소프트웨어공학 F조 제안서



|  |  |
| --- | --- |
| 과목 명 | 소프트웨어 공학 |
| 교수 명 | 강동현 교수님 |
| 조 이름 | F조 |
| 조 원 | 강병곤, 권철현, 김지현, 신재하, 안성민 |
| 개발 프로그램 | Tag The Place |
| 프로그램 타입 | 안드로이드 어플리케이션 |
| 제출 날짜 | 2021.11.14 |

목차

[1. 서문 1](#_Toc88079962)

[1.1 어플리케이션 설계의 목적 1](#_Toc88079963)

[1.2 유사한 어플리케이션 1](#_Toc88079964)

[1.3 어플리케이션의 기대 효과 2](#_Toc88079965)

[1.4 예상하는 사용자층 2](#_Toc88079966)

[2. 도입 2](#_Toc88079967)

[2.1 시스템의 필요성 2](#_Toc88079968)

[2.2 시스템의 기능 3](#_Toc88079969)

[3. 용어 사전 3](#_Toc88079970)

[4. 사용자 요구사항 정의 4](#_Toc88079971)

[4.1 준수해야 하는 프로세스 4](#_Toc88079972)

[4.2 유스케이스 다이어그램 4](#_Toc88079973)

[4.3 유스케이스 상세 명세 4](#_Toc88079974)

[5. 시스템 아키텍처 7](#_Toc88079975)

[5.1 클래스 다이어그램 7](#_Toc88079976)

[5.2 클래스 다이어그램 명세 8](#_Toc88079977)

[6. 시스템 요구사항 정의 11](#_Toc88079978)

[6.1 기능적 요구사항 11](#_Toc88079979)

[6.2 비기능적 요구사항 12](#_Toc88079980)

[7. 시스템 모델 13](#_Toc88079981)

[7.1 시퀀스 다이어그램 13](#_Toc88079982)

[8. 시스템 진화 13](#_Toc88079983)

[9. 부록 13](#_Toc88079984)

[9.1 서버 개설 및 데이터베이스 생성 13](#_Toc88079985)

[9.2 구글 클라우드: Maps SDK for Android 13](#_Toc88079986)

[10. 색인 13](#_Toc88079987)

**그림 목차**

[Figure 1: 유사한 어플리케이션 DATE POP 1](file:///C:\Study\3rd_year_2nd_semester\SE\소프트웨어공학%20F조%20제안서.docx#_Toc88079812)

[Figure 2: 유스케이스 다이어그램 4](#_Toc88079813)

[Figure 3: 클래스 다이어그램 8](file:///C:\Study\3rd_year_2nd_semester\SE\소프트웨어공학%20F조%20제안서.docx#_Toc88079814)

[Figure 4: Tag The Place 실행화면 1 12](file:///C:\Study\3rd_year_2nd_semester\SE\소프트웨어공학%20F조%20제안서.docx#_Toc88079815)

[Figure 5: Tag The Place 실행화면 2 12](file:///C:\Study\3rd_year_2nd_semester\SE\소프트웨어공학%20F조%20제안서.docx#_Toc88079816)

**표 목차**

[Table 1: U1. 위치권한 가져오기 4](#_Toc88079872)

[Table 2: U2. 지도 표시 5](#_Toc88079873)

[Table 3: U3. 장소 검색 5](#_Toc88079874)

[Table 4: U4. 사용자 위치 보여주기 5](#_Toc88079875)

[Table 5: U5. 마커 보여주기 6](#_Toc88079876)

[Table 6: U6. 마커 추가하기 6](#_Toc88079877)

[Table 7: U7. 상세정보 표시 6](#_Toc88079878)

[Table 8: U8. 태그 평가하기 7](#_Toc88079879)

[Table 9: U9. 사용자 위치 찾기 7](#_Toc88079880)

[Table 10: MainActivity 명세 8](#_Toc88079881)

[Table 11: OnMapReadyCallback 명세 9](#_Toc88079882)

[Table 12: OnInfoWindowClickListener 명세 9](#_Toc88079883)

[Table 13: PlaceData 명세 10](#_Toc88079884)

[Table 14: PlaceRepository 명세 10](#_Toc88079885)

# 1. 서문

## 1.1 어플리케이션 설계의 목적

스마트폰의 보급률 증가로 인해 SNS(Social Networking Service)의 사용량이 점차 증가하는 추세이다. 특히 여가생활 등을 통해 삶의 질을 향상시키기 위한 움직임이 더해지면서 특별한 활동을 할 수 있는 장소, 맛있는 식사를 할 수 있는 장소 등에 대한 사람들의 관심이 높아졌다.

이와 같은 특별한 장소에 관심이 있는 사용자는 장소에 대한 정보를 얻기 위해 포털 사이트와 SNS 등에서 목적에 맞는 장소에 대한 키워드를 검색해야 한다. 이 과정에서 사용자는 얻은 정보가 광고성 정보인지 그리고 실제 목적과 부합하는지 등의 신뢰성을 판단해야 한다. 사용자가 목적에 맞는 정보를 얻은 이후에는 해당 장소의 위치와 경로를 찾기 위해 지도에서 다시 한번 장소를 검색하는 과정을 거쳐야 한다.

이러한 과정을 개선하기 위해, 본 프로젝트에서는 지도를 기반으로 정보를 제공하는 장소 검색 어플리케이션 ‘Tag The Place’을 설계한다. 지도에서 검색을 진행하는 방식을 통해 검색 과정을 줄이고, 사용자들의 평가를 직관적으로 확인할 수 있게 하여 정보의 신뢰성을 높이고자 한다.

