 취소하고 postlist screen으로 이동한다. (Use case #2-2) ...7a2. User는 자신이 작성한 게시글을 삭제할 수 없게 된다.	
RELATED INFORMATION		
Performance	≤ 2 seconds	
Frequency	제한 없음	
<Concurrency>	제한 없음	
Due Date		

Use case #12-1 : 조별 게시판 열람
GENERAL CHARACTERISTICS

Summary	User는 'YU, wish A+'의 조별 게시판을 열람할 수 있다.
Scope	YU wish A+
Level	User level
Author	최성찬
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다.
Trigger	teamlist screen의 tab bar에서 조별과제 tab을 click한 경우
Success Post Condition	Google Firebase Database System에 접근한다. Google Firebase Storage에 저장되어 있는 조별과제 목록들에 접근한다. 접근한 목록들을 게시글로 대외활동 게시판에 list한다.
Failed Post Condition	대외활동 목록들을 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 main screen에서 조별과제 게시판을 click한다.
1	이 Use case 는 User가 teamlist screen에서 조별과제 tab을 click할 때 시작된다.
2	조별과제 tab을 click하면 Google Firebase Database System에 접근한다. Google Firebase Storage에 저장되어 있는 조별과제 목록들에 접근한다.
3	접근한 조별과제 목록들을 게시글의 형태로 조별과제 게시판에 list한다.
4	이 Use case는 조별과제 게시판에 게시글들이 list되면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
2	2a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 게시글 속성을 가져오지 못할 경우 ...2a1. 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

RELATED INFORMATION	
Performance	≤ 5 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #12-2 : 대외활동 게시판 열람

GENERAL CHARACTERISTICS

Summary	User는 'YU, wish A+'의 대외활동 게시판을 열람할 수 있다.
Scope	YU wish A+
Level	User level
Author	최성찬
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다.
Trigger	teamlist screen의 tab bar에서 대외활동 tab을 click한 경우
Success Post Condition	Google Firebase Database System에 접근한다. Google Firebase Storage에 저장되어 있는 대외활동 목록들에 접근한다. 접근한 목록들을 게시글으로 대외활동 게시판에 list한다.
Failed Post Condition	대외활동 목록들을 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 main screen에서 조모임 게시판을 click한다.
1	이 Use case 는 User가 teamlist screen에서 대외활동 tab을 click할 때 시작된다.
2	대외활동 tab을 click하면 Google Firebase Database System에 접근한다. Google Firebase Storage에 저장되어 있는 대외활동 목록들에 접근한다.
3	접근한 대외활동 목록들을 게시글의 형태로 대외활동 게시판에 list한다.
4	이 Use case는 대외활동 게시판에 게시글들이 list되면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
2	2a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 게시글 속성을 가져오지 못할 경우 ...2a1. 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

RELATED INFORMATION

Performance	≤ 5 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #13 : 게시글 상세보기	
GENERAL CHARACTERISTICS	
Summary	User는 'YU, wish A+'의 조별과제 게시글 또는 대외활동 게시글을 상세보기 할 수 있다.
Scope	YU wish A+
Level	User level
Author	최성찬
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다. 대외활동 게시판 또는 조별과제 게시판에 게시글이 한 개 이상 작성되어 있어야 한다.
Trigger	teamlist screen의 대외활동 tab 또는 조별과제 tab에서 게시글을 click한 경우
Success Post Condition	Google Firebase Database System에 접근한다. Google Firebase Storage에 저장되어 있는 해당 게시글의 속성들에 접근한다. 접근한 속성들을 dialogue로 표시한다.
Failed Post Condition	게시글의 속성들을 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

MAIN SUCCESS SCENARIO	
Step	Action
S	User가 조모임 게시판에서 대외활동 tab 또는 조별과제 tab을 click한다.
1	이 Use case 는 User가 대외활동 tab 또는 조별과제 tab에서 게시글을 click할 때 시작된다.
2	게시글을 click하면 Google Firebase Database System에 접근한다. Google Firebase Storage에 저장되어 있는 해당 게시글의 속성에 접근한다.
3	접근한 속성들을 dialogue에 list한다.
4	이 Use case는 dialogue에 게시글의 속성이 list되면 끝난다.

EXTENSION SCENARIOS	
Step	Branching Action
2	2a. 네트워크 연결 상태 이상으로 Google Firebase Database System에서 게시글 속성을 가져오지 못할 경우 ...2a1. 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

RELATED INFORMATION

Performance	≤ 3 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #14 : 조모임 게시글 작성
GENERAL CHARACTERISTICS

Summary	User는 조모임 게시판 열람 시 보여지는 게시글을 작성할 수 있다. 제목, 전공, 프로젝트명, 인원 수, 타과 인원 허용여부, 등록 마감일, 프로젝트 마감일, 내용을 작성할 수 있다.
Scope	YU wish A+
Level	User level
Author	최성찬
Last Update	
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	게시글을 작성하려는 User는 'YU, wish A+'에 회원 가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 게시글 작성을 위해 'YU, wish A+'의 team list screen에서 작성 버튼을 click 할 때
Success Post Condition	작성한 게시글의 정보가 Google Firebase Database System에 추가된다. User가 리스트를 새로고침 할 경우, 작성한 게시글이 추가된 것을 확인 할 수 있다.
Failed Post Condition	네트워크 연결 상태 이상으로 게시글의 정보가 Google Firebase Database System에 추가되지 않는다. User가 리스트를 새로고침 하여도, 작성한 게시글이 추가된 것을 확인 할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 조모임 게시글을 작성한다.
1	이 Use case 는 User가 조모임 게시글을 작성하기 위해 게시글 작성 버튼을 click할 경우 시작된다.
2	User가 조모임 게시글을 작성하기 위해 작성 버튼을 click 한다.
3	'YU, wish A+'는 게시글 작성을 위한 창을 띄워준다.
4	User는 제목, 전공, 프로젝트명, 인원 수, 타과 인원 허용여부, 등록 마감일, 프로젝트 마감일, 내용을 입력하고 작성 버튼을 click 한다.
5	'YU, wish A+'은 Google Firebase Database System에 접근하여 데이터를 추가를 요청한다.
6	'YU, wish A+'은 입력창을 닫고 team list screen을 출력한다.
7	User는 새로고침을 하여 등록된 조모임 게시글을 확인한다.
8	이 Use case는 User가 조모임 게시글 작성에 성공하거나 뒤로가기 버튼을 통해 게시글 작성을 취소하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
4	4a. 뒤로가기'버튼을 click하여 조모임 게시글을 작성하지 않을 시 ...4a1. 'YU, wish A+'은 게시글 작성 창을 닫는다. ...4a2. 'YU, wish A+'은 team list screen으로 이동한다.
	4b. User가 한 개 이상의 항목을 공백으로 하고 작성버튼을 누를 시 ...4b.1. "빈칸에 내용을 작성하라는 메시지를 출력한다"
5	5a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 데이터를 추가하지 못할 시 ...5a1. User는 작성한 자신의 게시글을 확인할 수 없다.

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #15 : 조모임 게시물 수정
GENERAL CHARACTERISTICS

Summary	User는 조모임 게시판 열람 시 보여지는 게시글을 수정할 수 있다. 제목, 전공, 프로젝트명, 인원 수, 타과 인원 허용여부, 등록 마감일, 프로젝트 마감일, 내용을 수정할 수 있다.
Scope	YU wish A+
Level	User level
Author	최성찬
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	조모임 게시글을 수정하려는 User는 'YU, wish A+'에 회원 가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다.
Trigger	User가 조모임 게시글 수정을 위해 'YU, wish A+'의 team list screen에서 게시글을 click 하여 나타난 detail screen에서 수정 버튼을 click 할 때
Success Post Condition	수정한 게시글의 정보가 Google Firebase Database System에 업데이트된다.
Failed Post Condition	네트워크 연결 상태 이상으로 게시글의 정보가 Google Firebase Database System에 업데이트되지 않는다. User가 리스트를 새로고침 하여 해당 게시글의 detail screen에 진입해도, 작성한 게시글이 수정된 것을 확인 할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 조모임 게시글을 수정한다.
1	이 Use case 는 User가 조모임 게시글을 수정하기 위해 수정 버튼을 click할 경우 시작된다.
2	User가 조모임 게시글의 상세내용을 조회하기 위해 수정할 조모임 게시글을 click 한다.
3	User가 조모임 게시글을 수정하기 위해 수정 버튼을 click 한다.
4	'YU, wish A+'는 게시글 수정을 위한 창을 띄워준다.
5	User는 제목, 전공, 프로젝트명, 인원 수, 타과 인원 허용여부, 등록 마감일, 프로젝트 마감일, 내용을 입력하고 수정 버튼을 click 한다.
6	'YU, wish A+'은 Google Firebase Database System에 접근하여 데이터 업데이트를 요청한다.
7	'YU, wish A+'은 입력창을 닫고 detail screen을 출력한다.
8	User는 team list screen으로 돌아가 새로고침을 하여 등록된 조모임 게시글을 확인한다.
9	이 Use case는 User가 조모임 게시글 수정에 성공하거나 뒤로가기 버튼을 통해 게시글 수정을 취소하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
4	<p>4a. 뒤로가기'버튼을 click하여 조모임 게시글을 수정하지 않을 시 ...4a1. 'YU, wish A+'은 게시글 수정 창을 닫는다. ...4a2. 'YU, wish A+'은 detail screen으로 이동한다.</p> <p>4b. User가 한 개 이상의 항목을 공백으로 하고 작성버튼을 누를 시 ...4b. 1. "빈칸에 내용을 작성하라는 메시지를 출력한다"</p>
5	<p>5a. 네트워크 연결 상태 이상으로 Google Firebase Database System에 데이터를 추가하지 못할 시 ...5a1. User는 수정하여 작성된 자신의 게시글을 확인할 수 없다.</p>

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #16 : 조모임 게시물 삭제
GENERAL CHARACTERISTICS

Summary	User는 조모임 게시판 열람 시 보여지는 게시물 중 자신이 작성한 조모임 게시물을 삭제할 수 있다.
Scope	YU wish A+
Level	User level
Author	최성찬
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	조모임 게시물을 수정하려는 User는 'YU, wish A+'에 회원 가입과 프로필 작성이 완료된 상태여야 한다. 정상적인 네트워크 이용이 가능해야 한다. 삭제할 대상의 조모임 게시물의 작성자가 자신이어야 한다.
Trigger	User가 조모임 게시물 삭제를 위해 'YU, wish A+'의 team list screen에서 게시물을 long-click 하거나, 게시물을 click 하여 나타난 detail screen에서 삭제 버튼을 click 할 때
Success Post Condition	'YU, wish A+'는 User가 삭제하려는 조모임 게시물을 Google Firebase Database System에서 삭제하여 다른 User들이 조회할 수 없는 상태가 된다.
Failed Post Condition	네트워크 연결 상태 이상으로 게시물의 정보가 Google Firebase Database System에서 삭제되지 않는다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 조모임 게시물을 삭제한다.
1	이 Use case 는 User가 조모임 게시물을 삭제하기 위해 삭제 버튼을 click할 경우 시작된다.
2	User가 조모임 게시물의 상세내용을 삭제하기 위해 수정할 조모임 게시물을 click 한다.
3	User가 조모임 게시물을 삭제하기 위해 삭제 버튼을 click 한다.
4	'YU, wish A+'는 게시물 삭제를 경고하는 알림을 띄워준다.
5	User는 알림에서 삭제를 click 한다.
6	'YU, wish A+'은 Google Firebase Database System에 접근하여 해당 게시물의 데이터를 삭제한다.
7	'YU, wish A+'은 입력창을 닫고 team list screen을 출력한다.
8	User는 team list screen으로 돌아가 새로고침을 하여 조모임 게시물이 삭제되어 나타나지 않는지 확인한다.
9	이 Use case는 User가 조모임 게시물 삭제에 성공하거나 취소 버튼을 통해 게시물 삭제를 취소하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
1	1a. team list screen의 삭제할 조모임 게시글을 long-click 할 시 ...1a1. 'YU, wish A+'는 게시글 삭제를 경고하는 알림을 띄워준다.
4	4a. 뒤로가기'버튼을 click하여 조모임 게시글을 삭제하지 않을 시 ...4a1. 'YU, wish A+'은 게시글 삭제 알림을 닫는다. ...4a2. 'YU, wish A+'은 detail screen으로 이동한다. 4b. User가 한 개 이상의 항목을 공백으로 하고 작성버튼을 누를 시 ...4b.1. "빈칸에 내용을 작성하라는 메시지를 출력한다"
5	5a. 네트워크 연결 상태 이상으로 Google Firebase Database System의 데이터를 삭제하지 못할 시 ...5a1. User는 team list screen으로 이동하지만, 자신의 게시글을 자신이나 타인이 확인할 수 있다.

RELATED INFORMATION

Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #17 : 쪽지 보내기
GENERAL CHARACTERISTICS

Summary	어플리케이션을 사용하는 User와 User 사이에 쪽지를 보낼 수 있다.
Scope	YU wish A+
Level	User level
Author	홍덕화
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	쪽지를 보내려는 User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다 .
Trigger	'YU, wish A+'에서 주소록 목록, 받은 쪽지함 목록, 보낸 쪽지함 목록, 조모임 게시판 목록을 click한 경우와 게시판 목록을 long click한 경우
Success Post Condition	6. 쪽지를 보내려고 한 User는 보낸 쪽지함에 쪽지가 추가된다. 7. 쪽지를 받는 User는 받은 쪽지함에 해당하는 쪽지가 추가된다. 8. 쪽지 전송이 완료되었다는 다이얼로그를 표시한다.
Failed Post Condition	9. 쪽지를 보내려고 한 User의 보낸 쪽지함에 쪽지가 추가되지 않는다. 10. 쪽지를 받아야 하는 User의 받은 쪽지함에 쪽지가 추가되지 않는다. 11. 쪽지를 보내려고 한 User의 화면에 쪽지 전송이 실패했다는 다이얼로그를 출력한다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 'YU, wish A+'의 main screen'에 있는 게시글 보기, 조모임, 쪽지함, 주소록 중 한 곳을 들어간다.
1	이 Use case는 User가 해당 게시판의 게시글을 click한 경우 시작한다.
2	User는 다른 User와 소통하기 다이얼로그에서 종이비행기 모양 아이콘을 탭한다.
3	쪽지의 title과 content를 입력하고 send 버튼을 누르면 click한 게시글의 User에게 쪽지가 보내진다.
4	쪽지 전송이 완료되면 쪽지 전송이 완료되었다는 dialogue가 나타나고 Use Case는 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
------	------------------

3

3a. 제목이나 내용이 비어있어서 쪽지 전송에 실패한다.
 ...3a1. 쪽지 전송에 실패했다는 dialogue를 보여준다 .
 ...3a2. 확인 버튼을 누르면 해당 게시판의 screen으로 돌아간다.

RELATED INFORMATION

Performance	≤ 1 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #18 : 쪽지함 확인하기
GENERAL CHARACTERISTICS

Summary	User는 자신의 쪽지함을 확인할 수 있다.
Scope	YU wish A+
Level	User level
Author	홍덕화
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	쪽지를 보내려는 User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다 .
Trigger	'YU wish A+'에서 쪽지함의 받은 쪽지함과 보낸 쪽지함 Tab을 click할 때
Success Post Condition	받은 쪽지함에는 자신이 받는 User인 쪽지의 목록들이 list된다. 보낸 쪽지함에는 자신이 보낸 User인 쪽지의 목록들이 list된다.
Failed Post Condition	쪽지 목록들을 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 Error 표시문이 출력된다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 'YU, wish A+'의 main screen에 들어간다.
1	이 Use case는 User가 쪽지함을 click할 때 시작된다.
2	User는 처음 들어갈 때 표시되는 받은 쪽지함과 보낸 쪽지함을 Tabbar를 이용해서 열람할 수 있다.
3	이 Use case는 쪽지들이 list되면 끝난다 .

EXTENSION SCENARIOS

Step	Branching Action
3	1a. 쪽지 목록들을 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못한다1a1. Error : 'error 발생 이유'를 출력한다 .

RELATED INFORMATION

Performance	≤ 3 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #19 : 전화하기

GENERAL CHARACTERISTICS

Summary	User가 다른 User에게 전화할 수 있다.
Scope	YU wish A+
Level	User level
Author	홍덕화/박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	이메일을 보내려는 User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다.
Trigger	'사용자와 소통하기' dialogue에서 전화 아이콘을 click한 경우
Success Post Condition	전화하고자 하는 User의 스마트폰의 Default Calling Application으로 화면전환을 한다.
Failed Post Condition	전화를 받을 대상 User의 전화번호 정보를 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 네트워크가 안정될 때 까지 Loading Prgress Indicator가 돌아간다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 '사용자와 소통하기' dialogue와 상호작용한다.
1	이 Use case는 User가 '사용자와 소통하기' dialogue에서 전화 아이콘을 click할 때 시작된다.
2	전화 아이콘을 click하면 화면이 전환되면서 Default Calling Application이 실행된다.
3	Default Calling Application에서 전화번호 입력란에 상대 User의 전화번호가 설정된다.
4	User는 Default Calling Application의 전화 버튼을 눌러 전화를 한다.
5	이 Use case 는 전화를 하면 끝난다 .

EXTENSION SCENARIOS

Step	Branching Action
2	2a. User의 전화번호 정보를 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못한다. ...2a1. 네트워크가 안정될 때 까지 Loading Progress Indicator가 돌아간다.

RELATED INFORMATION	
Performance	≤ 3 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #20 : 이메일 보내기
GENERAL CHARACTERISTICS

Summary	User가 다른 User에게 이메일을 보낼 수 있다.
Scope	YU wish A+
Level	User level
Author	홍덕화/박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	이메일을 보내려는 User가 'YU, wish A+'에 회원가입과 프로필 작성이 완료된 상태여야 한다.
Trigger	'사용자와 소통하기' dialogue에서 편지 아이콘을 click한 경우
Success Post Condition	이메일을 보내고자 하는 User의 Default E-mail Application으로 화면전환을 한다. 'YU, wish A+'에서 연락드려요.'로 제목이 설정된다. 받는 사람에는 받는 User의 email 주소가 설정된다.
Failed Post Condition	Default E-mail Application이 없는 경우 오류를 snackbar로 표시한다. 대상 User의 메일 주소 정보를 네트워크 연결 상태 이상으로 Google Firebase Database System에서 가져오지 못 할 경우 네트워크가 안정될 때 까지 Loading Prgress Indicator가 돌아간다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 '사용자와 소통하기' dialogue와 상호작용한다.
1	이 Use case는 User가 '사용자와 소통하기' dialogue에서 편지 아이콘을 click할 때 시작 된다.
2	편지 아이콘을 click하면 화면이 전환되면서 Default E-mail Application이 실행된다.
3	Default E-mail Application에서 제목과 주소가 설정된다. 사용자는 이메일을 완성하여 전송 버튼을 누른다.
4	이 Use case 는 이메일이 전송되면 끝난다 .

EXTENSION SCENARIOS

Step	Branching Action
2	2a. User의 스마트폰에 Default E-mail Application이 없어서 Default E-mail Applicatio n을 실행할 수 없다. ...2a1. 오류 메시지를 snackbar로 표시한다. 2b. User의 메일 주소 정보를 네트워크 연결 상태 이상으로 Google Firebase Database Sy stem에서 가져오지 못한다. ...2b1. 네트워크가 안정될 때 까지 Loading Prgress Indicator가 돌아간다.

RELATED INFORMATION

Performance	≤ 10 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #21 : 개발자 보기
GENERAL CHARACTERISTICS

Summary	User는 'YU wish A+'의 개발자들을 볼 수 있다.
Scope	YU wish A+
Level	User level
Author	박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	'YU wish A+'를 실행한 상태여야 한다.
Trigger	welcome screen의 '제작자' 글씨를 click한 경우
Success Post Condition	programmer screen으로 화면이 전환된다. 제작자들의 이름과 한줄 소개가 list된다.
Failed Post Condition	없음

MAIN SUCCESS SCENARIO

Step	Action
S	User가 'YU, wish A+'를 실행한다.
1	이 Use case 는 User가 welcome screen에 진입했을 때 시작한다.
2	User는 welcome screen에서 '제작자'라는 글씨를 click한다.
3	'제작자'가 click되면 programmer screen으로 화면을 전환한다.
4	이 Use case 는 programmer screen이 출력되면 끝난다 .

RELATED INFORMATION

Performance	≤ 1 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #22 : 오픈소스 보기
GENERAL CHARACTERISTICS

Summary	User는 'YU wish A+' 에 쓰인 open source 목록들을 볼 수 있다.
Scope	YU wish A+
Level	User level
Author	박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User가 welcomescreen에 있어야 한다.
Trigger	welcome screen의 'openSource' 글씨를 click한 경우
Success Post Condition	'YU, wish A+'는 Google Firebase Authentication System에 접근해 현재 로그인 중인 계정을 로그아웃시키고 welcom screen으로 이동하고, User는 'YU, wish A+'에 로그인하기 전 상태가 된다.
Failed Post Condition	opensource screen으로 화면이 전환된다. 'YU, wish A+'를 제작하는 데에 사용된 open source 목록들이 각각의 설명과 함께 list된다. open source 목록을 click하면 해당 open source의 공식 document로 이동한다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 'YU, wish A+'를 실행한다.
1	이 Use case 는 User가 welcome screen에 진입했을 때 시작한다.
2	User는 welcome screen에서 'openSource'라는 글씨를 click한다.
3	'openSource'가 click되면 opensource screen으로 화면을 전환한다.
4	opensource screen에 list된 open source의 목록 중 하나를 click한다.
5	화면 전환이 이루어진다. Default Web Application에 의해 open source에 해당하는 공식 document로 이동한다.
6	이 Use case는 공식 document로 이동하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
------	------------------

5	5a. 네트워크 연결이 불안정하여 공식 document로의 이동에 실패한다. ...5a1. Default Web Application이 인터넷 연결 없음을 메시지로 표시한다.
---	--

RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	제한 없음
<Concurrency>	제한 없음
Due Date	

Use case #23 : 로그아웃
GENERAL CHARACTERISTICS

Summary	User가 'YU, wish A+' 사용을 중단하거나 다른 계정으로 로그인할 수 있게 한다.
Scope	YU wish A+
Level	User level
Author	우재석/박재웅
Last Update	2020-11-06
Status	Analysis (Finalize)
Primary Actor	User
Preconditions	User는 'YU, wish A+'에 로그인하여 서비스를 이용하고 있어야한다.
Trigger	User가 'YU, wish A+'의 profile screen에 '로그아웃'버튼을 click하여 로그아웃하려고 할 때
Success Post Condition	'YU, wish A+'는 Google Firebase Authentication System에 접근해 현재 로그인 중인 계정을 로그아웃시키고 welcom screen으로 이동하고, User는 'YU, wish A+'에 로그인하기 전 상태가 된다.
Failed Post Condition	User는 'YU, wish A+'에 로그아웃을 실패하고 다른 계정으로 로그인할 수 없다.

MAIN SUCCESS SCENARIO

Step	Action
S	User가 'YU, wish A+'에 로그아웃을 한다.
1	이 Use case 는 User가 'YU, wish A+'가 제공하는 서비스를 이용을 중단하거나 다른 계정으로 로그인하려고 할 때 시작한다.
2	User는 로그아웃하기 위해 'YU, wish A+'의 profile screen에 '로그아웃'버튼을 click한다.
3	'YU, wish A+'는 현재 로그인되어있는 계정을 확인한다.
4	'YU, wish A+'는 Google Firebase Authentication System에 접근하여 확인된 계정을 로그아웃 시킨다.
5	'YU, wish A+'는 welcom screen으로 이동한다.
6	User는 'YU, wish A+'에 로그아웃하여 'YU, wish A+' 사용을 중단하고 다른 계정으로 로그인할 수 있다.
7	이 Usecase는 User가 'YU, wish A+'에 로그아웃을 성공하면 끝난다.

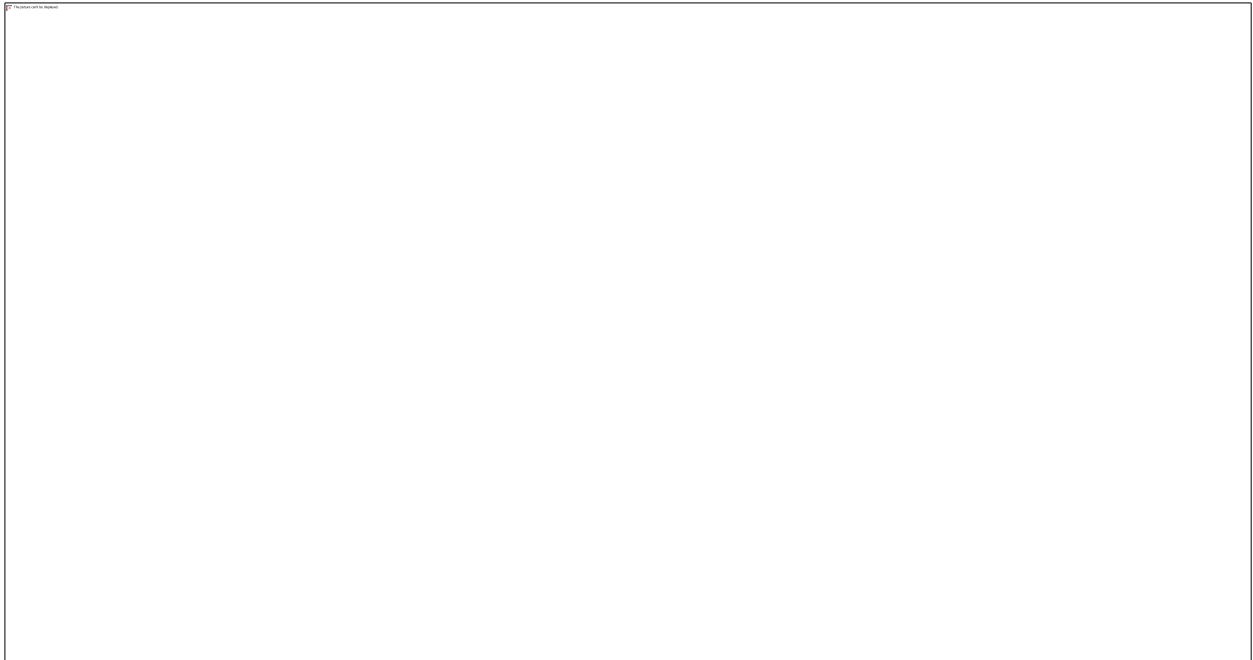
RELATED INFORMATION

Performance	≤ 10 seconds
Frequency	제한 없음

<Concurrency>	제한 없음
Due Date	

3. Class diagram

전체 클래스 다이어그램

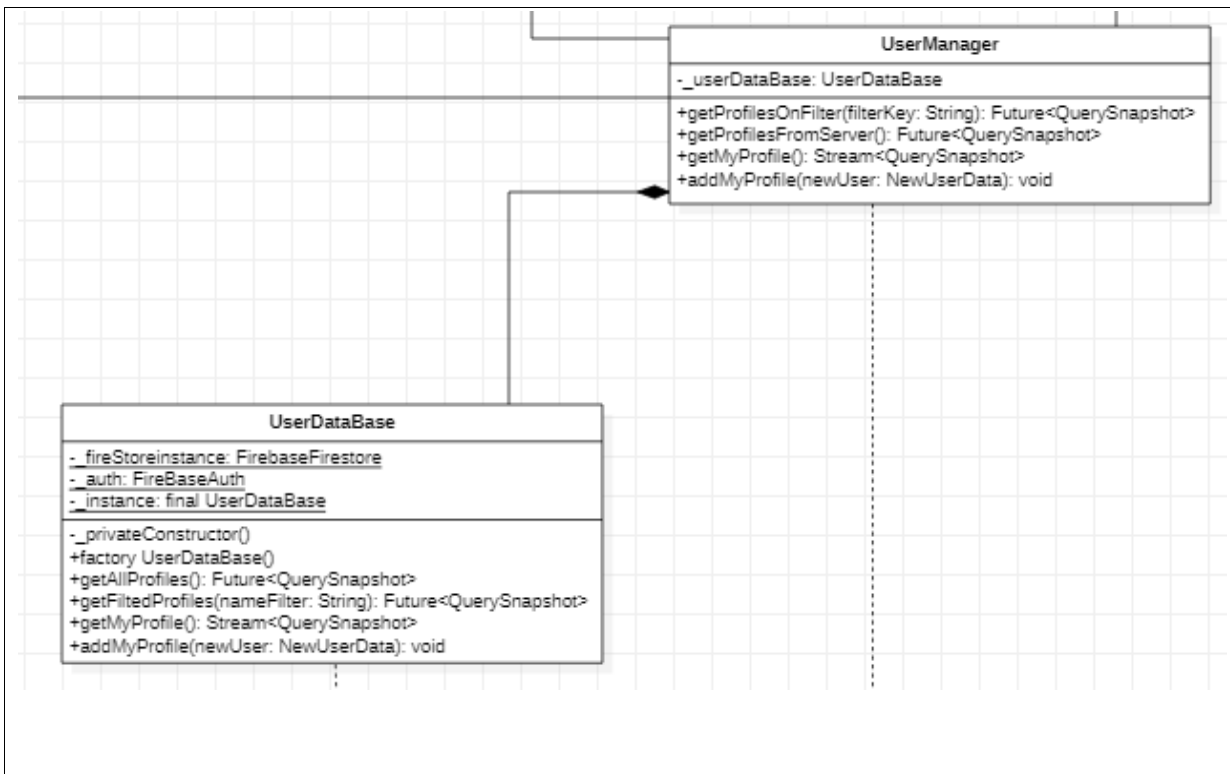


<YU, wish A+의 클래스 다이어그램>

위 그림은 'YU, wish A+' 어플리케이션의 가장 핵심적인 Class들을 다룬 Class Diagram이다. UI를 구성하는 Widget 클래스들과 Logic 클래스들이 어떻게 관계를 맺고 있는지 보여준다.

