Mobility Mate

교통약자를 위한 네비게이션 앱

Software Design Specification

2021 학기 **1** 학기 소프트웨어공학개론 **3** 팀

이석현 **2014311748**

안리아 **2016312126**

우준하 **2015318620**

이주민 **2017310695**

Table of Contents

Table of Figures

Table of Figures

22. < Table 장소 검색 >

23. < Table 장소 검색 응답 >

24. < Table 경로 찾기 >

25. < Table 경로 찾기 응답 >

26. < Table 경로 안내 >

27. < Table 경로 안내 응답 >

1.Preface

이 장에서는 독자 정보, 독자층, 범위, 이 문서의 목적과 이 소프트웨어 디자인 명세서의 문서구조를 포함하고 있다.

1.1. Readership

이 소프트웨어 디자인 명세서는 다양한 하위 섹션이 있는 10 개의 섹션으로 나눠진다. 소프트웨어 디자인 명세서의 구조는 이 SSD 문서 구조의 하위 섹션에서 아래로 나열된대로 찾을 수 있다. 이 명세서는 팀 3 가 주요 독자가 된다. 추가적으로 소프트웨어 공학 수업의 교수님, 조교 및 다른 학생들이 주요 독자가 될 수 있다.

1.2. Scope

이 디자인 명세서는 소프트웨어 엔지니어링과 소프트웨어 품질 엔지니어링에서 교통 약자들을 위한 안전하고 편한 경로 추천 시스템을 구현하는데 사용할 디자인 정의로 사용된다.

1.3. Objective

이 소프트웨어 설계 문서의 주요 목적은 교통 약자 맞춤 경로 안내 어플의 기술적 설계 측면에 대한 설명을 제공하는 것이다. 이 문서에서는 교통 약자 맞춤 경로 안내 어플의 구현을 위한 소프트웨어 아키텍처 및 소프트웨어 설계 결정에 대해 설명한다. 또한 시스템의 다양한 측면을 묘사하기 위해서 시스템의 구조적 개요를 제공한다. 그리고 요구사항 명세서에서 논의된 일부 모듈의 구조와 디자인을 지정하고 프로그래밍 팀이 어떻게 특정 모듈을 구현하는지 보여주는 class diagram 을 포함여 sequence diagram 과 activity diagram 으로 변환된 일부 사용 사례를 표시한다. 이 문서의 대상 독자는 교통 약자 맞춤 경로 안내 어플의 이해당사자, 개발자, 디자이너 그리고 소프트웨어 테스터들이다.

1.4. Document Structure

- 1. Preface: 이 장에서는 독자층, 문서의 범위와 구조, 시스템의 목적에 대해서 설명한다.
- 2. Introduction: 이 장에서는 이 문서에서 사용된 여러 도구와 이 문서와 참고 자료에서 사용된 여러 다이어그램, 이 프로젝트의 목적에 대해서 설명한다.
- 3. Overall System Architecture: 이 장에서는 context diagram, sequence diagram, use case diagram 을 사용해 시스템의 전체적인 구조에 대해서 설명한다.

- 4. System Architecture - Frontend: 이 장에서는 class diagram, sequence diagram 을 사용해 프론트엔드 시스템의 구조에 대해서 설명한다.
- 5. System Architecture - Backend: 이 장에서는 class diagram, sequence diagram 을 사용해 백엔드 시스템의 구조에 대해서 설명한다.
- 6. Protocol Design: 이 장에서는 클라이언트와 서버 간의 통신에 사용되는 여러 프로토콜 디자인에 대해 설명한다.
- 7. Database Design: 이 장에서는 ER diagram 과 SQL DDL 을 사용해 데이터베이스 디자인에 대해 설명한다.
- 8. Testing Plan: 이 장에서는 이 시스템을 테스트하는 계획에 대해 설명한다.
- 9. Development Plan: 이 장에서는 시스템 개발을 위해 사용할 툴, 제약사항, 가정, 종속성에 대해 설명한다.
- 10. Supporting Information: 이 장에서는 이 문서의 기준과 역사에 대해 설명한다.

1. Introduction

이 프로젝트는 교통 약자 개인에게 맞는 경로를 추천하는데 사용되는 모바일 어플리케이션을 개발하고자 설계하는 것이다. 시스템은 알고리즘을 사용해 사용자의 이동 수행 능력과 보도 상황에 맞는 경로를 찾아 사용자에게 가장 적합한 추천 경로를 결정할 수 있어야 한다. 이 디자인 명세서는 프로젝트를 구현하기 위해 사용되거나 의도된 설계를 보여준다. 설명된 디자인은 프로젝트를 위해 앞서 준비한 소프트웨어 요구사항 명세서에 지정된 요구사항을 따른다.

2.1. Objectives

이 장에서 우리는 디자인 단계에서 이 프로젝트에 적용할 다양한 툴과 다이어그램에 대해 설명한다.

2.2. Applied Diagrams

2.2.1. UML

UML 은 Unified Modeling Language 를 의미하는 약어다. 간단히 말해서 UML 은 소프트웨어를 모델링하고 문서작업하는 현대적인 접근법이다. 실제로 이것은 가장 인기 있는 비즈니스 프로세스 모델링 기법 중 하나다. UML 은 이미 존재하거나 새로운 사업 프로세스, 구조, 소프트웨어 시스템의 인공물의 동작을 설명하고 지정하고 설계하고 문서화하는 사업 분석가, 소프트웨어 구조 설계자, 개발자를 위한 공통 언어다. UML 은 다양한 어플리케이션 도메인에서 적용될 수 있다.

이것은 모든 주요 객체와 구성요소가 있는 소프트웨어 개발 방법과 다양한 구현 플랫폼에서 사용될 수 있다. 소프트웨어 구성 요소의 다이어그램적 표현을 기반으로 한다. 이러한 시각적인 표현을 사용함으로써, 우리는 소프트웨어나 사업 프로세스에서 생길 수 있는 결함이나 오류에 대해 더 잘 이해할 수 있다.

2.2.2. Use case Diagram

시스템의 토대가 되는 부분은 시스템이 충족하는 기능적 요구사항이다. Use case diagram 은 시스템의 추상적인 요구사항을 분석하는데 사용된다. 이러한 요구사항들은 서로 다른 use case diagrams 을 통해서 표현된다. UML diagram 의 세 가지 주요 구성 요소는 다음과 같다. Functional requirements - 행동을 설명하는 동사와 같은 use cases 를 사용해서 표현한다. Actors - actors 는 시스템과 상호 작용하며 Actors 는 사람이 될 수도 있고 조직이 될 수도 있고 외부적 또는 내부적 어플리케이션이 될 수도 있다. Actors 와 use cases 간의 관계 - 관계는 직선 화살표로 표현된다.

2.2.3. Sequence Diagram

Sequence diagrams 은 컴퓨터과학 커뮤니티 뿐만 아니라 비즈니스 어플리케이션 개발을 위한 디자인 레벨의 모델로서 가장 중요한 UML diagrams 일 것이다. 최근에 시각적으로 설명하기 쉬운 특성 때문에 비즈니스 프로세스를 묘사하는 분야에서 더 인기가 많아지고 있다. 이름에서 알 수 있듯이 sequence diagrams 은 actors 와 objects 간에 일어나는 메시지들의 순서와 상호작용을 설명한다. Actors 또는 objects 는 그들이 필요할 때 또는 다른 object 가 그들과 통신하기 원할 때 활성화 될 수 있다. 모든 의사소통은 시간 순으로 표시된다.

2.2.4. Class Diagram

UML 에서 class diagram 은 시스템의 클래스, 클래스의 속성, 메소드, objects 간의 관계를 보여주면서 시스템의 구조를 설명하는 정적 구조 다이어그램의 유형이다. 클래스는 객체의 구성요소기 때문에 class diagrams 은 UML 의 구성요소다. Class diagrams 에서 다양한 구성요소는 실제로 프로그래밍 될 클래스, 주요 객체, 또는 클래스와 객체 간의 관계를 보여줄 수 있다. 클래스 모양 자체는 세 개의 행이 있는 직사각형으로 이루어져 있다. 가장 위 행에는 클래스의 이름을 포함하고, 가운데 행은 클래스의 속성을 포함하고, 가장 아래 행에는 클래스가 사용하는 메소드 또는 operation 을 표현한다. 클래스와 하위 클래스는 각 객체 간의 정적인 관계를 보여주기 위해 같이 그룹화된다.

2.2.5. Context Diagram

시스템 context diagram 은 가장 추상적인 수준의 data flow diagram 이고 모델링될 시스템의 경계와 상황을 설정하는 전체 시스템을 표현하는 오직 한 프로세스만 포함한다. 시스템과 외부 entities(actors) 간의 정보의 흐름을 식별한다. Context diagram 은 전형적으로 요구사항 명세서에 포함된다. 프로젝트의 모든 이해당사자가 이것을 반드시 읽어야 하므로 쉬운 언어로 쓰여져야 하고, 모든 이해당사자가 항목을 이해할 수 있어야 한다. 시스템 context diagram 의 목적은 전체 시스템의 요구사항과 제약사항을 개발할 때 고려해야 하는 외부 요소들과 이벤트들에 초점을 맞추는 것이다. 시스템 context diagram 은 프로젝트의 초기 단계에서 조사하는 범위를 설정하기 위해 종종 사용된다. 또한 context diagram 은 시스템과 상호작용할 수 있는 모든 외부 entities 를 표현한다. 전체 소프트웨어 시스템은 하나의 프로세스처럼 보여진다. 그러한 다이어그램은 외적인 구조의 상세한 설명 없이 모든 외부의 entities 와 상호작용하는 시스템, 환경으로 둘러 싸인 시스템을 중앙에 나타낸다.

2.2.6. Entity Relationship Diagram

Entity relationship diagram 은 데이터베이스에 저장된 entity 집합의 관계를 보여준다. 이 context 에서 entity 는 데이터의 구성요소인 하나의 객체가 된다. 하나의 Entity 집합은 유사한 entity 들의 모음이다. 이러한 entity 들은 그것의 특성을 정의하는 속성을 가질 수 있다. Entity 들을 정의하고 그들간의 관계를 보여줌으로써 ER diagram 은 데이터베이스의 논리적 구조를 표현한다. Entity relationship diagrams 은 데이터베이스의 디자인을 스케치하는데 사용된다.

2.3. Applied Tools

2.3.1. Diagrams.net

Diagrams.net 은 순서도, 프로세스 다이어그램, 조직도, UML, ERD, 네트워크 다이어그램을 만들 수 있는 무료 온라인 다이어그램 소프트웨어다.

2.4. Project Scope

교통 약자 맞춤 경로 안내 어플은 공사 중인 보도, 경사로 등의 보행 환경과 교통약자가 본인의 이동 수행 능력을 고려하여 옵션을 통해 고도가 낮은 길 또는 빠른 길을 안내한다. 이때, 주변에 가까운 병원, 휠체어, 에스컬레이터, 수유실, 쉼터 등과 같은 시설 정보를 얻을 수 있다. 또한,

로그인을 통한 개인 맞춤 정보를 제공하므로, 교통약자가 직접 작성한 본인의 이동 수행 능력 정보를 수집할 수 있다. 관련된 정보로는 휠체어와 같은 이동 보조 기구 유무, 계단이나 오르막길을 다닐 수 있는 능력 등이 있다. GPS 트래킹 기능을 키면 교통약자의 이동 경로 또한 수집할 수 있다. 이는 다른 사용자에게 경로를 추천할 때 사용될 수 있다. 사용자가 기입한 정보와 사용한 이동 경로는 데이터베이스에 저장된다. 그 외에도 음성검색, 실시간 대중교통 승강기 정비 현황을 제공한다.

2.5. References

- Team 1, 2020 Spring 41 class Software Design Document, SKKU

3. System Architecture - Overall

3.1. Objectives

이 장에서, 프로젝트를 위한 어플리케이션의 프론트엔드 디자인부터 백엔드 디자인까지 시스템의 구성을 보여주고 설명한다.

3.2. System Organization

이 서비스는 client - server 모델을 적용해 설계되었다. 프론트엔드 어플리케이션은 사용자의 모든 상호작용에 반응을 하고 프론트엔드와 백엔드 어플리케이션은 JSON 형식으로 데이터를 주고받으며 HTTP 통신한다. 백엔드 어플리케이션은 데이터베이스로부터 필요한 정보를 가져와 프론트엔드에서 컨트롤러의 범위까지 송신한다. 유저가 접속했을 때 유저의 계정정보를 불러오고, 유저로부터 검색 및 어플리케이션의 기타 기능을 이용하려는 요청이 들어올 때 실시간으로 업데이트된 최신 DB 를 유저에게 제공한다. 유저에게 제공할 기본 지도 정보는 카카오맵 API 를 사용하여 데이터베이스에 저장한다. 유저가 자신의 정보를 변경하였을 때 정보를 관리하는 데이터베이스에 실시간으로 추가/수정/삭제가 반영되며 새로 업데이트된 정보를 기반으로 하여 유저에게 정보가 전달된다.

