

# 이번엔 어디 갈까?

## (TravelNext)



**정보통신대학 컴퓨터공학과**

# 서울과학기술대학교

1. 개요 및 목적.....	4
1.1 프로젝트 개요	4
1.2 개발 동기	4
2. 요구명세.....	5
2.1 User Requirements	5
2.1.1 로그인 및 로그아웃	5
2.1.2 여행지 기록	5
2.1.3 여행지 검색	6
2.1.4 여행지 추천	7
2.1.5 Dark Theme	8
2.2 System Requirements	9
2.2.1 Functional Requirement	9
2.2.2 Non-Functional Requirement	14
3. 설계 및 구현.....	15
3.1 프로젝트 일정	15
3.2 개발 환경	15
3.3 설계	16
3.3.1 Software Architecture	16
3.3.2 State Diagram	17

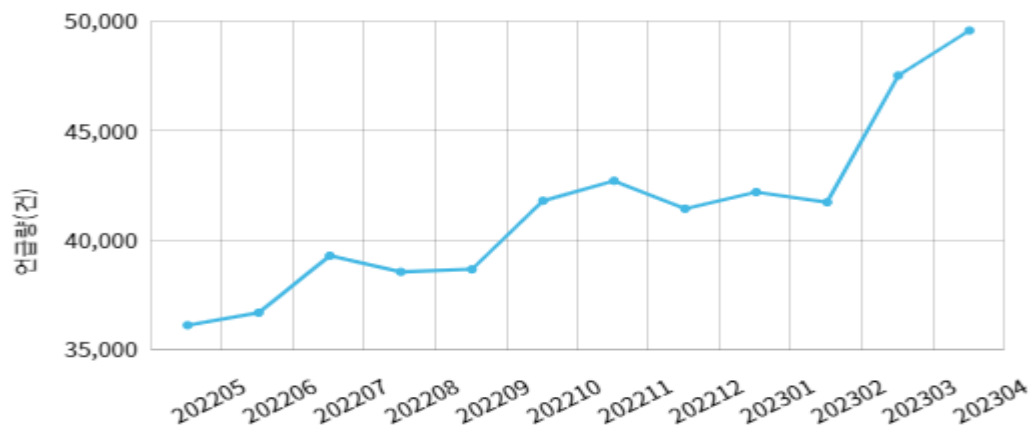
3.3.3 E-R Diagram	17
3.3.4 Application Structure	18
3.4 구현 및 적용기술	19
4. 실행 결과 .....	21
4.1 test 환경	21
4.2 test 결과	21
5. 향후계획.....	31
6. 참고문헌.....	31

# 1. 개요 및 목적

## 1.1 프로젝트 개요

‘이번엔 어디 갈까?’는 여행지 추천과 여행 검색 및 기록의 편리성을 제공하는 소프트웨어 개발을 목적으로 한다. 여행지 정보는 ‘Tour Api’를 이용한 국내여행지로 한국관광공사에서 만든 ‘한국관광지’를 기준으로 한다. 여행지 정보 검색뿐만 아닌 사용자의 여행 기록 정보를 이용한 비슷한 여행지나 색다른 여행지 추천을 받을 수 있다. 여행지 방문 후 사용자는 지도를 앨범으로, 지도의 마커에 사진을 등록하여 사용자가 경험한 여행을 기록할 수 있다.

## 1.2 개발 동기



<SNS, 커뮤니티에서의 연급량>

코로나 시국이 끝나고 여행에 대한 관심사가 점점 커지고 있다. 여행에 도움을 주는 여행지 탐색과 기록을 하나에 담은 어플을 기획하게 되었다. 또한 추천 기능은

사용자가 선호하는 여행지 기반으로 추천하기에 이를 통해 여행 수요가 더욱 늘어날 것으로 기대된다.

## 2. 요구명세

### 2.1 User Requirements

#### 2.1.1 로그인 및 로그아웃

- Kakao 계정을 이용하여 로그인을 할 수 있다.
- 기기에 카카오톡 앱이 미 설치되어 있다면, 직접 kakao 계정을 입력하여 로그인 할 수 있다.
- 로그아웃은 하단 네비게이션의 'My' 탭에서 가능하다.
- My 페이지는 사용자의 카카오톡 프로필 사진과 닉네임을 확인할 수 있고, 로그아웃 버튼 클릭 시 앱의 계정이 로그아웃 되고 시작화면인 로그인 페이지로 이동된다.

#### 2.1.2 여행지 기록

- 하단 네비게이션의 'Travelog' 탭 선택 시 이동된다.
- 기록을 확인할 수 있는 페이지는 지도형식과 리스트 형식으로 확인할 수 있고 각 형식의 이동은 하단의 버튼을 통해 전환가능하다.
- 지도형식에서 사용자의 카카오톡 닉네임을 표시하여 '(닉네임)은/는 이번엔 어디 갈까?'를 확인할 수 있다.

- 지도형식에서 기록할 여행지의 위치를 클릭 시, 기록 페이지로 이동하고 여행지 이름과 여행지의 카테고리, 여행지 방문일, 내용 그리고 마지막으로 여행지 사진을 등록할 수 있다.
- 지도형식은 기록된 정보의 사진 데이터가 마커로 표시되어 여행 기록을 한눈에 확인 가능하고, 마커들의 클러스터링 기능이 적용되어 있다.
- 리스트형식은 기록된 정보의 이미지와 여행지 이름, 방문일을 표시한 아이템들로 확인할 수 있다.
- 리스트형식의 아이템들은 '작성일 기준', '방문일 기준', '여행지 이름 기준'으로 오름차순, 내림차순으로 정렬하여 확인할 수 있다.
- 리스트형식의 기록된 아이템을 선택하면, 해당 아이템의 상세정보를 확인할 수 있는 페이지로 이동한다.
- 상세정보 페이지는 기록된 정보의 이미지와 여행지 이름, 방문일, 내용을 확인할 수 있다.
- 리스트형식의 기록된 아이템을 길게 선택하면, 해당 아이템의 삭제 여부를 결정할 수 있다.

