시스템 설계서

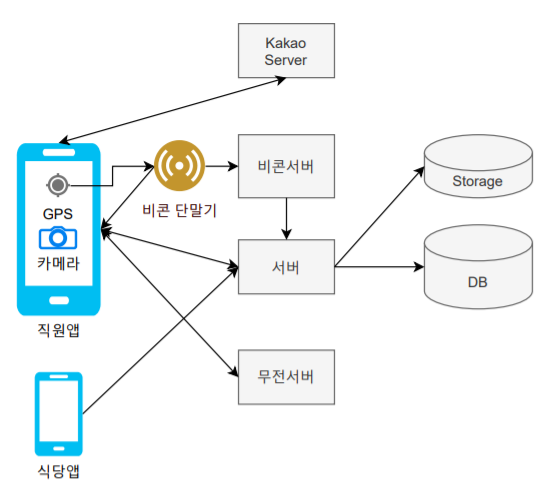
지도 교수님: 지정희 교수님

201711350 조대현

201811197 어혜령

**목차**

1. **Architecture**
   1. System Architecture
   2. Software Architecture
2. **Component Diagram**
   1. Component Definition
   2. Component Interface
   3. Component Algorithm
3. **Sequence Diagram**
   1. Use-Case-Sequence Diagram
4. **User Interface Configuration**
5. **ER Diagram**
6. **Programming Strategy**
   1. **Programming Language**
   2. **Programming Tool**
7. **Others**
   1. **실험 장소 설정**
8. **Architecture**
   1. System Architecture



직원앱: 안드로이드 버전만 Kotlin 개발합니다. MVVM패턴을 사용하여 개발합니다.

식당앱: 안드로이드 버전을 Kotlin으로 개발합니다.

서버+DB: 차량 정보와 유저 정보를 저장하고, 이를 통해 실시간 입출차, 데이터 추가, 삭제, 변경, 알림 전송을 처리합니다. 데이터 저장과 실시간 이벤트 감지가 가능한 Cloud Firestore를 사용합니다.

Storage: 차량 이미지를 저장하기 위해 Firebase용 Cloud Storage를 이용합니다.

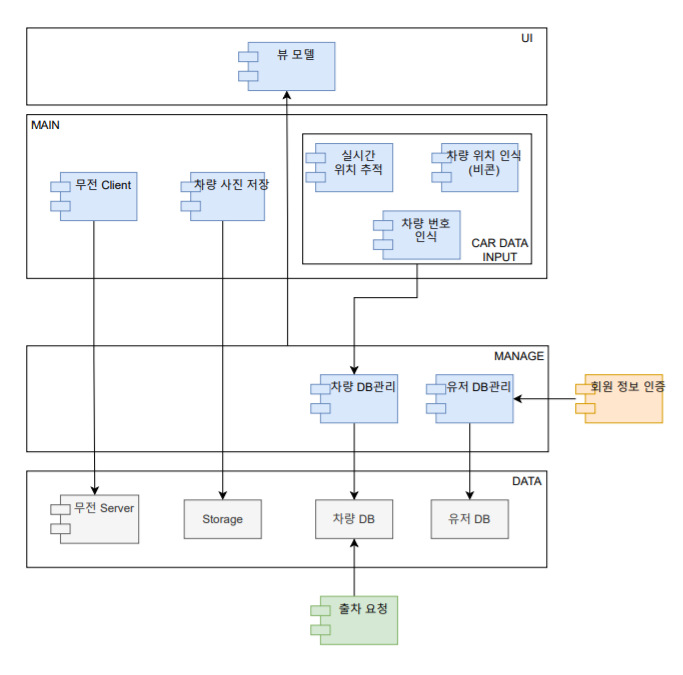
KAKAO서버: 카카오톡 로그인을 통해 계정 인증 후 카카오 서버로부터 authorization code 및 access token을 발급받아 사용자의 개인 정보 접근 이벤트를 처리하도록 해주는 외부 서버입니다.

무전 서버: Socket통신을 통해서 Client(직원앱)에서 발생하는 음성 파일을 받아서 서버에 접속한 Client들에게 전송하는 서버입니다.

비콘 단말기: IBeacon SDK에서 산출된 beacon의 UUID를 수신 신호 반경 이내의 사용자 핸드폰으로 전송합니다

비콘 서버 : 비콘 단말기로부터 전송된 정보를 직원앱이 비콘 서버로 송신하여 RSSI신호를 검출 받습니다. 이후 비콘의 물리적인 위치로부터 거리를 측정하고 오차를 최소화하여 산출된 위치 정보를 DB에 저장합니다.

* 1. Software Architecture



1. **Component Diagram**

|  |
| --- |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| 컴포넌트ID | 컴포넌트명 | 개요 | 관련 유스케이스 |
| US\_MNG | 유저 DB 관리 | 유저 DB를 통해 회원가입시 유저 정보 등록, 로그인 시 유저 정보 확인 | 회원가입, 로그인, |
| US\_AU | 회원 정보 인증 | Kakao serve를 통한 회원의 본인 인증 진행 | 회원가입 |
| CAR\_NM | 차량 번호 인식 | 번호판 사진 촬영, 번호판 객체 인식을 위한 오픈소스와 문자 인식을 위한 오픈소스를 통해 차량 번호를 인식 | 입차 차량 등록 |
| CAR\_LOC | 차량 위치 인식  (비콘) | 비콘 단말기에서 전송되는 RSSI신호를 감지하여 주차장 내의 사용자 위치 인식 | 입차 차량 등록 |
| CAR\_PIC | 차량 사진 저장 | 차량 사진 촬영, 촬영 사진 storage에 저장 | 입차 차량 등록 |
| CAR\_MNG | 차량DB관리 | 차량 DB를 통해 차량 정보 저장, 차량 상태 변경 감지, 출차 관리, 차량 상태 변경 | 입차 차량 등록, 출차 승낙 |
| RT\_LOC | 실시간 위치 추적 | 출차 중인 차량이 식당의 범위내에 들어온 것을 감지 | 근접 차량 확인 |
| EX\_RE | 출차 요청 | 차량 등록 여부 체크 후 차량 DB를 통해 출차 요청 | 출차 요청 |
| VOC\_CL | 무전 Client | 무전 서버와 연결하여 음성 메시지 전송 및 받은 음성 메시지 출력 | 무전,(무전누락 내역) |
| VOC\_SV | 무전 서버 | 접속한 Client의 음성 메시지 받아서 나머지 Client들에게 전달 | 무전,(무전누락 내역) |
| V\_M | 뷰 모델 | Activity의 수명 주기를 고려하여 UI 관련 데이터 저장 관리 |  |