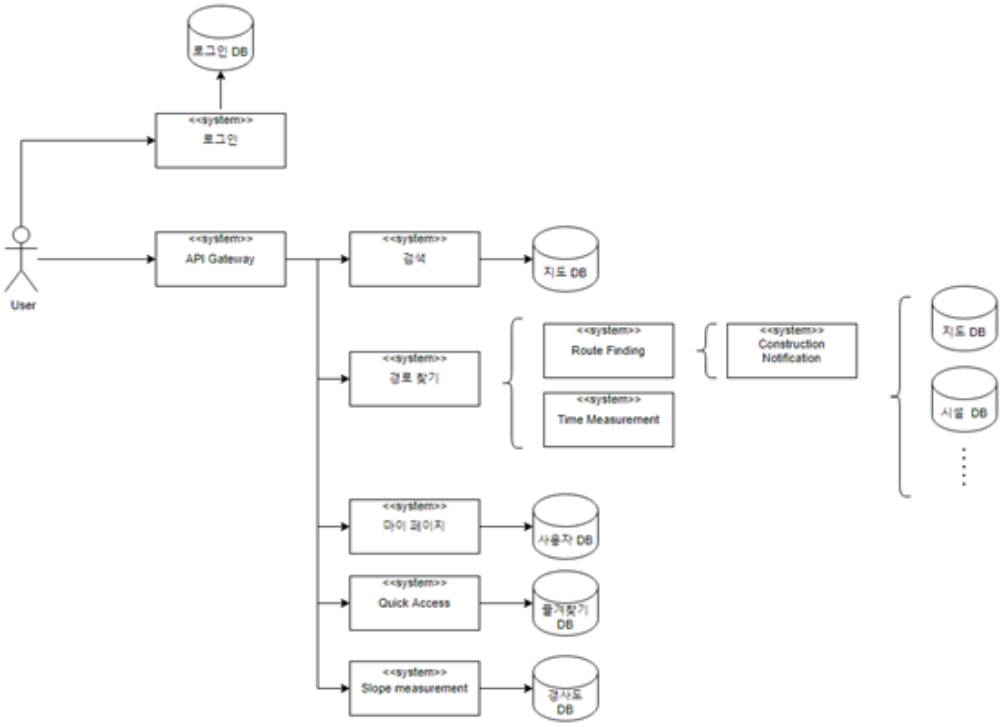


Diagram 1. Overall System Architecture

3.2.1 Context Diagram

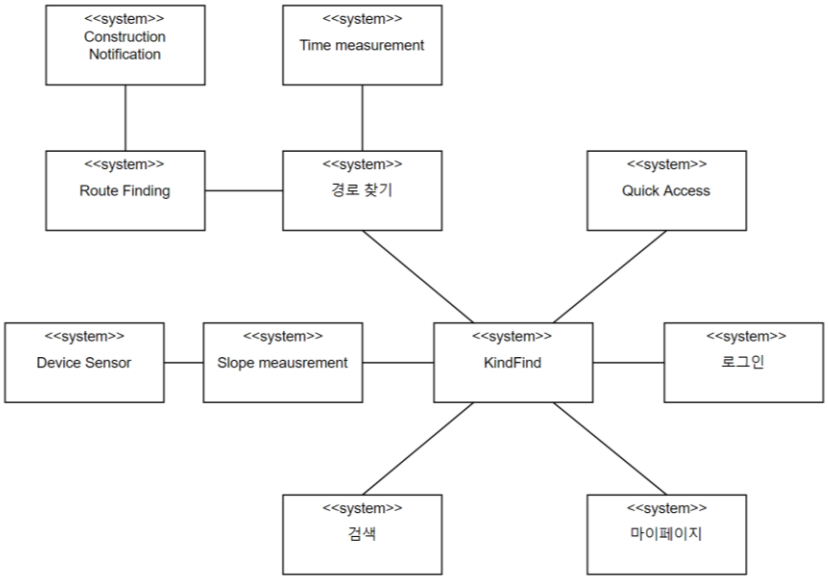


Diagram 2. Overall context diagram

3.2.2 Sequence Diagram

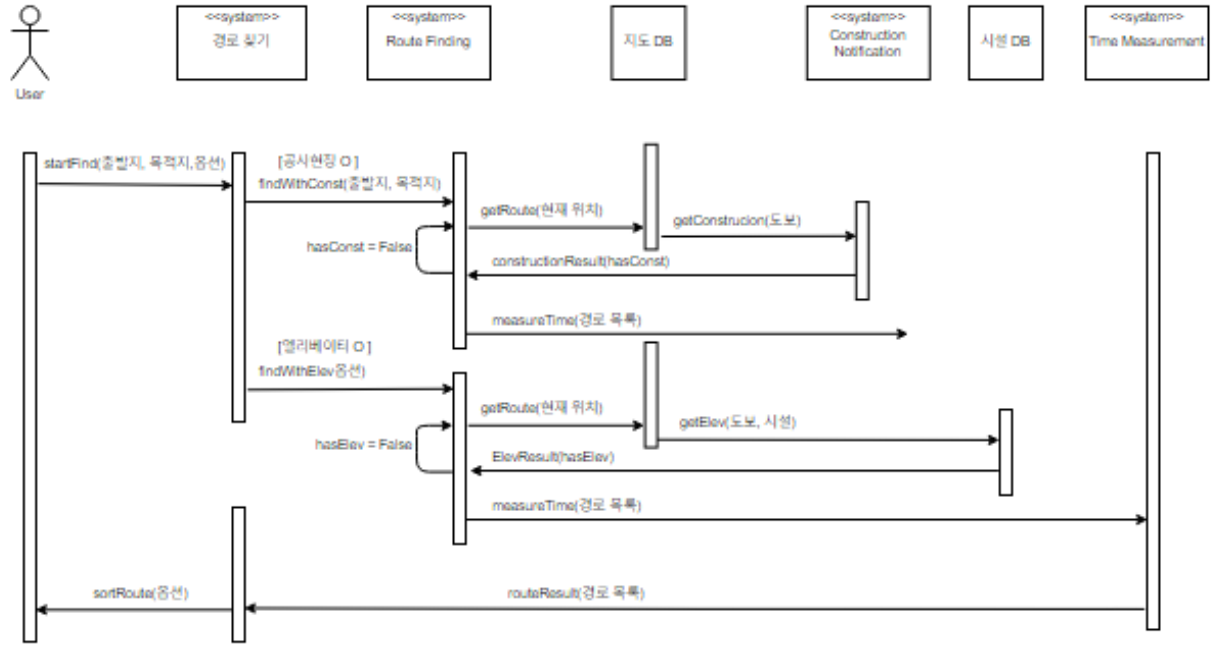


Diagram 3. Overall sequence diagram

3.2.3 Use Case Diagram

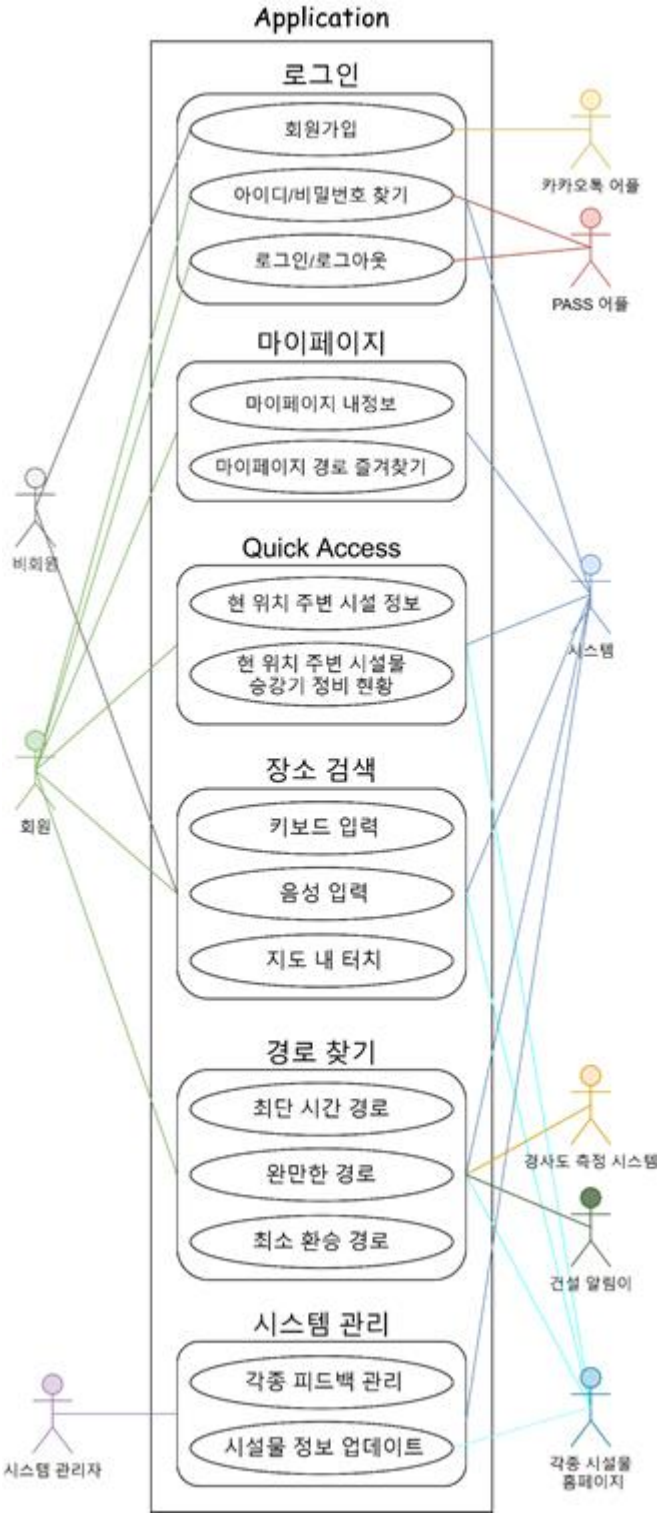


Diagram 4. Use case diagram

4. System Architecture - Frontend

4.1. Objectives

이 장에서는 프론트엔드 시스템의 구조, 속성, 기능과 각 구성요소의 관계에 대해 설명한다.

4.2. Subcomponents

4.2.1. 마이페이지

마이페이지 클래스는 기본적인 사용자 정보에 대해서 다룬다. 사용자가 등록을 한 후, 사용자는 '내정보'를 입력해야 한다. 사용자가 로그인한 후, 사용자는 사용자 정보를 수정할 수 있다.

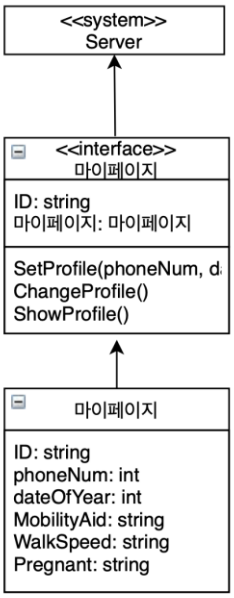
A. Attributes

- ID: 사용자가 입력한 ID
- 휴대폰 번호: 사용자의 휴대폰 번호
- 출생년도: 사용자의 출생연도
- 이동 보조기구: 사용자가 사용하는 이동 보조기구
- 보행속도: 사용자의 보행속도
- 임신여부: 사용자의 임신여부

B. Methods

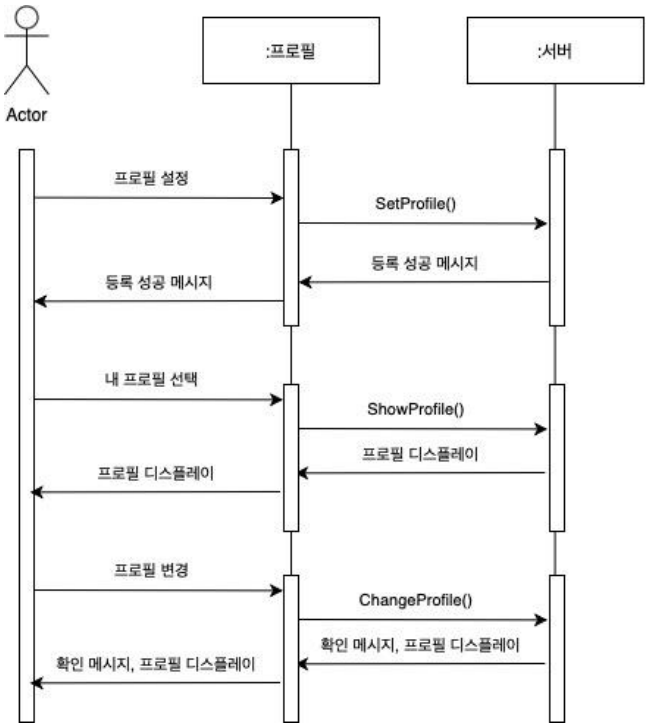
- SetProfile()
- ChangeProfile()
- ShowProfile()

C. Class Diagram



Class diagram - 마이페이지

D. Sequence Diagram



Sequence diagram - 마이페이지

4.2.2. 검색

검색 기능은 크게 키보드 입력, 음성 입력, 지도 내 터치 기능으로 구현되어있으며 유저로부터 원하는 정보를 입력받아 해당하는 정보를 제공한다.

