

2. SDS (Software Design Specification)

DGAJ



DGAJ

Student No	Name
21912113	송재민
21912085	김현수
22012114	김나연
22012079	구교영
22010642	홍아람
21912075	박해세

[Revision history]

Revision date	Version #	Description	Author
11/03/2023	1.00	DGAJ - SDS 작성	테크자이언트
11/17/2023	1.10	Class Diagram, Sequence Diagram 수정	테크자이언트
11/28/2023	1.20	Use Case Analysis, User Interface Prototype 수정	테크자이언트
12/05/2023	1.30	DGAJ - SDS 전체 수정	테크자이언트

= Contents =

1. Introduction	1
2. Use case analysis	2
3. Class diagram	21
4. Sequence diagram	70
5. State machine diagram	103
6. User interface prototype	106
7. Implementation requirements	133
8. Glossary	134
9. References	135

= Authors for each section =

Introduction - 구교영

Use case analysis - 구교영

Class diagram - 김나연, 박해세

Sequence diagram - 홍아랑, 송재민

State machine diagram - 김현수

User interface prototype - 김현수

Implementation requirements - 김현수

Glossary - 김현수

References - 구교영, 김나연, 박해세, 홍아랑, 송재민, 김현수

1. Introduction

본 문서는 테크자이언트¹⁾가 개발하고자 하는 시스템인 DGAJ의 design specification(S DS)이다. 이 문서는 기존의 SRS에서 정의한 요구사항을 기능적으로 구분하여 효과적으로 구현하기 위해 시스템을 여러 관점에서 설계한다.

DGAJ는 현대 사회에서 발생하는 여러 문제에 도전하고, 해결책을 제공하는 혁신적인 안드로이드 애플리케이션이다. 이 앱은 사용자들이 별도의 광고 없이 진정으로 원하는 음식점을 찾을 수 있도록 돕는 동시에, 소상공인들에게 맛과 서비스를 통한 공정한 경쟁 기회를 제공한다. 또한, 음식점에서 편리한 대기 서비스를 제공하고 남은 재료들을 기부할 수 있는 기능을 통해 사회·경제적 가치 창출을 기대할 수 있다. 이러한 기대는 현대 사회에서 다양한 관계자들에게 많은 이점으로 작용할 것이 예상된다.

본 문서에서는 Use case analysis, Class diagram, Sequence diagram, State machine diagram, User interface prototype을 이용하여 DGAJ의 기능과 특징을 효과적으로 설명한다. Use case analysis는 사용자 관점에서 소프트웨어가 제공하는 기능을 서술하는 단계이고, Use case diagram은 사용자와 사용자가 사용할 Use case 간의 상호작용을 보여주는 자료이다. DGAJ는 고객과 사업자로 두 가지의 Actor가 존재한다. Class diagram은 DGAJ의 클래스와 속성, 동작 방식 등 구조적 양상을 표현하는 자료이다. Sequence diagram은 각 기능의 동작 순서와 상호작용하는 객체 간의 관계를 표현하고, State machine diagram 특정 기능 상태에 따라 변경되는 모습을 도식화한 자료이다. User interface는 사용자의 인터페이스 관점으로 DGAJ를 서술한 것이다.

본 문서는 DGAJ의 개발자를 비롯한 모든 Stakeholder를 위한 문서이며, DGAJ 시스템의 전반적인 구조와 설계 과정에 대한 이해도를 높이는 것에 도움을 준다.

1) 본 프로젝트를 진행하는 개발팀의 명칭

2. Use case analysis

Use case analysis에서는 use case diagram과 use case description을 통해 시스템의 기능을 상세하게 설명한다.

- Use case diagram의 기능을 수행하는 Actor는 고객(Customer)과 사업자(Owner)로 분류된다. 고객은 가게 추천, 가게 줄서기 기능 등을 수행할 수 있고 사업자는 DGAJ를 통해 가게에서 제공하는 기능을 관리할 수 있다.
- DGAJ의 모든 기능은 로그인해야 이용할 수 있다. (단, 회원가입 기능 제외)
- Manage 로 시작하는 use case는 여러 기능을 합친 것이며, use case 내의 여러 기능은 use case description 고려사항에서 자세히 설명한다.

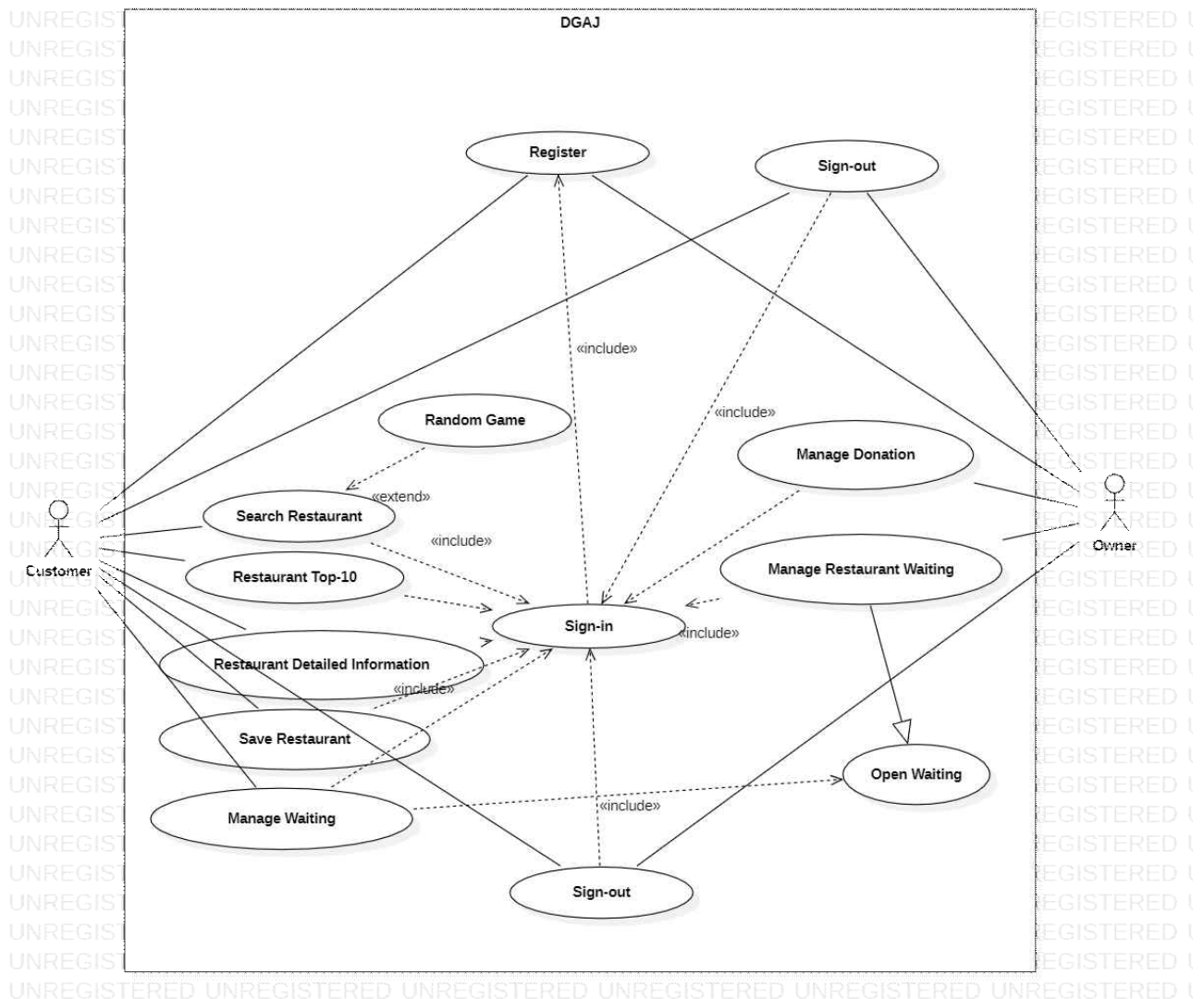


그림 3 Use case diagram

Use case #1 : Register	
GENERAL CHARACTERISTICS	
Summary	DGAJ의 모든 사용자는 시스템을 이용하기 위해서 회원가입을 통해 계정을 등록해야 한다.
Scope	DGAJ(Daegu Go And Joy)
Level	Customer, Owner
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer, Owner
Preconditions	사용자는 안드로이드 앱에서 DGAJ 앱을 설치한 상태이다.
Trigger	사용자가 첫 화면에서 회원가입 버튼을 누를 때
Success Post Condition	DB(Firestore)에 사용자의 계정이 등록되고 사용자는 로그인을 진행할 수 있다.
Failed Post Condition	사용자는 로그인을 진행할 수 없고 시스템을 사용할 수 없다.
MAIN SUCCESS SCENARIO	
Step	Action
S	사용자가 회원가입을 한다.
1	사용자가 로그인 화면에서 회원가입 버튼을 누른다.
2	회원가입 화면에서 사용자는 사용자 타입, 아이디, 비밀번호, 전화번호를 입력한다.
3	사업자 전용 회원가입에서는 가게 고유번호를 입력하여 사업장을 등록한다.
4	사용자는 아이디 체크 버튼으로 입력한 아이디가 중복되지 않은 아이디인지 확인한 후, 등록 버튼을 누른다.
5	시스템은 회원가입이 성공한 것인지 판단한다.
6	이 Use case는 회원가입이 성공하면 끝난다.
EXTENSION SCENARIOS	
Step	Branching Action
1	1a. 고객 계정으로 회원가입을 하려고 한다. ...1a1. 화면에서 고객과 사업자 계정 중 하나를 선택하는 버튼을 보여준다. ...1a2. 고객 버튼을 누른다. 1b. 사업자 계정으로 회원가입을 하려고 한다. ...1b1. 화면에서 고객과 사업자 계정 중 하나를 선택하는 버튼을 보여준다. ...1b2. 사업자 버튼을 누른다.
3	3a. 입력한 가게와 이미 연동된 계정이 있다. ...3a1. 입력한 고유번호가 이미 사용되는 가게라는 메시지를 Toast창으로 보여준다. ...3a2. 고유번호를 다시 입력하는 단계로 돌아간다. (Use case #1-3)
4	4a. 입력한 아이디가 이미 존재하는 아이디이다. ...4a1. 입력한 아이디가 중복된 아이디라는 메시지를 Toast창으로 보여준다.

	...4a2. 아이디를 다시 입력하는 단계로 돌아간다. (Use case #1-2)
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	사용자당 1회
<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #2 : Sign-in	
GENERAL CHARACTERISTICS	
Summary	DGAJ의 모든 사용자는 시스템을 사용하기에 앞서 로그인을 해야 한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer, Owner
Preconditions	사용자는 회원가입을 모두 마친 상태이다.
Trigger	로딩 화면 이후 로그인 화면이 출력될 때
Success Post Condition	사용자는 고객과 사업자로 나뉘어 각각의 시스템 기능을 이용할 수 있다.
Failed Post Condition	사용자는 로그인에 실패하여 DGAJ 앱의 기능을 사용할 수 없다.
MAIN SUCCESS SCENARIO	
Step	Action
S	사용자가 로그인을 한다.
1	사용자가 로그인 화면에서 아이디와 비밀번호를 입력한다.
2	로그인 정보를 모두 입력한 후, 로그인 버튼을 누른다.
3	시스템은 로그인이 성공한지 판단한다.
4	이 Use case는 로그인이 성공하면 끝난다.
EXTENSION SCENARIOS	
Step	Branching Action
2	2a. 입력한 아이디나 비밀번호의 오류로 로그인에 실패한다. ...2a1. 로그인 실패 메시지를 Toast창으로 보여준다. ...2a2. 아이디와 비밀번호를 다시 입력하는 단계로 돌아간다. (Use case #2-1)
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	시스템을 새로 시작할 때마다
<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #3 : Search restaurant
GENERAL CHARACTERISTICS

Summary	이 use case는 고객을 위한 기능으로 음식점을 검색하기 위한 검색어를 결정한다. 검색어는 메뉴 검색어와 위치 검색어로 나뉜다. 메뉴 검색어는 고객이 원하는 메뉴를 설정하거나 Use Case #4 Random Game을 통해 결정되고 위치 검색어는 GPS를 활용한 현재 위치나 고객이 설정한 위치로 결정된다. API를 사용하여 설정된 검색어를 조회한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer
Preconditions	고객은 고객 유형으로 로그인을 마친 상태이다.
Trigger	고객 유형의 아이디로 로그인을 완료하고 음식점 검색 기능을 이용하려 할 때
Success Post Condition	설정된 검색어에 부합하는 음식점을 API를 통해 검색한다.
Failed Post Condition	음식점 검색에 실패하고 화면이 전환되지 않는다.

MAIN SUCCESS SCENARIO

Step	Action
S	고객이 원하는 음식 메뉴 및 위치에 적합한 음식점을 검색한다.
1	고객이 음식점을 검색하는 화면에서 위치 검색어를 입력한다.
2	고객이 음식점을 검색하는 화면에서 메뉴 검색어를 입력한다.
3	고객은 두 가지의 검색어를 모두 입력하고 검색 버튼을 누른다.
4	시스템은 API를 통해 설정한 검색어에 부합하는 음식점을 조회한다.
5	시스템은 음식점 검색이 성공하였는지 판단한다.
6	이 Use case는 음식점 검색에 성공하면 끝난다.

EXTENSION SCENARIOS

Step	Branching Action
1	1a. 위치 검색어 직접 입력 ...1a1. 고객이 원하는 위치를 설정하기 위해 검색창에 정보를 직접 입력한다. ...1a2. 시스템은 입력된 검색어를 화면의 검색창에 보여준다.
1	1b 위치 검색어 자동 입력 ...1b1. 위치 검색어 우측의 현재 위치 버튼을 누른다. ...1b2. 고객의 현재 위치를 GPS로 파악하여 위치 검색어에 입력한다. ...1b3. 입력된 검색어를 화면의 검색창에 보여준다.

2	<p>2a. 메뉴 검색어 직접 입력</p> <p>...2a1. 고객이 원하는 메뉴를 설정하기 위해 검색창에 정보를 직접 입력한다.</p> <p>...2a2. 시스템은 입력된 검색어를 화면의 검색창에 보여준다.</p>
2	<p>2b 메뉴 랜덤 기능으로 메뉴 검색어 자동 설정</p> <p>...2b1. 고객은 화면에서 랜덤 버튼을 누른다.</p> <p>...2b2. 시스템은 설정된 메뉴 중에서 하나를 랜덤으로 선택하여 검색어에 입력한다.</p> <p>...2b3. 시스템은 입력된 검색어를 화면의 검색창에 보여준다.</p>
2	<p>2c 메뉴 고르기 게임 기능으로 메뉴 정하기</p> <p>...2c1. 고객이 화면에서 메뉴 정하기 버튼을 누른다.</p> <p>...2c2. 게임 화면으로 전환된 후, 고객은 룰렛 돌리기와 제비뽑기 중 하나의 게임을 선택해 해당 버튼을 누른다.</p> <p>...2c3. 고객은 게임을 통해 메뉴를 정한다.</p> <p>...2c4. 고객이 홈 버튼을 누르면 시스템은 음식점 검색 화면으로 이동한다.</p>
3	<p>3a. 위치 검색어나 메뉴 검색어 공백 처리</p> <p>...3a1. 시스템은 공백인 위치 검색어를 GPS로 파악한 고객의 현재 위치 정보로 대체한다.</p> <p>...3a2. 시스템은 공백인 메뉴 검색어를 모든 메뉴에 대한 검색으로 대체한다.</p>
RELATED INFORMATION	
Performance	≤ 1 seconds
Frequency	고객당 하루 평균 5회 이상
<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #4 : Random game	
GENERAL CHARACTERISTICS	
Summary	이 use case는 고객이 메뉴 검색어를 입력하기에 앞서 메뉴 선정에 도움을 주는 기능을 설명한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer
Preconditions	고객은 고객 유형으로 로그인을 마치고, 음식점 검색 화면이 제공되었다.
Trigger	고객이 메뉴 선정을 위한 게임을 실행하려고 할 때
Success Post Condition	게임이 실행되고 고객은 게임을 통해 메뉴를 결정한다.
Failed Post Condition	메뉴 정하기 게임 화면이 출력되지 않는다.
MAIN SUCCESS SCENARIO	
Step	Action
S	고객이 메뉴를 결정하기 위해 게임을 실행한다.
1	고객이 음식점을 검색하는 화면에서 메뉴 정하기 버튼을 누른다.
2	고객이 제비뽑기 버튼을 누른다.
3	고객이 6가지 후보 메뉴를 입력하고 게임하기 버튼을 누른다.
4	시스템은 6가지 중 한 가지 메뉴를 무작위 출력한다.
5	고객이 랜덤돌리기 버튼을 누른다.
6	고객이 6가지 후보 메뉴를 입력하고 게임하기 버튼을 누른다.
7	시스템은 6가지 중 한 가지 메뉴를 무작위 출력한다.
EXTENSION SCENARIOS	
Step	Branching Action
RELATED INFORMATION	
Performance	≤ 1 seconds
Frequency	고객당 하루 평균 3회 이상
<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #5 : Restaurant Top-10	
GENERAL CHARACTERISTICS	
Summary	Use case #3. Search Restaurant에서 호출되는 Use case이다. 시스템은 고객이 검색한 위치 검색어에 기반하여 가게 별 찜 개수가 많은 순서대로 가게를 검색하여 목록으로 출력한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer
Preconditions	고객은 음식점 검색 화면에서 검색어를 설정하여 Top 10 버튼을 누른 상태여야 한다.
Trigger	고객이 위치 검색어를 설정하여 검색한 음식점의 결과를 확인하려 할 때
Success Post Condition	화면을 통해 검색된 음식점을 확인할 수 있다.
Failed Post Condition	음식점 검색에 실패하여 올바른 탑 리스트 음식점 정보 확인이 불가능하다.
MAIN SUCCESS SCENARIO	
Step	Action
S	고객이 검색한 음식점의 결과를 확인한다.
1	시스템은 위치 검색어에 부합하는 가게를 검색하여 찜 개수가 많은 순서대로 화면에 제공한다.
2	시스템은 목록으로 각 가게의 이름과, 주소, 전화번호 등 간단한 정보를 고객에게 제공한다.
3	시스템은 Use case #6 Restaurant Detailed Information 기능으로 해당 음식점의 상세정보 페이지를 제공한다. 각 가게를 누르면 해당 가게의 상세정보를 확인할 수 있다.
EXTENSION SCENARIOS	
Step	Branching Action
3	3a. 가게 상세 정보 확인 ...3a1. 고객이 음식점을 선택한다. ...3a2. 시스템은 선택된 음식점의 상세정보를 제공한다.
RELATED INFORMATION	
Performance	≤ 3 seconds
Frequency	고객당 하루 평균 5회 이상
<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #6 : Restaurant Screen	
GENERAL CHARACTERISTICS	
Summary	Use case #3. Search Restaurant에서 호출되는 Use case이다. 시스템은 고객이 검색한 음식점 리스트를 화면에 출력하고, 해당 음식점의 위치를 지도를 통해 나타낸다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer
Preconditions	고객은 음식점 검색 화면에서 검색어를 설정하여 검색 버튼을 누른 상태여야 한다.
Trigger	고객이 위치 검색어와 메뉴 검색어를 설정하여 검색한 음식점의 결과를 확인하려 할 때
Success Post Condition	화면을 통해 검색된 음식점을 확인할 수 있다.
Failed Post Condition	음식점 검색에 실패하여 올바른 음식점 정보 확인이 불가능하다.
MAIN SUCCESS SCENARIO	
Step	Action
S	고객이 검색한 음식점의 결과를 확인한다.
1	시스템은 검색어에 부합한 음식점을 화면에 제공한다. (오늘의 가게 추천)
2	시스템은 현재 나의 위치를 화면 상단의 지도에 표시하여 제공한다.
3	시스템은 Use case #7 Restaurant Detailed Information 기능으로 해당 음식점의 상세정보 페이지를 제공한다.
EXTENSION SCENARIOS	
Step	Branching Action
1	1a. 위치 검색어 변경 ...1a1. 고객이 화면 상단 우측의 버튼을 누른다. ...1a2. 시스템은 현재 사용자의 위치를 업데이트하여 검색창에 입력한다. ...1a3. 시스템은 새로운 위치 검색어에 적합한 결과를 화면에 제공한다.
3	3a. 가게 상세 정보 확인 ...3a1. 고객이 음식점을 선택한다. ...3a2. 시스템은 선택된 음식점의 상세정보를 제공한다.
RELATED INFORMATION	
Performance	≤ 3 seconds
Frequency	고객당 하루 평균 5회 이상

<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #7 : Save Restaurant	
GENERAL CHARACTERISTICS	
Summary	이 Use case는 고객이 선택한 가게를 저장 목록에 추가, 조회, 삭제하는 기능을 설명한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 30
Status	Analysis
Primary Actor	Customer
Preconditions	저장목록에 저장할 가게의 상세 정보 페이지가 열린 상태이거나 나의 저장목록 화면이 열린 상태여야 한다.
Trigger	고객이 인터페이스에서 저장하기와 관련된 기능을 하려고 할 때
Success Post Condition	선택된 가게를 저장목록에 추가한다. 저장목록을 조회한다. 저장목록에서 특정 가게를 삭제한다.
Failed Post Condition	저장목록에 대한 변경 사항이 반영되지 않는다.
MAIN SUCCESS SCENARIO	
Step	Action
S	고객이 저장하기와 관련된 기능을 수행하려 한다.
1	고객은 가게 상세정보 페이지에서 각 가게의 하트 버튼을 누른다.
2	시스템은 나의 저장목록 화면에서 로그인 고객의 저장목록을 출력한다.
3	고객은 나의 저장목록에서 삭제하고 싶은 가게의 하트 버튼을 누른다.
4	시스템은 저장목록 삭제 확인 알림창을 출력한다.
5	고객은 삭제 확인 버튼을 누른다.
EXTENSION SCENARIOS	
Step	Branching Action
4	4a. 저장 목록 삭제 취소 ...4a1. 고객은 저장 목록 삭제 확인 알림창에서 취소 버튼을 눌러 삭제를 취소한다. ...4a2. 시스템은 저장 목록 삭제 알림창을 닫는다.
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	고객당 하루 평균 7회 이상
<Concurrency>	제한 없음
Due Date	2023.11.30.

Use case #8 : Manage Waiting	
GENERAL CHARACTERISTICS	
Summary	이 Use case는 고객의 가게 대기과 관련된 기능인 대기 신청, 대기 현황 조회, 대기 삭제를 설명한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 30
Status	Analysis
Primary Actor	Customer
Preconditions	대기 신청하고자 하는 가게의 상세정보 페이지를 열거나 사이드바에서 대기 현황 버튼을 누른다.
Trigger	고객이 인터페이스에서 대기과 관련된 기능을 하려고 할 때
Success Post Condition	선택한 가게에 대기자 신청을 한다. 대기 신청을 완료한 가게를 조회한다. 대기 신청 취소 버튼을 눌러 대기를 취소한다.
Failed Post Condition	대기에 대한 변경사항이 반영되지 않는다.
MAIN SUCCESS SCENARIO	
Step	Action
S	고객이 대기과 관련된 기능을 수행하려 한다.
1	고객은 가게 상세정보 페이지에서 줄서기 버튼을 누른다.
2	시스템은 대기 신청 정보를 입력할 수 있는 화면을 제공한다.
3	고객은 대기 신청 정보(전화번호, 이름, 인원수)를 기입하고 확인 버튼을 누른다.
4	시스템은 제공된 정보를 기반으로 대기자를 추가한다.
5	시스템은 대기 현황 페이지에서 현재 로그인한 고객의 대기 신청 완료 현황(대기 신청 가게, 대기 순번)을 출력한다.
6	고객은 대기 현황 페이지에서 대기 취소 버튼을 눌러서 대기 신청을 취소한다.
EXTENSION SCENARIOS	
Step	Branching Action
3	3a. 대기 신청 취소 ...3a1. 고객은 대기 신청 정보를 모두 기입하고 확인 버튼을 누른다. ...3a2. 시스템은 대기 신청을 확인하는 알림창을 출력한다. ...3a3. 고객은 취소 버튼을 눌러 대기 신청을 취소한다. 3b. 대기 신청 정보 미기입 ...3b1. 고객은 대기 신청 정보를 모두 작성하지 않고 확인 버튼을 누른다. ...3b2. 시스템은 모든 대기 신청 정보를 입력하라는 알림창을 출력한다.

6	6a. 신청 완료 대기 삭제 취소 ...6a1. 고객은 대기 현황 페이지에서 취소하고자 하는 대기의 취소 버튼을 누른다. ...6a2. 시스템은 대기 취소를 확인하는 알림창을 제공한다. ...6a3. 고객은 알림창의 취소 버튼을 눌러 대기 삭제를 취소한다.
---	--

RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	고객당 하루 평균 1회 이상
<Concurrency>	제한 없음
Due Date	2023.11.30.

Use case #9 : Manage Account	
GENERAL CHARACTERISTICS	
Summary	이 Use case는 사용자의 기본 정보(비밀번호, 전화번호)를 변경하는 기능과 계정 탈퇴 기능을 관리한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer, Owner
Preconditions	사용자는 DGAJ에 계정을 등록하고 로그인을 한 상태이다.
Trigger	사용자가 인터페이스에서 계정 정보를 변경하거나 계정을 탈퇴하려고 할 때
Success Post Condition	사용자의 계정 정보가 변경 또는 삭제된다.
Failed Post Condition	사용자 계정의 변경사항이 반영되지 않는다.
MAIN SUCCESS SCENARIO	
Step	Action
S	사용자가 계정 정보를 변경하거나 삭제하려고 한다.
1	사용자는 사이드바에서 내 계정 버튼을 누른다.
2	시스템은 내 계정의 세부 기능을 나타내는 화면을 제공한다.
3	사용자는 비밀번호 변경 버튼을 누르고 시스템은 비밀번호 변경 정보를 입력하는 화면을 제공한다.
4	사용자는 비밀번호 변경 창에 변경 전 비밀번호와 변경할 비밀번호, 비밀번호 확인을 입력하고 변경 버튼을 누른다.
5	사용자는 전화번호 변경 버튼을 누르고, 시스템은 전화번호 변경 정보를 입력하는 화면을 제공한다.
6	사용자는 전화번호 변경 창에 기존 전화번호를 입력하고 확인 버튼을 눌러 기존 정보가 맞는지 확인한 후, 변경 전화번호, 전화번호 확인을 입력하여 변경 버튼을 누른다.
7	사용자는 탈퇴하기 버튼을 누르고, 시스템은 탈퇴 여부를 재확인하는 알림창을 화면에 제공한다.
8	사용자는 사이드바의 다른 버튼을 눌러 다른 기능을 수행한다.
EXTENSION SCENARIOS	
Step	Branching Action
2	2a. 내 계정 화면 ...2a1. 시스템은 비밀번호를 변경하기 위해 비밀번호 변경 버튼을 화면에 제공한다.

	...2a2. 시스템은 전화번호를 변경하기 위해 전화번호 변경 버튼을 화면에 제공한다. ...2a3. 시스템은 계정을 탈퇴하기 위해 탈퇴하기 버튼을 화면에 제공한다.
4	4a. 비밀번호 변경 창의 모든 정보가 입력되지 않았다. ...4a1. 시스템은 비밀번호를 변경하기 위한 정보를 입력하라는 메시지를 출력한다. ...4a2. 비밀번호 변경 정보를 입력하는 단계로 돌아간다.
6	6a. 전화번호 변경 창의 모든 정보가 입력되지 않았다. ...6a1. 시스템은 전화번호를 변경하기 위한 정보를 입력하라는 메시지를 출력한다. ...6a2. 전화번호 변경 정보를 입력하는 단계로 돌아간다.
7	7a. 탈퇴하기 취소 ...7a1. 사용자는 탈퇴 기능을 확인하는 알림창에서 취소 버튼을 누른다. ...7a2. 시스템은 알림창을 닫고 내 계정 페이지를 화면에 출력한다.
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	사용자당 평균 2회 이상
<Concurrency>	제한 없음
Due Date	2023.11.03.

Use case #10 : Manage Restaurant Waiting
GENERAL CHARACTERISTICS

Summary	이 Use case는 사업자가 운영하는 가게의 대기열을 생성, 삭제, 수정하여 관리하는 기능을 설명한다. Use case #8 Manage Waiting과 연결된 Use case이다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 30
Status	Analysis
Primary Actor	Owner
Preconditions	사용자가 사업자 유형으로 로그인을 완료한 상태이다. Use case #1 Register 기능으로 연결된 가게가 존재하는 상태이다.
Trigger	사업자가 운영하는 가게의 대기열을 관리하는 기능을 하려고 할 때
Success Post Condition	가게의 대기열을 생성한다. 대기열을 삭제한다. 대기열에 추가된 고객들을 관리한다.
Failed Post Condition	해당 가게의 대기열에 대한 변경사항이 반영되지 않는다.

MAIN SUCCESS SCENARIO

Step	Action
S	사업자가 운영하는 가게의 대기열을 관리하고자 한다.
1	시스템은 DB와 연결하여 현재 가게의 대기열 상태를 화면에 제공한다.
2	사업자는 대기열을 생성하거나 변경한다.
3	사업자가 대기열 열기 버튼을 누른다.
4	사업자가 대기자를 선택하여 누른다.
5	시스템은 선택한 대기자(고객)의 대기 정보를 제공한다.
6	사업자가 입장시키고자 하는 대기자의 입장 버튼을 누른다.
7	시스템은 선택한 대기자 정보를 대기열에서 삭제하고 변경사항을 반영한다.
8	사업자가 대기 취소하고자 하는 대기자의 삭제 버튼을 누른다.
9	시스템은 선택한 대기자 정보를 대기열에서 삭제하고 변경사항을 반영한다.
10	사업자가 화면에서 대기열 닫기 버튼을 누른다.
11	시스템은 사업자의 가게 대기열을 삭제하고 변경사항을 반영한다.

EXTENSION SCENARIOS

Step	Branching Action
1	1a. 가게의 대기열이 존재하지 않는다. ...1a1. 시스템은 대기열 열기 버튼을 화면에 제공한다.

	...1a2. 사업자가 열기 버튼을 누른다. ...1a3. 시스템은 대기열을 생성한다. 1b. 가게의 대기열이 존재한다. ...1b1. 시스템은 대기열을 불러와 화면에 출력한다.
6	6a. 대기자 입장 취소 ...6a1. 시스템은 선택한 대기자의 입장을 확인하는 알림창을 출력한다. ...6a2. 사업자는 알림창에서 취소 버튼을 누른다. ...6a3. Step 1로 돌아간다.
8	8a. 대기자 삭제 취소 ...8a1. 시스템은 선택한 대기자의 삭제를 확인하는 알림창을 출력한다. ...8a2. 사업자는 알림창에서 취소 버튼을 누른다. ...8a3. Step 1로 돌아간다.
10	10a. 대기열 닫기 취소 ...10a1. 시스템은 대기열 닫기를 확인하는 알림창을 출력한다. ...10a2. 사업자는 알림창에서 취소 버튼을 누른다. ...10a3. Step 1로 돌아간다.
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	사업자당 하루 평균 10회 이상
<Concurrency>	제한 없음
Due Date	2023.11.30.

Use case #11 : Manage Donation	
GENERAL CHARACTERISTICS	
Summary	이 Use case는 사업자가 운영하는 가게의 재고를 다른 기관이나 가게에 기부하는 기능을 관리한다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 30
Status	Analysis
Primary Actor	Owner
Preconditions	사업자가 운영하는 가게는 시스템에 등록된 상태이고, 가까운 위치의 기부 대상 기관이 등록된 상태이다.
Trigger	사업자가 기부 관련 기능을 하려고 할 때
Success Post Condition	신청한 기부가 성공적으로 이루어진다.
Failed Post Condition	기부가 이루어지지 않는다.
MAIN SUCCESS SCENARIO	
Step	Action
S	사업자가 사업장 재고를 기부하고자 한다.
1	사업자는 사이드바에서 기부하기 버튼을 누른다.
2	시스템은 기부할 수 있는 센터 목록을 화면에 제공한다.
3	사업자는 원하는 센터의 기부하기 버튼을 누른다.
4	시스템은 해당 센터의 기부 신청하기 화면을 제공한다.
5	사업자는 기부자 이름, 기부 목록, 수량 등 기부 정보를 입력하고 기부하기 버튼을 누른다.
6	시스템은 내가 나눈 행복 화면에 사업자가 기부한 목록을 제공한다.
EXTENSION SCENARIOS	
Step	Branching Action
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	사업자당 이를 평균 1회 이상
<Concurrency>	제한 없음
Due Date	2023.11.30.