* 1. Component Definition

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 유저 DB 관리 | **컴포넌트ID** | | US\_MNG |
| **컴포넌트 개요** | 유저 DB를 통해 회원가입시 유저 정보 등록, 로그인 시 유저 정보 확인 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| Login | 1. 사용자가 입력한 로그인 정보가 DB에 저장되어 있는 정보와 일치 여부 판단하여 Boolean 값을 리턴하는 메소드  2. 뷰모델 Component의 setUserLiveData함수 호출하여 로그인 유저 정보 전달 메소드 | | | |
| SignUp | 1. Firebase instance 생성 및 컬렉션 초기화 메소드  2. 카카오톡 회원 정보 인증 성공 판별 후 DB에 새로운 사용자 정보 등록하는 메소드 | | | |
| User | 사용자의 닉네임, 근무지 정보 포함된 데이터 클래스 | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| UM\_TO\_VM | 로그인 유저 저장 | | setUserLiveData | |
| UDB\_TO\_UM | 유저 정보 IO | | get | |
| set | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **컴포넌트명** | | 회원 정보 인증 | **컴포넌트ID** | | US\_AU |
| **컴포넌트 개요** | | Kakao serve를 통한 회원의 본인 인증 진행 | | | |
| **내부 클래스** | | | | | |
| **클래스명** | | **비고** | | | |
| MasterApplication | | Application을 상속받은 class로, Activity가 호출되기 이전에 먼저 호출되어 어플리케이션 구동에 필요한 메소드 인스턴스화 진행   1. 발급받은 네이티브앱 키 정보를 KakaoSdk class 생성자로 지정하여 인스턴스화 진행하는 메소드 | | | |
| ConnectKakao  Server | | 1. 네트워크 작업 진행을 위해 INTERNET Permission을 포함해야 하므로 Permission 체크 및 요청 메소드 2. Permission 체크에 대한 결과를 확인할 수 있는 override 메소드 3. 카카오톡 설치 체크, 카카오톡 실행 및 개인 정보 제공 동의, 사용자의 개인 정보 접근을 위한 토큰 발급 메소드 4. 발급받은 토큰을 이용하여 카카오 API UserApiClient class의 instance를 리턴 | | | |
| **인터페이스 클래스** | | | | | |
| **ID** | **인터페이스명** | | | **오퍼레이션명** | |
| UA\_TO\_UM | 인증 정보 | | | getUserApiClient | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **컴포넌트명** | | 차량 번호 인식 | **컴포넌트ID** | | CAR\_NM |
| **컴포넌트 개요** | | 번호판 사진 촬영, 번호판 객체 인식을 위한 오픈소스와 문자 인식을 위한 오픈소스를 통해 차량 번호를 인식 | | | |
| **내부 클래스** | | | | | |
| **클래스명** | | **비고** | | | |
| DetectCarNumber  :Activity | | 사진을 촬영하고 차량 번호를 인식하여 반환   1. 사진 촬영 메소드 2. onActivityResult(콜백함수)를 재정의하여 미리보기 이미지 제공 메소드 3. 내부 저장소에 사진 저장 메소드 4. UseYOLOModel과 UseCRNNModel로 객체 전달 및 반환된 객체 저장 메소드 5. text 사용여부 확인 후 사용하지 않을 시에 InputCarNumber 수행 메소드 6. 차량 번호 반환 메소드(getCarNum) | | | |
| UseYOLOModel | | DetectCarNumber에서 번호판 사진을 받아서 번호판 영역을 번호판 객체를 추출하여 반환한다. | | | |
| UseCRNNModel | | DetectCarNumber에서 번호판 객체를 받아서 문자를 인식하여 반환한다. | | | |
| InputCarNumber | | 1. 사용자가 입력한 차량 번호 입력 형식 확인 메소드 2. 입력된 차량 번호 저장 및 반환 메소드 | | | |
| **인터페이스 클래스** | | | | | |
| **ID** | **인터페이스명** | | | **오퍼레이션명** | |
| CAR\_TO\_CM | 입차 정보 전달 | | | getCarNum | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 차량 위치 인식 | **컴포넌트ID** | | CAR\_LOC |
| **컴포넌트 개요** | 비콘 단말기에서 전송되는 RSSI신호를 감지하여 주차장 내의 사용자 위치 인식 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| ParkingBeacon | 1. BeaconManager 인스턴스화 메소드 2. 핸드폰에서 비콘이 감지된 경우 BeaconManager 객체의 addRangeNotifier의 파라미터로 didRangeBeaconsInRegion 오버라이드 메소드를 구현한 RangeNotifier 익명 객체를 전송하여 비콘으로부터 핸드폰의 거리를 얻는 메소드 3. 감지된 비콘의 UUID로부터 매핑 된 주차장 정보를 리턴하는 메소드(getParkingLot) | | | |
| InputParkingArea | 주차 구역 선택 및 리턴 메소드 | | | |
| ParkingArea | ParkingBeacon의 2번 3번 오퍼레이션을 호출하고 처리된 결과로부터 주차장 구역 정보를 계산하여 리턴하는 메소드(getParkingArea) | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| CAR\_TO\_CM | 입차 정보 전달 | | getParkingArea | |
| getParkingLot | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 차량 사진 저장 | **컴포넌트ID** | | CAR\_PIC |
| **컴포넌트 개요** | 차량 사진 촬영, 촬영 사진 storage에 저장 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| TakeCarPhoto  :Activity | 1. 사진 촬영 메소드 2. onActivityResult(콜백함수) 3. 내부 저장소 사진 저장 메소드 4. 계속 촬영 여부 확인 메소드 | | | |
| SaveCarPhoto | 1. Storage 인스턴스 생성 및 사진 저장 메소드 2. 촬영 후 저장하지 않은 사진 존재 여부 확인 메소드 | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| CAR\_TO\_ST | 사진 저장 | | putFile | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 차량DB관리 | **컴포넌트ID** | | CAR\_MNG |
| **컴포넌트 개요** | 차량 DB를 통해 차량 정보 저장, 차량 상태 변경 감지, 출차 관리, 차량 상태 변경 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| ManageCarDB | 1. firestore 인스턴스 생성 2. 입차 정보 저장을 위해 정보를 전달받아 firestore에 차량 정보 저장 메소드(set) 3. 출차 시 차량 정보와 변경 상태를 전달받아 firestore에 차량 정보 변경 4. 출차 시 차량 정보와 근접 여부를 전달받아 firestore에 차량 정보 변경 5. 구역의 차량들의 상태 변경 감지 이벤트 핸들러    1. 차량 추가/삭제/변경: 리스트 갱신 요청    2. 출차 요청 상태: 출차 요청 6. 뷰모델 Component의 setCarLiveData함수 호출하여 입차한 차 정보 전달 메소드 | | | |
| ExitCar | 1. 출차 요청 notification 생성 메소드(createNotification) 2. 출차 수락 시 차량DB관리Component를 통한 차량 상태 변경, 실시간 위치 추적Component를 통해 위치 감지를 하는 메소드 3. 출차 완료 시 차량DB관리Component를 통한 차량 상태 변경 메소드 | | | |
| data class Car | 차량번호, 주차, 주차장구역, 상태(입차,출차요청,출차진행중,완료), 근접여부 | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| CM\_TO\_VM | 입차 차량 저장 | | setCarLiveData | |
| CM\_TO\_CDB | 차량 정보 IO | | set | |
| update | |
| snapshotListener | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 실시간 위치 추적 | **컴포넌트ID** | | RT\_LOC |
| **컴포넌트 개요** | 출차 중인 차량이 식당의 범위내에 들어온 것을 감지 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| RealTime  Location | 1. 사용자 개인 위치 정보 액세스를 위한 ACCESS\_FINE\_LOCATION permission과 백그라운드에서 위치정보를 액세스 하기 위한 ACCESS\_BACKGROUND\_LOCATION permission 체크 및 요청 메소드 2. 요청한 permission에 대한 결과를 call back 받기 위한 오버라이드 메소드 3. Geofence 위치, 반경, 진입 또는 이탈, 이벤트 발생시 Broadcast Receiver에서 구분할 id 등의 정보를 생성자로 지정한 Geofence 객체 생성 메소드 4. Location API 사용을 위한 Geofence Client 객체 생성, 설정한 Geofence 반경에 Client의 진입 또는 이탈 여부를 확인하기 위한 인스턴스 생성 메소드 5. 모니터링할 Geofencing list 지정 및 이벤트 트리거 방법 지정 메소드 6. Client가 Geofence에서 정의한 이벤트를 전송 받기 위한 Pending Intent 정의 메소드 7. Geofencing Client에 Geofence 정보 및 트리거를 지정한 객체와 이벤트 발생시 처리할 Broadcast Receiver를 추가하는 메소드(startGeofence) | | | |
| Geofence  BroadcastReceiver | Geofence에 대한 변경 이벤트의 오버라이드 메소드 | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| RL\_TO\_CM | 추적 정보 전달 | | startGeofence | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **컴포넌트명** | | 출차 요청 | **컴포넌트ID** | | EX\_RE |
| **컴포넌트 개요** | | 차량 등록 여부 체크 후 차량 DB를 통해 출차 요청 | | | |
| **내부 클래스** | | | | | |
| **클래스명** | **비고** | | | | |
| ExitCarRequest | 1. 차량 DB에 접근할 firestore instance를 생성하는 메소드 2. 요청한 차량이 firestore의 해당 구역의 collection에 들어온 차량 번호와 일치하는 차량 유무 확인하여 Boolean값 반환 3. 차량 DB의 차량 상태를 출차 요청 상태로 변경하는 메소드 4. 출차 요청된 차량들의 근접 상태 변경 이벤트 감지 핸들러 | | | | |
| **인터페이스 클래스** | | | | | |
| **ID** | **인터페이스명** | | | **오퍼레이션명** | |
| CDB\_TO\_EX | 출차 정보 IO | | | get | |
| update | |
| snapshotListener | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 무전 Client | **컴포넌트ID** | | VO\_CL |
| **컴포넌트 개요** | 무전 서버와 연결하여 음성 메시지 전송 및 받은 음성 메시지 출력 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| Client | 서버의 IP와 PORT번호를 Socket의 생성자로 전달하여 인스턴스화 시키고 Connection을 맺는 메소드(clientConnection) | | | |
|  | 1. RadioRecord 앱에서 오디오를 녹음하기 위한 RECORD\_AUDIO, 녹음된 pcm 데이터를 저장하기 위한 EXTERNAL\_STROAGE permission의 체크 및 요청을 위한 메소드 2. AudioRecord 객체 생성 및 초기화 메소드 3. AudioRecord를 통해 읽어온 pcm data를 파일에 write하기 위한 Thread | | | |
| RadioOutput | 1. AudioTrack, FileInputStream 객체 생성 및 초기화 메소드 2. AudioTrack의 play 메소드 실행 및 DataInputStream 객체 생성 메소드 3. AuidoTrack의 write 메소드를 실행하여 스피커로 음성 데이터를 송출하는 메소드(stream) | | | |
| SendRadioData | 무전 서버로 pcm data 파일을 Socket을 통해 전송하기 위한 Thread 메소드(start) | | | |
| GetRadioData | 무전 서버로부터 pcm data 파일을 Socket을 통해 전송받기 위한 Thread 메소드 | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| VO\_CL\_SV | 음성 데이터 송수신 | | clientConnection | |
| start | |
| stream | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **컴포넌트명** | 무전 서버 | **컴포넌트ID** | | VO\_SV |
| **컴포넌트 개요** | 접속한 Client의 음성 메시지 받아서 나머지 Client들에게 전달 | | | |
| **내부 클래스** | | | | |
| **클래스명** | **비고** | | | |
| RadioServer | 1. 클라이언트 소켓으로부터 접속된 소켓을 할당하고 list를 확보하는 생성자 2. SendRadioData의 메소드로 전송한 음성 데이터를 InputStream으로 받고, OutputStream 객체를 통해 list에 존재하는 클라이언트들에게 데이터를 전송하는 Thread 3. thread를 구동 시킬 메소드로 접속된 client의 여부를 판단하는 메소드 | | | |
| **인터페이스 클래스** | | | | |
| **ID** | **인터페이스명** | | **오퍼레이션명** | |
| VO\_CL\_SV | 음성 데이터 송수신 | | println | |