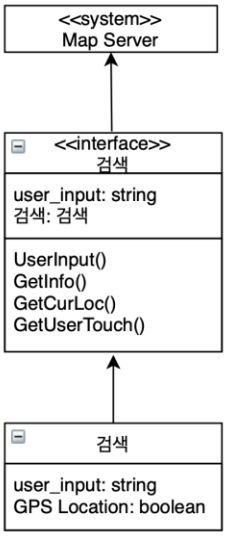
A. Attributes

- 사용자 입력
- GPS 현위치

B. Methods

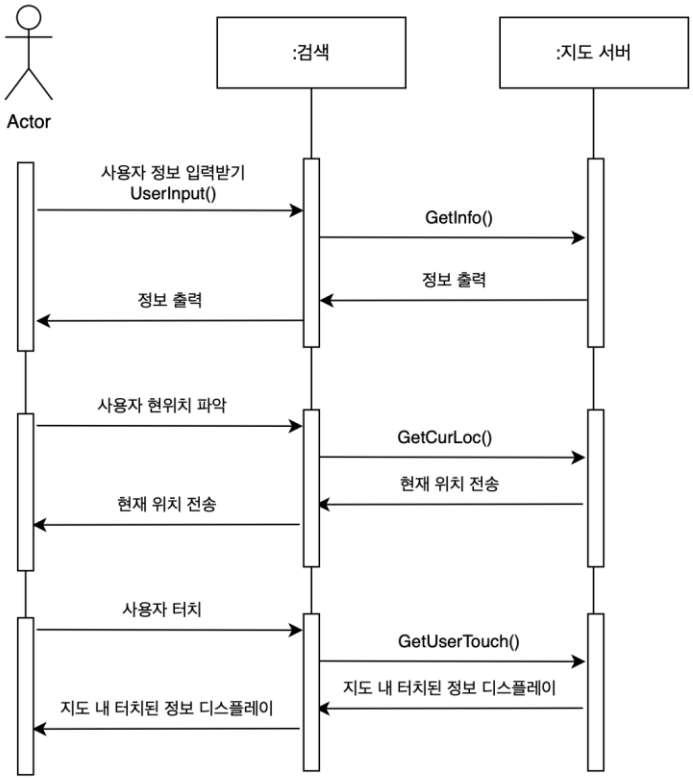
- UserInput()
- GetInfo()
- GetCurLoc()
- GetUserTouch()

C. Class Diagram



Class diagram - 검색

D. Sequence Diagram



Sequence diagram - 검색

4.2.3. 경로 찾기

경로 찾기 클래스는 사용자의 이동 수행 능력을 사용하여 작동한다. 출발지와 목적지를 입력 받고 사용자의 이동 수행 능력을 추천 시스템에 보낸다.

A. Attributes

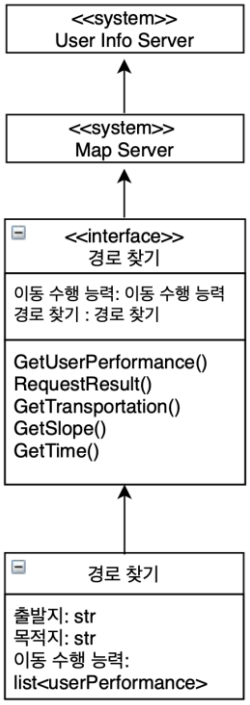
- 출발지
- 목적지
- 이동 수행 능력

B. Methods

- GetUserPerformance()
- RequestResult()
- GetTransportation()

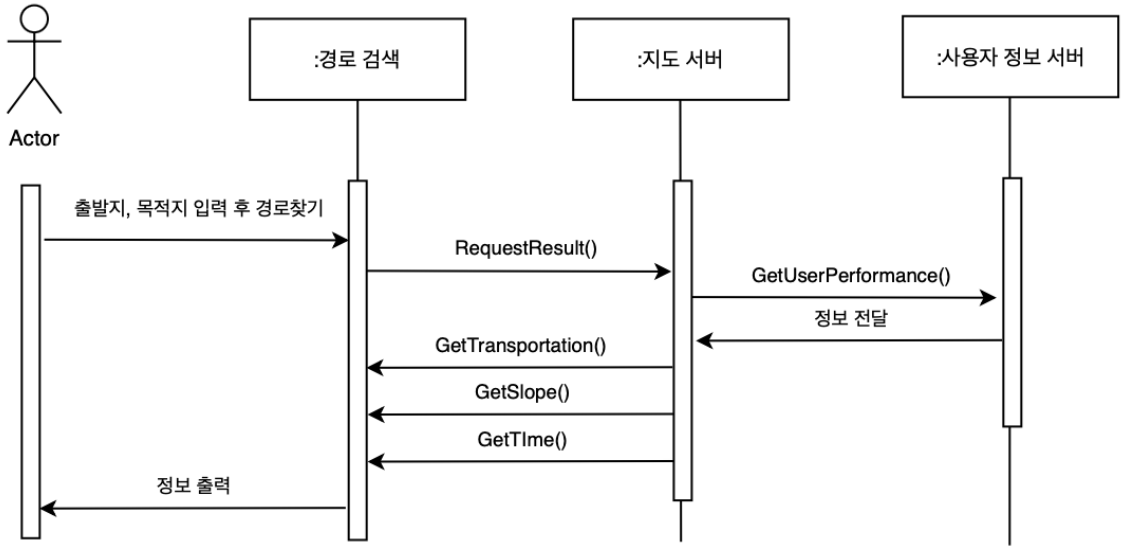
- GetSlope()
- GetTime()

C. Class Diagram



Class diagram - 경로 찾기

D. Sequence Diagram



Sequence diagram - 경로 찾기

4.2.4. Quick Access

Quick Access 는 현재 위치를 받아옴으로써 작동된다. 사용자가 설정한 반경 내에서 필요한 시설 정보, 승강기 현황 정보를 알려준다.

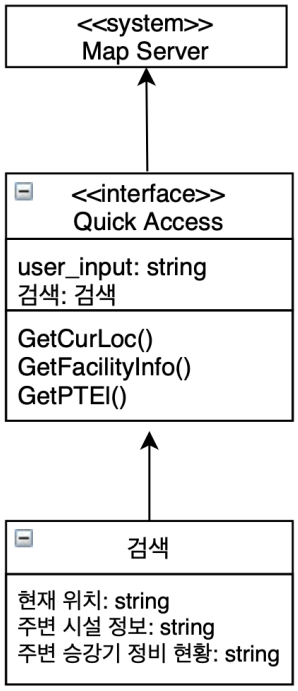
A. Attributes

- 현재 위치
- 주변 시설 정보
- 주변 승강기 정비 현황

B. Methods

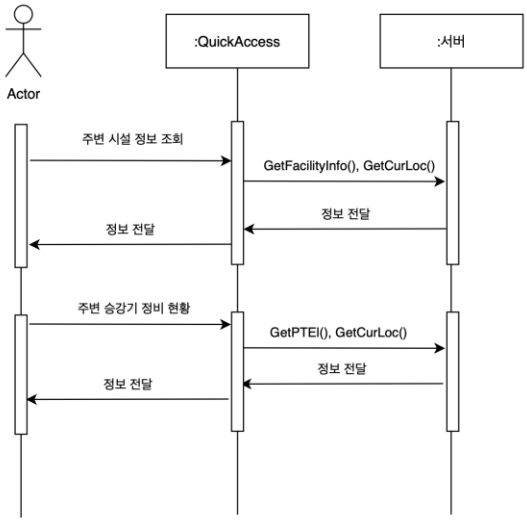
- GetCurLoc()
- GetFacilityInfo()
- GetPTEl()

C. Class Diagram



Class diagram - Quick Access

D. Sequence Diagram



Sequence diagram - Quick Access

5. System Architecture – Backend

5.1 Objectives

프로젝트 참여자간의 원활한 소통을 위해 데이터베이스와 시스템 그리고 하위 시스템들의 관계를 기술한다.

5.2. Overall Architecture

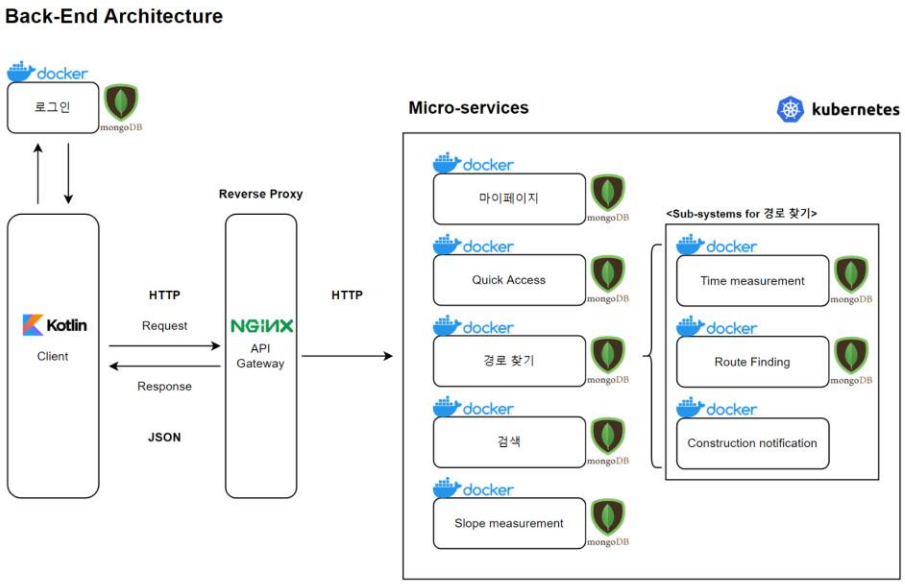


Diagram . Back-end Architecture

시스템간 응집도를 높이고 결합도를 낮추기 위해 각 기능들이 독립적으로 서비스를 제공할 수 있는 Micro service architecture 를 활용하였다. 또한, 데이터베이스를 공유하지 않고 독립적으로 가지는 Database-per-service 디자인 패턴을 사용하여 각 시스템의 에러가 다른 시스템으로 전이되지 않도록 하였다.

