

# 컴퓨터과학 종합설계: 최종보고서

기술개발 과 제	GoHome: 심야 귀가 서비스					
과제팀 이름	void* capstone;			지도교수	정형구	
개 발 기 간	2020년 3월 ~ 2020년 6월 (총 4개월)					
개발소요비용	총 액	600 (천원)		학교부담금	600 천원	
				과제팀부담금	0 천원	
과 제 팀 구 성 원	이름	강민성	김건호	이상엽	최연웅	홍찬표
	사진					
	학번	2015540001	2015920003	2015920029	2015540041	2015920063
	연락처	010-8613-3289	010-6265-5436	010-3883-0489	010-6556-2256	010-8565-6245

창의공학설계프로젝트 과제를 성실히 수행하고자 과제 제안서를 제출합니다.

2020년 6월 25일

과제 수행자1 : 강민성 (인)  
 과제 수행자2 : 김건호 (인)  
 과제 수행자3 : 이상엽 (인)  
 과제 수행자4 : 최연웅 (인)  
 과제 수행자5 : 홍찬표 (인)  
 지도교수 : 정형구 (인)

서울시립대학교 컴퓨터과학부 귀중

# 1. 서론

## 1.1 개발 과제의 개요

### 가. 개발 과제 요약

GoHome은 심야 귀가 서비스로, 사용자에게 귀가경로 안내, 위치 공유 등의 서비스를 제공한다. 2020년 5월 현재, Google Maps, 네이버 지도, 카카오맵, 티맵 등 시중의 인기 지도 서비스는 따릉이와 연계되지 않으며 도보 경로를 제공하지 못하는 것도 있다. GoHome은 이 문제를 해결하고, 나아가 AR 도보 안내를 구현하여 길찾기 서비스에서 AR의 활용성을 확인한다.

### 나. 개발 과제의 배경 및 효과

#### ◇ 배경

시중에 다양한 지도 어플리케이션이 존재하지만, 심야 귀가 인구를 집중 타겟으로 한 것은 찾아보기 어렵다. 또한 심야 귀가자는 주로 택시를 이용하여 비용 부담이 크다. 귀가 시 N버스(서울시 심야 버스)를 이용하는 방법이 있지만, 배차 간격이 길고 노선 수가 적으며 일부 정류장에만 정차하는 한계로 인해 이용하기 쉽지 않다. 이러한 상황을 해결하기 위해, N버스, 따릉이(서울시 공공자전거), 도보를 모두 고려한 귀가 경로를 제공하는 서비스가 필요하다.

#### ◇ 효과

GoHome은 귀가자의 현 위치에서 자택까지의 적절한 경로를 탐색하여 제공한다. 또한 도보 경로에서는 AR(Augmented Reality)을 활용한 길 안내를 제공하여 보다 편하고 효과적인 경로 안내를 제공한다. 심야 귀가자의 안심 귀가를 위해 가족 및 지인에게 현 위치를 실시간 공유할 수 있는 서비스 또한 제공한다. 마지막으로 지하철 및 버스의 막차 시간을 볼 수 있도록 하여 사용자의 귀가 편의를 제공한다. 사용자들은 심야 시간에 편하고 빠르게 귀가할 수 있다. 또한 따릉이와 N버스의 접근성을 높여 심야 시간 대중교통 이용자가 증가할 것이다.

## 다. 개발 과제의 목표와 내용

### ◇ 대중교통 및 따릉이를 이용한 경로 탐색

- 사용자가 빠르고 편하게 집에 가는 경로를 찾을 수 있도록 한다.
- 대중교통은 지하철과 버스, 야간 버스를 포함하고, 따릉이와 연계한 경로 탐색을 제공한다.
- 지하철과 버스가 끊긴 시간엔 N버스(서울시 심야 버스), 따릉이, 도보를 이용해 귀가할 수 있도록 경로를 제공한다.
- 따릉이 대여소에 잔여 자전거가 있는지 실시간으로 파악하여 이용 가능한 최적 경로를 찾는다.

### ◇ 막차 시간 알리기

- 지하철과 버스 시간표를 활용하여 막차 시간을 계산하여 알려준다.
- 사용자가 대중교통 막차 시간을 놓치지 않도록 몇 분 전에 알림을 보낸다.

### ◇ AR 길 찾기

- 사용자가 밤 중에 길을 잃지 않게 AR로 도보 경로 안내를 돕는다.
- 길 찾기 결과와 사용자의 현재 위치, 이동 방향을 이용해 AR 화면에 이동할 방향을 표시한다.
- AR 화면과 지도를 함께 표시하여 사용자가 현 위치를 파악하기 쉽도록 한다.

### ◇ 빠른 길 찾기

- 앱을 실행하는 즉시 사용자의 집으로 가는 가장 빠른 경로를 알려준다.

## 라. 개발과제의 기술적 기대효과

### ◇ Micro Service Architecture를 통한 효율적인 개발 수행

Micro Service Architecture를 통해 연관성 낮은 서비스를 서로 분리함으로써 테스트 시간을 단축하고 유연하게 개발할 수 있다. 본 프로그램 개발에서는 로그인 서버를 Django, 로직 서버를 Node.js로 구분하여 효율적인 구현을 실현한다.

### ◇ AR(Augmented Reality)의 지도 분야 활용 가능성 검증

AR은 다양한 분야에서의 활용 가능성을 가진 신기술이다. 이러한 AR이 실제로 지도 분야에서 효율적으로 활용될 수 있는지 검증하고, AR에 대한 경험 및 미래의 증강 현실 사회에 대한 대비를 수행한다.

### ◇ 경로 탐색 알고리즘의 실제 적용 및 효율성 재고

따릉이, N Bus를 포함한 경로를 효율적으로 탐색하는 경로 탐색 알고리즘을 활용하여 사용자가 최단 시간으로 원하는 목표 지점에 도달할 수 있도록 구현한다. 이를 통해서 경로 탐색 알고리즘이 실제 길을 찾는 과정에서도 성공적으로 수행되는지 관찰하고, 효율성을 검증한다.

## 마. 개발과제의 경제적 및 사회적 파급효과

### ◇ N버스, 따릉이를 포함한 경로 탐색을 통한 심야 귀가 시간 단축

기존의 심야 귀가 인원들은 N버스와 택시를 통해 귀가하였다. 하지만 이들은 각각 정거장의 부족한 개수, 긴 배차 간격과 부담스러운 요금으로 귀가 인원들에게 부담으로 다가갔다. 따라서 따릉이와 N버스를 결합한 경로를 제공함으로써 사용자의 귀가 시간 단축을 기대할 수 있다.

### ◇ 야간 따릉이 사용량 증가

서울시 공공자전거 서비스 따릉이는 연도를 거듭할수록 이용량이 증가하고 있다. 하지만 이용량이 출퇴근 시간에 많이 몰려 심야 이용량은 상대적으로 낮다. 심야 귀가 시 따릉이를 적절하게 이용하여 사용량 증가를 기대할 수 있다.

### ◇ 따릉이의 대중교통화

주간에는 버스 정류장의 수와 버스 노선 수가 많기 때문에, 따릉이와 대중교통을 연계하여 이동하는 경우가 별로 없다. 하지만 야간에는 따릉이를 이용하여 이동 시간을 크게 줄일 수 있다. 따릉이와 대중교통을 연계하여 따릉이의 인식을 대중교통으로 전환하고, 이를 통해 환승 서비스 또한 기대할 수 있다.

## 1.2 관련 기술의 현황

### 가. State of art

#### ◇ AR(Augmented Reality)

가상현실(VR)의 한 분야로 실제로 존재하는 환경에 가상의 사물이나 정보를 합성하여 마치 원래의 환경에 존재하는 사물처럼 보이도록 하는 컴퓨터 그래픽 기법이다. 국내 정책에서 2017년 '13대 혁신성장동력 추진계획'에서 선정되었으며, VR과 함께 ICT 사업에서 시장규모가 크게 증가할 것으로 예측되는 분야이다. AR을 경로 안내에 적용한 사례로는 Google Maps의 Live View가 있으며, 국내에서는 현재 네이버가 실내 길찾기 서비스를 개발 중이다. 현재 우리나라에서 AR을 통해 경로를 안내하는 서비스는 없다. AR을 눈에 직접 보여줄 안정된 장치, 높은 수준의 디지털 객체, 응용 프로그램, AR 생태계 등, 많은 기술발전이 필요하다. 현재 높은 수준의 안정된 장치의 보급과 5G 등 통신기술의 발전으로 앞으로 더 많은 AR 애플리케이션이 등장할 것으로 예상된다.

#### ◇ Micro-Service Architecture(MSA)

마이크로서비스(microservice)는 애플리케이션을 느슨히 결합된 서비스의 모임으로 구조화하는 서비스 지향 아키텍처(SOA) 스타일의 일종인 소프트웨어 개발 기법이다. 마이크로서비스 아키텍처에서 서비스들은 섬세하고 프로토콜은 가볍다. 애플리케이션을 조그마한 여러 서비스로 분해하여 모듈성을 개선하고 애플리케이션의 이해, 개발, 테스트를 더 쉽게 한다. 규모가 작은 자율적인 팀들이 팀별 서비스를 독립적으로 개발, 전개, 규모 확장을 할 수 있게 함으로써 병렬로 개발할 수 있게 한다. 또, 지속적인 리팩토링을 통해 개개의 서비스 아키텍처가 하나로 병합될 수 있게 한다.

#### ◇ REST API

HTTP 통신에서 어떤 자원에 대한 CRUD 요청을 Resource와 Method로 표현하여 특정한 형태로 전달하는 방식이다. 문서 없이 REST API 메시지를 읽는 것만으로도 메시지가 의도하는 바를 명확하게 파악할 수 있다. 또한, HTTP 인프라를 그대로 사용하기 때문에, REST API 사용을 위한 별도의 인프라 구축이 필요하지 않다.

#### ◇ 경로 탐색 알고리즘

출발점에서 목적지까지의 수많은 링크를 연결하여 여러 가지 경로를 구성하고, 편리하면서도 빠르게 목적지까지 도착할 수 있도록 계산한 최적의 경로를 안내하는 알고리즘을 구현한다. 기업마다 각각의 요소에 특정한 가중치를 두어 알고리즘을 구성하기 때문에, 같은 출발지/목적지에 대해서도 다른 경로로 안내가 될 수 있다. 각 기업들의 알고리즘은 대외비로 현재 알 수는 없으나, 경로 탐색에 AI와 실시간 교통정보를 반영하여 사용하는 것으로 알려져 있다.

## 나. 기술 로드맵

GoHome						
Time Span	3월	4월	5월	6월	최종 목표	
월별 목표	계획 수립	프로그램 개발	프로토타입 완성	테스트/Feedback	제품 완성	
핵심 기술	경로 탐색	API Data 요청				N버스, 따릉이, 도보를 포함한 경로 탐색
			따릉이/도보 경로 탐색 구현			
				N Bus 포함 경로 탐색 구현		
		효율적인 경로 탐색 구현				
	AR 길 안내	AR 이미지 구현				AR을 통한 직관적인 길 안내
			원하는 방향 및 위치에 AR 구현			
		효율적인 AR 이미지 구현				
	MSA	각 서버 구현				Django, Node.js로 로그인/로직 서버 구현
			서버 간 통신 구현			
				서버-클라이언트 통신 구현		

## 다. 특허조사

### ◇ 고령자를 위한 길찾기 어플리케이션 제공방법

- 출원번호 : 1020180004077

본 발명은 고령자의 신체에 착용하는 장치의 GPS 모듈을 통해, 고령자의 위치를 실시간으로 파악하여, 생활패턴을 파악한 건강 맞춤형 관리에 사용되거나, 낙상 등에 의한 신체 구조에 사용되며, 안내 정보 제공을 통해 노인의 고독사를 방지한다.

### ◇ AR을 이용한 내비게이션 신발(AR Navigate shoes)

- 출원번호 : 2020170002011

본 발명은 AR을 이용한 내비게이션 신발에 관한 것이다. 내비게이션 신발을 신고 길을 걷다가 길을 잃거나 되돌아갈 때 AR을 이용해 발자국을 보여준다. 장소의 모습을 사진으로 찍어 저장함으로써 이후 다시 찾아오기 위한 내비게이션 기능도 포함한다.