Use case #12 : Sign-out	
GENERAL CHARACTERISTICS	
Summary	이 Use case는 사용자가 현재 로그인한 계정을 로그아웃하는 기능이다.
Scope	DGAJ(Daegu Go And Joy)
Level	User level
Author	테크자이언트
Last Update	2023. 11. 03
Status	Analysis
Primary Actor	Customer, Owner
Preconditions	사용자가 특정 계정으로 로그인한 상태이다.
Trigger	사용자가 로그아웃하려고 할 때
Success Post Condition	성공적으로 로그아웃이 완료된다.
Failed Post Condition	로그아웃이 완료되지 않다.
MAIN SUCCESS SCENARIO	
Step	Action
S	사용자가 로그아웃 하고자 한다.
1	사용자는 사이드바에서 로그아웃 버튼을 누른다.
2	시스템은 로그아웃을 확인하는 알림창을 제공한다.
3	사용자는 알림창의 확인 버튼을 누른다.
4	시스템은 해당 계정을 로그아웃한다.
EXTENSION SCENARIOS	
Step	Branching Action
3	3a. 로그아웃 취소 ...3a1. 사용자는 알림창에서 취소 버튼을 누른다. ...3a3. Step 1로 돌아간다.
RELATED INFORMATION	
Performance	≤ 2 seconds
Frequency	사업자당 한 달 평균 3회 이상
<Concurrency>	제한 없음
Due Date	2023.11.03.

Restaurant			
Class Description	Restaurant 클래스는 식당 정보를 관리하는 클래스, 식당의 이름, 요리 종류, 전화번호, 주소, 설명, 영업시간, 대표 메뉴, 오픈데이터 ID 및 Pick 수를 저장하고 제공하는 class		
Attributes	Name	Type	Visibility
	Description		
	name	String	private
	식당의 이름을 나타내는 변수		
	cuisine	String	private
	식당의 요리 종류를 나타내는 변수		
	phoneNumber	String	private
	식당의 전화번호를 나타내는 변수		
	address	String	private
	식당의 주소를 나타내는 변수		
	description	String	private
	식당에 대한 설명을 나타내는 변수		
	businesshour	String	private
	식당의 영업시간을 나타내는 변수		
	topMenu	String	private
	식당의 대표 메뉴를 나타내는 변수		
	opendata_id	String	private
	오픈데이터 ID를 나타내는 변수		
	pickCount	int	private
	식당이 선택받은 (찜)수를 나타내는 변수		
Operations	Name	Argument	Returns
	Description		
	Restaurant	String name, String cuisine, String phone Number, String address, String description, String businesshour, String topMenu, String opendata_id, int pickCount	none
	Restaurant 클래스의 생성자. 매개변수로 받은 값을 각 속성에 설정		
	getName	None	String
	식당의 이름을 반환하는 메서드		
	getCuisine	None	String
	식당의 요리 종류를 반환하는 메서드		
	getPhoneNumber	None	String
	식당의 전화번호를 반환하는 메서드		
	getAddress	None	String
	식당의 주소를 반환하는 메서드		
	getDescription	None	String
	식당에 대한 설명을 반환하는 메서드		
	getBusinesshour	None	String
	식당의 영업시간을 반환하는 메서드		
	getTopMenu	None	String
	식당의 대표 메뉴를 반환하는 메서드		
	getOpenData_id	None	String

	오픈데이터 ID를 반환하는 메서드		
	getPickCount	None	int
	식당이 선택받은 (찜)수 반환하는 메서드		

RestaurantAdapter			
Class	RestaurantAdapter 클래스는 RecyclerView에 식당 목록을 표시하기 위한 어댑터 클래스. 식당 정보를 받아와서 각각의 아이템 뷰에 표시		
Description	Name	Type	Visibility
	Description		
Attributes	restaurantList	List<Restaurant>	private
	식당 목록을 담고 있는 리스트		
	context	Context	private
	어댑터를 사용하는 변수		
	listener	OnItemClickListener	private
	아이템 클릭 이벤트를 처리하기 위한 리스너		
	Name	Argument	Returns
	Description		
Operations	setOnItemClickListener	OnItemClickListener listener	None
	아이템 클릭 이벤트 리스너를 설정하는 메서드		
	RestaurantAdapter	List<Restaurant> restaurantList, Context context	None
	RestaurantAdapter 클래스의 생성자입니다. 식당 목록과 컨텍스트를 매개 변수로 받아 초기화하는 메서드		
	onCreateViewHolder	ViewGroup parent, int viewType	ViewHolder
	아이템 뷰를 위한 ViewHolder 객체를 생성하여 반환하는 메서드		
	onBindViewHolder	ViewHolder holder, int position	void
	ViewHolder에 식당 정보를 바인딩하여 표시하는 메서드		
	getItemCount	None	int
	식당 목록의 크기를 반환하는 메서드		
	ViewHolder	None	static class
	각 아이템 뷰의 레이아웃 요소를 저장하는 ViewHolder와 관한 메서드		
	OnItemClickListener	None	interface
	아이템 클릭 이벤트 리스너 인터페이스		

ChangeAddressToGeoPoint			
Class Description	hangeAddressToGeoPoint 클래스는 주소를 지리적 좌표로 변환하는 기능을 제공하는 클래스. Geocoder를 사용하여 주소를 좌표로 변환하고, 결과로 얻은 위도(latitude)와 경도(longitude)를 저장		
Attributes	Name	Type	Visibility
	Description		
	latitude	double	private
	주소의 위도 좌표를 저장하는 변수		
	longitude	double	private
	주소의 경도 좌표를 저장하는 변수		
	address	String	private
Operations	변환하고자 하는 주소를 저장하는 변수		
	Name	Argument	Returns
	Description		
	ChangeAddressToGeoPoint	String address	None
	주소를 초기화하기 위해 생성자에서 주소를 받아온다.		
	setAddress	String address	private
	주소를 설정하는 메서드		
	getAddress	None	String
	현재 저장된 주소를 반환하는 메서드		
	getGeoPoint	Context mContext	Location
	주소를 지리적 좌표로 변환하여 Location 객체로 반환하는 메서드		

ChangeGeoPointToAddress			
Class Description	ChangeGeoPointToAddress 클래스는 지리적 좌표를 주소로 변환하는 기능을 제공하는 클래스. Geocoder를 사용하여 좌표를 주소로 변환하고, 결과로 얻은 현재 위치 주소를 반환		
	Name	Type	Visibility
Attributes	Description		
	latitude	double	private
	현재 위치의 위도 좌표를 저장하는 변수		
	longitude	double	private
Operations	현재 위치의 경도 좌표를 저장하는 변수		
	Name	Argument	Returns
	Description		
	ChangeGeoPointToAddress	double latitude, double longitude	None
	위도와 경도를 초기화하기 위해 생성자에서 값을 받아온다		
	setLatitude	double latitude	None
	위도를 설정하는 메서드		
	setLongitude	double longitude	None
	경도를 설정하는 메서드		
	getLatitude	double	double
	현재 저장된 위도 값을 반환하는 메서드		
	getLongitude	double	double
	현재 저장된 경도 값을 반환하는 메서드		
	getAddress	Context mContext	String
	좌표를 주소로 변환하여 현재 위치 주소를 반환하는 메서드		

GetMyGeoPoint			
Class Description	GetMyGeoPoint 클래스는 현재 장치의 위치 정보를 가져오는 기능을 제공하는 클래스. LocationManager와 LocationListener를 사용하여 위치 정보를 수신하고, 마지막으로 수신된 위치 정보를 반환		
Attributes	Name	Type	Visibility
	Description		
	locationManager	LocationManager	private
	위치 관리자 변수		
	locationListener	LocationListener	private
	위치 변경 이벤트를 수신하는 리스너 객체		
Operations	lastLocation	Location	private
	마지막으로 수신된 위치 정보를 저장하는 변수		
	Name	Argument	Returns
	Description		
	getLocation	Context context, LocationCallback callback	None
	현재 장치의 위치 정보를 가져오는 메서드		
	interface LocationCallback	None	None
	위치 정보를 전달하기 위한 콜백 인터페이스		

Customer_mainScreen			
Class	사용자가 이용한 main화면을 관리하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	mMap	GoogleMap	private
	Google Map기능을 이용하기 위한 변수		
	adapter	RestaurantAdapter	private
	restaurantList를 사용자와 연결시켜주는 변수		
	restaurantList	List<Restaurant>	private
	사용자가 이용할 수 있는 모든 가게 정보를 가진 list		
	nowAddress	String	private
	사용자의 현 위치를 가진 변수		
	Lineup	TextView	private
	사용자가 이용할 줄서기 기능을 나타내는 변수		
	searchMenu	String	private
	검색하고 싶은 해당 음식을 기록하는 변수		
	recyclerMenu	RecyclerView	private
	상세 기능을 이용하기 위해 누르는 Button		
	searchPos	String	private
	사용자의 현위치를 간단하게 나타낸 변수		
	drawerLayout2	DrawerLayout	private
	화면 왼쪽에 상세 메뉴가 담긴 창을 숨겨놓았다가 사용자가 상세 메뉴 Button을 누르면 화면에 나타나는 레이아웃		
	menu2	Button	private
	상세기능을 가진 창을 해당 창에 나타내기 위한 Button		
	addressParts	String[]	private
	사용자가 있는 주소를 기록하는 변수		
	shop_page	TextView	private
	사용자가 이용할 가게 상세페이지를 나타내는 변수		
	address_gu	String	private
	사용자의 주소 중 해당 구를 나타내는 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	onNavigationItemSelected	MenuItem item	boolean
	화면 왼쪽에 숨겨놓은 상세메뉴가 담긴 창을 사용자가 상세메뉴Button을 눌렀을 때 사용자가 이용할수 있도록 창을 나타내주는 메서드		

Customer_searchScreen			
Class	오늘의 메뉴를 검색하거나 다른 세부 기능들을 이용할 수 있도록 연결시켜주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	txt_search_pos	EditText	private
	사용자의 위치를 기록하는 변수		
	searchPos	String	public
	사용자가 위치한 구를 기록하는 변수		
	nowAddress	String	private
	사용자의 현 위치를 기록하는 변수		
	mdate	Date	private
	오늘의 날짜를 기록하는 변수		
	date_search _screen	TextView	private
	오늘의 날짜를 나타내는 변수		
	searchMenu	String	public
	검색한 메뉴를 기록하는 변수		
	mNow	long	private
	현재 시간을 기록하는 변수		
	btn_random _select	Button	private
	랜덤으로 음식을 고르기 위한 버튼		
	txt_search_pos	EditText	private
	사용자가 위치한 구를 기록할 변수		
	mFormat	SimpleDateFormat	private
	시간을 나타내는 형식		
	btn_set_current _location	Button	private
	현 위치로 위치 설정을 변경할 수 있는 버튼		
	tz	TimeZone	private
	현재 위치한 국가의 시간대를 가져오는 기능		
	menuList	String[]	private
	random 기능을 이용했을 때 필요한 메뉴 리스트 배열		
	addressParts	String[]	private
	가져온 위치에서 나타낼 위치정보만 고르기 위해 만든 배열		
	btn_search	Button	private
	사용자가 해당 음식을 입력한 후 검색하기 위한 버튼		
	btn_game	Button	private
	사용자가 게임 기능을 이용할 때 선택하는 버튼		
	menu	Button	private
	사용자가 이용할 수 있는 기능 menu들을 모아놓은 창을 이용하기 위해 선택하는 버튼		
	customAnimation Dialog	CustomAnimation Dialog	private
	로딩화면을 구성하기 위한 변수		
	drawerLayout	DrawerLayout	private
	화면 왼쪽에 상세 메뉴가 담긴 창을 숨겨놓았다가 사용자가 상세 메뉴버튼을 누르면 화면에 나타나는 레이아웃		
	txt_search_menu	EditText	private
	검색하고 싶은 해당 음식을 기록하는 변수		

	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState State	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	onNavigationItemSelected	MenuItem	boolean
	화면 왼쪽에 숨겨놓은 상세메뉴가 담긴 창을 사용자가 상세메뉴버튼을 눌렀을 때 사용자가 이용할 수 있도록 창을 나타내주는 메서드		
	getTime	None	String
	실제 시간을 가지고 오는 메서드		

myWaitingAdapter			
Class	My Waiting View를 가져오기 위해 도움을 주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	restaurantList	List<Restaurant>	private
	모든 식당 정보를 저장하는 Restaurant List		
	context	Context	private
	현재 화면 상태를 보여주는 변수		
	myWaitinglist	ArrayList <myWaiting>	private
	사용자가 대기 중인 식당 정보를 저장하는 myWaiting ArrayList		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	uid	String	private
	사용자의 uid를 기록하는 변수		
	Name	Argument	Returns
	Description		
Operations	onCreateViewHolder	ViewGroup, int	None
	새로운 ViewHolder 인스턴스를 생성하고 반환하는 메서드		
	OnBindViewHolder	ViewHolder, int	None
	ViewHolder의 각 View에 대기 중인 식당 정보를 가져오고 클릭이벤트를 처리해주는 메서드		
	getItemCount	None	int
	대기 중인 식당 리스트의 크기를 반환하는 메서드		

Customer_applywaiting			
Class	줄서기 기능을 관리하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	lineup_tel	EditText	private
	줄서기 기능 이용시, 입력해야 하는 전화번호를 기록하는 변수		
	lineup_name	EditText	private
	줄서기 기능 이용시, 입력해야 하는 이름을 기록하는 변수		
	reservation_num	EditText	private
	줄선 사용자를 정의하기 위한 변수		
	btn_confirm_line_up	Button	private
	줄서기를 사용자가 확정할 때 누르는 Button		
	queue	long	private
	줄서는 사용자를 기록하기 위한 변수		
	mStore	FirebaseFirestore	private
	Firebase store의 접근하기 위한 변수		
	restaurant Opendata_id	String	public
	사용자가 줄 선 가게의 id를 저장하는 변수		
	uid	String	private
	사용자의 UID를 저장하는 변수		
	restaurantAddr_gu	String	private
	음식점 주소의 구를 저장하는 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	onSuccess	None	None
	사용자가 줄서기 하고 싶은 가게의 줄서기 명단에 사용자를 입력해주는 메서드		
	onFailure	Exception e	None
	줄서기 기능을 이용하고 싶은 사용자를 가게 줄서기 명단에 입력하지 못한 경우 예외처리를 위한 메서드		

myWaiting			
Class			
Description	사용자가 대기중인 식당의 정보를 관리하고 접근할 수 있도록 해주는 class		
Attributes	Name	Type	Visibility
	Description		
	RES_ID	String	private
	사용자가 대기 중인 식당의 고유 ID를 저장하는 변수		
	wait_rest_addr	String	private
	사용자가 대기 중인 식당의 주소를 저장하는 변수		
	wait_rest_FD	String	private
	사용자가 대기 중인 식당의 요리 테마 또는 특징을 저장하는 변수		
	wait_rest_name	String	private
	사용자가 대기 중인 식당의 이름을 저장하는 변수		
Operations	Name	Argument	Returns
	Description		
	getWait_rest_name	None	String
	사용자가 대기 중인 식당의 이름을 가져오는 메서드		
	getWait_rest_FD	None	String
	사용자가 대기 중인 식당의 요리 테마 또는 특징을 가져오는 메서드		
	getWait_rest_addr	None	String
	사용자가 대기 중인 식당의 주소를 가져오는 메서드		
	getWait_RES_ID	None	String
	사용자가 대기 중인 식당의 고유 ID를 가져오는 메서드		

Customer_restaurantDetail			
Class	사용자의 식당 세부 정보를 표시하고, 찜하기 및 대기열 신청 기능을 제공하는 class		
Description	Name	Type	Visibility
	Description		
Attributes	btn_lineup	Button	private
	대기열 신청 Button		
	btn_pick	ToggleButton	private
	찜하기를 위한 토글 Button		
	mStore	Firestore	private
	Firestore store에 접근하기 위한 변수		
Attributes	donating_angel	ImageView	private
	기부천사 이미지를 표시하는 ImageView		
	isOpendataIdExist	boolean	private
	사용자의 대기 신청 목록에 해당 식당의 opendata_id가 이미 존재하는지 여부를 저장하는 변수		
	uid	String	private
	Firestore store에 접근하기 위한 변수		
	restaurantOpenData_id	String	private
	식당의 opendata_id		
	address_gu	String	private
	식당의 주소에서 구를 추출한 값		
Name	Name	Argument	Returns
	Description		
Operations	onCreate	None	
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	onComplete	Task<Document Snapshot> task	None
	가게 상세정보를 계속해서 업데이트하는 메서드		
	handleToggleOn	None	None
	토글 Button이 클릭된 상태일 때의 동작을 처리하는 메서드입니다. 찜하기 기능을 수행하고 Firestore에 데이터를 업데이트		
	handleToggleOff	None	None
	토글 Button이 클릭되지 않은 상태일 때의 동작을 처리하는 메서드입니다. 찜 취소 기능을 수행하고 Firestore에 데이터를 업데이트		

SplashActivity			
Class			
Description	앱 실행 시 로딩 화면을 보여주기 위한 역할을 수행하는 클래스		
	Name	Type	Visibility
	Description		
Attributes	imageView	ImageView	private
	로딩 화면을 표시하기 위한 ImageView 객체		
	animationDrawable	AnimationDrawable	private
	로딩 화면의 애니메이션을 제어하기 위한 AnimationDrawable 객체		
	Name	Argument	Returns
	Description		
Operations	onStart	None	None
	일정 시간이 지난 후 Intent를 사용하여 SplashActivity에서 Login Activity로 이동하게 하는 메서드		

ImageDTO			
Class			
Description	별점 개수, 사용자별 별점 부여 여부를 관리하는 class		
	Name	Type	Visibility
	Description		
Attributes	firebaseDatabase	FirebaseDatabase	private
	Firebase Realtime Database와 접근할 수 있는 변수		
	startCount	int	private
	찜 개수를 저장하는 변수		
	stars	Map<String,Boolean>	private
	사용자별로 별점을 부여했는지 여부를 저장하는 Map 키는 사용자 ID를 나타내는 문자열이고, 값은 해당 사용자가 별점을 부여했는지 여부를 나타낸다.		
	mAuth	FirebaseAuth	private
	Firebase authentication에 접근해 인증받기 위한 변수		

Customer_myWaiting			
Class	사용자의 대기현황을 가져오는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	adapter	myWaitingAdapter	private
	사용자의 대기현황을 형식에 맞춰 보여주는 adapter		
	arrayList	ArrayList <myWaiting>	private
	사용자가 대기 중인 식당 정보를 저장하는 myWaiting ArrayList		
	waiting_addr_gu	String	private
	대기 중인 식당의 주소를 저장하는 변수		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	res_id	String	private
	대기 중인 식당의 고유 ID를 저장하는 변수		
	drawerLayout	DrawerLayout	private
	화면의 메뉴를 관리하는 DrawerLayout 변수		
	restaurantList	List<Restaurant>	private
	모든 식당 정보를 저장하는 Restaurant List		
	index	int	private
	현재 처리 중인 대기 목록 아이템의 인덱스를 저장하는 변수		
	recyclerView	RecyclerView	private
	대기현황을 보여주는 recyclerView		
	mapList	List<Map<String, Object>>	private
	Firestore에서 가져온 사용자의 대기 목록 정보를 저장하는 Map List		
	uid	String	private
	사용자의 uid를 기록하는 변수		
	TAG	String	private
	로그 메시지를 출력할 때 사용하는 변수		
Operations	Name	Argument	Returns
	Description		
	onCreate	Bundle savedInstanceState	None
	Firestore에서 사용자의 대기 목록 정보를 가져오고 공공 데이터 API를 통해 대기 중인 식당의 정보를 가져오는 메서드		
	fetchOpendataIds	None	None
	웹에서 저장목록을 위한 데이터들을 가지고 오는 메서드		

3.2 Account Class diagram

계정과 관련된 기능을 나타낸 account CD이다. Account 클래스들은 앱을 사용하는 모든 이용자의 계정 관리에 관련된 기능들을 담당하고 있다. Firebase Firestore를 이용하여 실시간 계정 정보를 관리하고 제공한다. 사용자의 로그인, 회원가입, 계정 정보 변경, 아이디/비밀번호 찾기 등의 기능을 통해 안전한 인증과 개인정보 관리를 지원한다. 사용자 경험을 개선하고 개인정보 관리를 지원해 애플리케이션의 보안성을 높이는데 중요한 역할을 한다.

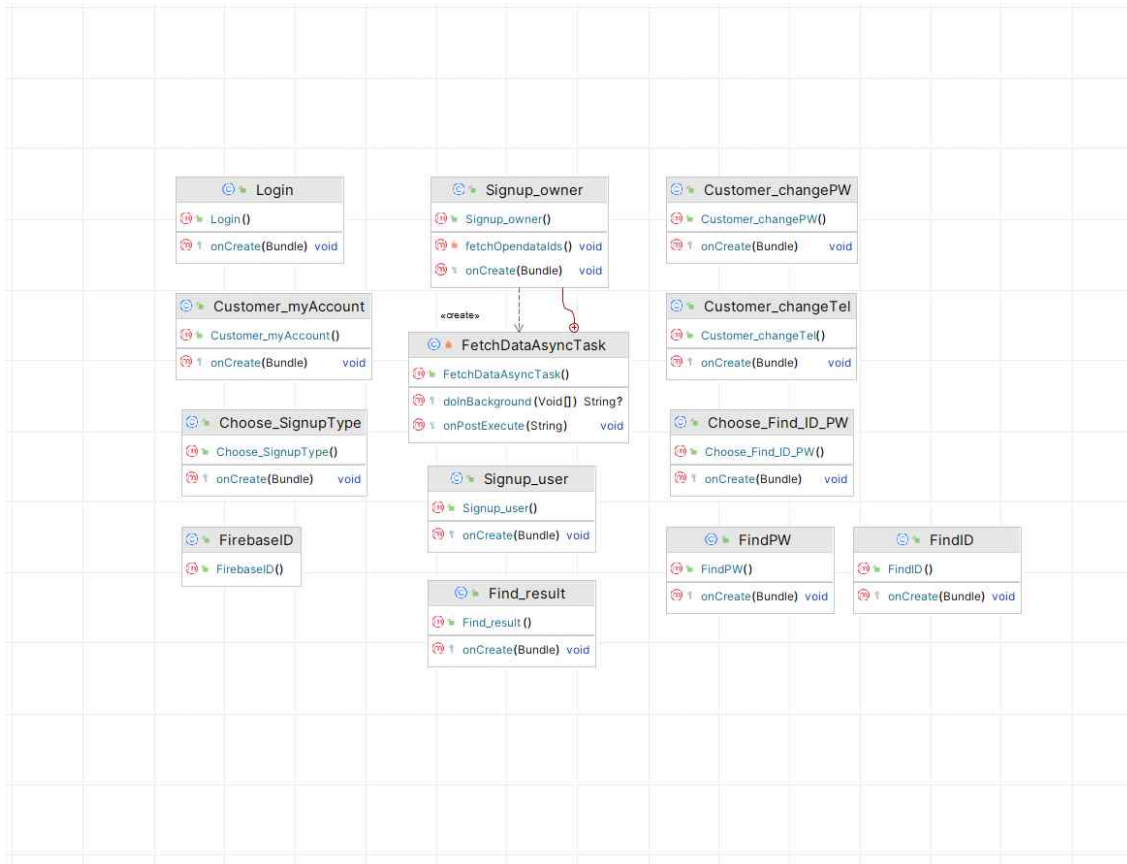


그림 3-2 Account CD

Login			
Class Description	Login 클래스는 사용자 로그인 기능을 구현하는 클래스. Firebase Authentication과 Firestore를 사용하여 사용자 인증 및 데이터 조회를 수행		
	Name	Type	Visibility
	Description		
Attributes	id_login	EditText	private
	이메일 입력을 위한 EditText		
	pw_login	EditText	private
	비밀번호 입력을 위한 EditText		
	btn_login	Button	private
	로그인 Button		
	btn_signup	Button	private
	회원가입 Button		
	mAuth	FirebaseAuth	private
	Firebase Authentication 인스턴스		
	mStore	Firestore	private
	Firestore 인스턴스		
	RES_ID	String	private
	상수로 정의된 식당 ID		
	btn_find_id_pw	Button	private
	아이디/비밀번호 찾기 화면으로 이동하는 Button		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	액티비티가 생성될 때 호출되는 메서드, 레이아웃 초기화 및 Button 클릭 이벤트 등을 설정, 로그인 Button 클릭 시 Firebase Authentication을 사용하여 로그인을 수행, Firestore에서 사용자 정보를 조회하고, 사용자 역할에 따라 액티비티를 전환		

Customer_myAccount			
Class	사용자의 계정을 관리하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	mStore	Firestore	private
	Firestore에 접근하기 위한 변수		
	btn_change_password	Button	private
	비밀번호를 변경하기 위해 선택하는 Button		
	btn_change_tel	Button	private
	전화번호를 변경하기 위해 선택하는 Button		
	btn_delete_account	Button	private
	계정을 탈퇴하기 위해 선택하는 Button		
	mFirebaseAuth	FirebaseAuth	private
	Firebase authentication에 접근하기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		

Choose_SignupType			
Class	Choose_SignupType 클래스는 회원가입 유형을 선택하는 화면을 나타내는 클래스입니다.		
Description			
	Name	Type	Visibility
	Description		
Attributes	type	String	private
	선택된 회원가입 유형을 저장하는 변수		
	btn_choose_user	Button	private
	사용자 회원가입 Button		
	btn_choose_owner	DrawerLayout	private
	소유자 회원가입 Button		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화와 Button 클릭 이벤트 처리를 수행합니다. 액티비티가 생성될 때 사용자 회원가입 Button과 소유자 회원가입 Button을 클릭하면 각각의 회원가입 화면으로 이동하는 기능이 구현되어 있습니다. 선택된 회원가입 유형은 인텐트에 "type"이라는 키로 저장되어 전달됩니다.		

Signup_owner			
Class Description	Signup 클래스는 식당 주인 회원가입 기능을 구현하는 클래스. Firebase Authentication과 Firestore를 사용하여 사용자 계정을 생성하고 정보를 저장		
	Name	Type	Visibility
	Description		
Attributes	pw_join	EditText	private
	비밀번호 입력을 위한 EditText		
	name_join	EditText	private
	이름 입력을 위한 EditText		
	id_join	EditText	private
	아이디 입력을 위한 EditText		
	tel_join01	EditText	private
	휴대폰 번호 첫 번째 자리를 입력하는 EditText		
	tel_join02	EditText	private
	휴대폰 번호 두 번째 자리를 입력하는 EditText		
	btn_join	Button	private
	회원가입 Button		
	btn_id_check	Button	private
	아이디 중복 체크 Button		
	choose_join_type	RadioGroup	private
	사용자 유형 선택을한 RadioGroup		
	mAuth	FirebaseAuth	private
	Firebase Authentication 인스턴스		
	mStore	FirebaseFirestore	private
	Firebase Firestore 인스턴스		
	spinnerEmailDomains	Spinner	private
	이메일 도메인 선택을 위한 Spinner		
	btn_opendata_id_check	Button	private
	오픈데이터 아이디 중복 체크 Button		
	type	String	private
	사용자 유형을 나타내는 문자열		
	opendata_id_join	String	private
	오픈데이터 아이디 입력을 위한 EditText		
	get_opendata_id	EditText	private
	입력받은 오픈데이터 아이디를 저장하는 문자열		
	opendata_resname	TextView	private
	오픈데이터에서 조회한 식당 이름과 주소를 표시하는 텍스트뷰		
	mapList	List<Map<String, Object>>	private
	멤버 변수로 선언된 맵 리스트		
	index	List<String>	private
	멤버 변수로 선언된 인덱스		
	searchPosList	RadioGroup	private
	검색할 지역 리스트		
	TAG	String	private
	디버깅 로그를 위한 태그 문자열		
	checkResult	Boolean	private
	중복 체크 결과를 저장하는 변수		
	searchPos	String	private

	검색할 지역을 저장하는 문자열		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState State	None
	<p>액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화, Button 클릭 이벤트 등을 설정합니다. 아이디 중복 체크 Button을 클릭하여 Firestore에서 중복된 아이디가 있는지 확인하고, 오픈데이터 아이디 중복 체크 Button을 클릭하여 오픈데이터에서 식당 정보를 조회합니다. 회원가입 Button을 클릭하여 Firebase Authentication을 사용하여 회원가입을 수행하고, Firestore에 사용자 정보를 저장합니다. 사용자가 비밀번호, 이름, 아이디, 휴대폰 번호, 오픈데이터 아이디를 입력하고 회원가입 Button을 클릭하면 Firebase Authentication을 사용하여 회원가입을 수행하고, Firebase Firestore에 사용자 정보를 저장합니다. 아이디 중복 체크 기능과 오픈데이터 아이디 중복 체크 기능을 제공하여 중복된 아이디인 경우 회원가입을 거부하거나 오픈데이터에서 식당 정보를 조회합니다. 사용자 유형은 이전 액티비티에서 전달된 type 변수를 사용하여 설정됩니다. 회원가입이 성공적으로 완료되면 로그인 화면으로 이동합니다.</p>		

Customer_changePW			
Class	사용자가 기존에 등록한 비밀번호를 다른 비밀번호로 변경하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	et_existPW	EditText	private
	기존의 비밀번호를 기록하기 위한 변수		
	mStore	FirestoreFirestore	private
	Firestore store에 접근하기 위한 변수		
	check	boolean	private
	기존의 비밀번호와 일치할 경우 true를 기록할 변수		
	btn_ok	Button	private
	비밀번호 변경을 확정하기 위한 Button		
	et_checkPW	EditText	private
	변경된 비밀번호를 기록하기 위한 변수		
	btn_checkPW	Button	private
	기존의 비밀번호와 일치하는 비밀번호를 입력했는지 확인하는 Button		
	et_newPW	EditText	private
	변경할 새로운 비밀번호를 기록할 변수		
	mAuth	FirebaseAuth	private
	Firestore authentication에 접근해 인증받기위한 변수		
	customAnimationDialog	CustomAnimationDialog	private
	로딩화면을 구성하기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	onComplete	Task<DocumentSnapshot> task	None
	사용자의 새로운 비밀번호를 변경해주는 메서드		

Customer_changeTel			
Class	사용자가 기존에 등록한 전화번호를 다른 전화번호로 변경하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	mStore	Firestore	private
	Firestore에 접근하기 위한 변수		
	et_existTel	EditText	private
	사용자의 기존 전화번호를 입력받는 변수		
	et_newTel	EditText	private
	사용자가 새로운 전화번호를 입력받는 변수		
	et_checkTel	EditText	private
	사용자가 새로운 전화번호를 확인하기 위해 다시 입력받는 변수		
	btn_ok	Button	private
	전화번호 변경을 확정하기 위한 Button		
	btn_checkTel	Button	private
	기존의 전화번호와 일치하는 전화번호를 입력했는지 확인하는 Button		
	mAuth	FirebaseAuth	private
	Firebase authentication에 접근해 인증받기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	onComplete	Task<DocumentSnapshot> task	None
	사용자의 새로운 전화번호를 변경해주는 메서드		

Signup_user			
Class Description	Signup_user 클래스는 일반 사용자 회원가입 기능을 구현하는 클래스. Firebase Authentication과 Firestore를 사용하여 사용자 계정을 생성하고 정보를 저장		
	Name	Type	Visibility
	Description		
Attributes	pw_join	EditText	private
	비밀번호 입력을 위한 EditText		
	name_join	EditText	private
	이름 입력을 위한 EditText		
	id_join	EditText	private
	아이디 입력을 위한 EditText		
	tel_join01	EditText	private
	휴대폰 번호 첫 번째 자리를 입력하는 EditText		
	tel_join02	EditText	private
	휴대폰 번호 두 번째 자리를 입력하는 EditText		
	btn_join	Button	private
	회원가입 Button		
	btn_id_check	Button	private
	아이디 중복 체크 Button		
	choose_join_type	RadioGroup	private
	사용자 유형 선택을 위한 RadioGroup		
	mAuth	FirebaseAuth	private
	Firebase Authentication 인스턴스		
	mStore	FirebaseFirestore	private
	Firebase Firestore 인스턴스		
	spinnerEmailDomains	Spinner	private
	이메일 도메인 선택을 위한 Spinner		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화, Button 클릭 이벤트 등을 설정합니다. 아이디 중복 체크 Button을 클릭하여 Firestore에서 중복된 아이디가 있는지 확인하고, 회원가입 Button을 클릭하여 Firebase Authentication을 사용하여 회원가입을 수행하고, Firestore에 사용자 정보를 저장합니다. 사용자가 비밀번호, 이름, 아이디, 휴대폰 번호를 입력하고 회원가입 Button을 클릭하면 Firebase Authentication을 사용하여 회원가입을 수행하고, Firebase Firestore에 사용자 정보를 저장합니다. 아이디 중복 체크 기능을 제공하여 중복된 아이디인 경우 회원가입을 거부합니다. 사용자 유형은 이전 액티비티에서 전달된 type 변수를 사용하여 설정됩니다. 회원가입이 성공적으로 완료되면 로그인 화면으로 이동합니다.		