5.2.1 Design Pattern

A. Database-per-service

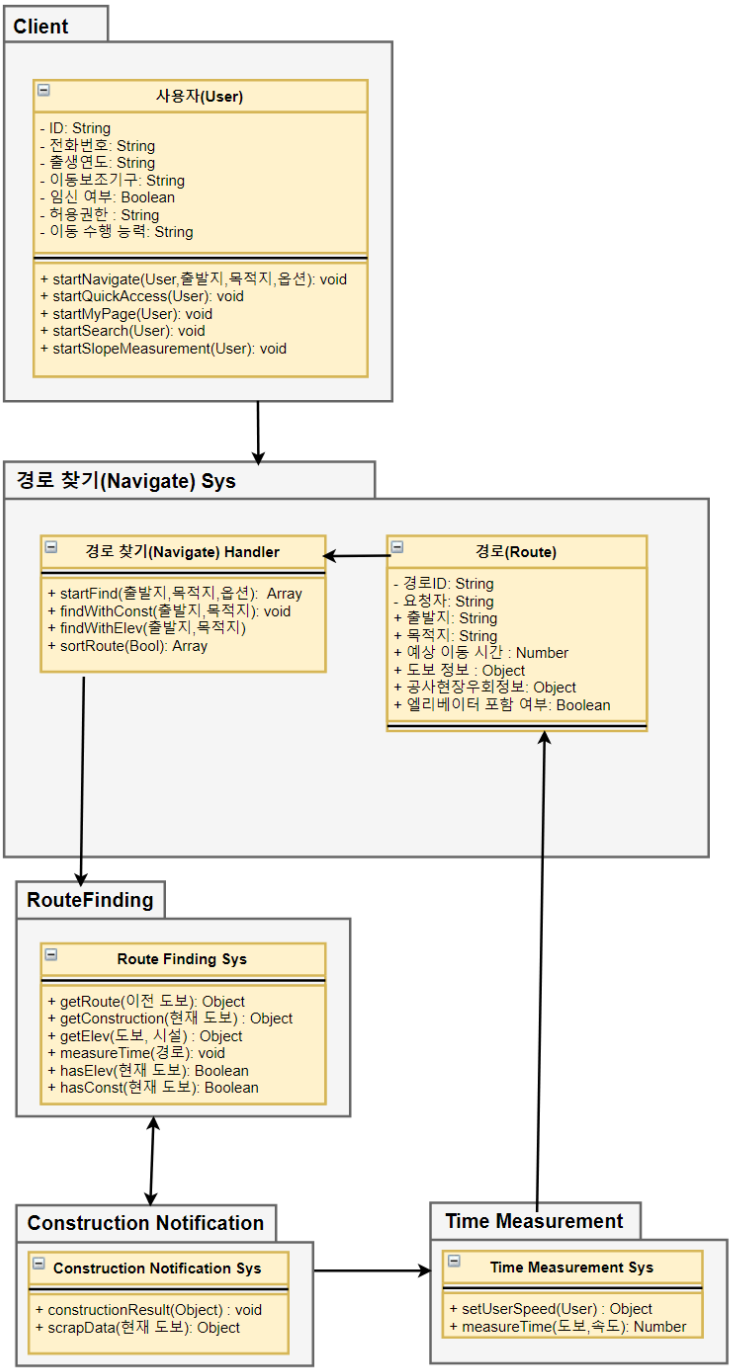
각 마이크로 서비스가 독립적으로 개발, 배포 및 확장 될 수 있도록 느슨하게 결합되기 위해 별도의 데이터베이스를 사용하도록 하는 방식이다. 같은 데이터베이스를 독립적인 서비스들이 공유하다 서비스 중 1 개 또는 데이터베이스에 문제가 발생 시 서비스가 동작하지 않게되기에 마이크로서비스 구조의 의의가 사라지게 된다. 이 때문에 서비스별 데이터베이스를 제공하고 각 서비스간 REST 통신을 통해 필요한 정보를 주고받는다.

5.3. Subcomponents

경로 찾기 시스템과 이를 구성하는 하위 시스템들에 대해 기술한다. 하위 시스템들로는 Route Finding system, Construction notification system, Time measurement system 이 있으며 각각이 독립적인 시스템으로 구성되어있다.

5.3.1. 경로 찾기 시스템

A.Class Diagram



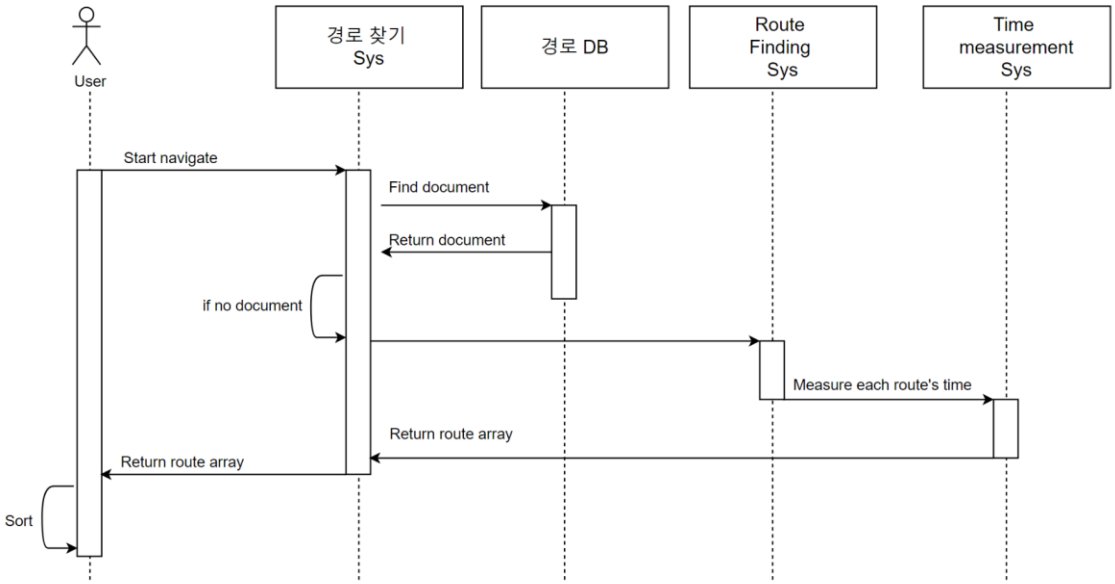
Class diagram – 경로 찾기 system

- Class Description

  ✓ 경로 찾기 handler: 경로 DB 와 다른 서비스를 호출하는 제어자로 활용된다.

유저가 경로를 찾기 위해 출발지와 목적지 그리고 공사 우회 또는 엘리베이터 포함 등의 옵션을 선택하는 것으로 경로 찾기가 시작된다.

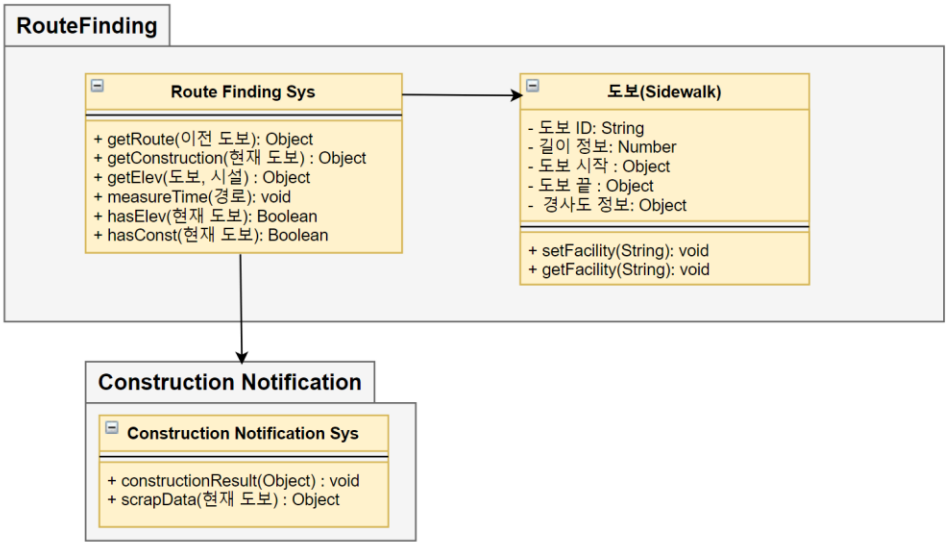
B.Sequence Diagram



Sequence Diagram . 경로 찾기 시스템

5.3.2. Route Finding System

A. Class Diagram

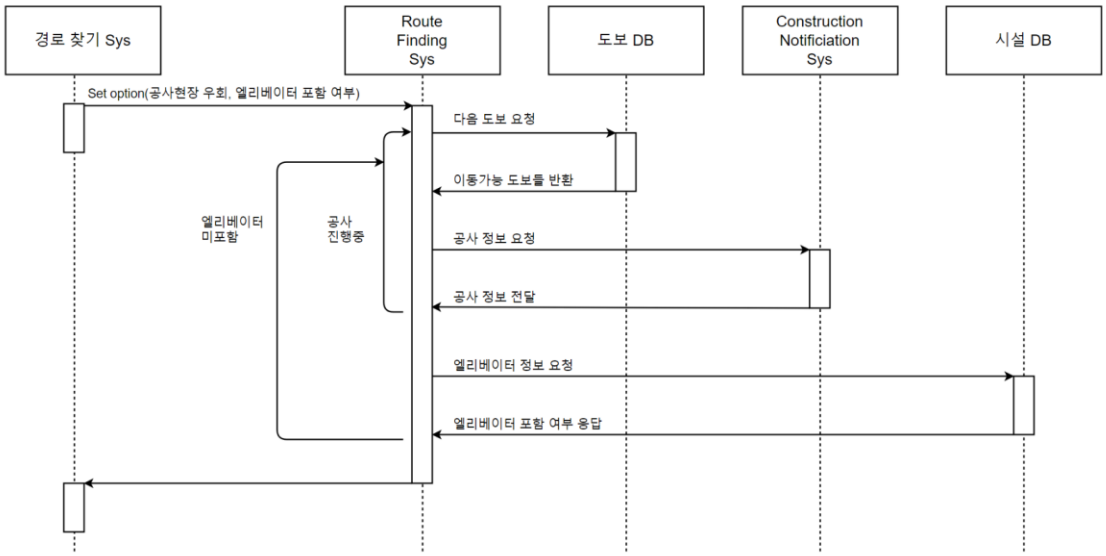


Class Diagram. Route finding system

- Class Description

✓ Route Finding System: 출발지와 목적지 그리고 옵션 사항을 토대로 이동 가능한 경로들을 현재 도보 기준 이동 가능한 다음 도보를 찾는 방식으로 진행된다.

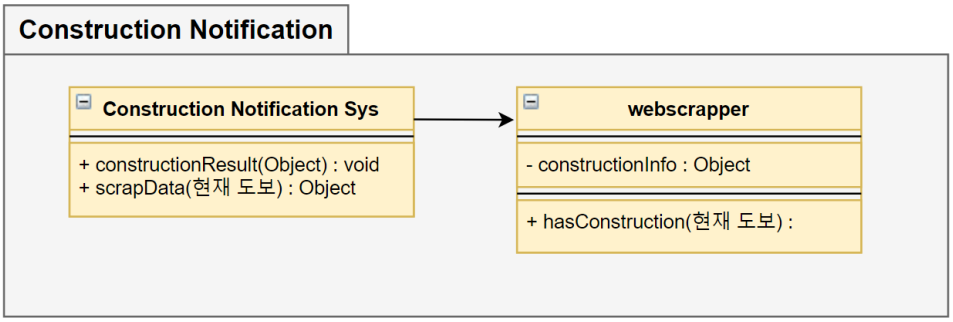
B.Sequence Diagram



Sequence Diagram . Route Finding 시스템

5.3.3. Construction Notification System

A.Class Diagram

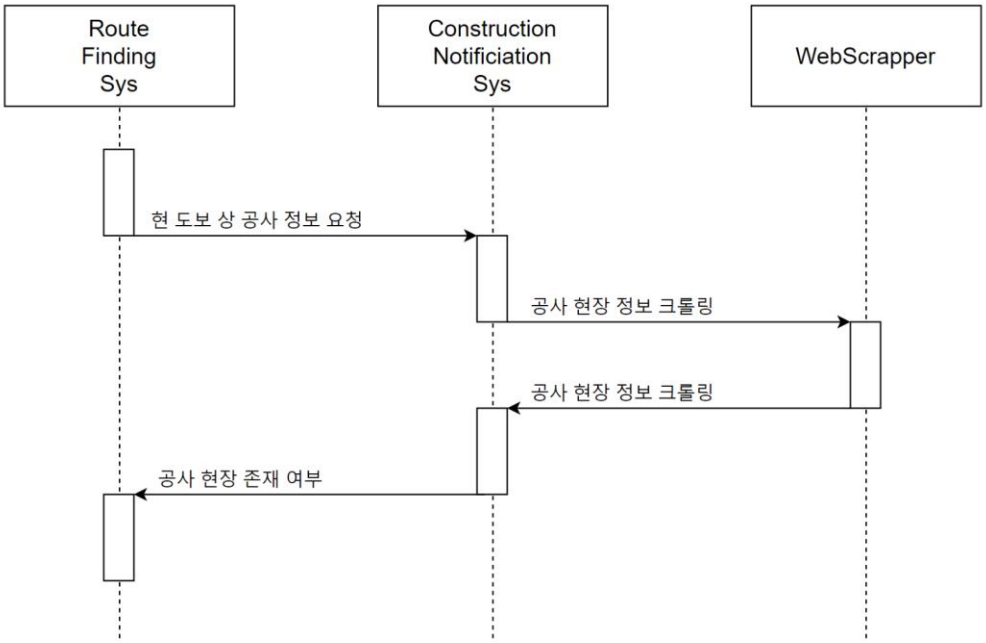


Class Diagram. Construction Notification System

- Class Description

✓ Construction notification system: 다음 도보를 찾고 있는 현재 도보를 기준으로 '건설알림이' 사이트에서 진행 중인 공사의 정보를 수집해온다.