각 Class에서 사용하는 변수들과 메소드들을 다루는 모습을 확인할 수 있다. 자세한 설명은 아래 내용을 통해 확인할 수 있다.

UserManager, UserDataBase

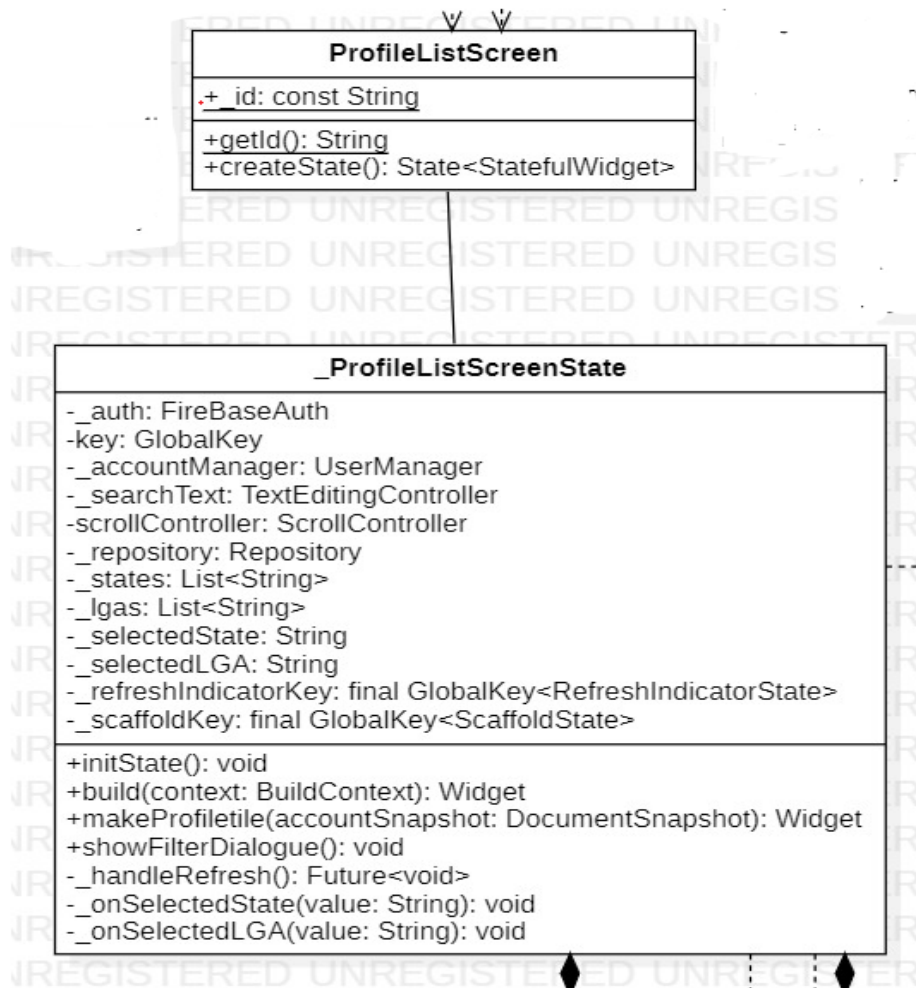


변수 및 메소드

- `_fileStoreInstance` : Google Firebase Database System에 데이터 조작을 요청하는 인스턴스이다.
- `_auth` : 로그인한 유저 정보를 나타내는 인스턴스이다.
- `getAllProfiles()` : Google Firebase Database System에서 모든 사용자에게 대한 정보를 이름 오름차순으로 가져온다.
- `getFiltedProfiles(nameFilter)` : 이름이 `nameFilter`로 시작하거나 끝나는 사용자 프로필만 가져온다.
- `getMyProfile()` : 나의 프로필 정보를 가져온다.
- `addMyProfile(newUser)` : Google Firebase Database System에 나의 프로필 정보를 추가한다.

기능 및 설명	<ul style="list-style-type: none"> · UserDataBase 클래스는 Google Firebase Database System에 게 요청하여 사용자 정보를 추가, 조회한다. · UserManager 클래스는 다른 클래스들이 UserDataBase를 직접 사용하지 않게 bridge역할을 하는 클래스이다. UserDataBase 는 Google Firebase Database System에 추가, 갱신하는 기본적 인 작업만 하기 때문에 추가적인 작업은 UserManager에서 처 리 후 UserDataBase를 이용한다.
---------	--

ProfileListScreen, _ProfileListScreenState



변수 및 메소드

- `_auth` : 로그인한 사용자의 정보를 가지는 인스턴스
- `_accountManager` : Google Firebase Database System으로부터 데이터를 전체 사용자 프로필을 가져오거나, 조건에 맞는 사용자 프로필을 가져올 때 사용한다.
- `_repository`, `_states`, `_lgas`, `_selectedState`, `_selectedLGA`: 조건에 맞는 사용자 프로필을 가져올 때, 학과 조건을 설정할 때 DropDown Box를 구성하기 위해 사용된다.
- `makeProfiletile(accountSnapshot)`: Google Firebase Database System으로부터 받아온 사용자 프로필 정보를 이용하여 `ListItem`을 구성하는 함수이다.
- `showFilterDialogue()`; 검색 조건을 정하기 위한 UI제공

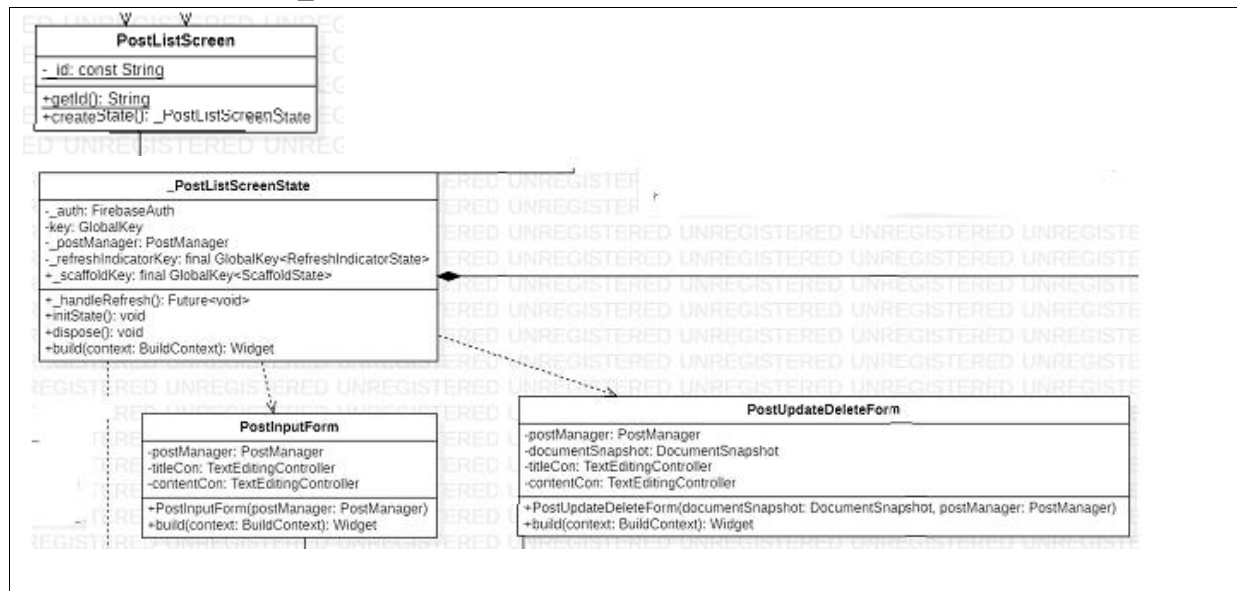
기능 및 설명

- `ProfileListScreen`은 `_ProfileListScreenState`를 상태로 가진

다. `_ProfileListScreenState`에서 상태 변화나 특정 함수를 사용시 `_ProfileListScreenState`를 다시 형성하며 UI가 바뀌게 된다.

· `_ProfileListScreenState` 클래스는 Google Firebase Database System으로부터 받아온 사용자 프로필을 출력한다. `_accountManager`의 메소드를 사용하여 사용자 프로필 정보를 받아온다. 검색 필터 다이얼로그를 지원하며, 최상단에서 down scroll 시 Google Firebase Database System으로부터 사용자 프로필 정보를 다시 가져와 화면을 구성한다.

PostListScreen, _PostListScreenState, PostInputForm, PostUpdateDeleteForm



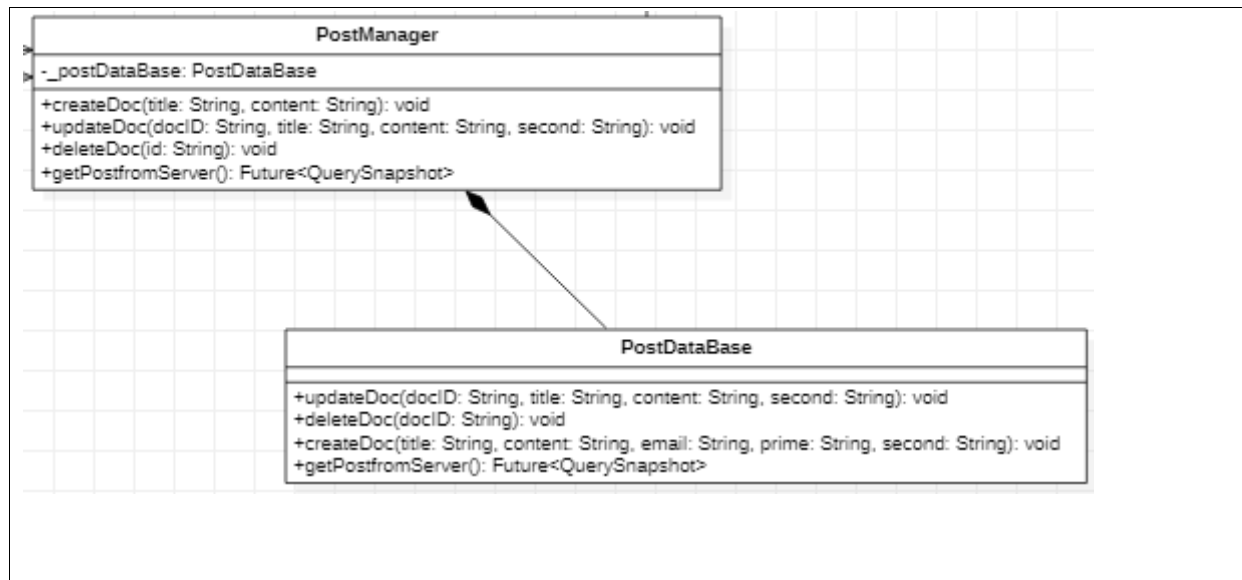
변수 및 메소드

- `_postManager` : 게시글을 조회, 추가, 갱신, 삭제를 할 수 있는 클래스이다.
- `titleCon` : 제목 텍스트 필드의 컨트롤러이다. 필드에 입력된 문자열을 가져온다.
- `contentCon`:내용 텍스트 필드의 컨트롤러이다.
- `documentSnapshot` : 수정, 삭제하려던 게시글의 정보를 가지고 있다.

기능 및 설명

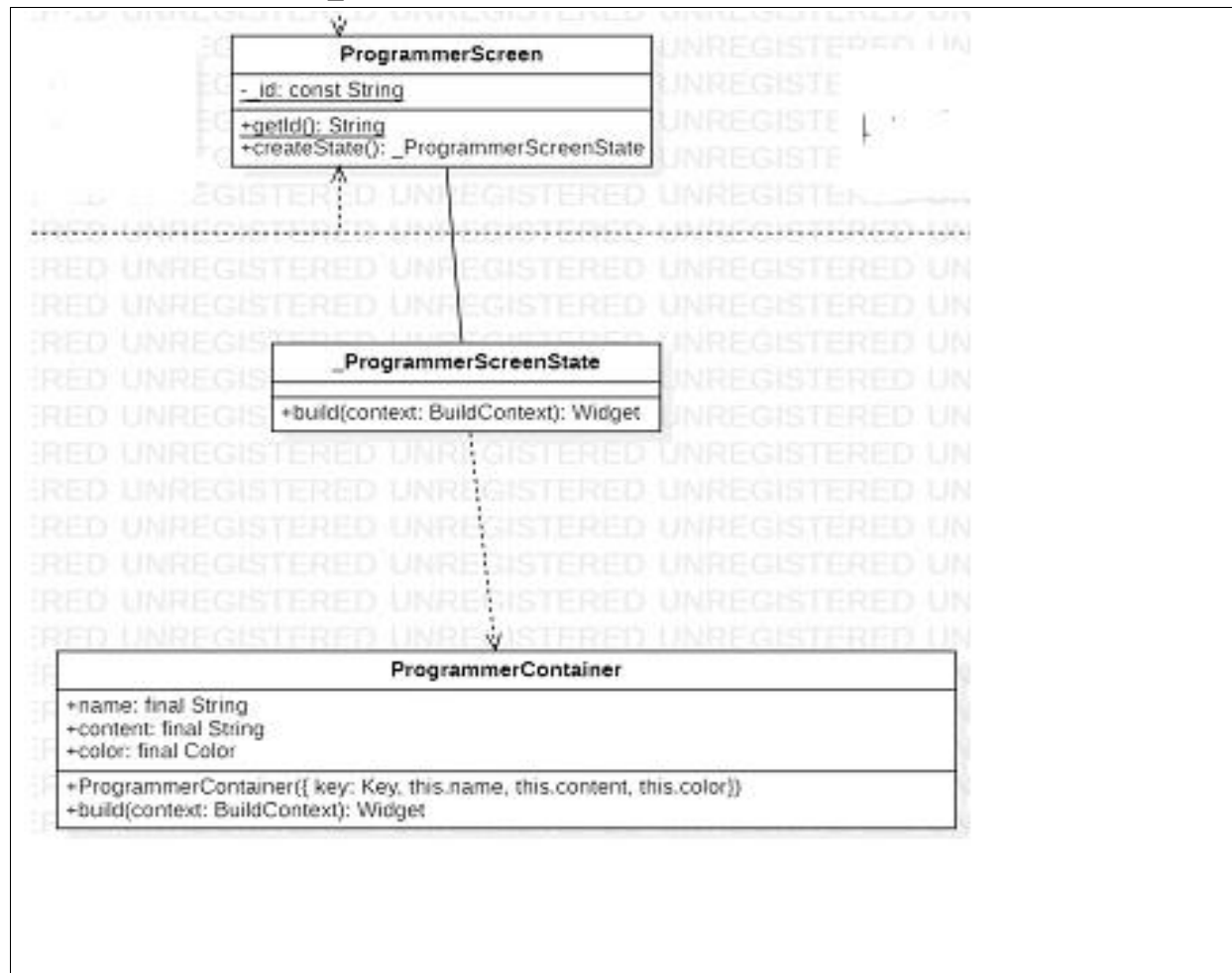
- `_PostListScreenState` 클래스는 `_postManager`를 이용하여 게시물을 조회, 추가, 갱신, 삭제하는 클래스이다. 최상단에서 down scroll시 업데이트가 된 게시물들로 refresh된다.
- `PostInputForm` : 게시글을 추가할 때, UI를 제공하는 클래스이다.
- `PostUpdataDeleteForm` : 자신의 게시글을 수정하거나 삭제할 때 UI를 제공하는 클래스이다. 원래 게시글의 제목과 내용을 채워 넣은 후 사용자에게 제공된다.

PostManager, PostDataBase



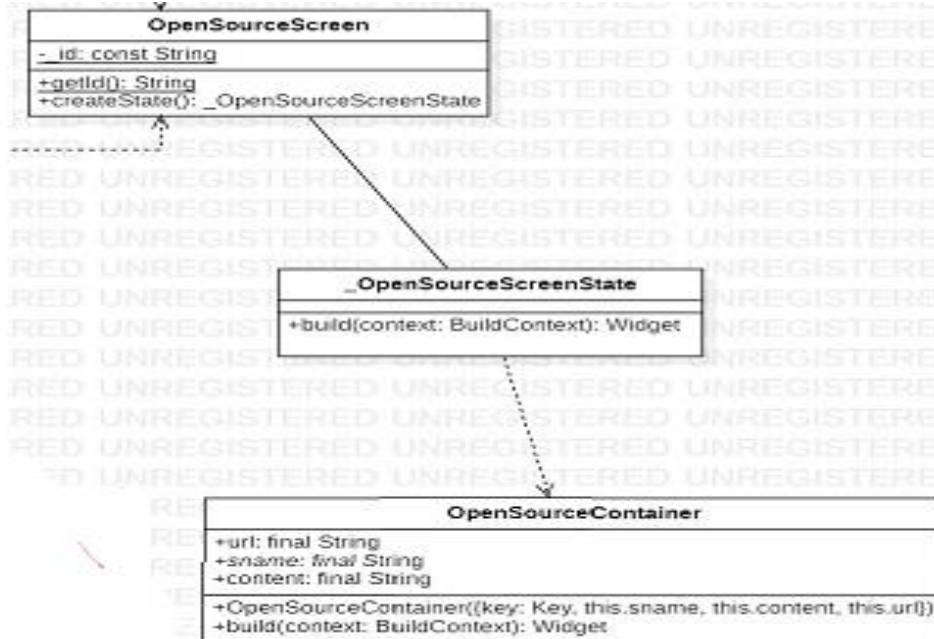
변수 및 메소드	<ul style="list-style-type: none"> · updateDoc: docID와 같은 ID를 가지는 도큐먼트의 업데이트를 Google Firebase Database System에 요청한다. · deleteDoc: docID와 같은 ID를 가지는 도큐먼트의 삭제를 Google Firebase Database System에 요청한다. · createDoc: 매개변수로 받은 정보를 이용하여 게시글 생성을 Google Firebase Database System에 요청한다. · getPostfromServer: Google Firebase Database System에게 전체 게시글들의 정보를 요청한다.
기능 및 설명	<ul style="list-style-type: none"> · PostDataBase 클래스는 게시글을 추가, 생성, 갱신, 삭제를 Google Firebase Database System에 요청하는 클래스이다. · PostManager 클래스는 PostDataBase를 이용하여 다른 객체들에게 게시글을 조작하는 bridge 역할을 한다. PostDataBase는 단순한 기능만 하기 때문에, 추가적인 기능을 보조해주는 작업을 한다.

ProgrammerScreen, _ProgrammerScreenState, ProgrammerContainer



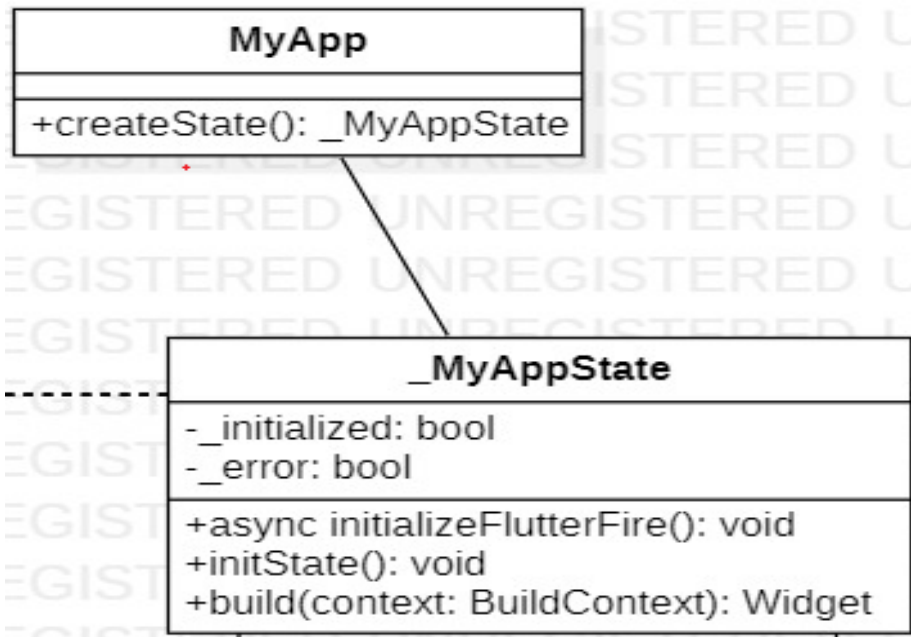
변수 및 메소드	<ul style="list-style-type: none"> · name:개발자 이름 · content:개발자의 한마디 · color:ListItem의 색깔을 결정하는 변수
기능 및 설명	<ul style="list-style-type: none"> · ProgrammerScreen 클래스는 StatefulWidget이다. _ProgrammerScreenState라는 상태를 가진다. · ProgrammerScreenState 클래스는 개발자 명단을 출력하는 widget이다. List 형식으로 UI를 구성한다. ListItem을 구현하기위해 ProgrammerContainer를 생성할 때 개발자에 관련된 매개변수를 주어 ListItem을 만든다.

OpenSourceScreen, _OpenSourceScreenState, OpenSourceContainer



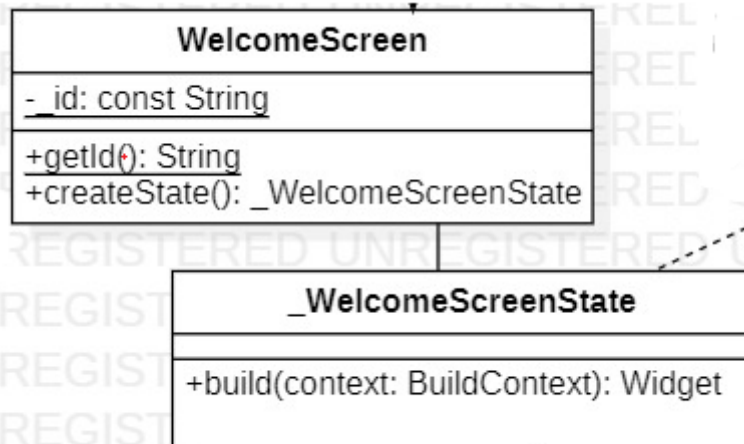
변수 및 메소드	<ul style="list-style-type: none"> · url: 오픈 소스의 url · sname: 오픈소스의 이름 · content: 오픈 소스 설명
기능 및 설명	<ul style="list-style-type: none"> · OpenSourceScreen 클래스는 StatefulWidget이다. _OpenSourceScreenState의 상태를 가진다. · OpenSourceScreenState : OpenSourceContainer를 이용하여 List형식으로 오픈 소스 정보를 나열한다. · OpenSourceContainer클래스는 ListItem을 구성하는데에 사용되며, 생성시 받은 url 정보를 이용하여 사용자가 click할 시 해당 url을 연다.

MyApp, _MyAppState



변수 및 메소드	<ul style="list-style-type: none"> · <code>_initialized</code> : Google Firebase Database System을 사용하기 위한 초기화 작업이 정상적으로 수행되면 True이다. · <code>_error</code>:Google Firebase Database System을 사용하기 위한 초기화 작업에 오류가 발생하면 True이다. · <code>async initializeFlutterFire</code> : Google Firebase Database System을 사용하기 위한 초기화 작업을 수행하며, 결과에 따라 <code>_initialized</code> 변수와 <code>_error</code>의 값을 설정한다. · <code>initState</code> : 클래스 생성 시 가장 먼저 실행되며 <code>initialize FulutterFire</code> 메소드를 실행한다. · <code>build</code>
기능 및 설명	<ul style="list-style-type: none"> · <code>MyApp</code> 클래스는 프로그램 실행 시 가장 먼저 실행되는 클래스이다. <code>StatefulWidget</code>이며 <code>_MyAppstate</code> 상태를 가진다. · <code>_MyAppState</code> 클래스는 Google Firebase Database System을 사용하기 위한 초기화 작업을 수행한다. 또한 기본 테마를 설정하며, 페이지 이동을 위한 <code>routes</code>를 설정한다.

WelcomeScreen, _WelcomeScreenState



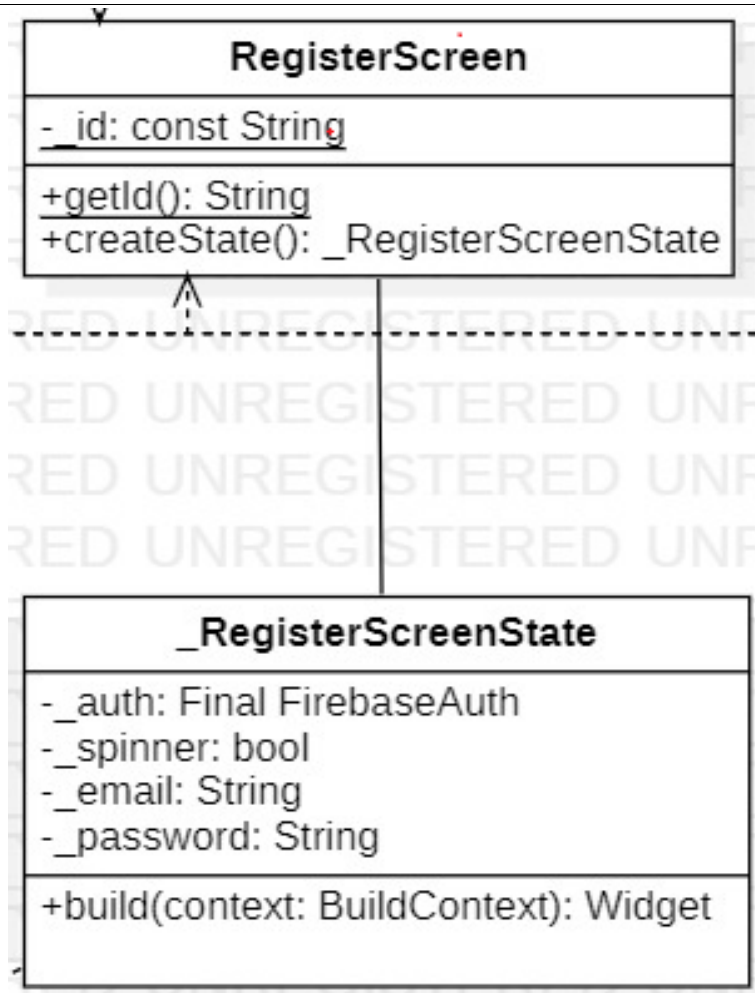
변수 및 메소드

- `_id : String` 형 `id` 변수를 갖고 있는 이유는 페이지 전환을 위한 페이지 식별 `id`를 제공하기 위해서이다.
- `getId()` : `private`으로 선언된 `id`를 반환해주기 위해 필요한 함수이다.
- `createState()` : 상태를 생성하여 UI를 구성하기 위한 함수이다.
- `build`: UI를 구성하며 이벤트 등을 설정한다.

기능 및 설명

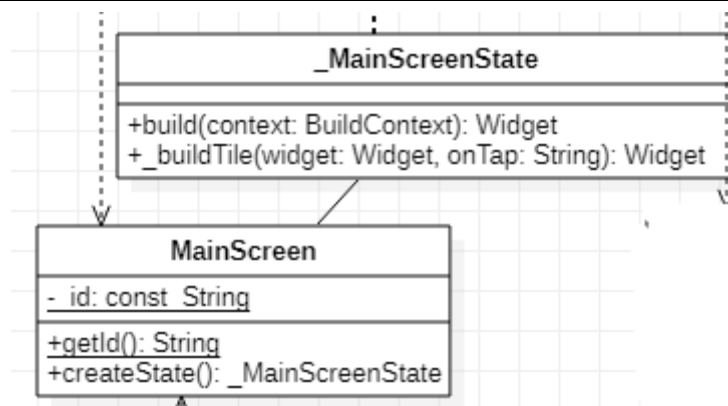
- `WelcomeScreen`은 `StatefulWidget`이다. `_WelcomeScreenState` 상태를 가진다.
- `_WelcomeScreenState` 클래스는 로그인과 회원 가입으로 이동하기 위한 UI이다.

RegisterScreen, _RegisterScreenState



변수 및 메소드	<ul style="list-style-type: none"> · email: 사용자가 입력한 ID를 저장한다. · password: 사용자가 입력한 비밀번호를 저장한다. · spinner: 아이디 패스워드 생성하는 동안 띄어주는 대기 창
기능 및 설명	<ul style="list-style-type: none"> · RegisterScreen 클래스는 StatefulWidget이다. _RegisterScreenState 상태를 가진다. · _RegisterScreenState 클래스는 회원가입과 관련된 클래스이다. 아이디와 비밀번호를 입력하기 위한 텍스트 필드를 가지며 정상적으로 회원가입 시, WelcomeScreen 화면으로 이동한다. 실패 시 WelcomeScreen으로 돌아가며 실패한 이유를 다이얼로그로 띄어준다.

mainScreen, _MainScreenState



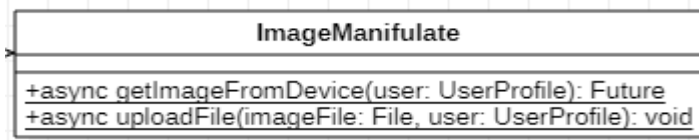
변수 및 메소드

· `_buildTile` : 매개변수로 받은 `Widget`을 꾸며주며, click시 실행되는 이벤트도 매개변수로 받아 Pack한다.