### 2.1.3 여행지 검색

- 하단 네비게이션의 'Search' 탭 선택 시 이동된다.
- 검색 페이지에서 여행 키워드를 통해 검색이 가능하다.
- 이전에 검색한 키워드들을 사용한 자동완성기능으로 검색할 수도 있다.

- 검색 결과를 보여주는 페이지에서 키워드에 해당하는 여행지들의 '이미지, 여행지 이름, 여행지 장소'를 리스트형식으로 확인할 수 있다.
- 리스트형식의 아이템을 클릭 시, 웹뷰를 통해 네이버에 검색된 정보를 얻을 수 있다.
- 검색 후 재검색도 가능하다.

#### 2.1.4 여행지 추천

- 하단 네비게이션의 'My' 탭 선택 시 이동된다.
- 사용자의 여행지 기록들을 참고해서 '비슷한 유형', '색다른 유형' 등을 구별하여 여행지 추천을 받을 수 있다.
- '비슷한 유형'의 여행지 추천은 사용자의 여행지 기록의 빈도가 가장 높은 카테고리의 여행지들을 추천한다.
- '색다른 유형'의 여행지 추천은 사용자의 여행지 기록의 빈도가 가장 낮은 카테고리의 여행지를 추천한다.
- 카테고리를 재 클릭 시, 추천 여행지를 새로 받을 수 있다.
- 여행지 추천 리스트는 5 개를 받을 수 있고, 추천 여행지 이미지와 이름으로 구성된 아이템으로 구성된다.
- 리스트의 아이템을 클릭 시, 여행지의 주소를 확인할 수 있다.

### 2.1.5 Dark Theme

- 사용자는 기기를 다크모드로 전환 시, 다크모드에 맞는 theme 를 제공받을 수 있다.



## 2.2 System Requirements

### 2.2.1 Functional Requirement

분류	요구사항명	요구사항 내용
LOGIN 페이지	유저 인증	<ul style="list-style-type: none"> <li>- 카카오를 이용한 로그인 기능 카카오톡이 있는 경우엔 앱을 통해 로그인을 진행한다. 설치되지 않은 경우 카카오 계정을 통해 로그인을 진행한다.</li> <li>- 로그인 실패 시, 스낵바를 통한 실패 메시지를 표시한다.</li> <li>- 카카오 계정의 이메일을 이용하여 Firebase의 인증을 받는다.</li> <li>- Firebase에 등록되지 않은 사용자라면 카카오 이메일과 UID를 통해 사용자 등록을 진행한다.</li> <li>- 로그인 성공 시, 메인 화면으로 이동시킨다.</li> </ul>
TRAVELOG 페이지	여행 기록 확인 (LIST)	<ul style="list-style-type: none"> <li>- 사용자의 여행 기록을 리스트 형식으로 표시한다.</li> <li>- 리스트의 아이템들은 '여행지 이름, 방문일, 이미지'를 표시한다.</li> </ul>

		<ul style="list-style-type: none"> <li>- 사용자가 아이템을 클릭하면, 해당 여행 기록의 상세 정보 페이지로 이동한다.</li> <li>- 사용자가 아이템을 길게 클릭하면, 아이템 삭제 여부를 확인하는 Dialog 가 표시된다. 사용자가 '삭제'를 선택하면, 선택된 여행 기록은 Firestore 데이터베이스에서 삭제된다.</li> <li>- 사용자는 리스트를 '작성일 기준, 방문일 기준, 이름 기준'으로 오름차순, 내림차순 정렬할 수 있다.</li> <li>- 사용자가 하단의 맵으로 전환되는 버튼을 클릭 시, 맵 페이지로 이동시킨다.</li> </ul>
TRAVELOG 페이지	여행 기록 확인 (MAP)	<ul style="list-style-type: none"> <li>- 사용자 정보를 이용하여 문구를 표시한다. '(닉네임)은/는 이번엔 어디 갈까?' 만약 닉네임이 설정되어 있지 않다면 기본으로 '여행자'가 입력된다.</li> <li>- 여행 기록을 확인할 수 있는 맵은 '한국'의 위치를 중심으로 설정한다.</li> <li>- 사용자는 지도를 확대 및 축소할 수 있다.</li> </ul>

		<ul style="list-style-type: none"> <li>- 지도 위의 마커들은 사용자가 기록한 여행정보를 기반한 위치와 이미지로 표시된다.</li> <li>- 지도의 마커들은 클러스터링이 가능하다.</li> <li>- 사용자가 지도의 위를 클릭 시, 위치정보를 가지고 여행 기록 페이지로 이동한다.</li> <li>- 사용자가 하단의 리스트로 전환되는 버튼을 클릭 시, 리스트 페이지로 이동시킨다.</li> </ul>
TRAVELOG 페이지	여행 기록 작성	<ul style="list-style-type: none"> <li>- '여행지 이름, 방문 날짜, 여행지 카테고리, 내용, 이미지'를 포함하여 여행 기록을 작성할 수 있다.</li> <li>- 업로드한 이미지는 삭제 후, 재업로드 할 수 있다.</li> <li>- 작성한 여행 기록을 Firestore 에 업로드할 수 있다.</li> <li>- 업로드 성공 시, 지도 화면으로 이동된다. 업로드 실패 시, 스낵바를 통한 실패 메시지를 표시한다.</li> <li>- 기록 작성 중 사용자는 작성을 취소하고 뒤로가기 버튼을 통해 이전 화면(지도)으로 돌아갈 수 있다.</li> </ul>