## 1.2 유사한 어플리케이션

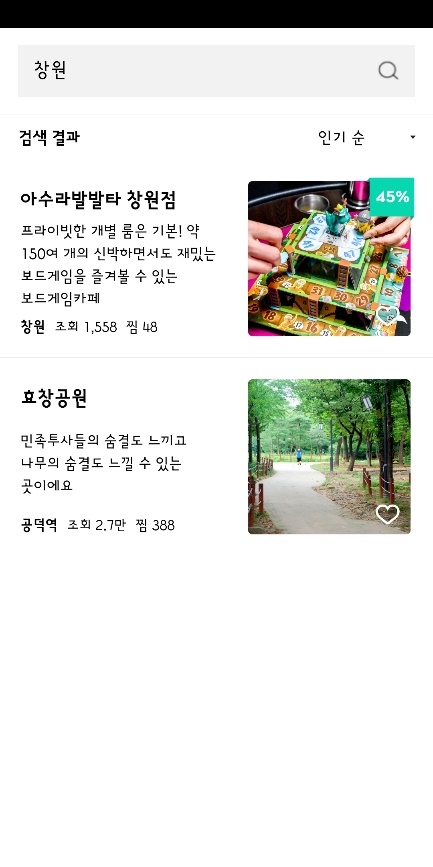


Figure : 유사한 어플리케이션 DATE POP

제안하는 프로젝트와 유사한 어플리케이션으로 ‘DATE POP’이 있다. 이는 인근에서 데이트 장소를 추천해주는 어플리케이션이다. **Figure 1**은 해당 어플리케이션의 실행 화면이다. **Figure1**의 좌측이 어플리케이션의 기본 화면, 우측이 검색 화면이다. ‘DATE POP’은 장소에 대한 검색이 가능하고, 해당 장소의 위치를 알려준다. 또, 사용자들의 후기를 통해 정보의 신뢰성을 나타낸다. 이 기능들은 제안하는 프로젝트와 유사하다. 하지만, 사용자가 직접 장소 정보를 공유할 수 없고, 장소 검색 결과를 지도에서 보여주지 않는다는 아쉬움이 존재한다.

따라서 제안하는 프로젝트에서는 해당 기능들을 보완하기 위해 사용자가 직접 장소 정보를 추가할 수 있는 기능을 포함하고, 검색 기능을 지도 기반으로 제작할 예정이다. 또, 정보의 신뢰성 판단을 위한 후기 기능을 사용자가 더 쉽고 직관적으로 판단할 수 있게 ‘좋아요’와 ‘싫어요’ 기능을 통해 정보를 평가하고 판단할 수 있도록 개선하려고 한다.

## 1.3 어플리케이션의 기대 효과

본 어플리케이션을 이용하는 사용자가 많아질 경우, 손쉽게 장소 정보를 공유할 수 있는 기능을 통해 전국 곳곳의 숨겨진 지역의 명소들에 대한 관심이 증가할 수 있다. 이로 인한 지역 관광 산업의 발달 효과를 기대할 수 있다.

## 1.4 예상하는 사용자층

본 프로젝트는 SNS 사용이 활발하고 활동하는 반경이 넓은 20~30대가 많은 사용을 보일 것이라고 예상한다. 또, 정년퇴직 이후에 삶의 여유를 즐기고 싶어하는 60대 이상 연령층에서의 많은 사용을 기대하고 있다.

# 2. 도입

## 2.1 시스템의 필요성

사회적으로 SNS나 블로그 등을 이용한 [바이럴 마케팅(viral marketing)](#_3._용어_사전)이 활성화되고 있다. 때문에 정보를 얻기 원하는 사용자가 정보의 신뢰성을 판단하기 어려워졌다. 특히 돈을 받고 후기를 체계적으로 조작하는 업체까지 등장하여 더욱 어려움이 커졌다. 따라서 업체를 배제하고 사용자끼리 신뢰성 있는 정보를 공유하고 얻을 수 있는 시스템이 필요하다.

사용자끼리 정보를 얻기 위한 기존의 SNS는 이미 수많은 광고성 정보로 가득하고, 유명 [인플루언서(Influencer)](#_3._용어_사전)들의 사용 후기는 대부분이 바이럴 마케팅일 정도이다. 이는 정보의 신뢰성을 판단할 수 없다는 문제점을 가진다.

이러한 문제점을 개선하기 위하여 본 프로젝트는 사용자의 평가가 또 다른 광고가 될 수 있는 후기 형식이 아닌 ‘좋아요’, ‘싫어요’의 수로 정보의 신뢰성을 판단하게 한다. 이 때, 한 사용자가 하나의 정보에 대하여 한 번의 평가만 실시할 수 있도록 어플리케이션을 설계하여 광고 업체가 평가를 조작하기 어렵게 한다.

## 2.2 시스템의 기능

본 어플리케이션은 크게 세 가지의 기능을 가진다. 이는 다음과 같다.

**• 태그 추가기능**: 사용자가 다른 사람과 공유하고 싶은 장소에 해시태그를 추가하여 저장할 수 있는 기능이다. 지도에서 장소 정보를 공유할 장소를 길게 누르면 정보를 입력할 수 있다.

**• 장소 검색기능**: 해시태그를 검색하면 해당 해시태그를 포함하고 있는 장소들의 마커를 보여주는 기능이다. 마커의 정보 창을 클릭하면 자세한 정보와 장소에 대한 평가를 나타낸다.

**• 태그 평가 기능**: 해시태그를 통해 검색한 장소에 대한 사용자의 평가를 반영할 수 있는 기능이다. 긍정적인 평가를 의미하는 ‘좋아요’와 부정적인 평가를 의미하는 ‘싫어요’ 두 가지 중 한 가지를 선택할 수 있다. 장소에 대한 평가는 ‘좋아요’와 ‘싫어요’의 수를 통해 나타낸다.

# 3. 용어 사전

용어사전에서는 문서에 나타나는 기술적인 용어를 나타낸다. 내용은 다음과 같다.

**• 바이럴 마케팅**: 바이럴은 바이러스(virus)의 형용사형으로 감염되는, 전이되는 등의 뜻을 가진다. 즉 사용자들의 입소문을 통해 마케팅을 하는 것이 바이럴 마케팅이다. 주로 SNS나 블로그에서 사용 후기를 가장해서 제품에 대한 광고를 한다.

**• 인플루언서**: 인플루언서는 타인에게 영향력을 끼치는 사람(Influence + er)이라는 뜻의 신조어이다. 주로 SNS상에서 영향력이 큰 사람들을 일컫는다.

**• API**: Application Programming Interface의 약어, 응용 프로그램에서 사용할 수 있도록, 운영 체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스를 뜻한다. 이 어플리케이션에서는 지도 기능을 제공하는 API를 사용한다.