FirebaseID			
Class	FirebaseID 클래스는 Firebase Firestore에서 사용되는 컬렉션 및 문서 식별자를 관리하는 클래스. 각 상수는 해당하는 컬렉션 또는 문서의 이름을 나타낸다. (변수 설정으로 오타 방지용 클래스)		
Description			
	Name	Type	Visibility
	Description		
Constants	id_list	String	public
	ID 목록 컬렉션의 이름		
	waiting_list	String	public
	대기 목록 컬렉션의 이름		
	post	String	public
	게시물 컬렉션의 이름		
	upload	String	public
	업로드 컬렉션의 이름		
	title	String	public
	제목을 나타내는 필드의 이름		
	contents	String	public
	내용을 나타내는 필드의 이름		
	save	String	public
	찜목록 컬렉션의 이름		
	documentId	String	public
	문서 식별자를 나타내는 필드의 이름		
	email	String	public
	이메일을 나타내는 필드의 이름		
	password	String	public
	비밀번호를 나타내는 필드의 이름		
	tel	String	public
	전화번호를 나타내는 필드의 이름		
	name	String	public
	이름을 나타내는 필드의 이름		
	type	String	public
	타입을 나타내는 필드의 이름		
	wait_name	String	public
	대기자 이름을 나타내는 필드의 이름		
	wait_tel	String	public
	대기자 전화번호를 나타내는 필드의 이름		
	wait_num	String	public
	대기 번호를 나타내는 필드의 이름		
	collectionId	String	public
	컬렉션 식별자를 나타내는 필드의 이름		

Choose_Find_ID_PW			
Class	Choose_Find_ID_PW 클래스는 아이디 또는 비밀번호를 찾기 위해 선택하는 화면을 나타내는 클래스입니다.		
Description	Name	Type	Visibility
	Description		
Attributes	btn_find_id	Button	private
	아이디 찾기 Button		
	btn_find_pw	Button	private
	비밀번호 찾기 Button		
Operations	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화와 Button 클릭 이벤트 처리를 수행합니다. 액티비티가 생성될 때 아이디 찾기 Button과 비밀번호 찾기 Button을 클릭하면 각각의 화면으로 이동하는 기능이 구현되어 있습니다.		

FindID			
Class	FindID 클래스는 사용자의 이름과 전화번호를 입력하여 아이디를 찾는 화면을 나타내는 클래스입니다.		
Description	Name	Type	Visibility
	Description		
Attributes	et_name	EditText	private
	사용자의 이름을 입력하는 EditText		
	et_tel	EditText	private
	사용자의 전화번호를 입력하는 EditText		
	btn_find	Button	private
Operations	아이디 찾기 Button		
	Name	Argument	Returns
Operations	Description		
	onCreate	Bundle savedInstanceState	None
Operations	액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화와 Button 클릭 이벤트 처리를 수행합니다. 사용자가 이름과 전화번호를 입력하고 아이디 찾기 Button을 클릭하면 Firebase Firestore에서 해당 정보와 일치하는 사용자를 검색하여 아이디를 찾습니다. 검색 결과에 따라 해당 아이디를 Find_result 액티비티로 전달하여 표시합니다. 검색 결과가 없거나 데이터를 가져오는 과정에서 오류가 발생하는 경우에는 적절한 토스트 메시지를 표시합니다.		

Find_result			
Class	Find_result 클래스는 아이디 또는 비밀번호 찾기 결과를 나타내는 화면을 나타내는 클래스입니다.		
Description	Name	Type	Visibility
	Description		
Attributes	txt_found_id_pw	TextView	private
	아이디 또는 비밀번호를 나타내는 텍스트뷰		
	found_content	EditText	private
	찾은 아이디 또는 비밀번호를 표시하는 EditText		
	btn_backToLogin	Button	private
	로그인 화면으로 돌아가는 Button		
Name	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화와 데이터 표시, Button 클릭 이벤트 처리를 수행합니다. Find_result 클래스는 아이디 또는 비밀번호 찾기 결과를 나타내는 화면을 나타내는 액티비티입니다. 액티비티가 생성될 때 인텐트에서 전달된 데이터를 확인하여 텍스트뷰와 EditText에 찾은 아이디 또는 비밀번호를 표시합니다. 또한, 로그인 화면으로 돌아가는 기능이 구현되어 있는 Button을 클릭하면 로그인 화면으로 이동합니다.		

FindPW			
Class	FindPW 클래스는 사용자의 이름, 전화번호, 아이디를 입력하여 비밀번호를 찾는 화면을 나타내는 클래스입니다.		
Description	Name	Type	Visibility
	Description		
Attributes	et_name	EditText	private
	사용자의 이름을 입력하는 EditText		
	et_tel	EditText	private
	사용자의 전화번호를 입력하는 EditText		
	et_id	EditText	private
	사용자의 아이디를 입력하는 EditText		
	btn_find	Button	private
	비밀번호 찾기 Button		
	Name	Argument	Returns
	Description		
	onCreate	Bundle savedInstanceState	None
Operations	<p>액티비티가 생성될 때 호출되는 메서드로, 레이아웃 초기화와 Button 클릭 이벤트 처리를 수행합니다. 사용자가 이름, 전화번호, 아이디를 입력하고 비밀번호 찾기 Button을 클릭하면 Firebase Firestore에서 해당 정보와 일치하는 사용자를 검색하여 비밀번호를 찾습니다. 검색 결과에 따라 해당 비밀번호를 Find_result 액티비티로 전달하여 표시합니다. 검색 결과가 없거나 데이터를 가져오는 과정에서 오류가 발생하는 경우에는 적절한 토스트 메시지를 표시합니다.</p>		

3.3 TOP10 Class diagram

TOP10 리스트와 관련된 기능을 나타낸 CD이다. TOP10 클래스는 인기 맛집 리스트를 제공하고, 이를 효과적으로 표시하여 식당을 고르는데 어려움을 겪는 사용자에게 많은 도움을 준다. Firebase에 기록된 찜 정보를 기반으로 많은 사용자들이 저장한 맛집들을 보여준다. 사용자의 찜 정보가 달라지면 실시간으로 Firestore 데이터베이스의 정보도 변경되고 변경된 정보를 바탕으로 인기 Top10 정보를 업데이트하고, 이를 사용자에게 제공한다.



그림 3-3 Top10 CD

TopListAdapter			
Class	Customer_toplist class가 ListView를 가져오기 위해 도움을 주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	topList	List<Toplist>	private
	toplist를 가지고 있는 변수		
	context	Context	private
	toplist안의 내용을 가지고 있는 변수		
	listener	OnItemClickListener	private
	클릭이벤트를 처리해주는 변수		
	uid	String	private
	사용자의 uid를 기록하는 변수		
	restaurantList	List<Restaurant>	private
	하나의 식당에 대한 정보를 담은 Restaurant List		
	mStore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreateViewHolder	ViewGroup parent,int viewType	ViewHolder
	ViewHolder기능을 이용하기 위해 ViewHolder를 생성하는 메서드		
	setOnClickListener	OnItemClickListener	None
	클릭이벤트를 처리해주는 메서드		
	getItemCount	none	TopList.size
	top10 list에 속한 가게의 크기를 확인할 수 있는 메서드		
	onBindViewHolder	ViewHolder holder , int position	None
	ViewHolder기능을 이용해 그 위치(position)에 따라 list를 구성해주는 메서드		

Customer_topList			
Class Description	Customer_topList 클래스는 사용자가 top 10에 속한 가게를 보고 선택에 도움을 주는 클래스이다. top 10 버튼은 누르면 현재 top10 가게를 볼 수 있다.		
Attributes	Name	Type	Visibility
	Description		
	searchPos	String	private
	사용자가 위치한 구 정보를 가지고 있는 변수		
	recyclerView	RecyclerView	private
	top 10을 나타내주는 기능		
	adapter	TopListAdapter	private
	top10 list를 사용자와 연결시켜주는 변수		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	restaurantList	List<Restaurant>	private
	하나의 식당에 대한 정보를 담은 Restaurant 리스트		
	uid	String	private
	사용자의 uid를 기록하는 변수		
	TAG	String	private
	로그 메시지를 출력할 때 사용하는 변수		
	love_list	ArrayList<Long>	private
	검색 결과로 얻은 식당들의 찜수를 저장하는 변수		
	nowAddress	String	private
	현재 사용자의 위치를 나타내는 변수		
	index	int	private
	현재 처리 중인 식당의 인덱스를 나타내는 변수		
	drawerLayout	DrawerLayout	private
	액티비티 내에서 사용자에게 메뉴나 선택 사항을 제공하는 기능		
	rest_idList	ArrayList<String>	private
	검색 결과로 얻은 식당들의 고유 ID를 저장하는 변수		
	RES_ID	String	private
	현재 선택된 식당의 고유 ID를 가진 변수		
Operations	Name	Argument	Returns
	Description		
	onCreate	Bundle savedInstanceState	None
Operations	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	fetchOpendataIds	None	void
	웹에서 저장목록을 위한 데이터들을 가지고 오는 메서드		

Toplist			
Class	top10 list 기능을 제공하고 관리하는 class		
Description			
Attributes	Name	Type	Visibility
	Description		
	totallove	String	private
	list에 속할 가게의 찜수를 가진 변수		
	top_address_gu	String	private
	list에 속한 가게들의 구(주소)를 가진 변수		
	RES_ID	String	private
	인기 있는 식당의 고유 ID를 가진 변수		
Operations	Name	Argument	Returns
	Description		
	getTop_address_gu()	None	String
	list에 속한 가게의 주소를 가지고 오는 메서드		
	getWait_RED_ID()	None	String
	list에 속한 가게의 고유 ID를 가지고 오는 메서드		
	getTotallove()	None	String
	list에 속한 가게의 각각의 찜수를 가지고 오는 메서드		

3.4 Lovelist Class diagram

Love 리스트와 관련된 기능을 나타낸 CD이다. 이 클래스들은 사용자가 '찜'한 식당 리스트를 제공한다. 사용자는 '찜'한 목록에 들어가 현재까지 '찜'한 가게들을 살펴볼 수 있으며 '찜' 취소, 가게 상세페이지 보기 기능 등을 사용할 수 있다. '찜'한 목록들은 최근 저장한 순으로 저장되고 만약 '찜'을 취소한 후 다시 '찜'하기 버튼을 누르면 가장 최근 '찜'한 가게로 올라가게 된다.

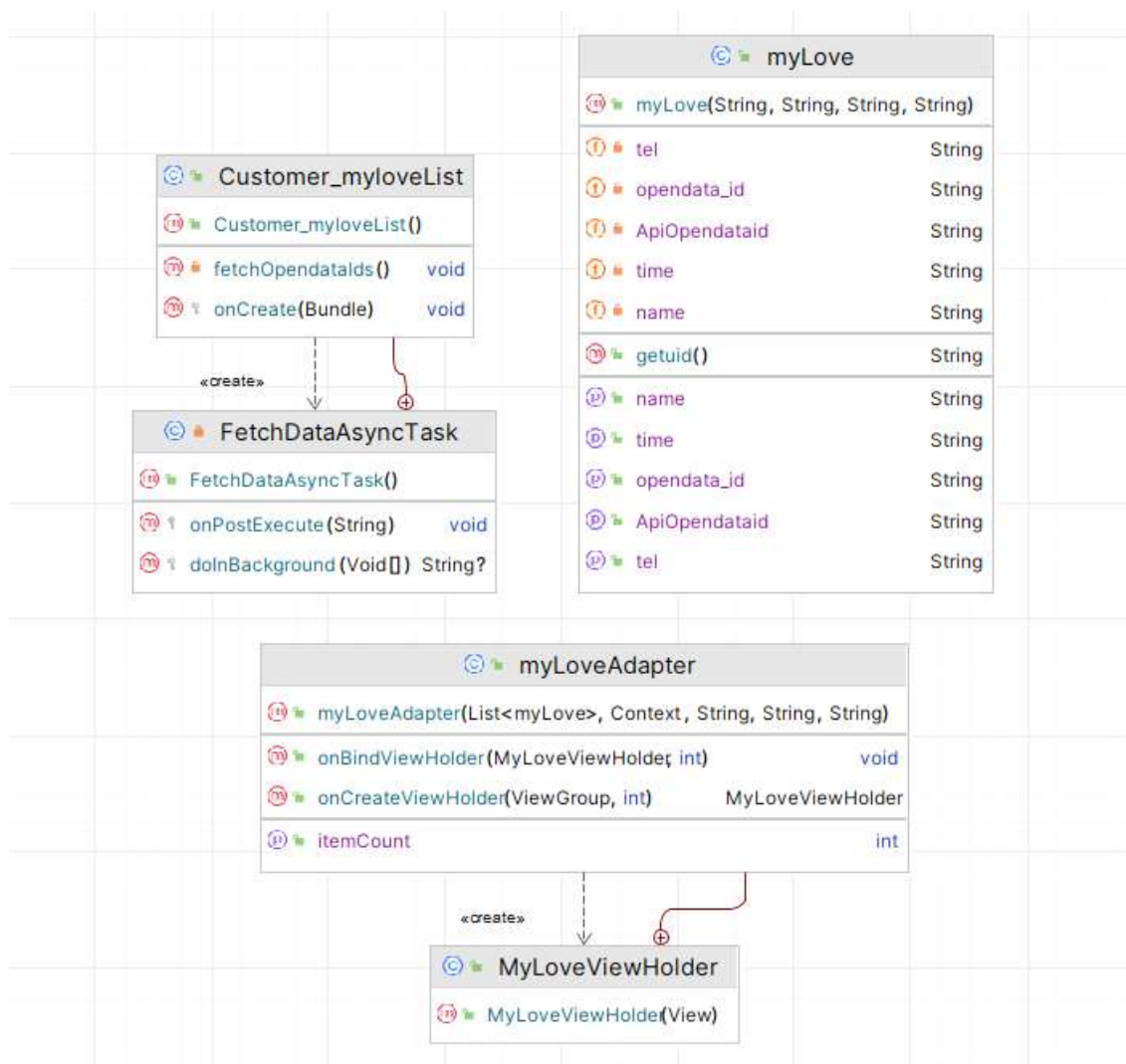


그림 3-4 Lovelist CD

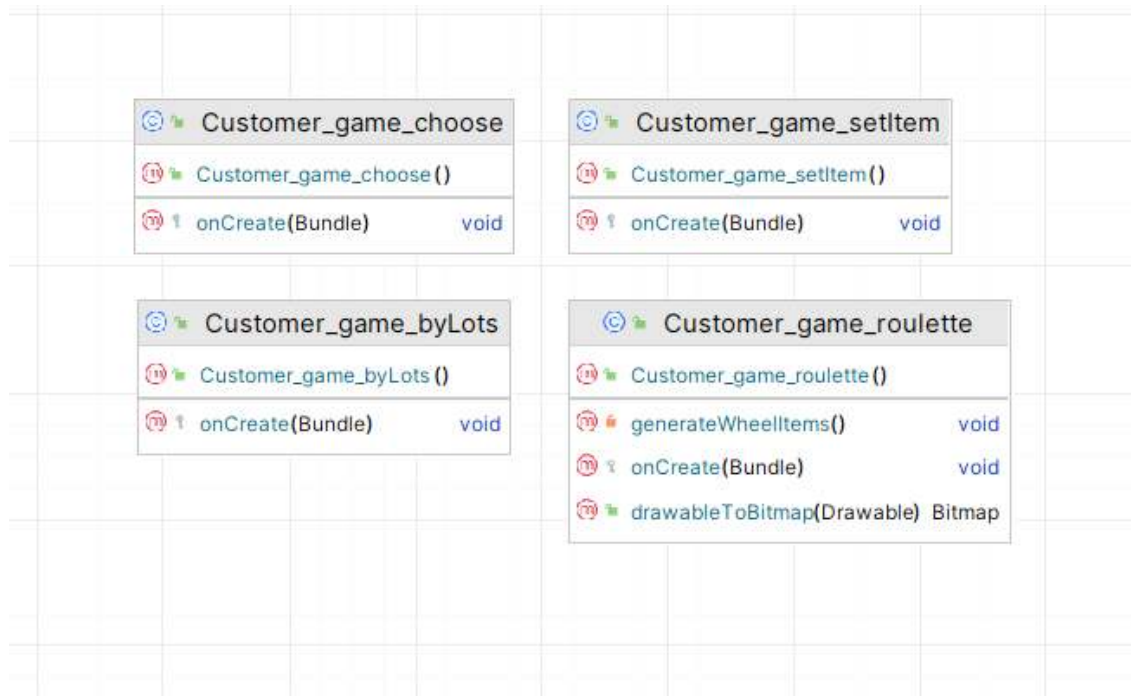
myLoveAdapter			
Class			
Description	My Love class가 ListView를 가져오기 위해 도움을 주는 class		
Attributes	Name	Type	Visibility
	Description		
	myLovelist	List<myLove>	private
	lovelist를 가지고 있는 변수		
	api_opendata_id	String	private
	사용자가 짚한 가게가 api에서의 번호를 가진 변수		
	opendata_id	String	private
	사용자가 짚한 가게의 opendata id를 가진 변수		
	context	Context	private
	mylovelist안의 내용을 가지고 있는 변수		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	uid	String	private
	사용자의 uid를 가진 변수		
Operations	Name	Argument	Returns
	Description		
	onCreateViewHolder	ViewGroup parent, int viewType	MyLoveViewHolder
	아이템 뷰를 위한 ViewHolder 객체를 생성하여 반환하는 메서드		
	onBindViewHolder	MyLoveViewHolder holder, int position	None
	ViewHolder기능을 이용해 그 위치(position)에 따라 list를 구성해주는 메서드		
	getItemCount	None	myLovelist.size
사용자가 저장한 가게의 개수를 확인할 수 있는 메서드			

myLove			
Class	사용자가 찜한 목록인 저장목록을 제공하고 관리하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	tel	String	private
	사용자가 찜한 가게의 전화번호를 가진 변수		
	ApiOpendataid	String	private
	사용자가 찜한 가게가 api에서의 번호를 가진 변수		
	uid	String	private
	사용자의 uid를 가진 변수		
	name	String	private
	사용자가 찜한 가게의 이름을 가진 변수		
	time	String	private
	현재 시간을 가진 변수		
	opendata_id	String	private
	사용자가 찜한 가게의 opendata id를 가진 변수		
	Name	Argument	Returns
	Description		
Operations	getTel	None	tel
	사용자가 찜한 가게의 전화번호를 가지고 오는 메서드		
	getApiOpendataid	None	apiopendataid
	사용자가 찜한 가게가 api에서의 번호를 가지고 오는 메서드		
	getName	None	name
	사용자가 찜한 가게의 이름을 가지고 오는 메서드		
	getuid	None	uid
	사용자의 uid를 가지고 오는 메서드		
	getOpendata_id	None	opendata_id
	사용자가 찜한 가게의 opendata id를 가지고 오는 메서드		
	getTime	None	time
	실제 시간을 가지고 오는 메서드		

Customer_myLoveList			
Class	사용자가 나의 저장 목록을 이용할 수 있도록 제공해주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	myLovelist	List<mylove>	private
	사용자의 저장 목록을 가지고 있는 배열		
	index	int	private
	나의 저장목록에 포함된 가게를 하나씩 불러오기 위해 사용하는 변수		
	opendata_id	String	private
	오픈데이터 ID를 나타내는 변수		
	myLoveAdapter	myLoveAdapter	private
	저장한 가게의 정보를 형식에 맞춰 보여주는 adapter		
	list	List<Map<String, Object>>	private
	파이어스토어의 cust_love_list컬렉션의 my_love_list 배열의 Map Value값으로 저장된 opendata_id값을 불러오기 위한 배열		
	recyclerView	RecyclerView	private
	사용자가 저장한 목록을 보여주는 recyclerView		
	uid	String	private
	사용자의 uid를 기록하는 변수		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	api_opendata_id	String	private
	사용자가 저장한 가게가 api에서의 번호를 가진 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	사용자의 id를 가져와 RecyclerView를 설정한 후 해당 RecyclerView에 어댑터인 myLoveAdapter를 연결한다. Firebase의 Firestore에서 특정 사용자의 즐겨찾기 목록을 가져오고 데이터를 성공적으로 가져오는 경우, onComplete 메서드가 호출한다. 그 후, onComplete 메서드 내에서는 작업의 성공 여부를 확인하고 성공적으로 데이터를 가져왔다면 해당 문서가 존재하는지 확인한다. 문서가 존재한다면, 해당 문서에서 즐겨찾기 목록을 가져와서 RecyclerView의 Adapter에 데이터를 전달하고, Adapter로 데이터가 변경되었음을 알린다. 만약 문서가 존재하지 않거나 데이터를 가져오는데 실패한 경우, 사용자에게 토스트 메시지를 통해 알려준다.		
	fetchOpendataIds	None	None
	웹에서 저장목록을 위한 데이터들을 가지고 오는 메서드		

3.5 Game Class diagram

메뉴선택을 위해 이용하는 게임 기능을 나타낸 Game CD이다. 제비뽑기 혹은 룰렛 돌리기를 통해 사용자는 결정하지 못했던 음식 메뉴를 결정할 수 있으며 만약 그 결과가 마음에 들지 않는 경우, 계속해서 그 게임을 반복할 수 있다. 메뉴를 결정한 후에는 홈 화면으로 나가 그 메뉴를 검색해 원하는 식당을 찾을 수 있다.



Customer_game_byLots			
Class	사용자가 이용할 수 있는 게임 중 제비뽑기 게임에 관한 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	txt_printMenu	TextView	private
	사용자가 이용할 게임을 나타내는 변수		
	btn_bylots	Button	private
	룰렛을 돌리는 Button		
	btn_reset	Button	private
	입력한 내용을 리셋하기 위해 이용하는 Button		
	et_input_menu	EditText	private
	룰렛에 들어갈 메뉴를 입력하는 변수		
	btn_inputMenu	Button	private
	룰렛에 들어갈 메뉴를 입력하고 룰렛에 넣는 Button		
	btn_home	Button	private
	게임 화면이 아닌 홈 화면으로 돌아가기 위해 이용하는 Button		
	txt_bylots	TextView	private
	제비뽑기 결과를 표시하는 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		

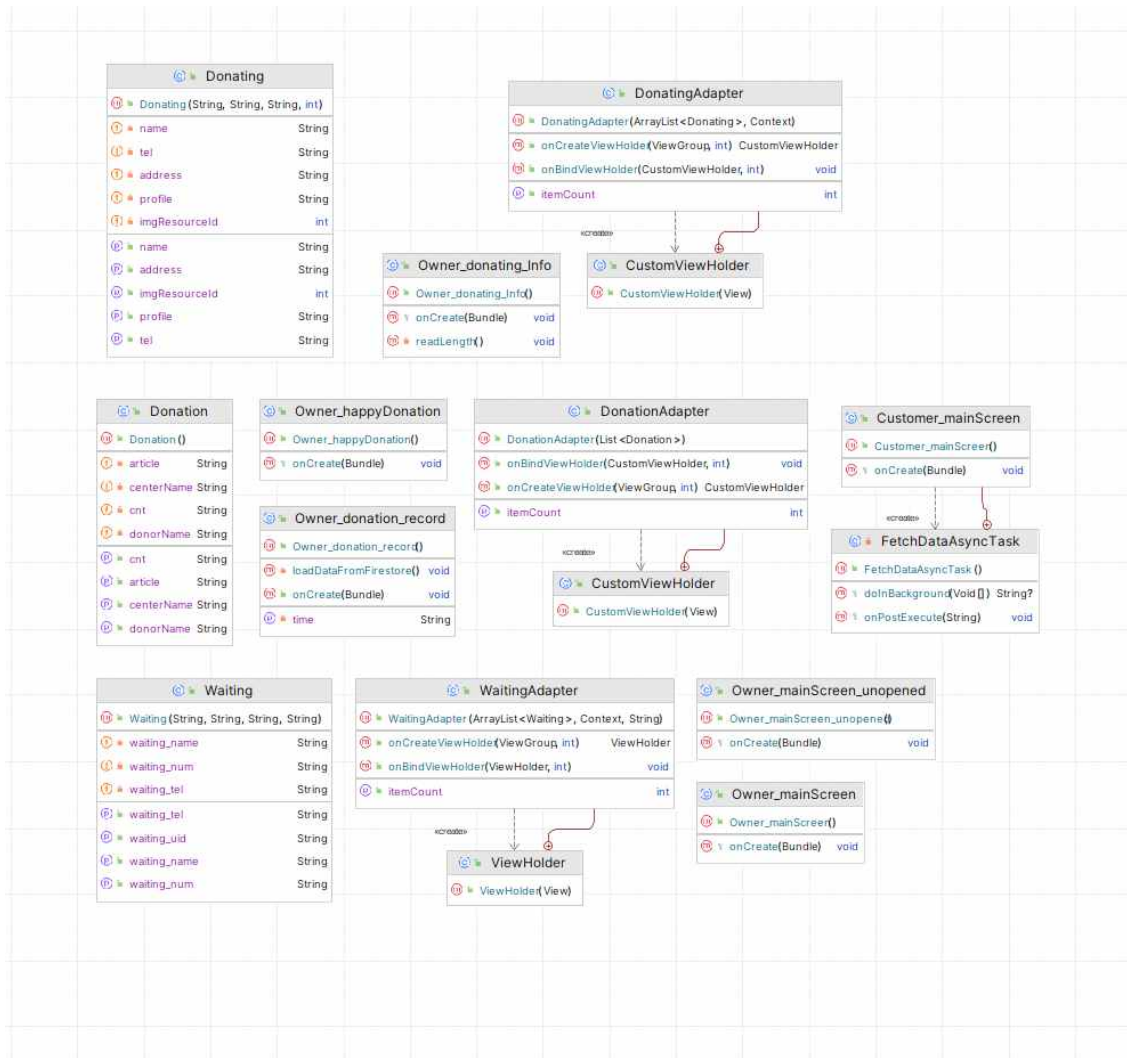
Customer_game_choose			
Class	사용자가 이용할 게임을 선택하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	btn_choose	ImageButton	private
	Roulette 게임중 룰렛을 선택하는 경우 이용하는 Button		
	btn_chooseBylots	ImageButton	private
	게임 중 제비뽑기를 선택하는 경우 이용하는 Button		
	txt_chooseGame	TextView	private
	사용자가 이용할 게임을 나타내는 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		

Customer_game_roulette			
Class	사용자가 이용할 수 있는 게임 중 룰렛 게임에 관한 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	wheelItem	List<WheelItem>	public
	룰렛판에 사용자가 입력한 메뉴들을 담고 있는 list		
	luckyWheel	LuckyWheel	private
	룰렛 기능을 담고 있는 변수		
	point	String	public
	룰렛의 결과를 나타내줄 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	rotateWheelTo	point	None
	룰렛을 돌리는 메서드		
	OnReachTarget	None	None
	룰렛 결과를 나타내주는 메서드		
	generate WheelItems	None	None
	룰렛을 꾸며주는 메서드		
	addWheelItems	wheelItems	None
	룰렛에 사용자가 입력한 메뉴를 넣는 메서드		
	drawable ToBitmap	Drawable drawable	None
	룰렛을 그리는 메서드		

Customer_game_setItem			
Class	사용자가 룰렛게임을 이용할 경우, 룰렛에 들어갈 item을 setting해주는		
Description	class		
Attributes	Name	Type	Visibility
	Description		
	item3	EditText	private
	룰렛에 적힐 메뉴를 기록할 변수3		
	btn_next	Button	private
	룰렛에 모든 메뉴를 입력하기 위해 누르는 Button		
	txt_menu_game	TextView	private
	사용자가 이용할 게임을 나타내는 변수		
	item1	EditText	private
	룰렛에 적힐 메뉴를 기록할 변수1		
	item	String[]	private
	룰렛에 적힌 모든 메뉴를 가지고 있는 string 배열		
	item4	EditText	private
	룰렛에 적힐 메뉴를 기록할 변수4		
	item2	EditText	private
	룰렛에 적힐 메뉴를 기록할 변수2		
	item5	EditText	private
	룰렛에 적힐 메뉴를 기록할 변수5		
	item6	EditText	private
	룰렛에 적힐 메뉴를 기록할 변수6		
Operations	Name	Argument	Returns
	Description		
	onCreate	Bundle savedInstanceState	None
시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드			

3.6 Owner Class diagram

사업자 회원이 이용할 수 있는 기능을 담은 Owner CD이다. 식당 사장님의 편의를 위한 서비스를 제공하며, 식당의 대기 관리, 영업 상태 관리, 기부 관련 기능 등을 제공한다. 또한, 이들은 Firebase 데이터베이스를 이용하여 실시간 정보를 제공하고, 이를 통해 식당의 운영을 보다 효율적으로 할 수 있게 돕는다.



Owner_mainScreen			
Class	Owner_mainScreen 클래스는 식당 주인의 메인 화면을 구현하는 클래스.		
Description	식당 주인은 대기 목록을 확인하고 대기창을 닫을 수 있다.		
Attributes	Name	Type	Visibility
	Description		
	menu	Button	private
	메뉴 Button		
	btn_res_close	Button	private
	대기창 닫기 Button		
	drawerLayout	DrawerLayout	private
	네비게이션 드로어를 포함하는 레이아웃		
	recyclerView	RecyclerView	private
	대기 목록을 표시하는 RecyclerView		
	adapter	WaitingAdapter	private
	대기 목록을 관리하는 어댑터		
	arrayList	ArrayList<Waiting>	private
	대기 목록 데이터를 담는 ArrayList		
	mstore	FirebaseFirestore	private
	Firebase Firestore 인스턴스		
	RES_ID	String	private
	상수로 정의된 식당 ID		
	uid	String	private
	사용자의 UID		
Operations	Name	Argument	Returns
	Description		
	onCreate	Bundle savedInstanceState	None
액티비티가 생성될 때 호출되는 메서드, 레이아웃 초기화 및 Button 클릭 이벤트 등을 설정, Firestore에서 대기 목록 데이터를 가져와 RecyclerView에 표시, 대기창 닫기 Button 클릭 시 대기 목록을 삭제하고 Owner_mainScreen_unopened 액티비티로 전환			

Owner_mainScreen_unopened			
Class Description	Owner_mainScreen_unopened 클래스는 식당 주인의 메인 화면(대기창이 아직 열리지 않은 상태)을 구현하는 클래스. 식당 주인은 대기창을 열 수 있다.		
Attributes	Name	Type	Visibility
	Description		
	menu	Button	private
	메뉴 Button		
	btn_res_open	Button	private
	대기창 열기 Button		
	drawerLayout	DrawerLayout	private
	네비게이션 드로어를 포함하는 레이아웃		
	mstore	FirebaseFirestore	private
	Firebase Firestore 인스턴스		
	res_id	String	private
	식당 ID		
	uid	String	private
	사용자의 UID		
Operations	Name	Argument	Returns
	Description		
	onCreate	Bundle savedInstanceState	None
액티비티가 생성될 때 호출되는 메서드, 레이아웃 초기화 및 Button 클릭 이벤트 등을 설정, 대기창 열기 Button 클릭 시 Firestore에 대기 목록을 추가하고 Owner_mainScreen 액티비티로 전환			

WaitingAdapter			
Class	Waiting class가 ListView를 가져오기 위해 도움을 주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	Waitinglist	ArrayList<Waiting>	private
	배열형태로서 줄서기한 사용자를 기록하는 변수		
	context	Context	private
	줄서기 한 사용자의 정보를 담은 변수		
	RES_ID	String	private static final
	사용자가 줄 서기한 식당을 기록하는 변수		
	mStore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	cust_uid	String	private
	대기자의 고유 식별자(ID)를 기록하는 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	viewGroup parent, int viewType	None
	ViewHolder기능을 이용하기 위해 ViewHolder를 생성하는 메서드		
	onBindViewHolder	WaitingAdapter.ViewHolder holder, int position	None
	생성된 ViewHolder에 data를 입력시키는 역할을 하는 메서드		
	getItemCount		
	줄서기 기능을 이용하는 손님이 있는지 그 수를 확인 할 수 있는 메서드		
	waitingAdapter	ArrayList<Waiting>waitinglist ,Context context, String RES_ID	None
	배열 형태의 Waitinglist를 연결하는 메서드		
	onSuccess	None	None
	줄서기 기능을 이용한 손님을 성공적으로 삭제 했는지 확인하는 메서드		
	onComplete	Task<DocumentSnapshot>	None
	줄서기 기능을 이용한 손님 명단을 계속해서 업데이트하는 메서드		
	onFailure	Exception e	None
	줄서기 기능을 이용한 손님이 제대로 삭제되지 않을 경우 예외처리를 위한 메서드		

Donating			
Class	기부할 기부센터의 보여주고 관리하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	imgResourceId	int	private
	기부센터마다 다른 이미지 사진을 위해 각각의 id를 저장하는 변수		
	address	String	private
	기부센터의 주소를 가지고 있는 변수		
	tel	String	private
	기부센터의 전화번호를 가지고 있는 변수		
	name	String	private
	기부센터의 이름을 가지고 있는 변수		
	Name	Argument	Returns
	Description		
Operations	getImgResourceId	None	None
	기부센터마다의 이미지 id를 가지고 오는 메서드		
	setImgResourceId	int	None
	기부센터마다의 이미지 id를 설정하는 메서드		
	setName	String	None
	기부센터의 이름을 설정하는 메서드		
	getName	None	String
	기부센터의 이름을 가지고 오는 메서드		
	getTel	None	String
	기부센터의 전화번호를 가지고 오는 메서드		
	setTel	String	None
	기부센터의 전화번호를 설정하는 메서드		
	setAddress	String	void
	기부센터의 주소를 설정하는 메서드		
	getAddress	None	String
	기부센터의 주소를 가지고 오는 메서드		

