시각장애인을 위한 버스 탑승 시스템 ‘We Tayo’

Development of Bus Boarding Service for the Visually Impaired ‘We Tayo’

팀장 : 강석원
팀원 : 박형근, 이지수, 홍의성

목 차

1 졸업 연구 개요
- 1.1 지난 지적사항에 대한 답변
- 1.2 연구 개발 배경
- 1.3 연구 개발 목표

2 관련 연구 및 사례
- 2.1 관련 연구 및 사례1
- 2.2 관련 연구 및 사례2
- 2.3 비교 분석

3 시스템 수행 시나리오
- 3.1 시스템 수행 시나리오

4 시스템 구성도
- 4.1 하드웨어 구성도
- 4.2 소프트웨어 구성도

5 시스템 모듈 상세설계
- 5.1 하드웨어 모듈 상세설계
- 5.2 소프트웨어 모듈 상세설계 (서버)
- 5.3 소프트웨어 모듈 상세설계 (앱)

6 개발 환경 및 방법
- 6.1 하드웨어 개발 환경
- 6.2 소프트웨어 개발 환경
- 6.3 시스템 개발 방법

7 데모 환경 설계
- 7.1 데모 환경 구성
- 7.2 데모 순서

8 업무 분담 및 수행 일정
- 8.1 업무 분담
- 8.2 수행 일정
- 8.3 진행 상황

9 참고 문헌
- 9.1 참고 문헌

10 질의응답
- 10.1 질의응답

1

졸업 연구 개요

1.1 지난 지적사항에 대한 답변
1.2 연구 개발 배경
1.3 연구 개발 목표

1.1 지난 지적사항에 대한 답변

시각장애인 모바일 앱 접근성 조사하고 적용할 것

국립장애인도서관에 정의된 **모바일 애플리케이션 접근성**(Mobile Application Accessibility)의 **7가지 준수사항**(대체 텍스트, 초점 등)과 **8가지 권장사항**(알림 기능, 범용 폰트 이용 등)을 참고하여 애플리케이션을 개발한다.

이윤혁. "시각장애인을 위한 앱(App) 디자인 사용성 향상 연구." 에 나와있는 시각장애 보조 앱 사례조사 및 GUI 분석을 참고하여 5개의 앱('Walk with You', '설리번+', '점자키보드', '봄', '목소리 천사')의 **장점은 살리고 단점은 보완**하여 애플리케이션을 개발한다.

1.1 지난 지적사항에 대한 답변

[사용자 인터페이스 일관성] 인터페이스 요소를 사용자가 다시 학습하지 않아도 되는 일관성 있는 배치를 제공

[대체 텍스트] 텍스트 아닌 콘텐츠는 대체 가능한 텍스트를 함께 제공

[초점(focus)] 모든 객체에 초점을 적용하고 초점은 위에서 아래로 이동될 수 있도록 제공

[명도 대비] 전경과 배경을 구분할 수 있도록 고대비 제공

1.1 지난 지적사항에 대한 답변

데모방법과 데모환경을 구체화 할 것

(1) 학교 프로그램인 지역사회연계를 활용해 **실제 주행중인 버스** 한대에 **부착해 전체 시나리오대로 동작하는 영상을 촬영**해 데모 발표시 영상을 송출한다.

(2) (1)이 안될경우를 대비해 **직접 자차를 사용**하여 **전체 시나리오대로 동작하는 영상을 촬영**해 데모 발표시 영상을 송출한다.

(3) (1)과 (2)는 별도로 시스템의 핵심 기능인 **무선하차벨의 동작**과 **실제 애플리케이션의 구동**을 직접 시연한다.