TRAVELOG 페이지	상세 기록 확인	<ul style="list-style-type: none"> <li>- 상세 기록 페이지는 '여행지 이름, 방문일, 내용, 이미지'를 확인할 수 있다.</li> <li>- 각 정보는 Firestore 에서 로드한다.</li> <li>- 뒤로가기 버튼을 클릭 시, 이전 화면(리스트)로 돌아갈 수 있다.</li> </ul>
SEARCH 페이지	여행지 검색	<ul style="list-style-type: none"> <li>- 사용자는 텍스트 입력란에 검색어를 입력할 수 있다.</li> <li>- 입력된 검색어는 로컬 저장소에 저장한다.</li> <li>- 로컬 저장소에 저장된 사용자가 이전에 입력한 검색어를 자동 완성 제안으로 표시한다.</li> <li>- 'Search!'버튼을 클릭 시, 검색 결과를 확인할 수 있는 페이지로 이동한다.</li> <li>- 검색 결과는 리스트형식으로 사용자에게 보여준다.</li> <li>- 검색은 Tour API 의 '관광사진 갤러리 키워드 검색 목록 조회'를 이용한다.</li> <li>- 리스트의 각 아이템은 '이미지, 여행지 이름, 여행지 장소'로 표시한다.</li> <li>- 아이템을 클릭 시, 해당 여행지를 검색한 웹뷰를 Dialog 로 띄워 상세 정보를 확인할 수 있다.</li> </ul>

		<ul style="list-style-type: none"> <li>- 검색 결과 페이지에서 사용자는 다시 검색할 수 있다.</li> </ul>
RECOMMAND 페이지	사용자 정보 확인	<ul style="list-style-type: none"> <li>- 카카오계정의 사용자 정보를 이용하여 사용자의 프로필 사진과 이미지를 표시한다.</li> </ul>
RECOMMAND 페이지	로그아웃	<ul style="list-style-type: none"> <li>- 사용자는 로그아웃 버튼을 클릭하여 Firebase 와 카카오 계정에서 로그아웃 할 수 있다.</li> <li>- 로그아웃 후 로그인 페이지로 이동한다.</li> </ul>
RECOMMAND 페이지	여행지 추천	<ul style="list-style-type: none"> <li>- Firestore 에 업로드 되어 있는 사용자의 여행 카테고리를 기반으로 추천여행지 키워드를 선정한다.</li> <li>- 빈도가 높은 카테고리 와 빈도가 낮은 카테고리를 사용자에게 추천 여행지로 제공한다.</li> <li>- 추천된 여행지들은 각 각 리스트형식으로 5 개를 추천한다.</li> <li>- 리스트의 각 아이템은 '이미지, 여행지 이름'로 표시한다.</li> </ul>

		<ul style="list-style-type: none"> <li>- 추천 아이템을 클릭 시, 숨겨져 있는 여행지의 위치정보를 확인할 수 있다.</li> </ul>
네비게이션 뷰	메뉴	<ul style="list-style-type: none"> <li>- 'Travelog, Search, My'로 구성된 하단 네비게이션 바로 구성된다.</li> </ul>

## 2.2.2 Non-Functional Requirement

분류	요구사항 내용
Product requirements	<ul style="list-style-type: none"> <li>- 평균 응답 시간은 200ms 이내가 되도록 한다.</li> <li>- 앱 오류율을 0.5% 이내로 한다.</li> </ul>
Organizational requirements	<ul style="list-style-type: none"> <li>- 개발 언어로는 Kotlin 을 사용하고 개발환경은 Android studio, Firebase 를 사용한다.</li> </ul>
External requirements	<ul style="list-style-type: none"> <li>- 권한이 없는 접근은 제한하고, 만약 발생한다면 현행 법률에 따라 처리한다.</li> </ul>

### 3. 설계 및 구현

#### 3.1 프로젝트 일정

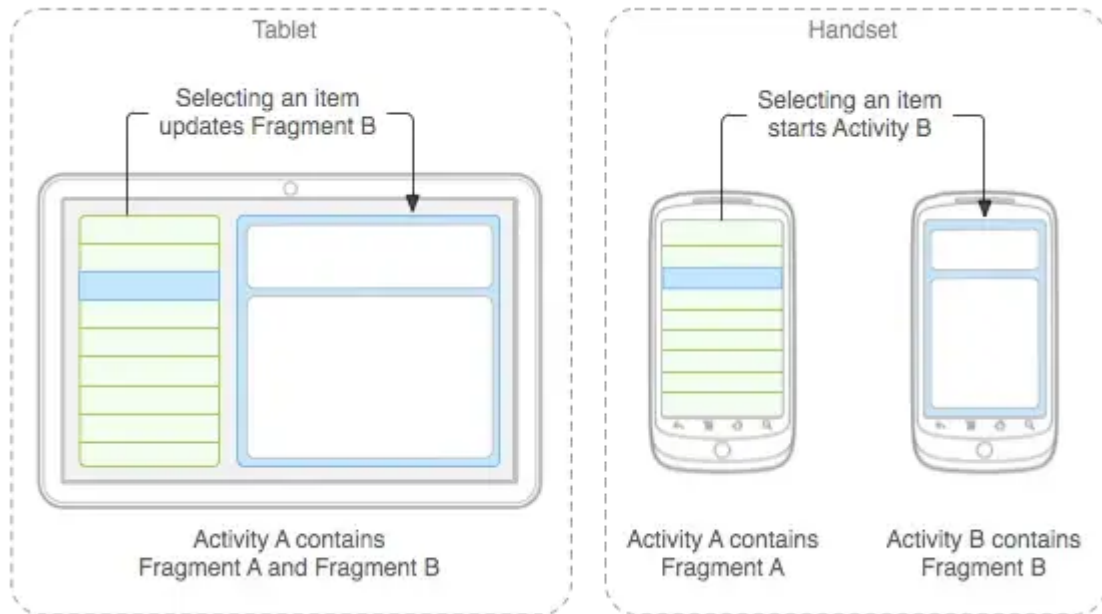
구분	내용	프로젝트 기간 (2023 년)									
		3 월	4 월	5 월	6 월	7 월	8 월	9 월	10 월	11 월	12 월
계획	서비스 기획										
	요구사항 분석										
스터디	Kotlin 스터디										
	Android										
	Firebase										
	API 활용										
디자인	UI / UX 디자인										
설계	개발 구조 및 DB 설계										
개발	앱 개발										
테스트	검증 및 기능 테스트										