Donation			
Class	사용자의 기부 품목을 입력받고 저장하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	cnt	String	private
	기부 품목 개수를 가지고 있는 변수		
	article	String	private
	기부 품목을 가지고 있는 변수		
	centerName	String	private
	기부할 센터의 이름을 가지고 있는 변수		
	donorName	String	private
	기부한 사용자의 이름을 가지고 있는 변수		
	Name	Argument	Returns
	Description		
Operations	getDonorName	None	String
	기부한 사용자의 이름을 가지고 오는 메서드		
	setCenterName	String	None
	기부할 센터의 이름을 설정하는 메서드		
	setArticle	String	None
	기부품목의 이름을 설정하는 메서드		
	getCnt	None	int
	기부 품목을 가지고 오는 메서드		
	setDonorName	String	None
	기부한 사용자의 이름을 설정하는 메서드		
	getArticle	None	String
	기부품목의 이름을 가지고 오는 메서드		
	getCenterName	String	String
	기부할 센터의 이름을 가지고 오는 메서드		
	setCnt	String	
	기부 품목을 설정하는 메서드		

Waiting			
Class	줄서기 기능을 제공하고 관리하는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	waiting_queue	int	private
	줄서기 기능을 이용하는 사용자 수를 기록하기 위한 변수		
	waiting_num	String	private
	줄서기 기능을 이용하는 사용자를 기록하기 위한 변수		
	waiting_name	String	private
	사용자가 줄서기 기능을 이용하기 위해 입력한 이름을 기록하기 위한 변수		
	waiting_tel	String	private
	사용자가 줄서기 기능을 이용하기 위해 입력한 전화번호를 기록하기 위한 변수		
	cust_uid	String	private
	고객 UID를 저장하기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	getWaiting_tel	None	None
	waiting_tel 변수를 가지고 오는 메서드		
	getWaiting_uid	None	None
	cust_uid 변수를 가지고 오는 메서드		
	getWaiting_name	None	None
	waiting_name 변수를 가지고 오는 메서드		
	getWaiting_num	None	None
	waiting_num 변수를 가지고 오는 메서드		

DonationAdapter			
Class	Donation class가 ListView를 가져오기 위해 도움을 주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	donationList	List<Donation>	private
	기부할 센터 정보를 가지고 있는 list 배열		
	Name	Argument	Returns
	Description		
Operations	onCreateView Holder	ViewGroup parent, int position	None
	아이템 뷰를 위한 ViewHolder 객체를 생성하여 반환하는 메서드		
	onBindViewHolder	CustomViewHolder, int	None
	firebase에서 가지고 온 정보를 변수에 입력시켜주는 메서드		
	getItemCount	None	int
	기부센터의 개수를 가지고 오는 메서드		

DonatingAdapter			
Class	Donating class가 ListView를 가져오기 위해 도움을 주는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	donatingList	ArrayList<Donating>	private
	기부정보를 가지고 있는 Donating ArrayList		
	context	Context	private
	현재 화면의 상태를 나타내는 변수		
	Name	Argument	Returns
	Description		
Operations	getItemCount	None	int
	RecyclerView에 표시될 항목의 수를 반환하는 메서드		
	onCreateView Holder	ViewGroup, int	None
	아이템 뷰를 위한 ViewHolder 객체를 생성하여 반환하는 메서드		
	onBindViewHolder	CustomViewHolder, int	None
	ViewHolder의 뷰에 데이터를 설정한 후 각 항목에 대한 클릭 리스너를 설정하는 메서드		

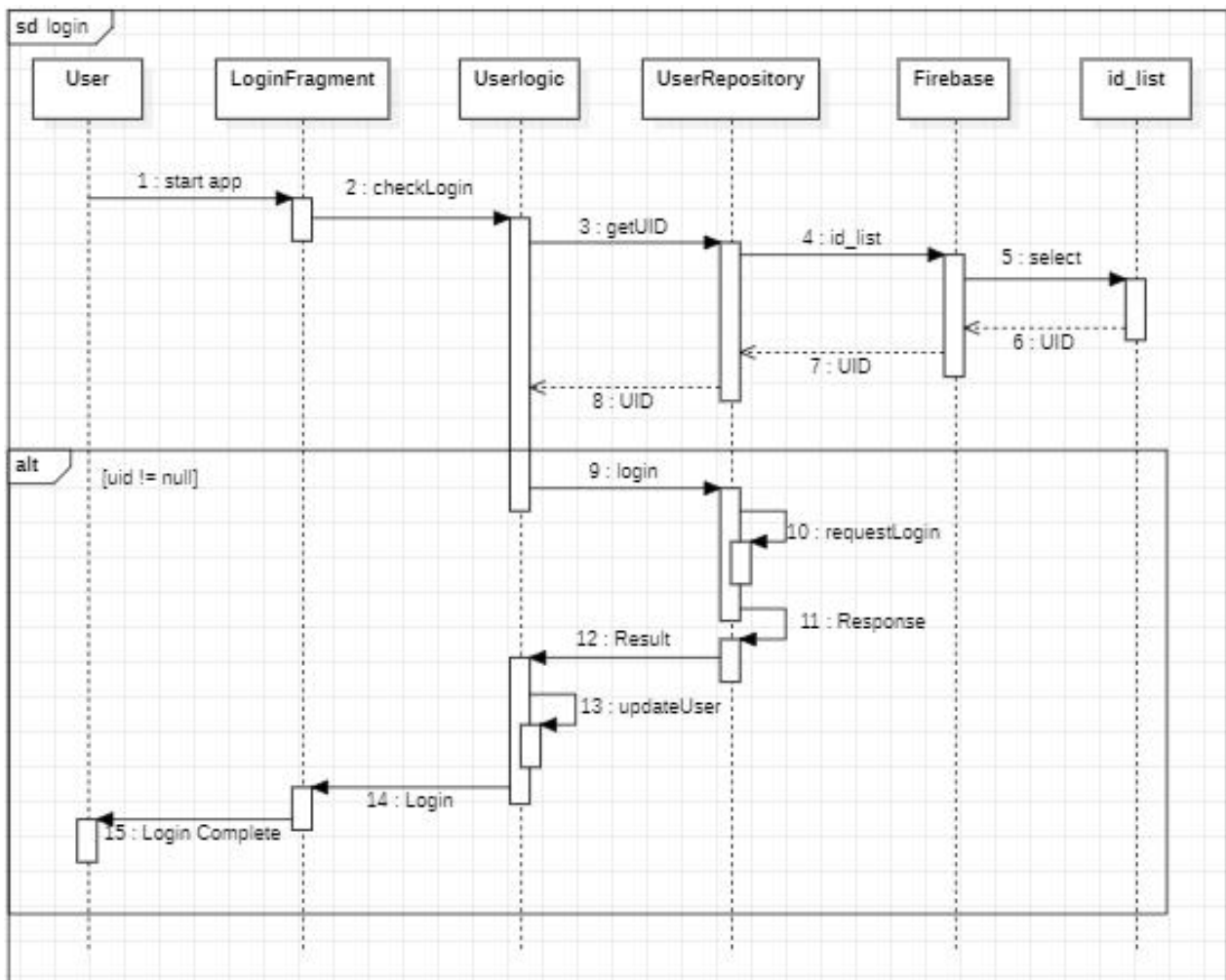
Owner_happyDonation			
Class			
Description			
	Name	Type	Visibility
	Description		
Attributes	recyclerView	RecyclerView	private
	기부할 센터를 보여주는 recyclerView		
	donationAdapter	DonatingAdapter	private
	기부할 센터의 정보를 형식에 맞춰 보여주는 adapter		
	arrayList	ArrayList <Donating>	private
	기부할 센터 list 배열		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		

Owner_donating_Info			
Class	사용자가 기부할 물품 정보를 입력하고 기부 신청을 할 수 있는 class		
Description			
	Name	Type	Visibility
	Description		
Attributes	name_sender	EditText	private
	기부자의 이름을 가진 변수		
	cnt_article	EditText	private
	물품 개수를 가진 변수		
	tel_donating01	EditText	private
	전화번호 첫 번째 부분을 가진 변수		
	article_to_donate	EditText	private
	기부할 물품을 가진 변수		
	happyDonation	Owner_happydonation	private
	기부 센터 이름을 가진 변수		
	list_cnt	int	private
	기부횟수를 가진 변수		
	button	Button	private
	기부를 신청할 수 있는 버튼		
	tel_donating02	EditText	private
	전화번호 두 번째 부분을 가진 변수		
	mstore	FirebaseFirestore	private
	Firebase store에 접근하기 위한 변수		
	mAuth	FirebaseAuth	private
	Firebase Authentication에 접근하기 위한 변수		
	Name	Argument	Returns
	Description		
Operations	onCreate	Bundle savedInstanceState	None
	기부 신청 버튼을 누르면 입력된 정보를 가져와서 빈칸이 없는 경우에만 Firebase Firestore에 저장한다. 저장하는 정보에는 기부자 이름, 기부할 물품, 물품 개수, 전화번호를 사용자 별로 기부 내역을 저장하는 donating_uid_list 컬렉션에 저장하고, 혹은 기부 센터 별로 기부 내역을 저장하는 children_center_list 컬렉션에 저장한다. 또한, 기부 횟수를 확인하고 기부 횟수가 3회 이상이면 해당 사용자를 '기부천사'로 표시한다. 이는 id_list 컬렉션 내 해당 사용자의 문서에 'donatingAngel' 필드를 추가하여 표시한다.		
	readLength	None	None
	기부횟수를 측정하는 변수		

Owner_donation_record			
Class	Owner_donation_record 클래스는 사용자가 기부한 기부내역을 현재 시간을 기준으로 알아보고 횟수와 자세한 기부내용을 알 수 있는 class이다		
Description	Name	Type	Visibility
	Description		
Attributes	date_record _donation	TextView	private
	기부한 날짜를 기록하는 변수		
	countDonationTextVie w	TextView	private
	기부한 횟수를 기록하는 변수		
	mdate	Date	private
	오늘의 날짜를 기록하는 변수		
	donationList	FirebaseFirestore	private
	기부내역을 기록하는 list		
	mNow	long	private
	현재 시간을 기록하는 변수		
	uid	String	private
	사용자의 uid를 기록하는 변수		
	tz	TimeZone	private
	현재 위치한 국가의 시간대를 가져오는 기능		
	donationAdapter	DonationAdapter	private
	기부내역을 형식에 맞추서 보여주는 기능		
	mFormat	SimpleDateFormat	private
	시간을 나타내는 형식		
	mAuth	FirebasAuth	private
	Firebase authentication에 접근해 인증받기위한 변수		
	db	FirebaseFirestore	private
	Firebase Firestore에 접근해 인증받기 위한 변수		
Name	Argument	Returns	
	Description		
Operations	onCreate	Bundle savedInstance State	None
	시스템이 실행될 때 항상 같이 실행되는 메서드로써 Bundle 상태가 null인지 확인하고 null일 경우 없어진 이전의 instance를 복원하지 않고 새로운 instance를 생성시켜주는 메서드		
	getTime	None	String
	실제 시간을 가지고 오는 메서드		
	loadDataFrom Firedtore	None	None
	Firebase에 저장되어 있는 기부내역을 가져와 출력형식에 맞춰 출력될 수 있게 하는 메서드		

4. Sequence diagram

Login

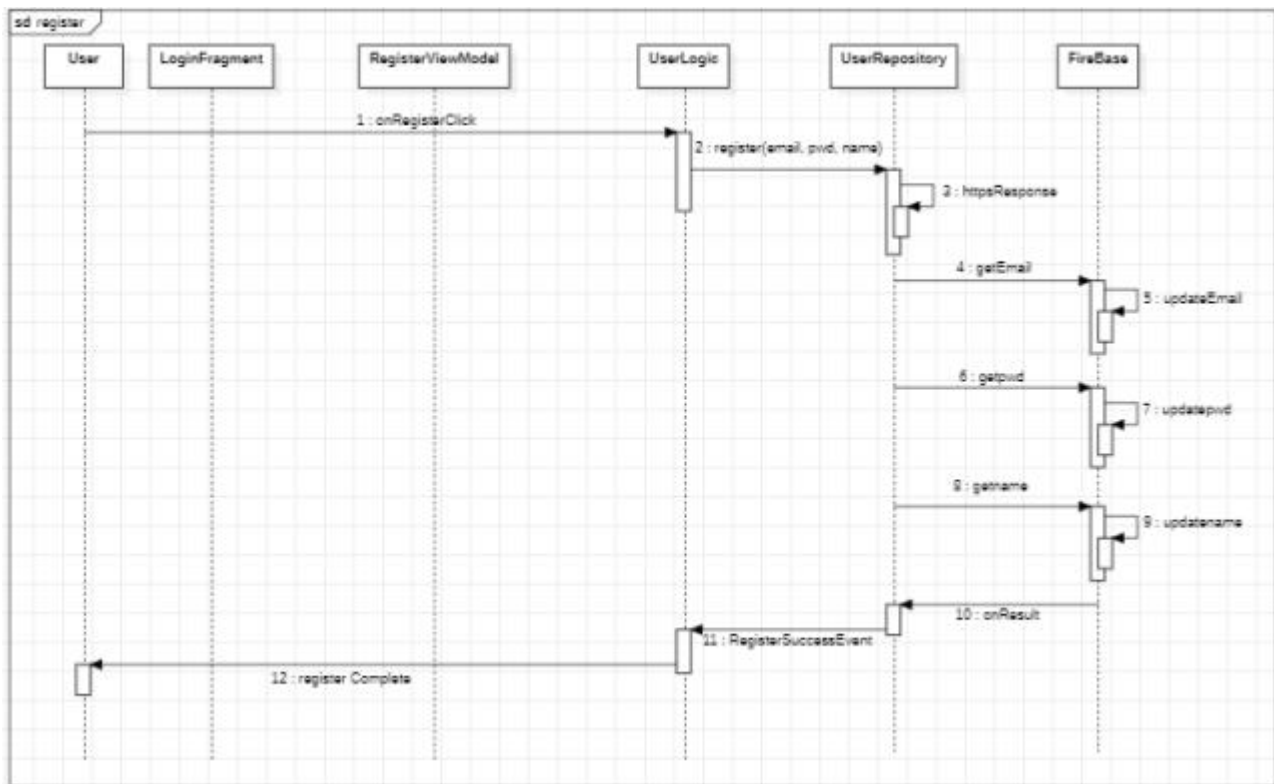


[그림 4-1] Login SD

사용자가 시스템에 로그인하는 Use Case를 나타내는 Sequence Diagram이다. User Class Description에서 <Use Case #2>의 경우이다.

앱이 시작되면서 기능을 실행한다. 사용자는 본인의 아이디와 비밀번호를 입력하고 '로그인' 버튼을 누른다. 사용자의 계정과 일치하는 계정이 FireStore에 등록이 되어있는 경우, 메뉴 검색 화면으로 전환되고 사용자에게 다음 기능들을 제공한다. 일치하는 계정이 존재하지 않을 경우, 계정이 존재하지 않는다는 메시지를 출력하고 고객은 다시 로그인을 시도할 수 있다.

Register

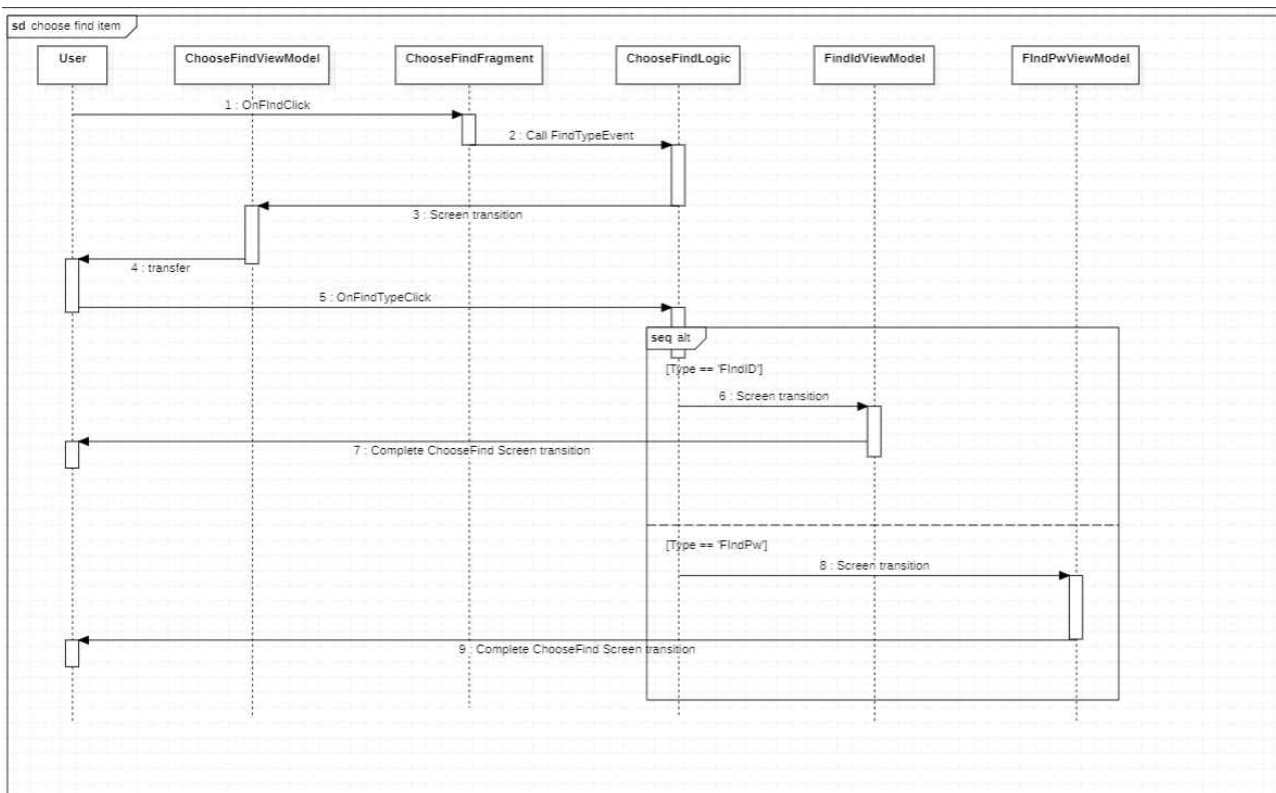


[그림 4-2] Register SD

사용자가 시스템에 회원가입하는 Use Case를 나타내는 Sequence Diagram이다. Class Description에서 <Use Case #1>의 경우이다.

시작화면에서 '회원가입' 버튼을 입력하면 실행된다. 사용자는 일단 사업자와 고객 중 본인의 유형을 선택한다. 다음으로 고객은 아이디(이메일), 비밀번호, 이름, 전화번호를 입력하고, 사업자는 식당고유번호를 포함한 여러 정보를 입력한다. 이와 같이 사용자는 해당 유형에 맞는 값을 입력한 후, '회원가입' 버튼을 누르면 회원가입 요청이 이루어지고, 사용자에게 고유한 uid를 부여해 Firebase(Authentication, Firestore)에 사용자의 계정 정보를 uid를 기준으로 저장한다. 이후, 로그인 화면으로 전환된다.

Choose Find ID/PW

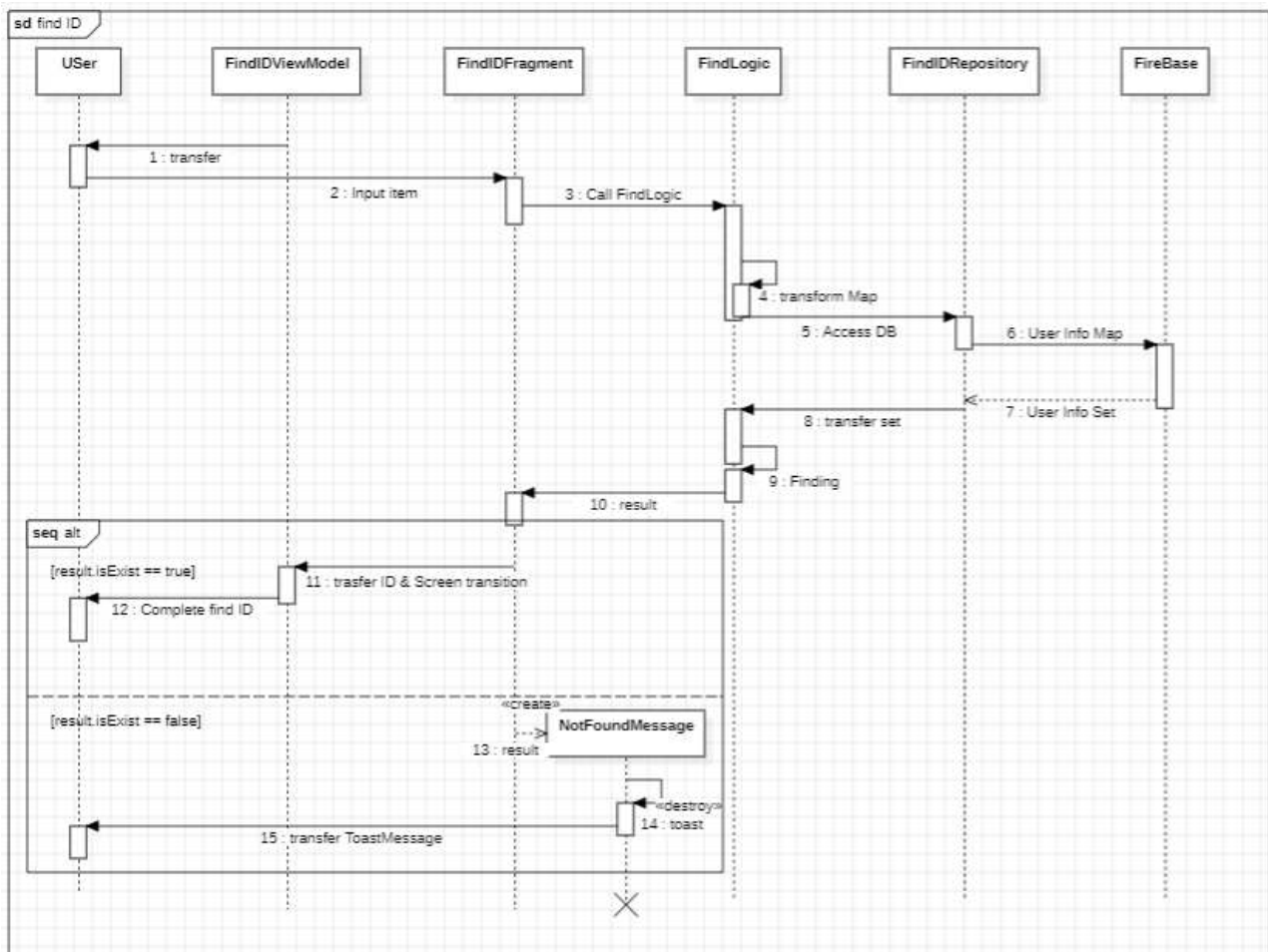


[그림 4-3] Choose Find ID/PW

사용자가 아이디, 비밀번호 중 어떤 유형의 정보를 찾을지 고르는 기능의 Use Case를 나타내는 Sequence Diagram이다.

사용자가 전 화면에서 아이디와 비밀번호 중 '아이디 찾기' 버튼을 입력하면 실행된다. 아이디를 찾는 화면에서 정해진 양식에 따라 이름, 전화번호를 입력하고 '찾기' 버튼을 입력하면 해당 정보와 일치하는 사용자를 Firebase(Authentication, Firestore)에서 검색한다. 일치하는 사용자가 있는 경우 화면이 전환되고 사용자가 찾고자 하던 아이디를 출력한다. '로그인 화면으로 돌아가기' 버튼을 누르면 사용자는 로그인 화면으로 전환되어 로그인을 진행할 수 있다. 일치하는 사용자가 없는 경우 사용자 정보가 없다는 메시지가 출력된다. (이는 alt문에 대한 설명이다.)

Find ID

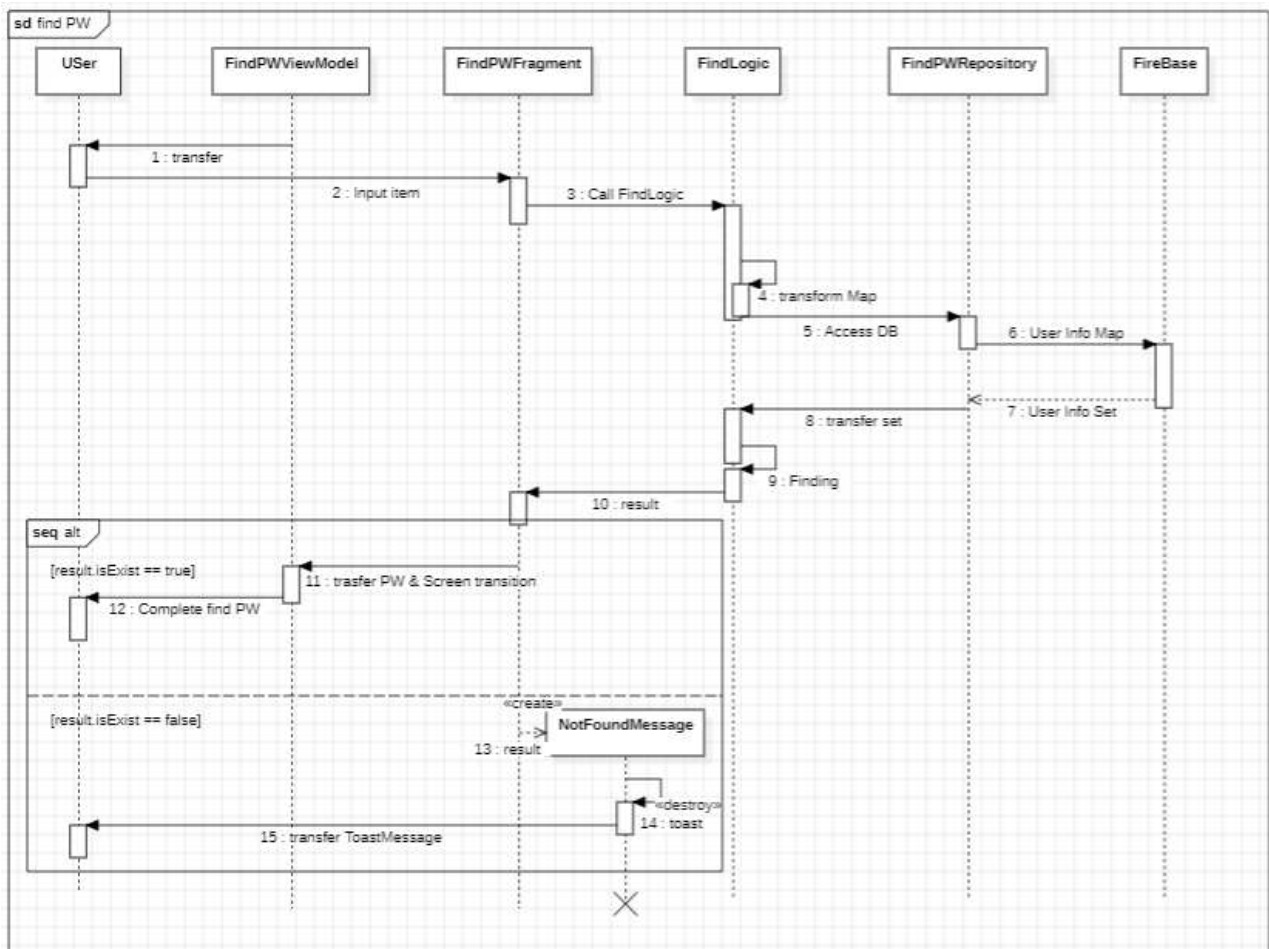


[그림 4-4] Find ID SD

사용자가 아이디를 분실했을 경우 찾는 기능의 Use Case를 나타내는 Sequence Diagram이다.

사용자가 전 화면에서 아이디와 비밀번호 중 ‘아이디 찾기’ 버튼을 입력하면 실행된다. 아이디를 찾는 화면에서 정해진 양식에 따라 이름, 전화번호를 입력하고 ‘찾기’ 버튼을 입력하면 해당 정보와 일치하는 사용자를 FireBase(Authentication, FireStore)에서 검색한다. 일치하는 사용자가 있는 경우 화면이 전환되고 사용자가 찾고자 하던 아이디를 출력한다. 로그인 화면으로 돌아가기 버튼을 누르면 사용자는 로그인 화면으로 전환되어 로그인을 진행할 수 있다. 일치하는 사용자가 없는 경우 사용자 정보가 없다는 메시지가 출력된다. (이는 alt문에 대한 설명이다.)

Find PW

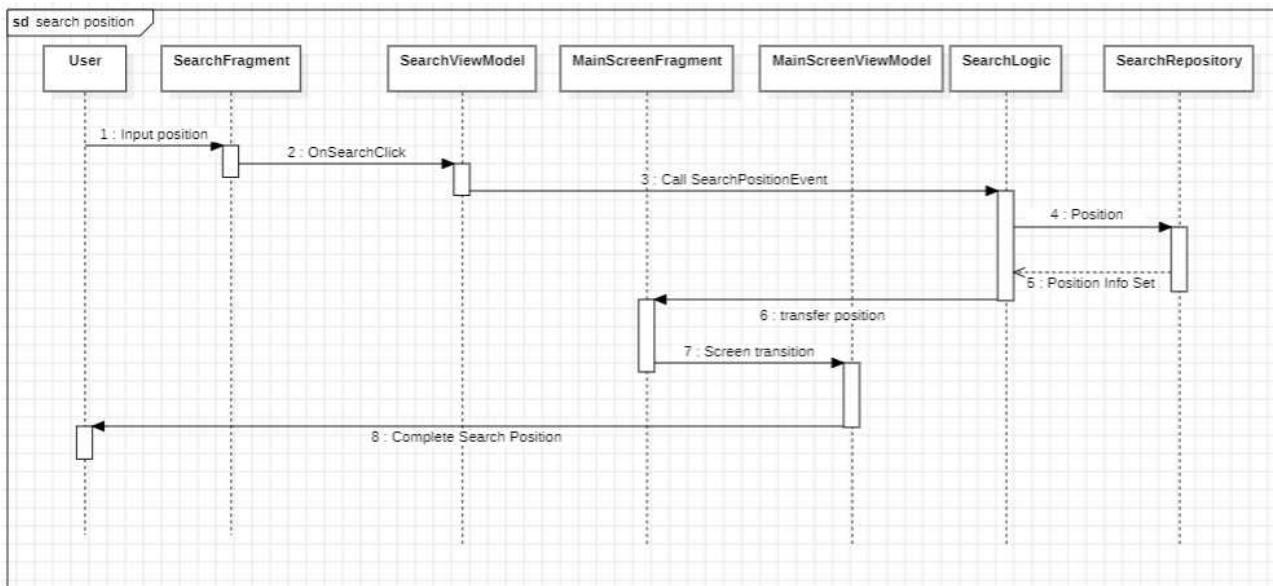


[그림 4-5] Find PW SD

사용자가 비밀번호를 분실했을 경우 찾는 기능의 Use Case를 나타내는 Sequence Diagram이다.

사용자가 전 화면에서 아이디와 비밀번호 중 '비밀번호 찾기' 버튼을 입력하면 실행된다. 비밀번호를 찾는 화면에서 정해진 양식에 따라 이름, 전화번호, 아이디를 입력하고 '찾기' 버튼을 입력하면 해당 정보와 일치하는 사용자를 FireBase(Authentication, Firestore)에서 검색한다. 일치하는 사용자가 있는 경우 화면이 전환되고 사용자가 찾고자 하던 비밀번호를 출력한다. '로그인 화면으로 돌아가기' 버튼을 누르면 사용자는 로그인 화면으로 전환되어 로그인을 진행할 수 있다. 일치하는 사용자가 없는 경우 사용자 정보가 없다는 메시지가 출력된다. (이는 alt문에 대한 설명이다.)

Search Position

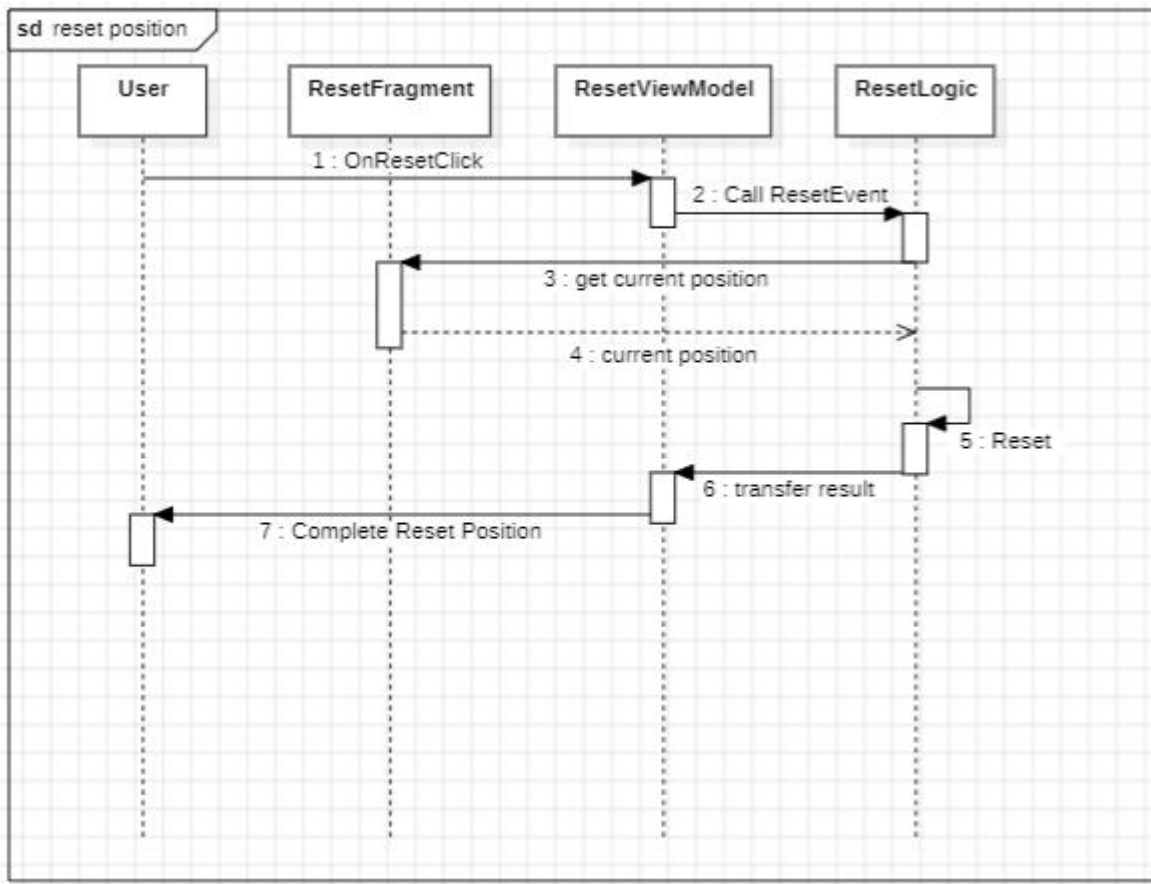


[그림 4-6] search position SD

고객이 검색하고자 하는 위치의 맛집을 검색하는 use case를 나타내는 sequence diagram이며, <Use Case #3-1a>의 경우이다.