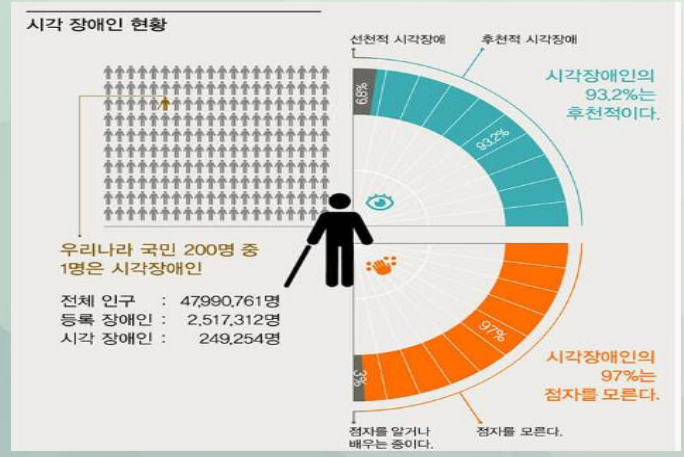
1.1 지난 지적사항에 대한 답변

교내 지원 프로그램 – 지역연계형 캡스톤 디자인 조사 할 것

[2021.01.29] 2021학년도 **지역연계형** 캡스톤 디자인 신청 완료

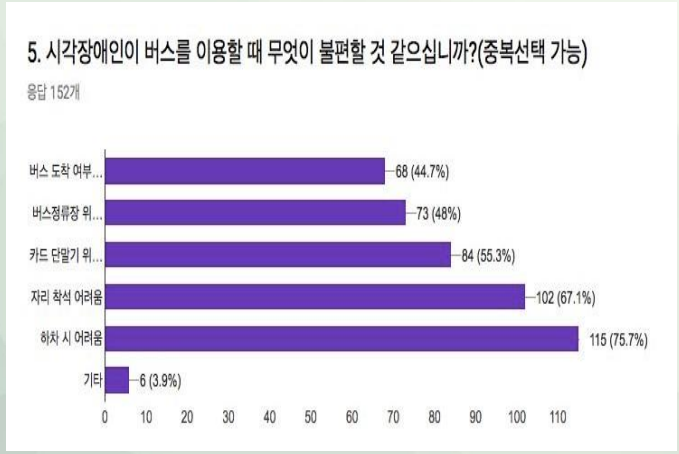
[2021.03.04] 지역연계형 캡스톤 디자인 **최종 선발**

1.2 연구 개발 배경



국내 시각 장애인 현황

우리나라 **국민 200명 중 1명은 시각장애인**
그 중 93.2% 후천적 시각장애를 갖고 있음



시각 장애인의 버스 이용시 불편사항

시각 장애인이 버스를 이용할 때 가장 불편함을 느끼는 1순위는 **'하차 시 어려움'** 이다.
<오마이 뉴스 2017년 조사자료>

시각장애인의 버스 탑승 국민 청원 내용

시각장애인의 **이동권 보장**을 요구하는 국민 청원 내용 3738명의 청원
<2019.11.27 청원 시작>

1.2 연구 개발 배경

"일단 버스를 타고 이동하는 것은 고사하고 내가 원하는 버스를 기다리는 시간만 평균 40분 ~ 50분 소요된다."

(MBC 소수의견 인터뷰 中)

"기존의 버스 알림 어플, 정류장 음성 시스템은 오류가 잦고 업데이트가 느려 실질적으로 도움이 되지 않는다."

(한국 시각장애인 연합회 참고 의견 中)

"우리(시각 장애인)들도 비장애인들과 다름없이 남 눈치 보지 않으면서 일상적인 삶을 사라고 싶다."

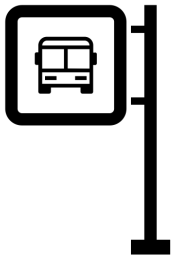
(청와대 국민청원 발췌문 中)

이동권은 시각장애인의 사회참여를 위해서 반드시 보장되어야 하는 기본권리

그들도 우리와 똑같은 사회구성원으로서 인정받으면서 살아가야함

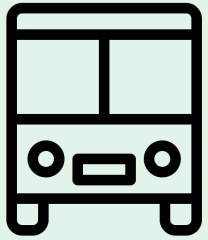
시각장애인의 주도적인 버스 이용을 위해 "WeTayo"를 개발 할 필요성을 느낌

1.3 연구 개발 목표



위치 정보 분석

사용자(시각장애인)의 위치와 버스 정류장의 위치를 정확하게 파악하여 가장 가까운 정류장에 쉽게 접근할 수 있도록 유도



탑승 여부 전송

사용자(시각장애인)의 APP에서 탑승을 희망하는 버스를 선택하면 가장 먼저 도착하는 버스 기사님의 APP의 탑승 여부를 전송

APP으로 하차벨 동작

기존 버스에 장착된 하차벨과 비콘 센서를 연동하여 사용자(시각장애인)의 APP에서 무선으로 동작

UI/UX 적극 반영

사용자가 시각장애인인점을 고려하여 사용자 경험을 적극 반영

2

관련 연구 및 사례

2.1 관련 연구 및 사례1
2.2 관련 연구 및 사례2
2.3 비교 분석

2.1 관련 연구 및 사례1

시민기자단 솔루션

- 음성 수신기를 통해 버스 도착과 승차 위치 알림
- 음성 수신기를 통해 버스 기사에게 하차 요청
- 음성 수신기를 통해 정류장의 정보 알림
- 실시간 버스 정보 확인 불가
- 사용자 위치에 기반한 근처 정류장 선별 안함

https://www.youtube.com/watch?v=saoX-lR-iJ0

2.2 관련 연구 및 사례2

삼성 투모로우 솔루션
(시각 장애인 지우 이야기)

- APP을 통해 버스 선택 후 도착 알림
- APP을 통해 버스 기사에게 하차 요구 알림
- 실시간 버스 정보 확인 가능
- 사용자 위치에 기반한 근처 정류장 선정 안함