|  |  |  |  |
| --- | --- | --- | --- |
| **컴포넌트명** | 뷰 모델 | **컴포넌트ID** | V\_M |
| **컴포넌트 개요** | Activity의 수명 주기를 고려하여 UI 관련 데이터 저장 관리 | | |
| **내부 클래스** | | | |
| **클래스명** | **비고** | | |
| MyViewModel | 1. User LiveData 값을 추가해주는 메소드(setUserLiveData) 2. Car 배열의 LiveData 값을 추가해주는 메소드(setCarLiveData) | | |

* 1. Component Interface

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | UM\_TO\_VM | **인터페이스명** | 로그인 유저 저장 |
| **오퍼레이션명** | setUserLiveData | | |
| **오퍼레이션 개요** | 유저DB관리 Component에서 로그인 유저 정보를 전달인자로 보내 뷰 모델 Component의 User LiveData에 저장하는 프로세스로 뷰 모델 Component의 MyViewModel의 setUserLiveData함수를 호출한다. | | |
| **파라미터** | 로그인한 User 객체 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | UDB\_TO\_UM | **인터페이스명** | 유저 정보 IO |
| **오퍼레이션명** | get | | |
| **오퍼레이션 개요** | 유저DB관리 Component에서 입력한 유저 정보가 유저DB에 존재 여부를 확인하는 프로세스로 DocumentReference의   get함수를 호출하고, addOnSuccessListener에서 비교 후 존재 여부를 판별한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | UDB\_TO\_UM | **인터페이스명** | 유저 정보 IO |
| **오퍼레이션명** | set | | |
| **오퍼레이션 개요** | 유저DB관리 Component에서 새로운 사용자 정보를 유저DB로 보내서 저장하는 프로세스로 DocumentReference   클래스의 set함수를 호출한다. | | |
| **파라미터** | HashMap<User> | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | UA\_TO\_UM | **인터페이스명** | 인증 정보 IO |
| **오퍼레이션명** | getUserApiClient | | |
| **오퍼레이션 개요** | 유저DB 관리 Component에서 회원 정보 인증 Component의 카카오톡 서버에 저장된 개인 정보를 얻기 위한 프로세스로 ConnectKakaoServer 클래스의  getUserApiClient 오퍼레이션을 호출하여 Kakao API의 UserApiClient class의 싱글톤 객체인 instance를 UserApiClient 타입으로 리턴한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | instance : UserApiClient | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CAR\_TO\_CM | **인터페이스명** | 입차 정보 전달 |
| **오퍼레이션명** | getCarNum | | |
| **오퍼레이션 개요** | 차량 번호 인식 Component에서 추출된 차량 번호를 차량 DB관리 Component에 보내는 프로세스로  DetectCarNumber클래스의 getCarNum 오퍼레이션을 호출하여 차량 번호를 리턴한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 차량 번호판 정보 : String | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CAR\_TO\_CM | **인터페이스명** | 입차 정보 전달 |
| **오퍼레이션명** | getParkingArea | | |
| **오퍼레이션 개요** | 차량 위치 인식 Component에서 추출된 차량 주차 구역 정보를 차량 DB관리 Component에 보내는 프로세스로, ParkingArea 클래스의 getParkingArea 오퍼레이션을 호출하여 차량 주차 구역을 리턴한다. | | |
| **파라미터** | distList : DoubleArray | | |
| **반환값** | 주차장구역 : String | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CAR\_TO\_CM | **인터페이스명** | 입차 정보 전달 |
| **오퍼레이션명** | getParkingLot | | |
| **오퍼레이션 개요** | 차량 위치 인식 Component에서 추출된 차량 주차 구역 정보를 차량 DB관리 Component에 보내는 프로세스로, ParkingBeacon 클래스의 getParkingLot 오퍼레이션을 호출하여 차량 주차 구역을 리턴한다. | | |
| **파라미터** | uuid : UUID | | |
| **반환값** | 주차장 : String | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CAR\_TO\_ST | **인터페이스명** | 사진 저장 |
| **오퍼레이션명** | putFile | | |
| **오퍼레이션 개요** | 차량 사진 저장 Component에서 내부 저장소에 저장된 사진을 Storage로 저장하는 프로세스로 FirebaseStorage클래스의 putFile 오퍼레이션을 호출한다. | | |
| **파라미터** | file의 Uri | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CM\_TO\_VM | **인터페이스명** | 입차 차량 저장 |
| **오퍼레이션명** | setCarLiveData | | |
| **오퍼레이션 개요** | 차량DB관리 Component에서 입차한 Car 정보를 전달인자로 보내 뷰 모델 Component의 Car LiveData에 저장하는 프로세스로 뷰 모델 Component의 MyViewModel의 setCarLiveData함수를 호출한다. | | |
| **파라미터** | 입차한 Car  객체 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CM\_TO\_CDB | **인터페이스명** | 차량 정보 IO |
| **오퍼레이션명** | set | | |
| **오퍼레이션 개요** | 차량 관리 DB Component에서 입차 차량 정보를 차량DB로 저장하는 프로세스로 DocumentReference클래스의 set 오퍼레이션을 호출한다 | | |
| **파라미터** | HashMap<Car> | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CM\_TO\_CDB | **인터페이스명** | 차량 정보 IO |
| **오퍼레이션명** | update | | |
| **오퍼레이션 개요** | 차량 관리 DB Component에서 차량 관련 정보를 차량DB로 저장하는 프로세스로 DocumentReference클래스의 update 오퍼레이션을 호출한다. | | |
| **파라미터** | field명, 차량 상태:Int | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CM\_TO\_CDB | **인터페이스명** | 차량 정보 IO |
| **오퍼레이션명** | snapshotListener | | |
| **오퍼레이션 개요** | 차량DB에서의 차량의 상태 변경을 차량DB관리 Component로알리는 프로세스로 DocumentReference 클래스의 addSnapshotListener를 호출한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | RL\_TO\_CM | **인터페이스명** | 추적 정보 전달 |
| **오퍼레이션명** | startGeofence | | |
| **오퍼레이션 개요** | 실시간 위치 추적 Component에서 발생한 Geofence에 대한 변경 이벤트를 감지하고 차량 DB관리 Component에 보내는 프로세스로, BroadcastReceiver를 상속받은 GeofenceBroadcaseReceiver 클래스에 onRecieve를 재정의한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CDB\_TO\_EX | **인터페이스명** | 출차 정보 IO |
| **오퍼레이션명** | get | | |
| **오퍼레이션 개요** | 출차 요청 Component에서 받은 차량 번호를 차량DB에서 차량 리스트를 가져와서 차량 유무를 확인하는 프로세스로 DocumentReference   클래스의 get을 호출한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CDB\_TO\_EX | **인터페이스명** | 출차 정보 IO |
| **오퍼레이션명** | update | | |
| **오퍼레이션 개요** | 출차 요청 Component에서 출차 요청된 차량을 차량DB에서 출차 요청 상태로 변경하는 프로세스로 DocumentReference  클래스의 update을 호출한다. | | |
| **파라미터** | field명, 출차 요청 상태: Int | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | CDB\_TO\_EX | **인터페이스명** | 출차 정보 IO |
| **오퍼레이션명** | snapshotListener | | |
| **오퍼레이션 개요** | 차량DB 차량의 상태 변경을 출차 요청 Component로 알리는 프로세스로 DocumentReference 클래스의 addSnapshotListener를 호출한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | VO\_CL\_SV | **인터페이스명** | 음성 데이터 송수신 |
| **오퍼레이션명** | clientConnection | | |
| **오퍼레이션 개요** | 무전 Client Component가 통신을 위해 무전 서버 Component와 연결을 하는 프로세스로 Client클래스의 clientConnection 오퍼레이션을 호출한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | VO\_CL\_SV | **인터페이스명** | 음성 데이터 송수신 |
| **오퍼레이션명** | start | | |
| **오퍼레이션 개요** | 무전 Client Component에서 감지된 녹음 이벤트 시작을 무전 Server Component로 전송하는 프로세스로 SendRadioData 클래스의 thread 메소드를 start 한다 | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | VO\_CL\_SV | **인터페이스명** | 음성 데이터 송수신 |
| **오퍼레이션명** | stream | | |
| **오퍼레이션 개요** | 무전 Client Component가 무전 Server Component로 부터 전송받은 pcm 파일을 출력해주기 위한 프로세스로 RadioOutput 클래스의 thread 메소드를 start 한다 | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **인터이페스 ID** | VO\_CL\_SV | **인터페이스명** | 음성 데이터 송수신 |
| **오퍼레이션명** | println | | |
| **오퍼레이션 개요** | 무전 서버 Component가 전달받은 데이터를 무전 Client Component로 전송하는 프로세스로 PrintWriter클래스의 println 오퍼레이션을 호출한다. | | |
| **파라미터** | 없음 | | |
| **반환값** | 없음 | | |