고객이 위치를 입력하면 이를 Input으로 받는다. 사용자가 '검색' 버튼을 누르면 검색 위치를 변수로 저장한 뒤에 해당 위치를 가진 맛집들을 API 리스트에서 Set으로 가져와 리사이클러뷰로 출력해 메인 화면으로 화면을 전환한다. 메인 화면에는 해당 위치를 기준으로 한 지도와 Set으로 가져온 맛집들을 추천 리스트로 제공한다. 사용자가 상세정보를 보기 원하는 식당을 클릭하면 시스템은 해당 식당에 대한 상세정보를 제공한다.

Reset Position

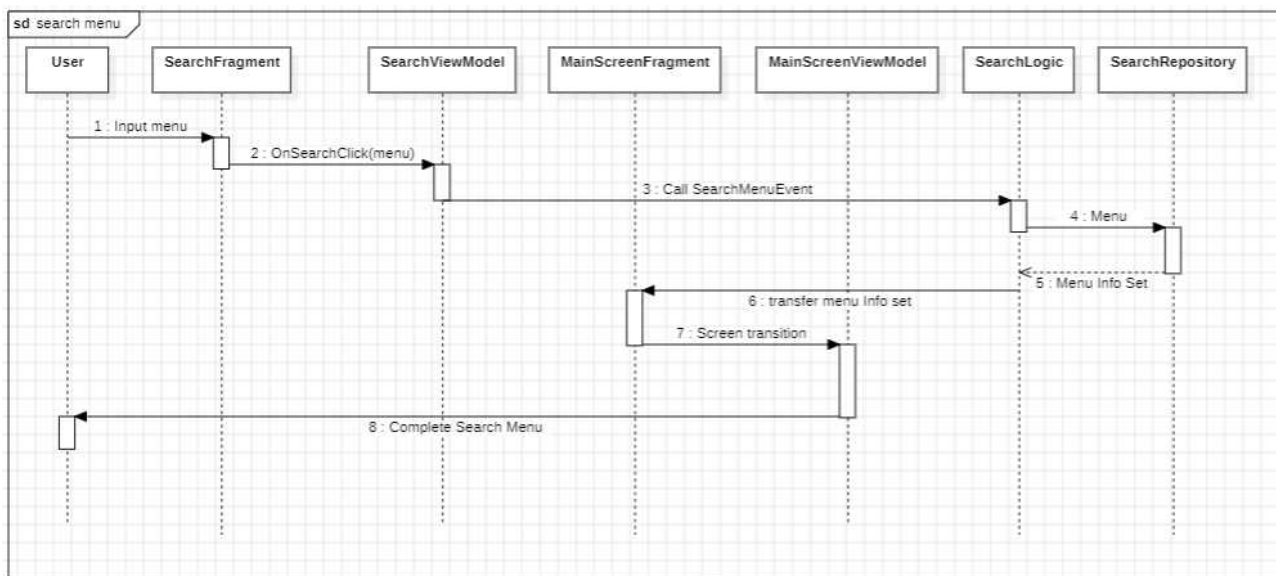


[그림 4-7] reset position SD

고객의 현 위치를 제공하는 기능의 use case를 나타내는 sequence diagram이며, <Use Case #3-1b>의 경우이다.

고객이 reset을 의미하는 아이콘 버튼을 입력하면 시스템은 현재 접속 중인 사용자의 현 위치를 받아와 위치를 출력하는 텍스트 박스에 띄운다. 고객이 제공받은 현 위치에 대해 '검색' 버튼을 입력하면 시스템은 해당 위치에 대한 맛집 정보를 메인 화면에서 제공한다.

Search menu

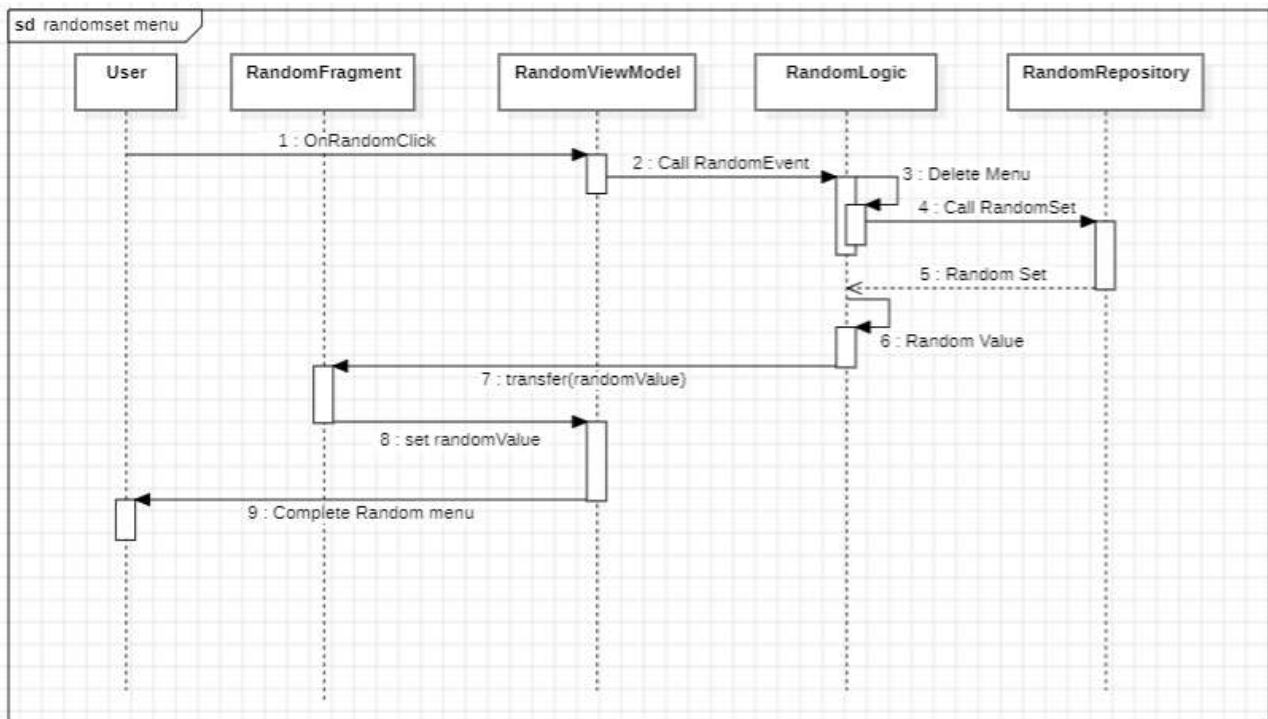


[그림 4-8] search menu SD

고객이 검색하고자 하는 메뉴를 가진 맛집들을 검색하는 use case를 나타내는 sequence diagram이며, <Use Case #3-2>의 경우이다.

고객이 메뉴를 입력하고 '검색' 버튼을 클릭하면 해당 메뉴를 입력받는다. 해당 메뉴를 변수로 저장한 뒤에 맛집 API 리스트에서 메뉴 변수로 검색해 일치하는 해당 메뉴를 가진 맛집들을 Set으로 들고 온 후, 메인 화면으로 화면을 전환한다. 시스템은 Set으로 가져온 해당 메뉴를 가진 맛집들을 메인 화면 내 추천 리스트로 제공한다.

Random menu

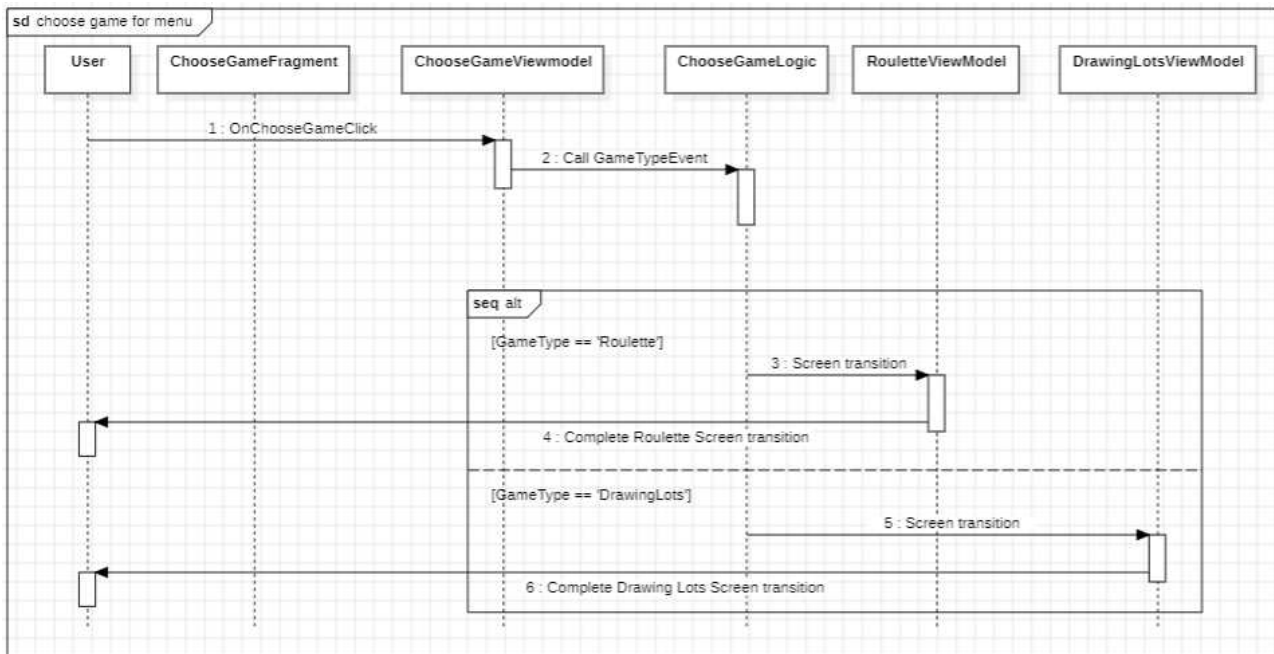


[그림 4-9] random menu SD

고객에게 랜덤으로 메뉴를 추천해주는 기능의 use case를 나타내는 sequence diagram이며, <Use Case #3-2b>의 경우이다.

고객이 '랜덤' 버튼을 누르면 우선 메뉴 검색창 내의 text를 삭제한다. 프로그램 내의 랜덤을 위한 메뉴 리스트 중 랜덤으로 하나의 메뉴를 가져와 고객에게 제공한다. 고객은 랜덤으로 추천받은 메뉴에 대해 '검색' 버튼을 입력하면 시스템은 메인 화면에서 해당 메뉴에 대한 맛집들을 제공한다.

Choose game for menu

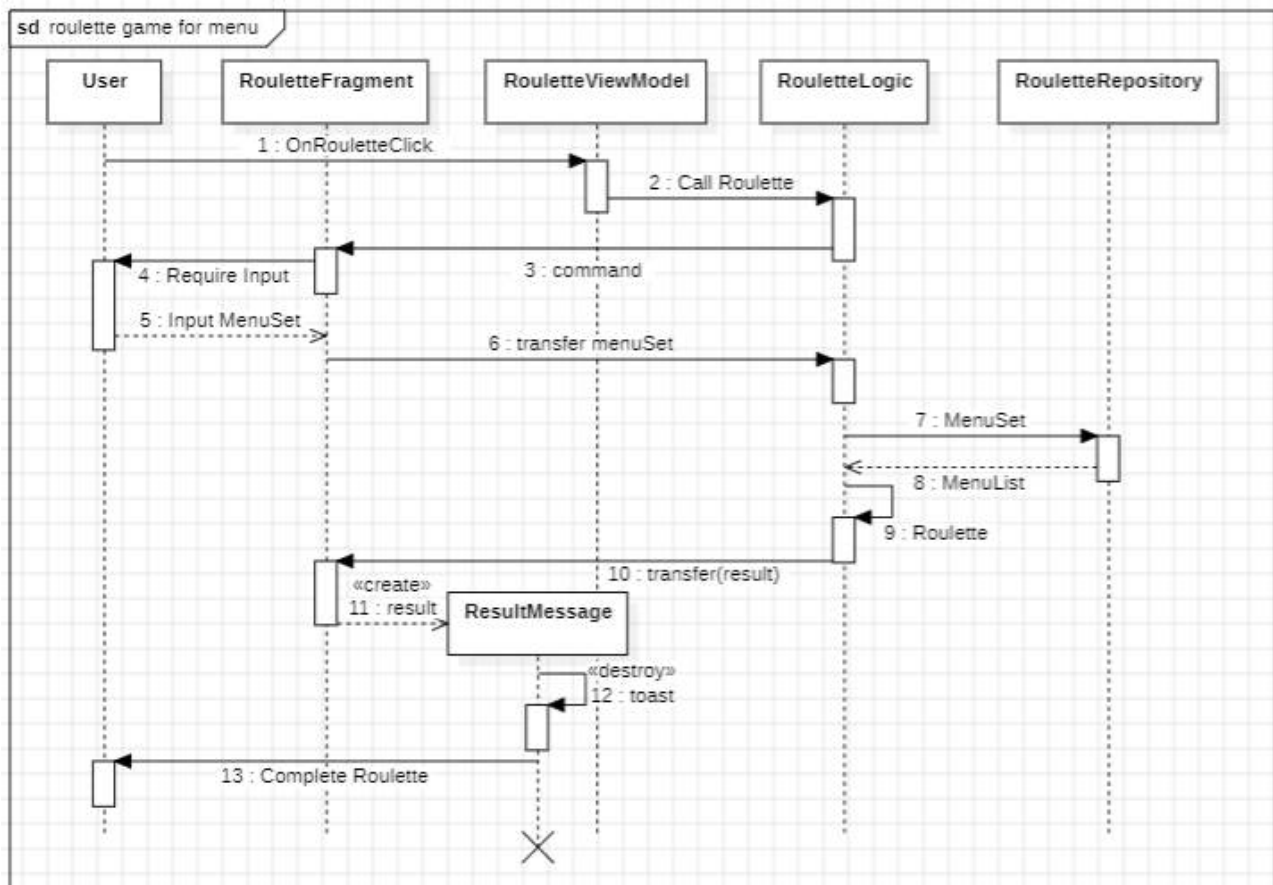


[그림 4-10] choose game for menu SD

고객이 고객의 메뉴를 결정하기 위한 게임 type을 고르는 use case를 나타내는 sequence diagram이며, <Use Case #3-2c>의 경우이다.

고객이 '메뉴 정하기' 버튼을 입력하면 게임 type을 정할 수 있는 화면으로 전환된다. 고객이 룰렛 버튼을 입력하면 룰렛 게임에 대한 화면으로 전환되고, 제비뽑기 버튼을 입력하면 제비 뽑기에 대한 화면으로 전환된다. (이는 alt문에 대한 설명이다.)

Roulette game for menu

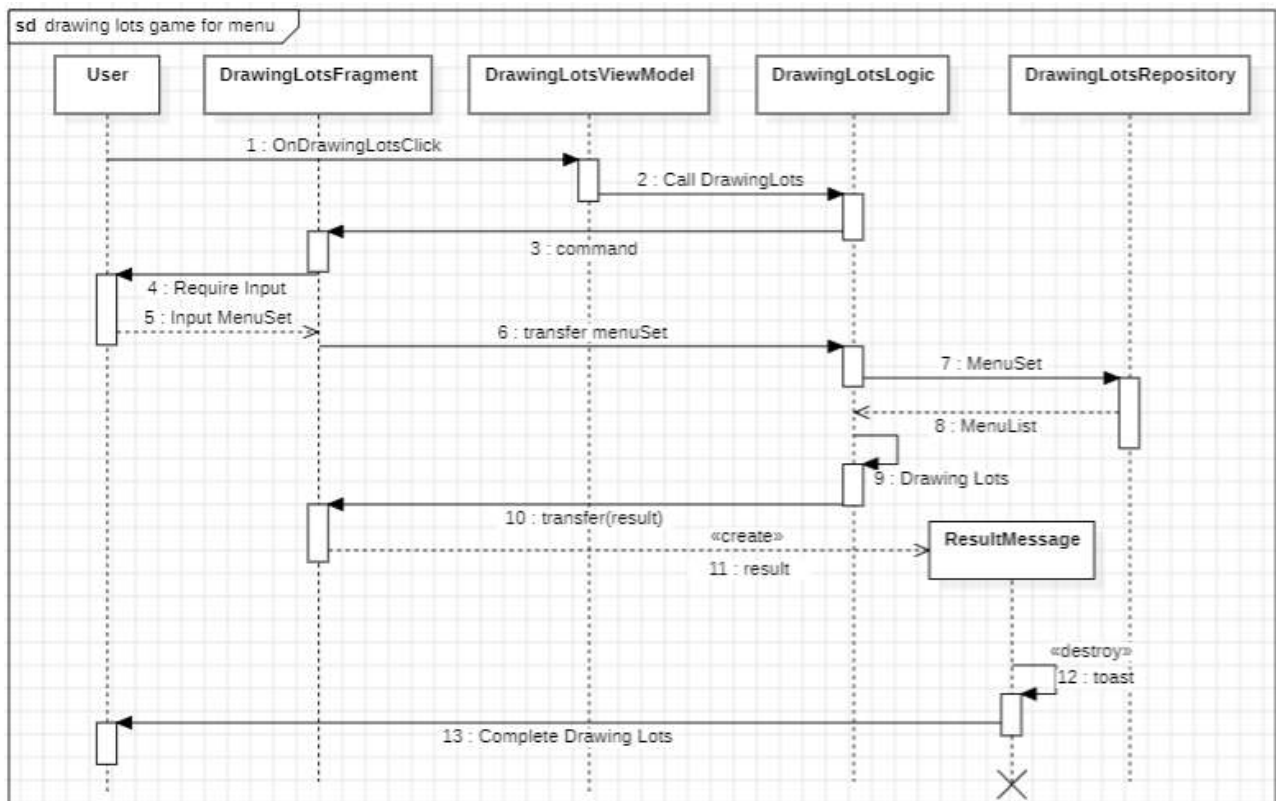


[그림 4-11] roulette game for menu SD

고객이 메뉴를 결정하기 위한 룰렛 게임의 use case를 나타내는 sequence diagram이며, <Use Case #3-2c2>의 경우이다.

고객이 룰렛을 돌릴 6가지 메뉴를 입력하면 해당 메뉴들을 Input Set으로 받는다. 고객이 '다음' 버튼을 누르면 고객이 입력한 6가지 메뉴가 적힌 룰렛이 있는 화면으로 전환된다. 고객이 '룰렛 돌리기' 버튼을 누르면 게임이 시작된다. 룰렛 게임의 결과로 나온 하나의 메뉴가 toast 메시지로 출력된다.

Drawing lots game for menu

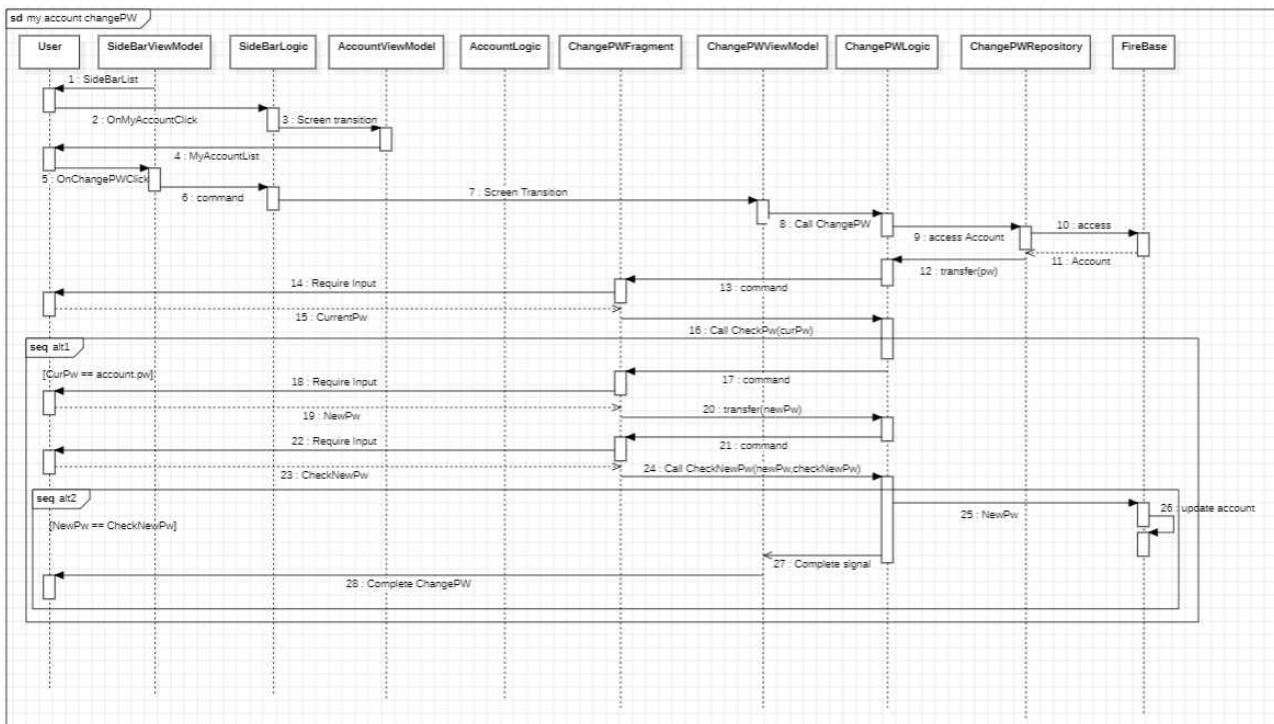


[그림 4-12] drawing lots game for menu SD

고객이 메뉴를 결정하기 위한 제비뽑기 게임의 use case를 나타내는 sequence diagram이며, <Use Case #3-2c2>의 경우이다.

고객이 제비뽑기를 위한 메뉴를 개수 상관없이 입력한다. 고객이 입력한 만큼 리스트에 저장된다. 고객이 '제비뽑기' 버튼을 누르면 고객이 입력한 메뉴 리스트 중 하나의 메뉴를 랜덤으로 뽑아 결괏값으로 저장한다. 제비뽑기 게임의 결과로 나온 하나의 메뉴가 toast 메시지로 출력된다.

My account change password



[그림 4-13] my account change password SD

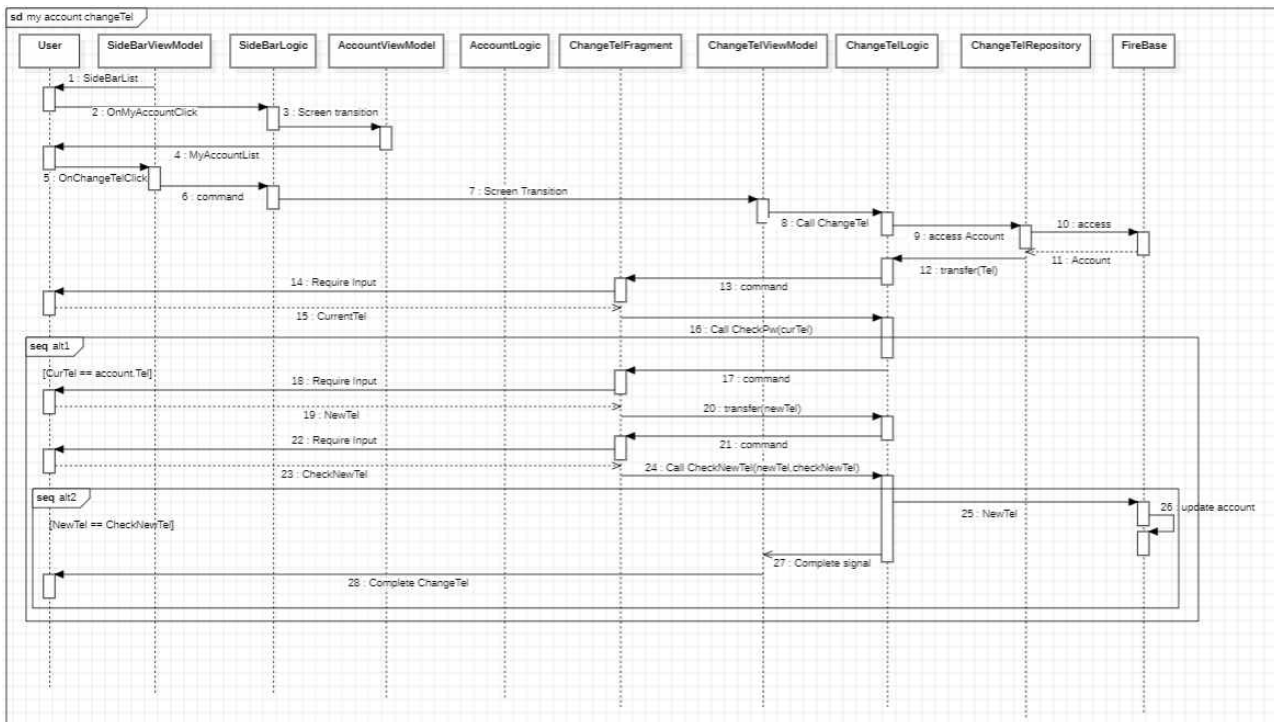
사용자의 비밀번호를 변경하는 기능의 use case를 나타내는 sequence diagram이며, <Use Case #9-3>의 경우이다.

사용자가 사이드바에서 '내 계정'을 클릭하면 계정에 대한 기능을 제공하는 화면으로 전환된다. 계정에 대한 기능 중 '비밀번호 변경' 버튼을 클릭하면 해당 화면으로 전환된다. 시스템은 현재 접속 중인 사용자의 uid로 FireBase 내 사용자의 정보에 접근해 현재 비밀번호를 들고 온다.

비밀번호 변경을 위해 사용자가 기존 비밀번호를 입력해 '확인' 버튼을 누른다. 입력한 비밀번호와 FireBase에 저장되어 있던 현재 비밀번호가 일치하면 변경할 새 비밀번호를 입력한다. (이는 alt1문에 대한 설명이다.)

새 비밀번호를 입력하고 확인을 위해 새 비밀번호를 한번 더 입력한 뒤, '변경' 버튼을 누르면 새 비밀번호와 확인을 위해 다시 입력한 새 비밀번호가 일치하는지 검사한다. 일치하면 새 비밀번호를 변수로 저장해 들고 와 FireBase에 접근해 사용자의 계정 내 비밀번호를 새 비밀번호로 변경하고 계정을 업데이트한다. (이는 alt2문에 대한 설명이다.)

My account change tel



[그림 4-14] my account change tel SD

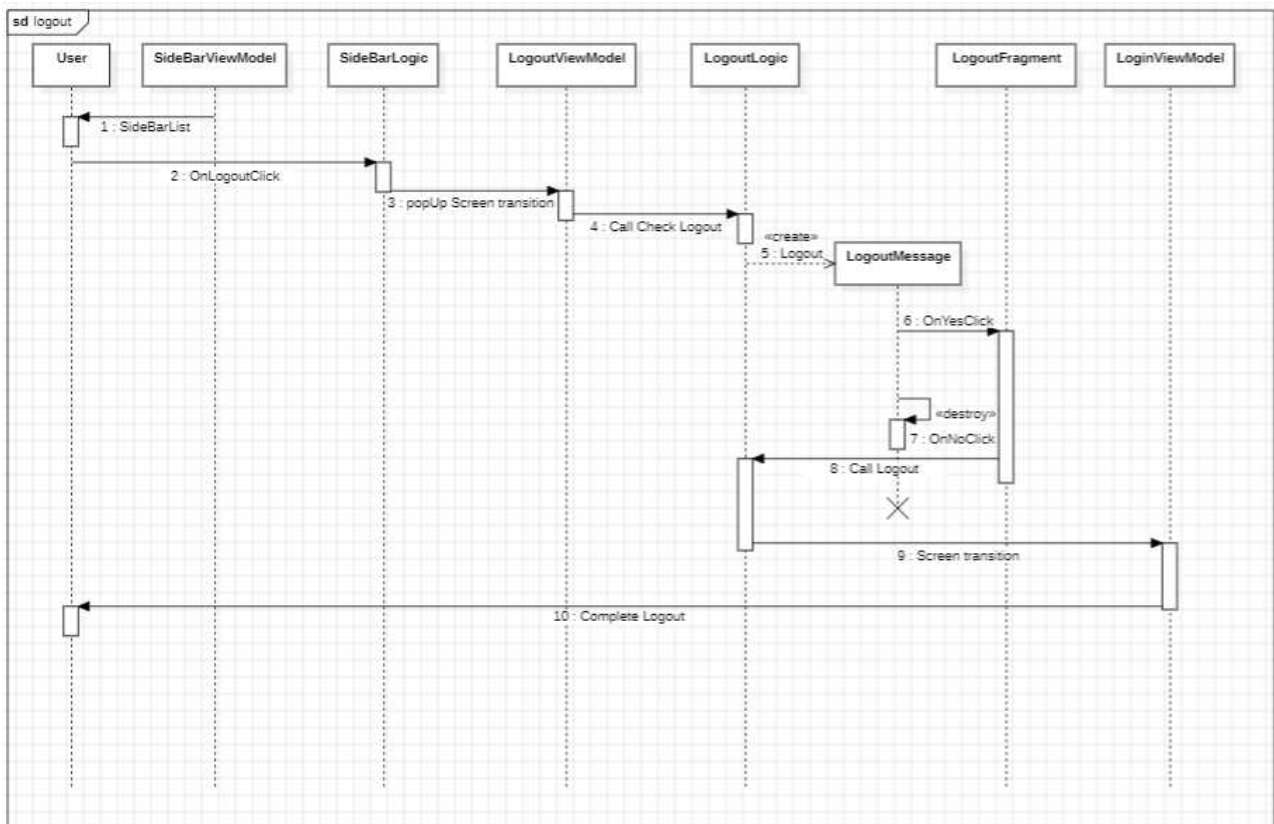
사용자의 전화번호를 변경하는 기능의 use case를 나타내는 sequence diagram이며, <Use Case #9-5>의 경우이다.

사용자가 사이드바에서 '내 계정'을 클릭하면 계정에 대한 기능을 제공하는 화면으로 전환된다. 계정에 대한 기능 중 '전화번호 변경' 버튼을 클릭하면 해당 화면으로 전환된다. 시스템은 현재 접속 중인 사용자의 uid로 Firebase 내 사용자의 정보에 접근해 현재 전화번호를 들고 온다.

전화번호 변경을 위해 사용자가 기존 전화번호를 입력해 '확인' 버튼을 누른다. 입력한 전화번호와 Firebase에 저장되어 있던 현재 전화번호가 일치하면 변경할 새 전화번호를 입력한다. (이가 alt1문에 대한 설명이다.)

새 전화번호를 입력하고 확인을 위해 새 전화번호를 한번 더 입력한 뒤, '변경' 버튼을 누르면 새 전화번호와 확인을 위해 다시 입력한 새 전화번호가 일치하는지 검사한다. 일치하면 새 전화번호를 변수로 저장해 들고 와 Firebase에 접근해 사용자의 계정 내 전화번호를 새 전화번호로 변경하고 계정을 업데이트한다. (이가 alt2문에 대한 설명이다.)