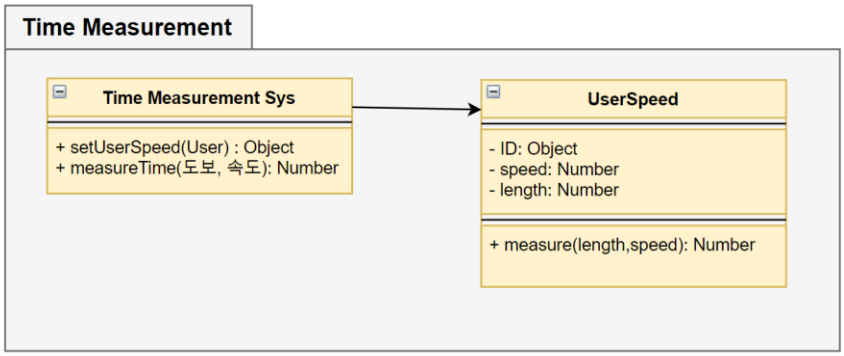
B.Sequence Diagram



Sequence Diagram .Construction notification system

5.3.4. Time measurement System

A.Class Diagram

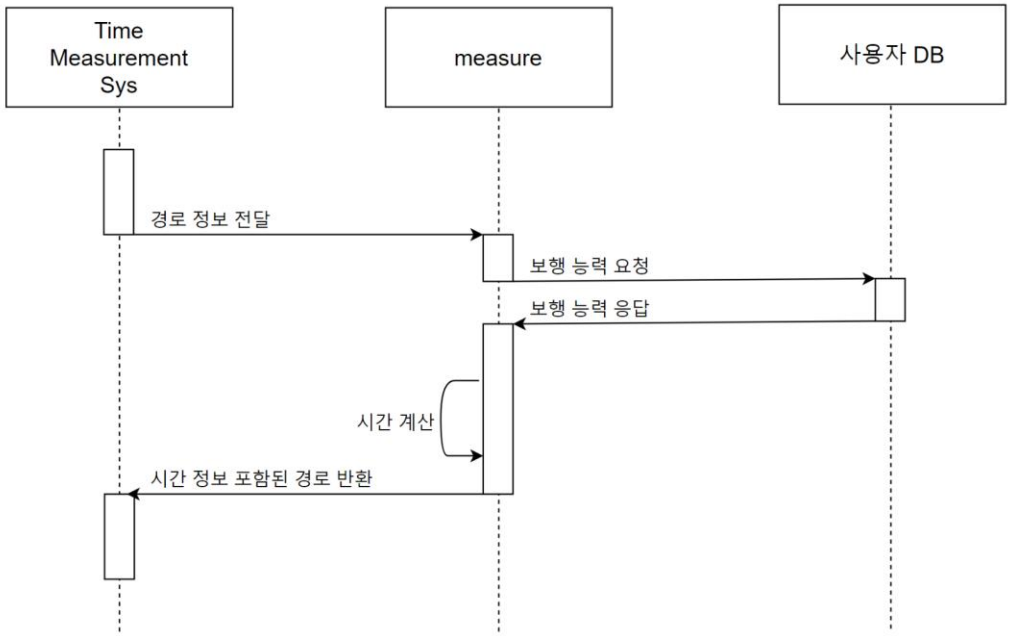


Class Diagram.Time measurement system

- Class Description

✓ Time measurement system: 탐색된 경로를 유저의 이동 수행능력을 토대로 유저에게 맞춤으로 이동 시간을 측정해준다.

B.Sequence Diagram



Sequence Diagram . Time measurement system

6. Protocol Design

6.1 Objectives

「교통약자를 위한 길안내 시스템」의 각 서브시스템 간 상호작용, 특히 프론트엔드 시스템과 백엔드 애플리케이션 서버 시스템 간 상호작용에 이용되는 프로토콜에 어떤 구조가 사용되는지 설명한다. 또한, 각 인터페이스와 프로토콜이 어떻게 정의되어 있는지를 기술한다.

6.2 REST API

본 시스템은 프론트엔드와 백엔드 사이의 통신에 웹 인터페이스, 즉 HTTP 를 이용하며, 요청과 응답 형식은 REST(Representational State Transfer) API 에 따른다. REST API 란 서버에 저장되어 있는 각 자원을 이름으로 구분하여 해당 자원의 상태를 주고받는 API 의 설계 형식을 의미한다. REST API 는 크게 다음 세 부분으로 구성되어 있다.

6.2.1 자원(Resource) : HTTP URI

HTTP URI 를 통해 각 자원을 명시한다. 이는 서버가 보관하고 있는 데이터를 나타낸다.

6.2.2 행위(Verb) : HTTP Method

서버의 자원에 접근해 상태를 조작하기 위한 요청 행위로서, 각 조작 행위는 POST, GET, PUT, DELETE 와 같은 HTTP Method 를 통해 표현된다.

POST	POST 를 통해 해당 URI 를 요청하면 리소스를 생성한다.
GET	GET 을 통해 리소스를 조회한다. 리소스를 조회하고 해당 도큐먼트에 대한 자세한 정보를 가져온다.
PUT	PUT 를 통해 해당 리소스를 수정한다.
DELETE	DELETE 를 통해 리소스를 삭제한다.

6.2.3 표현(Representation) : JSON

클라이언트의 요청에 대한 서버의 응답 형식으로, Interaction 을 위한 프로토콜로 HTTP 프로토콜을 사용할 때 XML 과 JSON 이 주로 사용되는데 단순하면서 확장성이 좋은 JSON 을 사용한다.

클라이언트와 서버 간의 통신에 REST API 를 사용할 경우 서버와 클라이언트 간의 의존성이 줄어들고, 프로토콜이 이해하기 쉬워지는 장점이 있다. 이를 통해 본 시스템에서 개발하는 프론트엔드 클라이언트뿐만 아니라 다른 시스템에서 서버의 자원을 쉽게 이용할 수 있기 때문에 확장성이 높다. 또한, HTTP 프로토콜의 인프라를 그대로 사용하므로 REST API 사용을 위한 별도의 인프라를 구축할 필요가 없다.

6.3 JSON

JSON 은 속성-값 쌍 또는 키-값 쌍으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷이다. 비동기 브라우저/서버 통신 (AJAX)을 위해, 넓게는 XML 을 대체하는 주요 데이터 포맷이다. 특히, 인터넷에서 자료를 주고 받을 때 그 자료를

표현하는 방법으로 알려져 있다. 자료의 종류에 큰 제한은 없으며, 특히 컴퓨터 프로그램의 변수 값을 표현하는 데 적합하다.

Fig. json logo

6.4 OAuth

인터넷 사용자들이 비밀번호를 제공하지 않고 다른 웹사이트 상의 자신들의 정보에 대해 웹사이트나 애플리케이션의 접근 권한을 부여할 수 있는 공통적인 수단으로서 사용되는, 접근 위임을 위한 **개방형 표준**이다. 이 메커니즘은 여러 기업들에 의해 사용되는데, 이를테면 아마존, 구글, 페이스북, 마이크로소프트, 트위터가 있으며 사용자들이 타사 애플리케이션이나 웹사이트의 계정에 관한 정보를 공유할 수 있게 허용한다.

6.5 Details

6.5.1 회원 가입

- Request(요청)

< Table 회원 가입 >

Method	POST	
URL	/auth/signup	
Parameters	user_id	아이디
	user_pw	비밀번호
	user_birth	출생연도
	user_phone	전화번호

	user_mobility	이동보조기구(휠체어, 전동기, 기타) 선택
	user_pregnent	임신 여부
	frist_responder	비상연락처

- Respond(응답)

< Table 회원 가입 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(ID 중복)	
Success Response Body	URL	회원가입인증 URL (PASS 어플)
	URL	권한 설정 URL
	URL	이동 수행 능력 측정 URL
	message	성공적으로 계정이 생성되었습니다.
Failure Response Body	message	Failure 원인에 대한 정보

6.5.2 회원 가입 인증

- Request(요청)

< Table 회원 가입 인증 요청 >

Method	GET	
URL	https://www.passauth.co.kr	
Parameters	통신사	통신사
	user_phone	전화 번호
	보안숫자입력	보안숫자 입력

- Respond(응답)

< Table 회원 가입 인증 응답 >

Success Code	200 OK	
Failure Code	404 Not Found(해당 인증 시간 만료)	
Success Response Body	signup_authorization	회원 가입 허가
Failure Response Body	message	Failure 원인에 대한 정보

6.5.3 이동 수행 능력 측정

- Request(요청)

< Table 이동 수행 능력 측정 >

Method	POST	
URL	/auth/measure-ability	
Parameters	GPS	위치이동 만큼의 이동거리 측정
	time	10 초 계산

- Respond(응답)

< Table 이동 수행 능력 측정 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(GPS 권한 오류)	
Success Response Body	user_ability	이동 수행 능력인 10 초동안 GPS 상에서 움직인 거리를 측정하여 속도 계산
Failure Response Body	message	Failure 원인에 대한 정보

6.5.4 권한 설정

- Request(요청)

< Table 권한 설정 >

Method	POST	
URL	/auth/set	
Parameters	user_set	위치, 마이크, 카메라, 사진, 알림 등 권한 허용 설정

- Respond(응답)

< Table 권한 설정 응답 >

Success Code	200 OK	
Failure Code	401 unauthorized	
Success Response Body	URL	회원가입 URL 또는 마이페이지 URL
Failure Response Body	message	Failure 원인에 대한 정보

6.5.5 로그인

- Request(요청)

< Table 로그인 >

Method	POST	
URL	/auth/login	
Parameters	user_id	아이디
	user_pw	비밀번호

- Respond(응답)

< Table 로그인 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(ID 없음, ID/PW 불일치)	
Success Response Body	User_data object	User 관련 object(user_id, user_pw, user_birth, user_phone, user_mobility, user_pregnent, frist_responder, user_ability, user set)
Failure Response Body	Success	False
	message	Failure 원인에 대한 정보

6.5.6 마이페이지

- Request(요청)

< Table 마이페이지 >

Method	GET	
URL	/mypage/info	
Parameters	-	-

- Respond(응답)

< Table 마이페이지 응답 >

Success Code	200 OK	
Failure Code	404 Not Found(로그인 정보 없음)	
Success Response Body	User_data object	유저의 정보
	URL	즐겨찾기 URL
Failure Response Body	message	Failure 원인에 대한 정보

6.5.7 마이페이지 수정

- Request(요청)

< Table 마이페이지 수정 >

Method	POST	
URL	/mypage/edit	
Parameters	User_data object	해당 유저의 변경 정보

- Respond(응답)

< Table 마이페이지 수정 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request	
Success Response Body	-	-
Failure Response Body	message	Failure 원인에 대한 정보

6.5.8 즐겨찾기

- Request(요청)

< Table 즐겨찾기 >

Method	GET	
URL	/mypage/bookmark	
Parameters	bookmark	경로찾기에서 빈도수가 높은 출발지, 목적지 쌍 제시

- Respond(응답)

< Table 즐겨찾기 응답 >

Success Code	200 OK
Failure Code	404 Not Found(로그인 정보 없음)

Success Response Body	user_bookmark	경로찾기에서 빈도수가 높은 출발지, 목적지 쌍 10 개를 제시하여 원하는 경로 제시
Failure Response Body	message	Failure 원인에 대한 정보

6.5.9 Quick Access

- Request(요청)

< Table Quick Access >

Method	GET	
URL	/quick-access	
Parameters	elevator_info	주변 시설물 엘리베이터 정비 현황
	near_info	근처의 병원, 휠체어 대여소, 수유실, 쉼터, 대중교통 정보

- Respond(응답)

< Table Quick Access 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request	
Success Response Body	URL	주변 시설물 엘리베이터 정비 현황 URL
	URL	주변 시설 정보 URL
Failure Response Body	message	Failure 원인에 대한 정보

6.5.10 주변 시설물 엘리베이터 정비현황

- Request(요청)

< Table 주변 시설물 엘리베이터 정비현황 >

Method	GET

URL	/quick-access/elevator-info	
Parameters	-	-

- Respond(응답)

< Table 주변 시설물 엘리베이터 정비현황 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(인터넷 연결 끊김 or GPS 끊김)	
Success Response Body	elevator_info	시설물 홈페이지에서 얻은 현위치 300m 이내의 시설물의 엘리베이터 위치 정보와 정비 현황
Failure Response Body	message	Failure 원인에 대한 정보