## 라. 특허전략

### ◇ AR 길찾기

현재 AR 시장은 정보를 획득하는 대상체(현실의 물체)와 이에 결합되는 가상데이터(가상의 대상)의 매칭 관계에 따라 다양한 서비스가 등장하고 있다. 기존 서비스들과의 현실대상 또는 가상대상 차이에 의해 해당 새로운 서비스 구현에 특화된 기술들이 개발되면, 이 또한 특허의 대상이 될 수 있다. 예를 들어 운동 종목에 대한 가상스포츠 서비스를 제공하고자 하면, 운동 종목 간의 특성 차이에 의해 해당 운동 종목에 특화되는 기술들이 산출되어 특허로 등록된다. 이에 대한 컨셉은 오래전부터 등장하였고 관련 기술들이 개발된지 오래되었기 때문에, 사업 또는 아이템 전체를 포괄하는 넓은 특허 확보는 어렵다.

## 1.3 관련 시장에 대한 분석

### 가. 경쟁제품 조사 비교

#### ◇ 막차

- 버스 및 지하철 등 대중교통의 막차 시간을 제공하고, 따릉이와 택시 등을 이용했을 때의 소요 시간을 비교하여 안내한다.
- 사용자가 이동 수단별 경로를 선택할 수 있도록 한다.
- 통금 시간에 맞는 막차를 알림으로 띄워준다.
- 목적지 선택이 일부 장소에만 국한된다.
- 이동 수단 각각(지하철, 버스, 지하철+버스, 택시, 따릉이)의 경로는 제공하나, 모든 이동 수단을 합쳐 최적 경로를 제공하지는 않는다.



그림 6 막차

#### ◇ 네이버 지도, 카카오맵

- 지하철, 버스, N버스, 자동차, 자전거 및 도보 경로를 제공한다.
- 따릉이 길 찾기는 제공하지 않는다.
- 사용자의 집과 회사를 등록하고 길 찾기에 이용할 수 있다.

#### ◇ Google Maps

- 대중교통 경로를 제공한다.
- 따릉이 길 찾기, 도보 경로 찾기 등은 제공하지 않는다.
- 사용자의 집과 회사를 등록하고 길 찾기에 이용할 수 있다.

#### ◇ 서울시 안심이

- 서울시에서 제공하는 안심 귀가 애플리케이션이다.
- 스마트폰 위치 정보와 서울시 정보 인프라를 이용하여 각종 범죄 위협으로부터 시민 보호를 목적으로 한다.
- 모니터링 및 긴급 신고 등의 기능을 제공한다.
- 길 안내는 제공하고 있지 않다.
- 복잡한 UI를 가지고 있다.

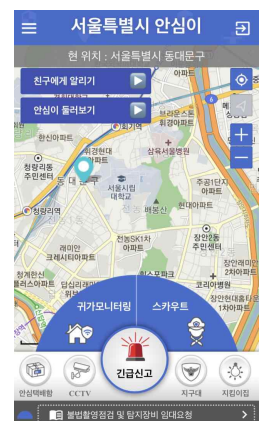


그림 7 서울시 안심이



## 나. 마케팅 전략

### ◇ AR을 이용한 길 찾기 강조

- 도보 이동 시 AR로 가야 할 방향을 알려줌으로써 단순히 지도만 보는 것보다 편리하게 길을 찾을 수 있음을 강조한다.
- 현재 AR을 이용한 상용 길 찾기 서비스가 국내에는 보급되지 않았기 때문에, 타 서비스 대비 경쟁력을 확보할 수 있다.

### ◇ 따릉이를 활용한 경로 탐색

- 현재 따릉이를 포함하여 길 찾기를 제공하는 애플리케이션은 거의 없다. 한국에서 가장 점유율이 높은 4개 지도 애플리케이션(네이버 지도, T map, Google Maps, 카카오맵) 역시 모두 따릉이를 사용한 길 찾기 서비스를 제공하지 않는다.
- 심야 시간 귀가하는 사용자에게 GoHome을 이용하면 빠르게 귀가할 수 있음을 어필한다.

### ◇ 직관적인 UI/UX 강조

- 현재 귀가를 중심으로 하는 애플리케이션은 '막차'가 있다. '막차'는 이동 수단별 귀가 경로를 제공하지만, 모든 이동 수단을 통합한 최적 경로를 제공하고 있지는 않다.
- GoHome은 N버스, 따릉이 등 모든 이동 수단을 통합한 최적 경로를 제공함으로써, 사용자에게 더 빠르고 편리한 귀가 경로를 제공함을 어필한다.

### ◇ SWOT 분석

#### 1. Strengths (강점)

- 가. 따릉이와 N버스를 이용하여 최적의 경로 안내를 제공한다.
- 나. AR 기술을 적용하여 시각적인 경로 안내 서비스를 제공한다.

#### 2. Weaknesses (약점)

- 다. 따릉이 이용 시 추가 비용이 발생할 수 있다.
- 라. 사용 가능 지역이 서울로 국한된다.

#### 3. Opportunities (기회)

- 마. 따릉이, N버스를 모두 포함하는 경로 안내 애플리케이션이 없다.
- 바. N버스의 긴 배차 간격과 정류장 간격 때문에 심야에 어쩔 수 없이 택시를 이용하는 사용자가 많다.

#### 4. Threats (위협)

- 사. 주요 지도 애플리케이션의 점유율이 높아 출시 후 신규 사용자 유치가 어렵다.

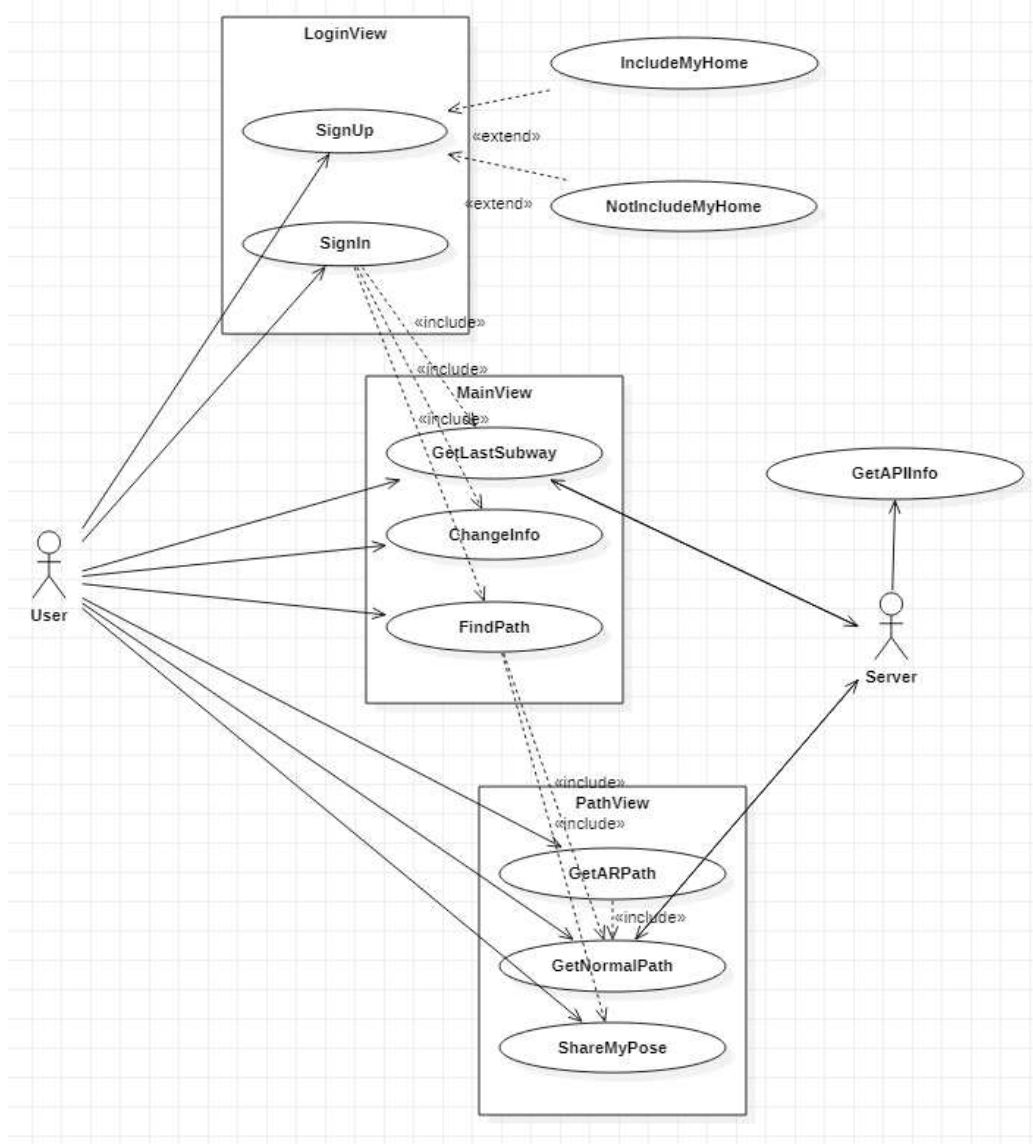
## 2. 설계

### 2.1 사용자 요구사항

#### 가. 요구사항 목록

번호	요 구 사 항	D or W
1	현 위치에서 목적지까지 최적 경로를 얻는다.	D
2	AR(Augmented Reality)을 통한 경로 안내를 사용한다.	D
3	사용자 정보를 변경한다.	D
4	지인과 위치공유 기능을 사용한다.	D
5	버스, 지하철 등 대중교통 막차를 안내한다.	W
6	현재 위치를 지도에 표시한다.	W
7	주변 숙박업소를 검색한다.	W
8	맞춤형 안내 서비스를 사용한다.	W
9	Client Application의 로딩시간은 2초를 넘기지 않는다.	D
10	Client Application의 UI/UX가 통일감을 준다.	W
11	Client Application의 색상이 통일감을 준다.	W

## 나. 유즈케이스



◇ 액터 목록

액터 명	구분	설명
User	사용자	회원가입이 되어 있는 사용자
Server	관리자	회원 정보, 경로 정보 등을 관리하는 서버

◇ 유즈케이스 설명

이름	SignUp	관련 액터	User
설 명	User가 신규 회원가입을 하는 유즈 케이스		
사건흐름	기본 흐름	1. User가 회원가입 버튼을 누른다. 2. 사용자 정보를 입력한 뒤 회원가입을 완료한다.	
	대안 흐름	2.a 사용자의 집 정보를 포함하여 가입한다. [IncludeMyHome] 2.b 사용자의 집 정보를 포함하지 않고 가입한다. [NotIncludeMyHome]	
	예외 흐름	1.a 이미 존재하는 ID, 비밀번호일 경우 오류 메시지를 출력한다. 1.b 사용자 정보 형식이 잘못된 경우 오류 메시지를 출력한다.	
조건	사전 조건	없음	
	사후 조건	없음	
참고사항			

이름	IncludeMyHome	관련 액터	User
설 명	User가 본인의 집 정보를 포함한 신규 회원가입을 하는 유즈 케이스		
사건 흐름	기본 흐름	1. User가 회원가입 버튼을 누른다. 2. 사용자 정보를 입력한 뒤 회원가입을 완료한다.	
	대안 흐름	없음	
	예외 흐름	1.a 이미 존재하는 ID, 비밀번호일 경우 오류 메시지를 출력한다. 1.b 사용자 정보 형식이 잘못된 경우 오류 메시지를 출력한다.	
조건	사전 조건	없음	
	사후 조건	없음	
참고사항			

이름	NotIncludeMyHome	관련 액터	User
설 명	User가 본인의 집 정보를 포함하지 않은 신규 회원가입을 하는 유즈 케이스		
사건 흐름	기본 흐름	1. User가 회원가입 버튼을 누른다. 2. 사용자 정보를 입력한 뒤 회원가입을 완료한다.	
	대안 흐름	없음	
	예외 흐름	1.a 이미 존재하는 ID, 비밀번호일 경우 오류 메시지를 출력한다. 1.b 사용자 정보 형식이 잘못된 경우 오류 메시지를 출력한다.	
조건	사전 조건	없음	
	사후 조건	없음	
참고사항			