#### 3.2 개발환경

구분		상세내용
S/W 개발환경	OS	Window
	개발환경(IDE)	Android Studio
	개발언어	Kotlin
프로젝트 관리환경	형상관리	Github
Android version		Android 9(Pie) ~ Android 13

### 3.3 설계

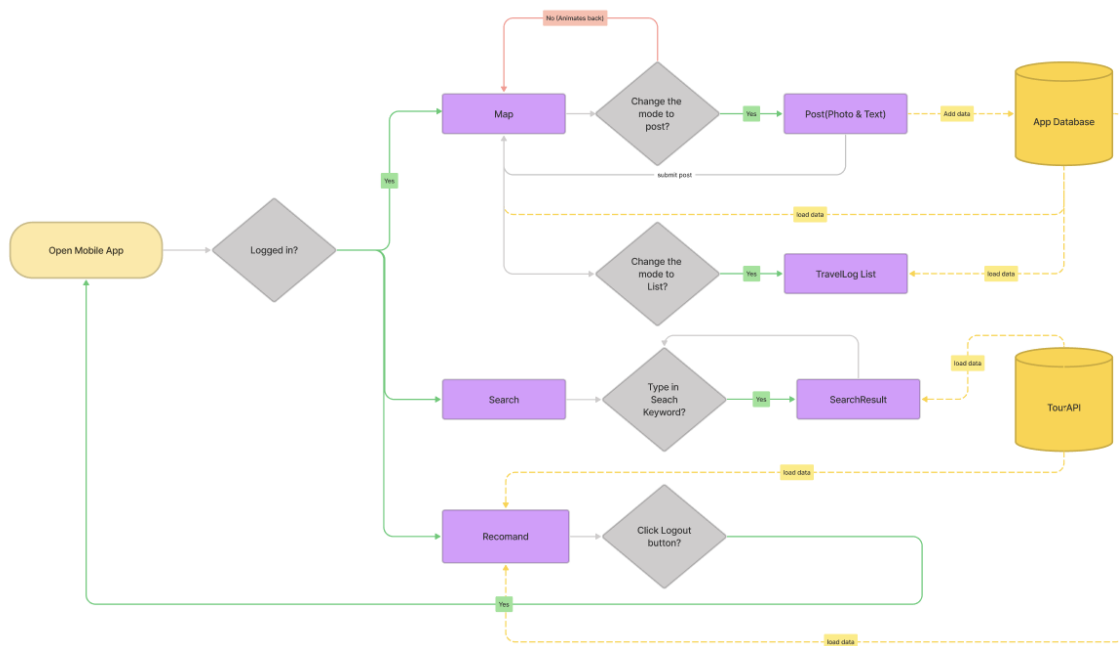
#### 3.3.1 Software Architecture



‘이번엔 어디 갈까?’는 적은 액티비티와 대부분의 fragment 들로 구성되어, 이동과 데이터전달을 NavHostFragment 와 NavController 를 이용하는 Jetpack Navigation Component 를 이용하여 구현되었다. 그렇기 때문에 주로 JetPack Navigation 과 함께 사용되는 구조인 Single Activity Architecture 을 이용하여 설계하게 되었다. SAA(Single Activity Architecture)란, 하나 혹은 적은 개수의 Activity 만을 사용하고 나머지 화면은 Fragment 로 구성한 구조이다. 이를 통한 이점으로는, Activity 는 Fragment 에 비하여 상대적으로 무겁기 때문에 메모리나 속도의 효율이 좋아진다. Fragment 는 상대적으로 가볍고, 데이터 공유, UI 의 이점, 관심사의 분리가 편한 이점들이 있다. 하지만 Lifecycle 이 복잡하고, Fragment 간의 동작이 비동기로 처리되기 때문에 이슈가 발생할 수 있는 단점 또한 가지고 있다.