기능 및 설명

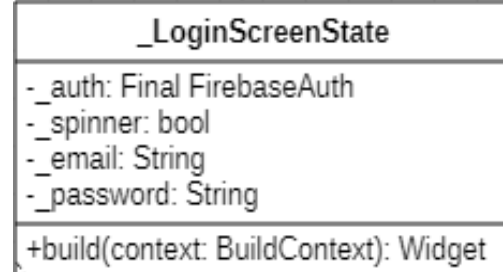
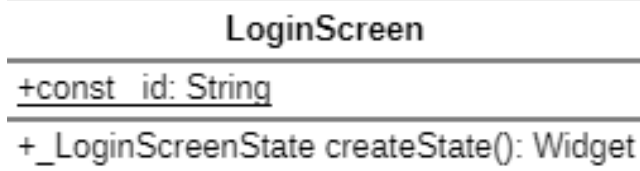
· **mainScreen** 클래스는 `StatefulWidget`이다. **_MainScreenState**를 상태로 가진다.
 · **_MainScreenState** 클래스는 '게시글 보기', '주소록', '조모임', '쪽지함', '내 프로필' `Widget`으로 이동할 수 있는 UI이다.

ImageManifulate



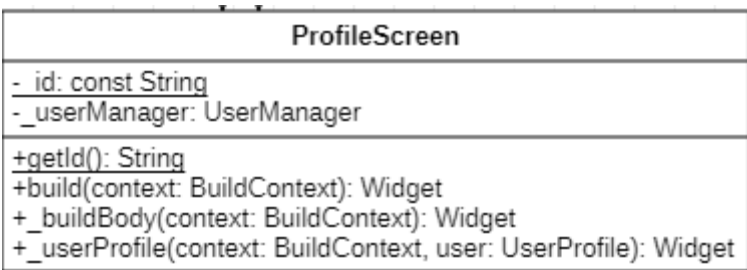
변수 및 메소드	<ul style="list-style-type: none"> · async getImageFromDevice : 모바일 디바이스의 갤러리에서 사진을 선택하여 경로를 가져온다. · async uploadFile: 받은 이미지 파일을 Google Firebase Storage에 저장한다.
기능 및 설명	<ul style="list-style-type: none"> · ImageManifulate는 Google Firebase Storage에 이미지를 저장하거나 가져오는 클래스이다.

LoginScreen & _LoginScreenState



변수 및 메소드	<ul style="list-style-type: none"> · _id : 페이지 전환을 위한 페이지 식별 id를 제공하는 String 형 id 변수 · getId() : 페이지 식별 id를 반환해주기 위한 함수 · _auth : FirebaseAuth 인스턴스를 초기화하는 변수 · _spinner : 비동기 작업을 Flag 및 Loading Progress Bar 실행 여부를 제공하는 변수
기능 및 설명	<ul style="list-style-type: none"> · 실제 Login 화면에 표현될 정보와 작동 방식은 _LoginScreenState의 build() 함수에 정의되어 있다. Class Diagram에서는 해당 build() 함수가 어떤 정보를 제공하는 지 표현하지 않는다. Dart 언어를 이용하여 Flutter로 작성된 어플리케이션의 경우, UI 구현과 동작구현이 명확히 분리되지 못하기 때문이다. 따라서 아래 Login screen을 보고, 대략적인 정보를 유

	<p>추할 수 있다.</p> <ul style="list-style-type: none"> · 이메일과 비밀번호를 입력할 수 있는 텍스트 필드 또한 <code>_LoginScreenState</code>의 <code>build()</code> 함수에서 제공된다. 그리고 실제 구현되는 로그인 관련 기능들 또한 <code>build()</code> 함수에서 제공된다. · Google Firebase Authentication System에 로그인하기 위해 <code>_auth</code>를 사용하여 사용자 로그인 메소드를 사용한다. · <code>_LoginScreenState</code>는 로그인을 위해 <code>_email</code>과 <code>_password</code> 변수를 가져야하며, Loading Progress Bar를 제공하기 위해 <code>spinner</code> 변수를 가져야 한다.
--	---

# Profile Screen	
 <pre> classDiagram class ProfileScreen { -id: const String -userManager: UserManager +getId(): String +build(context: BuildContext): Widget +_buildBody(context: BuildContext): Widget +_userProfile(context: BuildContext, user: UserProfile): Widget } </pre>	
변수 및 메소드	<ul style="list-style-type: none"> · <code>_id</code> : 페이지 전환을 위한 페이지 식별 id를 제공하는 String 형 id 변수 · <code>userManager</code> : Google Firebase Database System에 접근해 User의 Profile 정보를 가져오기 위한 UserManager 클래스의 인스턴스 · <code>_buildBody()</code> : Google Firebase Database System에 접근 여부에 따른 ProfileScreen의 Body를 구현하는 함수 · <code>_userProfile()</code> : Google Firebase Database System에서 가져온 User의 Profile를 출력하는 함수 · <code>getId()</code> : 페이지 식별 id를 반환해주기 위한 함수
기능 및 설명	<ul style="list-style-type: none"> · 실제 Profile이 화면에 표현될 정보와 작동 방식은 ProfileScreen의 <code>UserProfile()</code> 함수에 정의되어 있다.

· Google Firebase Database System에 접근하기 위한 UserManager 클래스를 인스턴스하여 사용한다.

ProfileCardHolder

ProfileCardHolder
+cardKey: GlobalKey<FlipCardState> +user: final UserProfile
+ProfileCardHolder({key: Key, this.user}) +build(context: BuildContext): Widget

변수 및 메소드

· cardKey : FlipCard의 상태를 저장하는 변수
· user : User의 Profile 정보가 저장된 데이터

기능 및 설명

· FlipCard를 구현하기 위해 실제 화면에 표현될 정보와 작동 방식을 build() 함수에 정의되어 있다
· ProfileCardHolder를 통해 FlipCard 동작을 구현하기 위한 card의 상태를 저장하는 cardKey 변수를 사용한다.

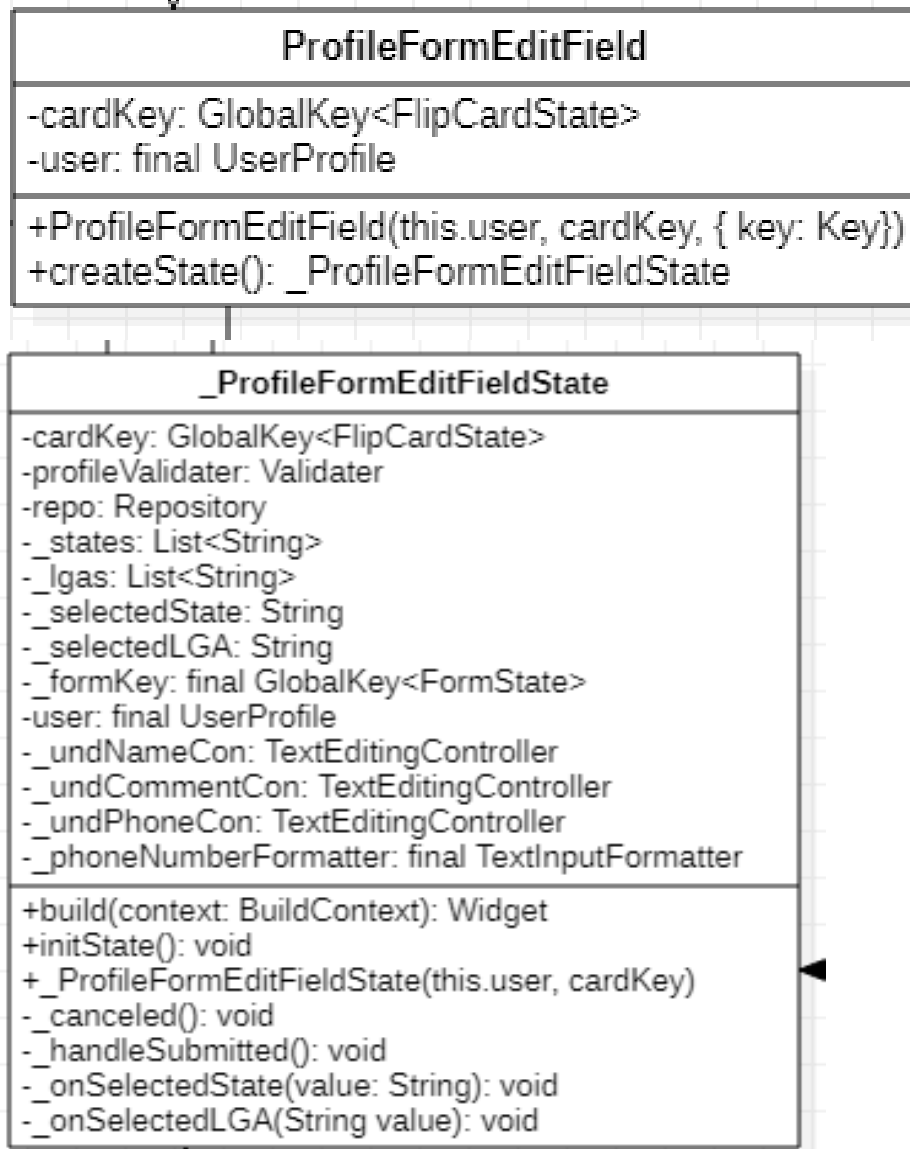
MyProfileCard

```

class MyProfileCard {
    -user: final UserProfile
    -cardKey: GlobalKey<FlipCardState>
    +MyProfileCard(this.user, cardKey, {key: Key })
    +user(): final UserProfile
}
    
```

변수 및 메소드	<ul style="list-style-type: none"> · user : User의 Profile 정보가 저장된 데이터 · cardKey : FlipCard의 상태를 저장하는 변수
기능 및 설명	<ul style="list-style-type: none"> · User의 Profile 정보가 저장된 데이터 user를 사용하여 실제 화면에 Profile을 출력한다. · FlipCard 동작을 구현하기 위해 card의 상태를 저장하는 cardKey 변수를 사용한다.

ProfileFormEditField & _ProfileFormEditFieldState



변수 및 메소드

- cardKey : FlipCard의 상태를 저장하는 변수
- user : User의 Profile 정보가 저장된 데이터
- profileValidator : 입력필드의 입력된 정보의 적합성을 검증하기 위한 Validator 클래스의 인스턴스
- repo : 대학/학과 정보가 저장된 데이터
- _states : 대학 정보가 저장된 리스트
- _lags : 학과 정보가 저장된 리스트
- _selectedState : 선택된 대학을 저장하는 String 변수
- _selectedLGA : 선택된 학과를 저장하는 String 변수
- _formKey : 텍스트 필드 폼의 상태를 저장하기 위한 변수
- _undNameCon : Name textFiled에 입력된 값을 제어하기 위한 TextEditController 객체의 인스턴스

	<ul style="list-style-type: none"> · <code>_undCommentCon</code> : Comment textFiled에 입력된 값을 제어하기 위한 <code>TextEditingController</code> 객체의 인스턴스 · <code>_undPhoneCon</code>: Phone textFiled에 입력된 값을 제어하기 위한 <code>TextEditingController</code> 객체의 인스턴스 · <code>_phoneNumberFormatter</code> : 전화번호 입력 형식을 정의하기 위한 <code>TextInputFormatter</code> 객체의 인스턴스 · <code>initState()</code> : <code>_ProfileFormEditFieldState</code>의 초기 상태를 설정하기 위한 함수 · <code>_canceled()</code> : 프로필 수정을 취소하기 위한 함수 · <code>_handleSubmitted()</code> : 프로필 수정을 완료하기 위한 함수 · <code>_onSelectedState()</code> : 대학 선택시 수행하는 함수 · <code>_onSelectedLGA()</code> : 학과 선택시 수행하는 함수
기능 및 설명	<ul style="list-style-type: none"> · 실제 프로필 수정 화면에 표현될 정보와 작동 방식은 <code>_ProfileFormEditFieldState</code>의 <code>build()</code> 함수에 정의되어 있다. · User의 프로필 정보가 저장된 데이터 <code>user</code> 인스턴스를 바탕으로 프로필을 수정할 수 있도록 한다. · FlipCard 동작을 구현하기 위해 <code>card</code>의 상태를 저장하는 <code>cardKey</code> 변수를 사용한다. · 프로필에 관련된 이름, 한 줄 소개, 전화번호, 대학/학과를 수정할 수 있는 텍스트 필드는 <code>_ProfileFormEditFieldState</code>의 <code>build()</code> 함수에서 제공된다. · <code>_handleSubmitted()</code> 함수를 통해 Google Firebase Data에 접근해 프로필을 수정하고 <code>_canceled()</code> 함수를 통해 프로필 수정을 취소한다.

UserProfile

<div style="border: 1px solid black; padding: 10px; margin: 10px;"> <div style="text-align: center; border-bottom: 1px solid black; margin-bottom: 10px;">UserProfile</div> <div> <code>+name: final String</code> <code>+email: final String</code> <code>+phone: final String</code> <code>+college: final String</code> <code>+comment: final String</code> <code>+department: final String</code> <code>+fromMap(map:Map<String, dynamic> , {this.reference})()</code> <code>+fromSnapshot(snapshot: DocumentSnapshot)</code> </div> </div>	
변수 및 메소드	<ul style="list-style-type: none"> · <code>reference</code> : Google Firebase Data의 프로필 정보를 참조하기 위한 변수

	<ul style="list-style-type: none"> · <code>userImage</code> : Google Firebase Storage에서 프로필 사진을 다운로드하기 위한 <code>url</code>을 저장하는 <code>String</code> 변수 · <code>private</code> : 프로필 공개 비공개 <code>flag</code>를 저장하는 <code>boolean</code> 변수 · <code>fromMap()</code> : Google Firebase Data에서 가져온 프로필 정보를 <code>UserProfile</code>에 매핑하는 함수 · <code>fromSnapshot()</code> : Google Firebase Data에서 가져온 프로필 정보를 <code>UserProfile</code> 데이터로 저장하는 함수
기능 및 설명	<ul style="list-style-type: none"> · Google Firebase Data에서 가져온 프로필 정보를 바탕으로 <code>fromSnapshot()</code>, <code>fromMap()</code> 함수를 통해 <code>UserProfile</code> 데이터를 생성한다.

Repository

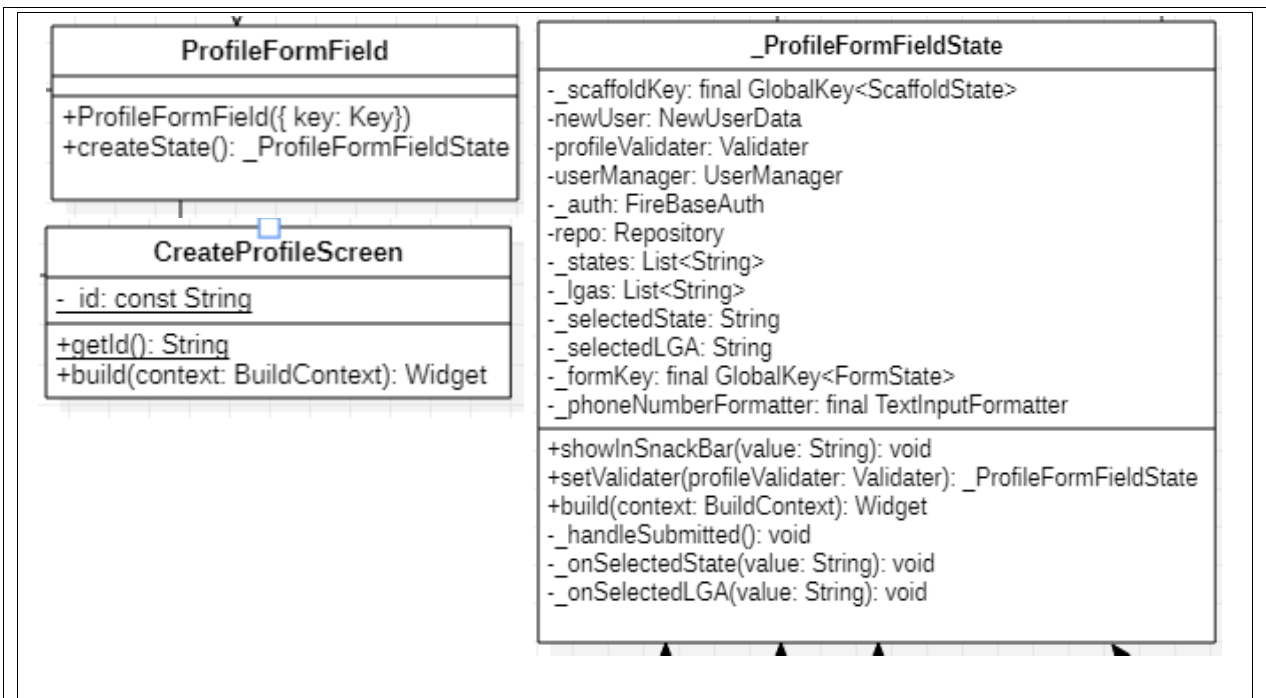
Repository
+_nigeria: List <dynamic>
+getAll(): List<Map>
+getLocalByState(state: String): List<String>
+getStates(): List<String>

변수 및 메소드	<ul style="list-style-type: none"> · _nigeria : 모든 대학/학과가 저장된 리스트 · getAll() : 모든 대학/학과를 반환하는 함수 · getLocalByState() : 선택된 대학에 대한 학과를 반환하는 함수 · getStates() : 모든 대학을 반환하는 함수
기능 및 설명	<ul style="list-style-type: none"> · _nigeria 리스트에 저장된 모든 대학/학과 정보를 바탕으로 getAll(), getLocalByState(), getStates() 함수를 통해 원하는 대학/학과 정보를 얻을 수 있다.

StateModel

<table border="1"> <thead> <tr> <th colspan="2">StateModel</th></tr> </thead> <tbody> <tr> <td>-state: String</td><td></td></tr> <tr> <td>-lgas: List<String></td><td></td></tr> <tr> <td>+fromJson(json:Map<String, dynamic>):StateModel()</td><td></td></tr> <tr> <td>+toJson():Map<String, dynamic>()</td><td></td></tr> </tbody> </table>		StateModel		-state: String		-lgas: List<String>		+fromJson(json:Map<String, dynamic>):StateModel()		+toJson():Map<String, dynamic>()	
StateModel											
-state: String											
-lgas: List<String>											
+fromJson(json:Map<String, dynamic>):StateModel()											
+toJson():Map<String, dynamic>()											
변수 및 메소드	<ul style="list-style-type: none"> · _state : 대학을 저장하는 String 변수 · _lgas : 선택된 대학의 학과가 저장된 리스트 · fromJson() ; 대학과 학과 리스트를 저장하는 StateModel 생성하는 함수 · toJson() : 대학/학과 데이터를 Map 구조로 제공하는 함수 										
기능 및 설명	<ul style="list-style-type: none"> · fromJson() 함수를 통한 대학과 그에 해당하는 학과 리스트가 저장된 StateModel 데이터를 생성하고, toJson() 함수를 통해 대학/학과 데이터를 Map 구조로 제공한다. 										

CreateProfileScreen & ProfileFormField & _ProfileFormFieldState

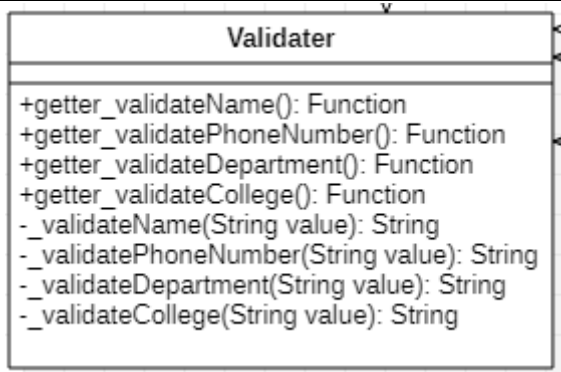


변수 및 메소드

- `_id` : 페이지 전환을 위한 페이지 식별 id를 제공하는 String형 변수
- `getId()` : 페이지 식별 id를 반환해주기 위한 함수
- `_scaffoldKey` : scaffold의 상태를 저장하는 변수
- `newUser` : 새로운 User의 프로필 정보를 저장하기 위한 데이터
- `profileValidator` : 텍스트 필드의 입력된 정보의 적합성을 검증하기 위한 Validator 클래스의 인스턴스
- `repo` : 대학/학과 정보가 저장된 데이터
- `userManager` : Google Firebase Database System에 접근해 User의 Profile 정보를 가져오기 위한 UserManager 클래스의 인스턴스
- `_auth` : FirebaseAuth 인스턴스를 초기화하는 변수
- `_states` : 대학 정보가 저장된 리스트
- `_lga` : 학과 정보가 저장된 리스트
- `_selectedState` : 선택된 대학을 저장하는 String 변수
- `_selectedLGA` : 선택된 학과를 저장하는 String 변수
- `_formKey` : 텍스트 필드 폼의 상태를 저장하기 위한 변수
- `_phoneNumberFormatter` : 전화번호 입력 형식을 정의하기 위한 TextInputFormatter 객체의 인스턴스
- `showInSnackBar()` : 텍스트 필드의 입력된 정보의 적합성이 맞지 않을 경우 오류 메시지를 출력하는 함수
- `setValidator()` : 텍스트 필드의 입력 validator를 설정하는 함수

	<ul style="list-style-type: none"> · <code>_handleSubmitted()</code> : 프로필 수정을 완료하기 위한 함수 · <code>_onSelectedState()</code> : 대학 선택시 수행하는 함수 · <code>_onSelectedLGA()</code> : 학과 선택시 수행하는 함수
기능 및 설명	<ul style="list-style-type: none"> · 실제 프로필 작성 화면에 표현될 정보와 작동 방식은 <code>_ProfileFormFieldState</code>의 <code>build()</code> 함수에 정의되어 있다. · 프로필을 작성할 수 있는 텍스트 필드와 관련 기능들은 <code>_ProfileFormFieldState</code>의 <code>build()</code> 함수에서 제공된다. · <code>_ProfileFormFieldState</code>는 <code>_handleSubmitted()</code> 함수와 <code>userManager</code>를 통해 Google Firebase Database System에 새로운 프로필을 추가하며, <code>profileValidator</code>와 <code>showInSnackBar()</code> 함수를 통해 텍스트 필드에 입력된 데이터가 적합하지 않으면 프로필 작성에 실패하고 오류를 메시지를 출력한다.

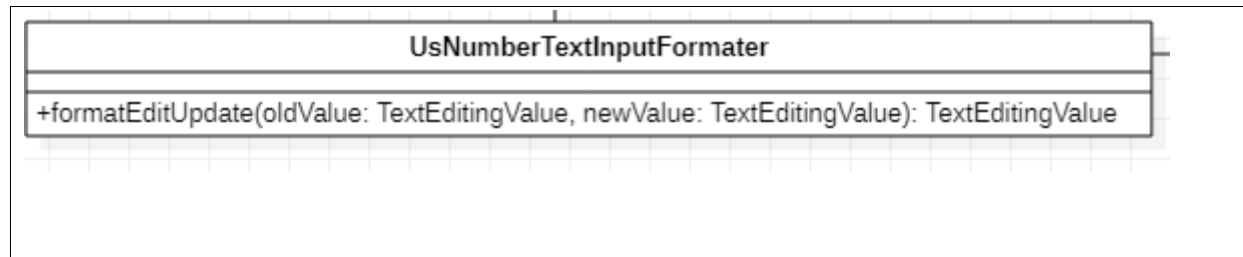
Validator



변수 및 메소드	<ul style="list-style-type: none"> · <code>getter_validateName()</code> : 프로필 이름 유효성을 확인 함수를 반환 · <code>getter_validatePhoneNumber()</code> : 프로필 전화번호 유효성을 확인 함수를 반환 · <code>getter_validateDepartment()</code> : 프로필 대학 유효성을 확인 함수를 반환 · <code>getter_validateCollege()</code> : 프로필 학과 유효성을 확인 함수를 반환 · <code>_validateName()</code> : 프로필 이름 입력 시 유효성을 확인하는 함수 · <code>_validatePhone()</code> : 프로필 전화번호 입력 시 유효성을 확인하는 함수 · <code>_validateDepartment()</code> : 프로필 대학 입력 시 유효성을 확인하는 함수 · <code>_validateCollege()</code> : 프로필 학과 입력 시 유효성을 확인하는 함수
----------	--

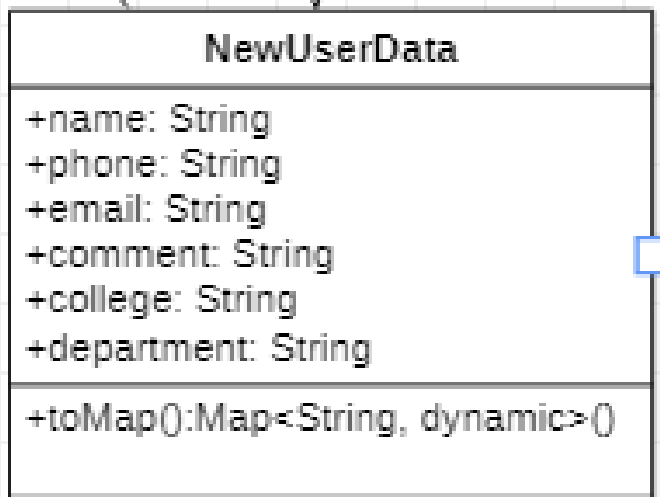
	는 함수
기능 및 설명	Validator는 이름, 전화번호, 대학, 학과 유효성 검사 함수들로 구성되어 있고 유효성 검사 함수를 반환하는 getter 함수들로 구성되어 있어 프로필 작성 및 수정 시 필요한 Validate 기능들을 제공한다.

UsNumberTextInputFormatter



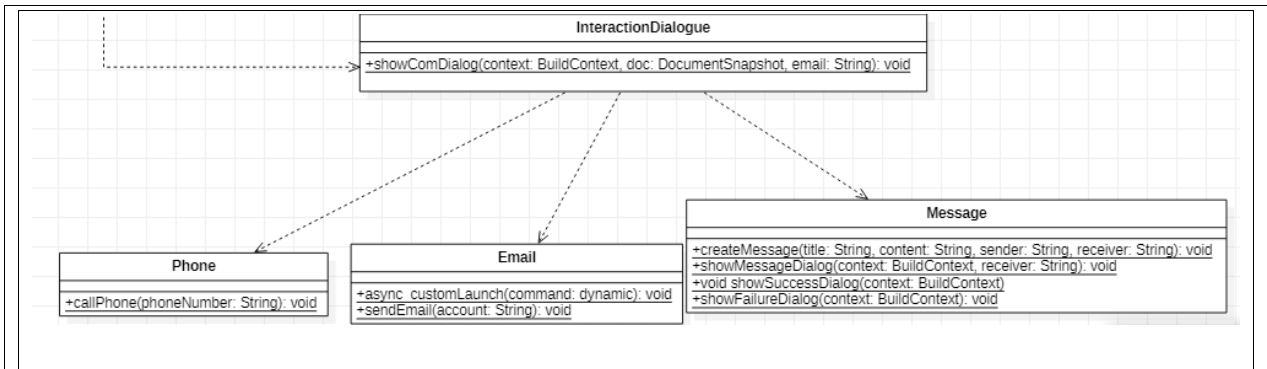
변수 및 메소드	· formatEditUpdate() : 전화번호 입력 형식을 맞추어 입력 및 출력해주는 함수
기능 및 설명	· formatEditUpdate() 함수를 통해 전화번호 입력 형식을 ###-####-#####로 설정하여 모든 Profile에 저장되는 전화번호 데이터의 format을 일관성있게 한다.

NewUserData



변수 및 메소드	<ul style="list-style-type: none"> · name : 새로운 Profile을 추가하기 위한 데이터 중 이름에 해당하는 String 변수 · phone : 새로운 Profile을 추가하기 위한 데이터 중 전화번호에 해당하는 String 변수 · email : 새로운 Profile을 추가하기 위한 데이터 중 이메일에 해당하는 String 변수 · comment : 새로운 Profile을 추가하기 위한 데이터 중 한 줄 소개에 해당하는 String 변수 · college : 새로운 Profile을 추가하기 위한 데이터 중 대학에 해당하는 String 변수 · department : 새로운 Profile을 추가하기 위한 데이터 중 학과에 해당하는 String 변수 · toMap() : 새로 추가할 Profile을 Map 구조 형식으로 반환하는 함수
기능 및 설명	<ul style="list-style-type: none"> · 새로 추가할 Profile에 대한 정보를 저장하는 데이터로 toMap() 함수를 사용하여 입력받은 모든 Profile 정보를 Map 구조 형식으로 만들어 반환하여 Google Firebase Database System에 추가하기 용이한 구조로 만든다.