**• 쿼리:** 정보 수집에 대한 요청에 쓰이는 컴퓨터 언어로, 데이터베이스에서 정보를 가져오기 위해 주로 사용한다.

**• 데이터베이스**: 데이터베이스는 여러 사람이 공유하여 사용할 목적으로 체계화해 통합, 관리하는 데이터의 집합이다.

**• 유스케이스 다이어그램**: 사용자의 기능적 요구를 정의하는 다이어그램이다. 유스케이스(Use Case, 기능)와 액터(Actor, 사용자 혹은 시스템)간의 관계를 표현한다.

**• 클래스 다이어그램**: 시스템의 클래스, 클래스의 속성, 동작 방식, 객체 간 관계를 표시함으로써 시스템의 구조를 기술하는 정적 구조 다이어그램의 일종이다.

**• 컴포넌트**: 컴포넌트(Component)란 프로그래밍에 있어 재사용이 가능한 각각의 독립된 모듈을 뜻한다. 해당 문서 [5.2 클래스 다이어그램 명세에서는](#_5.2_클래스_다이어그램) 구글 지도를 표시할 Fragment 객체를 의미한다.

# 4. 사용자 요구사항 정의

## 4.1 준수해야 하는 프로세스

본 어플리케이션은 안드로이드 환경을 목표로 제작한다. 따라서 언어는 Java를 사용하여 개발해야 한다. 하위 버전의 안드로이드를 사용하는 사용자를 위해 최소 안드로이드 API 버전은 21: Android 5.1(Lollipop)로 설정하고, 컴파일 SDK는 버전 31, 자바 버전은 1.8로 개발해야 한다. GPS 기능을 사용함으로, AVD(Android Virtual Machine) 환경이 아닌 실제 안드로이드 기기에서 동작해야 한다.

## 4.2 유스케이스 다이어그램

이번 단락에서는 본 프로젝트에서 제공할 서비스를 시각적으로 보여주기 위해 [유스케이스 다이어그램(Use Case Diagram)](#_3._용어_사전)을 포함한다. **Figure 2**는 제안하는 어플리케이션의 유스케이스 다이어그램이다.

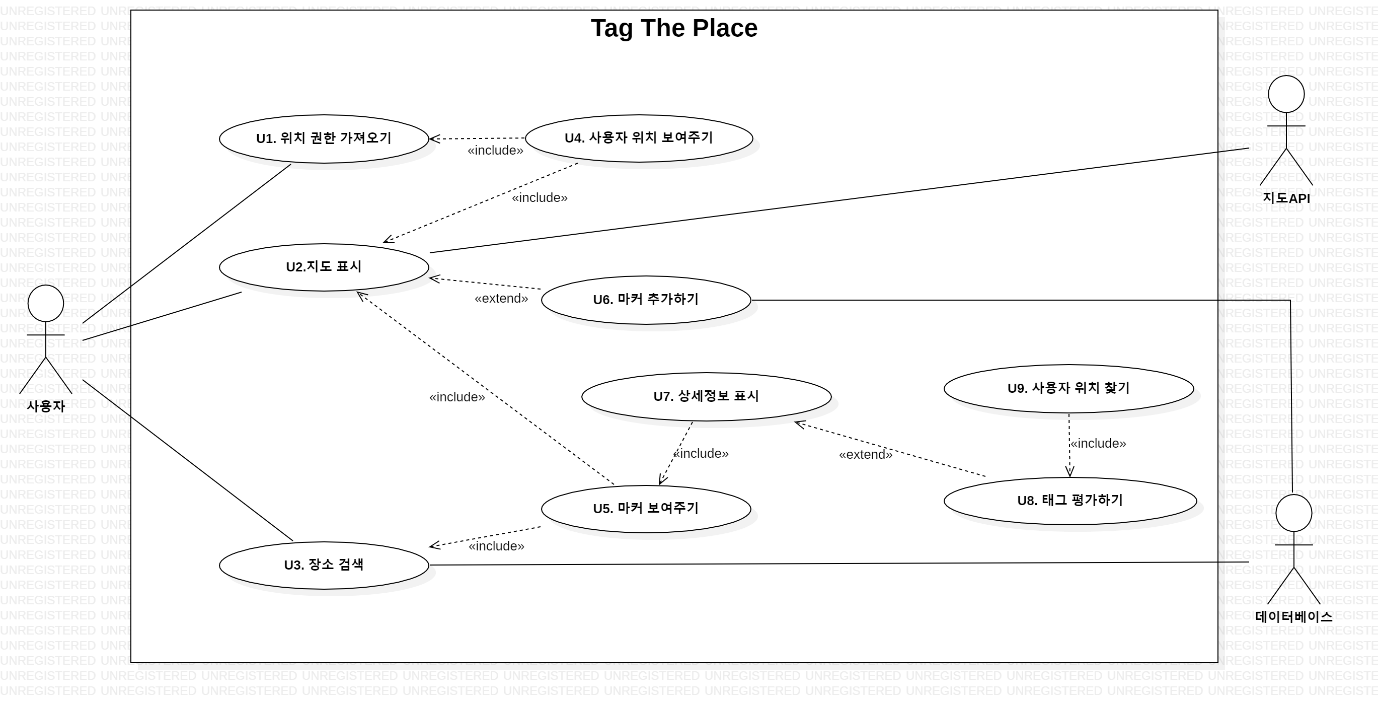


Figure : 유스케이스 다이어그램

## 4.3 유스케이스 상세 명세

앞선 단락에서의 유스케이스 다이어그램에 대한 상세한 명세를 나타낸다. 아래의 **Table 1 ~ 9** 은 각 유스케이스의 세부사항이다.