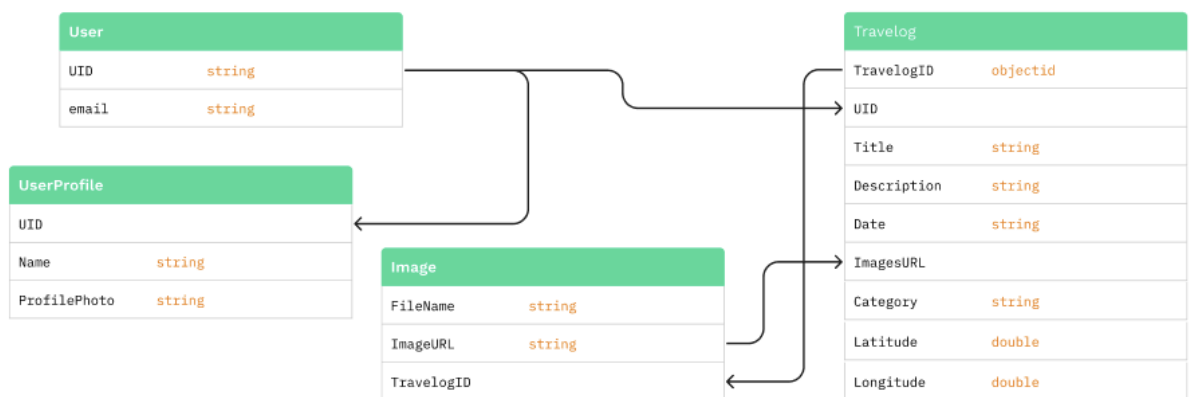


### 3.3.2 State Diagram



State Diagram 을 통해 UI Component 간의 상태 전환을 확인할 수 있다.

### 3.3.3 E-R Diagram

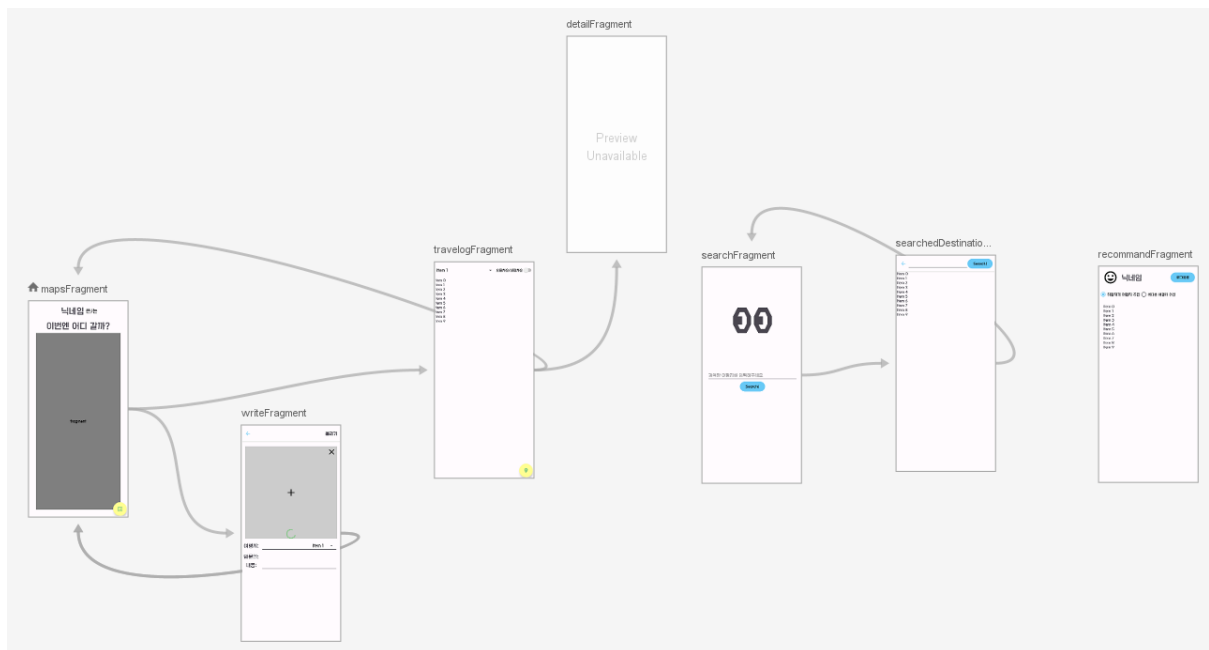


사용자의 카카오 로그인을 통해 발급받은 email 과 UID 를 User Table 에 저장한다.

그리고 사용자의 UID 를 통해 카카오 정보인 닉네임과 프로필 사진정보에 접근할 수

있다. Travelog Table 은 사용자의 여행정보가 기록된다. 사용자 마다 고유의 기록저장소를 갖기 위해서, DB 구성이 중요하다. 이를 위해 사용자 고유의 UID 를 collection 으로 사용하여 사용자별 저장소를 구성하고 여행기록을 업로드할 수 있다. imageUrl 는 대용량 이미지를 유연하게 다운받기 위해서, 먼저 firebase 의 Storage 에 업로드 후, 데이터를 로드가 필요할 때 ID 를 이용하여 다운받는 구조로 구성되어 있다.

### 3.3.4 Application Structure



Navigation 모듈의 nav\_graph.xml 을 통해 화면 전환 action 을 결정할 수 있고, fragment 간의 data 를 전달할 수 있다.

### 3.4 구현 및 적용기술

- Jetpack Navigation

사용자가 앱 내에서 탐색하는 과정을 쉽게 개발할 수 있도록 하는 컴포넌트이다.

nav\_graph.xml 를 이용한 fragment 들의 직관적인 흐름을 작성하고 사용할 수 있다.

- Tour API

여행지 검색 기능과 여행지 추천 리스트를 구현하기 위해서, 키워드 검색을 통해 조회하는 기능인 '관광사진갤러리 키워드 검색 목록 조회' 서비스를 json 형식으로 요청하여 이용한다.

- Retrofit2, converter-gson

REST API 를 사용하기 위한 HTTP 클라이언트 라이브러리이다. Service Interface 에 Callback 을 붙여주고, 이용하여 REST API 의 엔드포인트를 정의한다. 추가로 @Query 를 이용하여 요청 URL 에 매개변수를 설정할 수 있다.