* 1. Component Algorithm
     1. 차량 번호 인식
        1. 알고리즘
* 입차 하려는 차량의 번호판을 카메라를 통해 촬영한다
* 번호판 사진 파일을 저장한다
* YOLOv3 모델의 input으로 사진을 보내고 output으로 번호판 객체를 반환 받는다
* CRNN모델의 input으로 번호판 객체를 보내고 output으로 차량 번호 text를 반환 받는다.
* 차량번호 추출 정보가 잘 추출 되었는지 사용자에게 확인 받는다.
* 성공적으로 추출된 경우 차량 번호 text를 반환 받는다
* 정확한 추출 실패시, InputCarNumber Class를 통한 차량 번호를 수기로 입력받는다.
  + - 1. 인식 실패 시
* 사용할 만한 인식률이 나오지 않는다면 번호판 객체 인식 후 좌표 영역을 확대 / 축소 하여 인식률을 높이는 과정을 추가하고, CRNN의 학습 데이터에 잡음과 왜곡을 추가하여 인식률을 높일 예정입니다.
  + 1. 차량 위치 인식
       1. 알고리즘
* IBeacon SDK로부터 산출된 beacon의 ID정보를 사용자 핸드폰으로 전송
* 해당 정보를 비콘 서버로 송신하여 각 beacon에 해당하는 RSSI 신호를 검출
* RSSI신호 및 경로 감쇠 모델을 바탕으로 거리를 측정
* 삼각측량 기법을 이용하여 교점을 추출
* 교점의 좌표 값을 바탕으로 주차된 구역을 추출
  + - 1. 거리 정확도 향상 알고리즘
* 동일한 비콘 모델, 실험 환경, 비콘 설정 임에도 거리별 신호 강도가 상이하므로, 경로 감쇠 지수를 비콘별로 제시하여 거리 정확도를 향상시킨다.
* 경로 감쇠 지수는 장소에 따라 제안된 경로 감쇠 지수를 참고하여 실험을 통해 아래의 식으로 경로 감쇠 지수를 산정할 것입니다.