https://www.youtube.com/watch?v=gqGAf2EaIBs

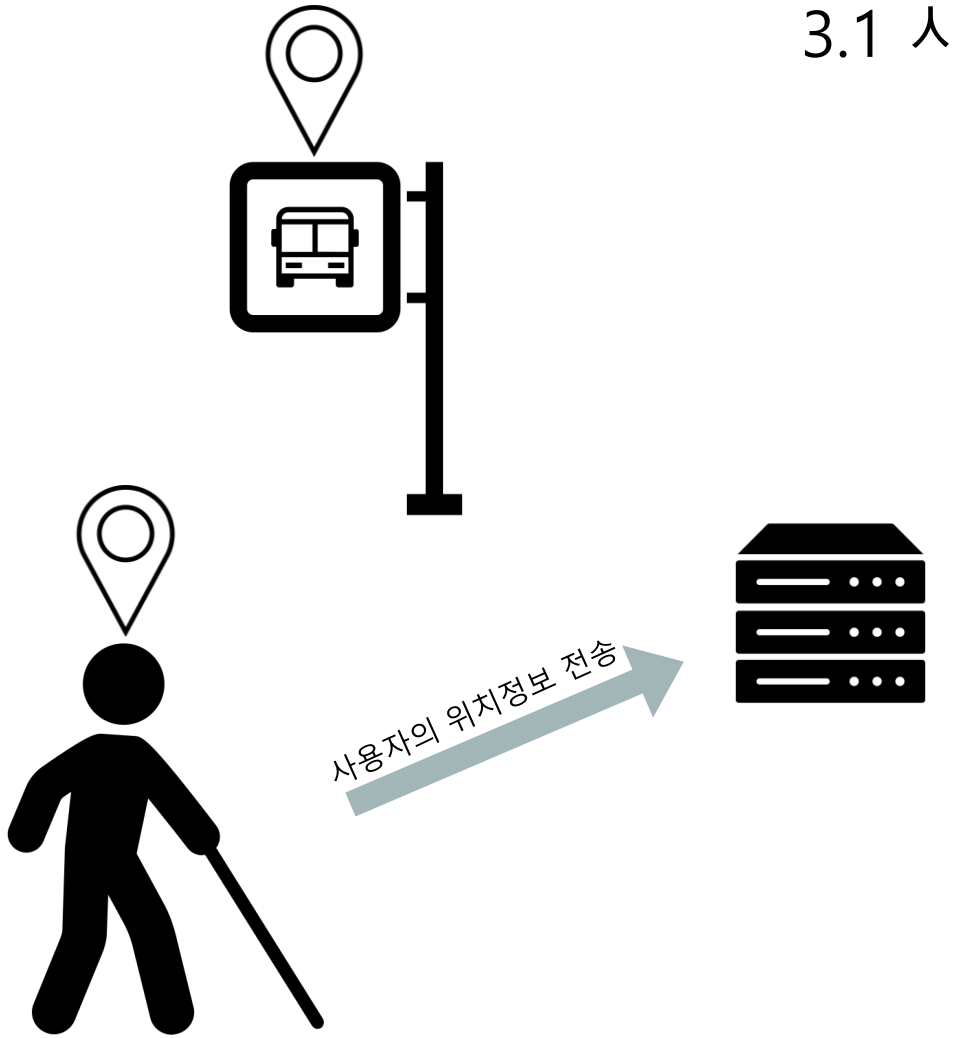
2.3 비교 분석

	버스정보 알림 스피커	무선 하차벨	실시간 버스 정보	APP을 통한 하차벨 동작	사용자 위치기반 정류장 선정
We Tayo	O	O	O	O	O
삼성 투모로우 솔루션	O	O	O	O	X
시민 기자단 솔루션	O	O	X	X	X

3 시스템 수행 시나리오

3.1 시스템 수행 시나리오

3.1 시스템 수행 시나리오



01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**한다.

3.1 시스템 수행 시나리오

01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**한다.