Logout



[그림 4-15] logout SD

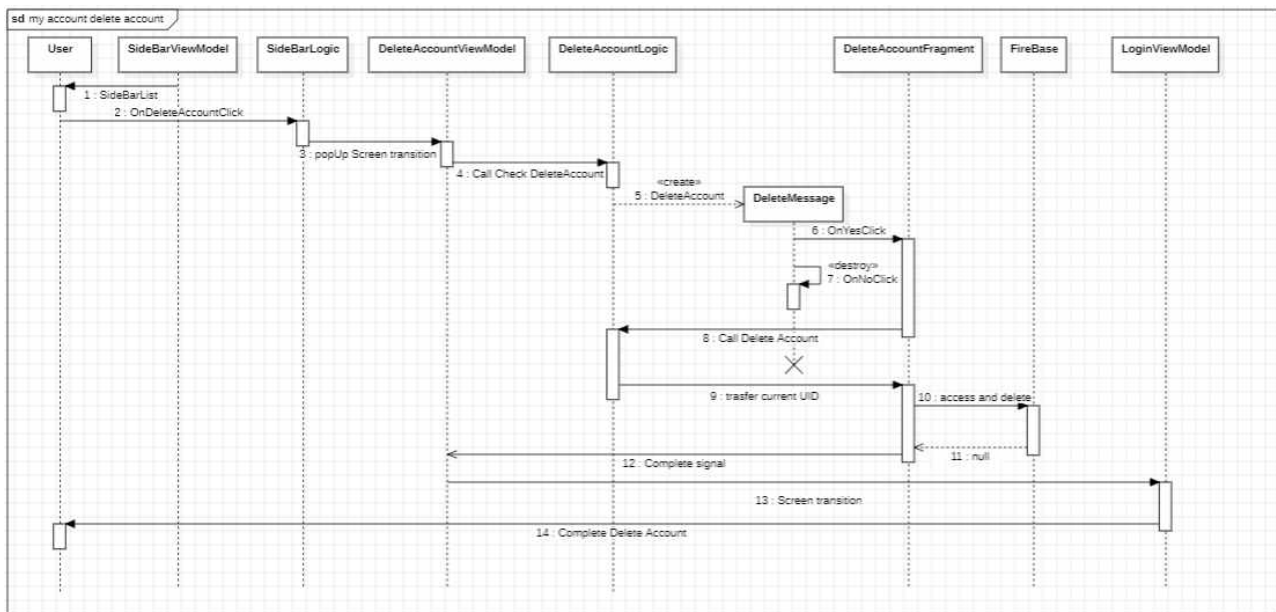
사용자가 앱에서 로그아웃을 하는 기능을 나타내는 sequence diagram이며, <Use Case #12>의 경우이다.

사용자가 사이드바에서 '로그아웃'을 클릭하면 팝업창을 통해 사용자에게 한번 더 로그아웃의 여부를 묻는다.

사용자가 'Yes' 버튼을 클릭하면 현재 접속 중인 사용자를 로그아웃시키고 로그인 화면으로 전환되며 이벤트는 종료된다.

사용자가 'No' 버튼을 클릭하면 어떠한 이벤트도 일어나지 않고 메시지는 삭제된다.

My account delete account



[그림 4-16] my account delete account SD

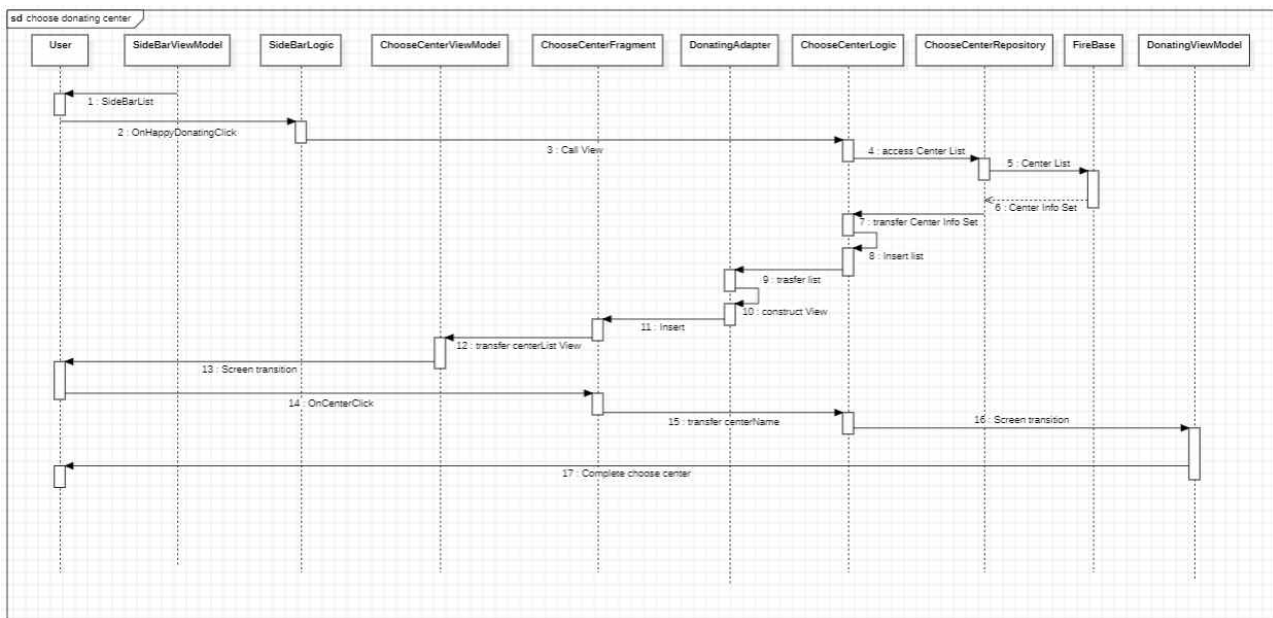
고객이 계정을 탈퇴하는 기능을 나타내는 sequence diagram이며, <Use Case #9-7>의 경우이다.

고객이 사이드바에서 '내 계정'을 클릭하면 계정에 대한 기능을 제공하는 화면으로 전환된다. 계정에 대한 기능 중 '계정 탈퇴' 버튼을 클릭하면 팝업창을 통해 고객에게 한번 더 탈퇴 여부를 묻는다.

고객이 'Yes' 버튼을 클릭하면 현재 접속 중인 사용자의 uid를 기준으로 Firebase 내 고객의 계정에 접근해 고객에 대한 모든 정보를 삭제하고 계정 탈퇴가 진행되고 로그인 화면으로 전환되며 이벤트는 종료된다.

고객이 'No' 버튼을 클릭하면 어떠한 이벤트도 일어나지 않고 메시지는 삭제된다.

Choose donating center

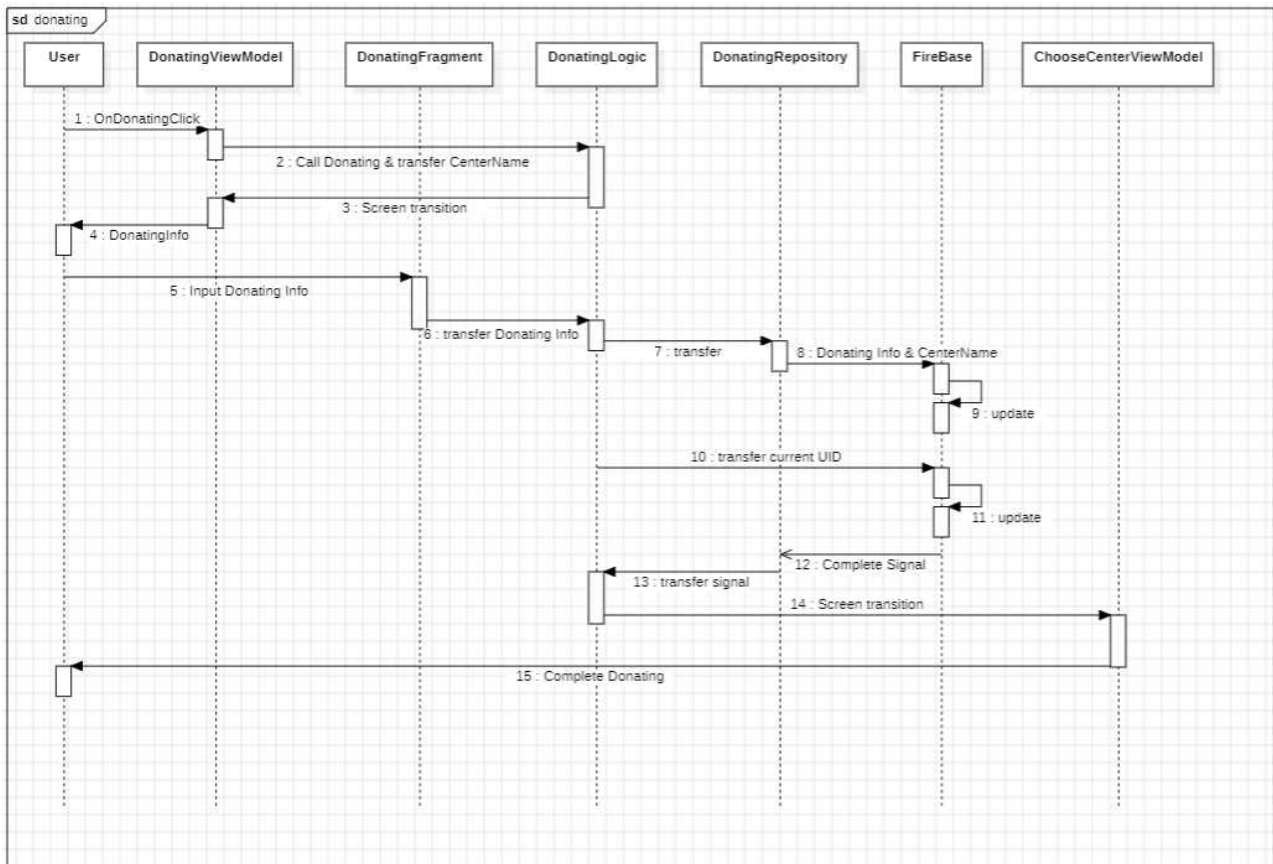


[그림 4-17] choose donating center SD

사업자가 나눔을 원하는 물품을 기부할 센터를 선택하는 기능을 나타내는 sequence diagram이며, <Use Case #11>의 경우이다.

사업자가 사이드바에서 '행복 나누기' 버튼을 클릭하면 시스템은 Firebase 내에 저장된 모든 아동센터들을 리스트로 저장하고 어댑터를 통해 리사이클러 뷰를 구성한 후, 사용자에게 제공한다. 사업자는 제공받은 아동센터들 중 기부할 아동센터를 선택한다. 시스템은 사업자가 선택한 센터의 이름을 변수로 저장하고 사업자에게 기부 정보를 입력할 수 있는 화면으로 전환한다.

Donating

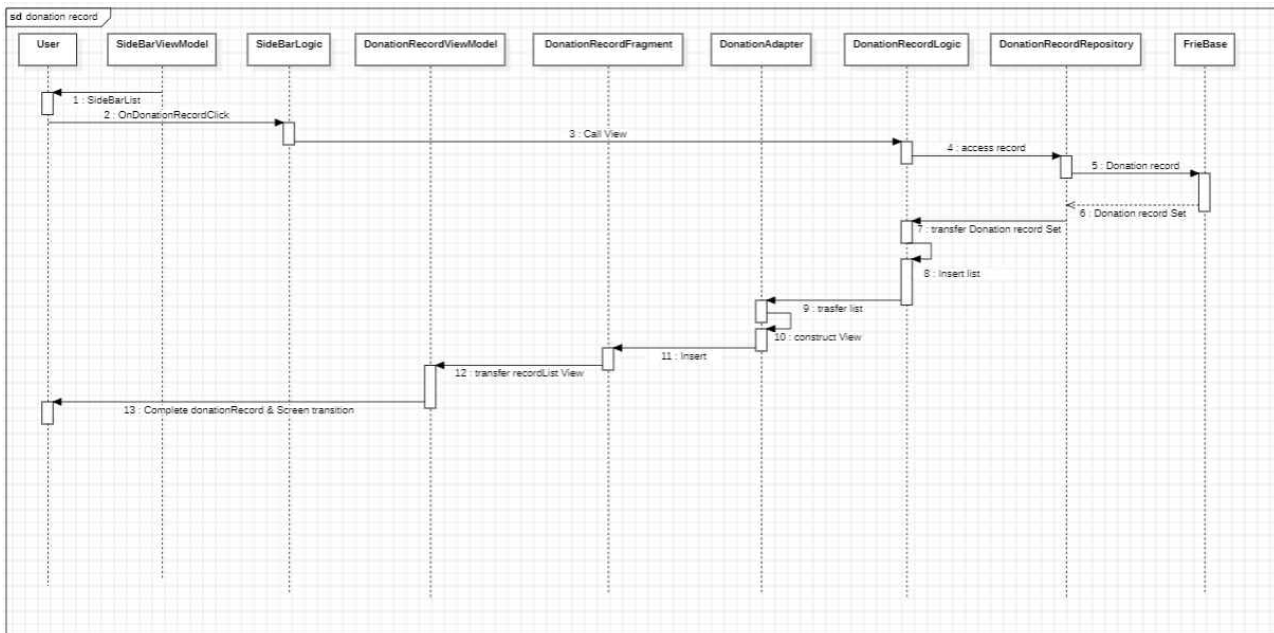


[그림 4-18] donating SD

사업자가 나눔을 원하는 물품을 아동센터에 기부하는 기능을 나타내는 sequence diagram이며, <Use Case #11-2>의 경우이다.

사업자가 기부할 센터를 선택한 후의 기능으로 사업자는 기부할 사람의 이름, 물품, 수량, 전화번호를 입력하고 '기부하기' 버튼을 입력한다. 시스템은 입력받은 정보를 가져와 FireBase에 저장한다. 사업자가 전에 선택한 아동센터의 이름을 가진 문서 내에 기부한 사람의 이름, 물품, 수량, 전화번호를 저장한다. 또 현재 접속 중인 사업자의 uid로 만들어진 문서 내에 사업자가 기부할 때 입력했던 이름을 기준으로 만들어진 배열 내 기부할 센터의 이름, 물품, 수량을 저장한다. 만약 사업자가 기부할 때 입력했던 이름의 배열이 없는 경우, 즉 입력했던 이름으로 기부를 처음 진행하는 경우에는 배열 필드를 새로 만들어 저장한다. 저장이 완료되면 다시 기부할 센터를 선택하는 화면으로 전환되며 해당 기능은 종료된다.

Donation record

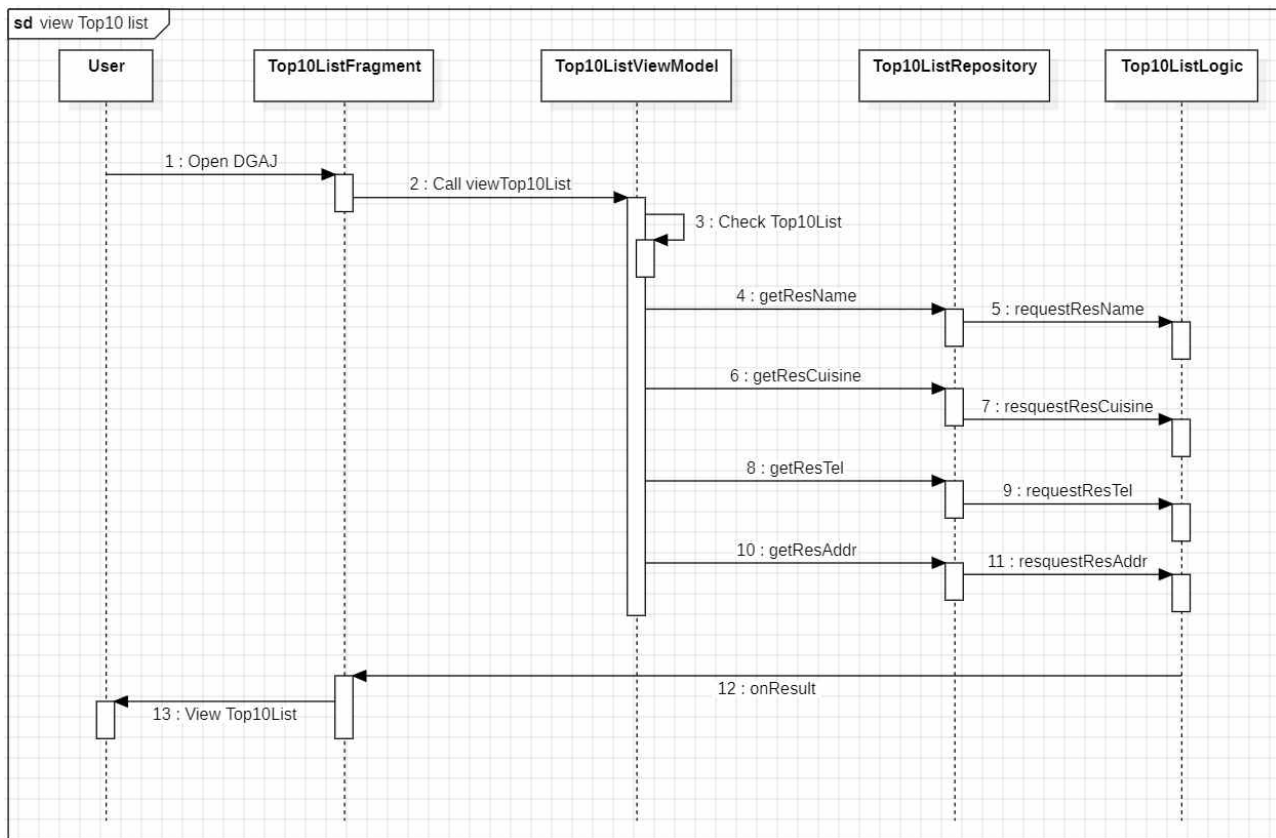


[그림 4-19] donation record SD

사업자가 아동센터에 기부했던 내역을 제공하는 기능을 나타내는 sequence diagram이며, <Use Case #11-6>의 경우이다.

사업자는 사이드바에서 '내가 나눈 행복' 버튼을 누르면 시스템은 현재 접속 중인 사용자의 uid를 가져와 FireBase에 접근한다. FireBase 내 기부 내역을 담은 컬렉션 내에 현재 접속 중인 사업자의 uid를 이름으로 가진 문서에 저장된 모든 기부 내역을 리스트로 저장하고 어댑터를 통해 리사이클러 뷰를 구성한 후, 사용자에게 제공한다. 또 사업자가 현재 기부한 횟수를 계산해 현재 날짜까지의 기부횟수를 출력해 사업자에게 제공한다.

View Top10 List

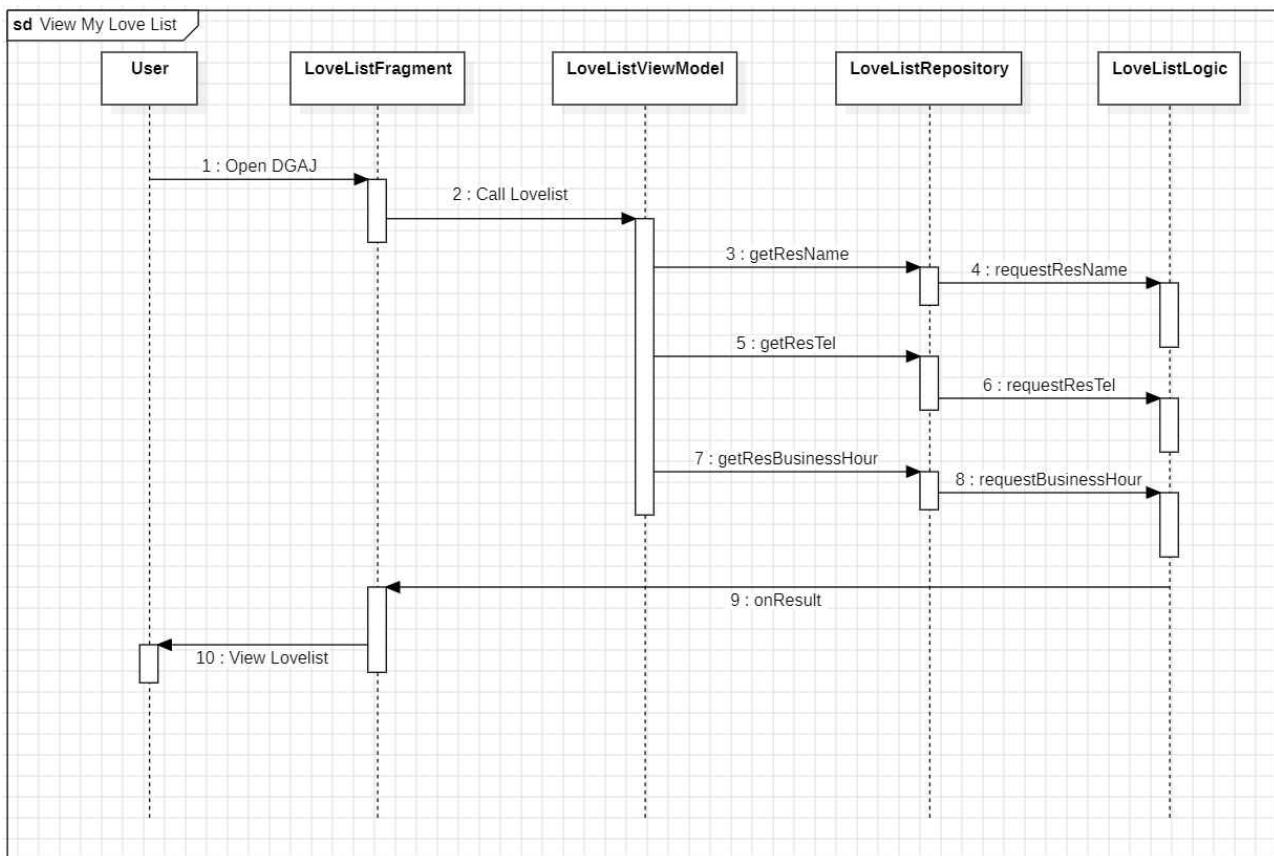


[그림 4-20] view Top10 List SD

사용자가 식당 검색 화면에서 Top10 버튼을 선택했을 때 보이는 화면이며, <Use Case #5>의 경우이다.

Top10의 정렬 기준은 사용자들이 자신들의 저장 목록에 추가한 식당들에 대한 전체 찜 수와 해당 사용자가 위치한 지역구가 기준이 된다. 예를 들어, 사용자가 대구광역시의 중구에 위치하고 있을 경우, API 내 주소지를 중구로 하는 식당들의 찜 수를 파이어베이스에서 불러온 후, 찜 수를 기준으로 내림차순 정렬을 한 기준으로 리사이클러 뷰로 보이게 한다.

View My Love List

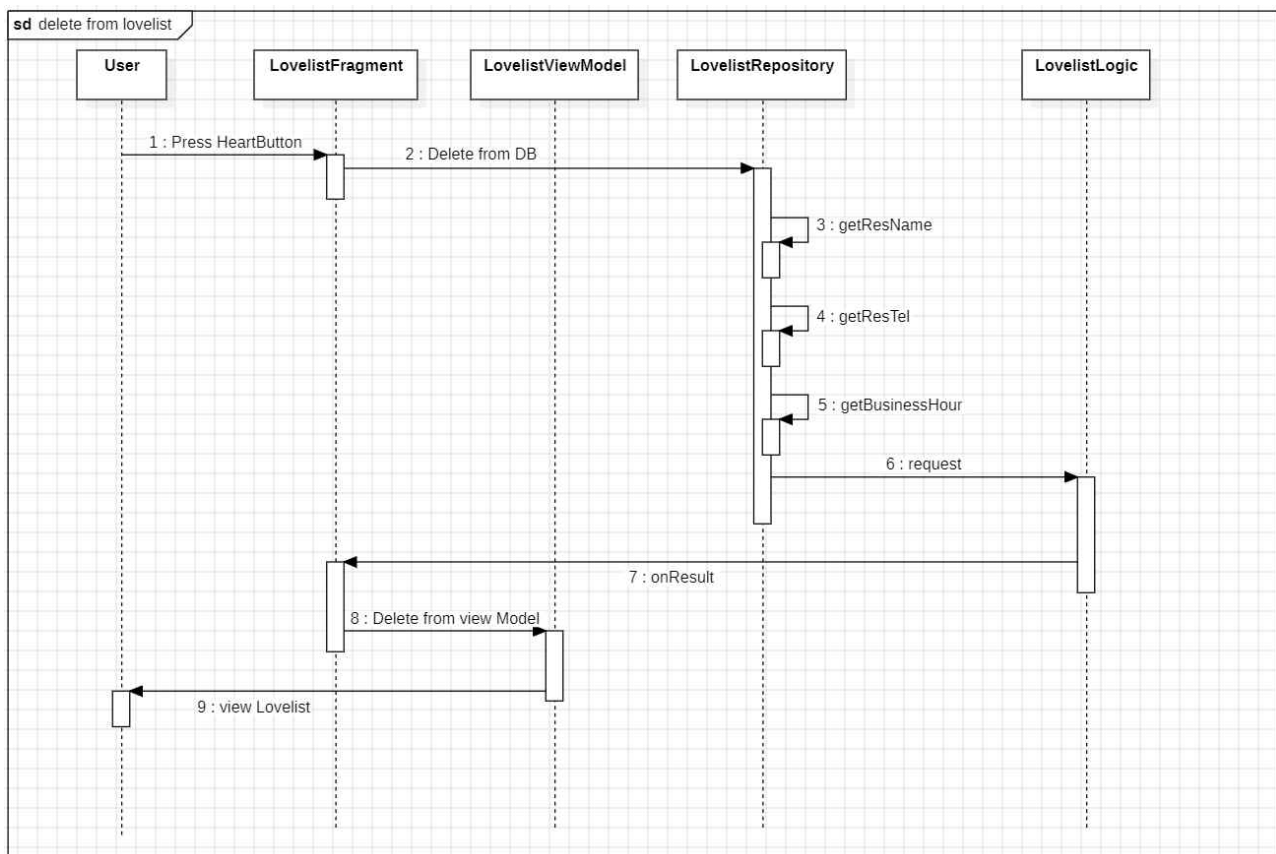


[그림 4-21] view myLove List SD

사용자가 사이드바에서 나의 저장 목록을 선택하였을 경우 보이는 화면이며, <Use Case #7>의 경우이다.

파이어 베이스 내, 자신의 계정에 본인이 등록한 식당 저장 목록이 없을 경우, 저장 목록이 없음을 알리는 메시지가 뜨며, 저장목록에 찜 내역이 있을 경우, DB에 저장된 식당의 고유번호와 대구광역시 내 식당 API를 교차검색하여 원하는 정보를 리사이클러뷰로 보이게 한다.

Delete From My Love List

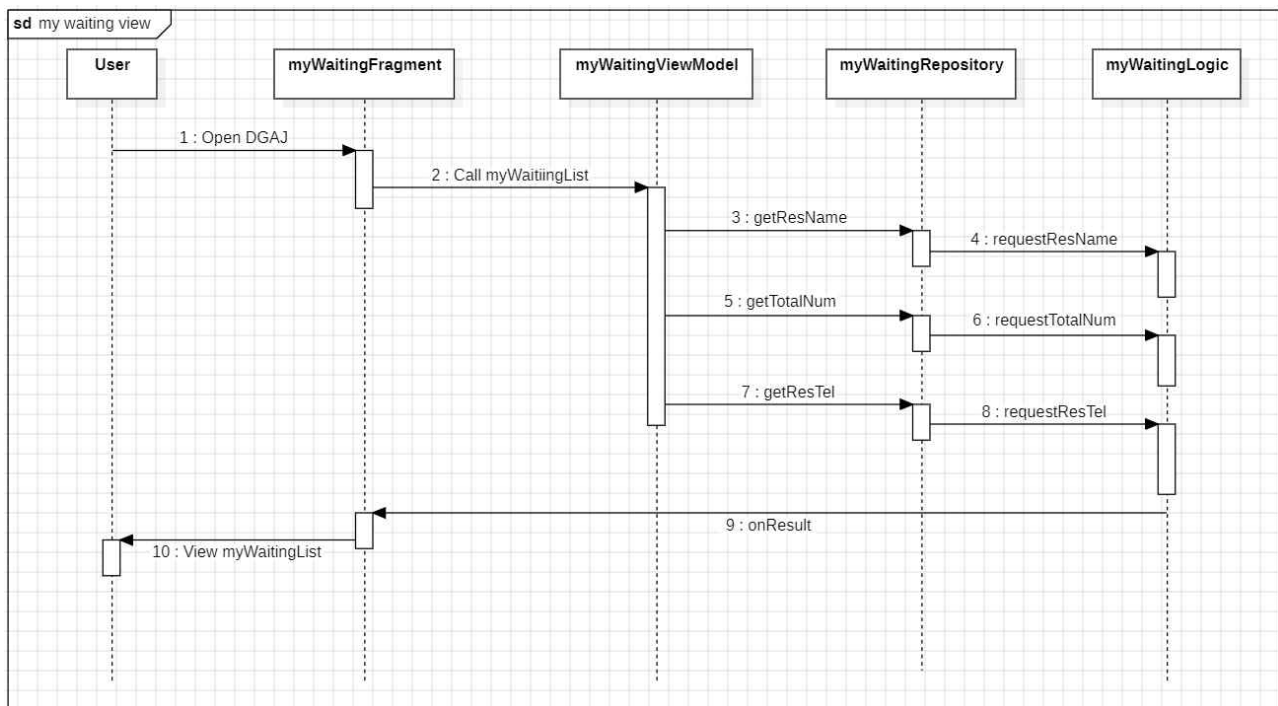


[그림 4-22] Delete From My Love List SD

사용자가 사이드바에서 나의 저장 목록을 선택하여 보이는 화면에서 리사이클러뷰 내의 아이템의 오른쪽에 있는 하트버튼을 눌렀을 때 발생하는 이벤트이며, <Use Case #7-3>의 경우이다.

하트 버튼이 뜬다는 얘기는 자신이 저장해둔 식당이 있다는 의미이다. 그렇게 보여진 하트 버튼을 누르게 되면 해당 리사이클러뷰에서도 사라지는 것은 물론, 파이어베이스 내, my_love_list에서도 삭제되어 다시 View My Love List를 실행하였을 때도, 삭제한 저장 가게는 보이지 않게 된다.

View My Waiting View

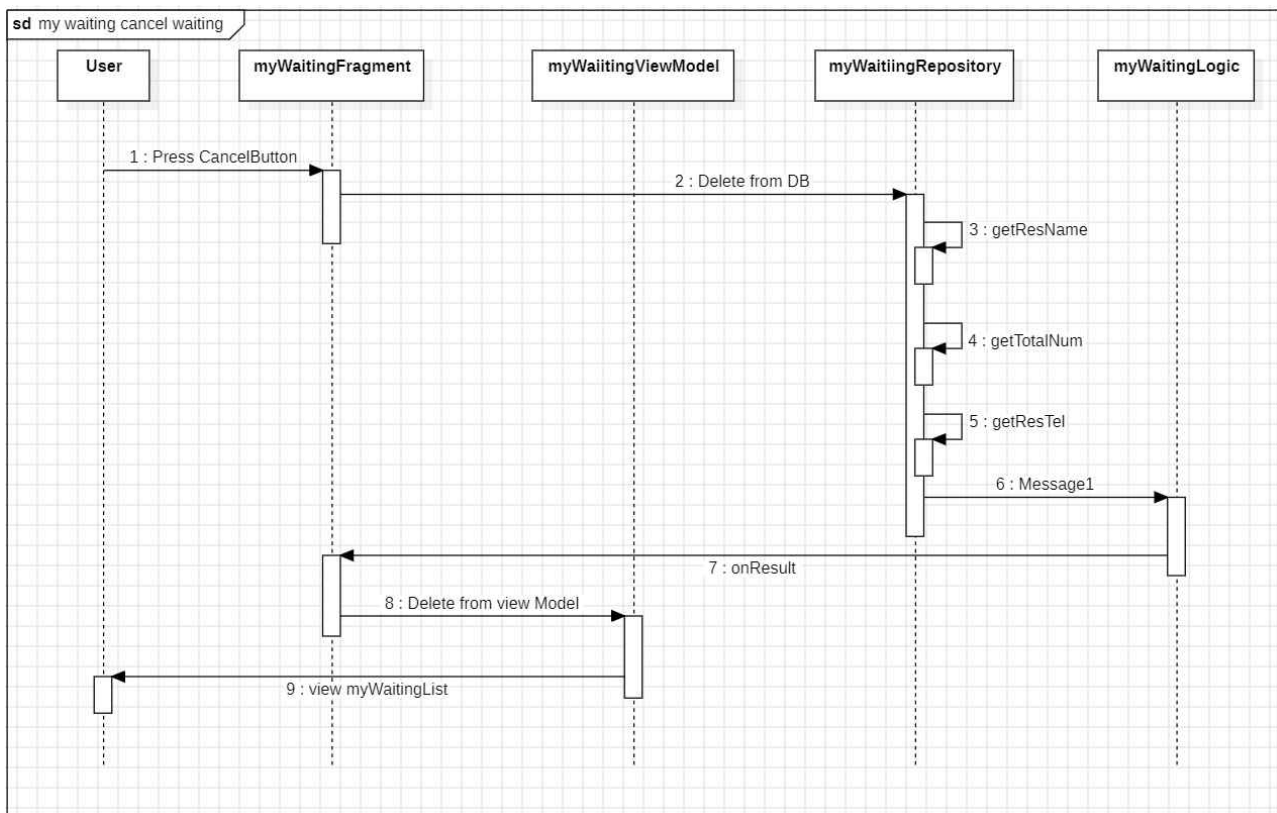


[그림 4-23] my Waiting View SD

사용자가 사이드바에서 대기 현황을 선택했을 때 보이는 화면이며, <Use Case #8-5>의 경우이다.