이름	GetLastSubway		관련 액터	User, Server
설 명	User가 현재 대중교통의 막차 시간을 전달받는다.			
사건 흐름	기본 흐름	1. User가 막차 시간을 요청한다. 2. Server가 버스, 지하철 API 정보를 받아온다. [GetAPIInfo] 3.a User에게 막차 시간을 계산하여 전달한다. 3.b 대중교통이 종료되었을 시 막차가 없음을 전달한다.		
	대안 흐름	없음		
	예외 흐름	없음		
조건	사전 조건	SignIn		
	사후 조건	없음		
참고사항	회원가입을 통해 User의 로그인 정보가 있어야 가능함			

이름	ChangeInfo	관련 액터	User
설 명	User가 사용자 정보를 갱신한다.		
사건흐름	기본 흐름	1. User가 사용자 정보 변경 버튼을 누른다. 2. 사용자 정보를 변경한다. 3. 변경 버튼을 눌러 사용자 정보를 변경한다.	
	대안 흐름	없음	
	예외 흐름	2.a 잘못된 아이디, 비밀번호 정보를 입력 시 오류 메시지를 출력한다. 3.a 잘못된 사용자 정보를 입력할 시 오류 메시지를 출력한다.	
조건	사전 조건	SignIn	
	사후 조건	없음	
참고사항			

이름	FindPath	관련 액터	User
설 명	User가 경로 탐색을 요청한다.		
사건흐름	기본 흐름	1.a User가 원하는 도착지 정보를 입력한다. 1.b 사용자 정보에 집이 입력되어 있을 경우 집 정보가 입력된다. 2. 경로 탐색을 요청한다.	
	대안 흐름	없음	
	예외 흐름	1.a.a 도착지 정보가 잘못되었을 경우 오류 메시지를 출력한다.	
조건	사전 조건	SignIn	
	사후 조건	GetNormalPath	
참고사항			

이름	GetNormalPath	관련 액터	User, Server
설 명	User의 현재 위치에서 도착지까지의 경로를 반환한다.		
사건흐름	기본 흐름	1. 유저로부터 현재 위치 및 도착지 경로를 전달받는다. 2. Server가 API를 통해 필요한 데이터들을 받아온다. [GetAPIInfo] 3. 경로를 계산하여 표시한다.	
	대안 흐름	없음	
	예외 흐름	3. 경로 계산에 실패하였을 경우 오류 메시지를 출력한다.	
조건	사전 조건	SignIn, FindPath, GetAPIInfo	
	사후 조건	없음	
참고사항			

이름	ShareMyPose	관련 액터	User
설 명	User의 위치를 지인에게 공유한다.		
사건 흐름	기본 흐름	1. 유저가 위치 공유 버튼을 클릭한다. 2. 위치를 URL을 통해 지인에게 공유한다.	
	대안 흐름	없음	
	예외 흐름	없음	
조건	사전 조건	SignIn	
	사후 조건	없음	
참고사항			

이름	GetARPath	관련 액터	User
설 명	User가 AR 이미지를 통한 경로 탐색을 사용한다.		
사건 흐름	기본 흐름	1. 유저가 AR 버튼을 클릭한다. 2. 현재 경로를 토대로 AR 경로 탐색을 제공한다.	
	대안 흐름	없음	
	예외 흐름	2.a AR 관련 오류가 발생하였을 경우 오류 메시지를 출력한다.	
조건	사전 조건	SignIn, FindPath, GetNormalPath	
	사후 조건	없음	
참고사항			



이름	GetAPIInfo	관련 액터	Server
설 명	Server가 버스, 지하철, 따릉이, T map API 정보를 받아온다.		
사건 흐름	기본 흐름	1. Server가 Open API에 정보를 요청한다. 2. 버스, 지하철, 따릉이, Tmap API 정보를 받아온다.	
	대안 흐름	2.a API 정보를 받아오지 못할 경우 오류 메시지를 출력한다.	
	예외 흐름	없음	
조건	사전 조건	없음	
	사후 조건	없음	
참고사항			

## 2.2 사용자 요구사항 만족을 위한 기능 정의 및 기능별 정량목표

### ◇ Client Application와 Gateway Server 간 통신

응답 시간: 1초 미만

### ◇ Gateway Server와 Server 간 통신

응답 시간: 1초 미만

### ◇ 따릉이/N버스를 포함하는 경로 탐색

경로 탐색 시간: 5초 미만

경로 정확도(시간 오차): 경쟁 서비스 대비 110% 미만의 최적 경로 소요시간

### ◇ AR로 도보 경로 표시

정확도: 실제와  $\pm 10^\circ$  미만의 오차로 정확한 방향/위치 표현

### ◇ Client Application에서 페이지를 로드

시간: 각 1초 이내 (통신 시간 제외)

### ◇ 내 위치 공유

시간: 5초마다 위치 갱신

## 2.3 기능을 구현을 위한 세부기술 선택사항 (디자인)

### 가. 시스템 구성

#### ◇ Micro-Service Architecture

MSA(Micro-Service Architecture)는 큰 애플리케이션을 여러 개의 작은 애플리케이션으로 나눠 만드는 아키텍처이다. 개발, 빌드, 테스트 시간을 단축할 수 있으며 본 프로젝트에서처럼 여러 개의 서버를 서로 다른 언어, 프레임워크로 사용할 수 있다는 점에서 유연성이 굉장히 높다. 또한, scaling하기에도 용이하기 때문에 트래픽이 높은 서버에 적합하다.

#### ◇ MongoDB

MongoDB는 NoSQL의 일종으로 key-value 형태의 BSON으로 데이터를 저장한다. NoSQL은 'Not Only SQL'로서 기존에 RDBMS와 다르게 SQL을 활용하지 않는 다른 방식의 저장기술이다. 단순하고 대량의 데이터를 다루기 용이하기 때문에 빅데이터가 떠오르면서 함께 급부상하고 있다. 또한, 스키마가 존재하지 않기 때문에 필드의 추가 및 제거가 매우 쉬워져 개발의 진입장벽이 낮다는 장점이 있다.

### 나. 통신 및 인증

#### ◇ RESTful API

REST란 'Representational State Transfer'의 약자로, 자원을 URI로 표시하여 해당 자원의 상태를 주고 받는 것을 의미한다. REST API는 설계규칙이 존재하는데, 이를 통해 언어나 플랫폼에 종속되지 않는 인터페이스를 만들 수 있다. 설계규칙은 공식기관에서 발표하는 방식이 아닌 개발자들이 비공식적으로 의견을 제시하기 때문에 명확한 정의가 존재하지 않는다. 하지만 일반적인 규칙은 다음과 같다.

1. URI는 자원을 표현하는 데 중점을 두기 때문에, 동사보다는 명사를 사용한다.
2. 자원에 대한 행위는 HTTP METHOD로 표현한다.
3. 슬래시(/)는 계층 관계를 나타내는 데 사용되고 마지막에는 사용하지 않는다.
4. 대문자와 언더바(\_)를 사용하지 않고 소문자와 하이픈(-)을 사용하여 나타낸다.

#### ◇ JWT

클라이언트의 인증을 위해 token 인증 방식을 사용하였다. 그중에서도 JWT를 사용하였는데 payload에 자주 참조되지만 보안 문제가 없는 데이터를 저장함으로써 서버와 DB의 트래픽을 낮추었다. JWT를 비롯한 토큰 인증 방식은 악성 클라이언트로부터 토큰 탈취 위험이 있으므로 access token과 refresh token을 사용하여 보안을 강화했다.

#### ◇ Retrofit2

Android 클라이언트에서 서버와 HTTP 통신을 위해 Retrofit2 라이브러리를 사용하였다. Retrofit2는 type-safe HTTP 클라이언트이다. REST API를 위한 자신만의 코드를 작성하는 것은 시간이 오래 걸리는 작업이지만, Retrofit2를 이용하면 connection, caching, retry, threading, response parsing, error handling 등 많은 기능을 훨씬 더 빠르고 안전하게 구현할 수 있다.

## 다. 앱 개발 및 AR

#### ◇ Fragment Layout

Client Application 개발에서 Intent를 사용하여 Activity를 전환하면 끊김 현상이 발생하였다. 더 부드러운 화면 전환을 위해 Fragment Layout을 사용하였다. Fragment를 상속하여 각 View를 구현했고, 한 Activity 내에서 화면 전환할 수 있도록 하였다.

#### ◇ ARCore 및 Sceneform

ARCore는 증강 현실 애플리케이션을 빌드 할 수 있도록 Google에서 개발한 소프트웨어 개발 키트이다. Sceneform은 ARCore 내에서 지원하는 라이브러리이며, 3D UI에 대한 개발을 지원한다. 안드로이드 스튜디오, 유니티, 언리얼 엔진 등에서 구현 가능하며 안드로이드 스튜디오 기반의 구현을 진행하였다.

#### ◇ .obj, .mtl, .sfa 파일

AR 이미지는 2D 또는 3D로 모델링 된 이미지를 사용자의 모바일 기기 화면에 투영한다. 본 애플리케이션에서는 3D 모델을 활용했으며, 다양한 3D 모델링 파일이 존재하는 라이브러리 사이트인 구글 Poly를 이용하거나, 윈도우 기본 제공 애플리케이션인 “3D 그림판”을 활용했다. 모델링 파일은 .obj, .mtl 확장자 파일로 이루어져있으며 .obj는 3D 모델의 정보를 담고 있는 파일이고, mtl은 3D모델의 재질 정보를 담고 있다. ARCore에서는 Sceneform 라이브러리를 통해 .obj 파일을 자체적으로 활용할 수 있는 .sfa 파일로 변환하여 사용한다.

#### ◇ Quaternion과 Euler Angles

AR 경로 탐색을 위해서는 사용자가 보는 방향 및 실제로 이동해야 하는 방향을 계산하여 AR 화살표를 알맞게 회전시켜 화면에 출력해야 한다. 3차원 물체의 회전에 대해서는 Euler Angles와 Quaternion의 개념이 있으며, ARCore에서는 Quaternion을 주로 사용한다. Euler Angles은 X, Y, Z 3개의 축으로 회전하는 직관적인 회전 개념이며, Quaternion은 (x,y,z,w) 의 4가지 원소를 가지며, x, y, z의 회전의 원점 벡터와 w의 회전 스칼라 값을 의미한다. Quaternion은 Euler Angles에서 나타나는 축이 겹치는 짐벌락 현상이 나타나지 않는 장점이 있다.

## 2.4 시스템 설계



### ◇ User

GoHome을 사용하는 사용자는 안드로이드 앱을 통해 서비스를 이용한다. 사용자는 요청에 대한 처리 결과를 확인한다.

### ◇ Front End

Android Studio와 ARCore를 사용해 작성된다. 사용자가 보는 화면을 생성하고, 발생하는 이벤트를 처리한다. 경로 안내와 함께 선택적으로 AR 안내를 제공한다. Back End 서버와 HTTP 통신을 통해 데이터를 주고받는다.

### ◇ Back End

상호 독립적인 기능구현을 위해 Micro-Service Architecture를 적용하여 User Management Server, Logic Server, API Gateway Server로 구성된다. User Management Server는 사용자 인증, 관리를 제공한다. Logic Server는 Open API 요청을 통해 데이터를 받아와 경로 탐색 요청을 처리한다. API Gateway Server는 클라이언트-서버 통신을 중개한다.

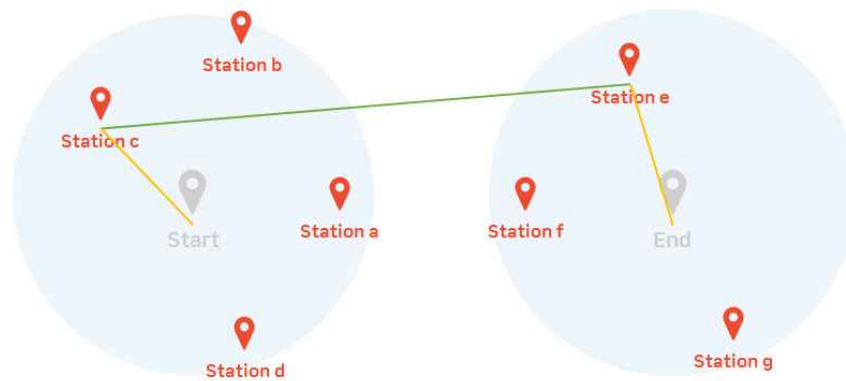
### ◇ DB

User Management Server는 MySQL을 통해 사용자 정보를 저장하고, Logic Server는 MongoDB를 사용해 자전거/버스 정보를 저장한다.