02 가장 가까운 정류장으로 식별된 정류장의 **위치 정보를 사용자 앱에 전송**

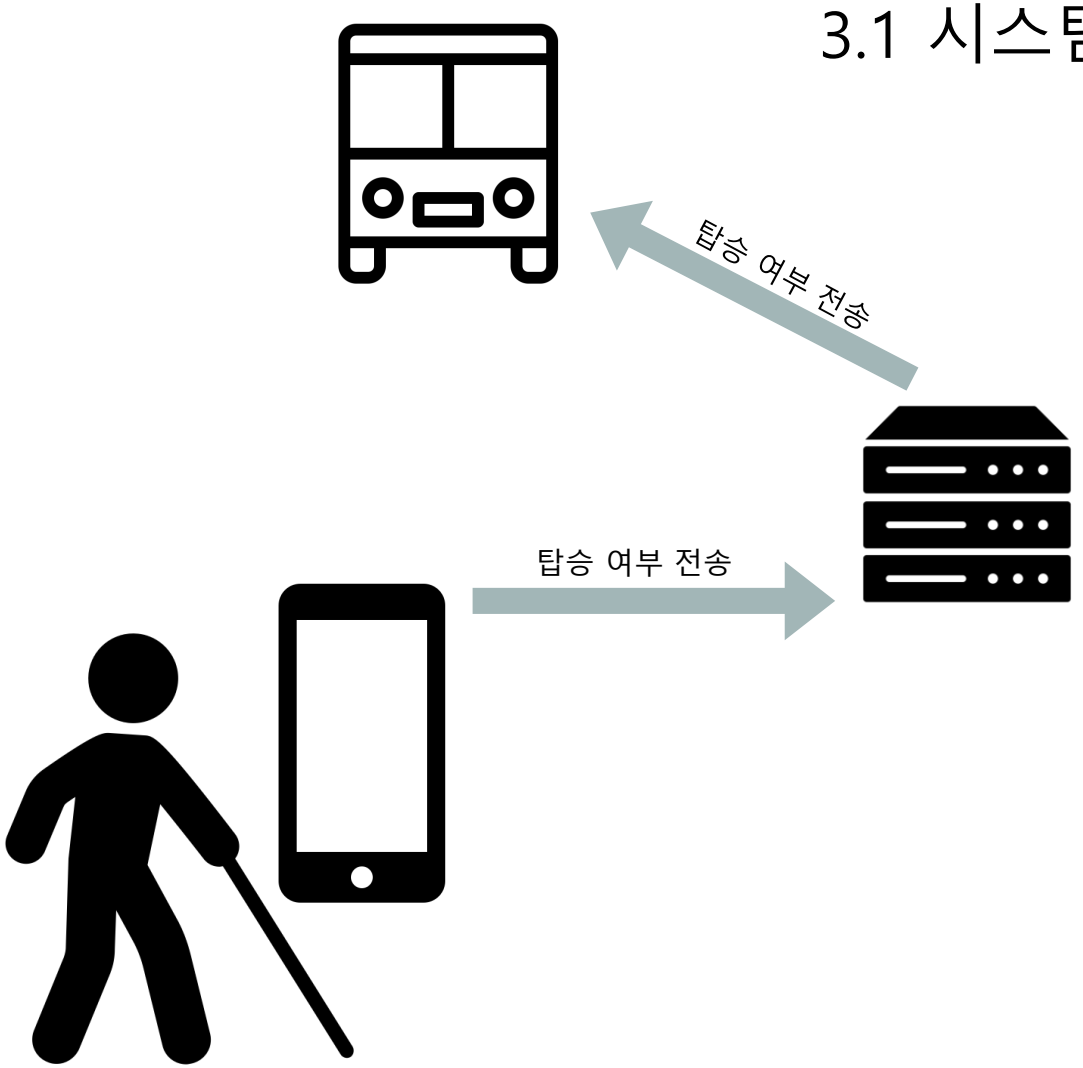
3.1 시스템 수행 시나리오

01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**

02 가장 가까운 정류장으로 식별된 정류장의 **위치 정보를 사용자 앱에 전송**

03 사용자 앱에 서버로부터 받은 정류장 위치의 도착 **버스를 좌우로 슬라이드하면 음성으로 안내하고 탑승할 버스를 선택**

3.1 시스템 수행 시나리오



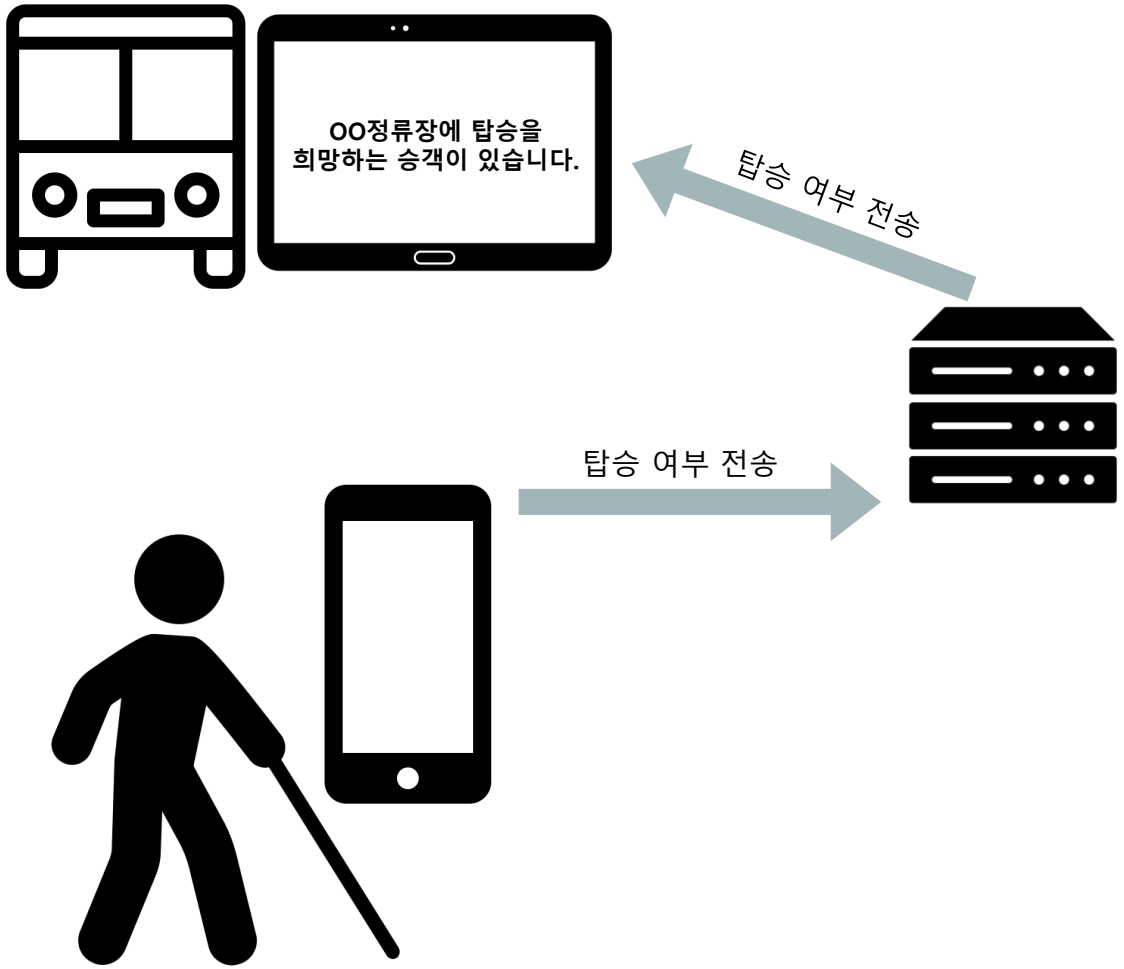
01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**

02 가장 가까운 정류장으로 식별된 정류장의 **위치 정보를 사용자 앱에 전송**

03 사용자 앱에 서버로부터 받은 정류장 위치의 도착 **버스를 좌우로 슬라이드하면 음성으로 안내하고 탑승할 버스를 선택**

04 탑승할 버스를 선택하여 **탑승여부를 버스기사님의 스마트폰 앱으로 전송**한다.