사용자가 예약신청을 한 식당에 대한 정보를 파이어베이스에 등록한 나의 대기 목록에 있는 가게 고유번호와 API내의 식당정보를 비교하여 원하는 정보를 띄우며, 자신이 예약할 당시에 기입한 내역을 확인하고 싶은 경우 나의 예약 정보를 누르면 자신이 기입한 예약정보를 파이어베이스에서 가져와 메시지로 띄우게 된다.

Cancel My Waiting

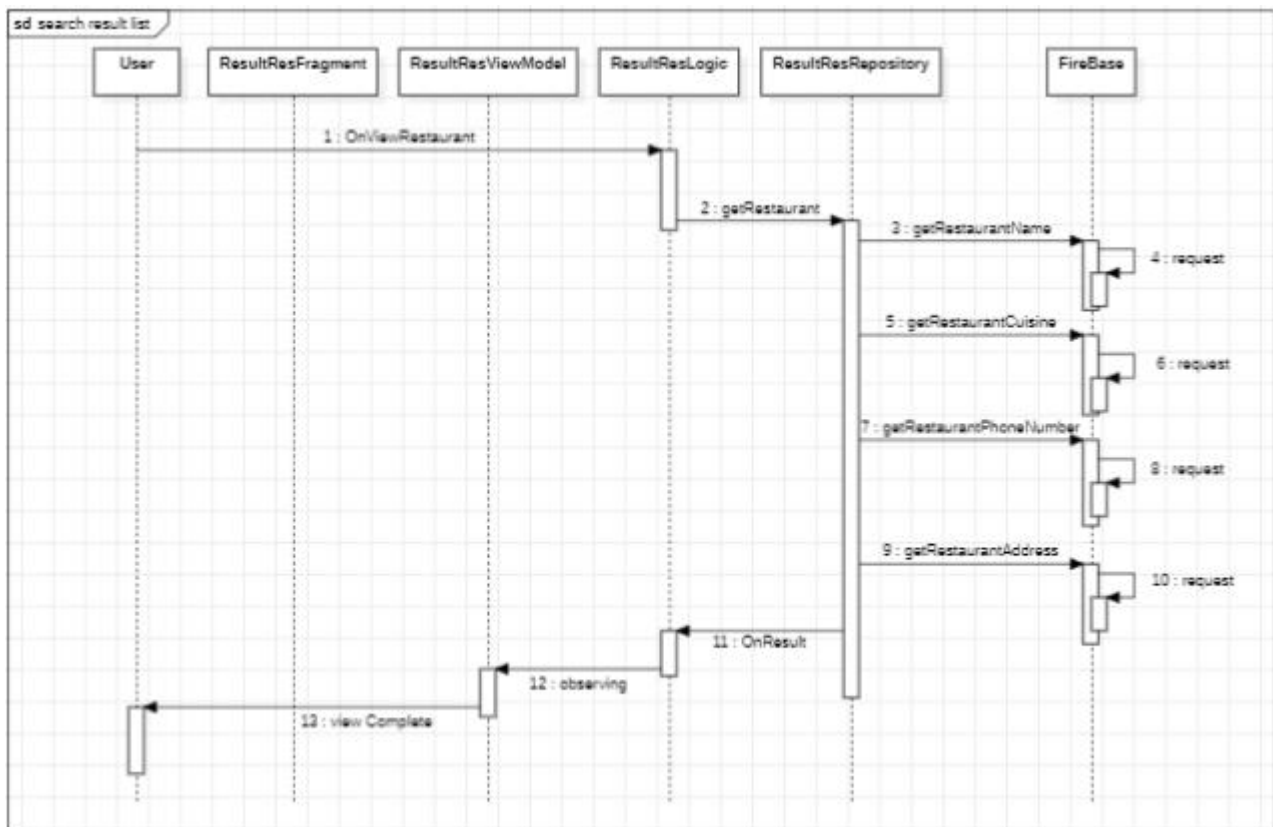


[그림 4-24] my Waiting cancel SD

사용자가 사이드바에서 대기 현황을 선택했을 때 보이는 화면에서 대기 취소 버튼을 눌렀을 때 발생하는 이벤트이며, <Use Case #8-6>의 경우이다.

사용자가 예약신청을 한 식당 아이템의 오른쪽 하단에 있는 대기 취소 버튼을 누르면 현재 리사이클러뷰에서 해당 가게에 대한 아이템이 사라지는 것은 물론, 파이어베이스 내, 자신의 예약 신청, 신청한 가게 내 예약 내역까지 모두 삭제되게 된다.

Search Result List

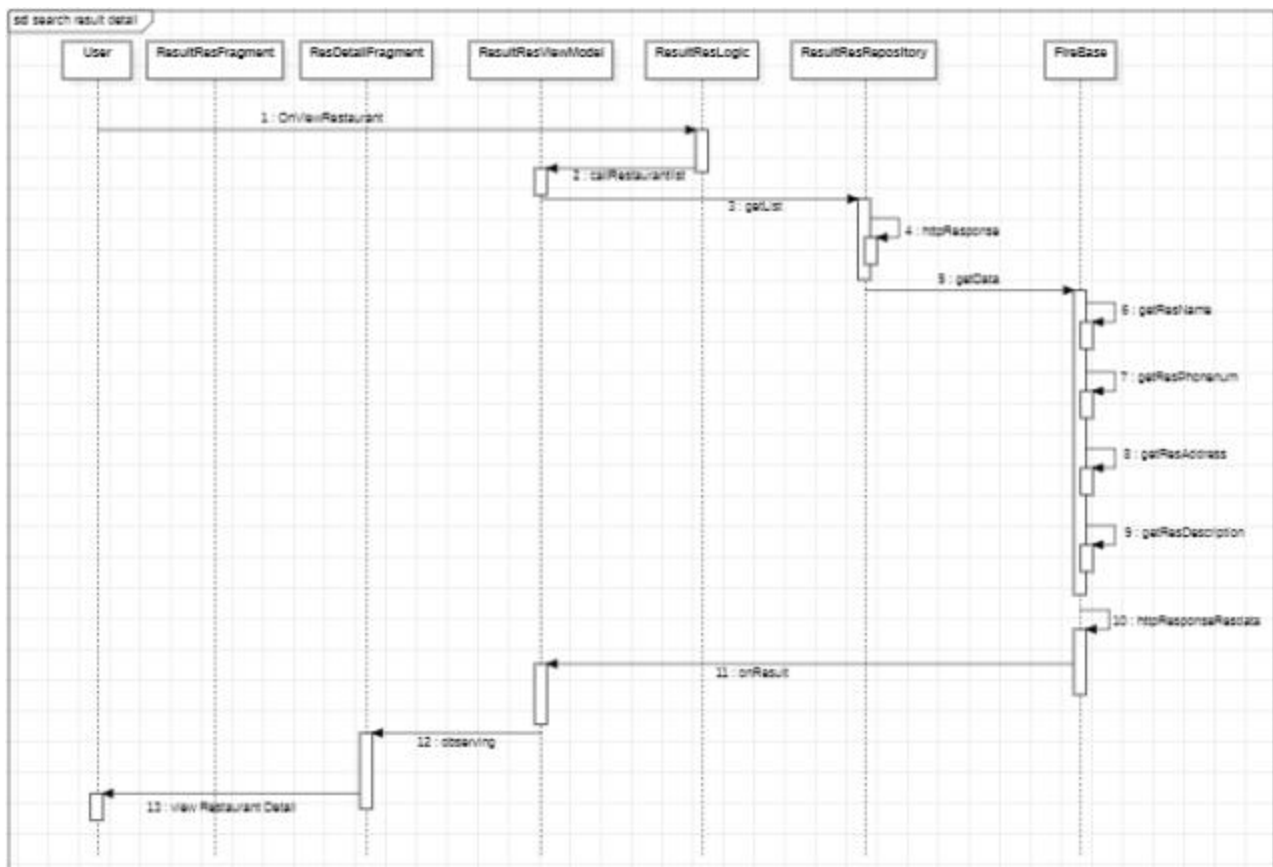


[그림 4-25] Search Result List SD

사용자가 메뉴를 검색했을 때 전환되는 화면에서 보여지는 식당들의 리스트 뷰이며, <Use Case #3>의 경우이다.

화면이 전환되기 전에 입력한 메뉴와 위치 정보에 대한 값에 알맞은 식당을 API에서 검색하여 불러온다. 불러온 식당의 정보 중 필요한 값을 추출하여 리스트 뷰에 보이게 하고, 검색 결과에 해당하는 식당이 여러 개가 있을 경우, 이를 리스트 형식으로 보이게 하여 화면에 출력한다.

Search Result Detail

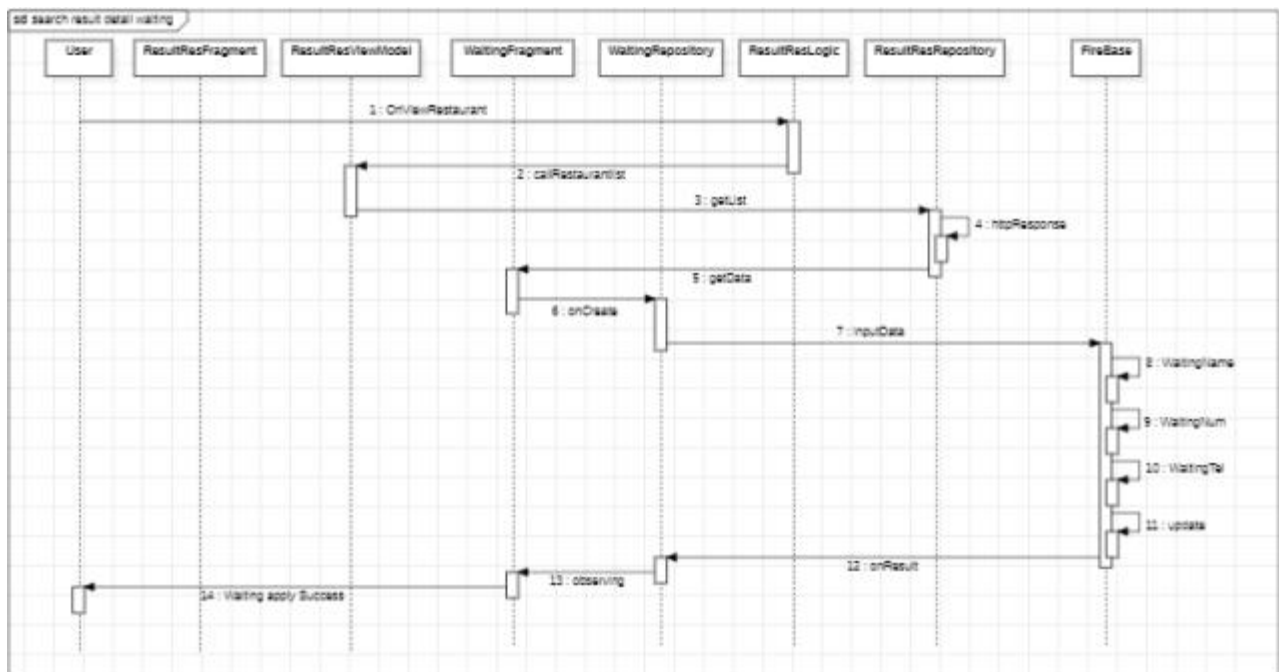


[그림 4-26] Search Result Detail SD

사용자가 식당을 선택했을 때 전환되는 식당에 상세 정보 페이지이며, <Use Case #6>의 경우이다.

사용자가 검색한 결과에 해당하는 식당들이 리스트 뷰로 보여지고, 그 중 하나를 선택할 경우, 해당 가게에 대한 이름, 전화번호, 주소, 카테고리, 대표 메뉴, 찜 개수가 출력되는 화면을 띄운다. 식당의 이름, 전화번호, 주소, 카테고리, 대표 메뉴에 대한 정보는 API에서 불러오며, 찜 개수의 경우 Firestore에서 값을 불러온다.

Search Result Detail Waiting

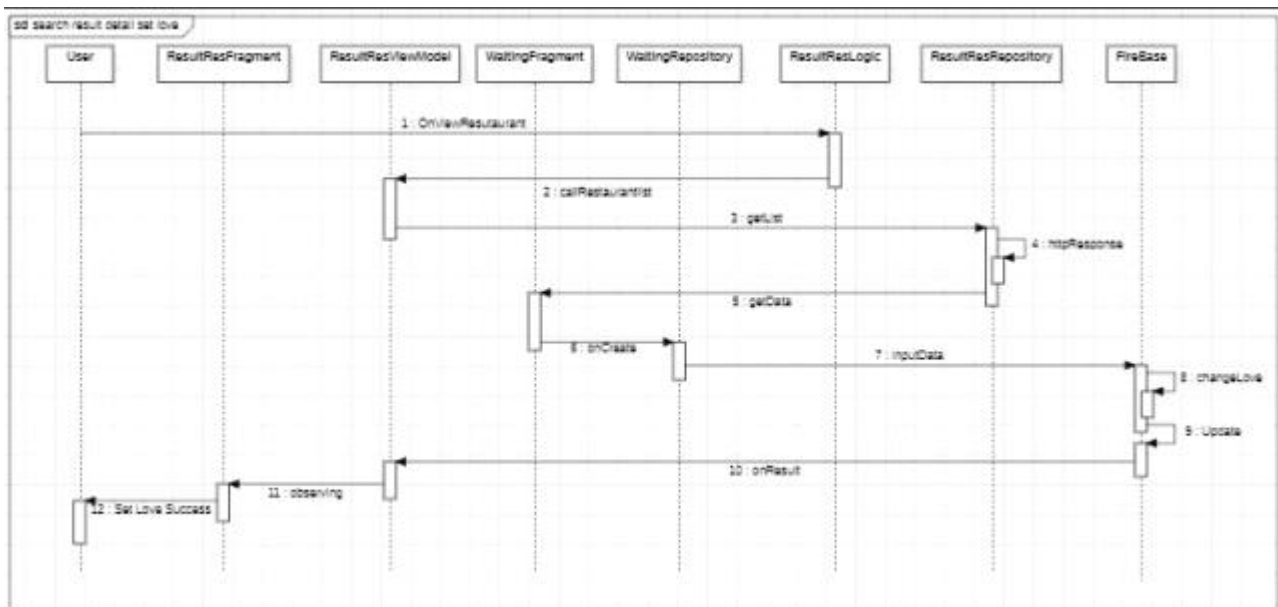


[그림 4-27] Search Result Detail Waiting SD

사용자가 가게 상세 페이지 화면에서 줄서기 버튼을 눌렀을 때 전환되는 화면이며, <Use Case #8>의 경우이다.

사용자가 가게 상세 페이지에서 줄서기 버튼을 눌렀을 때, 줄서기 신청자의 이름, 줄서기 전체 인원수, 신청자의 전화번호를 입력하고 확인 버튼을 누르면 현재 식당에 등록된 대기 팀의 수를 알려주는 창이 뜨고, 확인 버튼을 누르게 되면, Firestore에 해당 정보가 저장되게 된다.

Search Result Detail Set Love

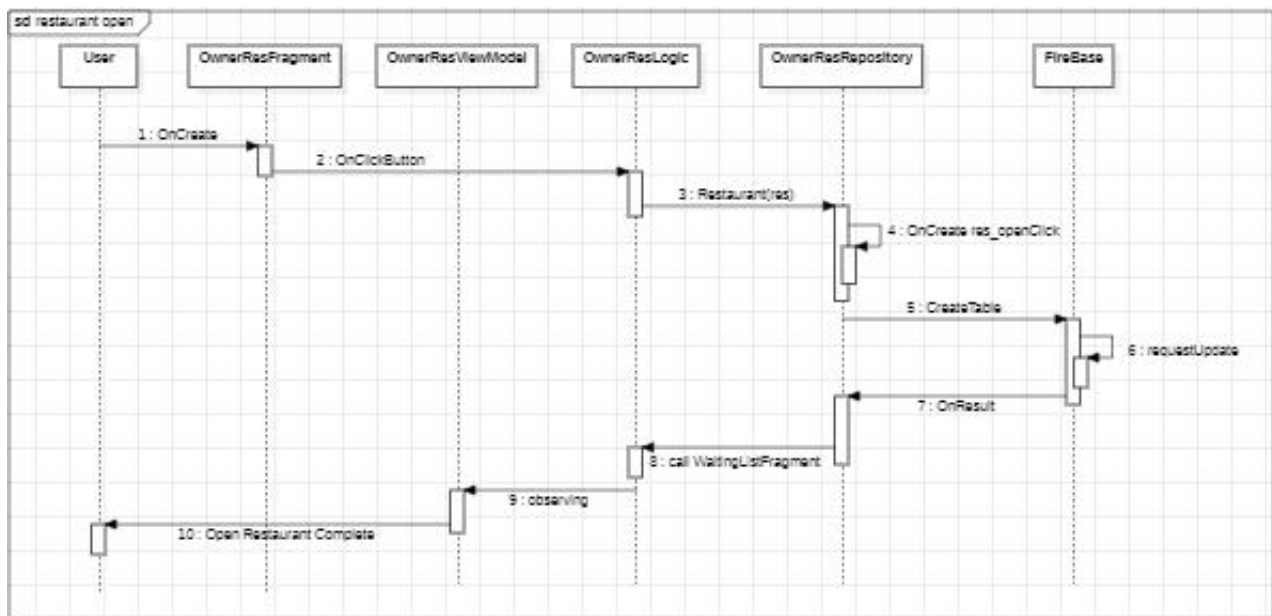


[그림 4-28] Search Result Detail Set Love SD

사용자가 가게 상세 페이지 화면에서 가게 찜하기 버튼을 눌렀을 때 실행되는 기능이며, <Use Case #7-1>의 경우이다.

사용자가 가게 상세 페이지에서 찜하기 버튼을 눌렀을 때, 해당 가게를 자신의 저장목록 리스트에 추가하게 된다. 이는 Firestore에 저장되어 있는 자신의 계정 정보 안에 값이 저장되게 된다.

Restaurant Open

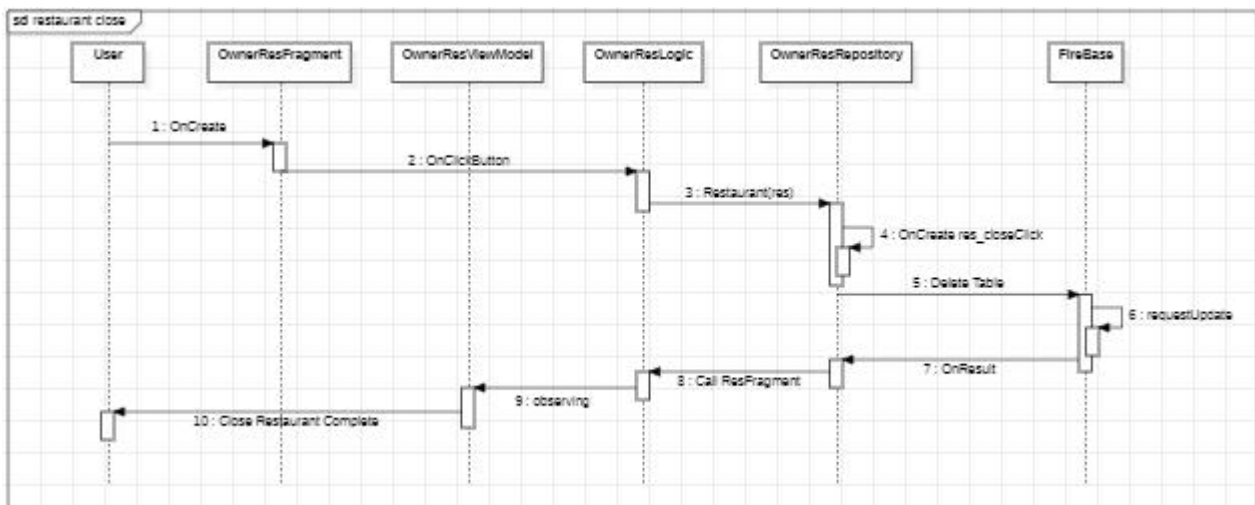


[그림 4-29] Restaurant Open SD

사용자가 사업자의 계정으로 로그인했을 때 보이는 화면이며, <Use Case #10-3>의 경우이다.

사용자가 자신의 식당의 대기 여부를 가능하게 하려고 할 때 실행되는 기능으로, Open 버튼을 누르면 일반 고객이 자신의 식당에 대기 신청이 가능하게 된다. 사용자가 Open을 누르지 않는다면, 일반 고객은 해당 식당에 대기 신청이 불가능하다.

Restaurant Close

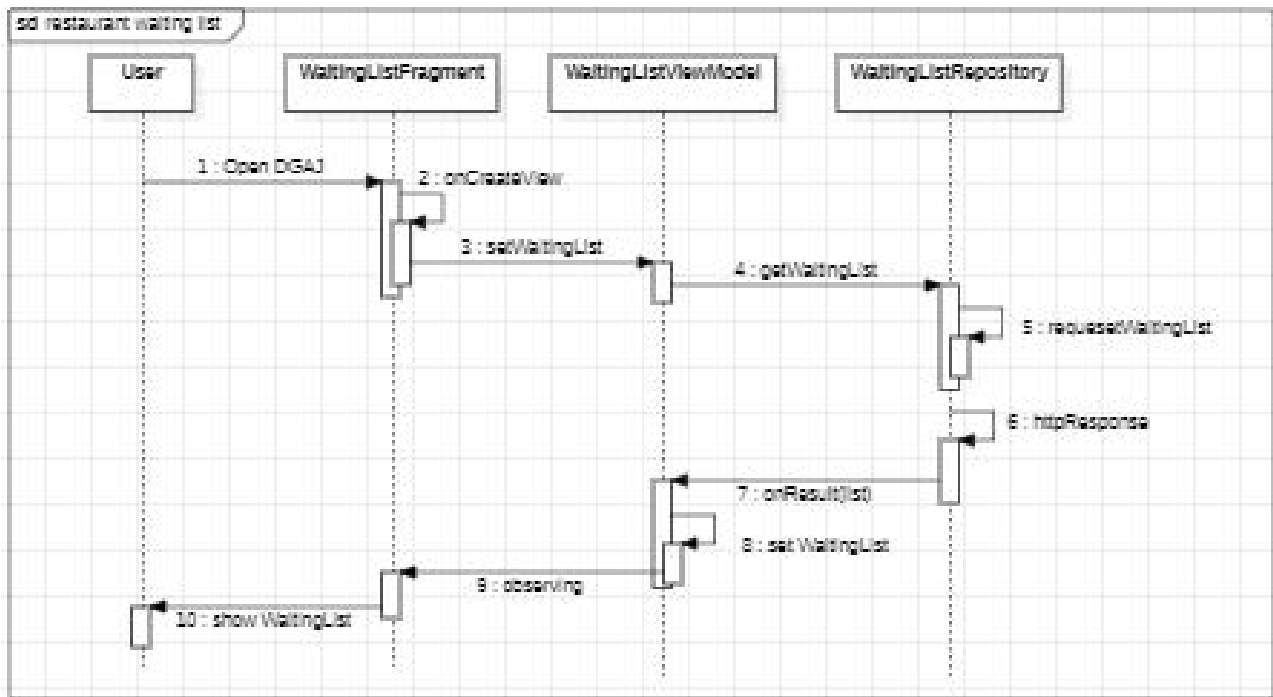


[그림 4-30] Restaurant Close SD

사용자가 사업자의 계정으로 로그인했을 때 보이는 화면의 하단에 있는 버튼으로 기능하며, <Use Case #10-10>의 경우이다.

사용자가 화면 하단의 Close버튼을 누르면 기능이 시작된다. 사용자가 자신의 식당의 대기 여부를 불가능하게 하려고 할 때 실행되는 기능으로, Close버튼을 누르면 일반 고객이 자신의 식당에 대기 신청이 불가능하게 된다.

Restaurant Waiting list

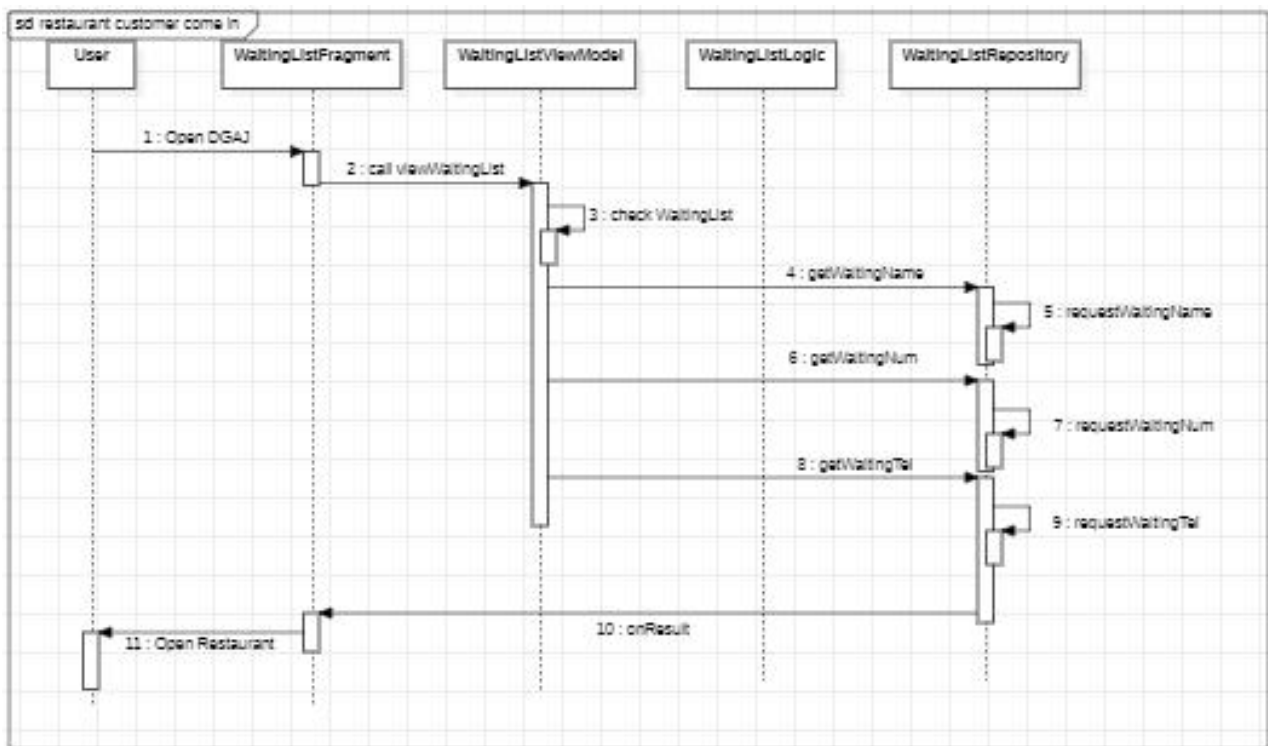


[그림 4-31] Restaurant Waiting List SD

사용자가 사업자의 계정으로 로그인했을 때 보여지는 화면이며, <Use Case #10>의 경우이다.

일반 고객이 자신의 식당에 대기 신청을 하였을 경우, 대기 신청 수만큼 리스트 뷰로 보이게 된다. 한 항목당 대기 신청자의 이름, 대기 총인원 수, 신청자의 전화번호가 뜨게 되고, 오른쪽에는 입장 버튼과, 삭제 버튼이 있다.

Restaurant Customer Come in

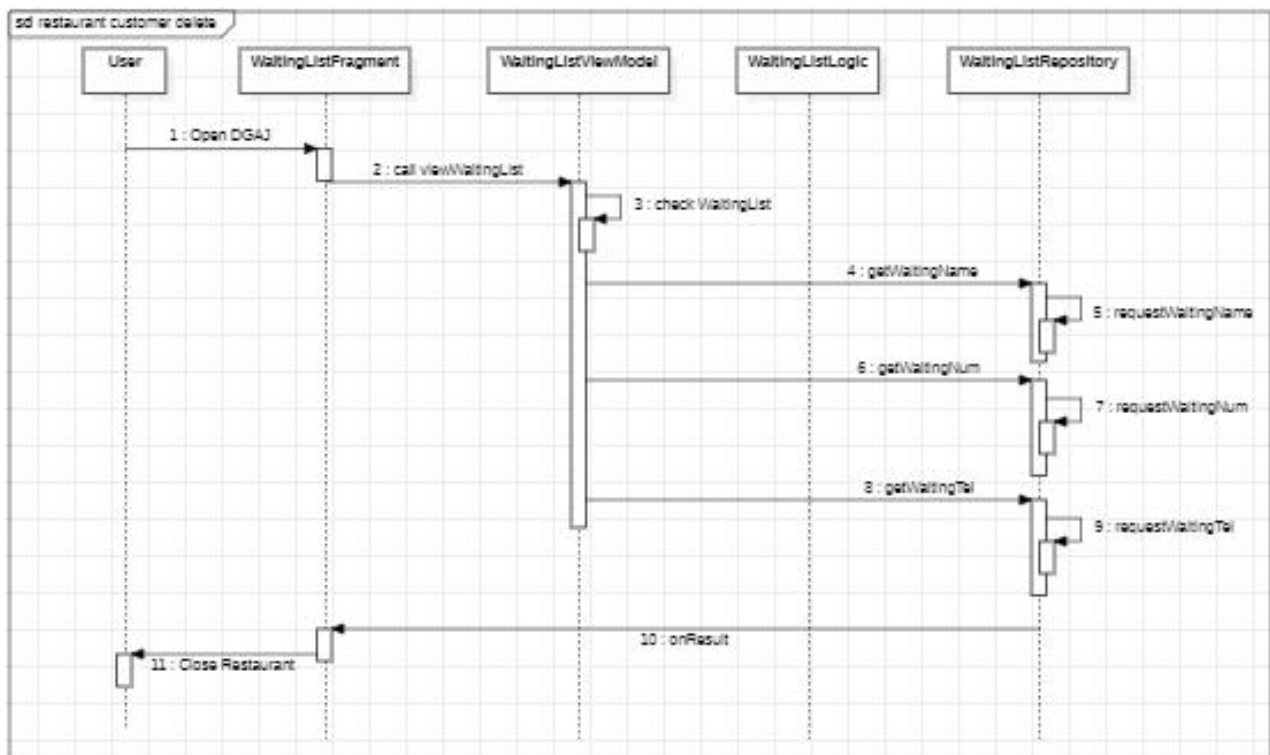


[그림 4-32] Restaurant Customer Come in SD

사용자가 사업자의 계정으로 로그인했을 때 보여지는 화면이며, <Use Case #10-6>의 경우이다.

자신의 식당에 대기 신청을 한 고객들의 정보가 뜨는 뷰의 입장 버튼을 누르면 실행되는 기능으로, 버튼을 누르면 Firestore의 데이터베이스에 등록된 대기 팀의 정보가 삭제된다.

Restaurant Customer Delete

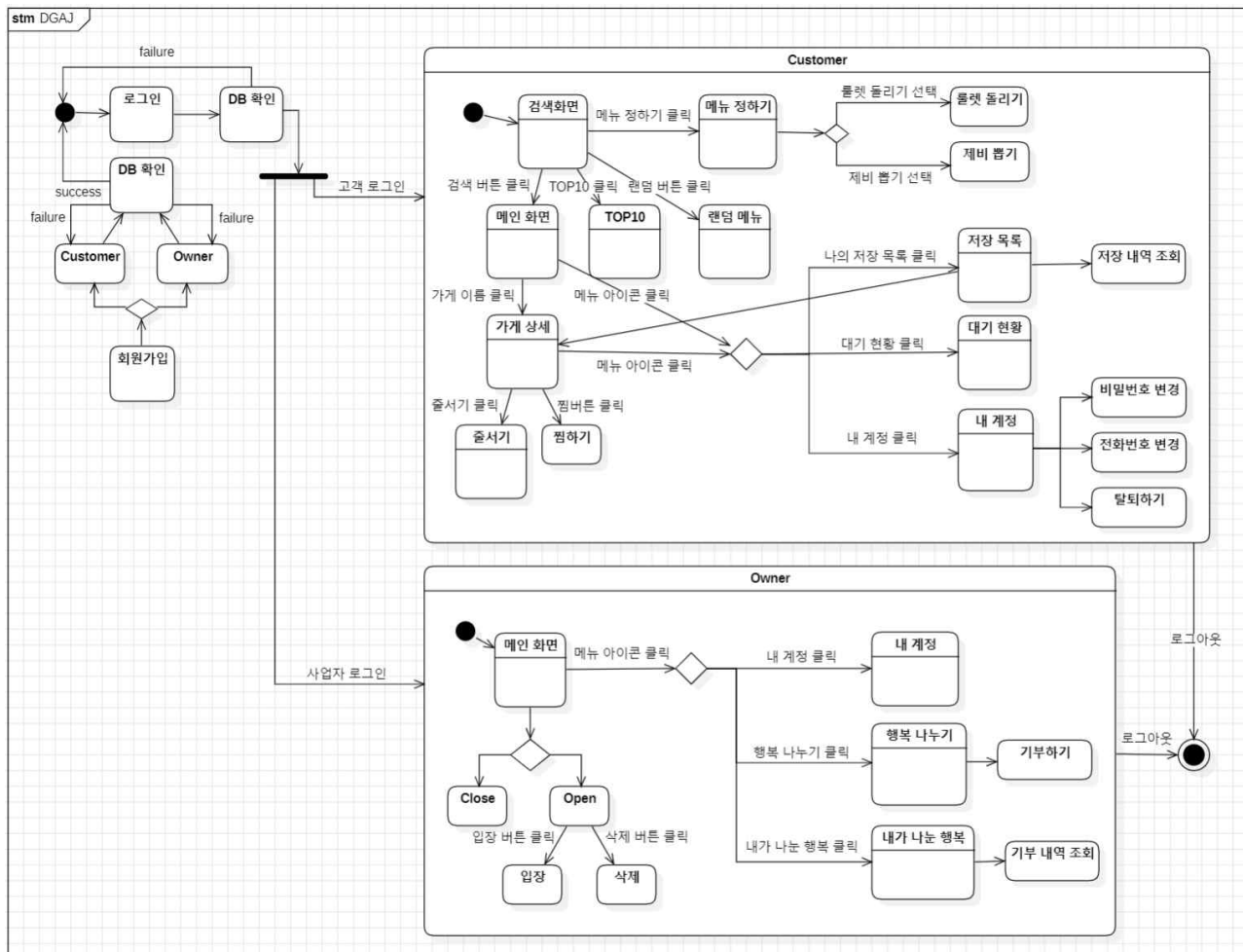


[그림 4-33] Restaurant Customer Delete SD

사용자가 사업자의 계정으로 로그인했을 때 보여지는 화면이며, <Use Case #10-8>의 경우이다.