Table : U1. 위치권한 가져오기

|  |  |
| --- | --- |
| **U1. 위치권한 가져오기** | |
| **액터** | 사용자 |
| **설명** | 사용자의 현재 위치를 활용하기 위해서 필요한 권한을 가져오는 기능이다. 안드로이드 기기에서 GPS를 사용하기 위해서는 권한이 필요하기 때문에, 해당 기능이 필요하다. |
| **데이터** | 사용자의 위치정보 |
| **자극** | 사용자의 위치 정보에 대한 접근을 허용 |
| **반응** | 사용자의 위치 정보를 활용 가능 |
| **주석** | 어플리케이션 실행 시 자동으로 권한을 요청하는 알림을 발생시킨다. |

Table : U2. 지도 표시

|  |  |
| --- | --- |
| **U2. 지도 표시** | |
| **액터** | 사용자, 지도 API |
| **설명** | 사용자에게 현재 위치 근처의 지도를 보여주기 위한 기능이다. 사용자 인근의 지도를 보여준다. 사용자가 어플리케이션을 실행하면 지도 화면을 보여준다. |
| **데이터** | 사용자의 위치정보, 지도 API |
| **자극** | 사용자가 어플리케이션 실행 |
| **반응** | 지도 API를 호출하여 지도와 위치를 표시 |
| **주석** | 위치 정보에 대한 권한이 없다면 서울 시청 인근을 보여준다. |

Table : U3. 장소 검색

|  |  |
| --- | --- |
| **U3. 장소 검색** | |
| **액터** | 사용자, 데이터베이스 |
| **설명** | 사용자가 해시태그를 통하여 장소를 검색하는 기능이다. 사용자가 해시태그를 통해 검색을 진행할 수 있다. 검색 내용을 포함하는 태그를 가진 정보들을 가져온다. |
| **데이터** | 해시태그 |
| **자극** | 사용자가 검색할 해시태그 입력 |
| **반응** | 데이터베이스에 입력 받은 해시태그를 사용한 검색 쿼리 전송 |
| **주석** | 결과가 없으면 결과가 없다는 알림을 보여준다. |

Table : U4. 사용자 위치 보여주기

|  |  |
| --- | --- |
| **4. 사용자 위치 보여주기** | |
| **액터** | 지도 API |
| **설명** | 사용자의 위치권한이 있다면 지도에서 사용자의 위치를 보여준다. 어플리케이션을 실행했을 때, 그리고 위치 버튼을 클릭했을 때, 지도 API에서 사용자의 위치를 표시한다. |
| **데이터** | 사용자의 위치정보, 지도 API |
| **자극** | 어플리케이션 실행, 위치 버튼 클릭 |
| **반응** | 사용자의 위치를 표시 |
| **주석** | 위치권한이 없다면 동작하지 않는다. |

Table : U5. 마커 보여주기

|  |  |
| --- | --- |
| **U5. 마커 보여주기** | |
| **액터** | 데이터베이스, 지도 API |
| **설명** | U3에서 검색한 결과로 얻어온 좌표를 마커로 만들어 지도에 보여준다. 이 마커는 어떤 장소를 나타내는지 장소의 이름과 해시태그를 보여준다. |
| **데이터** | 위도, 경도, 장소의 이름, 해시태그 |
| **자극** | 사용자가 검색할 해시태그 입력 |
| **반응** | 화면에 데이터베이스에서 받아온 좌표에 해당하는 마커 생성 |
| **주석** | 마커를 보여주기 위해서는 반드시 검색이 선행되어야 한다. 검색 내용이 없다면 마커를 보여주지 않는다. |

Table : U6. 마커 추가하기

|  |  |
| --- | --- |
| **U6. 마커 추가하기** | |
| **액터** | 사용자, 데이터베이스, 지도 API |
| **설명** | 사용자가 정보를 공유할 장소에 마커를 추가하는 기능이다. 추가한 마커는 다른 사용자가 확인하고 정보를 얻을 수 있다. |
| **데이터** | 위도, 경도, 해시태그, 장소의 이름 |
| **자극** | 사용자가 지도를 길게 클릭 |
| **반응** | 사용자의 입력이 일어나면 대화 상자를 나타내고, 여기에 사용자의 해시태그, 이름을 입력 받아 해당 장소의 좌표와 함께 데이터베이스에 저장 |
| **주석** | 장소의 이름과 해시태그는 필수적으로 기록해야 한다. 사용자가 마커를 추가하는 장소 근처에 없다면 마커를 추가할 수 없다. |

Table : U7. 상세정보 표시

|  |  |
| --- | --- |
| **U7. 상세정보 표시** | |
| **액터** | 사용자, 데이터베이스, 지도 API |
| **설명** | 사용자가 U5. 마커 보여주기의 결과로 나타난 마커의 정보 창을 클릭하면 장소의 이름, 평가 결과를 보여준다. 이 기능을 통해 장소에 대한 정보를 얻을 수 있다. |
| **데이터** | 위도, 경도, 장소의 이름, 해시태그 |
| **자극** | 사용자가 마커를 클릭 |
| **반응** | 장소의 이름, 태그, ‘좋아요’, ‘싫어요’의 수, ‘좋아요’ 버튼, ‘싫어요’ 버튼을 포함한 알림 생성 |
| **주석** | ‘좋아요’와 ‘싫어요’의 수가 0이라면 ‘처음으로 평가를 남겨 보세요’ 라는 메세지를 보여준다. |

Table : U8. 태그 평가하기

|  |  |
| --- | --- |
| **U8. 태그 평가하기** | |
| **액터** | 사용자, 데이터베이스 |
| **설명** | ‘좋아요’ ‘싫어요’ 버튼을 통해 태그 검색 결과를 평가할 수 있는 기능이다. 이 기능을 통해 정보의 신뢰성을 평가하고 판단할 수 있다. |
| **데이터** | 사용자의 입력 여부 |
| **자극** | 사용자가 U7. 상세정보 표시기능에서 나타난 버튼을 클릭 |
| **반응** | 버튼을 클릭하면 데이터베이스의 ‘좋아요’, 혹은 ‘싫어요’의 값에 +1을 하는 쿼리를 전송하고 버튼을 보이지 않게 처리 |
| **주석** | 각 장소에 대해 사용자는 한 번의 평가만 입력만 할 수 있다. |