3.1 시스템 수행 시나리오



01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**

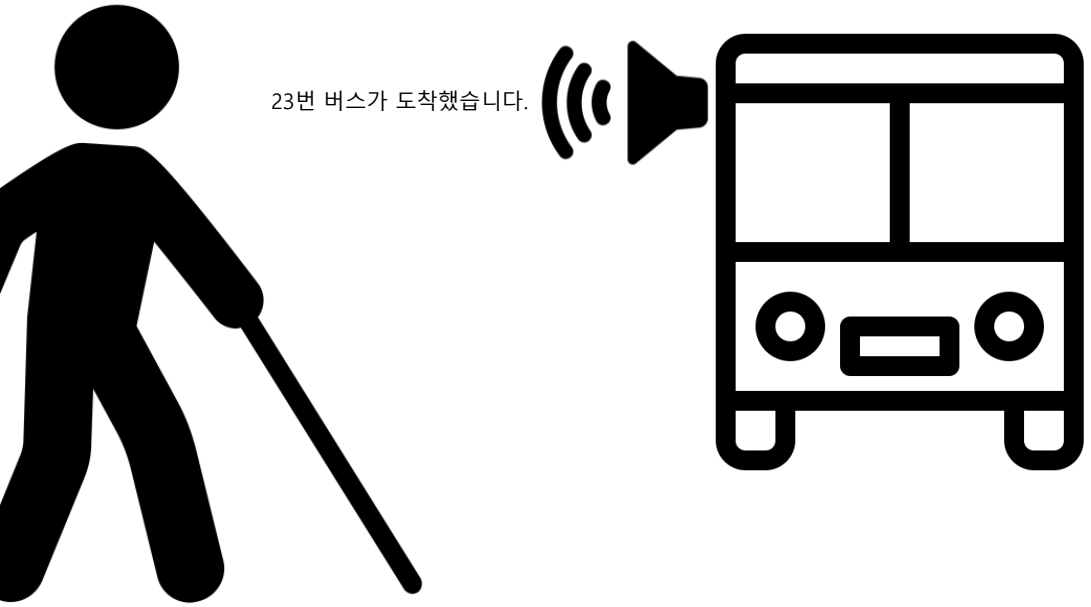
02 가장 가까운 정류장으로 식별된 정류장의 **위치 정보를 사용자 앱에 전송**

03 사용자 앱에 서버로부터 받은 정류장 위치의 도착 **버스를 좌우로 슬라이드하면 음성으로 안내하고 탑승할 버스를 선택**

04 탑승할 버스를 선택하여 **탑승여부를 버스기사님의 스마트폰 앱으로 전송**한다.

05 버스기사님의 스마트폰 앱을 통해 해당 정류장에 탑승을 희망하는 **시각장애인의 존재를 인식**시킨다.

3.1 시스템 수행 시나리오



01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**

02 가장 가까운 정류장으로 식별된 정류장의 **위치 정보를 사용자 앱에 전송**

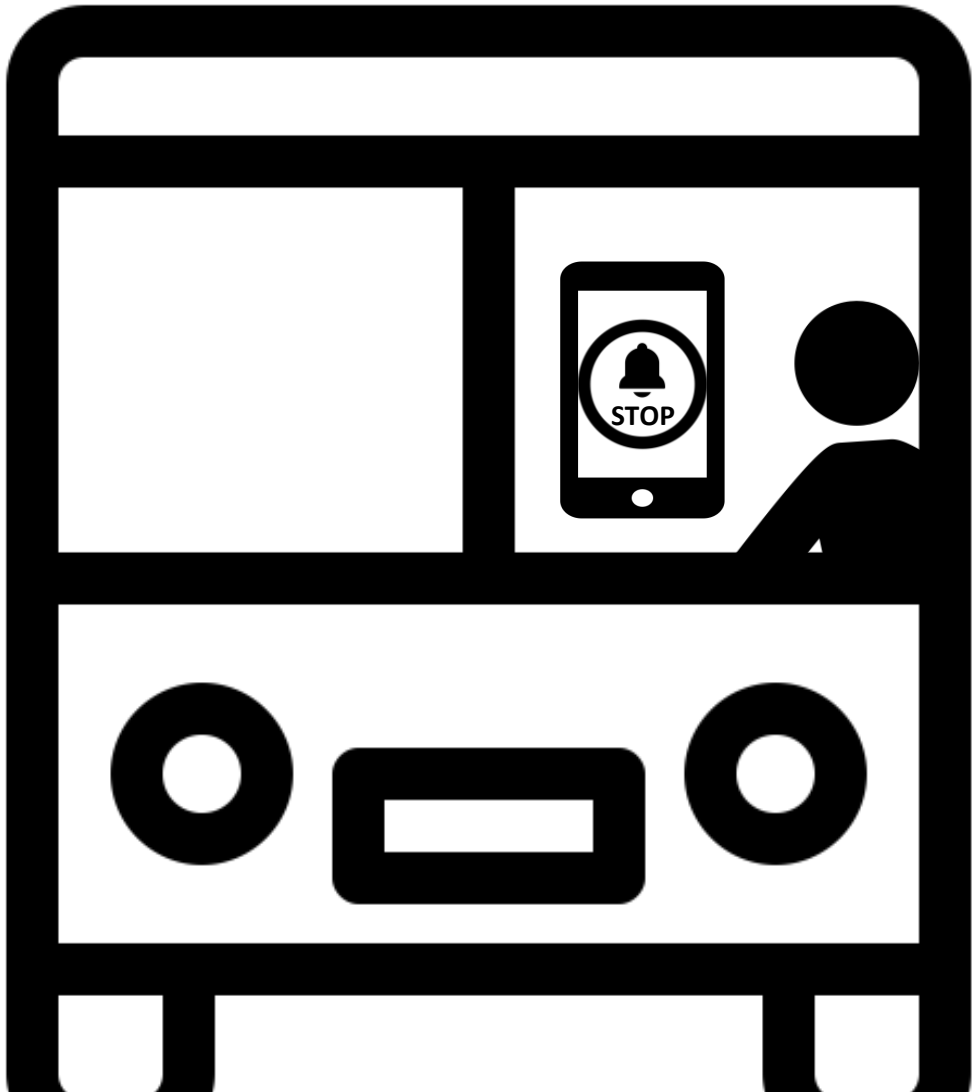
03 사용자 앱에 서버로부터 받은 정류장 위치의 도착 **버스를 좌우로 슬라이드하면 음성으로 안내하고 탑승할 버스를 선택**

04 탑승할 버스를 선택하여 **탑승여부를 버스기사님의 스마트폰 앱으로 전송**한다.