## 2.5 이론적 계산 및 시뮬레이션

### ◇ 따릉이를 포함하는 경로 탐색 알고리즘

출발지 주변의 따릉이 대여소와 도착지 주변의 따릉이 정류장을 1:1 선택하고, 직선 경로에 대한 소요 시간을 어렵한다. 이때, 대여소 간 소요시간은 캐시된 것을 사용할 수 있다.



어림한 소요 시간이 적은 순으로 실제 경로 및 소요 시간을 구한다. 이때 Tmap의 도보 경로 탐색 API를 사용한다. 실제 소요 시간은 어림한 시간보다 반드시 더 길다. 따라서 한 경로에 대한 실제 소요 시간을 얻을 때마다, 그보다 더 긴 어림한 소요 시간을 가진 경로는 API를 통해 실제 경로를 탐색하지 않는다.

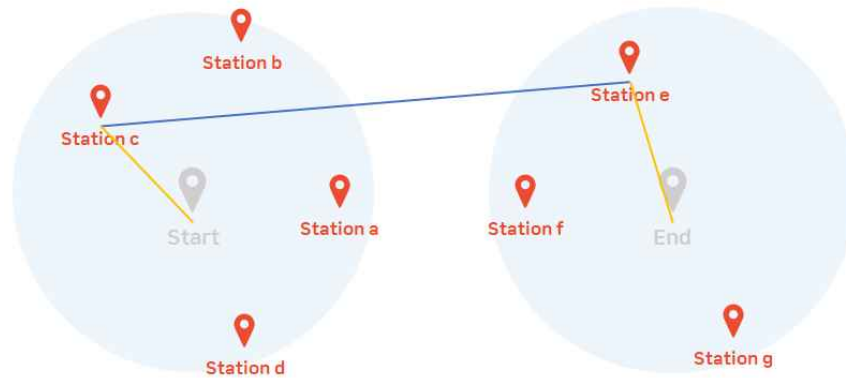
	Estimated Time	Real Time
	1500	2200
	1600	2100
	1800	2700
	1900	3300
	2200	
	2300	
⋮	⋮	

위 그림은 모든 경로를 어림한 소요 시간이 짧은 순으로 정렬한 것이다. 어림한 소요 시간이 2200, 2300인 경로는 그보다 짧은 시간의 실제 소요 시간을 가지는 경로가 존재하므로 탐색 대상에서 제외된다.

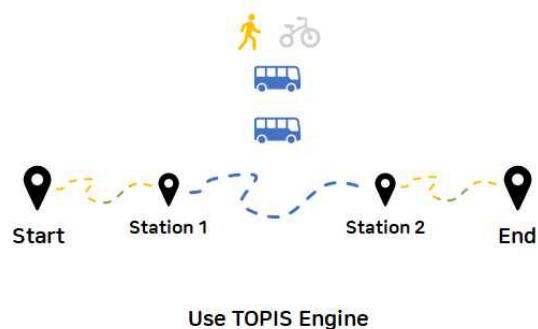
모든 경로에 대한 탐색이 끝나면, 실제 소요 시간이 가장 짧은 경로 몇 가지를 최종 반환한다. 실제 경로 탐색이 이루어진 모든 경우에 대해 대여소 간 소요 시간을 캐시한다.

#### ◇ N버스를 포함하는 경로 탐색 알고리즘

출발지 주변의 따릉이 정류장과 도착지 주변의 N버스 정류장을 1:1 선택하고, 직선 경로에 대한 소요 시간을 어렵한다. 이때, 정류장 간 소요시간은 캐시된 것을 사용할 수 있다.



어려한 소요 시간이 적은 순으로 실제 경로 및 소요 시간을 구한다. 이때 버스 정류장 간 구간은 TOPIS 엔진을 호출하여 구하고, 나머지 구간은 <따릉이를 포함하는 경로 탐색 알고리즘>을 사용한다. 실제 소요 시간은 어렵한 시간보다 반드시 더 길다. 따라서 한 경로에 대한 실제 소요 시간을 얻을 때마다, 그보다 더 긴 어렵한 소요 시간을 가진 경로는 API를 통해 실제 경로를 탐색하지 않는다.



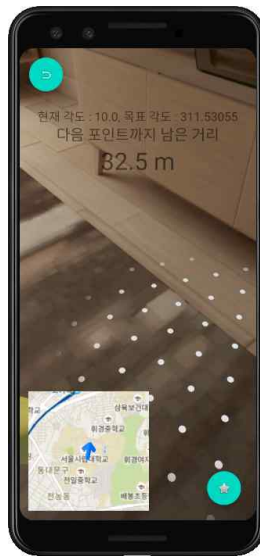
모든 경로에 대한 탐색이 끝나면, 실제 소요 시간이 가장 짧은 경로 몇 가지를 최종 반환한다. 실제 경로 탐색이 이루어진 모든 경우에 대해 정류장 간 소요 시간을 캐시한다.

#### ◇ AR 도보 안내의 이론적 과정 및 시뮬레이션

AR 도보 안내는 경로 탐색을 통해 도출된 (위도, 경도)로 이루어져 있는 경로 포인트 정보를 활용하여 사용자에게 AR 이미지를 출력한다. 그 과정은 다음과 같다.

##### 1) 평면 인식

3D 모델을 기기 내 화면에 출력하기 위해서는 모델을 위한 평면을 인식해야 한다. ARCore를 이용하여 모델을 출력할 수 있는 평면을 탐색한다. 화면의 색상을 통해 평면을 인식하는 특성으로 인해 흰색 평면은 잘 인식하지 못한다. [그림 1]과 같이 흰 점의 집합으로 평면을 인식하여 화면에 출력한다.



[그림 1] 화면 인식

##### 2) 노드 및 모델링 객체 생성

인식한 평면 위에 3D모델을 생성할 수 있는 노드를 새롭게 생성한다.

3D 모델링 파일을 Sceneform을 통해 기존 obj 모델링 파일을 활용할 수 있는 .sfa 및 .sfb 파일로 변환한 뒤 ModelRenderable 객체에 담는다.

##### 3) 노드 회전 및 AR 이미지 출력

노드를 사용자가 이동해야 하는 방향으로 회전시킨다. 회전 방법은 다음과 같다.

사용자가 이동해야 하는 방향을 계산하기 위해 방위각 개념을 이용하였다. 현재 위치 및 목표 위치의 위도, 경도 값을 받아와 라디안 값으로 변환한 후, sin 및 cos 및 acos 함수를 이용하여 라디안 거리를 계산하였다. 계산된 거리를 바탕으로 이를 라디안 각도로 변환한 뒤 다시 방위각으로 변환했다. 변환 공식은 다음과 같다.

$$\text{라디안변환 공식} : \frac{\text{위도(경도)} * \pi}{180}$$

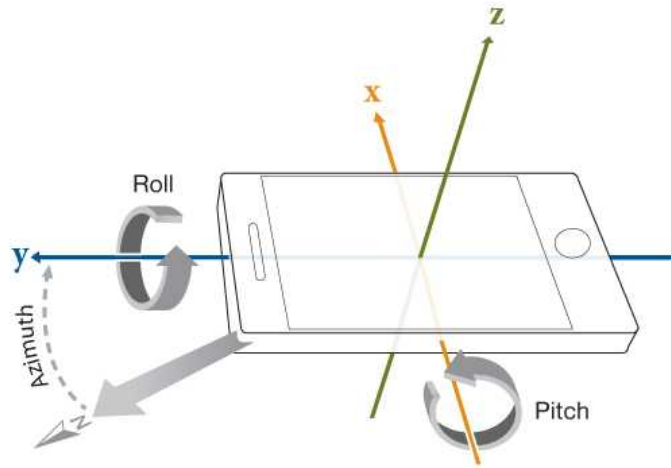
$$\begin{aligned} \text{라디안거리 공식} : & \text{acos}(\sin(\text{현재위도}) * \sin(\text{목표위도}) \\ & + \cos(\text{현재위도}) * \cos(\text{목표위도}) * \cos(\text{현재경도} - \text{목표경도})) \end{aligned}$$



$$\text{라디안각도 공식} : \arccos\left(\frac{\sin(\text{목표위도}) - \frac{\sin(\text{현재위도}) * \cos(\text{라디안거리})}{e + \cos(\text{현재위도}) * \sin(\text{라디안거리})}}{e}\right)$$

$$\text{라디안각도 - 각도 변환} : \frac{\text{라디안각도} * 180}{\pi}$$

사용자가 향하고 있는 방향 계산을 위해서 안드로이드 기기의 가속도 센서 및 자기장 센서를 이용했다. 두 센서를 통해 회전의 정보를 담고 있는 Azimuth, Pitch, Roll 값을 받아 온 뒤 Azimuth 값을 통해 평면에서 사용자가 향하는 방향을 계산했다.



두 값을 기준으로 이미지가 회전해야 하는 각도를 계산하였다. 먼저 현재 사용자가 향한 각도만큼 역으로 회전시켜 북쪽을 바라보게 한 뒤, 목표 방위각만큼 다시 정 방향으로 회전시키는 형식으로 이미지의 각도를 설정했다. 예를 들어 정북 방향이 0도라고 할 때 사용자가 현재 보는 방향이 270도이고 목표 방향이 45도라면, -270도를 회전시켜 0도로 정북 방향으로 만든 뒤 다시 +45도를 회전시켜 이미지가 목표 방향으로 회전할 수 있도록 했다. [그림 2]와 [그림 3]과 같이 북쪽 및 원하는 각도로 정상적으로 AR 이미지가 회전하고, 각도 오차는  $\pm 3^\circ$  정도로 크지 않게 나타남을 알 수 있다.



[그림 2] 정북방향



[그림 3] 원하는 방향

회전된 노드를 기준으로 모델링 객체를 통해 AR 이미지를 출력한다. 이 과정을 반복하며 지속적으로 AR 이미지를 갱신한다.