Table : U9. 사용자 위치 찾기

|  |  |
| --- | --- |
| **U9. 사용자 위치 찾기** | |
| **액터** | 사용자, 지도 API |
| **설명** | 사용자의 위치를 찾아서 확인하는 기능이다. 마커를 추가할 때 사용자 인근의 위치를 추가하는지 확인하기 위해 동작한다. |
| **데이터** | 사용자의 위치 |
| **자극** | 사용자가 마커를 추가 |
| **반응** | 가장 최근에 기록된 사용자의 위치를 반환한다. |
| **주석** | 기록된 위치가 존재하지 않을 때, 사용자의 위치를 새로 요청해서 반환한다. |

# 5. 시스템 아키텍처

## 5.1 클래스 다이어그램

이번 단락에서는 제안하는 어플리케이션의 구조를 나타내기 위해 [클래스 다이어그램(Class Diagram)](#_3._용어_사전)을 포함한다. 클래스 다이어그램에서는 클래스가 어떤 메소드를 가지는지만을 나타내고, 메소드와 메소드의 상호작용은 추후 [**7.1 시퀀스 다이어그램**](#_7.1_시퀀스_다이어그램)에서 나타낸다. **Figure 3**은 제안하는 어플리케이션의 클래스 다이어그램이다.

## 5.2 클래스 다이어그램 명세

Figure : 클래스 다이어그램

Table 10 ~ 14는 클래스 다이어그램에 대한 명세를 나타낸다. 대략적으로 어떤 동작을 하는 클래스인지 그리고 클래스의 구성 요소는 무엇을 의미하는지를 설명한다.

**•** MainActivity는 어플리케이션의 주 화면과 연결되어 대부분의 동작을 실행하는 클래스이다. 해당 클래스에 대한 명세는 아래 **Table 10**에서 나타낸다. 명세에 나타낸 구글 지도 API에 대한 내용은 추후 서술한다.

Table : MainActivity 명세

|  |  |  |
| --- | --- | --- |
| **MainActivity** | | |
| **Attribute** | +mapFragment: SupportMapFragment | 지도 표시를 위한 [컴포넌트(Component)](#_3._용어_사전) |
| +googleMap: GoogleMap | 지도 컴포넌트 위에 구현될 지도 객체, 구글 지도 API를 사용 |
| +PERMISSIONS: String [] | GPS 사용 권한들의 String 값을 저장한 배열 |
| +DEFAULT\_ZOOM: int | 지도의 배율을 결정하는 상수 |
| +CITY\_HALL: LatLng | GPS 사용 권한이 없을 때 지도의 시작 위치를 결정하는 좌표 |
| **Operation** | #onCreate (Bundle): void | 어플리케이션의 생명주기 중 하나이다. 어플리케이션이 실행되면 동작한다. GPS 권한을 요청하고, 지도를 보여주는 기능을 한다. |
| +onRequestPermissionsResult (int, String, int) : void | onCreate에서 요청한 GPS 권한을 확인하고, 권한을 획득했다면 그 결과를 처리하는 메소드이다. |
| +onMapReady (GoogleMap): void | SupportMapFragment에 GoogleMap 객체를 보여주는 메소드이다. 지도에 대한 LongClick 이벤트 처리, 사용자 위치 표시기능을 포함한다. |
| +onInfoWindowClick (Marker): void | 마커의 정보 창을 클릭했을 때 동작하는 메소드이다. AlertDialogue를 나타내고, 해당 Dialogue에는 장소에 대한 정보, ‘좋아요’, ‘싫어요’ 버튼이 나타난다. |
| +onCreateOptionsMenu (Menu): boolean | 메뉴에 대한 처리를 담당하는 메소드이다. 검색 메뉴를 포함하고, 검색시에 일치하는 결과에 대한 마커를 지도에 나타내는 기능을 한다. |
| -getMyLocation (): LatLng | 사용자의 위치를 가져오는 메소드이다. |
| -checkPermission (): boolean | GPS 관련 권한을 확인하여 boolean을 반환하는 메소드이다. |

**•** OnMapReadyCallback, OnInfoWindowClickListener 두 인터페이스는 각각 구글 지도를 보여줄 때 사용하는 onMapReady(GoogleMap)와 마커의 정보 창을 클릭했을 때의 처리를 담당하는 onInfoWindowClick(Marker)를 필수적으로 구현해야 한다. 구글 지도와 마커 정보 창의 클릭 기능을 사용하기 위해 MainActivity가 상속받는다. 구조는 아래 **Table 11, Table 12**와 같다.

Table : OnMapReadyCallback 명세

|  |  |  |
| --- | --- | --- |
| **<<interface>>**  **OnMapReadyCallback** | | |
| **Attribute** | +onMapReady (GoogleMap): void | GoogleMap 객체를 상속받아 구현하기 위한 인터페이스 |

Table : OnInfoWindowClickListener 명세

|  |  |  |
| --- | --- | --- |
| **<<interface>>**  **GoogleMap.OnInfoWindowClickListener** | | |
| **Attribute** | +onInfoWindowClick(Marker): void | 마커의 정보 창에 대한 클릭 이벤트를 구현하기 위한 인터페이스 |

**•** PlaceData는 장소에 대한 정보를 하나의 자료형처럼 사용하기 위한 클래스이다. 이는 데이터베이스의 테이블이기도 하다. 데이터베이스에 대한 내용은 후술한다. 본 클래스는 장소의 이름, 태그, 좌표, ‘좋아요’, ‘싫어요’ 수 그리고 식별을 위한 ID를 가진다. 아래 **Table 13**은 PlaceData에 대한 명세이다.