테이블이(가) 표시된 사진

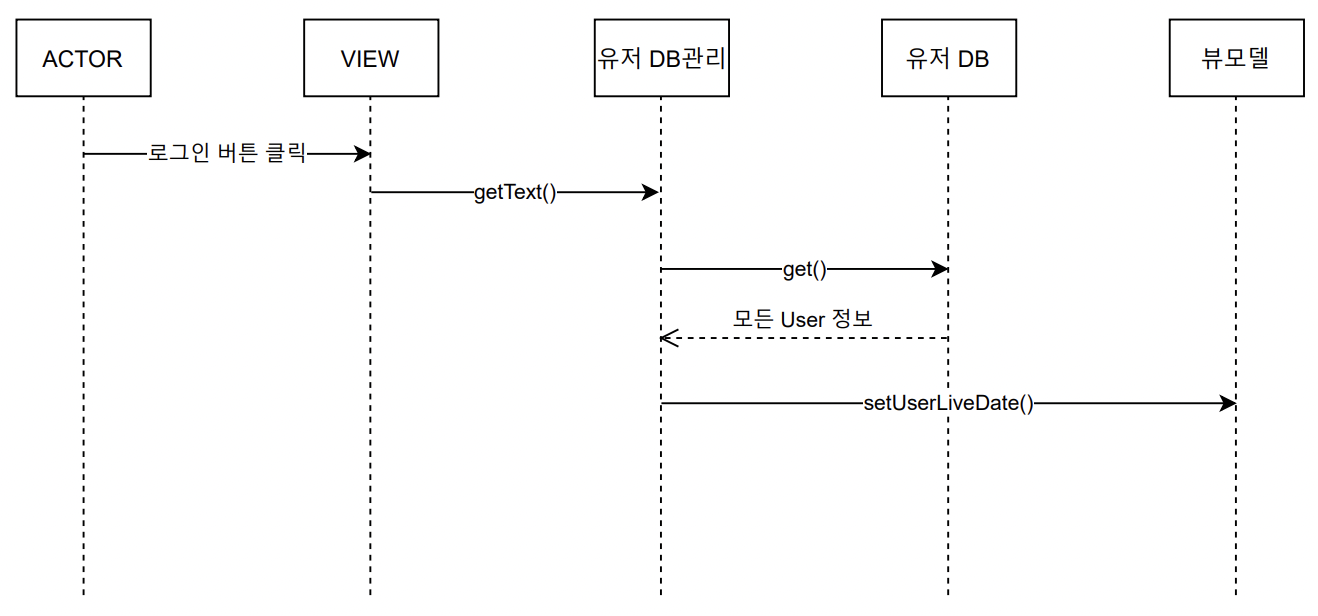
자동 생성된 설명

텍스트이(가) 표시된 사진

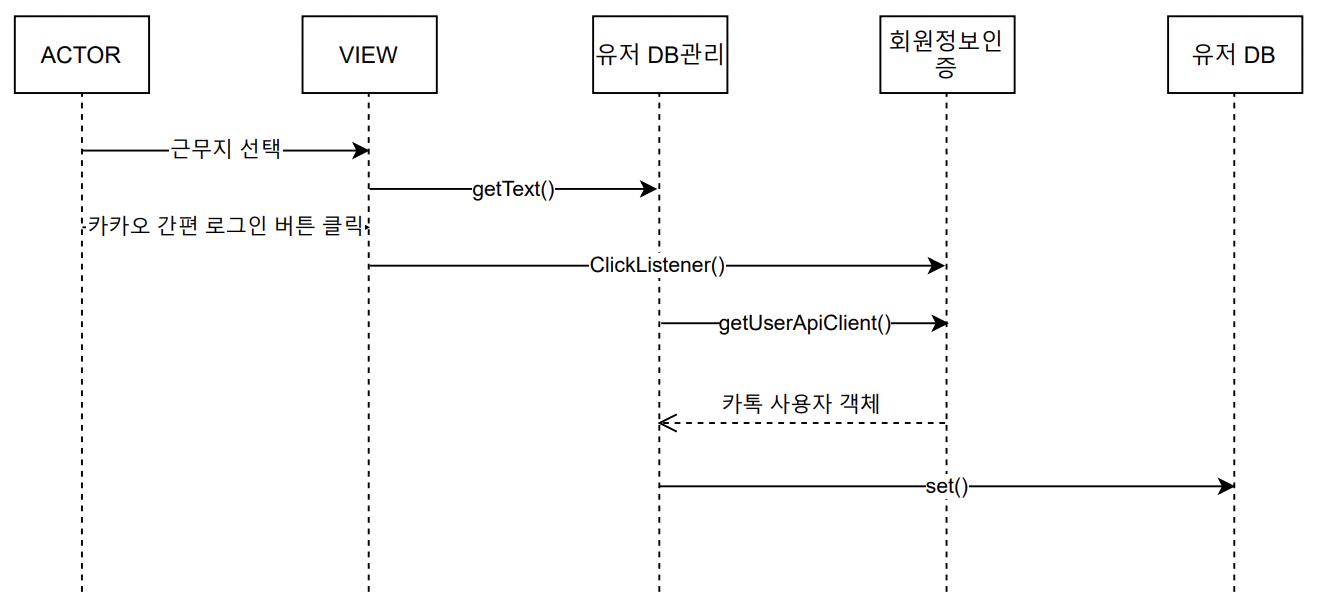
자동 생성된 설명

* 비콘의 내부 설정 중 출력 강도와 신호 출력 간격을 조정 후 분석하여 송출 세기와 송출 간격을 설정할 것입니다.
* 실험실 공간에 맞는 최소 비콘 개수를 직접 측정하는 것이 좋지만, 주차장을 대신할 실내 실험 공간 활용 및 금액상 제약이 있어 3차원 공간에서 최소 요구하는 개수인 4개를 사용할 것입니다.
  + 1. 출차 요청
       1. 알고리즘
* 식당 앱에서 입력된 차량 번호가 Database에 입차 상태로 등록되었는지 체크한다
* 차량 Database의 차량 상태 정보를 출차 요청으로 변경한다
* 직원 어플리케이션에서 차량 database의 차량 상태 정보 이벤트를 감지 한다
* 출차 알림 notification을 직원 어플리케이션으로 전송한다
* 직원이 notification에 대한 승낙을 한 경우 해당 직원의 gps 정보를 실시간으로 감지한다
* 미리 정의 해둔 식당의 geofence 반경으로 사용자의 gps 위치 정보가 진입 할 경우 차량 database의 접근여부 값을 업데이트 한다
* 식당 어플리케이션에서 차량 근접 여부 변경 이벤트를 감지하고 해당 사항을 반영하여 화면에 출력한다
  + - 1. Notification에 대한 응답 미 존재 시
* Notification 전송 후 timer를 작동 시켜 10000ms 이내에 사용자로부터 응답이 없는 경우 notification을 재전송 한다.
  + 1. 무전 Client & 무전 Server
       1. 알고리즘
* android Media recoder로 음성 데이터를 녹음한다
* UDP socket 통신을하여 음성 데이터를 서버로 전송한다
* 서버 측에서 음성데이터를 전송한 사용자와 동일 구역에 존재하는 사용자 리스트 추출한다
* 추출된 리스트의 사용자 수 만큼 쓰레드를 생성하여 클라이언트와 connection한다
* connection이 맺어진 client로 서버에서 음성 데이터와 서버에서 변환된 음성의 텍스트 데이터를 send한다
* client가 음성데이터를 receive하여 android Media player를 통해 음성 데이터를 출력한다
  + - 1. Packet Corruption 예방
* 클라이언트와 서버, 서버와 클라이언트간 packet을 전송하는 과정에 있어, data corruption이 일어날 위험이 존재하므로, Checksum을 packet에 추가하 확인하는 과정을 진행 할 것이다.
  + - 1. Packet Loss 예방
* 전송한 packet에 대한 loss가 일어날 위험성이 존재하므로, 클라이언트와 서버간 전송되는 packet에 대하여 sequence number을 부여하고, 해당 packet에 대한 Ack packet을 생성하여 상호간 loss된 packet의 여부를 확인하는 과정을 진행 할 것이다.
  + - 1. 무전 내역 미 청취 사용자에 대한 방안
* 서버로부터 전송 받은 음성 데이터를 local database에 저장해 놓고, 원하는 시기에 다시 스트리밍 할 수 있도록 한다.

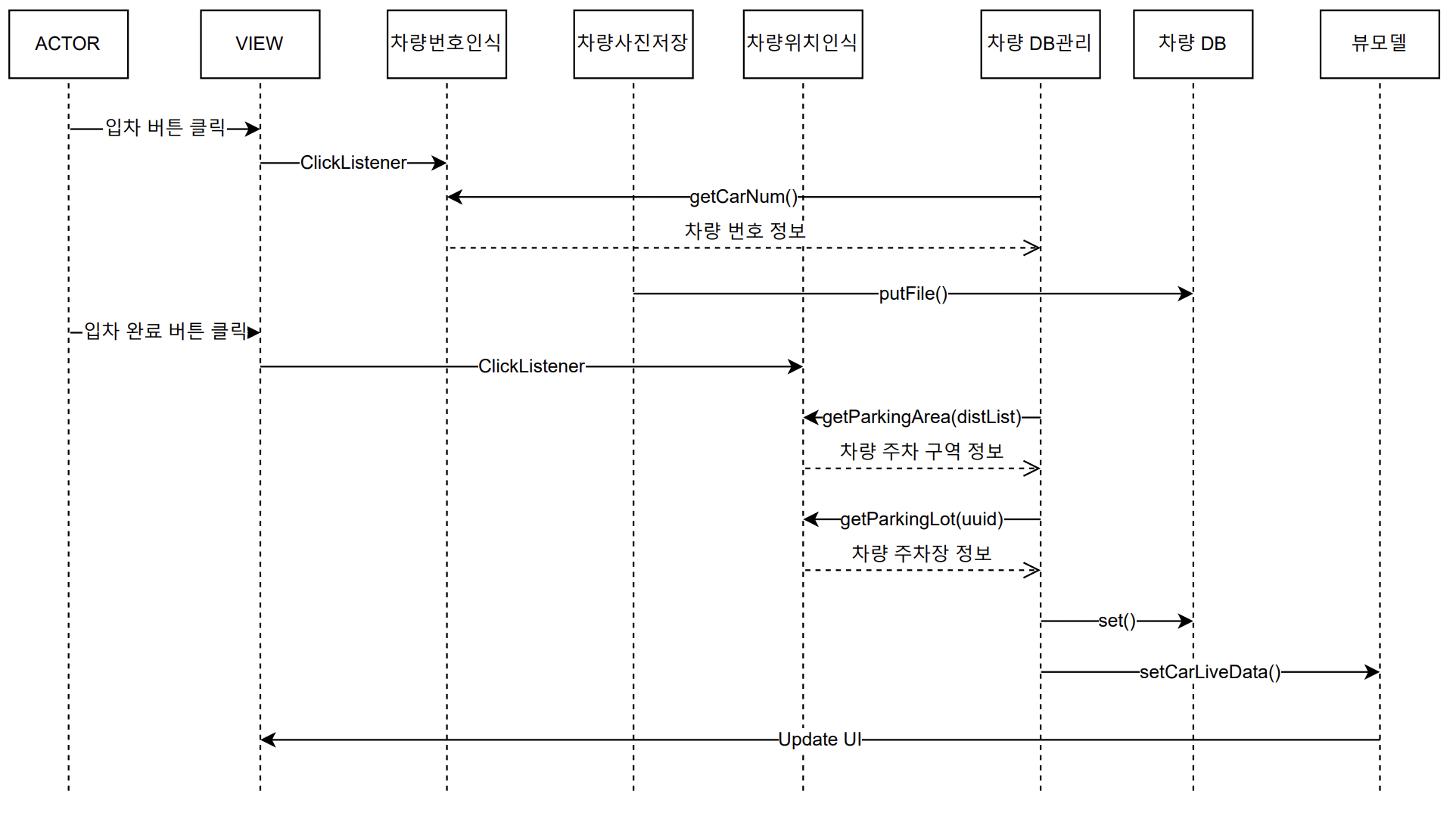
1. **Sequence Diagram**
   1. Use-Case-Sequence Diagram
      1. 로그인