InteractionDialogue & Phone & Email & Message



변수 및 메소드

- InteractionDialogue class의 showComDialog(context, doc, email)는 'YU, wish A+'의 다른 User와 연락할 수 있는 dialogue를 띄워주는 역할을 하는 메소드이다.
- Phone class의 callPhone(phoneNumber)는 전화번호를 매개변수로 받는 전화를 걸어주는 역할을 하는 메소드이다.
- Email class의 customLaunch(command)는 email을 보내기 위한 앱을 실행 시킬 때, 오류가 발생할 경우 오류를 출력해 주기 위한 메소드이다.
- sendEmail(account)는 email을 String형 변수 account를 매개변수로 받아 account로 email을 전송해주는 역할을 하는 메소드이다.
- Message class의 showMessageDialog(context, receiver)는 String형 매개변수 receiver를 받아서 receiver에게 쪽지를 보내는 dialogue를 띄워주는 역할을 하는 메소드이다.
- createMessage(title, content, sender, receiver)는 showMessageDialog에서 띄워진 dialogue 안에 입력되는 title, content, sender, receiver를 매개변수로 받아 Google Firebase Storage에 쪽지를 저장하는 역할을 하는 메소드이다.
- showSuccessDialog(context)는 쪽지 전송이 성공할 경우 쪽지 전송의 성공을 알리는 dialogue를 띄워주는 메소드이다.
- showFailureDialog(context)는 쪽지 전송이 실패할 경우 쪽지 전송의 실패를 알리는 dialogue를 띄워주는 메소드이다.

기능 및 설명

· 'YU, wish A+'에서는 User가 다른 User와 소통, 즉 연락을 취하기 위해서는 InteractionDialogue 클래스의 showComDialog 메소드를 이용하여야 한다. 메소드가 실행되면 '사용자와 소통하기'라는 제목의 dialogue가 띄워진다. dialogue 안에는 뒤로가기 아이콘, 전화 아이콘, 이메일 아이콘, 쪽지 아이콘이 있다.

- 뒤로가기 아이콘은 원래 화면으로 돌아가게 한다.
- 전화 아이콘을 click하면 상대 User의 전화번호를 Google Firebase Database System에 접근한 후 찾아온다. 전화번호는 callPhone 메소드의 매개변수로 전달한다. callPhone 메소드는 User의 화면을 전환하여 스마트폰의 Default Calling Application으로 이동시킨다. 전화번호를 입력하는 칸에 매개변수의 전화번호를 입력해준다.
- 이메일 아이콘을 click하면 상대 User의 이메일 주소를 Google Firebase Database System에 접근한 후 찾아온다. 이메일 주소는 sendEmail 메소드의 매개변수로 전달한다. sendEmail 메소드는 User의 화면을 전환하여 스마트폰의 Default E-mail Application으로 이동시킨다. 메일의 제목을 'YU, wish A+에서 연락드려요.'로 설정한다. 내용을 '안녕하세요!'로 설정한다.
- 쪽지 아이콘을 click하면 상대 User의 계정으로 보내는 쪽지의 제목, 내용을 입력하는 dialogue를 showMessageDialogue 메소드를 이용하여 띄워준다. 쪽지의 제목과 내용을 입력하고 send를 누르면 createMessage 메소드를 실행한다. createMessage 메소드는 Google Firebase Storage에 보내고자 하는 쪽지를 저장한다. createMessage는 sender는 현재 User, receiver는 상대 User로 자동으로 매개변수를 받으며 title과 content는 User가 입력한 내용을 받는다.
- 쪽지 전송이 성공하면 showSuccessDialogue 메소드를 실행하여 쪽지 전송의 성공을 알리는 dialogue를 띄워준다. 쪽지의 양식 중 하나라도 비어 있으면, showFailureDialogue 메소드를 실행하여 쪽지 전송의 실패를 알리는 dialogue를 띄워준다.

MessageScreen & _MessageScreenState

MessageScreen	_MessageScreenState
<pre>- id: const String +getId(): String +createState(): _MessageScreenState +MessageScreen({Key key})</pre>	<pre>- _messageManager: MessageManager - scaffoldKey: final GlobalKey<ScaffoldState> - _pageController: PageController - myTabs: final List<Tab> +initState(): void +dispose(): void +build(context: BuildContext): Widget +receivedMessagePage(context: BuildContext): Widget +sendMessagePage(context: BuildContext): Widget</pre>

변수 및 메소드	<ul style="list-style-type: none"> · <code>_id</code> : 페이지 전환을 위한 페이지 식별 id를 제공하는 String 형 id 변수 · <code>getId()</code> : 페이지 식별 id를 반환해주기 위한 함수 · <code>_messageManager</code> : Google Firebase Storage에서 쪽지목록을 읽어와주기 위한 인스턴스 · <code>_pageController</code> : 화면에 나타낼 page를 control하기 위한 PageController 형 변수 · <code>myTabs</code> : tab bar를 이용하기 위한 List 형 변수 · <code>receivedMessagePage(context:BuildContext)</code> : 받은 쪽지함을 나타내는 context를 매개변수로 하는 Widget형 메소드 · <code>sendMessagePage(context:BuildContext)</code> : 보낸 쪽지함을 나타내는 context를 매개변수로 하는 Widget형 메소드
기능 및 설명	<ul style="list-style-type: none"> · 실제 쪽지함 화면에 표현될 정보와 작동 방식은 <code>_MessageScreenState</code>의 <code>build()</code> 함수에 정의되어 있다. Class Diagram에서는 해당 <code>build()</code> 함수가 어떤 정보를 제공하는 지 표현하지 않는다. Dart 언어를 이용하여 Flutter로 작성된 어플리케이션의 경우, UI 구현과 동작구현이 명확히 분리되지 못하기 때문이다. 따라서 아래 Message screen을 보고, 대략적인 정보를 유추할 수 있다. · tab bar는 <code>myTabs</code> 변수를 활용하여 구현한다. tab bar는 받은 쪽지함과 보낸 쪽지함 두 개의 tab으로 나누어져 있다. 받은 쪽지함 tab에 대한 정보는 <code>receivedMessagePage</code>에 있고, 보낸 쪽지함 tab에 대한 정보는 <code>sendMessagePage</code>에 있다. · Google Firebase Storage에 쪽지들이 저장되어 있으며, 그 쪽지들을 Google Firebase Database System에 접근하여 가져오는 기능은 <code>_messageManager</code> 인스턴스를 이용하게 구현했다.

MessageManager

MessageManager

- _colName: final String
- _auth: FirebaseAuth

+getSendedMessageList(): Stream<QuerySnapshot>
+getRecivedMessageList(): Stream<QuerySnapshot>

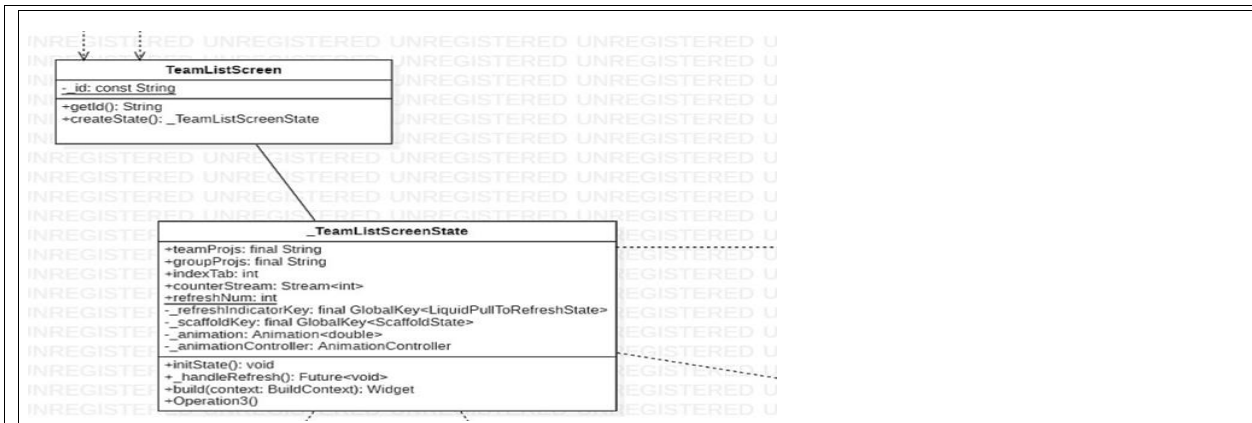
변수 및 메소드

- _colName : Google Firebase Database System에 접근할 때 사용하는 collection의 이름을 저장하는 String형 변수
- _auth : FirebaseAuth 인스턴스를 초기화하는 변수
- getSendedMessageList() : 받은 쪽지함의 쪽지 리스트를 snapshot의 형태로 받아서 list 해주는 메소드
- getRecivedMessageList() : 보낸 쪽지함의 쪽지 리스트를 snapshot의 형태로 받아서 list 해주는 메소드

기능 및 설명

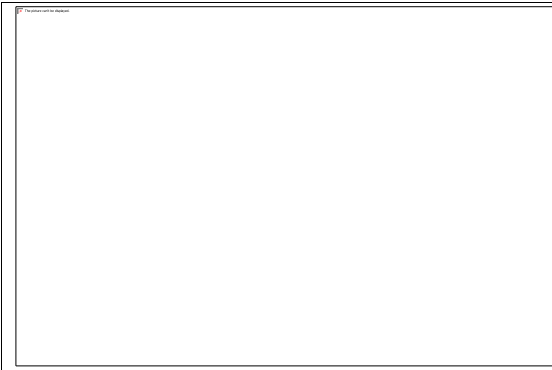
- _colName에 저장된 String형 변수는 'sendMessage'이다. 이는 Google Firebase Database System에서 쪽지 목록들을 저장하는 collection name이다.
- Google Firebase Authentication System에서 User 본인의 이메일 정보를 가져오기 위해 _auth를 사용한다. _auth.currentUser.email을 이용하여 가져올 수 있다.
- getSendedMessageList()는 Google Firebase Database System에 접근한다. collection name으로 _colName을 가지는 collection에 접근하여 쪽지를 보낸 User, 즉 sender가 User 본인인 쪽지들만을 선택하여 그것을 snapshot의 형태로 받아서 list 해준다.
- getRecivedMessageList()는 Google Firebase Database System에 접근한다. collection name으로 _colName을 가지는 collection에 접근하여 쪽지를 받는 User, 즉 receiver가 User 본인인 쪽지들만을 선택하여 그것을 snapshot의 형태로 받아서 list 해준다.

TeamListScreen, TeamListScreenState



<p>변수 및 메소드</p>	<ul style="list-style-type: none"> · <code>_id</code> : 페이지 전환을 위한 페이지 식별 id를 제공하는 String형 변수 · <code>getId()</code> : 페이지 식별 id를 반환해주기 위한 함수 · <code>teamProjs</code> : Google Fire Base Database 데이터 중 컬렉션명에 해당하는 String 변수 · <code>groupProjs</code> : Google Fire Base Database 데이터 중 컬렉션명에 해당하는 String 변수 · <code>indexTab</code> : 조모임 게시판의 3가지 탭에 정수형 데이터를 부여하기 위한 int 변수 · <code>counterStream</code> : 게시판의 새로고침 스트림을 일정 간격으로 반복적으로 시도하는 횟수를 세기 위한 변수. · <code>refreshNum</code> : 새로고침이 완료되는 스트림의 횟수를 지정하기 위한 변수. · <code>_refreshindicatorKey</code> : 새로고침 기능의 상태를 저장하는 변수. · <code>_scaffoldKey</code> : scaffold의 상태를 저장하는 변수 · <code>_animation</code> : 버튼의 애니메이션 가짓수를 저장하는 변수. · <code>_animationController</code> : 버튼의 애니메이션을 컨트롤하기 위한 <code>animationController</code>형 변수. · <code>_handleRefresh</code> : 새로고침 기능을 적용시키는 Future형 메소드.
<p>기능 및 설명</p>	<p>· <code>_TeamListScreenState</code> 클래스는 조별과제, 대외활동, 게시판을 탭으로 제시하고, 각 탭 별로 <code>Listitem</code>클래스를 리스트형태로 제공하는 클래스이다. 최상단에서 아래로 스크롤하면 업데이트가 된 게시물들로 새로고침이 적용된다.</p>

#SUBDetailScreen, _SUBDetailScreenState, CreateDocument, _CreateDocumentState



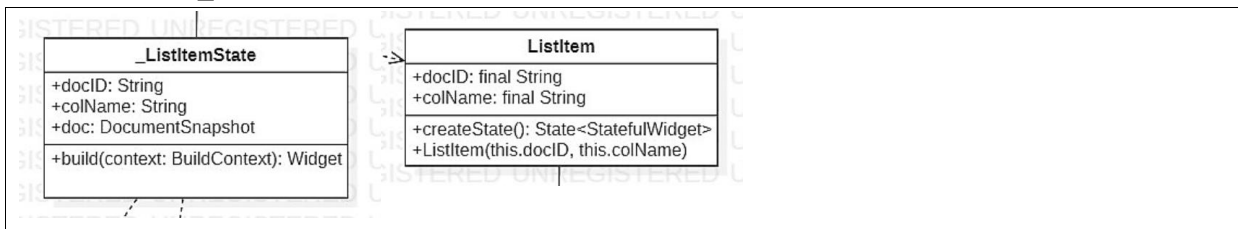
변수 및 메소드

- colName : (SUBDetailScreen) Google Firebase Database의 컬렉션 이름을 매개변수로 받기 위한 변수.
- docID : (SUBDetailScreen) Google Firebase Database의 Document ID를 매개변수로 받기 위한 변수.
- docID : (_SUBDetailScreenState) Google Firebase Database의 Document ID를 저장하기 위한 변수.
- colName : (_SUBDetailScreenState) _TeamListScreenState 클래스에서 불러온 Google Firebase Database의 컬렉션 이름을 저장하기 위한 변수.
- colName: (CreateDocument) 게시글을 작성 시 Google Firebase Database에 저장할 컬렉션 위치 이름을 저장하기 위한 변수.
- docID : (_CreateDocumentState) Google Firebase Database의 Document ID를 저장하기 위한 변수.
- colName : (_CreateDocumentState) Google Firebase Database의 컬렉션 이름을 저장하기 위한 변수.

기능 및 설명

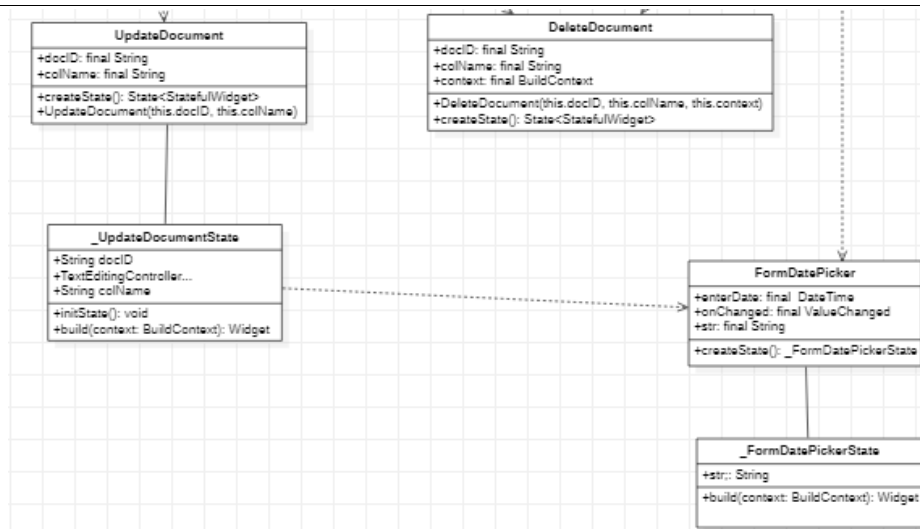
- _SUBDetailScreenState 클래스는 _TeamListScreenState 클래스에서 불러온 컬렉션 명과 Document ID를 사용하여 Google Firebase Database에 있는 게시글을 추적하여 내용을 User에게 제공하기 위한 클래스이다.
- _CreateDocumentState 클래스는 User에게 제목, 전공, 프로젝트명, 인원 수, 타과 인원 허용여부, 등록 마감일, 프로젝트 마감일, 내용을 입력하는 Screen을 제공하며, _TeamListScreenState 클래스에서 불러온 컬렉션 명을 통해 Google Firebase Database 에 해당 정보들을 저장한다.

ListItem, _ListItemState



변수 및 메소드	<ul style="list-style-type: none"> · docID : (ListItem) Google Firebase Database의 Document ID를 매개변수로 받기 위한 변수. · colName: (ListItem) Google Firebase Database의 컬렉션 이름을 매개변수로 받기 위한 변수. · docID : (_ListItemState) Google Firebase Database의 Document ID를 저장하기 위한 변수. · colName : (_ListItemState) Google Firebase Database의 컬렉션 명을 저장하기 위한 변수.
기능 및 설명	<ul style="list-style-type: none"> · TeamListScreen에 Google Firebase Database에서 게시글 정보를 받아 한 개의 카드 형태로 생성해주는 클래스이다.

UpdateDocument, _UpdateDocumentState, DeleteDocument, FormDatePicker, _FormDatePickerState



변수 및 메소드

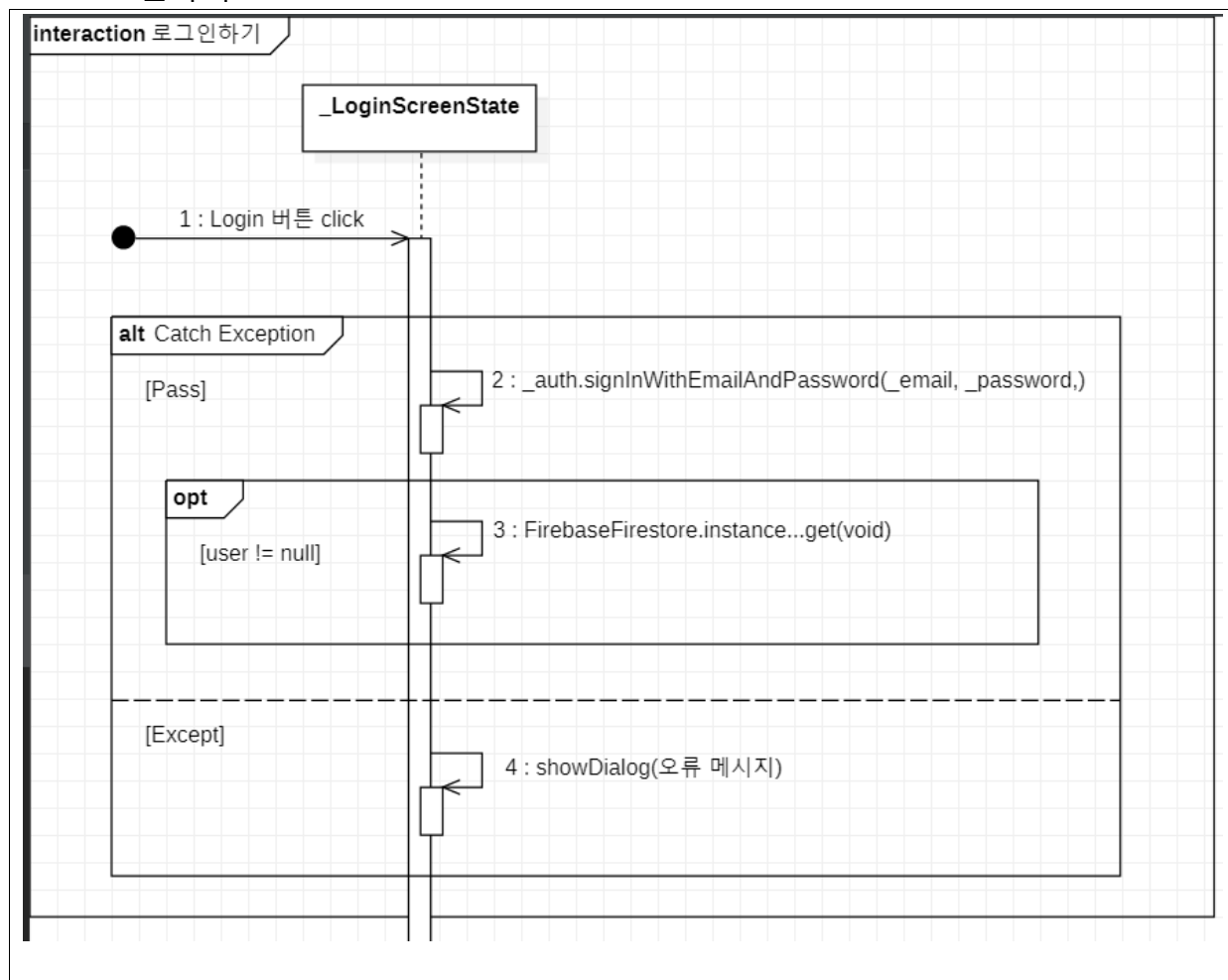
- colName : (SUBDetailScreen) Google Firebase Database의 컬렉션 이름을 매개변수로 받기 위한 변수.
- docID : (SUBDetailScreen) _Google Firebase Database의 Document ID를 매개변수로 받기 위한 변수.

기능 및 설명

- _UpdateDocumentState는 User에게 게시글을 수정하기 위해 새로운 정보를 입력할 창을 띄워주며, User가 게시글 수정을 진행할 경우, Google Firebase Database에 존재하는 해당 게시글의 정보들을 입력된 정보로 갱신한다.
- DeleteDocument는 User에게 게시글을 삭제하는지 안내문을 띄워주며, User가 게시글 삭제를 진행할 경우, Google Firebase Database에 존재하는 해당 게시글의 정보들을 삭제한다.
- _FormDatePicker 사용자가 등록마감일, 프로젝트 마감일을 설정할 때, 캘린더를 제공한다.

4. Sequence diagram

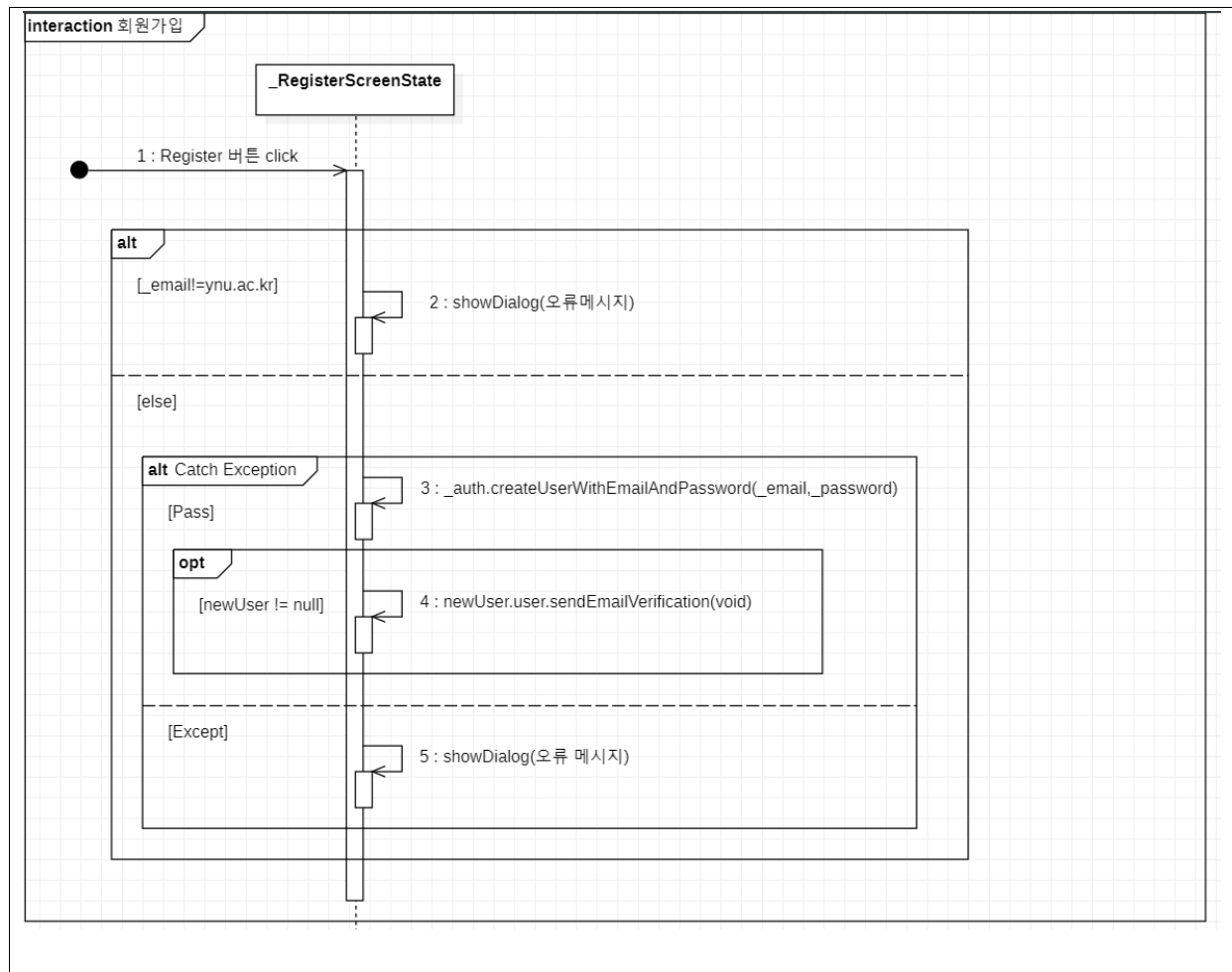
1. 로그인하기



기능 및 설명

- 'YU, wish A+' 어플리케이션을 사용하기 위해 회원인증을 위한 로그인하기 Use case의 과정을 나타낸 Sequence Diagram이다.
- User가 회원인증하기 위해 `_LoginScreenState` 클래스로부터 회원가입 때 입력한 email과 password를 입력하고 Login 버튼을 click하면 `_auth.signInWithEmailAndPassword()` 함수를 통해 Google Firebase Auth System에 접근해 유효한 email과 올바른 password인지 확인한다.
- 회원인증이 확인되면 로그인한 후 로그인한 email을 바탕으로 Google Firebase Database System에 접근하여 해당 계정의 프로필 작성 여부를 확인하고 여부에 따라 `createProfileScreen`이나 `mainScreen`으로 이동한다.
- 회원인증이 되지 않으면 오류 메시지를 출력한다,

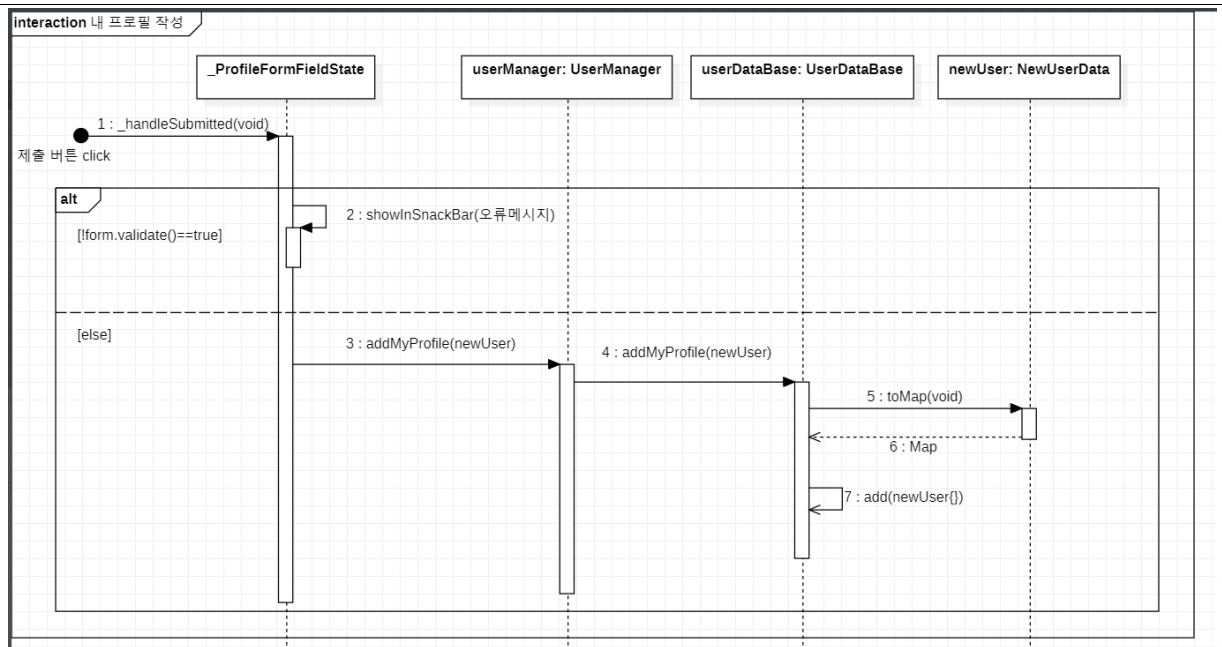
2. 회원가입



기능 및 설명

- ‘YU, wish A+’ 어플리케이션을 사용하기 위해 회원가입하는 과정의 Use case를 나타낸 Sequence Diagram이다.
- User가 `_RegisterScreenstate` 클래스로부터 올바른 email과 password가 입력되면 Google Firebase Auth System에 회원을 추가하여 정상적인 회원가입이 이루어지고, 올바르지 않은 email이나 password가 입력되는 경우에는 AlertDialog에 에러 메시지를 담아 출력한다.