## 2.6 하드웨어 설계

(해당 없음)

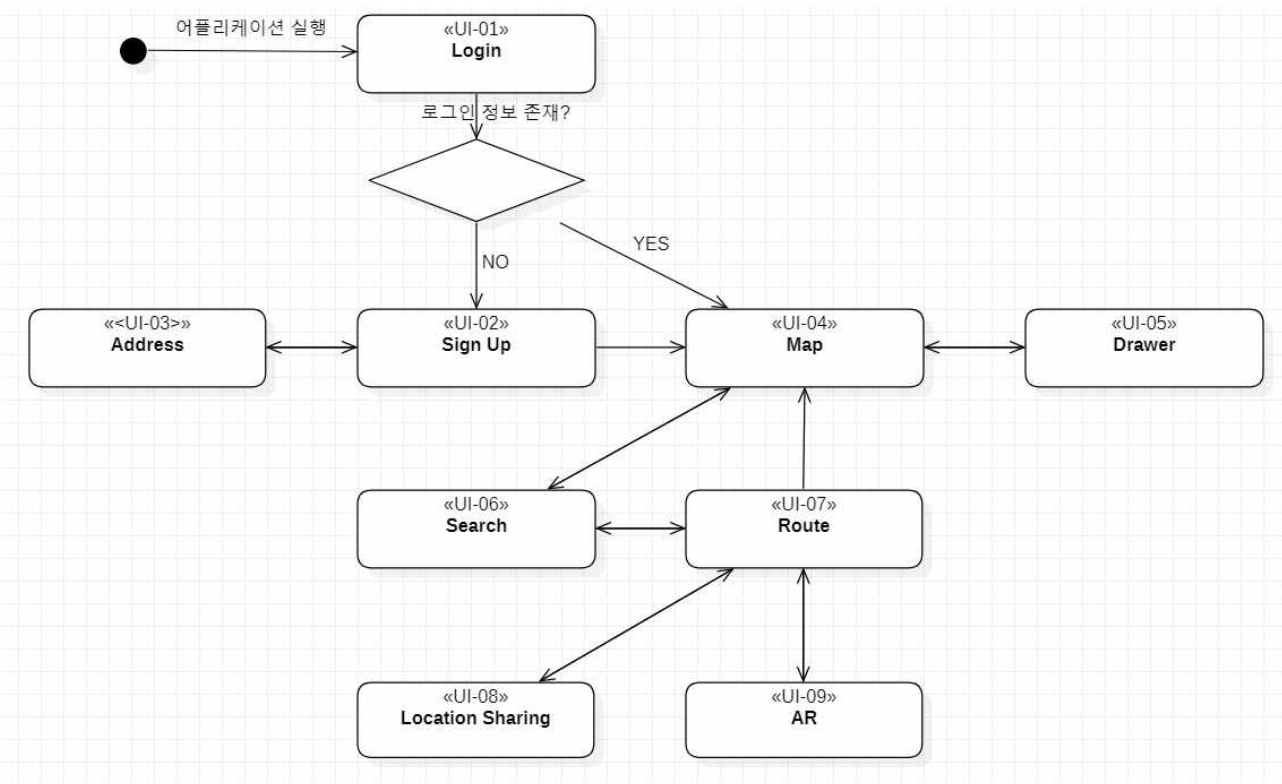
## 2.7 소프트웨어 설계

### ◇ Client Application Views

#### 1) 화면 목록

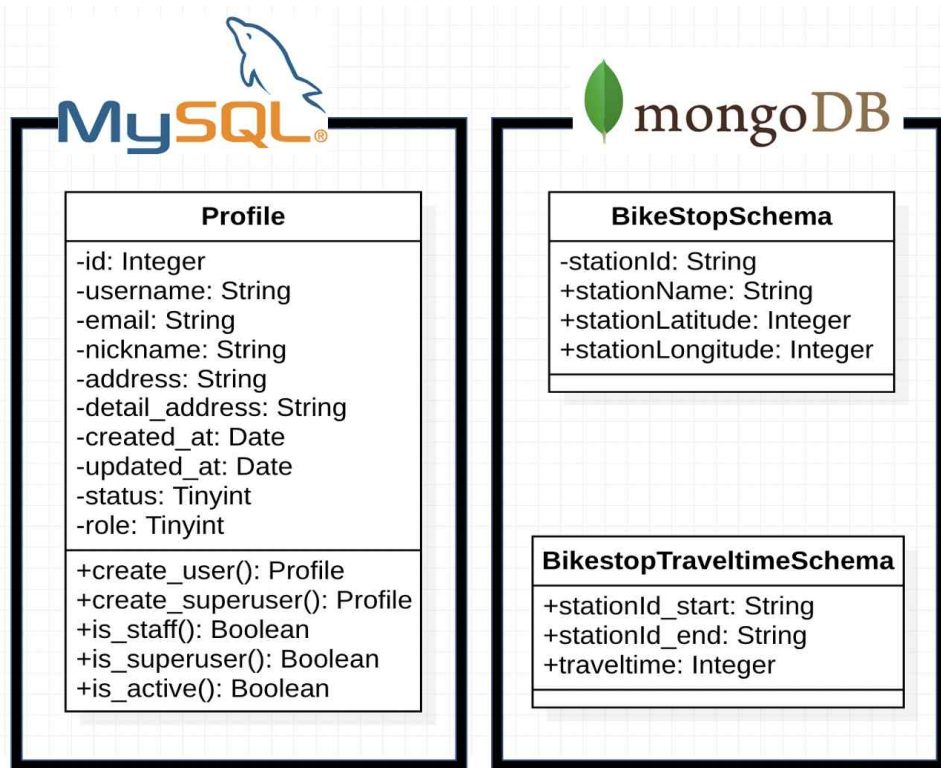
화면 이름	화면 설명	구성 요소
Login (UI-01)	로그인 화면	ID, password 입력 창
		Sign in, sign up 버튼
Sign Up (UI-02)	회원가입 화면	사용자 정보 입력 창
		Sign up 버튼
Address (UI-03)	주소 검색 화면	주소 입력 창
		검색 결과
Map (UI-04)	지도에 현재 위치 표시	지도 (T-map)
		Drawer 열기 버튼
		검색 창
		위치 공유 버튼
Drawer (UI-05)	사용자 정보, 막차 및 첫차 정보 표시	사용자 정보
		막차 및 첫차 정보
Search (UI-06)	경로 검색 화면	검색 창
		출발지 도착지 변경 버튼
		경로 표시 화면
Route (UI-07)	경로 표시	경로 표시된 지도
		위치 공유 버튼, AR 경로 안내 버튼
		경로 표시 bottom sheet
Location Sharing (UI-08)	위치 공유 팝업	위치 공유 링크 복사 버튼
		취소 버튼
AR (UI-09)	AR 경로 안내	AR 화면 (화살표로 방향 표시)
		지도 화면

## 2) UI Flow



## ◇ DB Table(Collection) Configurations

### 1) 개요



### 2) 상세

데이터베이스명	gohome	테이블명	core_profile	설명	유저 정보
컬럼명	설명	키유형	Nullable	데이터형	
id	유저 ID	PK	NOT NULL	int(11)	
username	계정 이름	UNIQUE	NOT NULL	varchar(16)	
password	비밀번호		NOT NULL	varchar(128)	
email	이메일		NOT NULL	varchar(50)	
nickname	별칭	UNIQUE	NOT NULL	varchar(10)	
address	주소		NULL	varchar(50)	
detail_address	상세주소		NULL	varchar(30)	
created_at	가입일		NOT NULL	datetime(6)	
updated_at	수정일		NOT NULL	datetime(6)	
status	상태		NOT NULL	varchar(2)	
role	역할		NOT NULL	varchar(2)	

데이터베이스명	db_logic	테이블명	Bikestop	설명	자전거 정거장
컬럼명	설명	키유형	Nullable	데이터형	
stationid	정거장 ID	UNIQUE	NOT NULL	String	
stationName	정거장 이름		NOT NULL	String	
stationLatitude	정거장 위도		NOT NULL	Number	
stationLongitude	정거장 경도		NOT NULL	Number	

데이터베이스명	db_logic	테이블명	BikestopTravelItem	설명	유저 정보
컬럼명	설명	키유형	Nullable	데이터형	
stationid_start	유저 ID		NOT NULL	String	
stationid_end	계정 이름		NOT NULL	String	
traveltime	비밀번호		NOT NULL	Number	

#### ◇ API

##### - 로그인

- ▷ path : login/
- ▷ method : POST
- ▷ request body

Name	Type	Mandatory	Default	Example	Description
username	String	Y	-	rkdalstjd9	아이디
password	String	Y	-	qwe123	비밀번호

- ▷ response field

Name	Type	Mandatory	Default	Example	Description
token	String	Y	-	토큰값	access token

##### - Refresh Token

- ▷ path : refresh/
- ▷ method : POST
- ▷ request body

Name	Type	Mandatory	Default	Example	Description
token	String	Y	-	토큰값	만료된 access token

- ▷ response field

Name	Type	Mandatory	Default	Example	Description
token	String	Y	-	토큰값	access token

- 회원가입

- ▷ path : signup/
- ▷ method : POST
- ▷ request body

Name	Type	Mandatory	Default	Example	Description
profile	Object	Y	-	-	아이디
profile.username	String	Y	-	qwe123	비밀번호
profile.password	String	Y	-	qwe123	비밀번호
profile.email	String	Y	-	rkdalstjd9@naver.com	이메일
profile.nickname	String	Y	-	arkss	닉네임
profile.address	String	N	""	서울특별시 동대문구 전농동 103-45	API에 검색 가능한 주소
profile.detail_address	String	N	""	주영리빙텔 108호	상세 주소
profile.address_lat	Float	N	0.0	37.5642135	위도
profile.address_log	Float	N	0.0	127.0016985	경도

▷ response field

Name	Type	Mandatory	Default	Example	Description
response	String	Y	success	success	성공
message	String	Y	profile이 성공적으로 생성되었습니다.	profile이 성공적으로 생성되었습니다.	상세 메시지

- 유저 상세 정보

- ▷ path : profile/
- ▷ method : GET
- ▷ request parameter

Name	Type	Mandatory	Default	Example	Description
-	-	-	-	-	-

▷ response field

Name	Type	Mandatory	Default	Example	Description
response	String	Y	success	success	성공
data	Object	Y	-	-	data를 감싸는 object 이름
data.username	String	Y	-	rkdalstjd9	아이디
data.email	String	Y	-	rkdalstjd9@naver.com	이메일
data.nickname	String	Y	-	arkss	닉네임
data.address	String	Y	-	서울특별시 동대문구 전농동 103-45	API에 검색 가능한 주소
data.detail_address	String	Y	-	주영리빙텔 108호	상세주소
data.address_lat	Float	Y	-	37.5642135	위도
data.address_log	Float	Y	-	127.0016985	경도



- 공유 경로 생성

▷ path : share/route/

▷ method : POST

▷ request body

Name	Type	Mandatory	Default	Example	Description
-	-	-	-	-	-

▷ response field

Name	Type	Mandatory	Default	Example	Description
response	String	Y	success	success	성공
message	String	Y	route가 성공적으로 생성되었습니다.	route가 성공적으로 생성되었습니다.	상세 메세지
data	Object	Y	-	-	data를 감싸는 object 이름
data.profile	Integer	Y	-	5	유저의 아이디
data.route_id	Integer	Y	-	2	생성한 경로의 아이디

- 공유 경로에 위치 좌표 생성

▷ path : share/<int:route\_id>/position/

▷ method : POST

▷ request body

Name	Type	Mandatory	Default	Example	Description
lat	Float	Y	-	37.56442135	위도
log	Float	Y	-	127.0016985	경도

▷ response field

Name	Type	Mandatory	Default	Example	Description
response	String	Y	success	success	성공
message	String	Y	position이 성공적으로 생성되었습니다.	position이 성공적으로 생성되었습니다.	상세 메세지
data	Object	Y	-	-	data를 감싸는 object 이름
data.route	Integer	Y	-	2	경로의 아이디
data.lat	Float	Y	-	37.56442135	생성한 경로의 아이디
data.log	Float	Y	-	127.0016985	경도

- 공유 경로에 위치 좌표 조회

▷ path : share/<int:route\_id>/position/

▷ method : GET

▷ request parameter

Name	Type	Mandatory	Default	Example	Description
-	-	-	-	-	-

▷ response field

Name	Type	Mandatory	Default	Example	Description
response	String	Y	success	success	성공
data	List	Y	-	-	data를 담는 배열
data[].route	Object	Y	-	3	경로의 아이디
data[].lat	Float	Y	-	37.56442135	위도
data[].log	Float	Y	-	127.0016985	경도

- 자전거 대여소 목록 조회 (주변의 따릉이 대여소 정보를 반환)

▷ path : /api/bikestops

▷ method: GET

▷ request parameter

Name	Type	Mandatory	Default	Example	Description
lat	Number	Y	0	37.58396981971035	위도
lon	Number	Y	0	127.05907344818158	경도
n	Number	N	MAX	5	검색할 대여소의 최대 수

▷ response field

Name	Type	Mandatory	Default	Example	Description
result	Number	Y	1	1	1: 정상 처리됨 -1: 오류
message	String	Y	'success'	'13 found'	요청 처리 결과
data	Object	Y			
data.n	Number			1	검색된 대여소 수
data.bikestops	Array				검색된 대여소 목록
data.bikestops[].rackTotCnt	Number				거치대 개수
data.bikestops[].stationName	String				대여소 이름
data.bikestops[].parkingBikeTotCnt	Number				자전거 수
data.bikestops[].shared	Number				거치율
data.bikestops[].stationLatitude	Number				위도
data.bikestops[].stationLongitude	Number				경도
data.bikestops[].stationId	String				대여소 ID
data.bikestops[].distance	Number				대여소까지의 직선 거리