Table : PlaceData 명세

|  |  |  |
| --- | --- | --- |
| **PlaceData** | | |
| **Attribute** | -id: int | 데이터를 식별하기 위한 ID |
| -name: String | 장소의 이름 |
| -tag: String | 장소의 태그 |
| -like: int | ‘좋아요’ 수 |
| -dislike: int | ‘싫어요’ 수 |
| -lat: float | 장소의 위도 |
| -lng: float | 장소의 경도 |
| **Operation** | +getId (): int | id 값을 반환하는 메소드 |
| +getName (): String | 장소 값을 반환하는 메소드 |
| +getTag (): String | 태그 값을 반환하는 메소드 |
| +getLike (): int | ‘좋아요’ 수를 반환하는 메소드 |
| +getDislike (): int | ‘싫어요’ 수를 반환하는 메소드 |
| +getLat (): float | 장소의 위도를 반환하는 메소드 |
| +getLng (): float | 장소의 경도를 반환하는 메소드 |
| +setId (int): void | id 값을 설정하는 메소드 |
| +setName (String): void | 장소 값을 설정하는 메소드 |
| setTag (String): void | 태그 값을 설정하는 메소드 |
| setLike (int): void | ‘좋아요’ 수를 설정하는 메소드 |
| setDislike (int): void | ‘싫어요’ 수를 설정하는 메소드 |
| setLat (float): void | 장소의 위도를 설정하는 메소드 |
| setLng (float): void | 장소의 경도를 설정하는 메소드 |

**•** PlaceRepository는 데이터베이스와 통신을 하기 위한 클래스이다. 본 어플리케이션에서는 java.sql.Connection을 통해 데이터베이스와 통신한다. 데이터베이스 및 서버 연결에 대한 자세한 내용은 후술한다. 아래 **Table 14**는 PlaceRepository의 명세이다.

Table : PlaceRepository 명세

|  |  |  |
| --- | --- | --- |
| **PlaceRepository** | | |
| **Attribute** | +conStr: String | 서버 통신을 위한 주소 |
| **Operation** | +getPlacesByTag (String): List<PlaceData> | 태그 값을 입력 받고, 그 태그를 가진 데이터들을 데이터베이스에서 PlaceData List로 가져오는 메소드 |
| +getPlaceById (int): PlaceData | 데이터의 id를 입력 받고, 데이터베이스에서 해당 id를 가진 PlaceData를 가져오는 메소드 |
| +insertDataToDB (PlaceData): void | PlaceData를 데이터베이스에 새로 추가하는 메소드 |
| +increaseLikeById (int): void | 데이터의 id를 입력 받고, 데이터베이스에서 해당 id를 가진 PlaceData의 ‘좋아요’ 수를 1 증가시키는 메소드 |
| +increaseDislikeById (int): void | 데이터의 id를 입력 받고, 데이터베이스에서 해당 id를 가진 PlaceData의 ‘싫어요’ 수를 1 증가시키는 메소드 |

# 6. 시스템 요구사항 정의

## 6.1 기능적 요구사항

본 프로젝트에서 제안하는 어플리케이션은 본 문서의 [**4. 사용자 요구사항 정의**](#_4._사용자_요구사항)에서 설명한 내용에 대한 기능을 포함해야 한다. 다시 말해 사용자의 위치를 지도에 나타내고, 마커를 생성할 수 있어야 하며, 검색을 통해 필요한 정보에 대한 마커만 나타내야 한다.

아래 **Figure 4, Figure 5**는 제안하는 어플리케이션의 예상 실행 화면이다. **Figure4**의 **(a)**는 어플리케이션의 최초 실행화면으로, 사용자의 위치를 나타낸다. **Figure4**의 **(b)**는 ‘낭만’이라는 태그를 검색한 결과 화면으로, 검색한 태그에 포함되는 모든 장소들을 지도에 마커로 나타낸다. 마커의 정보 창을 클릭하면 **Figure4**의 **(c)**와 같이 정보를 확인하고 평가할 수 있는 화면을 나타낸다.