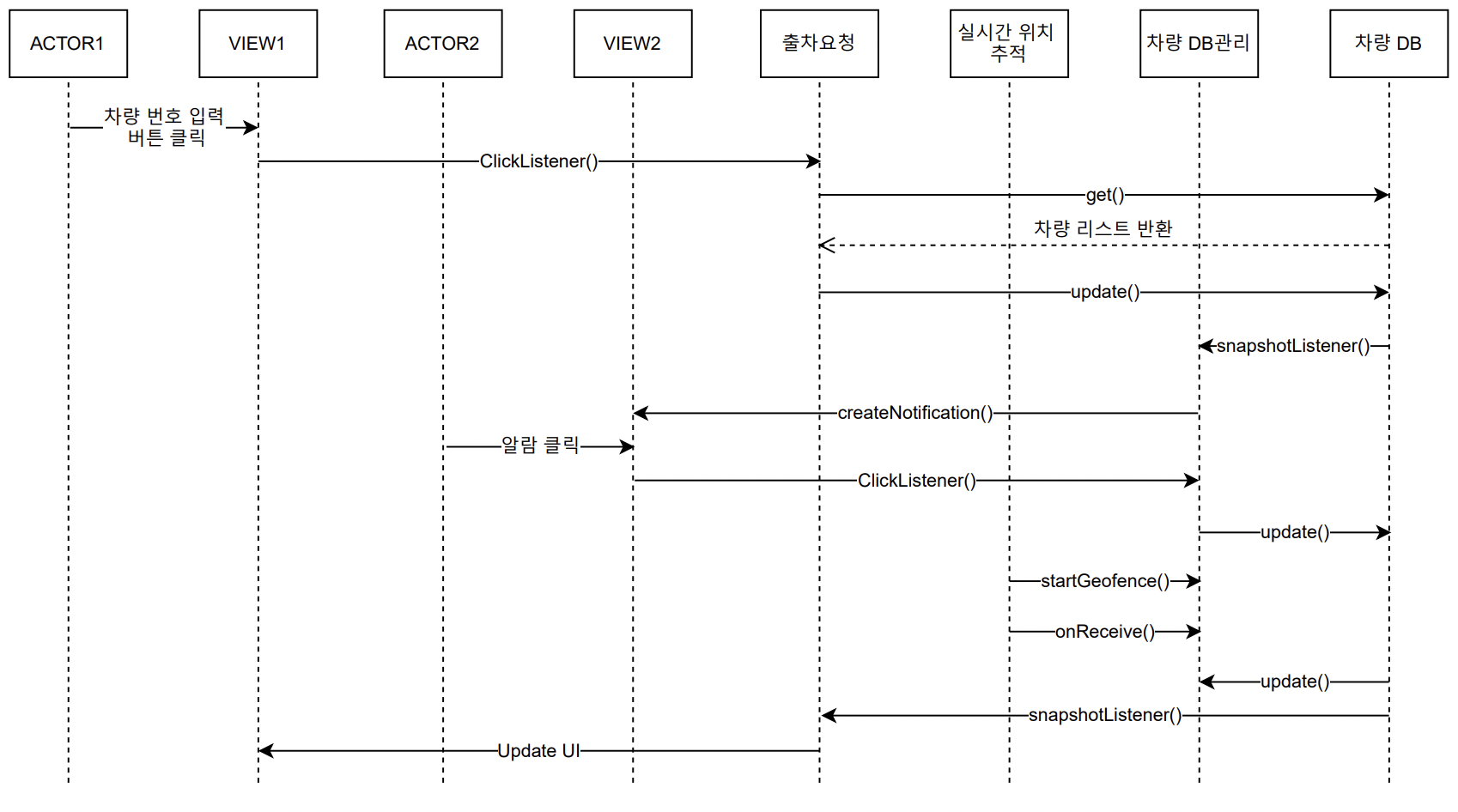
* + 1. 회원가입



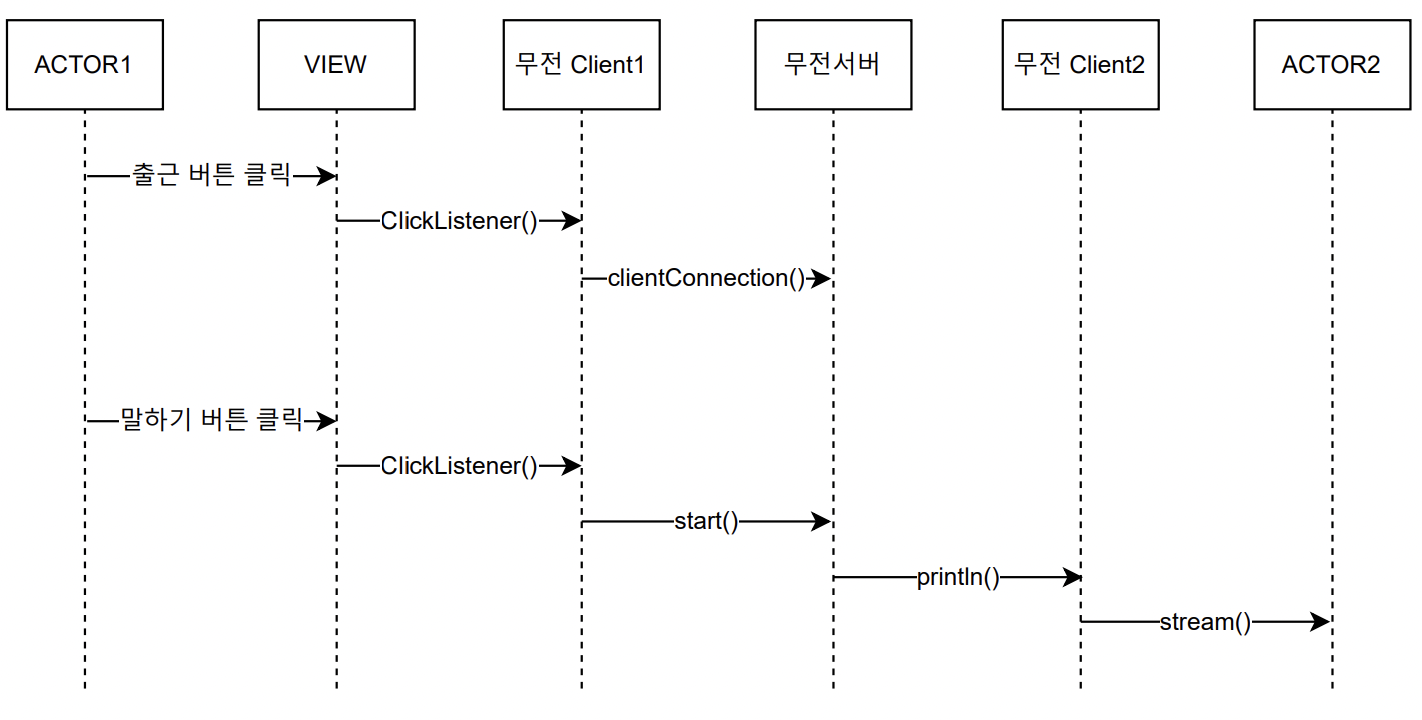
* + 1. 입차



* + 1. 출차



* + 1. 무전



1. **User Interface Configuration**
   1. **로그인, 회원가입**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **화면ID** | | LG\_SU\_00 | | **화면명** | | 회원가입 로그인 | |
| **관련 유스케이스 ID** | | 회원가입, 로그인 | | | | | |
| **관련 시퀀스도 ID** | | 3.3.1, 3.3.2 | | | | | |
| **화면유형** | | 입력 | | **메뉴경로** | | 초기 화면 | |
| **화면개요** | | 어플리케이션을 사용하기 위해 회원가입 또는 회원가입이 완료되었던 사용자에 한하여 로그인을 처리하는 화면 | | | | | |
|  | | | | | | | |
| 입출력 항목 | | | | | | | |
| **항목명** | 컨트롤명 | | 타입 및 길이 | | **속성** | | **Validation Check** |
| 로고 이미지 | logo\_img | | ImageView | | O / R | | - |
| 근무지 입력 | restaurant\_input\_text | | String(10) | | I / E | | 필수항목 |
| 카카오 로그인 | kakao\_login\_btn | | ImageView | | I | | - |
| \*속성 I:Input, O:output, R:ReadOnly, E:Editable, H:Hidden | | | | | | | |
| 처리내용 | | | | | | | |
| * **근무지 입력**   : 사용자가 근무하는 근무 지역을 입력한다.   * **카카오 로그인**   : 근무지 입력란에 valid한 정보를 입력할 경우 버튼이 활성화되며, 기존에 회원가입 과정이 진행되어 데이터베이스에 근무지와, 카카오톡 닉네임 정보에 해당하는 사용자가 존재하는 경우, 바로 메인 화면으로 전환되며, 첫 회원가입을 진행하는 사용자의 경우 카카오톡 아이디, 비밀번호를 입력할 수 있는 화면으로 전환된다. | | | | | | | |
| 기술적 고려사항 | | | | | | | |
| * 회원가입 및 로그인 과정 중 어플리케이션을 비정상적으로 종료할 경우 처리되던 진행 과정은 복구되지 않는다. | | | | | | | |