자신의 식당에 대기 신청을 한 고객들의 정보가 뜨는 뷰의 삭제 버튼을 누르면 실행되는 기능으로, 버튼을 누르면 Firestore의 데이터 베이스에 등록된 대기 팀의 정보가 삭제된다.

5. State machine diagram



5.1 State machine diagram description

앱 실행 시 로딩 화면이 시작되며 로딩 화면이 종료되면 Login 페이지가 나타난다. 이전의 가입 유무에 따라 로그인 기능과 회원가입 기능을 사용할 수 있고, 이전에 가입한 정보가 기억나지 않을 시 아이디/비밀번호 찾기 기능을 사용하여 해당 정보 조회가 가능하다. 회원가입의 경우 고객용과 사업자용 두 가지로 나뉘며, 사용자가 입력한 정보가 설정된 형식에 적합할 경우 해당 사용자의 정보를 DB에 저장한다. 사업자의 경우, 회원가입 시에 본인 가게의 고유번호를 입력하여 등록된 가게인지 확인하는 절차를 필요로 한다. 로그인과 아이디/비밀번호 찾기 기능은 DB에 저장된 사용자 정보를 바탕으로 적절한 절차를 거쳐 인증이 완료되면 정상적인 동작이 가능하다. 로그인에 성공할 경우, 로그인된 사용자의 타입(고객, 사업자)에 따라 각각 다른 초기 화면이 나타나게 된다.

고객 계정으로 로그인이 성공하면 위치와 메뉴를 입력하여 해당 위치를 기준으로 입력한 메뉴를 판매하는 주변 가게들의 검색이 가능한 검색 화면으로 전환된다. 위치 설정은 사용자가 위치 설정 창에서 임의로 입력하여 설정하거나, 우측 상단의 현 위치 버튼을 클릭하여 현재 위치로 초기화할 수 있다. 메뉴 입력을 하는 방식은 랜덤 버튼의 사용 여부에 따라 달라지는데, 랜덤 버튼을 클릭할 때마다 DB에 저장된 가게 정보를 바탕으로 랜덤하게 메뉴 명이 출력되어 사용자로 하여금 재미와 편리함을 가져다 줄 수 있다. 이 밖에도 TOP10, 메뉴 정하기와 같이 메뉴 선정에 도움을 주는 다양한 기능들이 구현되어 있다. TOP10은 시스템에서 설정한 검색기준을 바탕으로 상위 10개의 가게 리스트를 뽑아 추천해주는 기능이다. 메뉴 정하기 기능은 쉽게 메뉴를 결정하기 힘들거나 재미있게 메뉴 선정을 하고 싶은 사용자들을 위해 마련된 항목으로써, '룰렛 돌리기', '제비뽑기' 등의 게임을 이용하여 랜덤으로 메뉴 선정이 가능하다. 앞서 소개한 방식들을 바탕으로 위치와 메뉴를 입력하고 검색 버튼을 누르면 메인 화면으로 전환된다. 메인 화면에서는 입력한 주소와 메뉴에 해당하는 가게 리스트가 지도와 함께 나타난다. 가게 리스트에 가게를 클릭하면 해당 가게의 상세 정보 페이지로 이동하여 상세 주소와 메뉴 등 선택한 가게의 상세 정보를 확인할 수 있다. 가게 상세 정보 페이지에서는 해당 가게를 '나의 저장 목록'에 추가/삭제 할 수 있는 '찜 기능'과 웨이팅 신청을 할 수 있는 '줄서기' 기능을 이용할 수 있다. 줄서기의 경우 해당 가게의 사업자가 본인의 가게의 웨이팅 기능을 활성화 하였을 경우에만 가능하며, 웨이팅에 성공하였을 경우에는 사이드 바의 '대기 현황'을 통하여 본인이 신청한 웨이팅 목록의 확인이 가능하다. 사이드바(고객 메뉴 창)는 좌측 상단의 메뉴 아이콘을 클릭함으로써 활성화되는데, 해당 메뉴 창을 통해 이전에 저장(찜)했던 가게 리스트를 확인할 수 있는 '나의 저장 목록', 본인이 신청한 웨이팅 상황을 확인할 수 있는 '대기 현황', 개인정보 수정이 가능한 '내 계정'으로의 화면 이동이 가능하고 '로그아웃'을 클릭하여 로그아웃을 할 수 있다.

사업자 계정으로 로그인에 성공할 경우, 메인 화면에서 Open / Close 버튼을 이용하여 각각 본인의 가게의 줄서기 기능을 활성화하거나 비활성화할 수 있다. Open 버튼을 클릭하여 줄서기를 현재 본인의 가게에 웨이팅을 요청한 고객들의 목록을 확인할 수 있다. 사업자 메뉴 창에서는 '내 계정', '행복 나누기', '내가 나눈 행복'으로의 화면 이동이 가능하고 '로그아웃'을 클릭하여 로그아웃을 할 수 있다.

6. User interface prototype



[그림 6-1] 로딩 화면

앱이 실행되었을 때 나타나게 되는 화면으로, 캐릭터가 햄버거를 먹는 모습으로 움직이는 로딩화면이다.

오류가 발생하지 않고 성공적으로 앱 실행에 성공한다면 로딩화면이 종료된 후, 로그인 화면으로 전환된다.



The image shows a mobile app login screen for 'DGAJ'. It features a light orange background with a darker orange header and footer. The title 'DGAJ Login' is centered in a bold, dark orange font. Below the title are two input fields: 'Email' and 'Password', both with orange borders and placeholder text. A dark orange '로그인' (Login) button is positioned below the password field. At the bottom, there are two links: '회원가입' (Sign Up) and '아이디/비밀번호 찾기' (Find ID/Password), separated by a vertical line.

[그림 6-2] 로그인 화면

앱 실행이 성공하였을 때(로딩화면 종료된 후) 나타나게 되는 화면으로, 기존 가입 정보가 존재한다면 이메일과 비밀번호를 입력한 후 ‘로그인 버튼’을 클릭하여 로그인을 할 수 있다. 이전에 가입한 이력이 없는 경우 ‘회원가입 버튼’을 클릭하여 회원가입으로 이동할 수 있다. 이전에 회원가입을 했을 경우, ‘아이디/비밀번호 찾기 버튼’을 클릭하면 아이디 및 비밀번호 찾기가 가능하다.

아이디/비밀번호 찾기

찾을 항목을 선택해 주세요

ID 찾기

PW 찾기

[그림 6-3] 아이디/비밀번호 찾기

이전에 회원가입을 했을 경우, ‘아이디/비밀번호 찾기 버튼’을 클릭하면 아이디 및 비밀번호 찾기가 가능하다. ‘ID 찾기’ 버튼과 ‘PW 찾기’ 버튼을 클릭하고, 주어진 양식에 맞추어 내용을 입력하면 각각 아이디와 비밀번호 찾기가 가능하다.

아이디 찾기

이름

이름 입력

전화번호

전화번호 입력

찾기

[그림 6-4] 아이디 찾기

‘아이디/비밀번호 찾기’ 화면에서 ‘ID 찾기’ 버튼을 클릭하면 아이디 찾기가 가능하다. 이름과 전화번호를 입력하고, 찾기 버튼을 누르면 DB에 존재하는지 여부에 따라 ID나 경고 메시지가 출력된다.

비밀번호 찾기

이름

이름 입력

전화번호

전화번호 입력


아이디

이메일 입력

찾기

[그림 6-5] 비밀번호 찾기

‘아이디/비밀번호 찾기’ 화면에서 ‘ID 찾기’ 버튼을 클릭하면 아이디 찾기가 가능하다. 이름과 전화번호, 그리고 아이디를 입력하고, 찾기 버튼을 누르면 DB에 존재하는지 여부에 따라 PW나 경고 메시지가 출력된다.



[그림 6-6] 회원가입

로그인 화면에서 ‘회원가입’ 버튼을 클릭하면 회원가입 유형 선택 화면으로 이동이 가능하다.
사용자는 회원과 사업자에 따라 서로 다른 회원가입 창으로 이동이 가능하다.

User 회원가입

아이디@gmail.com ▼

아이디 체크

비밀번호

이름

전화번호 010 --

등록

[그림 6-7] 회원가입(Customer)

‘회원가입’ 화면에서 ‘회원’ 버튼을 클릭하면 고객 회원가입 화면으로 이동이 가능하다. 양식에 따라 내용을 기입하고, 조건에 만족한다면, 등록 버튼을 눌렀을 때 고객 유형으로 회원가입이 가능하다.

Owner 회원가입

식당
고유번호

등록 확인

아이디

@gmail.com ▼

아이디 체크

비밀번호

이름

전화번호

010 -

-

회원 가입

[그림 6-8] 회원가입(Owner)

‘회원가입’ 화면에서 ‘사업자’ 버튼을 클릭하면 사업자 회원가입 화면으로 이동이 가능하다. 양식에 따라 내용을 기입하고, 조건에 만족한다면, 등록 버튼을 눌렀을 때 사업자 유형으로 회원가입이 가능하다. 사업자로 회원가입을 할 경우에는 본인의 가게 고유번호를 입력하여 등록된 가게인지를 확인하는 절차가 필요하다.

[그림 6-9] 검색 화면

고객 아이디로 로그인에 성공하면 위와 같이 메뉴를 검색할 수 있는 검색 화면으로 전환된다. 상단의 위치 입력창에 위치(ex. 수성구)를 입력하여 기준이 될 위치를 지정하고 화면 중단부의 메뉴 입력창에 메뉴(ex. 한식)를 입력한 후, '검색' 버튼을 클릭하면 입력받은 입력한 위치를 기준으로 입력한 메뉴에 해당하는 가게 목록들이 나타나는 메인 화면으로 전환된다. 이 외에도 'TOP 10', '메뉴 정하기' 등 다양한 기능의 이용이 가능하다.

Top 10

장모님국밥

한식

053-425-9347

대구광역시 중구 삼덕동2가 149-6

춘천옥

한식

053-422-3333

대구광역시 중구 동인동4가 4

조뽕

양식

070-8888-0505

대구광역시 중구 봉산동 37-33

토담집

한식

053-255-3257

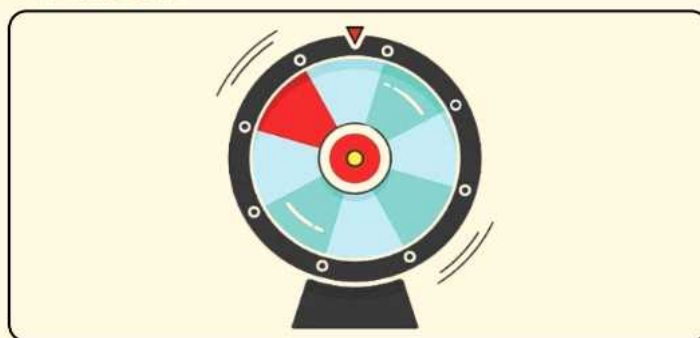
대구광역시 중구 남산동 464

[그림 6-10] TOP 10

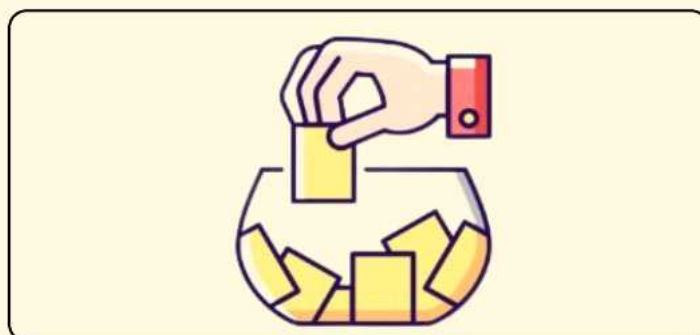
검색 화면에서 'Top 10' 버튼을 누르면 위와 같은 Top 10 화면으로 전환된다. Top 10 화면에는 찜 개수와 거리 등의 시스템에서 설정한 정렬 기준을 바탕으로 가장 상위에 있는 10개의 가게들의 리스트가 나타난다.

GAMES

GAME 1 | 룰렛 돌리기



GAME 2 | 제비뽑기



[그림 6-11] 메뉴 정하기

검색 화면에서 ‘메뉴 정하기’ 버튼을 클릭하여 게임 선택 화면으로 이동할 수 있다. 사용자는 ‘룰렛 돌리기’ 또는 ‘제비뽑기’ 게임을 활용하여 재미있고 랜덤하게 메뉴선택이 가능하다.

메뉴 정하기

게임에 넣을 메뉴들을 입력해주세요 (6개)

마라탕

초밥

삼겹살

피자

냉면

쌀국수

다음

돌리기

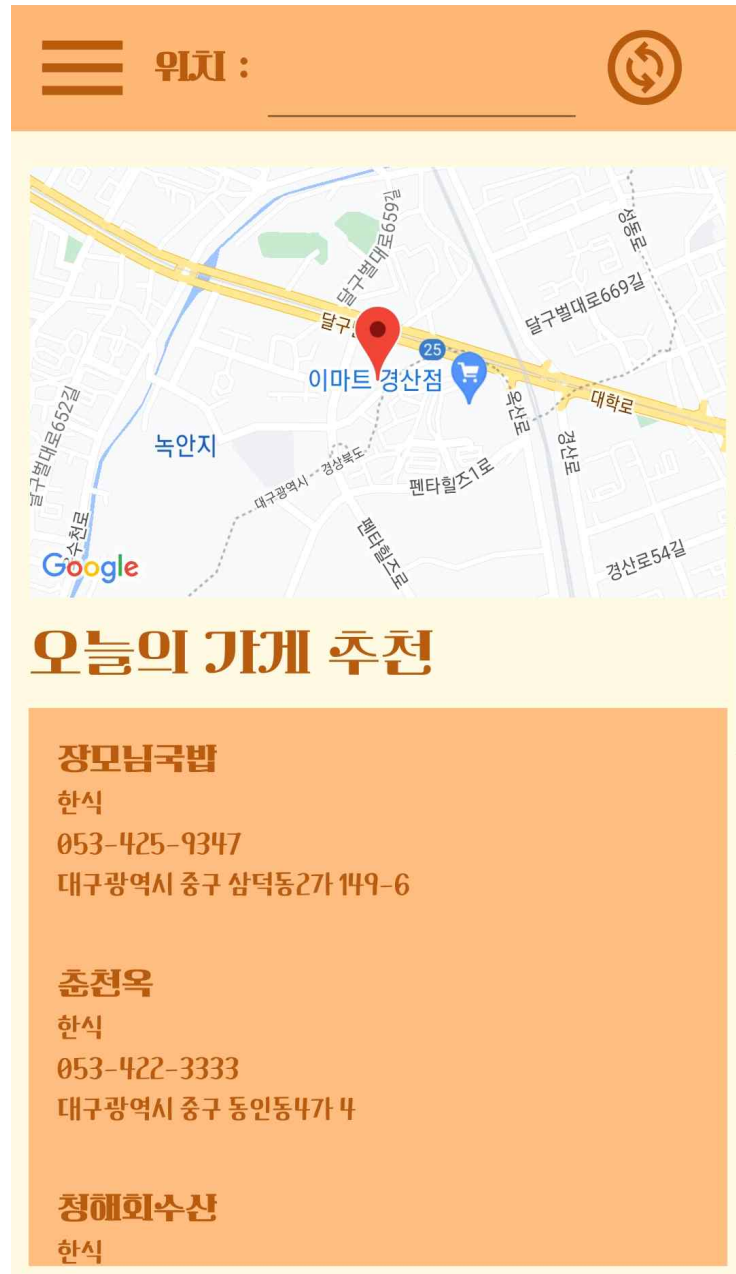
[그림 6-12] 룰렛 돌리기

게임 선택 화면에서 ‘룰렛 돌리기’ 버튼을 클릭하면 위와 같이 ‘룰렛 돌리기’ 게임 화면이 나타난다. 사용자가 6가지 항목들을 입력하여 ‘다음’ 버튼을 누르면 입력한 항목들로 이루어진 룰렛이 생성되고, ‘돌리기’ 버튼을 클릭하면 랜덤으로 하나의 메뉴가 팝업 화면으로 출력된다.



[그림 6-13] 제비 뽑기

게임 선택 화면에서 ‘제비뽑기’ 버튼을 누르면 제비뽑기 게임 화면으로 전환된다. 사용자는 제비뽑기를 하고 싶은 메뉴명이나 종류를 Input Menu에 입력하고 메뉴 넣기 버튼을 눌러 List에 항목들을 추가할 수 있고, ‘제비뽑기’ 버튼과 ‘초기화’ 버튼을 이용하여 게임 실행 및 List 초기화가 가능하다.



[그림 6-14] 사용자 메인 화면

검색 화면에서 현재 위치와 메뉴를 입력한 후 '검색' 버튼을 누르면 위와 같은 고객 메인 화면으로 전환된다. 고객 메인 화면에는 고객이 입력한 위치를 기반으로 검색한 메뉴의 주변 가게들의 리스트가 지도와 함께 나타난다. 사용자는 스크롤 기능을 이용하여 많은 가게 정보들을 확인할 수 있고, 가게를 클릭하면 해당 가게의 상세 정보 페이지로의 이동도 가능하다.



[그림 6-15] 가게 상세 페이지

고객 계정으로 로그인에 성공하여 메뉴 검색을 하였을 때, 목록에서 가게를 선택하면 위와 같이 해당 가게의 상세 정보를 나타내는 가게 상세 페이지로 화면이 전환된다. 사용자는 해당 페이지를 통해 가게의 상세 정보와 대표 메뉴 등을 확인할 수 있고, ‘출서기’ 버튼과 ‘가게 찜하기’ 버튼을 활용하여 웨이팅 신청과 해당 가게를 저장 목록에 등록이 가능하다.

가게 상세 정보 페이지에서 줄서기 버튼을 클릭하면 위와 같이 대기 신청 화면으로 전환된다. 전화번호, 이름, 인원수 등을 기입하여 확인 버튼을 누르면 팝업 창을 통해 실시간으로 현재 대기 현황 확인이 가능하고, 'YES'버튼을 클릭하면 대기 신청이 완료된다.



[그림 6-17] 고객 메뉴 사이드바

고객 계정으로 로그인에 성공하였을 때, 메뉴 아이콘이 있는 화면에서 메뉴 아이콘을 클릭하면 위의 사진과 같이 사용자 사이드바가 화면 좌측에서 나타나게 된다. 사용자는 사이드바를 통해 ‘나의 저장 목록’, ‘대기 현황’, ‘내 계정’으로의 화면 이동이 가능하고 ‘로그아웃’을 클릭하여 로그아웃을 할 수 있다.



[그림 6-18] 나의 저장 목록

고객 사이드바에서 '나의 저장 목록'을 클릭하면 위와 같이 나의 저장 목록을 화면으로 전환된다. 사용자는 해당 페이지를 통하여 본인이 찜(저장)했던 가게들의 목록을 살펴볼 수 있고, 가게를 선택하여 해당 가게의 상세 정보 페이지로 이동하거나 해당 가게의 찜 해제를 할 수 있다.



[그림 6-19] 대기 현황

고객 사이드바에서 '대기 현황'을 클릭하면 위와 같이 대기 현황 화면으로 전환된다. 사용자는 해당 페이지를 통하여 본인이 대기 신청을 한 가게의 이름과 현재 대기 현황 및 대기 순번을 확인할 수 있고, '대기 취소' 버튼을 클릭하여 대기 취소도 가능하다.



[그림 6-20] 내 계정

사이드바에서 ‘내 계정’을 클릭하면 위 그림과 같이 정보 수정 화면으로 넘어간다. 사용자는 비밀번호 변경, 전화번호 변경, 탈퇴하기 기능을 사용할 수 있다.

비밀번호 변경

기존 비밀번호

기존 비밀번호 입력 

확인

변경 비밀번호

변경할 비밀번호... 

비밀번호 확인

변경할 비밀번호... 

변경

[그림 6-21] 비밀번호 변경 화면

사이드바에서 ‘내 계정’을 클릭하여 비밀번호 변경 버튼을 클릭하면 위와 같이 비밀번호 변경 화면으로 전환된다. 사용자는 기존 비밀번호를 입력하여 존재 여부를 확인한 후, 새로운 비밀번호를 입력하여 기존 비밀번호를 업데이트할 수 있다.

전화번호 변경

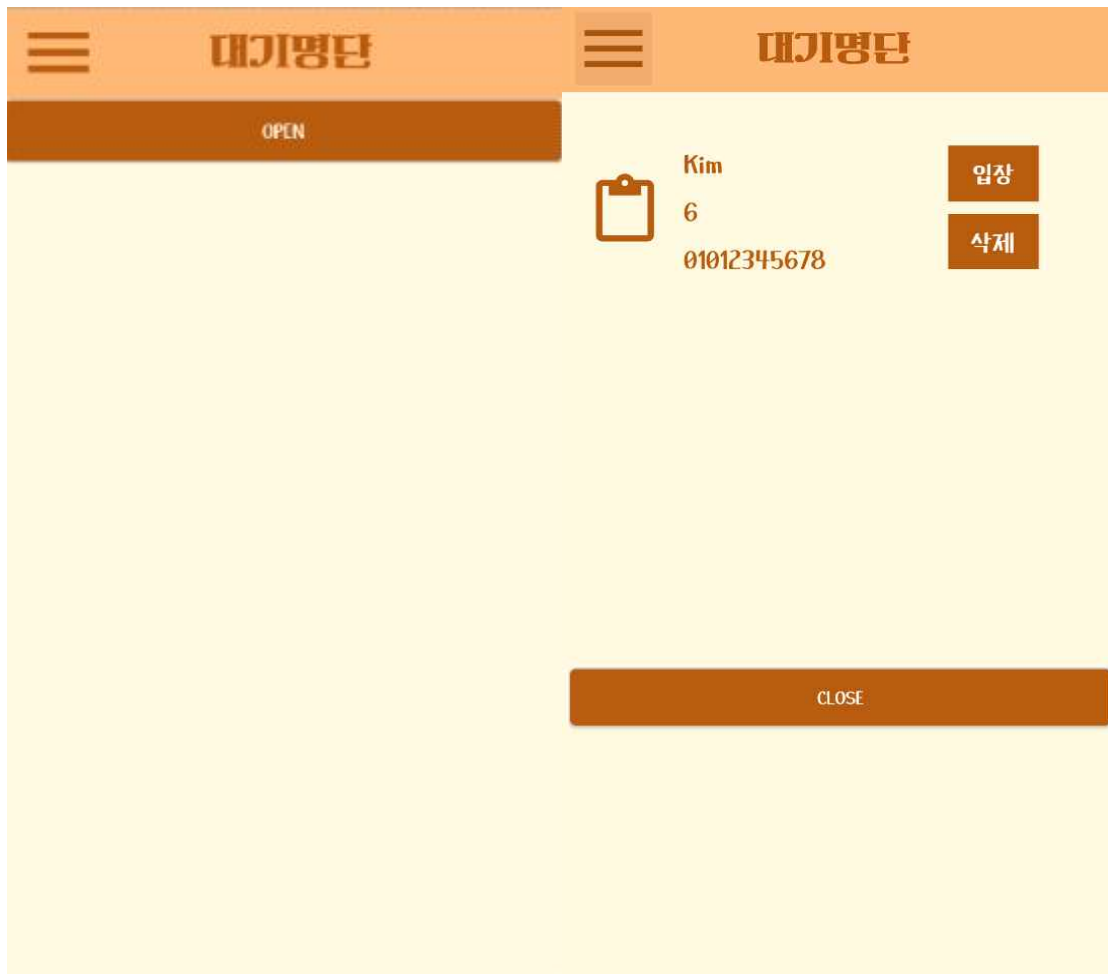
기존 전화번호

변경 전화번호

전화번호 확인

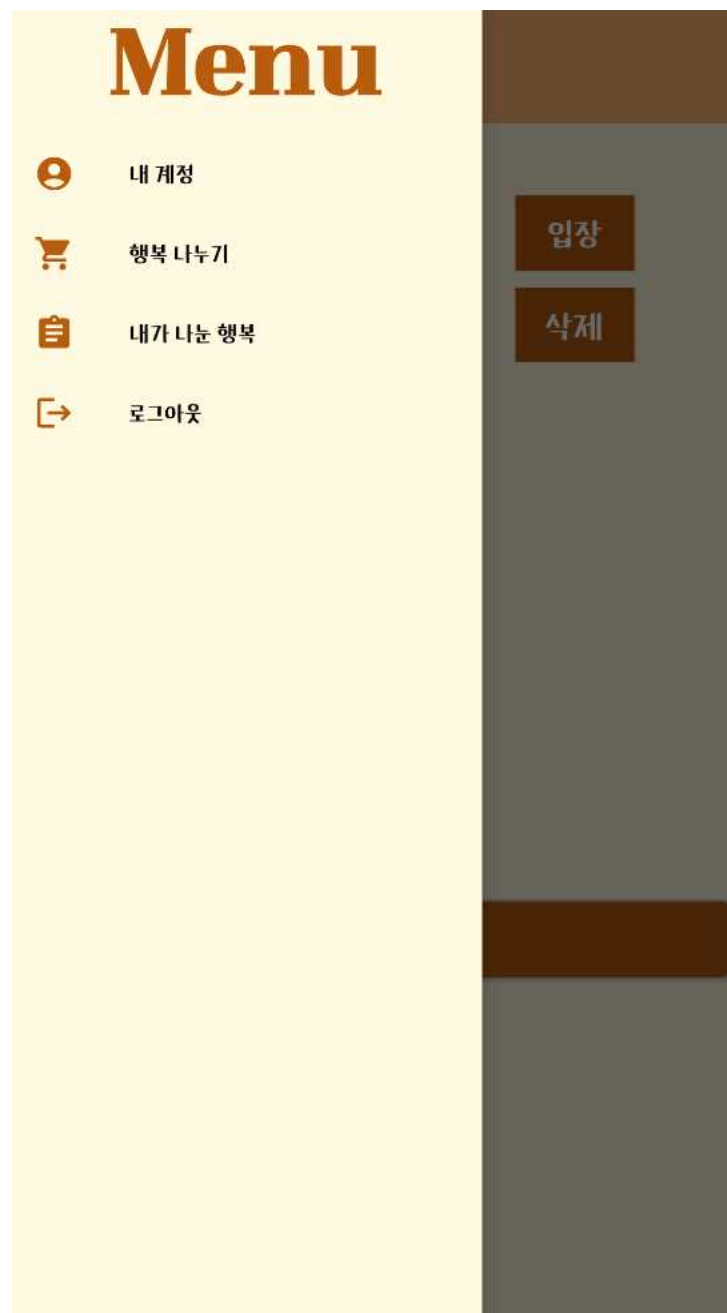
[그림 6-22] 전화번호 변경 화면

사이드바에서 '내 계정'을 클릭하여 전화번호 변경 버튼을 클릭하면 위와 같이 전화번호 변경 화면으로 전환된다. 사용자는 기존 전화번호를 입력하여 존재 여부를 확인한 후, 새로운 전화번호를 입력하여 기존 전화번호를 업데이트할 수 있다.



[그림 6-23] 사업자 메인 화면

사업자 계정으로 로그인에 성공할 경우 위와 같이 현재 해당 가게의 대기 명단이 메인 화면으로 노출되게 된다. 사업자는 'OPEN' 버튼과 'CLOSE' 버튼을 활용하여 본인 가게의 웨이팅을 활성화나 비활성화할 수 있고, '입장' 버튼과 '삭제' 버튼을 통하여 실시간으로 대기 현황을 업데이트할 수 있다.



[그림 6-24] 사업자 메뉴 사이드바

사업자 계정으로 로그인에 성공하였을 때, 메뉴 아이콘이 있는 화면에서 메뉴 아이콘을 클릭하면 위의 사진과 같이 사업자 사이드바가 화면 좌측에서 나타나게 된다. 사용자는 사이드바를 통해 ‘사업자 등록’, ‘가게 상세 정보 변경’, ‘내 계정’으로의 화면 이동이 가능하고 ‘로그아웃’을 클릭하여 로그아웃을 할 수 있다.



[그림 6-25] 행복 나누기

사업자 사이드바에서 ‘행복 나누기’ 항목을 클릭하면 위와 같이 행복 나누기 화면으로 전환이 된다. 사업자는 목록에 있는 센터들 중 하나를 선택하여 해당 센터로 각종 물품들을 나누어줄 수 있다. ‘기부’ 버튼을 누르면 ‘기부하기’ 페이지로 화면이 전환된다.

기부 하기

보내시는 분

나눌 물품

수량

전화번호 010 - -

[그림 6-26] 기부 하기

행복 나누기 페이지에서 '기부' 버튼을 클릭하면 위와 같이 '기부하기' 화면으로 전환된다. 사업자는 보내는 사람과 물품을 형식에 따라 입력한 후, '완료' 버튼을 눌러서 해당 센터로 물품을 기부할 수 있다.

지금까지 나는 행복

기부자 이름: 너구리
센터 이름: 구름아동센터
물품: 햄버거
개수: 1

기부자 이름: Jung
센터 이름: 미소아동센터
물품: Phone
개수: 2

기부자 이름: Song
센터 이름: 구름아동센터
물품: Cards
개수: 30

2023년 12월 06일까지
나는 행복은 3 회 입니다.

[그림 6-27] 내가 나누 행복

사업자 사이드바에서 '내가 나누 행복' 항목을 클릭하면 위와 같이 내가 나누 행복 화면으로 전환이 된다.
해당 페이지에서는 사업자 자신이 기부한 센터들의 목록이 리스트 형태로 나타난다.

7. Implementation requirements

<H/W platform requirements>

Application

- Processor: Qualcomm Snapdragon 835 MSM8998+
- RAM: 4GB+
- Storage: < 200MB
- Network: connected to WAN over TCP/IP

Server

- Firebase의 경우 서버의 사양을 직접 제어하지 않음. 대신, Google Cloud Platform에서 관리되는 서버에서 호스팅 되는, Firebase의 다양한 서비스를 이용한다.

<S/W platform requirements>

Application

- Android Studio 2020.3.1+
- Android SDK Tools 23+
- JDK 1.8.0+
- JRE 1.8.0+
- App OS: Pixel 3a API 34
- Develop Environment OS: Windows 8/10
- Implementation Language: Java, xml

Server

- Firebase SDK: Firebase 서비스를 이용하기 위한 SDK를 사용.
- DBMS: Firebase Realtime Database 또는 Firestore
- Authentication: Firebase Authentication
- Storage: Firebase Cloud Storage

8. Glossary

Words	Account
Customer	앱을 이용하는 사용자(고객)
Owner	앱을 이용하는 사업자
WaitingAdapter	Waiting class가 ListView를 가져오기 위해 도움을 주는 class
RestaurantAdapter	RecyclerView에 식당 목록을 표시하기 위한 어댑터 클래스.
GetMyGeoPoint	현재 장치의 위치 정보를 가져오는 기능을 제공하는 클래스.
FirebaseID	FirebaseID 클래스는 Firebase Firestore에서 사용되는 컬렉션 및 문서 식별자를 관리하는 클래스.

9. References

- [1] Ian Sommerville, Software Engineering, 10th Edition, Pearson, 2016.
- [2] Ian Sommerville, Engineering Software Products: An Introduction to Modern Software Engineering, Pearson, 2021.
- [3] Shari Lawrence Pfleeger and Joanne M. Atlee, Software Engineering: Theory and Practice, 4th Edition, Pearson, 2009.
- [4] 파이어베이스 주소
https://console.firebase.google.com/u/1/project/dgaj-2a484/firestore/data/~2Fid_list~2F3mmuIIISYvVU8Vd0CE810ZxBI1Tg2?hl=ko
- [5] Google Cloud Console
<https://console.cloud.google.com/apis/credentials?authuser=1&project=dgaj-2a484&supportedpurview=project>
- [6] 공공데이터 포털
<https://www.data.go.kr/index.do>