05 버스기사님의 스마트폰 앱을 통해 해당 정류장에 탑승을 희망하는 **시각장애인의 존재를 인식**시킨다.

06 버스가 정류장에 도착하면 시각장애인이 탑승 희망한 **버스 외부에 설치된 스피커를 통해 버스 도착을 알린다.**

3.1 시스템 수행 시나리오



01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**

02 가장 가까운 정류장으로 식별된 **정류장의 위치 정보를 사용자 앱에 전송**

03 사용자 앱에 서버로부터 받은 정류장 위치의 도착 **버스를 좌우로 슬라이드하면 음성으로 안내하고 탑승할 버스를 선택**

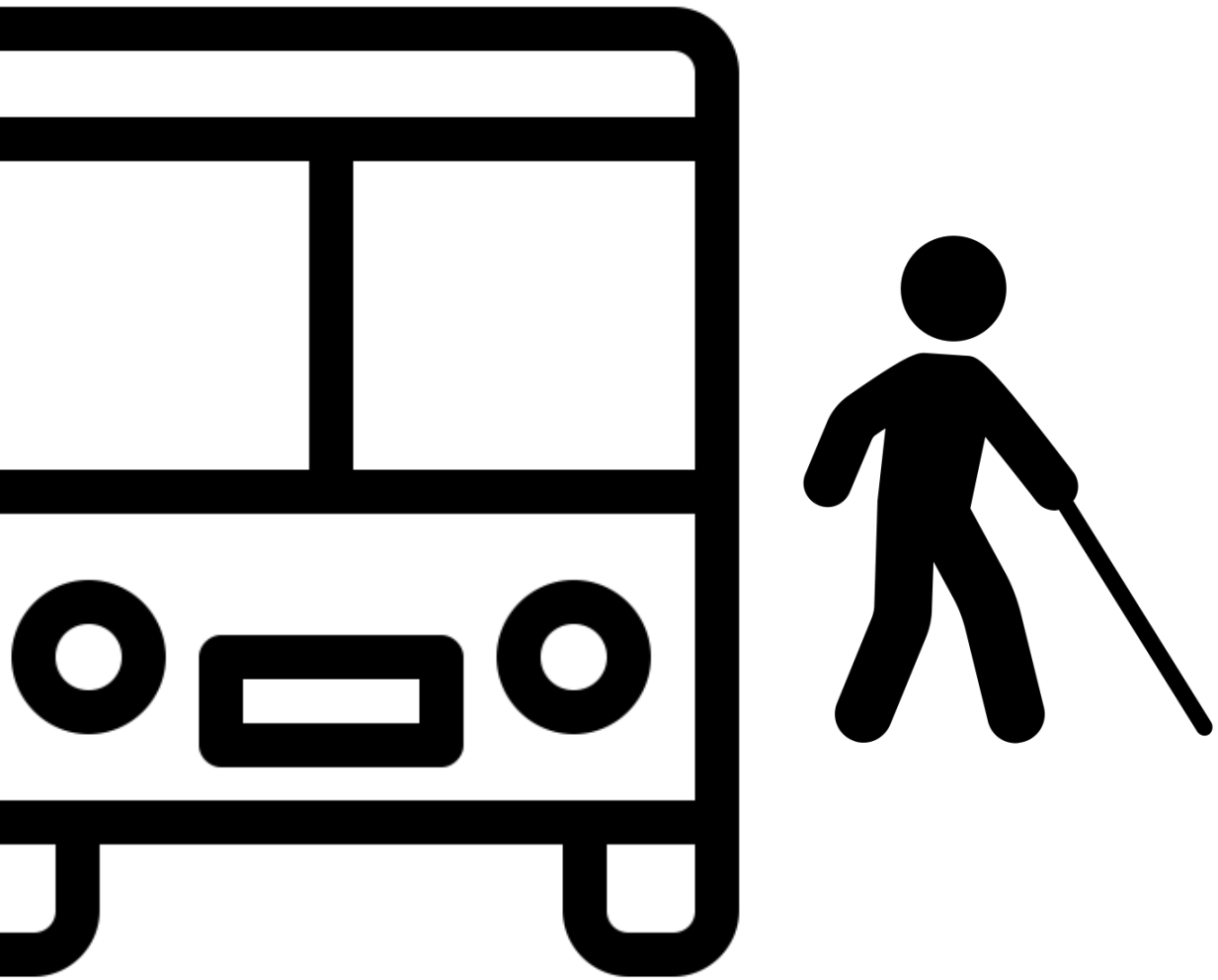
04 탑승할 버스를 선택하여 **탑승여부를 버스기사님의 스마트폰 앱으로 전송**한다.

05 버스기사님의 스마트폰 앱을 통해 해당 정류장에 탑승을 희망하는 **시각장애인의 존재를 인식**시킨다.

06 버스가 정류장에 도착하면 시각장애인이 탑승 희망한 **버스 외부에 설치된 스피커를 통해 버스 도착을 알린다.**

07 사용자가 버스에 탑승하고 하차할 때 기존 버스에 부착된 버스 **하차벨과 연동된 비콘을 이용하여 사용자 APP으로 하차벨 동작**

3.1 시스템 수행 시나리오



01 사용자가 버스 정류장에 접근하면 정류장 GPS와 사용자의 스마트폰 GPS 정보를 비교해 **가장 가까운 정류장을 식별**

02 가장 가까운 정류장으로 식별된 **정류장의 위치 정보를 사용자 앱에 전송**

03 사용자 앱에 서버로부터 받은 정류장 위치의 도착 **버스를 좌우로 슬라이드하면 음성으로 안내하고 탑승할 버스를 선택**

04 탑승할 버스를 선택하여 **탑승여부를 버스기사님의 스마트폰 앱으로 전송**한다.