6.5.11 주변 시설물 정보

- Request(요청)

< Table 주변 시설물 정보 >

Method	GET	
URL	/quick-access/near-info	
Parameters	-	-

- Respond(응답)

< Table 주변 시설물 정보 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(인터넷 연결 끊김 or GPS 끊김)	
Success Response Body	near_info	지도 상에서 얻은 현위치 300m 이내의 병원, 휠체어 대여소, 수유실, 쉼터, 대중교통 정보

Failure Response Body	message	Failure 원인에 대한 정보

6.5.12 장소 검색

- Request(요청)

< Table 장소 검색 >

Method	GET	
URL	/place-search	
Parameters	place	정보를 알고자 하는 장소 검색(시설명 or 시설 위치)

- Respond(응답)

< Table 장소 검색 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(인터넷 연결 끊김)	
Success Response Body	place_info	시설명, 시설군, 영업시간, 연락처, 승강기 정비현황 정보
Failure Response Body	message	Failure 원인에 대한 정보

6.5.13 경로 찾기

- Request(요청)

< Table 경로 찾기 >

Method	GET	
URL	/navitation	
Parameters	route	출발지, 도착지
	option	승강기 이용 길, 공사현장 우회 길 옵션 선택

- Respond(응답)

< Table 경로 찾기 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(인터넷 연결 끊김)	
Success Response Body	URL	경로 안내 URL
Failure Response Body	message	Failure 원인에 대한 정보

6.5.14 경로 안내

- Request(요청)

< Table 경로 안내 >

Method	GET	
URL	/navitation	
Parameters	route	출발지, 도착지
	option	승강기 이용 길, 공사현장 우회 길 옵션 선택
	slope	길의 경사도
	user_ability	이동 수행 능력

- Respond(응답)

< Table 경로 안내 응답 >

Success Code	200 OK	
Failure Code	400 Bad Request(인터넷 연결 끊김 or GPS 연결 끊김)	
Success Response Body	routh_object	출발지와 목적지 경로를 옵션, 길의 경사도와 회원 이동 수행 능력을 반영하여 최단 시간 경로, 완만한 경로, 최소 환승 경로 3 가지로 길을 안내, 이 중 user 가 경로 선택

Failure Response Body	message	Failure 원인에 대한 정보

7. Database design

7.1 Objectives

이 장에서는 요구사항 명세서에서 작성한 데이터베이스 요구사항에 대하여 실질적인 데이터베이스 설계를 기술한다. ER Diagram 을 통해 개체 간 관계를 보여주고, 개체 속성의 역할을 명시하고, 실제 개발에 대하여 Relational Schema, 데이터 정의와 접근 양식에 대하여 기술한다.

7.2 ER diagram

본 시스템에서는 로그인, 비상연락처, 회원, 경로, 도보, 경사도, 센서, 경로 즐겨찾기, 시설, 주소 총 10 개의 DB entity 가 존재한다. 각각의 entity 는 직사각형의 형태로 표현되고, entity 간 관계는 마름모꼴로 표현한다. 이때 특정 entity 가 다른 entity 와 가지는 1:N, N:N 관계를 표현해주었다. 각 entity 가 가지는 attribute 는 타원형으로 표현되는데, 각 entity 를 식별하는 Primary key 는 그 라벨에 밑줄을 그어 표시하였다.

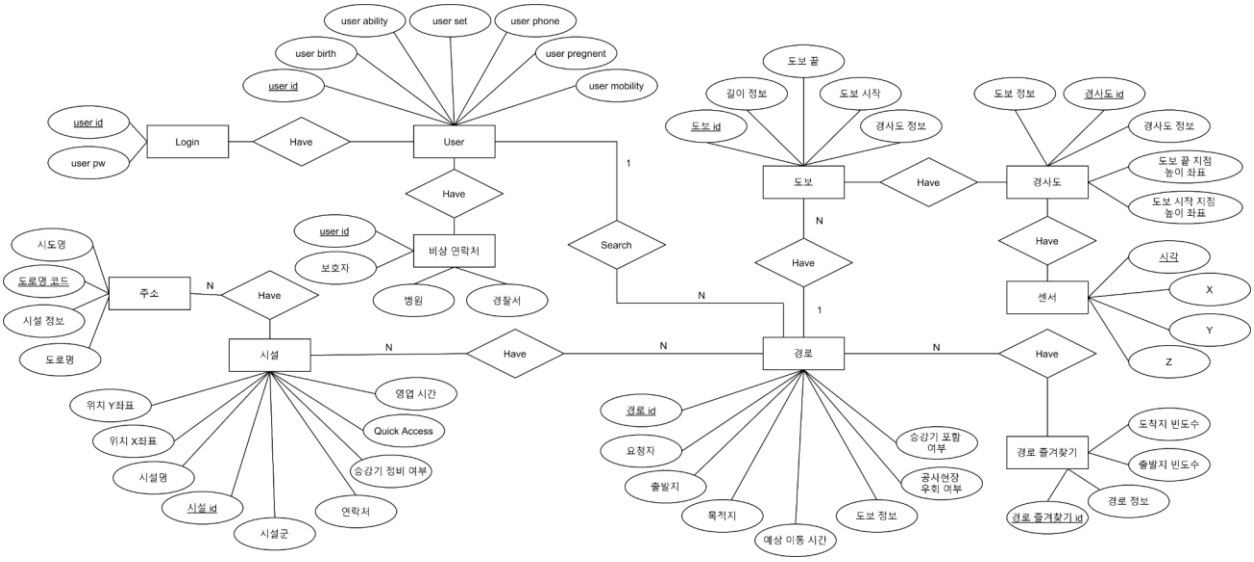


Diagram . ER diagram

7.3 Entities

7.3.1 로그인

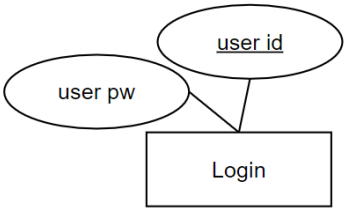


Diagram . 로그인 entity

로그인 entity 는 아이디와 비밀번호로 구성되어 있다. 이때 사용자 고유 식별 ID 인 user id 가 primary key 이다.

7.3.2 회원

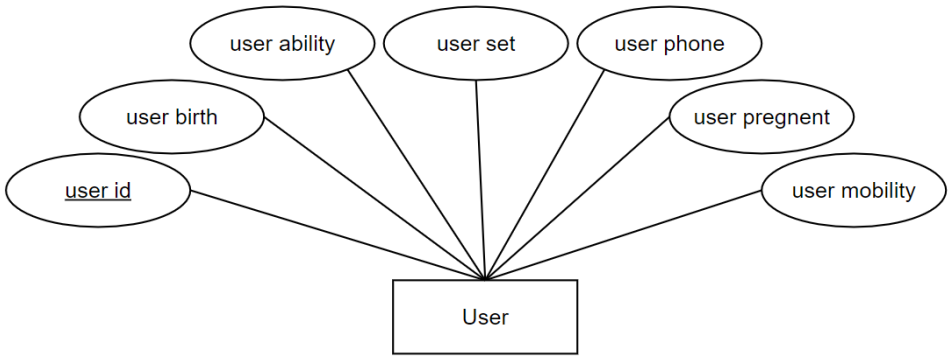


Diagram . 회원 entity

회원 entity 는 회원에 대한 정보를 담고 있다. 회원의 id, 생년월일, 이동수행능력, 권한 설정(위치, 마이크, 카메라, 사진, 알림), 전화번호, 임신여부, 이동보조기구가 이에 속한다. 이때 사용자 고유 식별 ID 인 user id 속성이 primary key 이다.

7.3.3 비상연락처

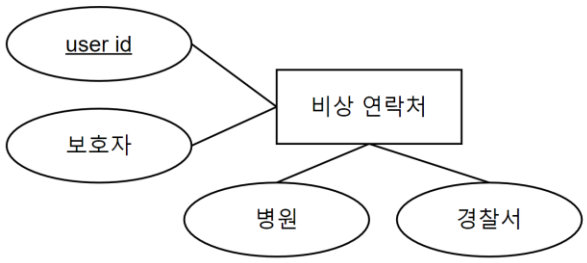


Diagram . 비상연락처 entity

비상연락처 entity 는 회원이 위급한 상황일 때 연락할 수 있는 번호에 대한 정보를 담고 있다. 회원의 id, 보호자/병원/경찰서 전화번호가 포함된다. User_id 는 primary key 로 사용된다.

7.3.4 도보

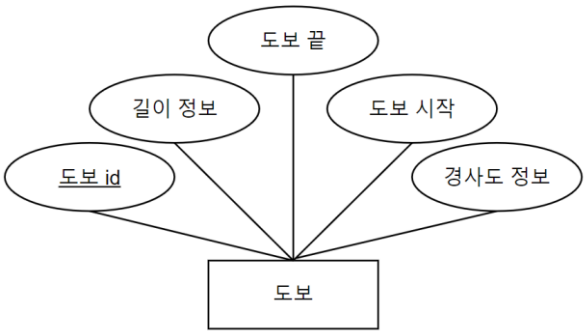


Diagram . 도보 entity

도보 entity 는 경로 찾기를 위한 도보에 관한 정보를 담고 있다. 도보 id, 길이 정보, 도보 끝/시작 지점(X, Y 좌표), 도보 시작 위치에서 끝 위치로 향하는 방향으로의 경사도 정보를 담고 있다. 경로 고유 식별 ID 인 도보 id 가 primary key 이다.

7.3.5 경사도

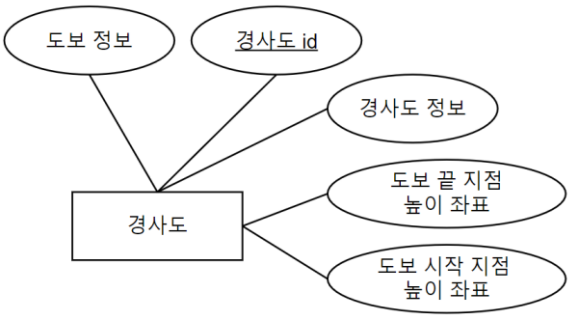


Diagram . 경사도 entity

경사도 entity 는 도보 entity 에 포함될 경사도 정보를 담은 entity 로 도보 정보, 경사도 id, 경사도 정보, 도보 끝 지점 높이 좌표(z), 도보 시작 지점 높이 좌표(z)가 포함된다. 경사도 고유 식별 ID 인 경사도 id 가 primary key 이다.

7.3.6 센서

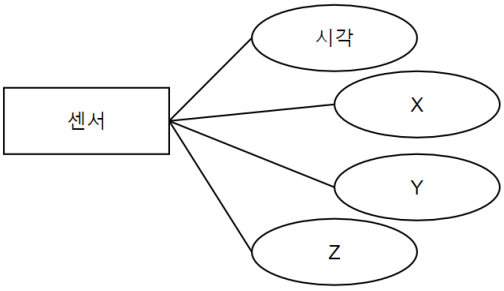


Diagram . 센서 entity

센서 entity 는 경사도를 측정하기 위하여 만들어졌으며 측정한 경사도 정보를 경사도 entity 로 넘겨주기 전에 담고 있다. 디바이스에서 수집한 시각 정보, GPS 상의 x, y 좌표, 3 축 가속계 센서 연산으로 얻은 z(높이) 값이 이에 포함된다. 시각정보가 primary key 이다.

7.3.7 경로 즐겨찾기

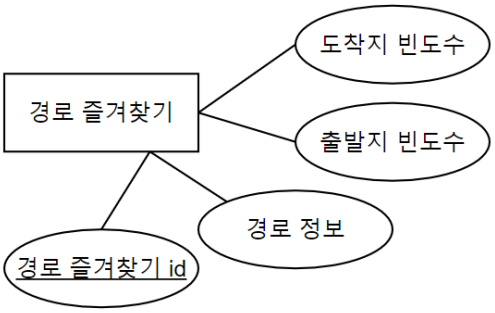


Diagram . 경로 즐겨찾기 entity

경로 즐겨찾기 entity 는 경로 찾기 때 자주 찾은 경로를 즐겨찾기로 나타내기 위한 정보를 담고 있다. 경로 즐겨찾기 id, 도착지/출발지로 설정한 빈도수, 경로 정보가 이에 포함된다. 경로 즐겨찾기 고유 식별 ID 인 경로 즐겨찾기 id 가 primary key 이다.