- Kakao SDK - USER

카카오 로그인을 이용해 OAuth 로그인 및 사용자 정보를 제공한다. OAuth 로그인은 비밀번호를 제공하지 않기에, 서버(firebase)에서 부여받은 access token 를 통해서 카카오 서버에 접근하여 사용자 정보를 이용할 수 있다.

- Firebase

Authentication 인증 서비스를 사용하여 사용자의 로그인을 관리한다.

Firestore(Database)는 NoSQL 로 collection 과 document 를 이용한 빠른 쿼리가

가능하다. Realtime database 와 달리 기기가 오프라인 상태여도 데이터를 읽고 쓸 수 있다. Storage 는 cloud 에 있는 저장소로, 업로드된 파일의 URL 을 사용할 수 있어서 대용량 파일인 이미지를 관리하기에 용이하다.

- GoogleMap Android Maps SDK

GoogleMap 객체를 통해 지도 뷰를 제공하고 위치 데이터를 마커로 표시하여 시각적으로 표현할 수 있다. 또 다양한 마커 커스텀과 클러스터링도 가능하다.

- Glide

이미지를 로드하고 표시하는 라이브러리이다. Storage 의 이미지 URL 를 경로를 지정한 후, 다운로드하고 이미지에 표시될 뷰를 지정하여 이미지를 표시할 수 있다.

- Material3 Design

Material Design 의 12 개의 color 들을 설정하여 Day Theme 와 Dark Theme 를 구성할 수 있다.

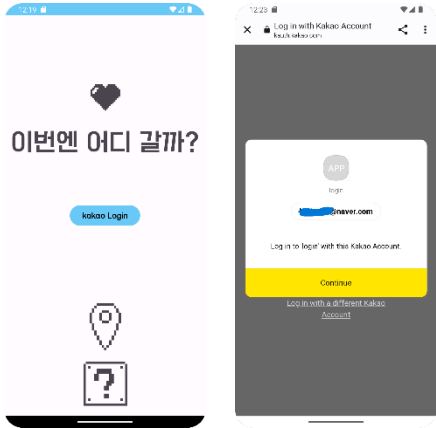
## 4. 실행결과

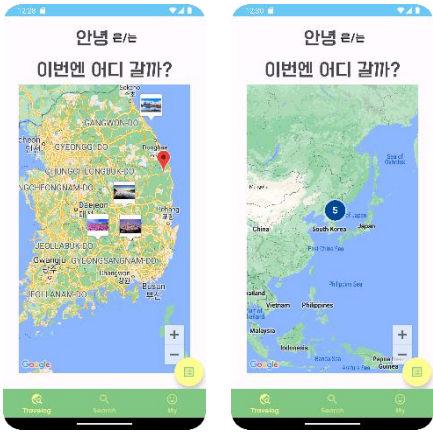
### 4.1 Test 환경

- Android 13 Emulator

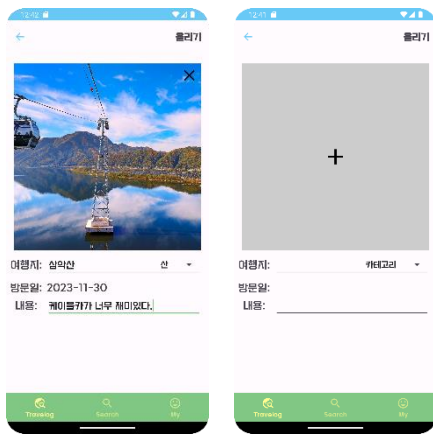


### 4.2 Test 결과

LOGIN 페이지	상세설명
 <p>[LOGIN_01]</p>	<ul style="list-style-type: none"> <li>- 앱 첫 진입 화면이자 로그인 페이지로 kakao login 을 통한 회원로그인 접속이 가능하다.</li> <li>- Kakao SDK – USER 를 이용하여, isKakaoTalkLoginAvailable()를 이용하여 loginWithKakaoTalk(), loginWithKakaoAccount()중 어떤 메소드를 이용할지 선택한다. 위 두 개의 메소드는</li> </ul>

	<p>카카오톡이 설치되어 있는 사용자의 로그인과 그렇지 못해 직접 계정을 이용하는 사용자의 로그인 기능을 수행한다.</p> <ul style="list-style-type: none"> <li>- kakao 로그인을 통해 얻은 email 정보와 kakaoTalk 의 UID 를 이용하여 Firebase 인증한다.</li> <li>- Firebase 의 Auth 에 등록된 User 를 확인하여 로그인 성공여부를 판단한다.</li> <li>- 로그인을 완료하면 [LOG_01]로 이동한다.</li> </ul>
TRAVELOG 페이지	상세설명
 <p>[LOG_01]</p>	<ul style="list-style-type: none"> <li>- 여행 기록 추가와 기록 확인을 위한 맵 페이지다.</li> <li>- 상단의 문구는 kakao user 정보 중 사용자의 닉네임정보를 이용하여 보여준다. 닉네임 정보가 없는 경우, '여행자'로 표시된다.</li> <li>- moveCamera()를 이용하여 지도의 시작을 한국의 위도와 경도로 맞춰준다.</li> </ul>