* 1. **식당 어플리케이션**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **화면ID** | | RT\_EX\_00 | | **화면명** | | 출차 요청 | |
| **관련 유스케이스 ID** | | 근접 차량 확인, 출차 요청 | | | | | |
| **관련 시퀀스도 ID** | | 3.1.4 | | | | | |
| **화면유형** | | 입력, 요청 조회 | | **메뉴경로** | | 메인 화면 | |
| **화면개요** | | 식당의 차 주가 차량번호 끝 네 자리를 입력하여 현장 주차 직원에게 출차 요청 전송 및 출차 요청 수락, 차량 근접 유/무 확인 화면 | | | | | |
| **테이블이(가) 표시된 사진  자동 생성된 설명** | | | | | | | |
| **입출력 항목** | | | | | | | |
| **항목명** | **컨트롤명** | | **타입 및 길이** | | **속성** | | **Validation Check** |
| 출차 현황 | car\_exit\_title | | String(4) | | R | | - |
| 차량 리스트 | car\_exit\_list | | RecyclerView | | O / R | | - |
| 차량번호 입력 | car\_num\_input\_text | | String(4) | | I / E | | 필수항목 |
| 출차 요청 | car\_exit\_request\_btn | | Button | | I | | - |
| \*속성 I:Input, O:output, R:ReadOnly, E:Editable, H:Hidden | | | | | | | |
| **처리내용** | | | | | | | |
| * **차량 리스트**   : 입력한 차량 번호에 대하여 현장 주차 직원이 출차 요청을 수락한 경우 차량 번호와 ‘출차중’ 이라는 문자열이 리스트에 추가되고, 식당에 근접한 경우 ‘출차중’ 문자열이 근접 이라는 문자열로 변경된다.   * **차량번호 입력**   : 차량 주인이 입력한 차량번호 4자리에 대하여 올바른 입력 형태인지 체크한다. 올바른 입력 형태가 아닌 경우 오류 메세지를 출력한다.   * **출차 요청**   : 입력한 차량번호가 valid한 경우 버튼이 활성화되고, 출차 요청을 전송한다. 해당 번호가 데이터베이스에 등록 되어 있지 않은 경우 또는 차량 번호에 해당하는 차량이 출차중, 근접 상태인 경우 화면에 오류 메세지를 출력한다. | | | | | | | |
| **기술적 고려사항** | | | | | | | |
| * 입력한 차량이 입차 상태로 존재하는 경우에만 출차 준비 상태로 변경한다. * 출차 준비 상태로 변경된 차량 정보를 해당 차량의 출차 요청이 이뤄진 식당에서 근무하는 직원의 어플리케이션으로만 notification을 전송해야 한다. * 출차 요청된 차량에 대하여 notification을 전송 받은 직원 중 한명만 출차 요청에 응답을 해야한다. * 출차 중인 차량이 식당에 근접한 경우 직원의 GPS위치 정보를 geofence 반경에 진입한 경우 실시간으로 데이터 베이스에 업데이트 해야한다. * 업데이트된 데이터베이스 정보를 인식하여 실시간으로 차량 리스트를 업데이트 해야한다. | | | | | | | |

* 1. **무전**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **화면ID** | | RD\_CM\_00 | | **화면명** | | 무전 | |
| **관련 유스케이스 ID** | | 근접 차량 확인, 출차 요청 | | | | | |
| **관련 시퀀스도 ID** | | 3.1.4 | | | | | |
| **화면유형** | | 입력, 요청 조회 | | **메뉴경로** | | 메인화면의 무전 탭 선택 | |
| **화면개요** | | 식당의 차 주가 차량번호 끝 네 자리를 입력하여 현장 주차 직원에게 출차 요청 전송 및 출차 요청 수락, 차량 근접 유/무 확인 화면 | | | | | |
|  | | | | | | | |
| **입출력 항목** | | | | | | | |
| **항목명** | **컨트롤명** | | **타입 및 길이** | | **속성** | | **Validation Check** |
| 무전 | radio\_title | | String(2) | | R | | - |
| 마이크 | radio\_record\_btn | | ImageView | | I | | (무전 전송시)  필수항목 |
| 무전 리스트 | radio\_received\_title | | String(5) | | R | | - |
| 무전 누락 리스트 | radio\_recevied\_list | | RecyclerView | | O | | - |
| 무전 길이 | radio\_length\_text | | String(5) | | R | | - |
| 재생 아이콘 | radio\_play\_btn | | ImageView | | O | | (무전 청취시)  필수항목 |
| \*속성 I:Input, O:output, R:ReadOnly, E:Editable, H:Hidden | | | | | | | |
| **처리내용** | | | | | | | |
| * 마이크   : 사용자의 발화 내용을 녹음하고 사용자와 같은 무전 서버 내에 존재하는 사용자들에게 전송한다.   * 무전 누락 리스트   : 사용자와 동일한 무전 서버 내에 존재하는 다른 사용자들이 전송한 음성 데이터를 무전 누락 리스트에 최근에 도착한 순서대로 추가한다.   * 무전 길이   : 사용자가 전송 받은 음성 데이터들의 재생 시간 정보가 보여진다.   * 재생 아이콘   : 사용자가 전송 받은 음성 데이터를 스피커를 통해 송출한다. | | | | | | | |
| **기술적 고려사항** | | | | | | | |
| * 다른 사용자가 무전 서버로 전송한 음성 데이터가 존재하는 경우 실시간으로 같은 구역에 존재하는 사용자들 에게만 전송한다. * 사용자가 녹음한 음성 데이터 패킷에 대한 loss가 일어나지 않도록 socket통신을 구현한다. * 다른 사용자가 이전에 전송한 무전 누락 리스트에 존재하는 음성 데이터를 재생하는 경우 현재 실시간으로 전송 받아 송출 되려고 하는 음성 데이터와 겹치지 않도록 예외처리 한다. | | | | | | | |

* 1. **출차 요청 승낙**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **화면ID** | | RQ\_EX\_00 | | **화면명** | | 출차 요청 확인 | |
| **관련 유스케이스 ID** | | 근접 차량 확인, 출차 요청 | | | | | |
| **관련 시퀀스도 ID** | | 3.1.4 | | | | | |
| **화면유형** | | 입력, 요청 확인 | | **메뉴경로** | | 출차 요청 notification | |
| **화면개요** | | 식당 어플리케이션에서 출차 요청이 직원 어플리케이션으로 전송되었을 때, 특정 직원의 출차 승낙에 대한 응답을 최종으로 확인하는 화면 | | | | | |
| **텍스트이(가) 표시된 사진  자동 생성된 설명** | | | | | | | |
| **입출력 항목** | | | | | | | |
| **항목명** | **컨트롤명** | | **타입 및 길이** | | **속성** | | **Validation Check** |
| 출차승낙알람 | exit\_request\_dialog | |  | | IO | | OK or NO 선택 |
| \*속성 I:Input, O:output, R:ReadOnly, E:Editable, H:Hidden | | | | | | | |
| **처리내용** | | | | | | | |
| * **출차승낙알람**   : 차량 주인이 출차 요청을 진행한 이후 주차 직원이 어플리케이션에 전송된 notification을 클릭 시 나오는 dialog 화면이다. ‘NO’ 를 클릭한 경우 차량 관리 화면이 보여지며, ‘YES’를 클릭한 경우 차량 관리 화면에 해당 차량의 상태 정보가 ‘출차중’ 메세지로 변경된다.   * ‘YES’를 누른 직원이 존재한 이후에 다른 직원이 ‘YES’를 누르게 되는 경우 이미 출차 직원이 선정되었다는 오류 메세지를 화면에 출력한다. | | | | | | | |
| **기술적 고려사항** | | | | | | | |
| * 특정 직원이 ‘OK’를 누른 경우, 데이터베이스의 차량 상태 정보를 실시간으로 업데이트 해야 한다. * 직원 어플리케이션으로 출차 요청은 식당 어플리케이션에서 요청과 동시에 이뤄져야 한다. | | | | | | | |