- 자전거 거치 수 조회

▷ path: /api/bikestop\_parked\_counts

▷ method: GET

▷ request parameter

Name	Type	Mandatory	Default	Example	Description
-	-	-	-	-	-

▷ response field

Name	Type	Mandatory	Default	Example	Description
result	Number	Y	1	1	1: 정상 처리됨 -1: 오류
message	String	Y	'success'	'13 found'	요청 처리 결과
data	Object	Y			
data.n	Number			1	검색된 대여소 수
data.bikestops	Array				검색된 대여소 목록
data.bikestops[].stationId	String				대여소 ID
data.bikestops[].parkingBikeTotCnt	Number				자전거 수

- 경로탐색

▷ path: /routes

▷ method: GET

▷ request parameter

Name	Type	Mandatory	Default	Example	Description
lat_start	Number	Y		37.586759	출발지 위도
lon_start	Number	Y		127.053907	출발지 경도
lat_end	Number	Y		37.582931	도착지 위도
lon_end	Number	Y		127.060980	도착지 경도
include_bike	String		Y	N	저전거 포함 여부
include_bus	String		Y	N	버스 포함 여부

▷ response field

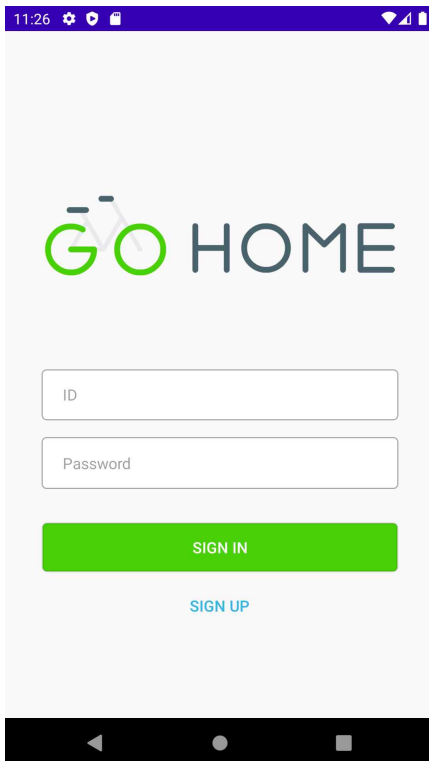
Name	Type	Mandatory	Default	Example	Description
result	Number	Y	1	1	1: 정상 처리됨 -1: 오류
message	String	Y	'success'	'13 found'	요청 처리 결과
data	Array	Y			검색된 경로 목록
data[].buspath	Object				버스구간의 추가정보
data[].sections	Array	Y			경로의 구간 목록
data[].buspath.time	Number	Y		39	소요시간(분)
data[].buspath.transit_count	Number	Y		1	버스 환승 횟수
data[].buspath.routeNames	Array	Y		[ ' N 6 2 ' , 'N13']	탑승 버스 노선번호 (탑승 순)
data[].sections[].time	Number	Y		270	구간 소요시간
data[].sections[].points	Array	Y		[[37.534636452132624, 126.99430937255515], ...]	경로 체크포인트(위도, 경도 순)
data[].sections[].type	Number	Y		1	구간 타입 1: 도보 2: 자전거 3: 버스
data[].sections[].stationNameStart	String		-	'서울 중부기 술 교 육 원 . 불 루스퀘어'	버스나 자전거 구간의 탑승 지 이름
data[].sections[].stationNameEnd	String		-	'우산빌딩앞'	버스나 자전거 구간의 하차 지 이름
data[].sections[].stationLatitudeStart	Number		-		버스나 자전거 구간의 탑승 지 위도
data[].sections[].stationLongitudeStart	Number		-		버스나 자전거 구간의 탑승 지 경도
data[].sections[].stationLatitudeEnd	Number		-		버스나 자전거 구간의 하차 지 위도
data[].sections[].stationLongitudeEnd	Number		-		버스나 자전거 구간의 하차 지 경도

## 3. 결과 및 평가

### 3.1 완료작품 소개

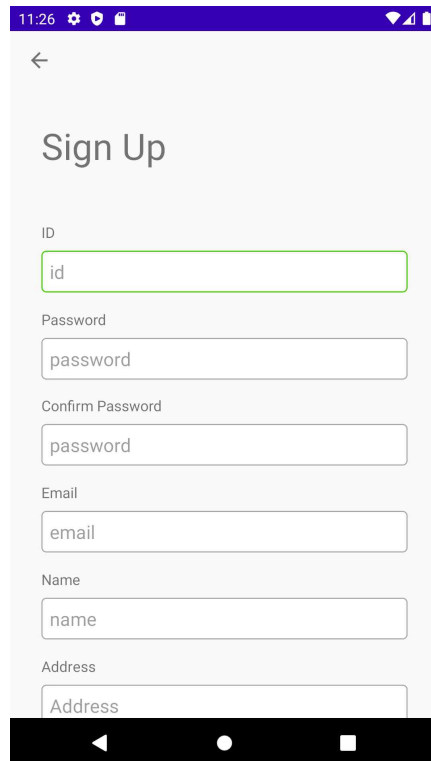
#### 가. 프로토타입 사진

◇ Login (UI-01)



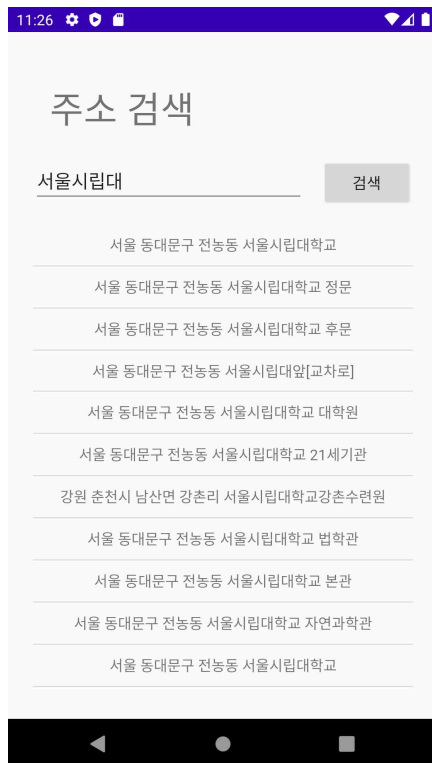
The login screen features the 'GO HOME' logo at the top, with 'GO' in green and 'HOME' in blue. Below the logo are two input fields: 'ID' and 'Password'. A prominent green 'SIGN IN' button is positioned below the password field, and a blue 'SIGN UP' link is located at the bottom of the screen.

◇ Sign Up (UI-02)

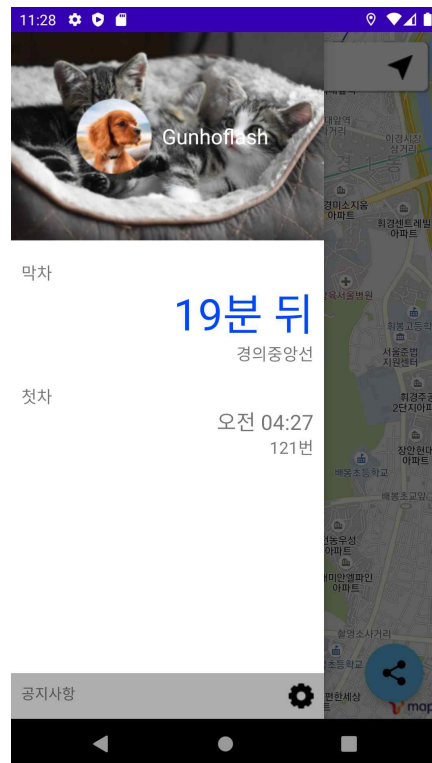


The sign up screen includes a back arrow at the top left and the title 'Sign Up'. It contains six input fields: 'ID', 'Password', 'Confirm Password', 'Email', 'Name', and 'Address'. Each field is accompanied by a placeholder text matching the field's label. The fields are arranged vertically, and the screen has a clean, minimalist design.

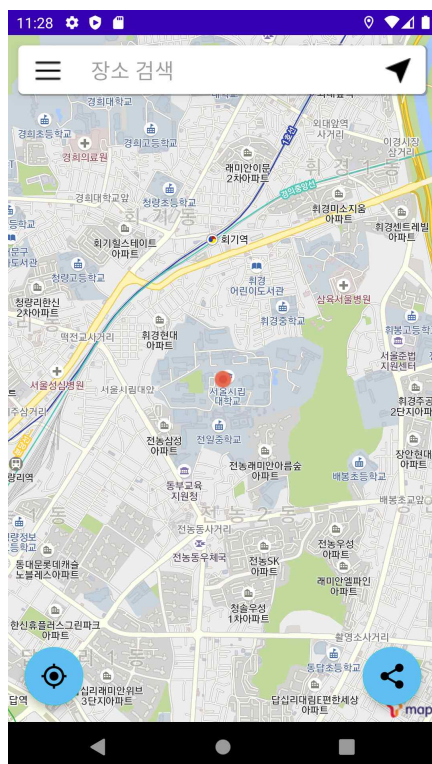
### ◇ Address (UI-03)



### ◇ Drawer (UI-05)



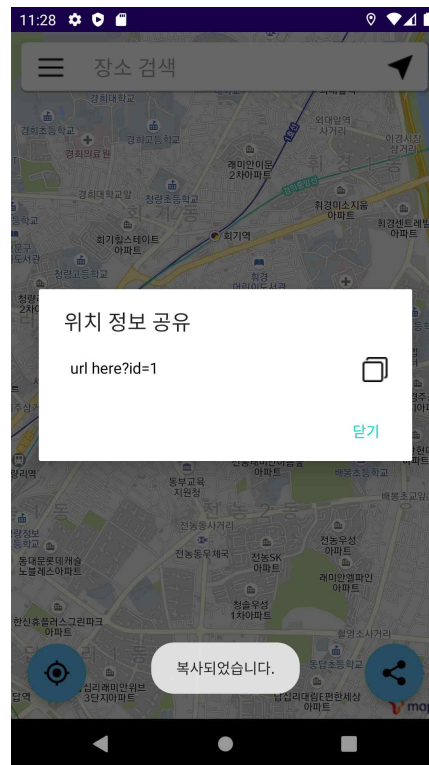
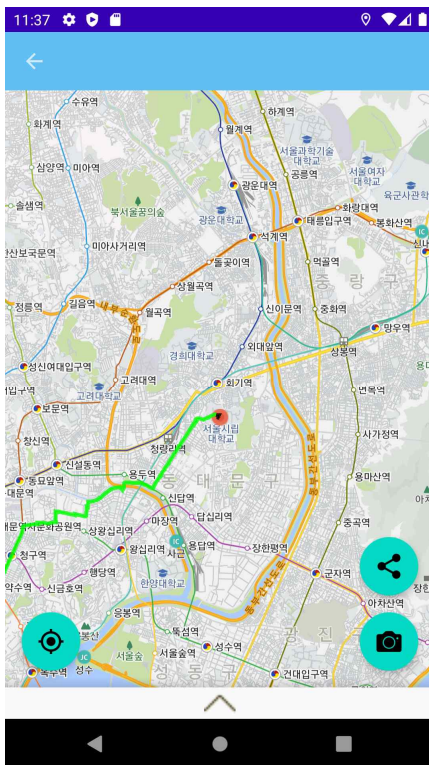
### ◇ Map (UI-04)



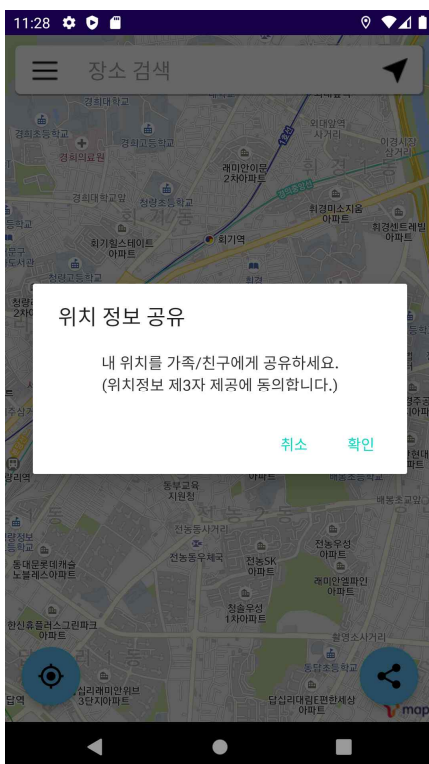
### ◇ Search (UI-06)