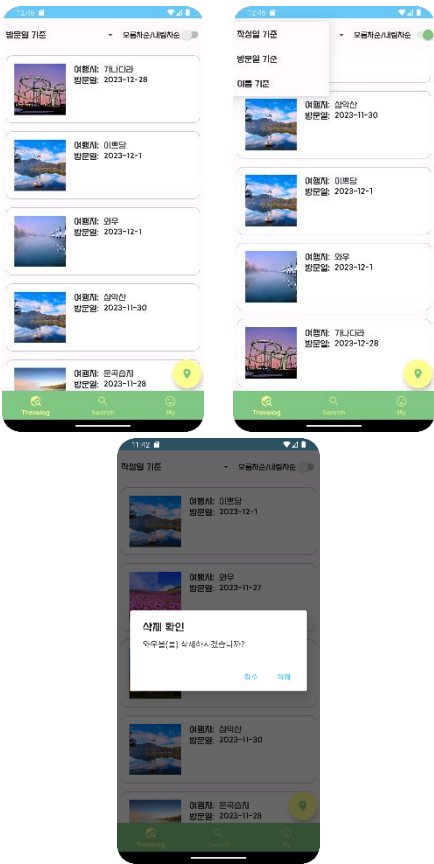
	<ul style="list-style-type: none"> <li>- 여행기록들은 Firestore 에서 MyClusterItem 객체를 통해 사용할 데이터 구조를 가져올 수 있다.</li> <li>- ClusterManger() 객체를 이용하여, 마커 클러스터링을 관리할 있다. 정의한 MyClusterRenderer 클래스를 이용하여 마커를 기록된 여행지 이미지로 커스텀되어 있다.</li> <li>- setOnMapClickListener 를 통해 지도의 클릭 이벤트를 처리해 준다. 지도 위를 클릭 시, 해당 위치정보(LatLng)을 [LOG_02]로 페이지 이동과 함께 전달한다.</li> <li>- 하단의 Floating Button 을 클릭 시, findNavController()를 이용하여 맵 형태의 기록 확인 페이지 [LOG_1]에서 리스트 형태인 [LOG_3]으로 화면으로 전환된다.</li> </ul>
--	--



[LOG\_02]

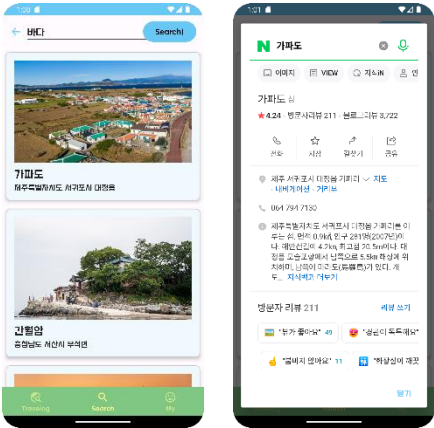
- 여행을 기록하기 위한 페이지로 기록 후 Firestore 에 업로드된다.
- findNavController()를 이용하여 뒤로가기 버튼을 구현하였고, 클릭 시 [LOG\_01]로 이동한다. 기록 작성 중에도 작성이 취소되고 이동이 가능하다.
- 사용자의 여행 이미지를 업로드하기 위해서, PickMedia 를 이용하여 내부 저장소, 갤러리의 이미지를 선택하여 업로드할 수 있다.
- 사진이 업로드 되어 있을 때는 업로드 된 사진을 삭제할 수 있고, 없을 때는 이미지를 업로드할 수 있게 ImageView 의 클릭 기능이 구분된다.
- 여행지 카테고리를 spinner 를 통해 선택할 수 있다.
- 여행지 방문일을 DataPickDialog 를 이용하여 선택할 수 있다.



	<ul style="list-style-type: none"> <li>- 올리기 버튼을 통해 Firestore 에 정의한 data class 형식으로 업로드 된다.</li> <li>- 업로드 되는 동안 진행상황을 progressbar 로 보여준다.</li> </ul>
 <p>[LOG_03]</p>	<ul style="list-style-type: none"> <li>- 여행 기록을 RecyclerView 와 Adapter 를 적용하여 리스트 형식으로 볼 수 있는 페이지다.</li> <li>- Firestore 의 collection("users").document(uid). collection("travelogs") 경로를 통해 사용자별 여행 기록을 조회하여 보여준다.</li> <li>- 이미지는 Glide 를 사용하고 다른 정보들은 data class 를 통해 사용한다.</li> <li>- 기록된 아이템을 클릭하면 findNavController()를 통해 기록을 상세히 확인할 수 있는 [LOG_04]로 이동한다.</li> </ul>

	<ul style="list-style-type: none"> <li>- 기록된 아이템을 길게 클릭하면, 사용자에게 해당 아이템의 삭제를 확인하는 AlertDialog 가 표시된다.</li> <li>- 사용자는 AlertDialog 에서 '삭제'와 '취소'를 선택할 수 있다.</li> <li>- 삭제를 선택하면, Firestore 에 해당 여행 기록을 삭제하기 위해 delete()를 사용한다. 그리고 삭제의 성공 여부에 따라, addOnSuccessListener 에 RecyclerView 의 아이템들을 다시 호출하여 삭제된 아이템을 반영한다.</li> <li>- 리스트의 아이템들은 sortBy()를 이용하여 '작성일 기준, 방문일 기준, 이름 기준'으로 오름차순, 내림차순 정렬이 가능하다.</li> <li>- 하단의 Floating Button 을 클릭 시, findNavController()를 이용하여 리스트 형태의 기록 확인 페이지 [LOG_3]에서 맵 형태인 [LOG_1]으로 화면으로 전환된다</li> </ul>
--	---