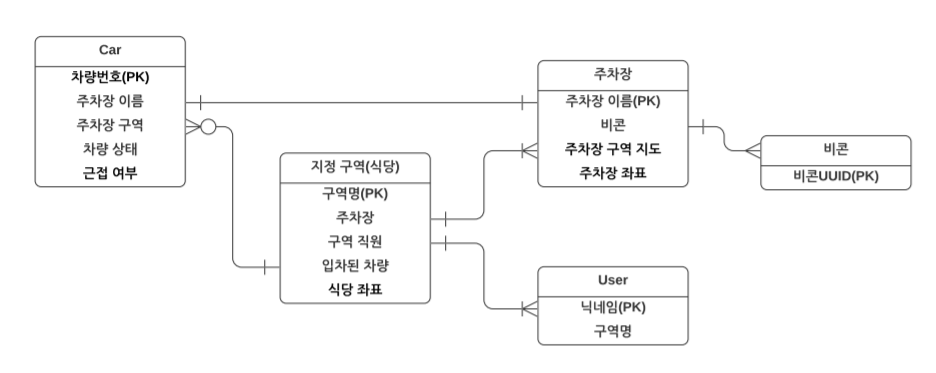
* 1. **차량 리스트**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **화면ID** | | CR\_LT\_00 | | **화면명** | | 차량 관리 | |
| **관련 유스케이스 ID** | |  | | | | | |
| **관련 시퀀스도 ID** | | 3.1.3, 3.1.4 | | | | | |
| **화면유형** | | 요청 조회 | | **메뉴경로** | | 메인화면의 차량 관리 메뉴 선택 / 출차 승낙 알람의 ‘OK’클릭/ 출차 승낙 알람의 ‘NO’클릭 | |
| **화면개요** | |  | | | | | |
|  | | | | | | | |
| **입출력 항목** | | | | | | | |
| **항목명** | **컨트롤명** | | **타입 및 길이** | | **속성** | | **Validation Check** |
| 돋보기 | search\_image | | ImageView | | R | | - |
| 주차장 스피너 | park\_lot\_spinner | | Spinner | | I | | (주차장 선택시)  필수항목 |
| 차량 리스트 | car\_list\_recycler\_view | | RecyclerView | | O/R | |  |
| 남은 주차 공간 | park\_space\_text | | String(3~5) | | O/R | | - |
| 차량 상태 | car\_state\_text | | String(2~4) | | O/R | | - |
| 차량 번호 | car\_num\_text | | String(7~8) | | O/R | | - |
| 담당자 | car\_person\_charge\_text | | String(4~) | | O/R | | - |
| 주차구역 | car\_section\_text | | String(7~8) | | O/R | | - |
| \*속성 I:Input, O:output, R:ReadOnly, E:Editable, H:Hidden | | | | | | | |
| **처리내용** | | | | | | | |
| * 주차장 스피너   : 미리 저장되어 있는 식당 주변 주차장 리스트가 화면에 띄워진다. 각 주차장 명을 클릭할 경우 해당 주차장에 등록되어 있는 차량 리스트 및 남은 주차 공간 정보가 갱신된다.   * 차량 리스트   : 특정 차량이 입차 차량으로 등록된 경우 정보가 추가 되어 리스트가 갱신된다.   * 남은 주차 공간   : 선택된 주차장의 전체 주차 공간 대비 남은 주차 공간의 자리 수를 띄워준다.  : 차량의 입차, 출차대기, 출차중 상태중 데이터베이스에 저장되어 있는 값에 따라 알맞게 띄워준다.   * 차량 번호, 차량 상태, 담당자, 주차구역   : 특정 직원이 담당한 입차 또는 출차 중인 차량의 번호와 차량 상태 정보, 주차된 구역 정보가 매핑 되어 보여진다. | | | | | | | |
| **기술적 고려사항** | | | | | | | |
|  | | | | | | | |

* 1. **입차**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **화면ID** | CR\_ET\_00 | | **화면명** | | 차량 입차 | |
| **관련 유스케이스 ID** | 번호판 수기 입력, 카메라 번호판 인식, 위치 자동 인식, 주차장 위치 선택, 입차 차량 등록, 차량 사진 촬영 | | | | | |
| **관련 시퀀스도 ID** | 3.1.3 | | | | | |
| **화면유형** | 등록 | | **메뉴경로** | | 메인화면 | |
| **화면개요** | 차량의 손상 여부 사진 촬영 기능 및 차량 번호, 차량이 주차된 주차장, 주차장의 구역 정보 데이터를 입력하고 입차 차량으로 등록하기 위한 화면 | | | | | |
| **텍스트이(가) 표시된 사진  자동 생성된 설명** | | | | | | |
| **입출력 항목** | | | | | | |
| **항목명** | **컨트롤명** | **타입 및 길이** | | **속성** | | **Validation Check** |
| 입차 | car\_enter\_title | String(2) | | R | | - |
| 번호판 촬영 | car\_num\_camera | ImageView | | I | |  |
| 차량 번호 입력란 | car\_num\_input | String(7~8) | | IO/E | | 필수항목 |
| 위치 자동 인식 | gps\_beacon | ImageView | | I | |  |
| 주자창 스피너 | Park\_lot\_spinner | Spinner | | IO | | 필수항목 |
| 차량 구역 | car\_section | String(2~3) | | O/R | | 필수항목 |
| 입차완료 | complete\_btn | Button | | I | | - |
| 사진촬영 | car\_capture\_btn | Button | | I | | - |
| \*속성 I:Input, O:output, R:ReadOnly, E:Editable, H:Hidden | | | | | | |
| **처리내용** | | | | | | |
| * 번호판 촬영   : 해당 버튼 클릭 시 차량의 번호판을 캡쳐 할 수 있는 카메라 화면이 보여진다.   * 차량 번호 입력란   : 카메라를 통한 번호판 촬영이 잘 이뤄지지 않은 경우 수동으로 차량 번호를 기입 할 수 있다. 따라서 입력한 값이 차량 번호 형식에 맞게 입력 되었는지 체크한다.   * 위치 자동 인식   : 해당 버튼 클릭시, 차량이 위치한 주차장 위치 및 주차된 주차 구역 정보를 인식하여 주차장 스피너 및 차량 구역 입력 란에 보여준다.   * 입차 완료   : 차량 번호, 차량 위치 정보가 모두 올바른 형식으로 입력된 경우 해당 버튼 클릭 시, 데이터 베이스에 입력된 정보가 추가된다.   * 사진촬영   : 차량의 외관을 촬영할 수 있는 버튼으로, 해당 버튼 클릭 시 카메라 화면이 보여진다. 카메라를 통해 촬영 된 사진은 storage에 업로드 된다. | | | | | | |
| **기술적 고려사항** | | | | | | |
| * Beacon을 통한 정확한 차량 위치 인식을 위하여 RSSI 신호가 간섭 및 굴절되지 않게 벽, 기둥 등에 가려져 설치되지 않도록 한다. | | | | | | |

1. **ER Diagram**



1. **Programming Strategy**
   1. **Programming Language**

Application : Kotlin, Radio Server : Java

* 1. **Programming Tool**

Android Studio

1. **Others**
   1. **실험 장소 설정**
      1. **주차장**

실제 주차장에서 실험하기에는 공간 확보가 어렵고, 차들의 이동으로 위험하기 때문에 새천년관 501호에서 실험을 할 예정입니다.

선정한 이유는 501호에 존재하는 책상 간의 간격과 한 자동차 구역의 넓이가 거의 일치하기 때문입니다.

501호 공간의 넓이는 14.5\*6.7\*2.5(m)로 비콘 4개를 사용하는 경우에 알맞은 중소형 공간이기 때문에 선정하였습니다.

주차장의 공간은 이보다 더 넓은 공간이지만, 그에 따라 비콘의 개수가 선형적으로 비례 하진 않지만 일정 이상의 개수가 되면 실험실 공간과 같이 측정 가능할 것으로 예상됩니다.

* + 1. **식당**

**지도이(가) 표시된 사진

자동 생성된 설명**

위 사진은 네이버 지도로 측정한 건국대학교 새천년관, 건국대학교 신공학관 사이의 거리입니다. 직선거리로 약 300m 의 거리에 위치한 두 공간을 각 각 주차장과 식당을 대체할 장소로 지정할 것입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

위사진은 <https://developer.android.com/training/location/geofencing?hl=ko>

에 기술된 내용으로, 식당에 대한 geofence 설정 시 최상의 결과를 얻기 위하여 위와 같이 설정했습니다.