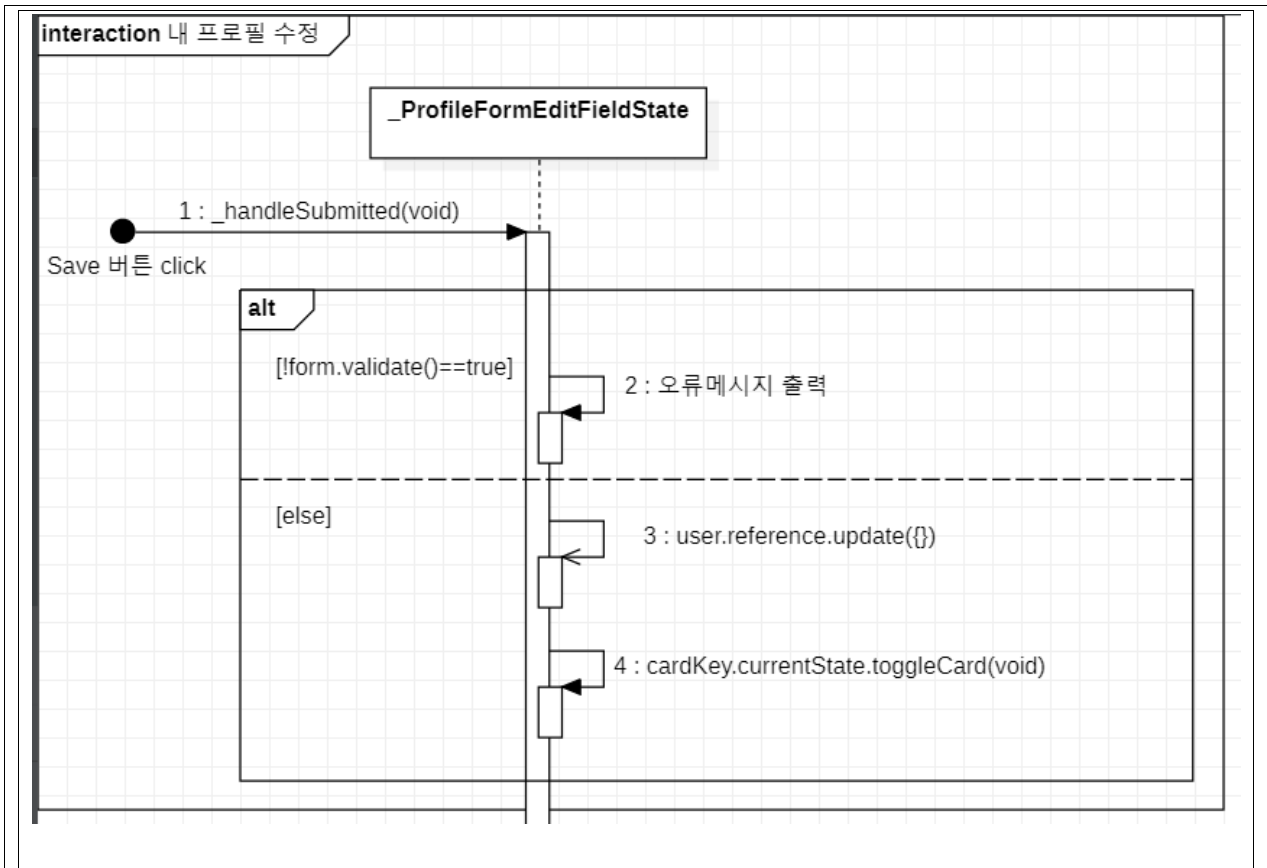
3. 내 프로필 작성



기능 및 설명

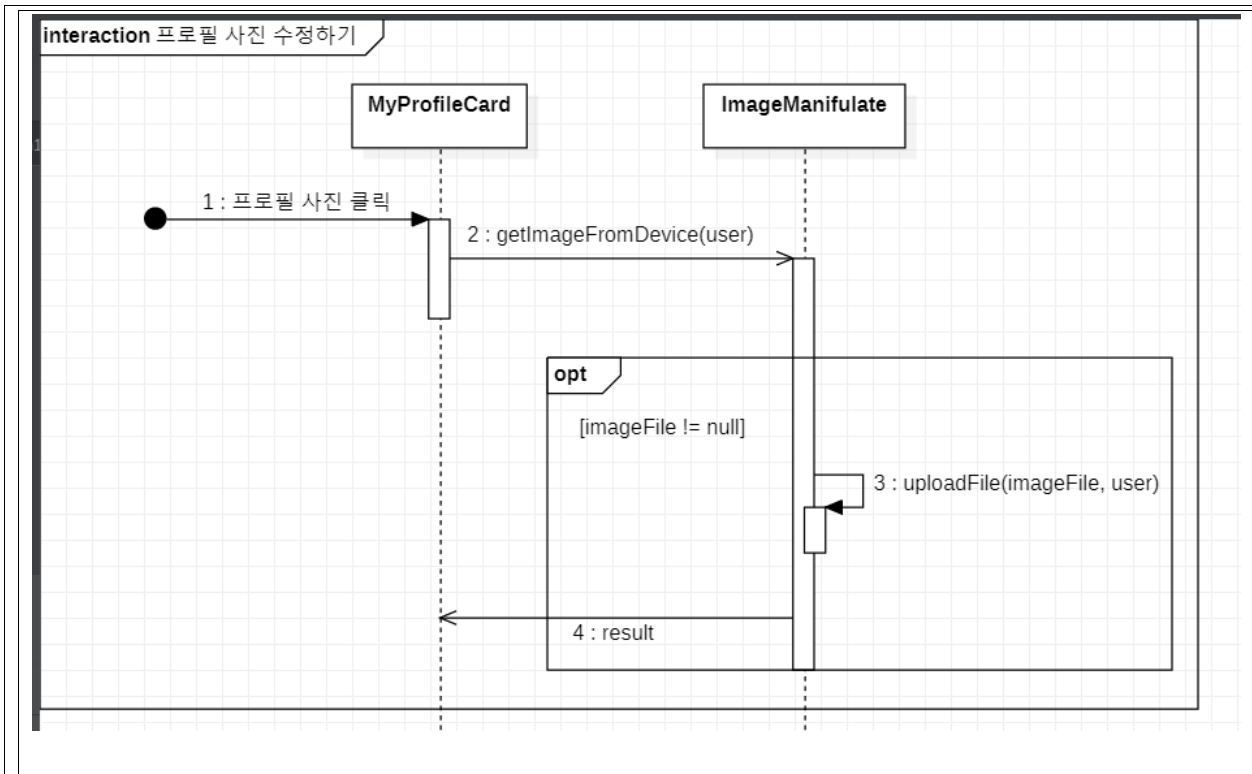
- ‘YU, wish A+’ 어플리케이션을 통해 프로필을 작성하기 위한 내 프로필 작성 Use case의 과정을 나타낸 Sequence Diagram이다.
- User가 `_ProfileFormFieldState` 클래스로부터 작성할 프로필 정보를 입력하고 제출 버튼을 click 하면 `_handleSubmitted()`를 사용하여 유효한 프로필 정보가 입력된 지 확인하여 올바른 프로필 정보 입력시 `userManager`와 `userDataBase`를 통해 Google Firebase Database System 접근하여 새로운 프로필 데이터를 추가하고, 정상적인 내 프로필 작성이 이루어진다.
- 유효하지 않은 프로필 정보가 입력되는 경우에는 에러 메시지를 출력한다.

4-1. 내 프로필 수정



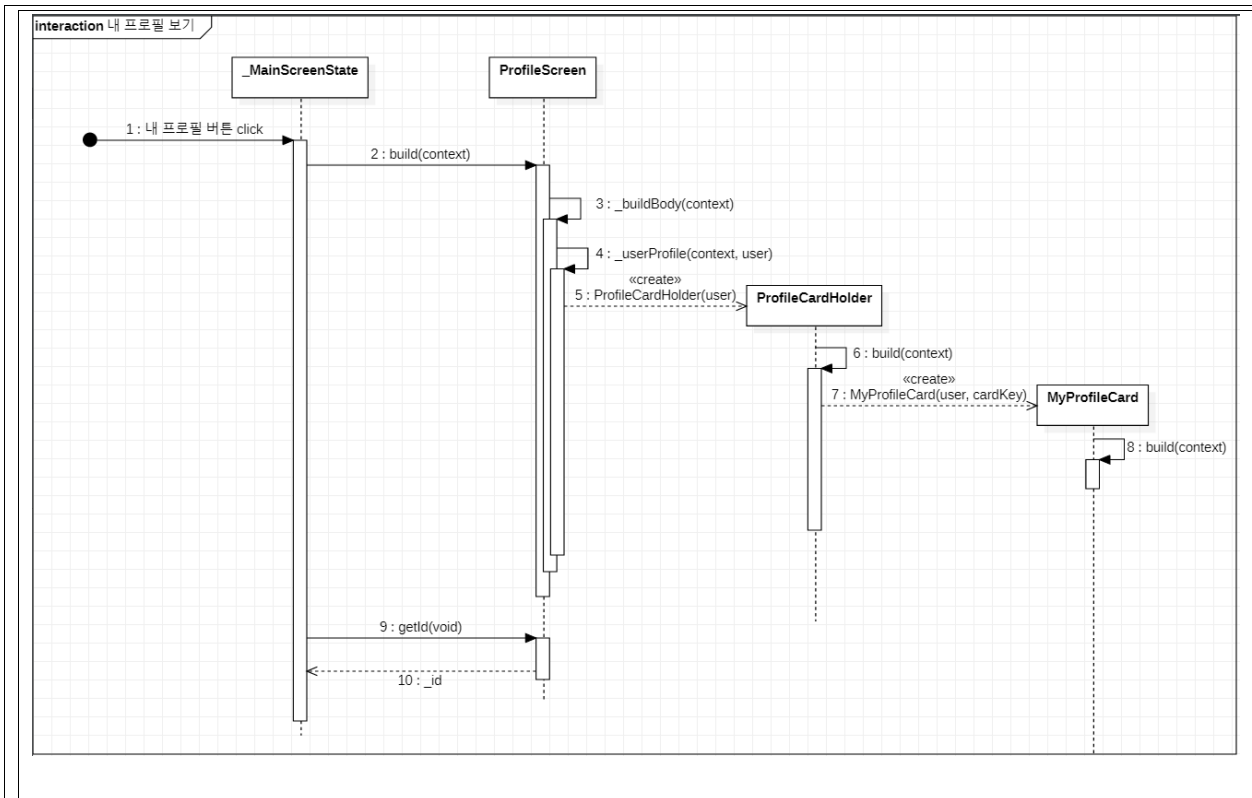
기능 및 설명

- ‘YU, wish A+’ 어플리케이션을 통해 본인이 작성한 프로필을 수정하기 위한 내 프로필 수정 Use case의 과정을 나타낸 Sequence Diagram이다.
- User가 _ProfileFormEditFieldState 클래스로부터 수정할 프로필 정보를 입력하고 Save 버튼을 click 하면 _handleSubmitted()를 사용하여 유효한 프로필 정보가 입력된 지 확인하여 올바른 프로필 정보 입력시 user를 통해 Google Firebase Database System 접근하여 프로필 데이터를 수정하고, 정상적인 내 프로필 수정이 이루어지고 cardKey를 통해 수정이 반영된 프로필을 출력한다.
- 유효하지 않은 프로필 정보가 입력되는 경우에는 에러 메시지를 출력한다.



기능 및 설명

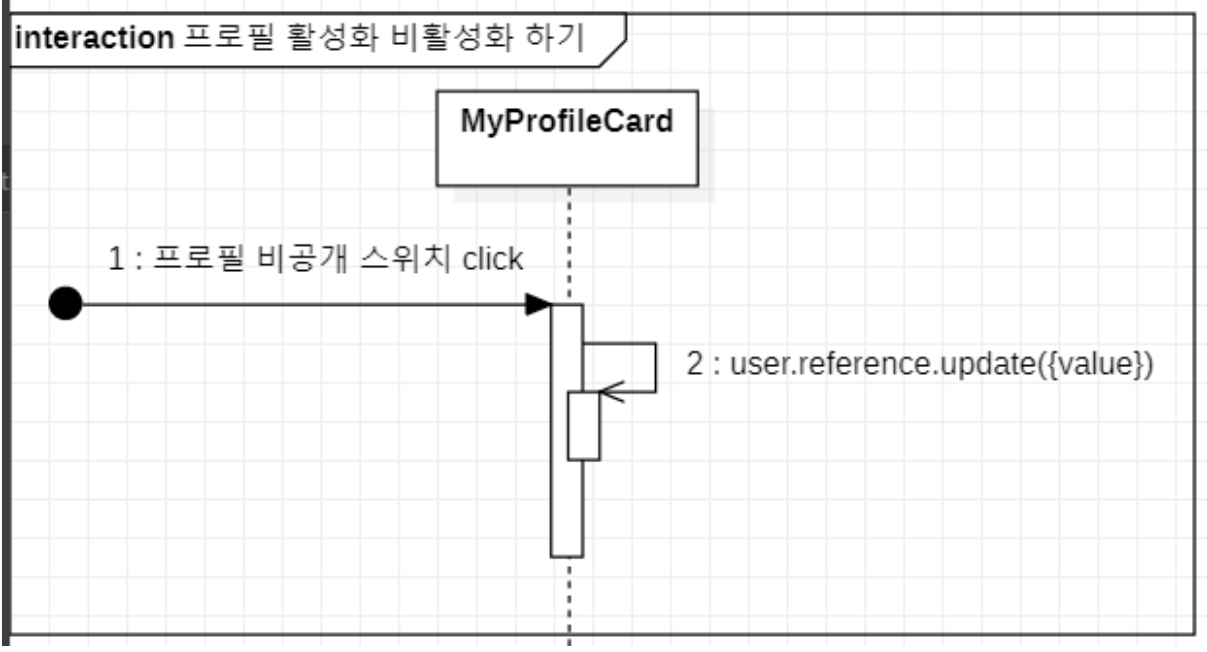
- ‘YU, wish A+’ 어플리케이션을 통해 프로필 사진을 수정하기 위한 내 프로필 사진 수정하기 Use case의 과정을 나타낸 Sequence Diagram이다.
- User가 현재 프로필 사진을 수정하기 위해 MyProfileCard 클래스로부터 현재 프로필 사진을 click하면 ImageManipulate 클래스의 getImageFromDevice() 함수를 사용하여 현재 스마트폰의 Default Photos Application를 실행하여 스마트폰 내에 저장된 사진을 선택할 수 있는 상태가 된다.
- User가 실행된 Default Photos Application에서 사진을 선택하면 ImageManipulate 클래스의 uploadFile() 함수를 사용하여 Google Firebase Storage에 접근하여 프로필 사진을 수정하고, 수정된 프로필 사진으로 반영된 프로필 결과를 출력한다.
- User가 실행된 Default Photos Application에서 사진을 선택하지 않으면, 프로필 사진 수정하기 Use case는 바로 종료한다.

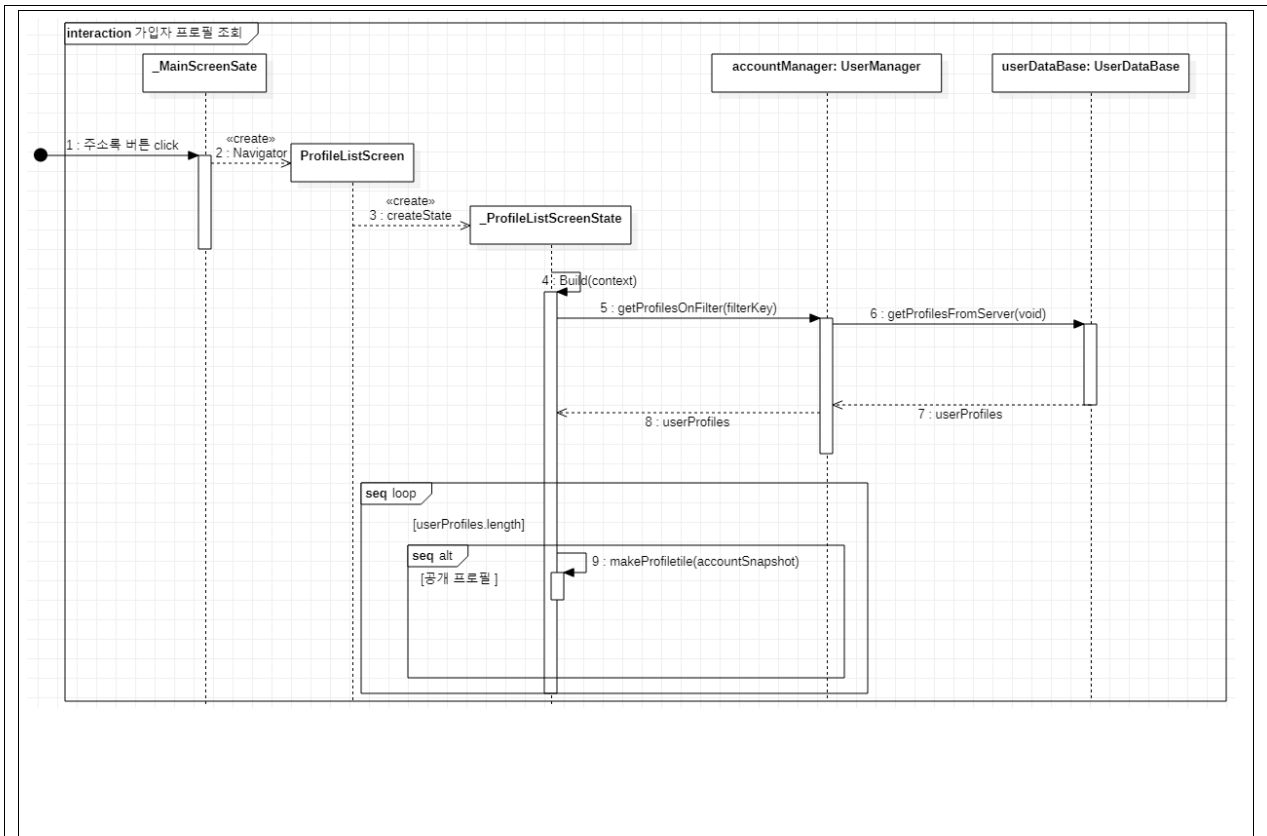


기능 및 설명

- ‘YU, wish A+’ 어플리케이션을 통해 작성된 자신의 프로필을 조회하기 위한 내 프로필 보기 Use case의 과정을 나타낸 Sequence Diagram이다.
- User가 현재 자신의 프로필을 조회하기 위해 _MainScreenState 클래스로부터 내 프로필 버튼을 click하면 ProfileScreen 클래스의 build() 함수를 사용하여 Google Firebase Database System에서 가져온 User의 프로필을 출력할 전반적인 Body(ProfileCardHolder와 MyProfileCard)를 생성한다.
- _MainScreenState 클래스는 프로필을 출력할 Body가 생성된 후 getId() 함수를 통해 ProfileScreen의 _id를 얻어 ProfileScreen으로 이동하여 User에게 프로필을 조회할 수 있게 제공한다.

6. 프로필 활성화 비활성화 하기

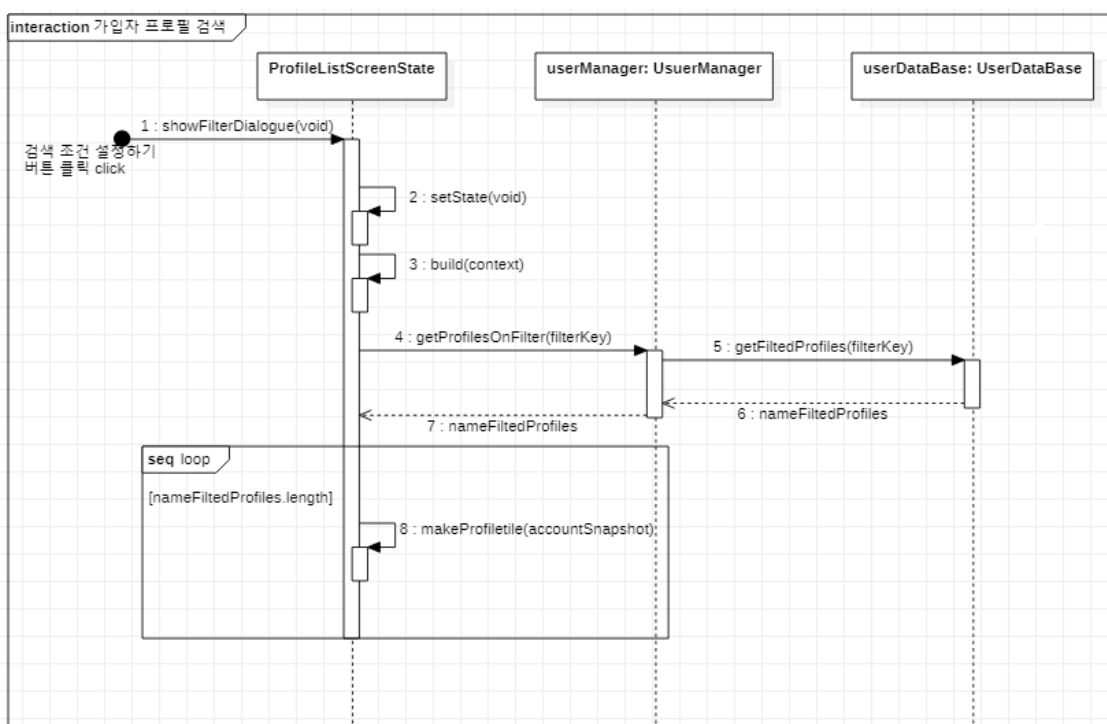
	 <pre> sequenceDiagram participant User participant MyProfileCard User->>MyProfileCard: 1 : 프로필 비공개 스위치 click activate MyProfileCard MyProfileCard->>MyProfileCard: 2 : user.reference.update({value}) deactivate MyProfileCard </pre>
<p>기능 및 설명</p>	<ul style="list-style-type: none"> · ‘YU, wish A+’의 주소록에서 User의 프로필이 출력될지 여부를 설정하기 위한 과정의 Use case를 나타낸 Sequence Diagram이다. · User가 MyProfileCard 클래스로부터 프로필 비공개 스위치를 click하면 Google Firebase Database System 접근하여 현재 사용하고 있는 계정의 프로필 공개 여부 설정 값을 변경한다.



기능 및 설명

- _MainScreenState의 위젯에서 '주소록' 버튼을 클릭할 경우 실행된다. ProfileListScreen이 생성되면 화면이 전환되게 된다. ProfileListScreen는 createState를 통해 state를 생성한다. 생성된 ProfileListScreenState는 build(context)를 통해 _ProfileListScreenState는 화면을 제공한다. 화면 제공시 활용되는 resource인 가입자 프로필을 getProfilesOnFilter를 사용하여 가져온다. filterKey:String가 empty이면 getProfilesFromServer를 실행시킨다. empty가 아닌 경우는 가입자 프로필 검색 sequence에서 볼 수 있으므로 생략한다. Google Firebase Database System으로 받은 가입자 프로필 정보는 화면을 구성하는데 사용된다. _ProfileListScreenState는 받은 가입자 프로필 정보를 makeProfiletile을 사용하여 출력한다. makeProfiletile은 비공개, 공개 프로필을 구분하여 위젯을 반환한다.

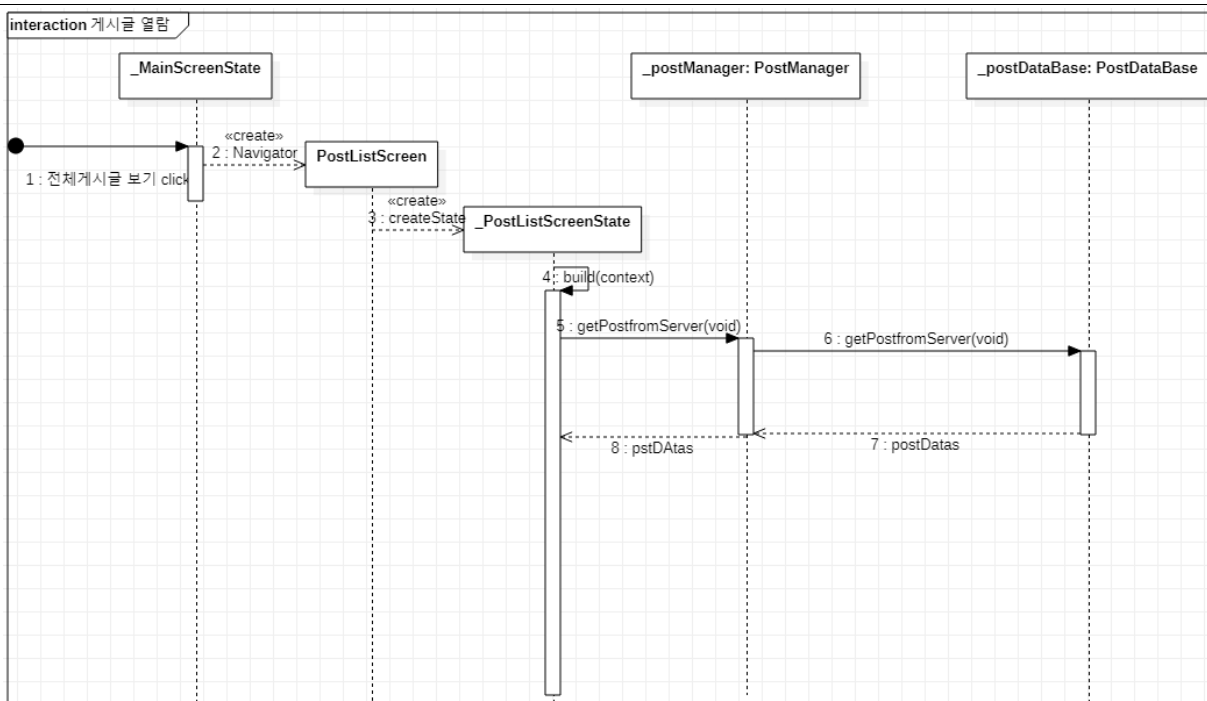
8. 가입자 프로필 검색



기능 및 설명

· 가입자 프로필 검색은 ProfileListScreenState가 제공하는 UI에서 검색 기능 버튼을 클릭 시 시작된다. showFilterDialogue로 이름 혹은 학과 조건키를 설정할 수 있다. 적용 시 setState를 통해 화면을 재구성하게 된다. ‘사용자 프로필 조회’와는 다르게 조건키가 생겼기 때문에 getProfilesOnFilter는 getFiltedProfiles를 실행시킨다. 이름으로 검색했을 경우, 학과로 선택했을 경우에 따라 userDataBase는 조건에 부합하는 데이터들만 넘겨준다. _ProfileListScreenState는 받은 사용자 프로필 정보를 이용해 화면을 구성한다.

9. 게시물 열람

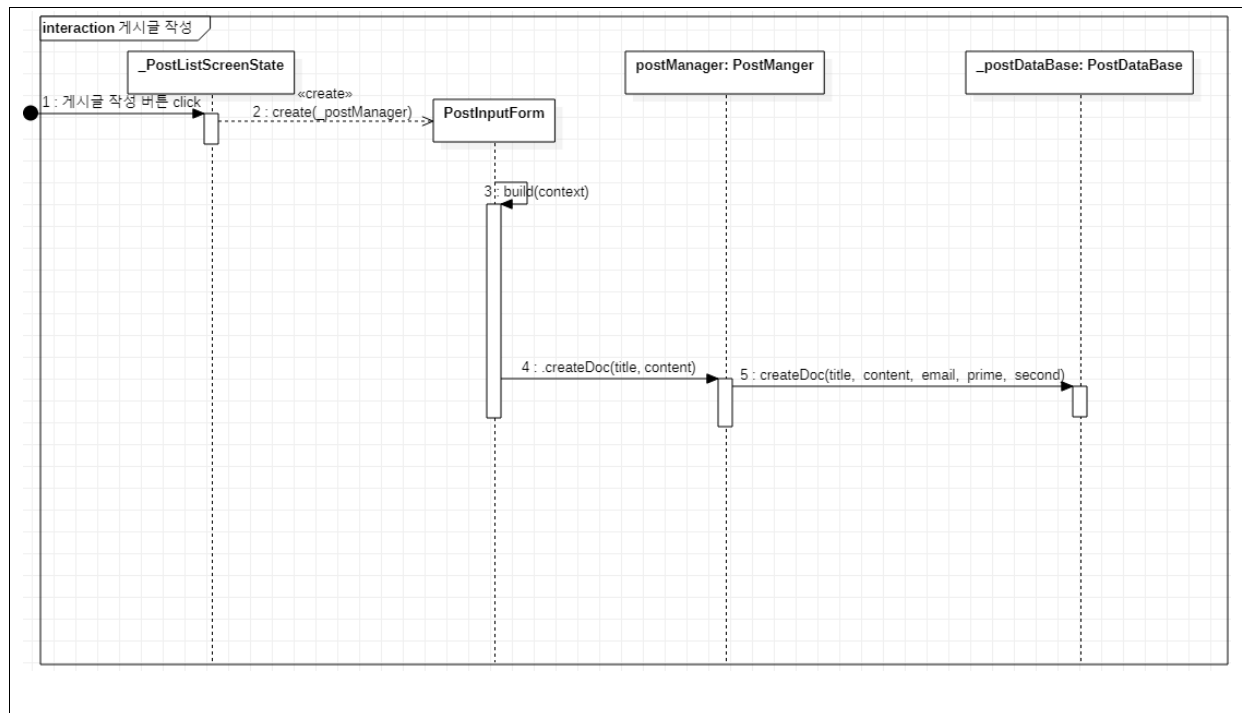


기능 및 설명

· `_MainScreenState`에서 전체 게시물 보기를 click하면 실행되는 시퀀스이다. 화면 전환을 위해 `Navigator`에 `PostListScreen` 인스턴스를 준다. `PostListScreenState`는 `_PostListScreenState`를 생성하고, `_PostListScreenState`는 화면을 구성하기 위한 resource인 게시물 정보를 얻어오기 위해 `_postManager`의 `getPostfromServer`를 이용한다.