05 버스기사님의 스마트폰 앱을 통해 해당 정류장에 탑승을 희망하는 **시각장애인의 존재를 인식**시킨다.

06 버스가 정류장에 도착하면 시각장애인이 탑승 희망한 **버스 외부에 설치된 스피커를 통해 버스 도착을 알린다.**

07 사용자가 버스에 탑승하고 하차할 때 기존 버스에 부착된 버스 **하차벨과 연동된 비콘을 이용하여 사용자 APP으로 하차벨 동작**

08 **사용자가 버스에서 내린다.**

4 시스템 구성도

4.1 하드웨어 구성도
4.2 소프트웨어 구성도

4.1 하드웨어 구성도

라즈베리파이 3B+

버스 제어 부분
마그네틱 도어 센서
하차벨
스피커

- 마그네틱 도어 센서를 이용해 정류장 도착 여부 확인
- 문이 열리면 스피커 작동

통신 부분
비콘
WIFI

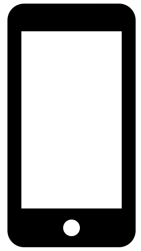
- 비콘 센서를 이용해 사용자의 하차 여부 통신
- 버스 WIFI를 이용해 서버 통신

무선 통신

Amazon
EC2

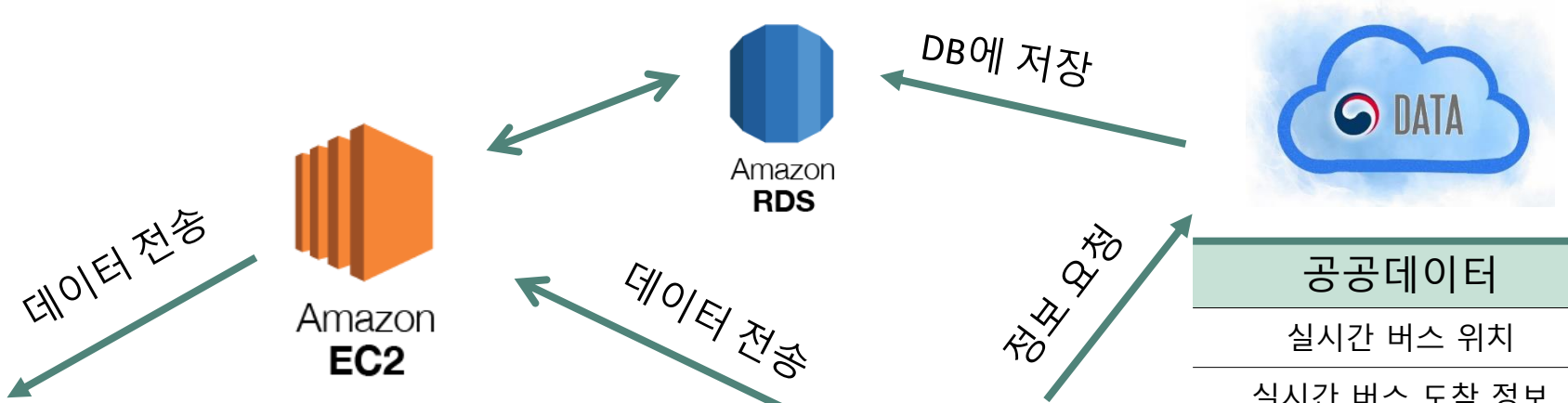
정류장에 시각장애인 대기 여부 송신

무선 통신



블루투스 통신 이용해 하차 여부 전송

4.2 소프트웨어 구성도



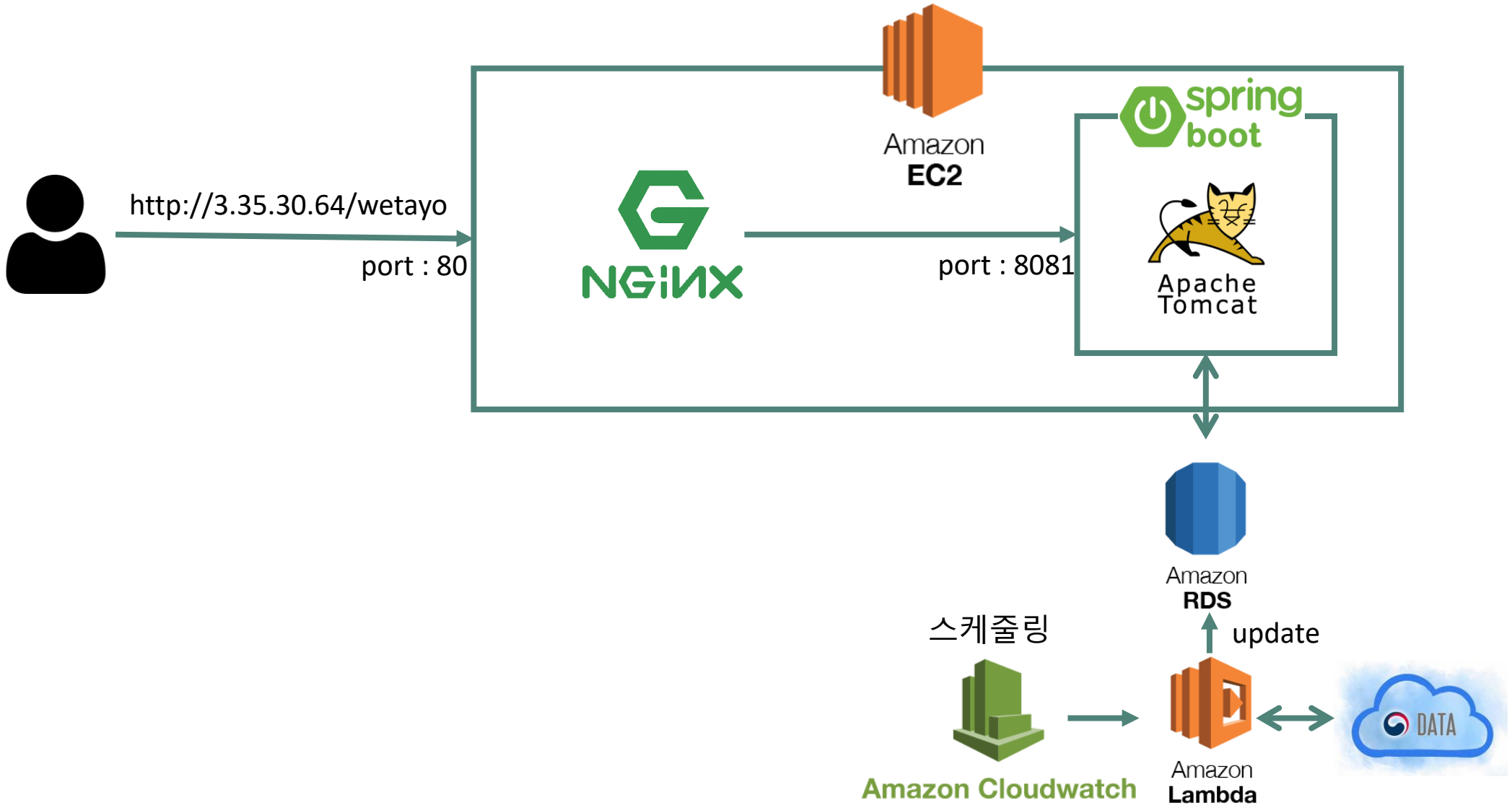
공공데이터
실시간 버스 위치
실시간 버스 도착 정보
시흥시 버스 정류장 정보