7.3.8 경로

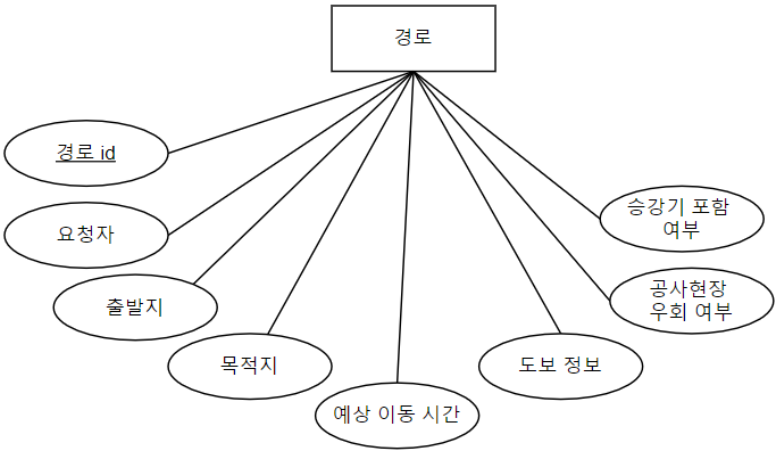


Diagram . 경로 entity

경로 entity 는 회원이 원하는 정보인 출발지에서 도착지까지의 경로를 승강기 포함 여부와 공사현장 우회 여부, 회원의 이동수행능력을 고려하여 추천하기 위한 정보를 담고 있다. 결과값으로 경로와 예상 이동 시간을 내보낸다. 따라서, 경로 id, 요청자, 출발지 좌표, 목적지 좌표, 예상 이동 시간, 도보 정보, 공사현장 우회 여부(True or False), 승강기 포함 여부(True or False)를 포함한다. 경로 고유 식별 ID 인 경로 id 가 primary key 이다.

7.3.9 시설

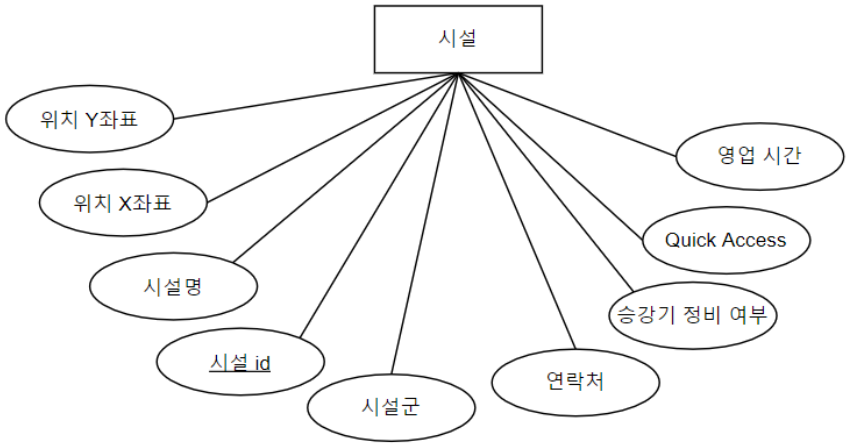


Diagram . 시설 entity

시설 entity 는 Quick Access 나 장소 검색 시 얻는 시설 정보를 담고 있다. 시설 id, 위치 x, y 좌표, 시설명, 시설군, 연락처, 영업시간, 승강기 정비 여부(True or False), Quick Access(True or False)가 포함된다. 이때 주변 시설 정보들이 반환됐을 때 고유 식별 번호인 시설 id 가 primary key 이다.

7.3.10 주소

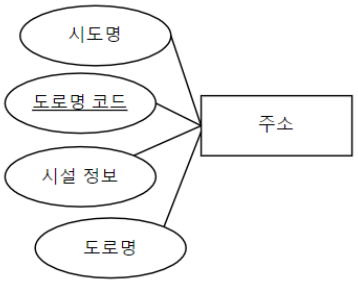


Diagram . 주소 entity

주소 entity 는 시설 entity 를 위한 주소 정보를 담기 위한 entity 로 도로명 코드, 해당 주소가 속한 지도명, 해당 주소의 도로명, 시설정보가 포함된다. 이때 주소 체계상 고유 도로명인 도로명 코드가 primary key 이다.

7.4 Relational Schema

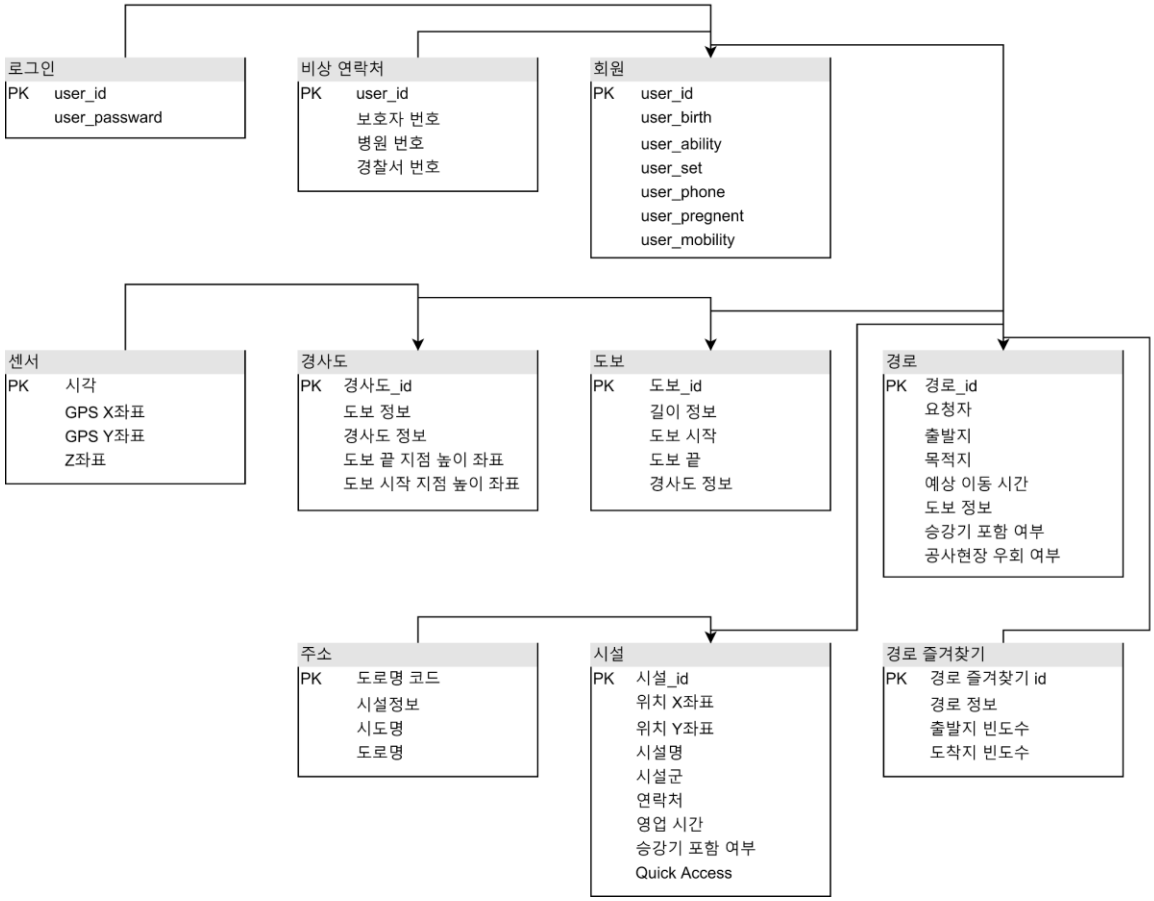


Diagram . Relational schema

7.5. SQL DDL

NoSql 데이터베이스인 몽고 DB 를 활용하기 때문에 각 다큐먼트들의 정보를 기존의 SQL DDL 형식으로 표현하는 것으로 대체하였다. 또한, 데이터 타입 또한 몽고 DB 의 타입들을 사용하였다.

7.5.1 로그인

```
create table 로그인 (
    user_id          String,
    user_password    String,
)
```

7.5.2. 비상 연락처

```
create table 비상_연락처 (
    user_id          String NOT NULL,
    보호자 번호       String NOT NULL,
    병원 번호         String NOT NULL,
    경찰서 번호       String NOT NULL
)
```

7.5.3. 회원

```
create table 회원 (
    user_id          String NOT NULL,
    user_birth       String NOT NULL,
    user_ability     String NOT NULL,
    user_set         String NOT NULL,
    user_phone       String NOT NULL,
    user_pregnent    String,
    user_mobility    String NOT NULL,
)
```

7.5.4. 센서

```
create table 센서 (
    시각             String NOT NULL,
    GPS X 좌표       String NOT NULL,
```

```
    GPS Y 좌표        String NOT NULL,
    Z 좌표            String NOT NULL,
)
```

7.5.5. 경사도

```
create table 경사도 (
    경사도_id                  String NOT NULL,
    도보 정보                  String,
    경사도 정보                 String,
    도보 끝 지점 높이 좌표       String,
    도보 시작 지점 높이 좌표      String,
)
```

7.5.6. 도보

```
create table 도보 (
    도보_id          String NOT NULL,
    길이 정보        String NOT NULL,
    도보 시작        String NOT NULL,
    도보 끝          String NOT NULL,
    경사도 정보      String
)
```

7.5.7. 경로

```
create table 경로 (
    경로_id              String NOT NULL,
    요청자               String NOT NULL,
    출발지               String NOT NULL,
    목적지               String NOT NULL,
    예상 이동 시간        String NOT NULL,
    도보 정보            Object NOT NULL,
    승강기 포함 여부      Boolean,
    공사현장 우회 여부    Boolean,
)
```

7.5.8. 주소

```
create table 주소 (
    도로명 코드          String NOT NULL,
    시설정보             Object NOT NULL,
    시도명               String NOT NULL,
    도로명               String NOT NULL
)
```

7.5.9. 시설

```
create table 시설 (
    시설_id              String NOT NULL,
    위치 X 좌표          String NOT NULL,
    위치 Y 좌표          String NOT NULL,
    시설명               String NOT NULL,
    시설군               String NOT NULL,
    연락처               String,
    영업 시간            String,
    승강기 포함 여부     Boolean NOT NULL,
    Quick Access         Boolean NOT NULL,
)
```

7.5.10. 경로_즐겨찾기

```
create table 경로_즐겨찾기 (
    경로 즐겨찾기 id    String NOT NULL,
    경로 정보           String NOT NULL,
    출발지 빈도수       String,
    도착지 빈도수       String
)
```

8. Testing Plan

8.1. Objectives

이 장에서는 개발 과정 및 배포 이후의 유지 보수 및 시스템 성능과 안정성을 위한 테스트와 사용자로부터의 사용성 검증을 위한 계획을 기술한다. 이는 사용자의 요구사항에 맞춰 각 시스템을 개발할 때 뿐만 아니라 시스템을 통합하는 과정에서 발생할 수 있는 잠재적인 오류와 결함을 사전에 인지하고 해결하도록 하며 배포 이후에도 안정적인 운영과 효율적인 유지보수를 목표로한다.

8.2. Testing Policy

8.2.1. Development Testing

Development testing 이란 시스템을 개발하는 과정이나 실제 배포 시 발생할 문제를 찾아내고 해결하는 과정이다. 이 단계에서는 개발 과정에서 소프트웨어가 충분한 테스트를 거치지 않았기 때문에 시스템 개발 및 통합 과정에서 안정성에 문제가 있을 수 있으며 구성 요소가 서로 충돌 할 수 있다. 이를 해결하기 위해서는 1) 성능, 2) 안정성, 3) 보안과 같은 비기능적 요소들을 통해 검증한다.

A. Performance

경로 추천 시스템이 가장 많은 시간이 소요되는 작업이기 때문에 추천 경로를 필터링하고 사용자에게 제시하는 시간을 단축하는 것이 중요하다. 다양한 이동 수행 능력과 경로에 대한 테스트 케이스를 준비하고 추천 기능 속도를 평가하고 서버와의 통신을 개선한다.

B. Reliability

시스템이 안정적으로 작동하기 위해서 시스템을 구성하는 하위 구성 요소와 서브시스템이 잘 작동하고 안정적으로 연결되어야 한다. 그러므로 유닛 개발 단계부터 개발 테스트를 거치고 유닛을 통합하는 단계에서도 오류를 확인해야 한다.

C. Security

시스템의 정보 보안은 개발자가 가장 신경 써야할 중요한 문제 중 하나다. 정보의 가치에 관계 없이 원치 않는 방문자로부터 정보를 보호해야 한다.

시스템의 보안을 완성하기 위해서는 거의 마무리된 버전의 앱에서 접근해야 보안 문제를 식별할 수 있고 코드 검토를 통해 보고서를 작성할 수 있다.