`_getPostformServer`는 `_postDataBase`의 메서드를 사용하여 받아온 게시물 정보를 `_PostListScreenState`에 넘겨준다. `_PostListScreenState`는 받은 게시물 정보를 이용하여 List 형태로 화면을 구성한다.

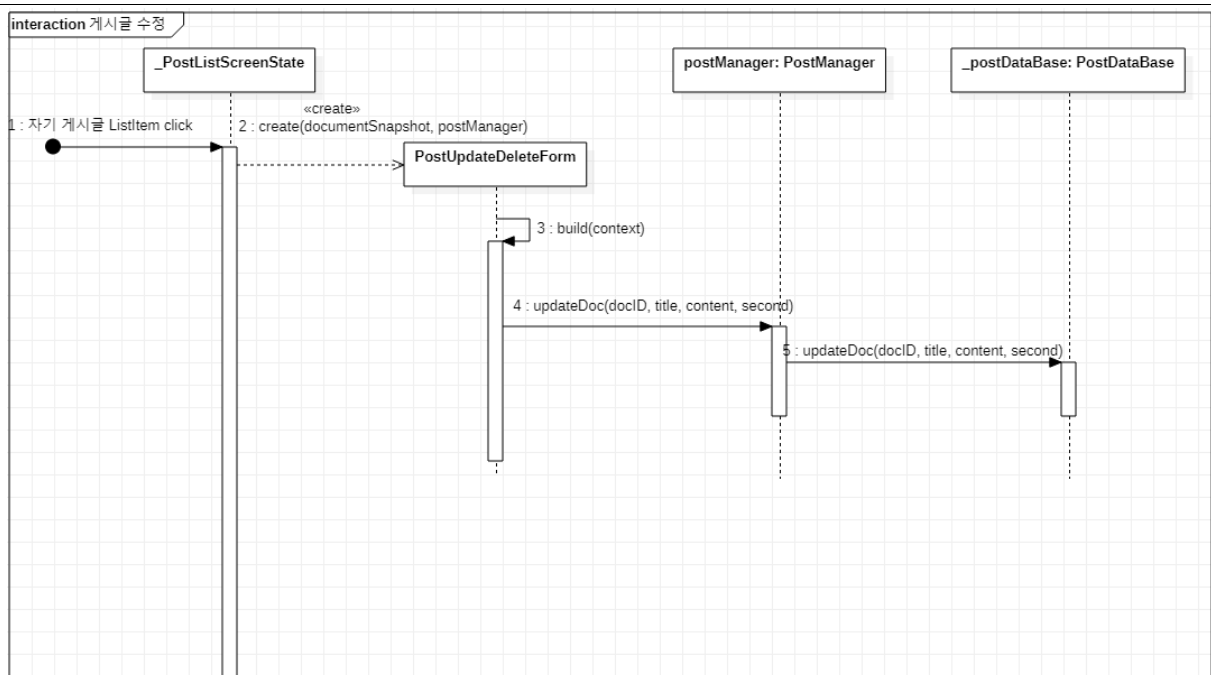
10. 게시물 작성



기능 및 설명

· 게시물 작성 시퀀스는 _ProfileListScreenState가 제공해주는 UI에서 게시물 작성 버튼을 click하면 시작된다. PostInputForm을 생성하여 게시물 작성을 하는 창을 생성한다. User가 제목과 내용을 입력하고 작성 버튼을 click 시 postManager.createDoc가 시작된다. 사용자 이메일을 통해 사용자 프로필 정보를 얻어와 _PostDataBase.createDoc의 매개변수로 사용한다. _PostDataBase.createDoc Google Firebase Database System에 작성한 게시물 정보 입력을 요청한다.

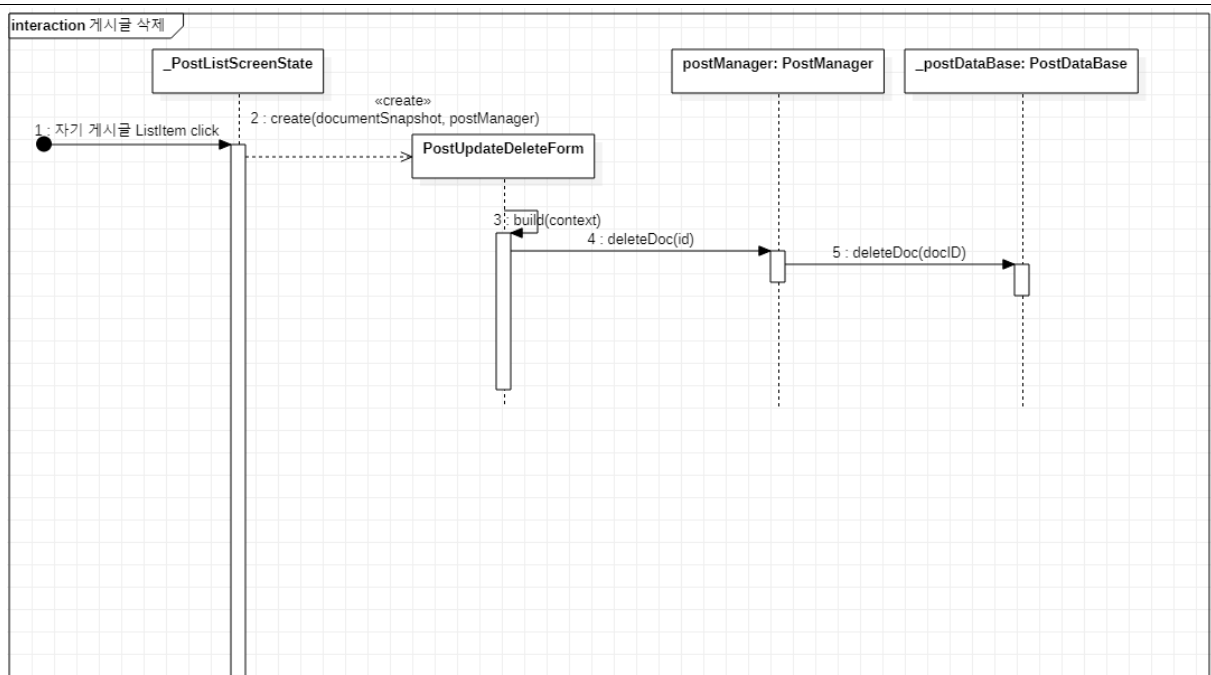
11-1. 게시물 수정



기능 및 설명

· 게시물 열람 시 User는 List형식의 게시물들을 볼 수 있다. 물론 자신의 게시물도 있는데, 자신의 게시물을 long click하면 이번 시퀀스가 실행된다. PostListScreenState는 이벤트 발생 시 PostUpdateDeleteForm을 생성시키면 화면을 전환시킨다. PostupdateDeleteForm은 원래 게시물의 제목 내용을 그대로 채워 넣어진 채로 사용자에게 UI를 제공한다. 사용자가 수정 후 버튼을 click시 postManager.updateDoc가 실행된다. 매개변수로 제목, 내용, 학과 그리고 Google Firebase Database System에서 도큐먼트의 고유 ID를 매개변수로 받는다. postManager.updateDoc에서는 postDatabase.updateDoc를 실행하게 되고 받은 정보를 Google Firebase Database System에 데이터를 저장하게 된다. 그 후 PostUpdateDeleteForm은 닫히고 PostList Screen State 화면으로 돌아간다.

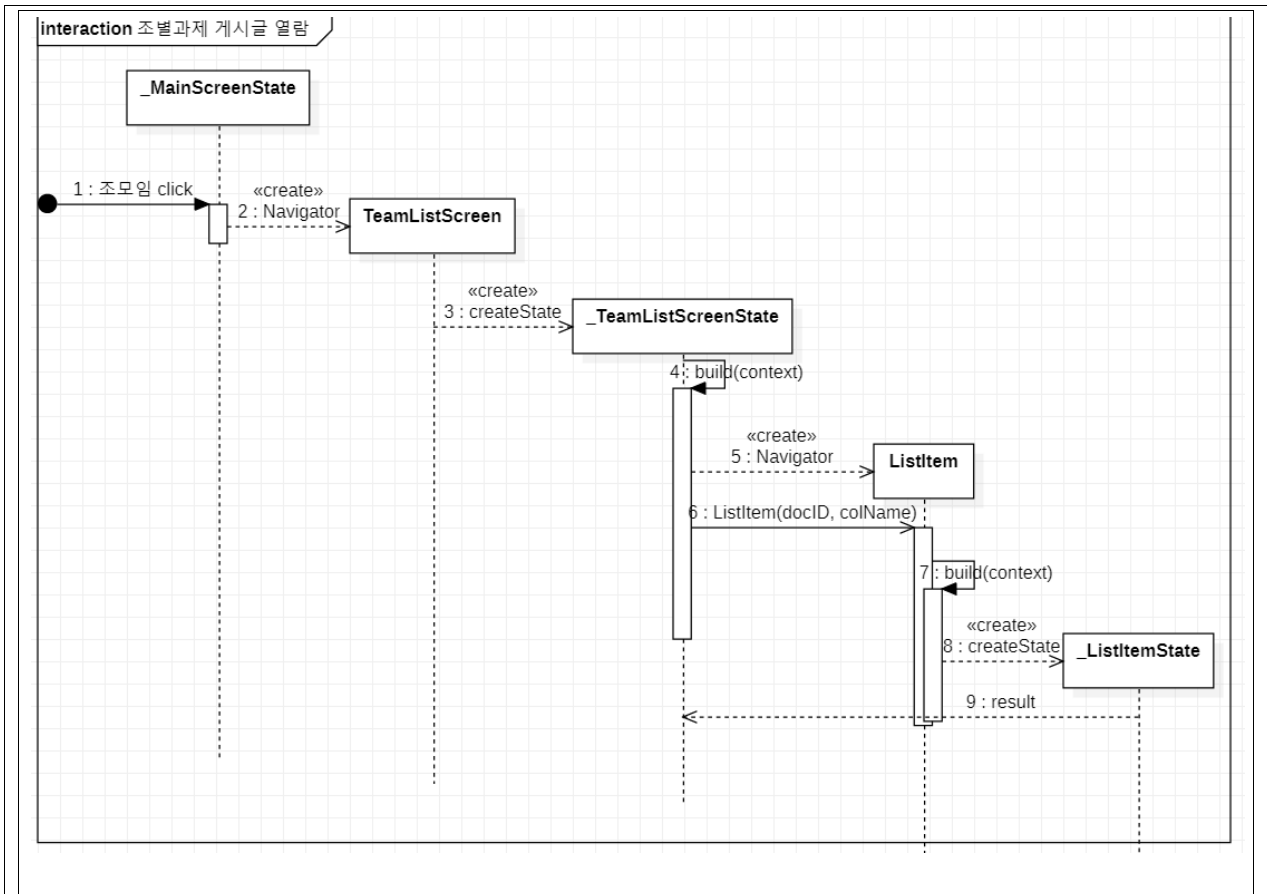
11-2. 게시물 삭제



기능 및 설명

· 게시물 열람 시 User는 List형식의 게시물들을 볼 수 있다. 물론 자신의 게시물도 있는데, 자신의 게시물을 long click하면 이번 시퀀스가 실행된다. PostListScreenState는 이벤트 발생 시 PostUpdateDeleteForm을 생성시키면 화면을 전환시킨다. PostupdateDeleteForm은 원래 게시글의 제목 내용을 그대로 채워 넣어진 채로 사용자에게 UI를 제공한다. 사용자가 삭제 버튼을 click하면 postManager.deleteDoc가 실행되고 postDataBase.deleteDoc가 실행된다. postDataBase.deleteDoc에서는 Google Firebase Database System에서 해당 게시물 문서를 삭제한다. 그 후 Post UpdateDeleteForm은 닫히고 PostListScreenState 화면으로 돌아간다.

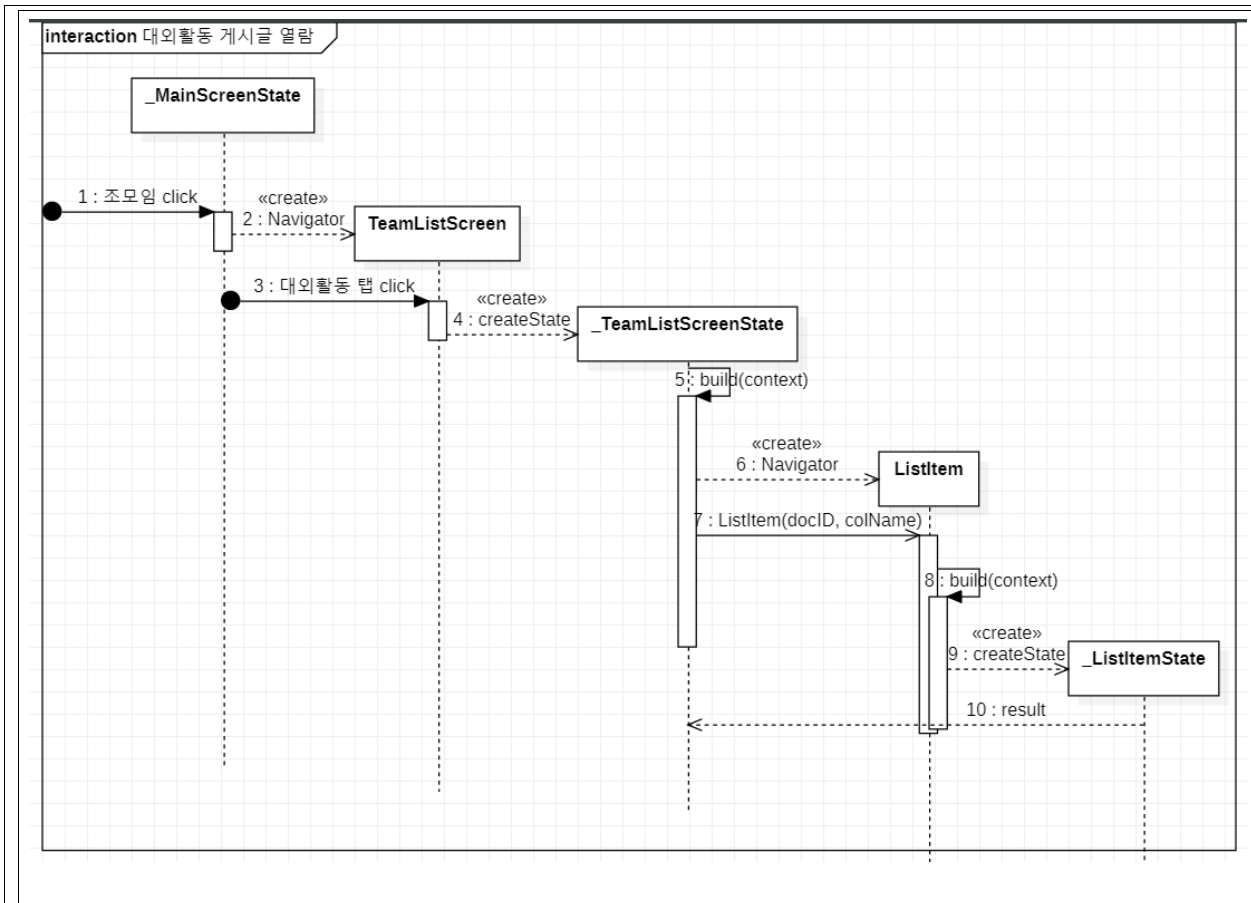
12-1. 조별과제 게시물 열람



기능 및 설명

· _MainScreenState에서 조모임을 click하면 실행되는 시퀀스이다. 화면 전환을 위해 Navigator에 TeamListScreen 인스턴스를 준다. TeamListScreenState는 _TeamListScreenState를 생성하며, _TeamListScreenState는 게시글을 리스트화 하기 위해 ListItem을 사용한다. ListItem은 Google FireBase DataBase System Server에 있는 TeamProject 컬렉션의 정보를 받아 카드형태로 정보를 구성하며, _TeamListScreenState는 구성된 ListItem을 조별과제 탭에 List 형태로 화면을 구성한다.

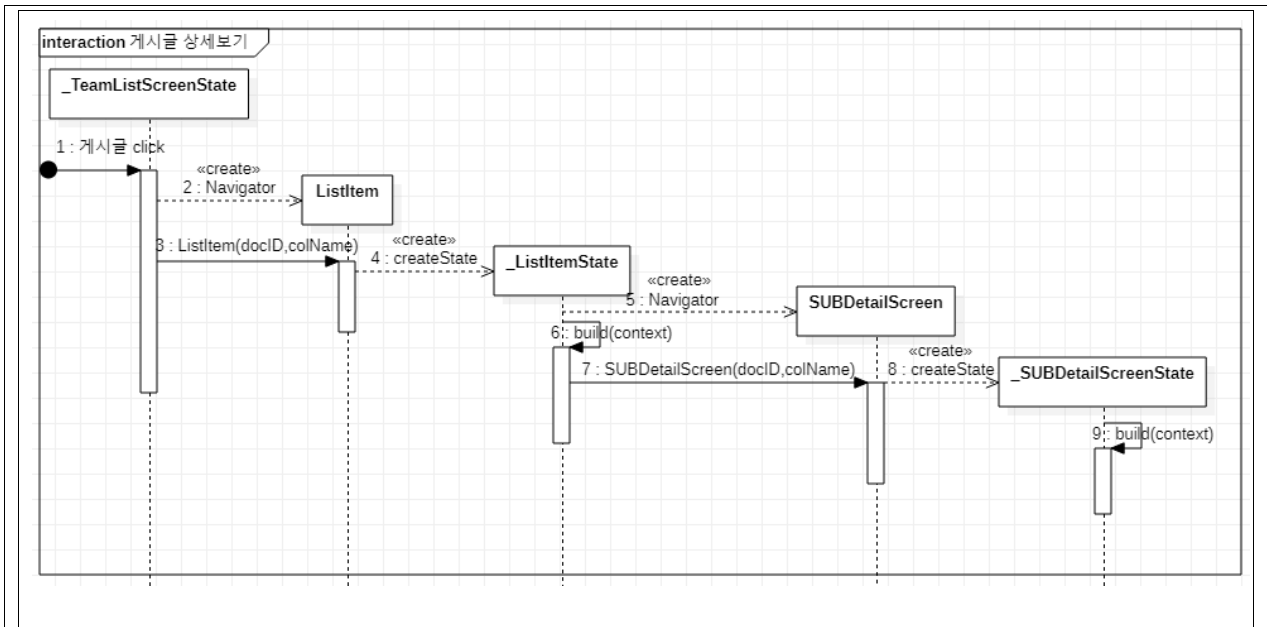
12-2. 대외활동 게시글 열람



기능 및 설명

· _MainScreenState에서 조모임을 click하고, 대외활동 탭을 click하면 실행되는 시퀀스이다. 화면 전환을 위해 Navigator에 TeamListScreen 인스턴스를 준다. TeamListScreen은 _TeamListScreenState를 생성하며, _TeamListScreenState는 게시글을 리스트화 하기 위해 ListItem을 사용한다. ListItem은 Google FireBase DataBase System Server에 있는 GroupProject 컬렉션의 정보를 받아 카드형태로 정보를 구성하며, _TeamListScreenState는 구성된 ListItem을 대외활동 탭에 List 형태로 화면을 구성한다.

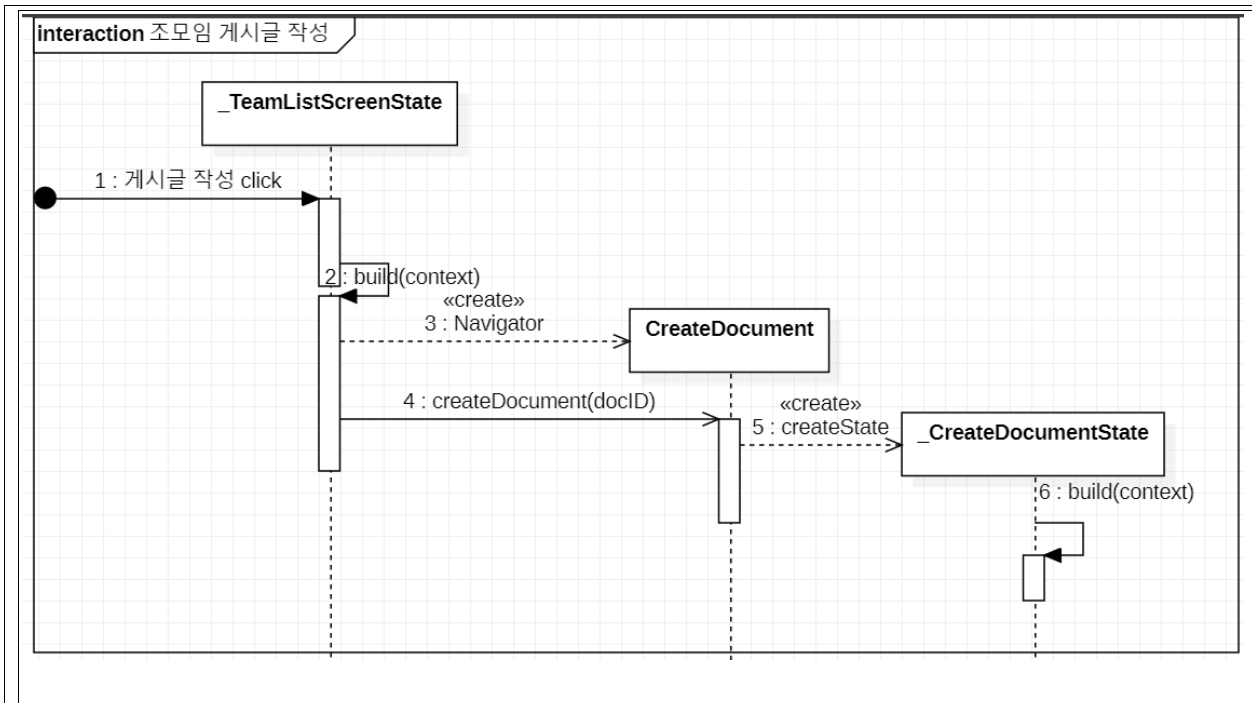
13. 게시글 상세보기



기능 및 설명

· _TeamListScreenState에서 ListItem을 click하면 실행되는 시퀀스이다. 화면 전환을 위해 Navigator에 SUBDetailScreen 인스턴스를 준다. SUBDetailScreenState는 _SUBDetailScreenState를 생성하며, _SUBDetailScreenState는 컬렉션 정보와 Document ID 정보로 Google FireBase DataBase System Server에 있는 게시글의 정보를 받아서 화면을 구성한다.

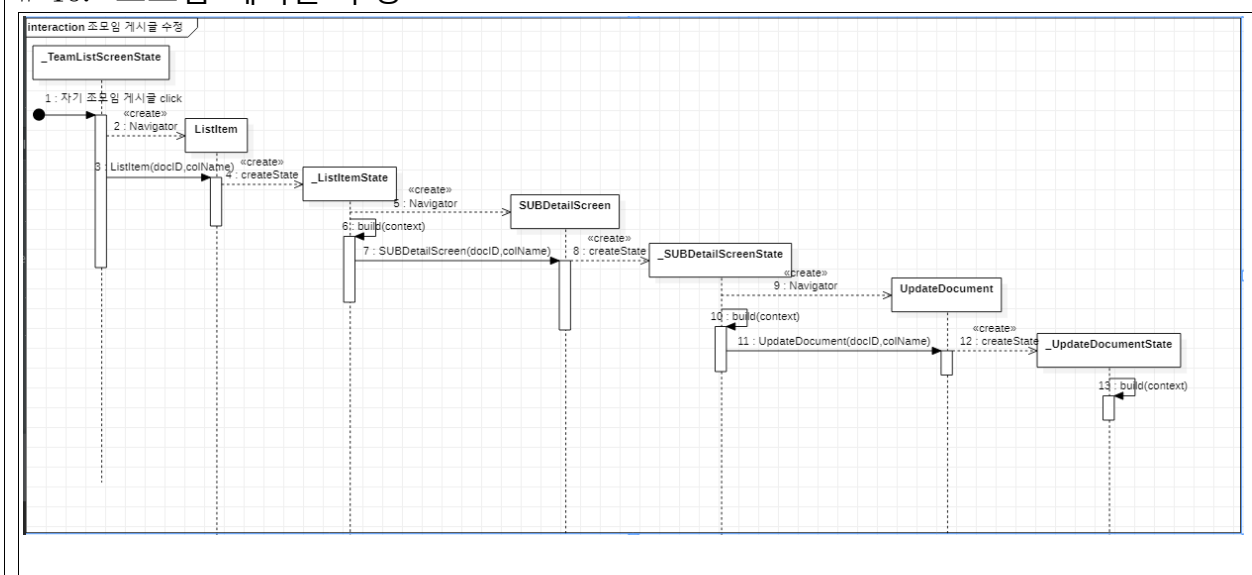
14. 조모임 게시글 작성



기능 및 설명

· _TeamListScreenState의 게시물 작성 버튼을 click하면 실행되는 시퀀스이다. 화면 전환을 위해 Navigator에 CreateDocument 인스턴스를 준다. CreateDocumentState는 _CreateDocumentState를 생성하며, _CreateDocumentState는 게시글을 작성 하기 위해 필요한 정보를 사용자에게 입력받는다. 입력된 정보를 Google FireBase DataBase System Server에 있는 컬렉션에 정보를 저장한다.

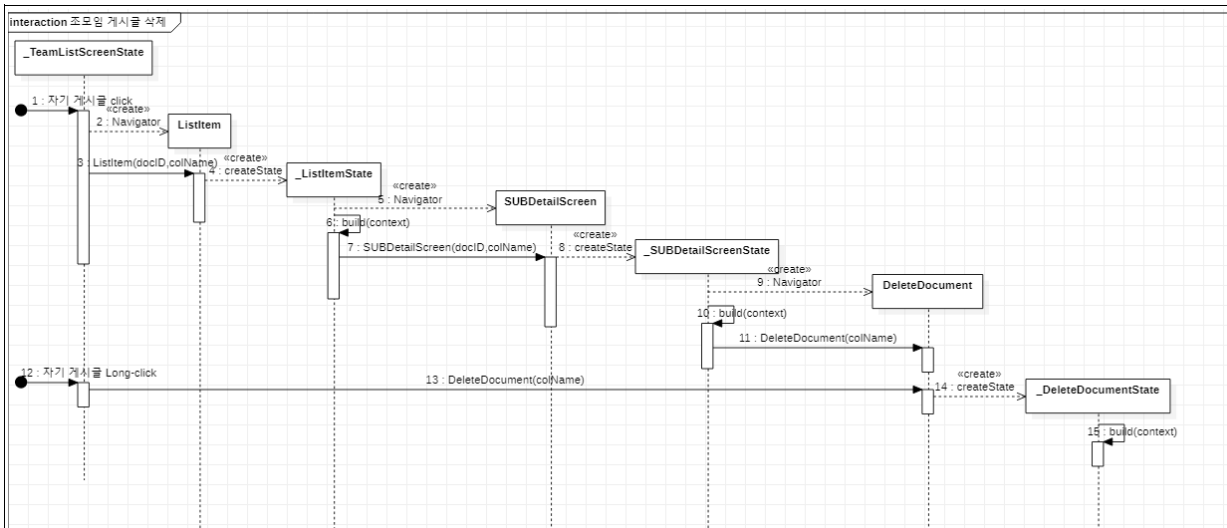
15. 조모임 게시물 수정



기능 및 설명

· `_TeamListScreenState`의 조모임 게시글의 `DetailScreenState`에서 수정 버튼을 click 하면 실행되는 시퀀스이다. 화면 전환을 위해 `Navigator`에 `UpdateDocument` 인스턴스를 준다. `UpdateDocumentState`는 `_UpdateDocumentState`를 생성하며, `_UpdateDocumentState`는 게시글을 수정하기 위해 필요한 정보를 사용자에게 입력받는다. 입력된 정보를 Google FireBase DataBase System Server에 있는 해당 게시글에 갱신한다.