### ◇ Route (UI-07)



### ◇ Location Sharing (UI-08)



### ◇ AR (UI-09)

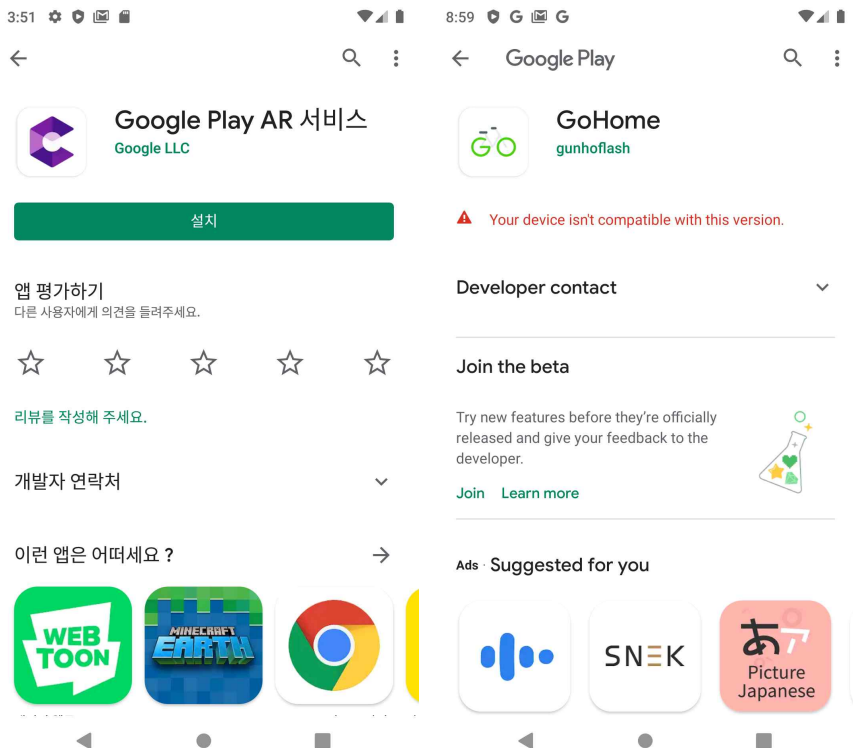


## 3.2 설치 (Configuration)

### 가. SW/HW 구성 방법 (단계에 따른 화면 포함)

#### ◇ 일반 사용: Android Application

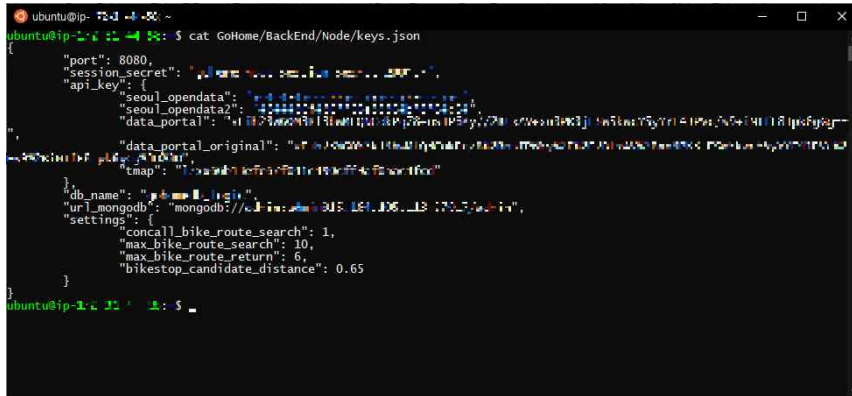
1. Android 7.0(API 수준 24) 이상의 운영체제를 가진 스마트폰이 필요하다.
2. Google Play Store에서 <Google Play AR 서비스>를 설치한다.
  - 1.1. 설치 링크: <https://play.google.com/store/apps/details?id=com.google.ar.core&hl=ko>
2. Google Play Store에서 <GoHome>을 설치한다.
  - 2.1. 설치 링크: <https://play.google.com/store/apps/details?id=com.uos.gohome>





## ◆ 서버 구축: Node.js

1. 실행환경에 Node.js, NPM을 설치한다.
2. Github을 통해 서버 코드를 실행환경에 내려받고, 같은 경로에 keys.json을 작성한다.
  - 2.1. keys.json에는 port, api key 등의 설정값이 포함된다.



3. 명령창에서 `npm install`을 실행하여 필요한 파일을 내려받는다.

◇ 서버 구축: Django Server

1. Github을 통해 서버 코드를 실행환경에 내려받고, 같은 경로에 secrets.json을 작성한다.
  - 1.1. secrets.json에는 프로젝트의 key값, DB정보, Email 정보 등의 설정값이 포함된다.
2. requirements.txt에 있는 pip패키지들을 실행환경에 다운받는다.
3. 서버를 실행한다.

### 3.3 실행 (Run)

### 가. SW 실행 방법 (단계에 따른 화면 포함)

◇ 일반사용자 실행: Android Application

1. 인터넷 연결 확인 후, 기기에 설치된 GoHome 앱을 실행한다.



◇ 서버 실행: Node.js Server

1. 명령창에서 `npm start`를 실행한다.

[illegible]

◇ 서버 실행: Django Server

1. settings/debug를 통해 debug 모드로 서버를 실행할 수 있다.
  - 1.1. `python manage.py runserver --settings=config.settings.debug`
2. settings/deploy를 통해 deploy 모드로 서버를 실행할 수 있다.
  - 2.1. `python manage.py runserver --settings=config.settings.deploy`

### 3.4 성과물

#### 가. 대외전시 포스터

# 새벽 귀가. GoHome으로 해결하세요

# N버스 # 파릉이 # AR # 위치공유 #막차안내



**새벽 귀가.  
파릉이와 함께!**

N버스 정류장이 너무 멀어  
택시를 타시곤 했나요?  
파릉이를 포함한 길 안내로  
귀가를 해보세요!



**AR 길찾기로  
길치 구원!**

길찾기를 하고서도  
방향을 모르는 당신!  
AR을 통한 길찾기로  
길을 탈출해보세요!



**위치 공유기능으로  
안전한 귀가를!**

"어디쯤 오고 있어?"  
위치 공유기능으로  
안전한 새벽귀가를  
경험해보세요!



**막차 알림으로  
놓치지 말자!**

12시만 되면 떠나는  
신데렐라같은 당신!  
막차 알림 서비스로  
막차를 놓치지 말아요!

 **지금 구글 플레이스토어 에서 다운로드 받으세요!**



서울시립대학교  
UNIVERSITY OF SEOUL



GO HOME

## 나. 로고



그림 37 기본형



그림 38 세로형



그림 39 가로형

### 3.5 완료 작품의 평가

평 가 항 목	평 가 방 법	적 용 기 준	개 발 목표치	비중 (%)	평가결과
경로 탐색 정확도	경쟁 서비스와 최적 경로 소요시간 비교	첫 번째 최적 경로 소요시간, 총 10회 이상 평가	경쟁 서비스 대비 평균 110% 미만의 소요시간	30%	평균 100% 미만
경로 탐색 속도	소요 시간 측정	10회 이상 측정	평균 5초 미만	20%	평균 2.6초
Client-Gateway 간 통신 속도	소요 시간 측정	10회 이상 측정	평균 1초 미만	15%	평균 0.5초
Gateway-Server 간 통신 속도	소요 시간 측정	10회 이상 측정	평균 1초 미만	15%	평균 0.5초
AR 표현 정확성	나침반 앱과 비교하여 각도 차이 측정	3회 이상 측정	평균 $\pm 10^\circ$ 미만	10%	평균 $\pm 3^\circ$
앱 로딩 속도	화면 전환 시간 측정	10회 이상 측정	평균 1초 미만	10%	평균 1초 미만

\* 모든 평가 항목은 조원의 테스트 기기(Galaxy S10e)를 기준으로 평가함.

```

ubuntu@ip-172-31-44-58: ~
2020-06-23 20:27:13.546 +0900 [info]: (59ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:13.550 +0900 [info]: (60ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:13.557 +0900 [info]: (62ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:13.709 +0900 [info]: (206ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:13.876 +0900 [info]: (372ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:13.877 +0900 [info]: (375ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:13.881 +0900 [info]: (364ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:13.881 +0900 [info]: (151ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:14.164 +0900 [info]: (148ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:14.176 +0900 [info]: (168ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:14.179 +0900 [info]: (167ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:14.692 +0900 [info]: (172ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:14.696 +0900 [info]: (169ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:14.709 +0900 [info]: (187ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:15.037 +0900 [info]: (161ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:15.228 +0900 [info]: (185ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:15.563 +0900 [info]: (174ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.061 +0900 [info]: (164ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.250 +0900 [info]: (168ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.259 +0900 [info]: (178ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.263 +0900 [info]: (2856ms) routes
2020-06-23 20:27:16.533 +0900 [info]: (44ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:16.537 +0900 [info]: (43ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:16.540 +0900 [info]: (34ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:16.671 +0900 [info]: (169ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:16.672 +0900 [info]: (159ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.834 +0900 [info]: (305ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.843 +0900 [info]: (318ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:16.999 +0900 [info]: (332ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.160 +0900 [info]: (141ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.320 +0900 [info]: (148ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.325 +0900 [info]: (149ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.689 +0900 [info]: (158ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.704 +0900 [info]: (175ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.840 +0900 [info]: (156ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:17.857 +0900 [info]: (176ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:18.197 +0900 [info]: (160ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:18.373 +0900 [info]: (186ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:18.535 +0900 [info]: (185ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:19.017 +0900 [info]: (153ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:19.204 +0900 [info]: (181ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:19.213 +0900 [info]: (2809ms) routes
2020-06-23 20:27:19.388 +0900 [info]: (38ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:19.391 +0900 [info]: (43ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:19.396 +0900 [info]: (41ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:19.524 +0900 [info]: (171ms) https://topis.seoul.go.kr/map/getRoute.do
2020-06-23 20:27:19.525 +0900 [info]: (159ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:19.526 +0900 [info]: (163ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:19.687 +0900 [info]: (322ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:19.853 +0900 [info]: (200ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:20.022 +0900 [info]: (151ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:20.028 +0900 [info]: (160ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1
2020-06-23 20:27:20.034 +0900 [info]: (159ms) https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1

```

경로 탐색 속도 및 통신 속도를 측정하는 테스트에서의 log 캡처



실제 나침반

157°

계산 각도

156°



349°

345°



186°

192°



299°

296°

평균 3° 오차

실제 나침반 앱과의 비교로 각도 오차를 측정하는 모습

## 3.6 향후평가

### 가. 어려웠던 내용들

#### ◇ T map API 사용

T map API 사용에 다양한 어려움이 있어 개발에 어려웠다.

T map API는 경로탐색 요청이 일 1천 건으로 제한된다고 명시되어 있다. 하지만 실제 요청 시 초당 제한량도 존재하여, 시간 집중적인 요청이 불가하다. 따라서 경로 탐색 과정에 필요한 T map API 요청의 수를 제한해야 했다. 또한, 동시에 여러 사용자로부터 요청이 몰리면 그 처리 속도가 급격히 낮아질 것으로 예상된다.

SK open API 페이지(<https://openapi.sk.com/api/detailView>)에서는 T map API를 통해 자전거 경로 탐색이 가능하다고 명시되어 있다. 하지만 이 기능은 이미 오래전 중단된 것으로 확인되었다. 이 때문에 따릉이를 포함하는 경로 탐색에 예상치 못한 큰 문제가 발생했다.

Web V2 API Guide가 매우 부실하다. Node.js Server를 개발하면서 테스트를 위해 간단히 시각화하고 싶었는데, Web V2 API에 대한 Guide가 매우 부실하여 어려움이 컸다. 많은 부분이 V1으로부터 유추해야 하는 것이어서 더욱 어려웠다.