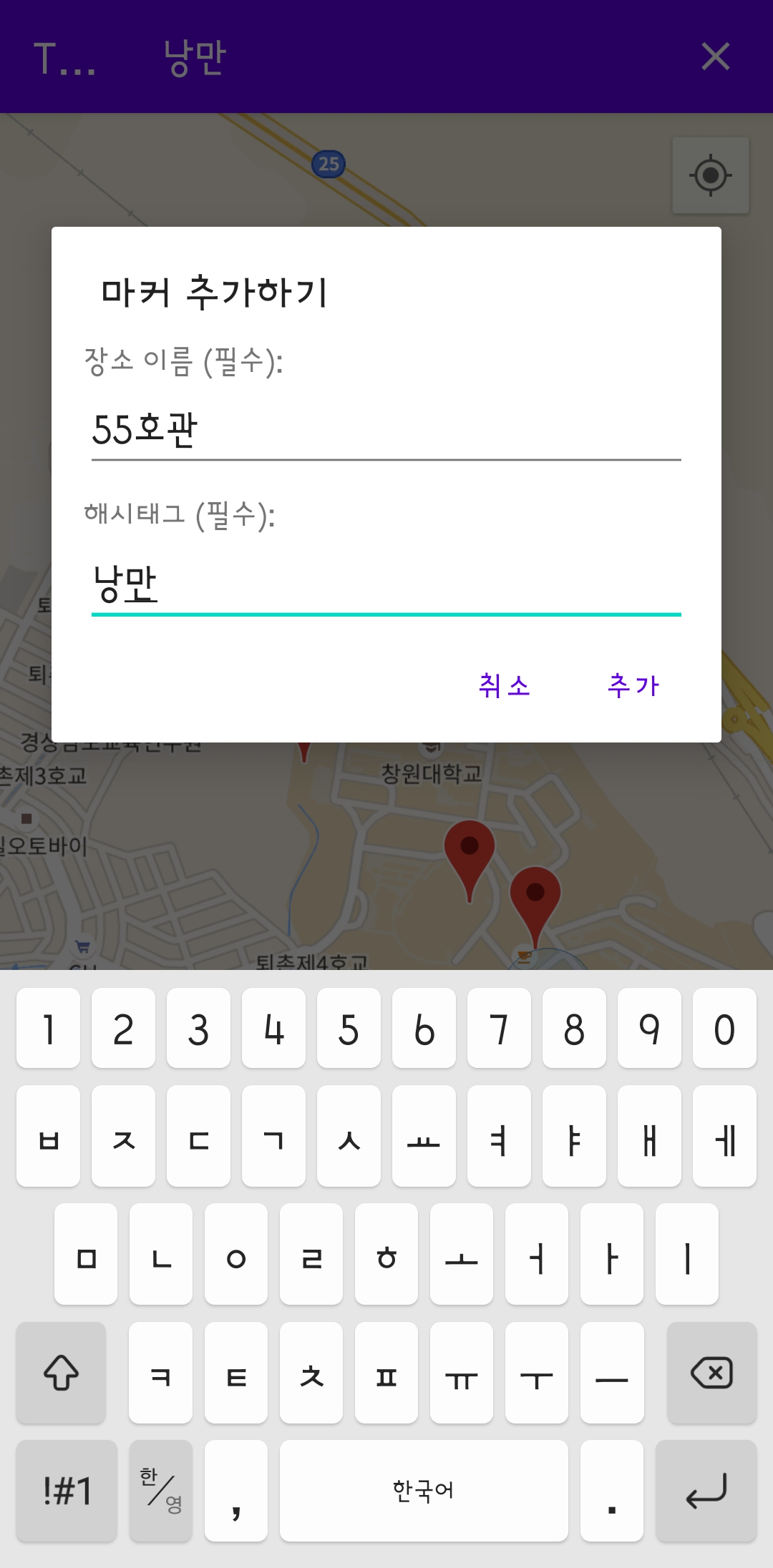
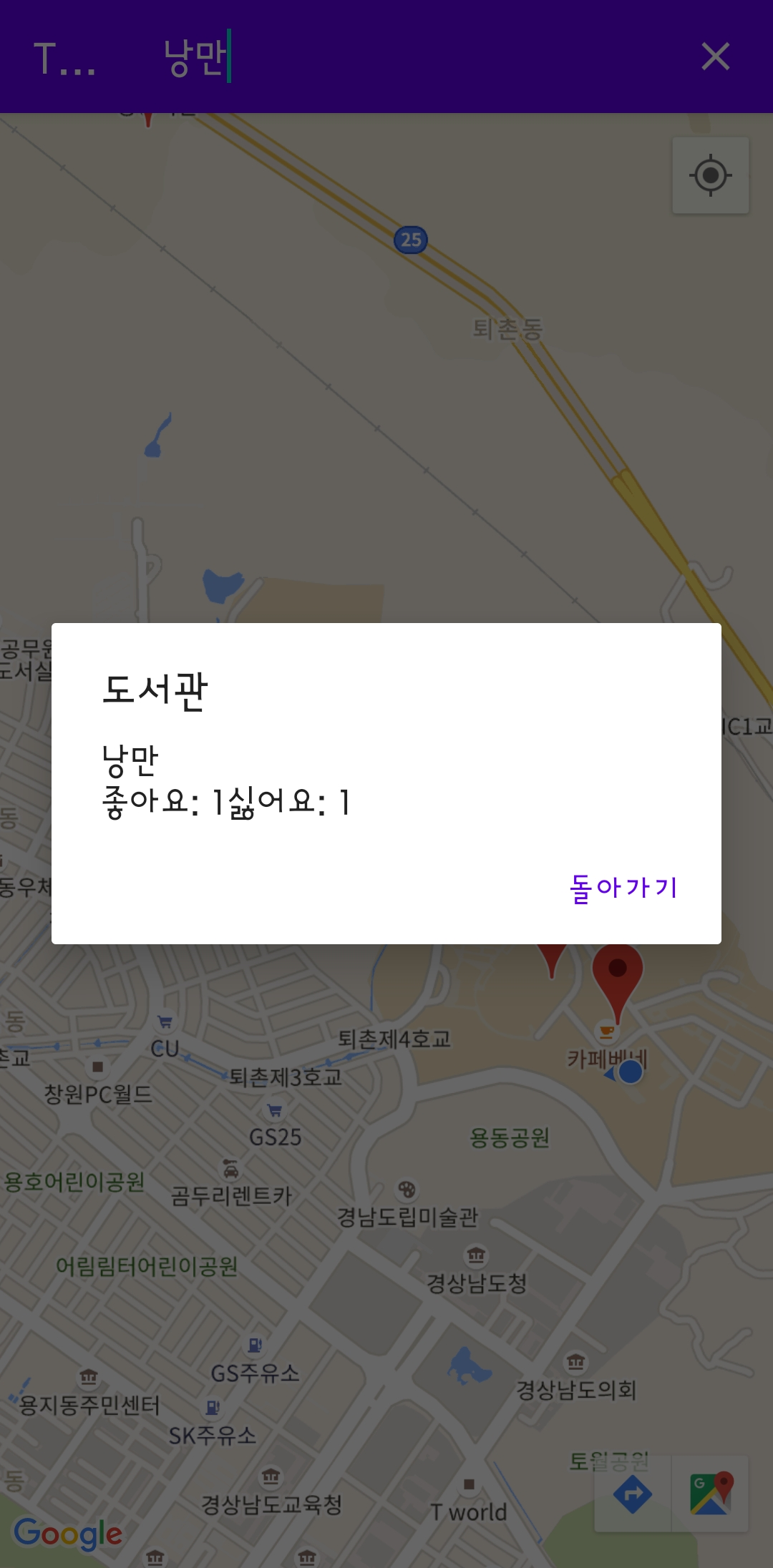
정보에 대한 평가 기능과 마커를 추가하는 기능은 **Figure 5**에서 보여준다. **Figure 5**의 **(a)**는 **Figure 4 (c)**의 화면에서 ‘싫어요’버튼을 누른 후의 모이다. 이처럼 한 번 평가를 하고 난 후에 정보 화면에는 ‘좋아요’와 ‘싫어요’ 버튼이 보이지 않게 되어야 한다.

**Figure 5**의 **(b)**는 지도 위의 한 지점을 길게 눌렀을 때 나타나는 화면이다. 해당 화면에서 장소의 이름과 태그를 기록하고 ‘추가’ 버튼을 누르면 **Figure 5**의 **(c)**와 같이 새로 마커가 추가된다.