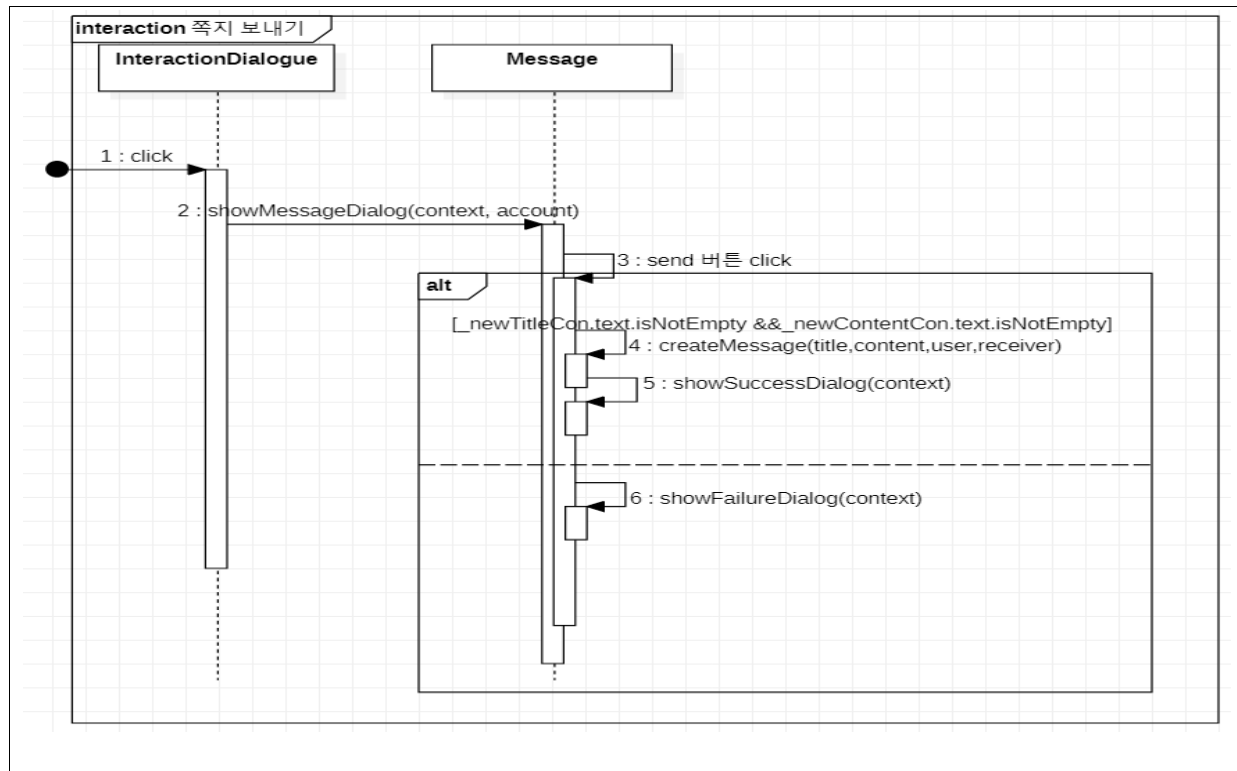
16. 조모임 게시글 삭제



기능 및 설명

· `_TeamListScreenState`의 게시글을 Long-click하면 실행되는 시퀀스이다. 또한 `_TeamListScreenState`의 대외활동 탭의 게시글의 `DetailScreenState`에서 삭제 버튼을 click 하면 실행되는 시퀀스이다. `DeleteDocumentState`는 `_DeleteDocumentState`를 생성하며, `_DeleteDocumentState`는 게시글을 삭제여부를 사용자에게 알림창을 통해 묻는다. 사용자가 삭제를 click 하면 Google FireBase DataBase System Server에 있는 해당 게시글 정보를 삭제한다.

17. 쪽지 보내기

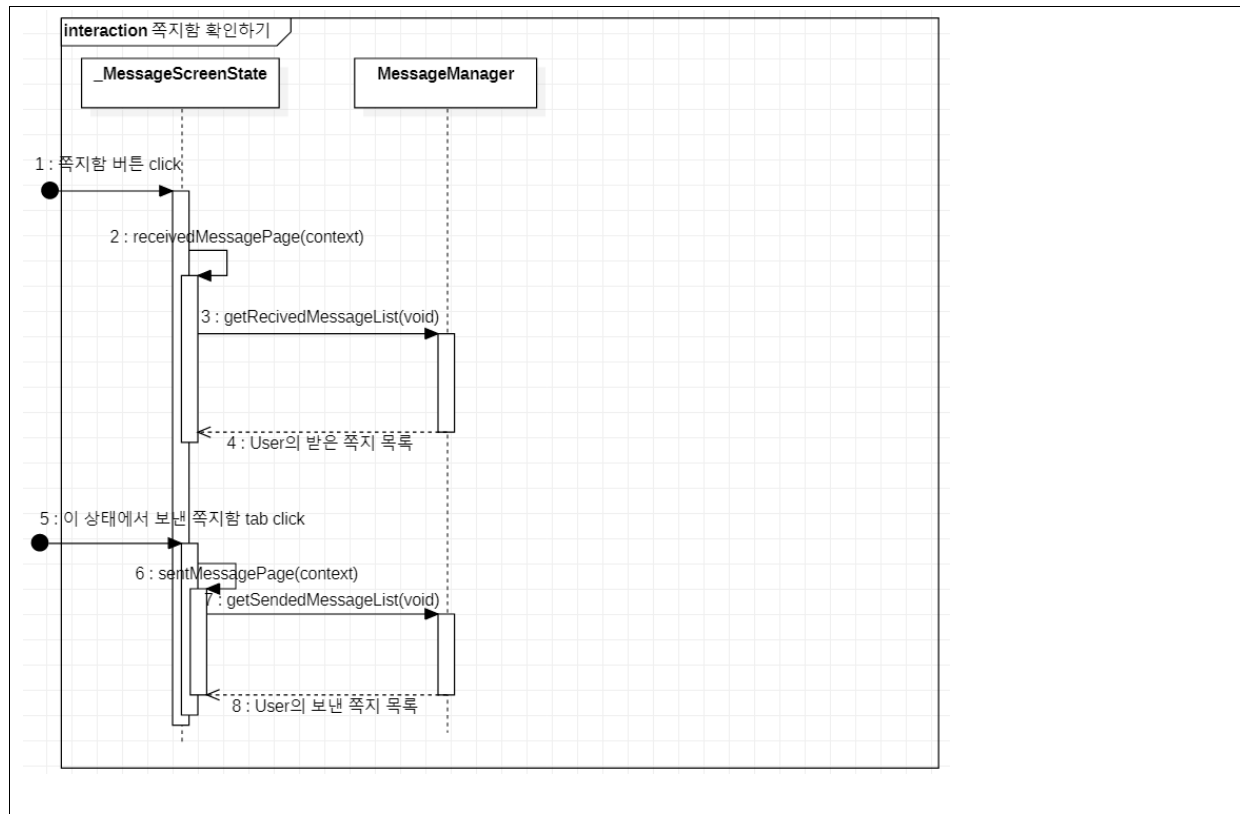


기능 및 설명

- ‘YU, wish A+’에서 쪽지를 보내는 과정의 Use case를 나타낸 Sequence Diagram이다.
- User가 특정한 상황의 click 동작으로 InteractionDialogue 클래스의 showComDialog 메소드를 실행시킨 상태에서 시작한다. 즉 ‘사용자와 소통하기’ dialogue가 띄워져 있는 상태에서 시작하는 것이다.
- dialogue가 띄워져 있는 상태에서 User는 쪽지 아이콘을 click한다. 쪽지 아이콘이 click되면 Message 클래스의 showMessageDialog(context, account) 메소드가 실행된다.
- showMessageDialog 메소드에서는 User에게 쪽지의 제목과 쪽지의 내용을 입력할 수 있는 text field를 제공한다. User는 작성을 완료하고 ‘Send’ 버튼을 click한다.
- ‘Send’ 버튼이 click되면 if문에 진입하여 쪽지의 제목과 내용을 확인한다. 쪽지의 제목과 내용이 비어있지 않으면 if문을 실행한다. if문을 실행하면 createMessage 메소드를 실행하여 Google Firebase Database System에 쪽지를 저장한다. 이후 showSuccessDialog(context) 메소드를 실행하여 쪽지 전송에 성공하였다는 dialogue를 띄워준다.
- 쪽지의 제목과 내용이 비어있으면 else문을 실행한다. else문을 실행하면 showFailureDialog(context) 메소드를 실행하여 쪽지 전송에 실패했다는 dialogue를 띄워준다.

· 쪽지 전송 성공 dialogue와 쪽지 전송 실패 dialogue는 모두 '확인' 버튼을 click하면 원래의 화면으로 돌아오게 하였다. 이는 버튼을 click하면 `Navigator.pop(context)`를 두 번 실행하게 하여 구현하였다.

18. 쪽지함 확인하기



기능 및 설명

· 'YU, wish A+'에서 쪽지함을 확인하는 과정의 Use case를 나타낸 Sequence Diagram이다.
· User가 main screen에서 쪽지함 버튼을 click 한다. 쪽지함의 screen인 message screen은 두 개의 tab의 tab bar를 이용한 screen이다. 처음 들어가면 받은 쪽지함 tab의 화면이 기

본으로 보이게 설정되었다. `_MessageScreenState` 클래스의 `receivedMessagePage(context)` 메소드를 실행하여 받은 쪽지함 tab의 내용을 보여준다.

- `receivedMessagePage`를 실행하면 `MessageManager` 클래스의 `getReceivedMessageList(void)` 메소드를 실행한다. 이를 통해 Google Firebase Database System에서 쪽지의 collection에 접근한다. collection의 목록들 중 쪽지의 receiver가 현재 User인 쪽지를 시간 내림차순으로 읽어온다.

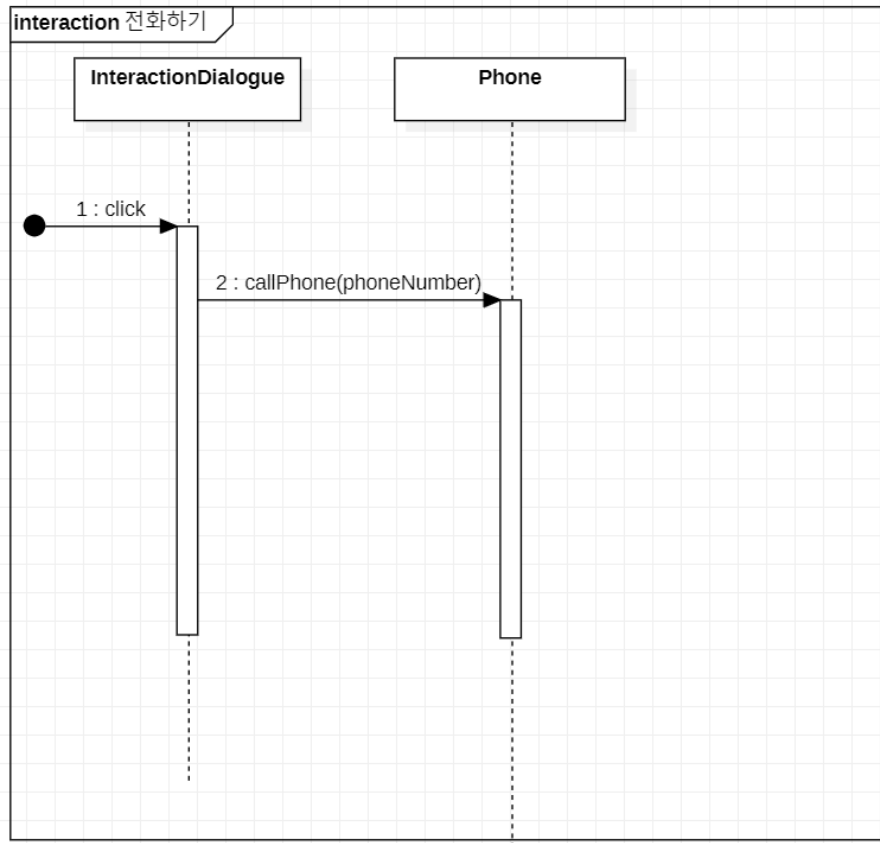
- 받은 쪽지 목록은 `receivedMessagePage(context)`의 메소드에서 적절한 UI로 리스트된다.

- 이 상태에서 보낸 쪽지함 tab을 click하면 `sendMessagePage(context)` 메소드를 실행하여 보낸 쪽지함 tab의 내용을 보여준다.

- `sendMessagePage`를 실행하면 `MessageManager` 클래스의 `getSenddMessageList(void)` 메소드를 실행한다. 이를 통해 Google Firebase Database System에서 쪽지의 collection에 접근한다. collection의 목록들 중 쪽지의 sender가 현재 User인 쪽지를 시간 내림차순으로 읽어온다.

- 보낸 쪽지 목록은 `sendMessagePage(context)`의 메소드에서 적절한 UI로 리스트된다.

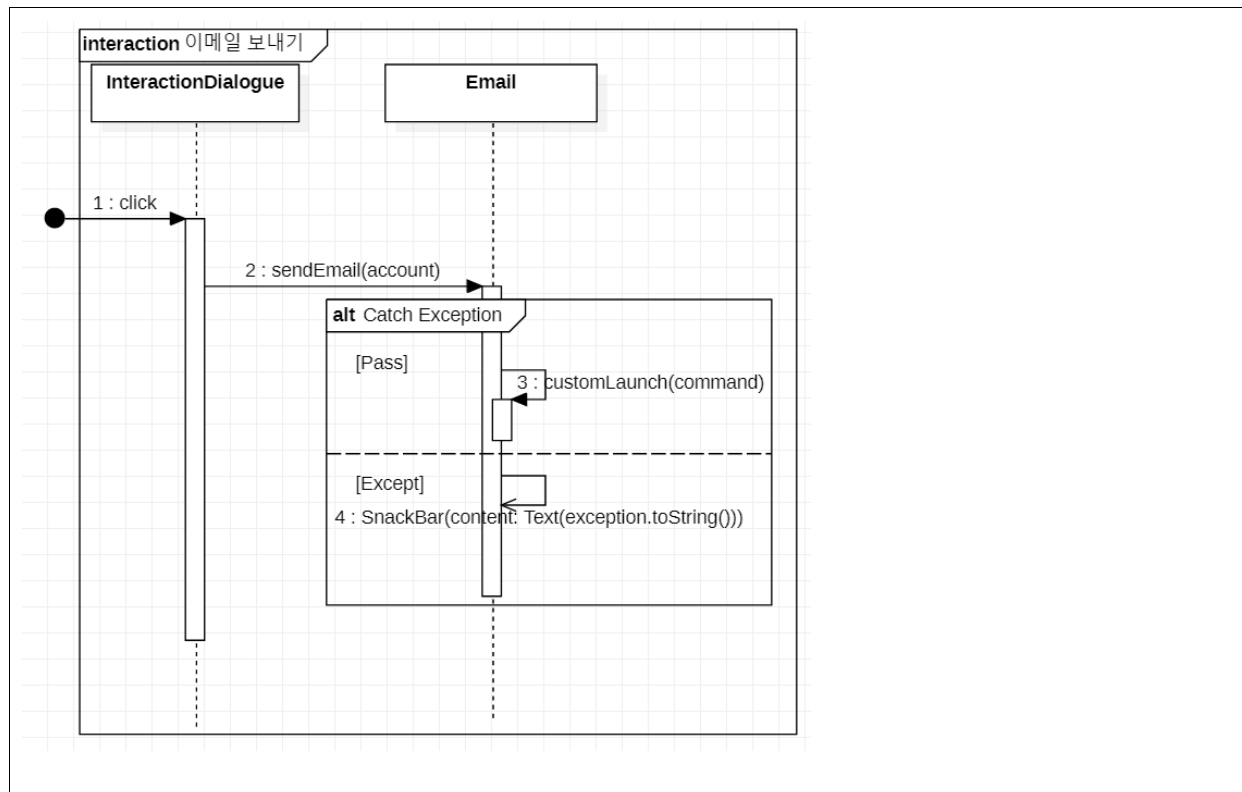
19. 전화하기



기능 및 설명

- ‘YU, wish A+’에서 전화하는 과정의 Use case를 나타낸 Sequence Diagram이다.
- User가 특정한 상황의 click 동작으로 InteractionDialogue 클래스의 showComDialog 메소드를 실행시킨 상태에서 시작한다. 즉 ‘사용자와 소통하기’ dialogue가 띄워져 있는 상태에서 시작하는 것이다.
- dialogue가 띄워져 있는 상태에서 User는 전화 아이콘을 click한다. 아이콘이 click되면 Phone 클래스의 callPhone(phone Number) 메소드가 실행된다. Default Calling Application으로 화면이 전환되고 전화번호 입력란에 상대 User의 전화번호가 입력된다.

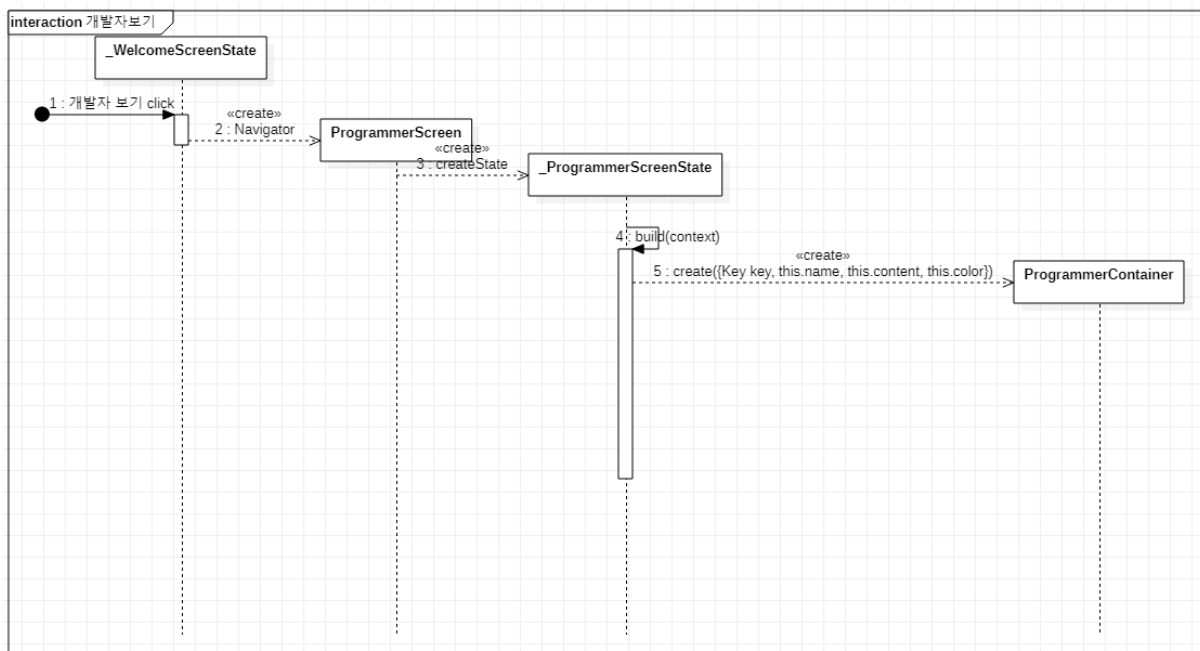
20. 이메일 보내기



기능 및 설명

- ‘YU, wish A+’에서 이메일을 보내는 과정의 Use case를 나타낸 Sequence Diagram이다.
- User가 특정한 상황의 click 동작으로 InteractionDialogue 클래스의 showComDialog 메소드를 실행시킨 상태에서 시작한다. 즉 ‘사용자와 소통하기’ dialogue가 띄워져 있는 상태에서 시작하는 것이다.
- dialogue가 띄워져 있는 상태에서 User는 이메일 아이콘을 click한다. 아이콘이 click되면 Email 클래스의 sendEmail(account) 메소드가 실행된다.
- 이후 Email 클래스의 customLaunch(command) 메소드가 실행된다. 메소드가 정상적으로 실행되면 Default E-mail Application으로 화면이 전환된다. 메일 제목에 ‘YU, wish A+에서 연락드려요.’가 입력된다. 메일 내용은 ‘안녕하세요!’로 입력된다.
- 메소드가 exception을 발생시키면 exception문을 snackbar로 출력한다.

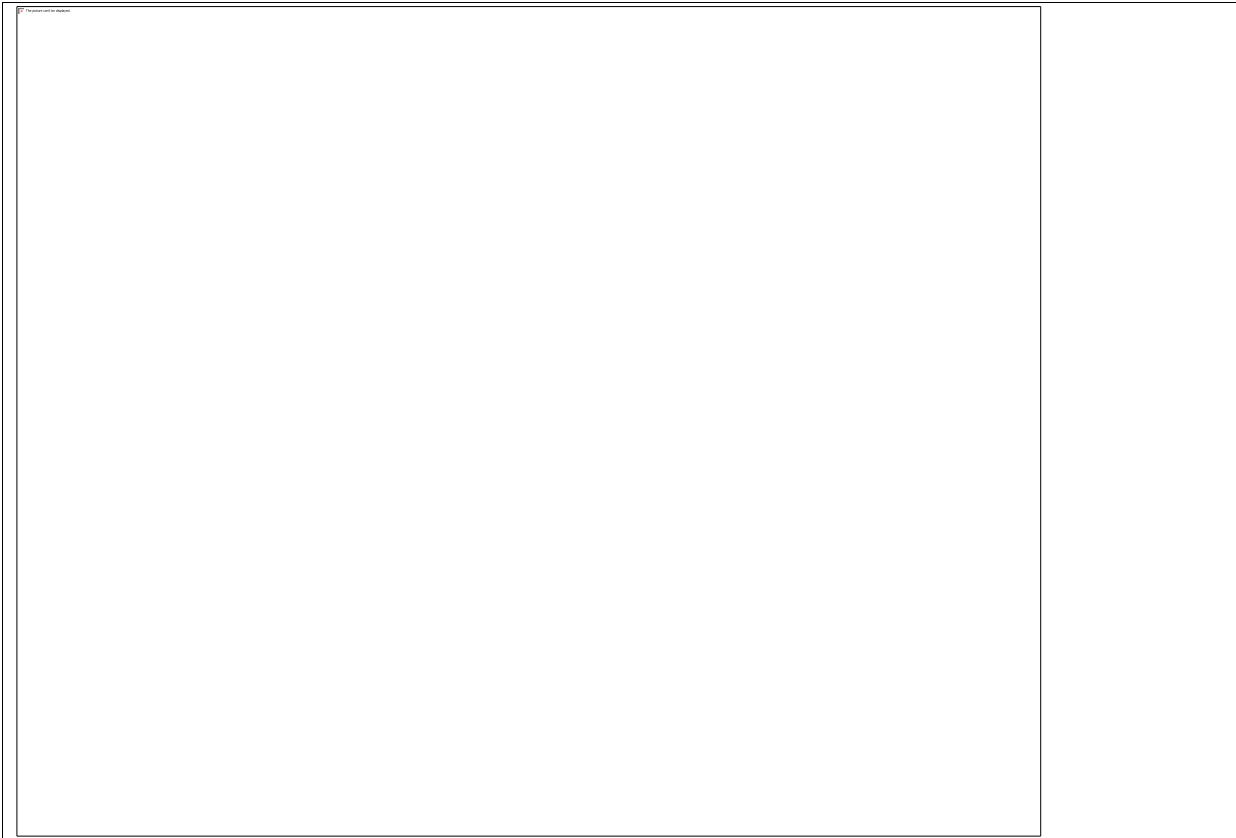
21. 개발자 보기



기능 및 설명

· _WelcomeScreenState가 제공하는 UI에서 개발자 보기 버튼을 click시 실행되는 시퀀스이다. ProgrammerScreen을 생성하고 Navigator로 화면 전환을 시킨다. ProgrammerScreen은 _ProgrammerScreenState를 생성하여 화면을 구성시킨다. _ProgrammerScreenState는 화면을 구성할 때 ProgrammerContainer 위젯을 생성한다. 생성 시 인자를 주어 값을 다르게한다.

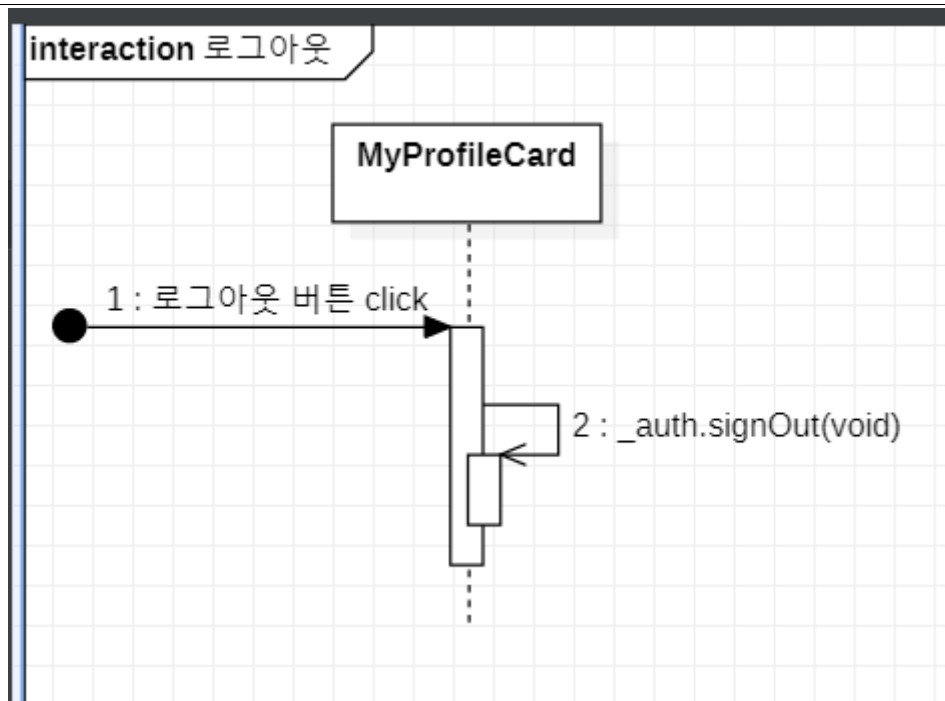
22. 오픈소스 보기



기능 및 설명

- ‘YU, wish A+’에서 쓰인 오픈소스들을 보는 과정의 Use case를 나타낸 Sequence Diagram이다.
- User가 main screen에서 openSource 글씨를 click한다. _OpenSourceScreenState에서 opensource screen에 진입한다.
- screen의 UI 중 오픈소스의 목록들을 list하기 위해 OpenSourceContainer 클래스의 객체를 생성한다. 객체는 각 오픈소스의 이름, 설명을 가지는 박스형 UI이다. 박스를 click하면 그 오픈소스의 공식 document 웹페이지로 이동한다.
- screen에 list 되어야 하는 오픈소스들 각각 다 객체의 생성과 return이 반복된다.
- 완성된 scaffold UI를 return 하여 screen을 나타낸다.

23. 로그아웃

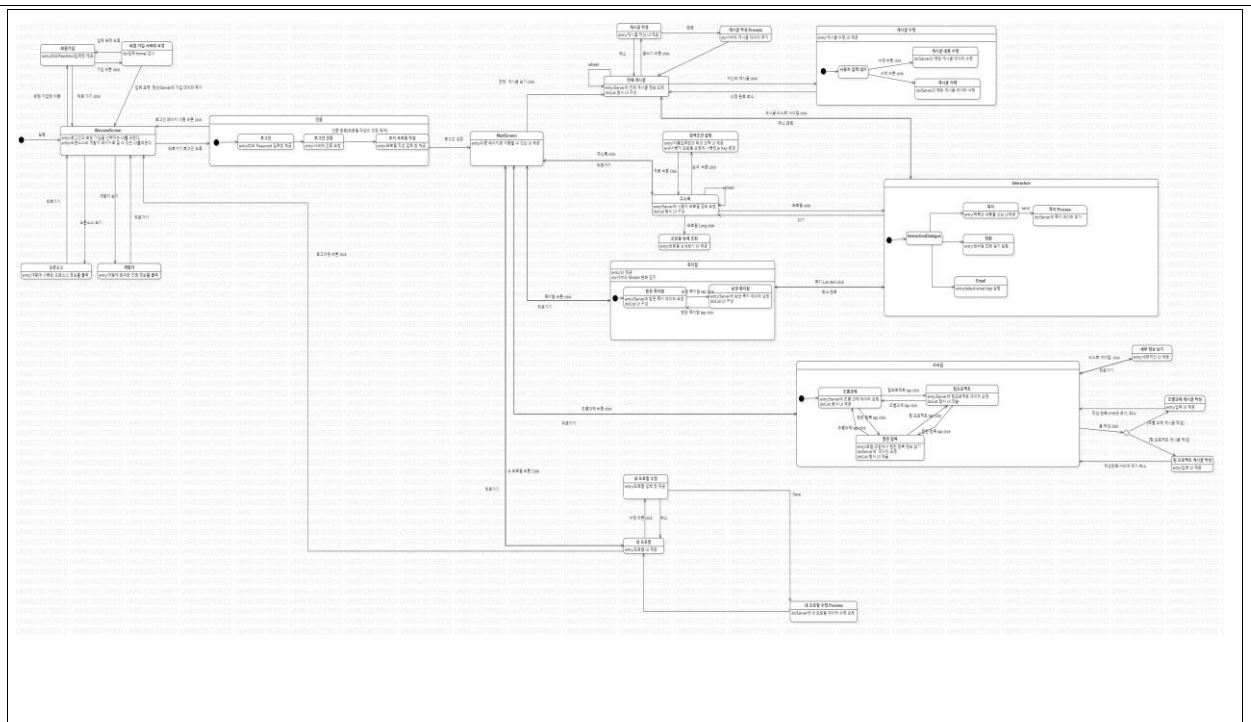


기능 및 설명

· 'YU, wish A+' 어플리케이션 사용을 중단하거나 다른 계정으로 로그인하기 위해 로그아웃하는 과정의 Use case를 나타낸 Sequence Diagram이다.