#### ◇ 자전거를 포함한 경로 탐색

SK OPEN API에서 자전거 경로 탐색 API를 제공 중단함에 따라, 따릉이 대여소를 경유지로 하는 도보 경로를 탐색 후 소요 시간만 재계산하는 방법을 선택했다. 어떤 대여소 쌍을 선택하는가에 따라 이동 거리와 소요 시간이 달라질 텐데, SK OPEN API에 도보 경로를 요청하는 것은 시간이 제법 걸리기 때문에(앞서 언급한 초당 요청 제한 때문), 최적의 대여소 쌍을 잘 결정하는 것이 매우 중요하다.

이를 위해 경로의 소요 시간을 어림하는 다양한 방법을 생각했고, 대여소 간 실제 소요 시간을 캐시해두어 더 정확한 어림이 가능하도록 했다. 하지만 근본적인 문제의 해결책은 아니므로 더 개선해야 한다.

#### ◇ 버스를 포함한 경로 탐색

환승, 막차 시간, 배차 간격, 대기시간, 정류장 선택 등, 고려해야 하는 사항이 너무 많아서 시작이 막막했다. 공공데이터 포털에 대중교통 경로 탐색 API가 제공 중이지만, N버스는 탐색 대상이 아니기 때문에 이용할 수 없었다. 버스정류장, 노선 등의 요소를 모두 그래프로 표현하여 탐색하는 방법을 고려했으나 개발 및 테스트 기간이 길어지고 처리성능이 낮을 것 같아 임시로 ODSay의 API를 사용했다. ODSay의 API 또한 자전거를 고려하지는 않고 N버스가 우선순위에서 낮아 실질적 이용이 어렵다. 이후 서울 연린데이터 광장에 문의한 결과 TOPIS 엔진을 직접 호출하면 가능할 것이라고 안내받았다. 따라서 현재는 TOPIS 엔진을 직접 호출하여 N버스 경로를 탐색한다.

#### ◇ API gateway 및 Authentication 서버

클라이언트와 다른 서버들의 매개체로서 통신하는 점이 어려웠다. 인증 보안을 위해 access token과 refresh token을 발급하고 관리하는 것이 어려웠다.



#### ◇ 안드로이드 개발

Java와 Android 개발에 익숙하지 않아서 어려움을 많이 겪었다. 각종 UI 구성 요소를 사용하는 데 많은 시간이 걸렸다. Java와 함께 Kotlin을 사용했으면 더 빠른 개발이 가능했을 것 같다.

#### ◇ 서버와 통신 실패

외부 Library Retrofit을 사용하여 Django 서버와 통신을 시도하였으나 실패하였다. Retrofit에서 제공하는 Sample과 다른 예제들을 동일하게 작성하였음에도 불구하고 오류가 발생하였고, 어떤 부분에서 오류가 발생했는지 찾기가 쉽지 않아 디버깅에 어려움이 있었다. Android Studio에서 http를 통해 데이터를 받아 오는 것을 거부하기 때문에 발생한 문제였고, Manifest에서 network-security-configuration을 변경하여 해결하였다.

#### ◇ Quaternion에 대한 이해 및 적용

Quaternion은 3차원 객체에 대한 회전 정보를 담고 있는 개념이다. 일반적으로 회전 정보를 담는 데에는 고정축의 X, Y, Z 회전 값을 받는 Euler Angles 개념을 사용하기에 개념 자체를 이해하는 것이 매우 어려웠다. 또한, Euler Angle은 여러 개의 회전 정보를 차례로 적용하면 회전 정보가 합쳐진 최종 회전이 가능하지만, Quaternion은 값의 합성을 통해 회전을 적용해야 했기에 구현에 어려움을 겪었다.

#### ◇ 손전등 기능의 적용

심야 시간 AR 경로 탐색을 요하는 프로젝트의 특성상 3D모델링을 위한 평면을 탐색하는 과정에서 인식을 높이기 위해 AR 경로 탐색 중 손전등을 동시에 On/Off 해야 할 필요성이 있었다. 모바일 기기 내의 하드웨어를 담당하는 Camera2 객체를 직접 접근하여 조작하는 방법, Torch 클래스를 활용하여 손전등을 켜는 방법 등을 활용하여 구현을 시도했지만, ARCore 자체에서 AR과 손전등을 동시에 사용하기가 불가능하여 구현할 수 없었다. 손전등이 ARCore가 사용하고 있는 카메라 권한을 얻어오지 못하는 것이라고 생각한다.

#### ◇ 방위각의 계산

AR 경로 탐색에서 사용자가 진행해야 하는 방향을 표시하는 과정에서 현재 위치에서 목표 위치의 방위각 계산이 필요했다. 안드로이드 기기 내의 가속도 센서, 자기장 센서를 활용했는데 사용자가 이동할 때마다 가속도 센서값의 변화로 인해 방위각 값의 오차가 매우 커지는 현상이 발생했다. 이로 인해 3D 화살표 모델이 크게 흔들려 제대로 된 경로를 표시하지 못했다. 이를 해결하기 위해 가속도 센서를 통한 방위각의 보정 값을 조정하여 오차를 크게 줄였다.

## 나. 차후 구현할 내용

### ◇ AR 도보 안내 화면에서 지도 표시

우리의 AR 도보 안내는 Google Maps의 Live View의 국내 사용 불가에 대한 문제를 해결하고 AR 도보 안내의 실용성을 검증하고자 구상되었다. Google Maps의 Live View에서는 AR 이미지로 경로를 안내할 뿐만 아니라, 화면 하단에 지도를 함께 띄워줌으로써 더 효과적인 경로 안내를 제공한다. GoHome은 현재 AR 이미지만 표시할 뿐, 지도까지 함께 보여주지는 않는다. Live View처럼 지도를 함께 표시하려 했으나, 개발 난이도가 높아 구현하지 않았다.

### ◇ 외부 독립적인 버스 경로탐색 알고리즘 작성

현재 버스를 포함하는 경로탐색은 TOPIS의 경로탐색 엔진을 일부 활용하고 있다. 하지만 이 엔진은 공식적으로 제공되는 것이 아니어서(하지만 관련 문의 과정에서 사용을 안내받았다) 품질이 좋지 않다. 특히 요청에 대한 응답속도가 매우 상이(수십ms ~ 수만ms)하다. 따라서 버스 경로탐색 과정에서 TOPIS 의존성을 줄이거나 없앨 필요가 있다. 이를 위해 직접 N버스 데이터를 가공/연산하여 최적 경로를 빠르게 탐색하는 알고리즘을 개발해야 한다.

### ◇ GPS 오차 보정

GPS 값은 오차가 심하다. 현재는 GPS 값을 그대로 사용하고 있는데, GPS의 오차 때문에 AR 도보 안내와 위치공유 등에서 낮은 안정성을 보인다. 이를 해결하기 위해 GPS 값을 보정해야 한다. 가장 단순한 방법은 직전 GPS 값과의 평균을 사용하는 것이다. 사용자의 실제 위치가 극단적으로 빠르게 변화할 가능성은 적고, 그 속도가 버스보다 빠를 경우는 고려하지 않아도 될 수준이다. 따라서 현재 GPS 값과 이전에 측정된 GPS 값들을 바탕으로 적절한 가중치를 적용해 최종 GPS 값을 계산함으로써 GPS 오차를 보정할 수 있을 것이다. 더 나아가 AI를 이용하여 GPS 값을 추정할 수도 있을 것이다.

### ◇ bus와 지하철에 대한 막차, 첫차 알림

일반적으로 심야 귀가는 더 오랜 시간이 걸리기 때문에 N버스가 아닌 일반적인 대중교통을 이용할 수 있도록 막차와 첫차 알림을 지원한다.

### ◇ 주변 숙박업소 추천

심야에 여러 교통수단을 통해서도 귀가가 힘들 경우를 대비하여 주변 숙박업소에 대한 정보를 제공하고 추천한다.

### ◇ HTTP/2 통신

HTTP/2 통신은 헤더압축, 서버푸시 등을 지원한다. GoHome의 위치 공유 기능 등에 사용하면 통신량과 지연을 낮춰 통신 효율을 크게 늘릴 수 있을 것이다.

◇ 더 자세한 위치 공유 기능

현재 위치 공유기능은 개발 일정 조율 등의 사유로 현 위치만 공유되며 집까지의 경로는 공유되지 않는다. 이를 지원하기 위해서는 상세 경로 정보를 저장하고 효율적으로 관리해야 한다.

◇ 자동로그인 기능

자동로그인 기능은 프로젝트에서 개발 우선순위가 매우 낮아 결국 구현하지 않았다. 하지만 사용자 편의를 위해 필요한 사항 중 하나로, 차후 구현이 필요하다.

## 4. 개발 사업비 정산

### 4.1 구성원 및 추진체계

#### ◇ 구성원

이름	책임	세부 내용
김건호	프로젝트 관리자(PM) Logic Server 구현	1. 전체 프로젝트 관리/감독 2. 최종 검토 및 승인 3. Node.js를 이용하여 로직 서버 구현 4. 길 찾기 알고리즘 구현
강민성	User Management Server 구현 API Gateway Server 구현	1. Django를 이용하여 회원 관리 서버 구현
이상엽	문서 작성 AR 구현	1. ARCore 및 Sceneform을 이용한 AR 경로 안내 구현 2. 회의록 및 문서 작성
최연웅	Front End 구현	1. Android Application 작성 2. 클라이언트-서버 간 통신 구현
홍찬표	Front End 구현	1. Android Application 작성 2. 클라이언트-서버 간 통신 구현

#### ◇ 추진체계



## 4.2 자재소요서

(해당 없음)

## 4.3 개발사업비 내역서

	항 목 (품명, 규격)	수 량	단 가	금 액			비 고
				계	현금		
직 접 개 발 비							
	합 계	0	0	0			

(단위: 천원)

## 부 록

### A-1 참고문헌 및 참고사이트

1. Django
  - 1.1. <https://docs.djangoproject.com/en/3.0/>
2. drf
  - 2.1. <https://www.django-rest-framework.org>
3. Node.js
  - 3.1. <https://nodejs.org/api/documentation.html>
4. MongoDB
  - 4.1. <https://docs.mongodb.com>
5. Android
  - 5.1. <https://developer.android.com/docs?hl=ko>
6. ARcore
  - 6.1. <https://developers.google.com/ar/reference>
  - 6.2. <https://en.wikipedia.org/wiki/ARCore>
  - 6.3. <https://poly.google.com>
  - 6.4. <https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/40876/versions/8/previews/sensorgroup/Examples/html/CapturingAzimuthRollPitchExample.html>

### A-2 관련특허

(해당 없음)

### A-3 소프트웨어 프로그램 소스

[Github] <https://github.com/arkss/GoHome>

## 설계구성요소 및 제한요소

설계구성요소	과제보고서내의 항목	내용 요약
목 표 설 정	1.1 개발과제의 개요	과제의 배경, 목표, 내용을 설정함
합 성	2.2 사용자 요구사항 및 정량목표 2.3 기능구현을 위한 세부기술 2.4 시스템 설계 2.7 소프트웨어 설계	구성요소를 개념 및 세부적으로 설계함
분 석	2.1 사용자 요구사항 2.3 기능구현을 위한 세부기술 2.4 이론적 계산 및 시뮬레이션 2.7 소프트웨어 설계	요구사항에 대한 구성요소를 분석함
제 작	3.1 완료작품소개 3.2 설치 3.3 실행	실제 서비스에 대한 설치, 실행 과정
시 험 및 평 가	3.5 완료작품의 평가 3.6 향후평가	완성 작품에 대한 평가

설계제한요소	과제보고서내의 항목	내용 요약
원 가	1.3 관련시장에 대한 분석 4.2 자재소요서 4.3 개발사업비 내역서	시장에 따른 소요 비용 내역
안정성 및 윤리성	1.1 개발과제의 개요 1.3 관련 시장에 대한 분석	관련 시장에 따른 기대효과 분석
신뢰성	2.2 기능 정의 및 기능별 정량목표 3.5 완료 작품의 평가 3.6 향후평가	완성 작품에 대한 수행능력 정량 분석
사회에 미치는 영향	1.1 개발과제의 개요 1.3 관련 시장에 대한 분석	관련 시장에 따른 영향 분석
환경요인	2.4 시스템 설계 2.5 이론적 계산 및 시뮬레이션	서비스에 대한 설계 내용
기 타	3.6 향후평가	차후 구현하고자 하는 내용