8.2.2. Release Testing

Release testing 이란 시스템 또는 서브 시스템의 새로운 버전이 출시되어 적용시켜야 할 때 거쳐야 할 테스트로 새로운 버전이 기존의 시스템과 결합 되어도 문제가 발생하지 않고, 요구사항을 잘 충족하는지 확인하는 과정이다. 일반적으로 High level 인 application level 에서 이루어진다. 하지만 결함이 발생했을 경우 이를 해결하기 위해 low level 인 component 를 수정해야 한다.

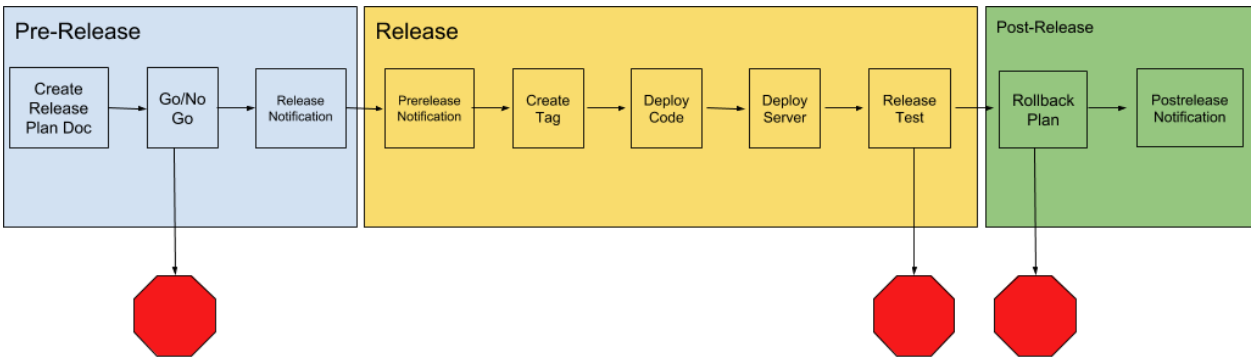


Fig . release test

Release test 는 fig 와 같이 3 단계에 걸쳐 이루어진다. 이때 주의할 점은 사용자들이 시스템 업데이트를 해야한다는 것이다. 강제로 업데이트 시킬지 아니면 버전 별로 다르게 관리할 것인지에 대해 결정해야 한다.

8.2.3. User Testing

User testing 이란 비 기능적 요구사항과 사용자의 기능적 요구사항을 충족하였는지 확인한다. 단, 본 시스템의 개발에 있어서 실제 유저의 테스트를 적용시키기는 어려우므로 필수적인 사용자 테스트를 진행할 수 있는 가능한 시나리오와 현실적인 상황을 설정해야 한다. 우리는 20 명의 어플리케이션 사용자가 있다고 가정한다. 그리고 Mobility Mate 어플리케이션 베타 버전을 배포하고 사용자의 후기를 모은다.

8.2.4. Testing Case

Testing case 는 3 가지 기본적인 기능, 성능, 보안의 측면을 따라 설정한다. 우리는 각 측면에 따라 5 가지 테스트 케이스를 준비할 것이고 Mobility Mate 어플리케이션에 대한 테스트를 진행하고 평가지를 작성할것이다.

9. Developing Plan

9.1 Objectives

이 장에서는 애플리케이션 개발을 위해 요구되는 기술 및 환경을 설명한다.

9.2 Frontend Environment

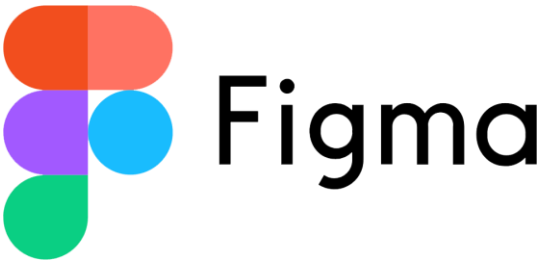
9.2.1. Adobe Illustrator

어도비 일러스트레이터(Adobe Illustrator)는 어도비 시스템즈사에서 개발한 벡터 드로잉 프로그램이다. 맥 OS 와 윈도 플랫폼을 지원한다. 파일의 용량이 적고 벡터방식이기 때문에 그림을 확대해도 선명하다는 장점이 있다. 앱 아이콘, 로고 등 BI 제작 시 사용된다.

9.2.2. Adobe Photoshop

어도비 포토샵(Adobe Photoshop)은 미국의 어도비 시스템즈사에서 개발한 레스터 그래픽 편집기이다. 맥 OS 와 윈도 플랫폼을 지원한다. 픽셀을 기본단위로 하는 비트맵방식의 툴이다. 컴포넌트의 채도 및 명도 조절 및 색조합에 사용된다.

9.2.3. Figma



디자인 설계 및 유지보수에 편리한 UI/UX 제작 웹 기반 프로토타입 툴이다. 웹 기반이기 때문에 피그마 계정 소유자가 링크를 공유하여 여러 명이 아트 보드를 확인하며 동시에 온라인으로 실시간 작업을 진행할 수 있다. 또한 디자이너와 개발자의 소통을 도와주는 기능이 많아 개발자가 참조할 수 있는 정보를 주는 '개발 툴바'가 있으며, 마우스 툴바로도 수치값을 확인할 수 있어 가이드라인이 없어도 빠른 작업이 가능하다. 내보내기 기능을 이용해 다른 형식의 파일로 변환하여 활용이 가능하다.

9.2.4. Zeplin

어플리케이션을 디자인하고 나서 해당 디자인을 어떻게 개발에 적용해야할지를 개발단계에서 손쉽게 확인할 수 있도록 도와주는 핸드오프 툴이다. 더 명확하고 손쉬운 프론트엔드 화면 구축을 위해 컴포넌트 이미지의 크기, 위치,.색, 폰트 정보 등 다양한 정보를 정리하여 전달한다.

9.2.5. Android Studio

애플리케이션을 외부 API 에 연결하여 추가 기능을 추가하고 XML 파일을 애플리케이션의 활동에 연결하기 위해 사용한다. 또한 데이터 관리를 위해 애플리케이션을 DB 서버에 연결한다.

9.3 Backend Environment

9.3.1. Github

깃허브(GitHub)는 분산 버전 관리 툴인 깃 (Git)을 사용하는 프로젝트를 지원하는 웹호스팅 서비스이다. GitHub 는 영리적인 서비스와 오픈소스를 위한 무상 서비스를 모두 제공한다. 깃(Git)이 텍스트 명령어 입력 방식인데 반해, 깃허브는 화려한 그래픽 유저 인터페이스(GUI)를 제공한다. 깃허브는 페이스트빈(pastebin)과 유사한 서비스인 Gist 와 위키를 각 저장소마다 운영하고 있으며, 깃 저장소를 통해 고칠 수 있다. 팀원들과 단일 프로젝트를 함께 개발할 수 있기 위해 Github 를 활용하여 코드를 남기고 리뷰를 통해 이슈 등을 게재하며 협업 과정에서 효율적인 소통을 도모한다.

9.3.2. mongoDB

몽고 DB(MongoDB)는 크로스 플랫폼 **도큐먼트 지향 데이터베이스** 시스템이며 가장 유명한 NoSQL 데이터베이스 시스템이다. JSON 과 같은 동적 스키마형 도큐먼트들을 선호함에 따라 전통적인 테이블 기반 **관계형 데이터베이스** 구조의 사용을 삼간다. 이로써 특정한 종류의 애플리케이션을 더 쉽고 더 빠르게 데이터 통합을 가능케 한다. **크레이그리스트, 이베이, 포스퀘어,**

소스포지, 뉴욕 타임즈, 구글, 페이스북과 같은 수많은 주요 웹사이트 및 서비스에 백엔드 소프트웨어로 채택되고 있다.

9.3.3. Android Studio

프론트 개발 환경과 동일하게 안드로이드 스튜디오를 활용하여 IDE 를 통일시킨다.

9.3.4. Docker

독립적으로 실행되는 각각의 서비스들을 원활하게 운용하기 위해 호스트의 OS 를 가상화한 컨테이너라는 격리된 공간을 만들 수 있는 도커(Docker)를 활용한다. 이는 호스트의 OS 를 할당받는 가상 머신(Virtual Machine)보다 성능과 리소스 소비 차원에서 우위를 가진다.

9.3.5. Kubernetes

쿠버네티스(Kubernetes)는 컨테이너화된 애플리케이션의 자동 디플로이, 스케일링 등을 제공하는 관리시스템으로, 오픈 소스 기반이다. 원래 구글에 의해 설계되었고 현재 리눅스 재단에 의해 관리되고 있다. 목적은 여러 클러스터의 호스트 간에 애플리케이션 컨테이너의 배치, 스케일링, 운영을 자동화하기 위한 플랫폼을 제공하기 위함이다. 도커를 포함하여 일련의 컨테이너 도구들과 함께 동작한다.

9.4 Constraints

- 경로 선택지가 최소 3 개는 필요하다
- 경로를 찾는 시간이 5 초를 초과하여서는 안된다.
- 시스템 관리 도구는 한 명의 관리자만 지원한다.
- 시스템은 각 모바일 장치에 대해 한 명의 동시 사용자만 지원한다. 시스템은 동일한 장치에서 다중 연결을 지원하지 않는다. 단, 연결 해제 후 사용자는 계정 및 액세스를 전환 할 수 있다.
- - 시스템은 최소 1GB RAM 및 1.0GHz 단일 프로세서가있는 모바일 장치에서 원활하게 실행되어야한다. 그리고 시스템은 Android 7.1 이상의 최신 버전을 지원한다.
- 시스템은 최소 8000 명의 동시 사용자를 위해 원활하게 실행된다. 그리고 시스템은 최소 100,000 개의 활성 사용자 계정과 프로필을 처리 할 수 있다.
- 애플리케이션은 5 초 이내에 실행되어야 한다.
- 각 계정은 연결 후 2 초 이내에 활성화되어야 한다.
- 로그인은 3 초 이내에 완료된다.
- 장소 검색 결과는 1 초 이내에 표시된다.
- 프로필 입력시 데이터는 1 초 이내에 데이터베이스에 저장되어야하며, 데이터베이스 업데이트는 2 초 이내에 완료되어야 한다.
- 사용자 로그는 2 초 이내에 데이터베이스에서 불러올 수 있어야 하며 1 초 이내에 삭제되어야 한다. 변경된 데이터베이스 업데이트는 2 초 이내에 완료되어야 한다.
- 사용자가 사용하기 편한 동작과 인터페이스를 고려해야 한다.

9.5 Assumptions and Dependencies

해당 문서의 모든 시스템은 Android 기기 및 오픈 소스를 기반으로 설계 및 구현된다는 가정하에 작성되었다. 따라서 모든 콘텐츠는 최소 API 버전 23 의 Android 운영 체제를 기반으로 작성되었으며 다른 운영 체제 또는 버전에는 적용되지 않을 수 있다.

10. Supporting Information

10.1 Software Requirement Specification

이 소프트웨어 설계 사양은 IEEE 권장 사항(IEEE Recommended Practice for Software Design Description, IEEE-Std-1016)에 따라 작성되었다.

10.2 Document History

<Table > Document History

Date	Version	Description	Writer
2021/05/10	0.1	Title and Content	이주민
2021/05/11	1.0	Addition of 1.1, 1.2, 1.3, 1.4	우준하
2021/05/12	1.1	Addition of 2.1, 2.2, 2.4, 4.1	우준하
2021/05/13	1.2	Addition of 6.1, 6.2, 6.3, 6.4	이주민
2021/05/13	1.3	Addition of 3.1, 3.2	우준하
2021/05/14	1.4	Addition of 6.5	이주민
2021/05/14	1.5	Addition of 5.1, 5.2	이석현
2021/05/15	1.6	Addition of 7.1, 7.2, 7.3	이주민
2021/05/15	1.7	Addition of 5.3	이석현
2021/05/16	1.8	Addition of 3.2	안리아
2021/05/16	1.9	Addition of 7.4	이주민
2021/05/16	2.0	Addition of 7.5	이석현
2021/05/16	2.1	Addition of 4.2	안리아

2021/05/16	2.2	Addition of 8.1, 8.2.1, 8.2.4	우준하
2021/05/16	2.3	Addition of 8.2.3	이주민
2021/05/16	2.4	Addition of 10.1, 9.1, 9.2, 9.4, 9.5	안리아
2021/05/16	2.5	Addition of 9.3	이석현
2021/05/16	최종	Table of Figure,Table & 검수	이석현