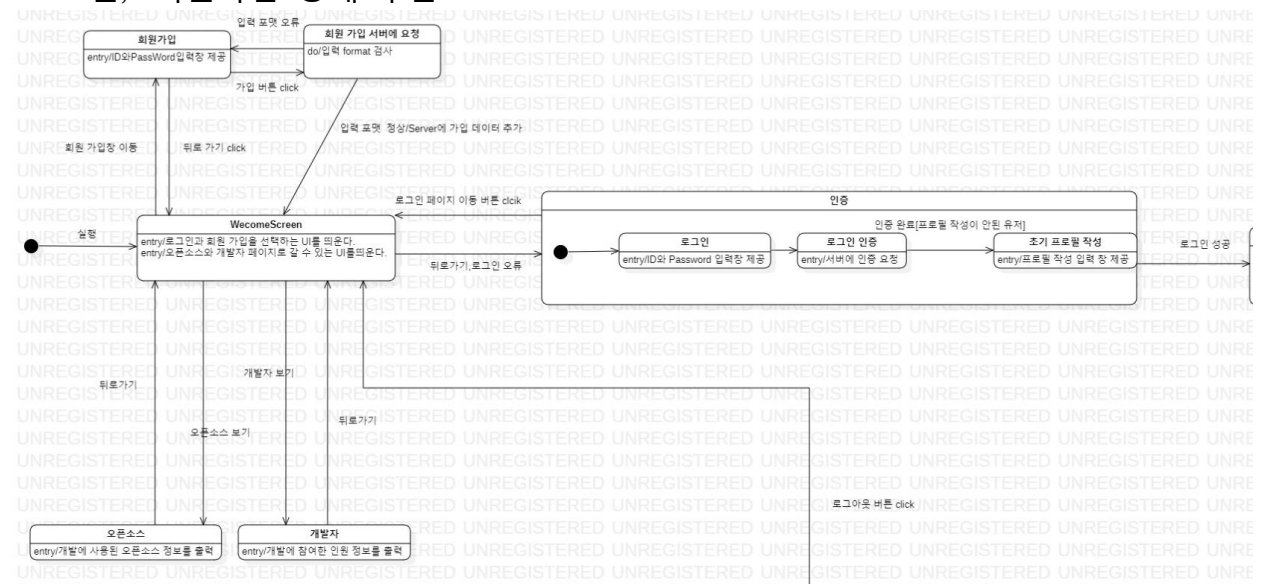
· User가 MyProfileCard 클래스로부터 로그아웃 버튼을 click 하면 _auth를 통해 Google Firebase Auth System에 접근하여 현재 로그인된 계정을 로그아웃한다.

5. State machine diagram



전체적인 state diagram을 보면, 로그인 상태가 가장 먼저 실행되는 것을 볼 수 있다. 로그인 상태에서 성공적으로 인증 성공 시 상대적 오른쪽 State들로 흐름이 넘어가게 된다. 오른쪽에 있는 State들은 MainScreen 상태를 기점으로 흐름이 상호적으로 이동한다. MainScreen에서 크게 5가지의 전이를 볼 수 있다. 전체 게시글, 주소록, 조모임, 쪽지함 그리고 내 프로필 보기 상태이다. 각각들에 대해 대표적인 몇가지들만 밑에 설명하겠다.

<로그인, 회원가입 상태 부분>

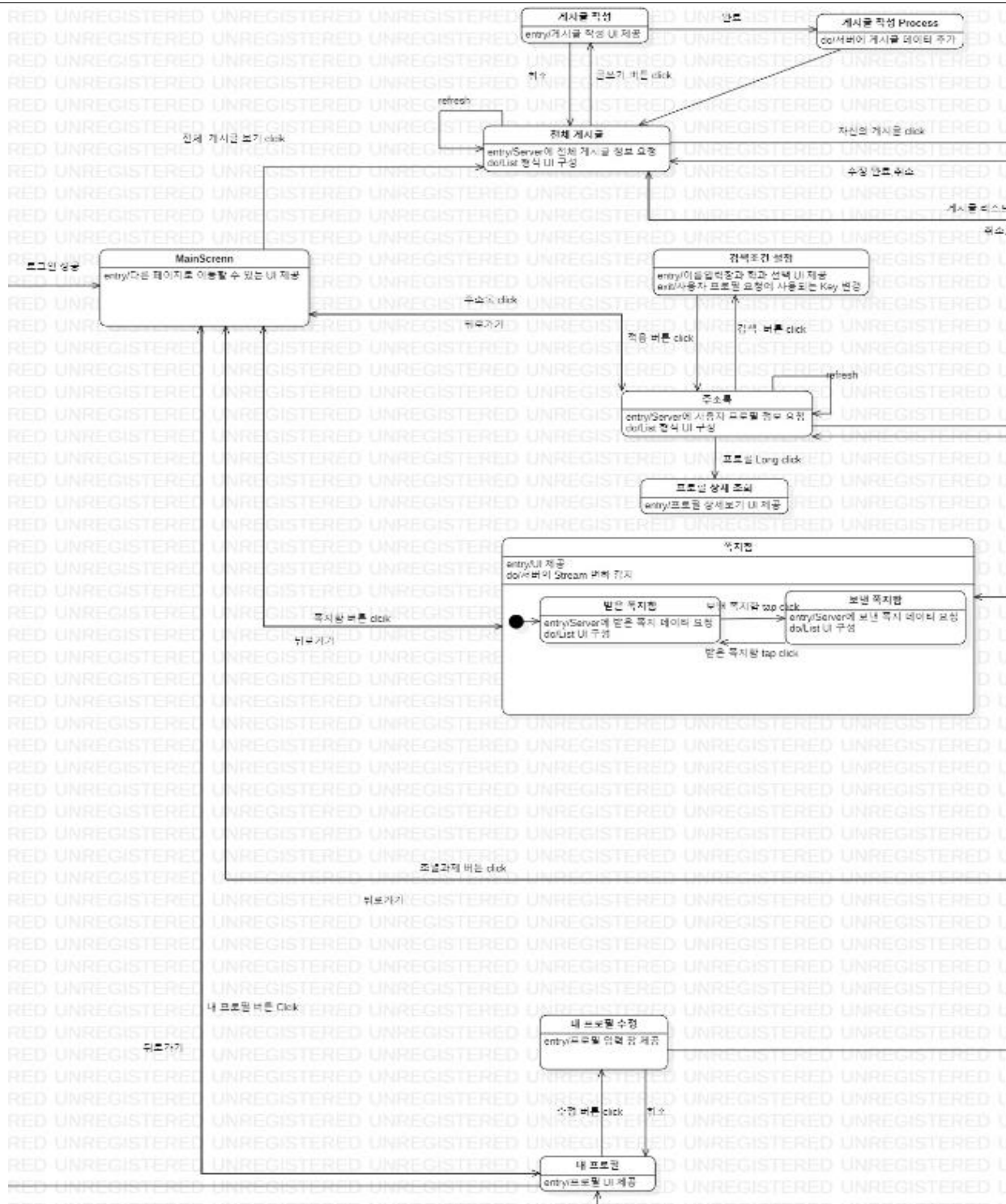


‘YU, wish A+’ 어플리케이션을 처음 실행 한 경우, WelcomeScreen이 실행된다. 기

본적으로 Login 혹은 Register 버튼을 통해 User는 기본에 사용하던 회원 정보를 이용하여 Login하거나 해당 어플리케이션을 사용하기 위해 새로운 E-mail 계정을 생성할 수 있다.

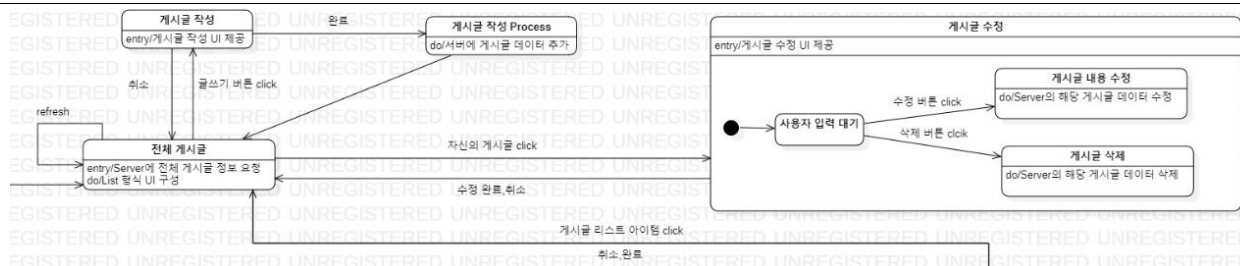
User의 Login과 Register는 정상적인 입력이 이루어진 경우에만 다음 상태로 넘어갈 수 있으며, 그렇지 못한 경우에는 Alert Dialog를 통해 어떤 입력이 잘못된 것인지 User에게 알려준다. 또한 다시 각각의 LoginScreen과 RegisterScreen의 상태로 돌아가는 것을 확인할 수 있을 것이다. 또한 오픈소스, 개발자 페이지를 보여주는 상태로 가는 것도 확인 할 수 있다.

<MainScreen 상태 부분>



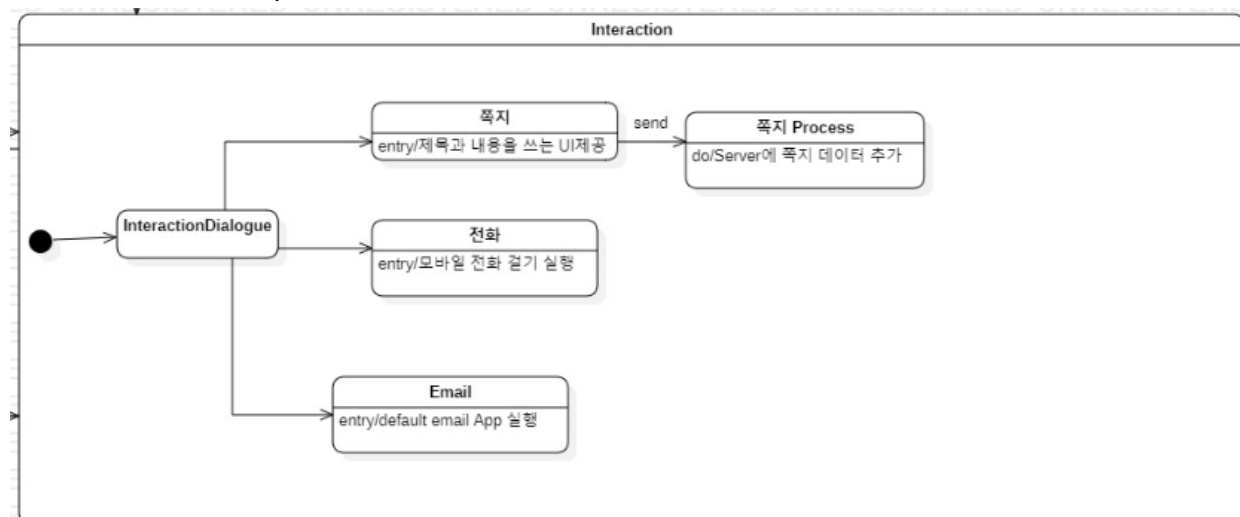
위의 그림을 보면 로그인 성공 시 MainScreen상태로 가는 것을 알 수 있다. MainScreen은 전체 게시물, 주소록, 쪽지함, 조모임, 내 프로필 수정으로 상태가 전환되는 Bridge역할을 하는 것을 볼 수 있다. 실제로 MainScreen상태가 제공하는 UI에서 Button을 클릭 시 다른 상태로 전이되며, UI가 바뀐다.

〈게시글 작성 부분〉



전체 게시글 상태에서는 게시글 작성, 게시글 수정, Interaction 상태로 전이 될 수 있다. 글쓰기 버튼을 click시 게시글 작성 창이 나타난다. 작성 후 완료를 누르면 게시글을 Server에 등록하는 상태가 되었다가 바로 전이되는 것을 볼 수 있다. 게시글 수정의 경우 상태의 흐름이 2가지로 나뉜다. 삭제 처리 상태와 수정 처리 상태로 나뉘어 Activity를 수행 후 전체 게시글 상태로 돌아온다.

<Interaction 상태>



Interaction 상태는 여러 상태에서 전이 될 수 있다. 전화, 쪽지, Email 기능을 사용하기 위해 사용자에게 dialogue를 제공해준다. 사용자의 선택에 따라 쪽지, 전화, Email 상태가 된다. 전화와 Email 같은 경우는 직접 구현하는 것이 아니라 Mobile에 탑재되어있는 애플리케이션으로 넘어간다.

위의 State Diagram에서 표현하지 못한 에러들이 발생할 수 있다. 해당 문제는 Google Firebase Auth, Google Firebase firestore, Google firebase DB의 문제로 발생할 수 있으며, Default E-mail Application과의 충돌 문제로 에러를 발생할 수 있다. 또한 Android의 버전 문제에 따른 에러가 발생할 수 있어, 최소 안드로이드 버전을 22로 권고하는 바이다. 만약 해당 어플리케이션을 안드로이드 휴대폰이 아닌 컴퓨터를 통해 컴파일 하게 되면, 해당 운영체제와 Flutter간의 호환성, Android 에뮬레이터에 의한 버전 문제 또한 발생할 수 있다.

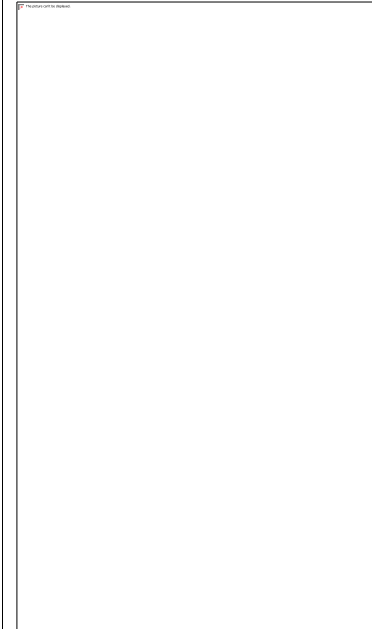
6. User interface prototype

12. 여기서 작성된 모든 프로토타입 화면은 Figma를 통해 작성되었다.

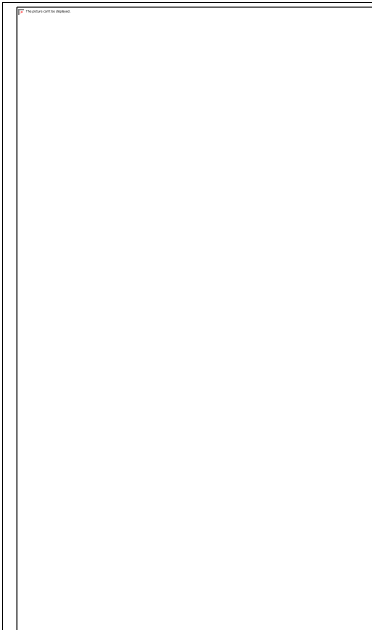
13. <https://www.figma.com/file/75VGM5CdK7IV82HESGQvtF/Untitled?node-id=0%3A1>

여기서 말하는 기본 제공이란, 실제로 해당 화면에서 일어나는 모든 기능을 설명하는 것이 아니라 화면 상에서 보이는 기능들만 언급한 것임을 밝힌다.

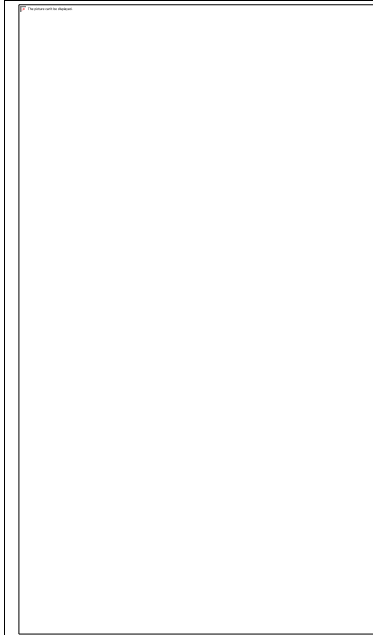
14. Welcome Screen

	<p>‘Yu, wish A+’ 어플리케이션이 실행되고 나서 가장 먼저 사용자에게 보일 화면이다. 가장 기본적인 내용들을 제공한다.</p> <p>15. 기본 제공</p> <p>가. Login 버튼 : Login Screen으로 이동할 수 있는 기능을 제공한다.</p> <p>나. Register 버튼 : Register Screen으로 이동할 수 있는 기능을 제공한다.</p> <p>16. 추가 제공</p> <p>가. 제작자 확인 버튼 : Programmer Screen으로 이동할 수 있는 기능을 제공한다.</p> <p>나. 오픈 소스 확인 버튼 : OpenSource Screen으로 이동할 수 있는 기능을 제공한다.</p>
--	--

17. Login Screen & Register Screen

	<p>Welcome Screen의 Login 버튼을 누르고 나서 사용자에게 보일 화면이다. 혹은 Register 버튼을 누르고 나서 사용자에게 보일 화면이다.</p> <p>18. 기본 제공</p> <p>가. E-mail 입력 텍스트 공간 : 사용자가 login 혹은 register 할 경우 사용할 사용자의 이메일을 입력하는 창이다.</p> <p>나. Password 입력 텍스트 공간 : 사용자가 login 혹은 register할 경우 사용할 사용자의 비밀번호를 입력하는 창이다.</p> <p>다. Login 혹은 Register 버튼 : 사용자가 어플리케이션에 정상적으로 login 혹은 register하여 접속하고 싶은 경우에 사용하는 버튼이다.</p>
---	--

19. Main Screen



Login Screen 혹은 Register Screen에서 정상적으로 접속된 후 사용자에게 보일 화면이다. 기본적인 네비게이션 역할을 제공한다.

20. 기본 제공

가. 게시글 보기 버튼 : Post List Screen으로 이동할 수 있는 기능을 제공한다.

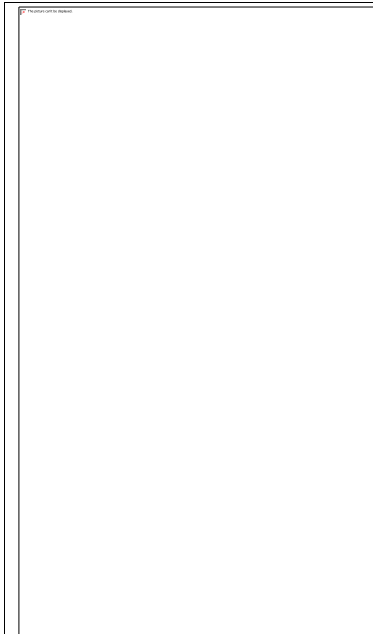
나. 주소록 보기 버튼 : Profile List Screen으로 이동할 수 있는 기능을 제공한다.

다. 조모임 버튼 : Team List Screen으로 이동할 수 있는 기능을 제공한다.

라. 쪽지함 버튼 : Message Screen으로 이동할 수 있는 기능을 제공한다.

마. 내 프로필 버튼 : Profile Screen으로 이동할 수 있는 기능을 제공한다.

21. Post List Screen



Main Screen에서 사용자가 게시글 보기 버튼을 클릭한 경우 사용자에게 보일 화면이다.

22. 기본 제공

가. 글 쓰기 버튼 : 사용자가 자신의 계정으로 게시판에 기본적인 글을 작성할 수 있도록 하는 기능이다.

나. 게시글 보기 : 모든 사용자가 작성한 게시글을 게시판 형태로 표현하여 리스트로 보여주는 기능이다.

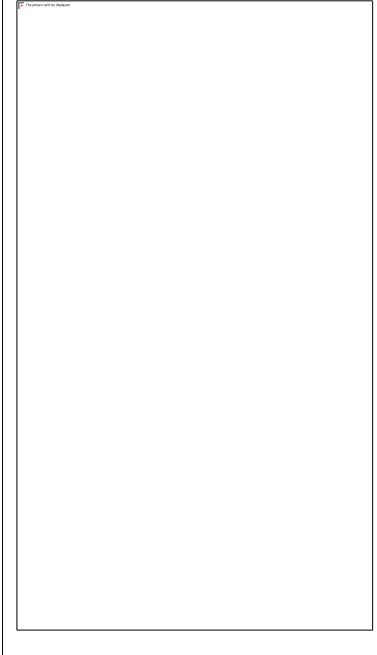
23. Programmer Screen

	<p>Welcome Screen에서 사용자가 제작자 버튼을 클릭할 경우 사용자에게 보일 화면이다.</p> <p>24. 기본 제공</p> <p>가. 개발자 명단 : 어플리케이션을 제작에 참여한 모든 개발자 명단을 제공한다.</p> <p>나. 개발자 명단에는 개인적으로 작성한 추가적인 한 줄 소개를 포함한다.</p>
--	---

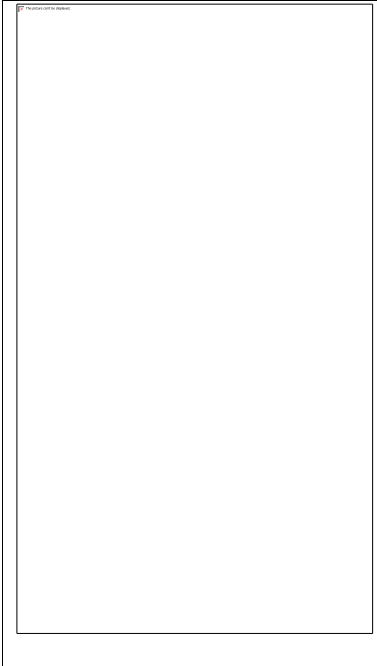
25. Profile Screen

	<p>Main Screen에서 사용자가 내 프로필 버튼을 클릭할 경우 사용자에게 보일 화면이다.</p> <p>26. 기본 제공</p> <p>가. 이메일 : 사용자 계정의 이메일을 보여준다.</p> <p>나. 전화번호 : 사용자 계정의 전화번호를 보여준다.</p> <p>다. 학과 : 사용자 계정의 학과를 보여준다.</p> <p>라. 프로필 수정 버튼 : 사용자 계정의 프로필을 수정할 수 있도록 하는 기능을 제공한다.</p> <p>마. 로그아웃 버튼 : 현재 어플리케이션에 로그인 되어있는 사용자의 계정을 로그아웃하는 기능을 제공한다.</p> <p>바. 프로필 비공개 스위치 : 사용자 계정의 공개 설정을 변경하는 기능을 제공한다.</p>
--	---

27. Opensource Screen

	<p>Welcome Screen에서 사용자가 OpenSource 버튼을 클릭할 경우 사용자에게 보일 화면이다.</p> <p>28. 기본 제공</p> <p>가. 오픈 소스 명단 : 어플리케이션 제작에서 사용된 모든 외부 라이브러리 오픈 소스 명단을 제공한다.</p> <p>나. 추가적으로 해당 오픈 소스의 기능에 대한 짧은 한 줄 소개를 포함한다.</p>
---	---

29. Team List Screen

	<p>Main Screen에서 사용자가 조모임 버튼을 클릭한 경우 사용자에게 보일 화면이다.</p> <p>30. 기본 제공</p> <p>가. 조별 과제 탭 : 소규모 조별과제 게시글 리스트를 볼 수 있는 화면을 제공하는 기능이다.</p> <p>나. 대외 활동 탭 : 학과와 관련이 없는 대외 활동 리스트를 볼 수 있는 화면을 제공하는 기능이다.</p> <p>다. 글 쓰기 버튼 : 해당 탭에 게시글을 작성할 수 있는 버튼이다.</p>
---	--

7. Implementation requirements

해당 소프트웨어는 다음과 같은 환경에서 돌아간다. 만약 Target 프로그램에 대한 설명이 듣고 싶은 경우에는, 가장 아래에 적혀 있는 Connected device의 API나 버전을 확인하기를 바란다.

[✓] Flutter (Channel stable, 1.20.3, on Mac OS X 10.15.7 19H2, locale ko-KR)

- Flutter version 1.20.3
- Framework revision 216dee60c0 (6 weeks ago), 2020-09-01 12:24:47 -0700
- Engine revision dlbc06f032
- Dart version 2.9.2

[✓] Android toolchain - develop for Android devices (Android SDK version 29.0.3)

- Android SDK at /Users/parkjaewoong/Library/Android/sdk
- Platform android-29, build-tools 29.0.3
- Java binary
- Java version OpenJDK Runtime Environment (build 1.8.0_202-release-1483-b49-5587405)
- All Android licenses accepted.

[✓] Xcode - develop for iOS and macOS (Xcode 11.7)

- Xcode at /Applications/Xcode.app/Contents/Developer
- Xcode 11.7, Build version 11E801a
- CocoaPods version 1.9.3

[✓] Android Studio (version 3.5)

- Android Studio
- Flutter plugin version 43.0.1
- Dart plugin version 191.8593
- Java version OpenJDK Runtime Environment (build 1.8.0_202-release-1483-b49-5587405)

[✓] VS Code (version 1.50.0)

- VS Code at /Applications/Visual Studio Code.app/Contents
- Flutter extension version 3.15.0

[✓] Connected device (1 available)

• AOSP on IA Emulator (mobile) • emulator-5554 • android-x86 • Android 9 (API 28) (emulator) or other Emulator

8. Glossary

Term	Description
게시글	사용자가 각 게시판에 작성한 게시글 각각을 의미한다. 사용자는 일반적인 게시판에 게시글을 작성할 수 있으며, 추가적으로 조모임 게시판에서 각각의 게시글을 작성할 수 있다.
프로필 화면	프로필 화면은 해당 사용자 계정의 프로필을 의미한다. 다른 사용자의 프로필은 프로필 화면에서 확인할 수 없으며, Profile List로 게시글의 형태로 확인할 수 있다.
default 이메일 어플리케이션	이메일을 작성하기 위해서는 사용자의 휴대폰에 기본으로 설정된 default 이메일 어플리케이션이 필요하다. 일반적인 android 스마트폰에서는 google mail 어플리케이션이 default 이메일 어플리케이션으로 지정되어 있다.
Figma	피그마란, 웹 기반 벡터 그래픽 편집기 및 프로토타이핑 도구이다. Figma는 실시간 협업에 중점을 두고 사용자 인터페이스 및 사용자 경험 디자인 사용에 중점을 둔 툴이다.
team project	조별 과제 리스트를 제공하는 게시판이다. 다른 게시판과 착각하지 않기 위해 따로 정의해 두었다.
group project	대외 활동 리스트를 제공하는 게시판이다. 다른 게시판과 착각하지 않기 위해 따로 정의해 두었다.
게시판	사용자가 작성한 게시글의 리스트를 게시판라고 부른다. 사용자는 게시판에 게시글을 작성할 수 있다.

9. References

<https://flutter.dev/docs/development/ui/widgets>
<https://api.flutter.dev>
<https://medium.com/flutter>
<https://firebase.google.com>
<https://firebase.flutter.dev>
https://pub.dev/packages/font_awesome_flutter
<https://pub.dev/packages/intl>
https://pub.dev/packages/firebase_core
https://pub.dev/packages/firebase_auth
https://pub.dev/packages/cloud_firestore
https://pub.dev/packages/bottom_navy_bar
https://pub.dev/packages/animated_text_kit
https://pub.dev/packages/flutter_staggered_grid_view
https://pub.dev/packages/url_launcher
https://pub.dev/packages/liquid_pull_to_refresh
https://pub.dev/packages/flip_card
https://pub.dev/packages/modal_progress_hud
<https://staruml.io>
<https://developer.android.com/studio/index.html?hl=ko>

31. Flutter에 대한 기본적인 정보를 제공하는 사이트와 어플리케이션 개발에서 실질적으로 사용하게 될 외부 라이브러리에 대한 정보를 모두 제공하였다.

32. Google Firebase 시스템을 굳이 여러 시스템으로 분리한 이유는 보다 확실하게 기능을 구별하고 싶었기 때문이다. Google Firebase는 다양한 기능들을 제공하는데, 일반적인 데이터베이스로서의 역할로 사용하기도 하지만, 추가적으로 유저 인증을 위한 Auth기능에서 추가적인 보안을 제공하기도 한다. 또한 사진을 저장하는 경우에는 Firebase firestore를 통해 일반적인 데이터가 저장되는 곳이 아닌 다른 공간에 저장하기도 한다. 정확한 정보를 확인하고 싶은 경우에는 <https://firebase.google.com> 를 통해 정보를 얻을 수 있다.