## 6.2 비기능적 요구사항

Figure : Tag The Place 실행화면 1

Figure : Tag The Place 실행화면 2



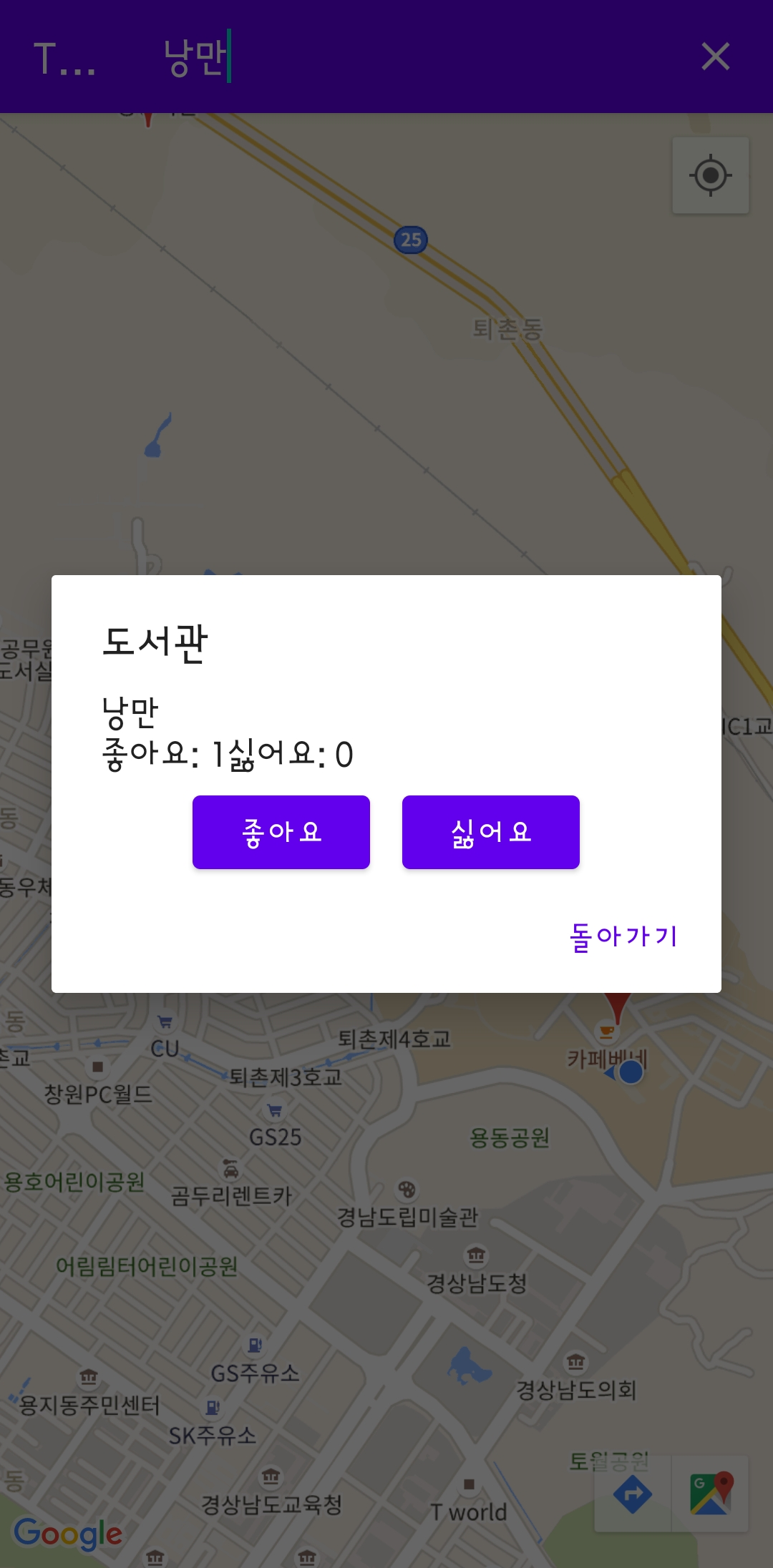
**(a)**

**(b)**

**(b)**

**(c)**

**(c)**



**(a)**

**(a)**

**(b)**

**(b)**

**(c)**

**(c)**

비기능적 요구사항은 어플리케이션의 기능적인 부분 이외의 제약사항이나 개발 환경에 대한 제한사항 등을 포함한다. 본 프로젝트에서는 단락 [**4.1 준수해야하는 프로세스**](#_4.1_준수해야_하는) 이외의 비기능적 요구사항은 아래와 같다.

**•** 마커를 추가하는 기능은 사용자의 위치 권한이 없으면 실행할 수 없다.

**•** 마커를 추가할 장소가 사용자의 위치로부터 반경 약 450m(위도, 경도 +- 0.003 범위) 이내에 위치해야 한다.

**•** 본 어플리케이션에서 데이터베이스에 저장한 장소 정보를 다른 사용자가 확인하기 위해서는 데이터베이스를 서버와 연동해야 한다. 서버 연동 및 데이터베이스 생성 방법은 단락 [**9.1 서버 개설 및 데이터베이스 생성**](#_9.1_서버_개설)에서 설명한다.

**•** 본 어플리케이션에서 지도 API는 구글 클라우드 플랫폼에서 제공하는 Maps SDK for Android를 사용한다. 방법에 대해서는 단락 [**9.2 구글 클라우드:** **Maps SDK for Android**](#_9.2_구글_클라우드:)에서 설명한다.

# 7. 시스템 모델

## 7.1 시퀀스 다이어그램

# 8. 시스템 진화

//여기에는 뭐가 들어가야할까요 여러분

# 9. 부록

## 9.1 서버 개설 및 데이터베이스 생성

## 9.2 구글 클라우드: Maps SDK for Android

# 10. 색인

이 단락은 참조 문헌의 색인이다. 하이퍼링크가 연결되어 있다.  
[1] Azure SQL Database 단일 데이터베이스 만들기 설명서: <https://docs.microsoft.com/ko-kr/azure/azure-sql/database/single-database-create-quickstart?tabs=azure-portal>

[2] Microsoft Azure: https://portal.azure.com/#home