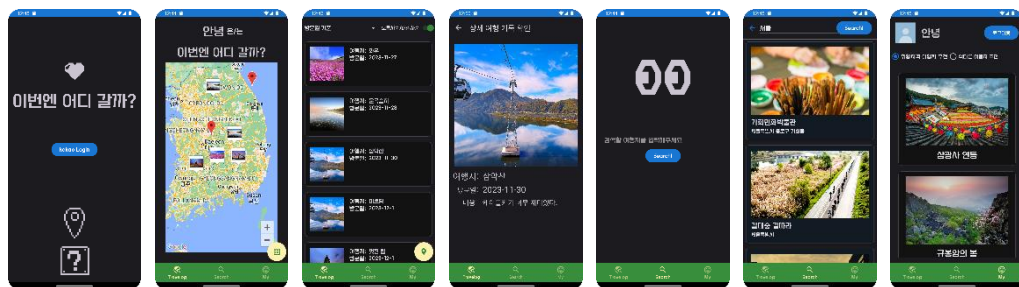
<div data-bbox="322 392 529 817" data-label="Image"> </div> <p data-bbox="368 835 489 869">[LOG_04]</p>	<ul style="list-style-type: none"> <li>- [LOG_03]에서 선택된 아이템의 상세 내용을 확인할 수 있는 페이지다.</li> <li>- 이미지는 Glide 를 사용하고 다른 정보들은 data class 를 통해 사용한다.</li> <li>- '내용'은 긴 경우, 전부 확인할 수 있게 scrollView 를 적용한다.</li> <li>- findNavController()를 이용하여 뒤로가기 버튼을 구현하였고, 클릭 시 [LOG_03]으로 이동한다.</li> </ul>
SEARCH 페이지	상세설명
<div data-bbox="207 1276 646 1706" data-label="Image"> </div> <p data-bbox="343 1724 510 1758">[SEARCH_01]</p>	<ul style="list-style-type: none"> <li>- 사용자가 여행 정보를 검색할 수 있는 페이지다.</li> <li>- 검색어가 입력될 EditText 를 autoCompleteTextView 와 sharedPreferences 이용하여, 한글자만 같아도 이전에 검색한 검색어들 중 자동완성 제안을 받을 수 있다.</li> <li>- 'Search! 버튼을 클릭이나 키보드의 '검색'이 눌렸을 때, [SEARCH_02]로 이동하게 되고,</li> </ul>

	<p>AutoCompleteTextView 에 입력되어 있는</p> <p>Text 가 sharedPreferences 에 저장되게 된다.</p>
 <p>[SEARCH_02]</p>	<ul style="list-style-type: none"> <li>- [SEARCH_01]에서의 action 을 통해 전달 받은 args 인 검색어를 retrofit 을 이용하여 API 호출하고 응답받는다.</li> <li>- 검색 결과를 RecyclerView 와 Adapter 를 적용하여 리스트 형식으로 볼 수 있는 페이지다.</li> <li>- 각 item 들을 클릭하면 dialog 를 통해 WebView 를 보여준다. WebView 는 해당 아이템의 여행지 이름을 네이버에 검색한 페이지다.</li> <li>- [SEARCH_01]과 같이 상단에 AutoCompleteTextView 가 있어서, 해당 페이지에서 재검색도 가능하다.</li> </ul>

	<ul style="list-style-type: none"> <li>- findNavController()를 이용하여 뒤로가기 버튼을 구현하였고, 클릭 시 [SEARCH_01]로 이동한다</li> </ul>
RECOMMAND 페이지	상세설명
 <p>[RECOMMAND_01]</p>	<ul style="list-style-type: none"> <li>- 페이지의 상단에 사용자 정보인 닉네임과 프로필 사진을 Glide 를 통해 보여준다.  그리고 '로그아웃' 버튼을 클릭 시, 카카오계정과 Firebase 가 로그아웃 되고 [LOGIN_01]로 이동한다.</li> <li>- 추천 여행지를 RecyclerView 와 Adapter 를 적용하여 리스트 형식으로 볼 수 있는 페이지이다.</li> <li>- 아이템의 위치를 추적하여, 아이템 클릭 시 숨겨져 있던 여행지의 장소정보를 확인할 수 있고 재 클릭 시 다시 아이템의 원래 사이즈로 복구된다.</li> <li>- 추천 여행지는 사용자의 Firestore 에 기록된 여행지 카테고리 빈도를 기반으로 선택된다.</li> </ul>

	<ul style="list-style-type: none"> <li>- 업로드 되어 있는 기록들을 '카테고리, 작성 시간'을 확인하며 순회한다. 작성 시간을 확인하는 이유는 동일한 빈도의 카테고리들이 발생한다면, 가장 최근에 방문한 카테고리를 또는 가장 옛날에 방문한 카테고리를 우선순위에 둔다.</li> <li>- 카테고리의 빈도는 <code>maxWithOrNull</code>, <code>minWithOrNull</code> 을 적용하여 찾을 수 있다.</li> <li>- 결정된 추천 여행지 카테고리에 맞게 TourAPI 에 호출하고 응답받은 여행지를 랜덤하게 5 개의 아이템으로 사용자에게 추천한다.</li> <li>- 각 추천 카테고리의 라디오버튼을 재 클릭 시, 추천을 새롭게 받을 수 있다.</li> </ul>
--	---

## DARK THEME



## 5. 향후계획

- 이미지에 사용한 Glide 라이브러리의 비동기 보완하여 이미지 로드 효율화
- Kakao API 를 통한 여행지 정보 공유 기능 추가
- Material 를 이용한 UI/UX 개선

## 6. 참고문헌

- Dmitry Jemerov, 'Kotlin in Action'
- 안드로이드 개발 문서

<https://developer.android.com/guide>

- Firebase

<https://firebase.google.com/docs/android>

- 카카오 개발 문서

<https://developers.kakao.com/docs/latest/ko/kakaologin/common>

- Tour API

<https://api.visitkorea.or.kr/>

- Material

<https://m2.material.io/components?platform=android>

- SNS, 커뮤니티에서의 언급량 표

<https://datalab.visitkorea.or.kr/datalab/portal/loc/getAreaDataForm.do>

- SAA 아키텍처 설명

<https://heegs-develop.tistory.com/12>