Client (버스 기사)
도착 여부 전송
위치 정보 데이터
시각 장애인 탑승 여부 확인

Server
Ubuntu Linux
MySQL - 버스 정보 - 버스 정류장 위치 정보 - 사용자 정보
로그인 / 로그아웃 기능
사용자 근접 정류장 확인 알고리즘
시각 장애인 탑승 여부 전송

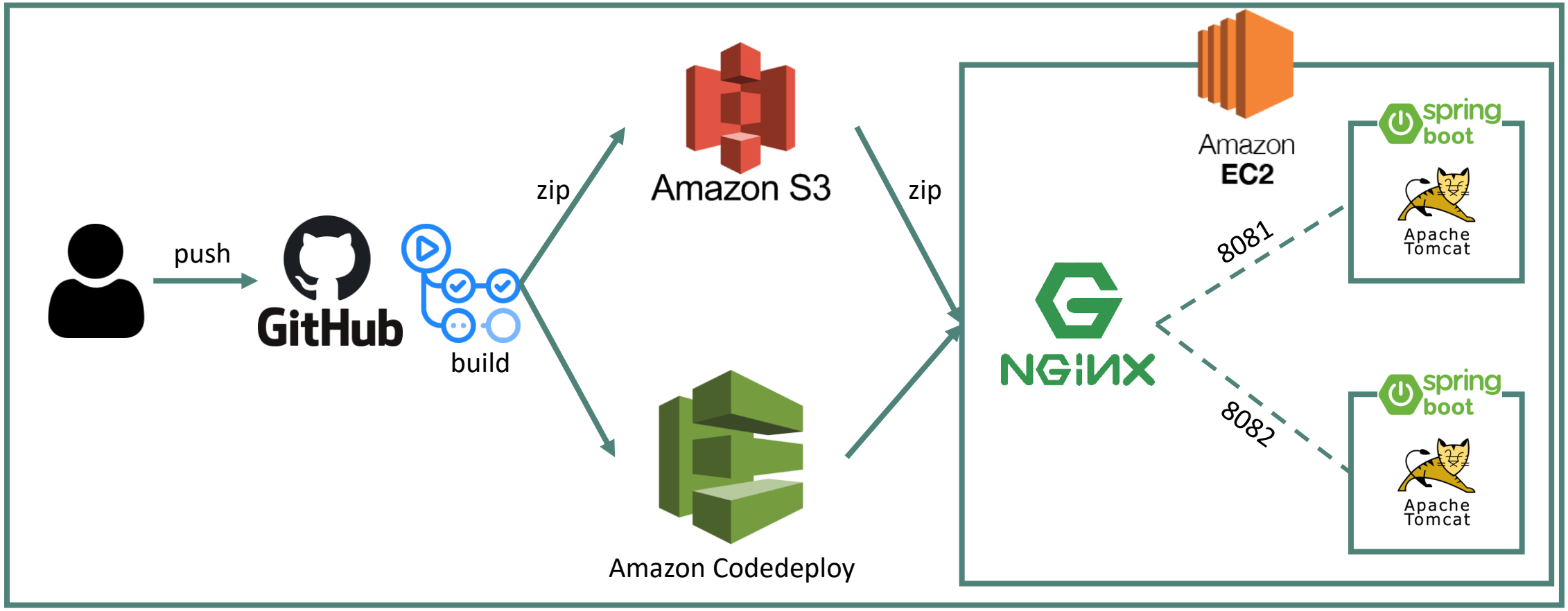
Client (사용자)
블루투스 통신
위치 정보 데이터
음성 출력(TTS)
시각장애인을 위한 UI/UX

4.2 소프트웨어 구성도 (서버)



4.2 소프트웨어 구성도 (서버)

무중단 자동 배포 설정

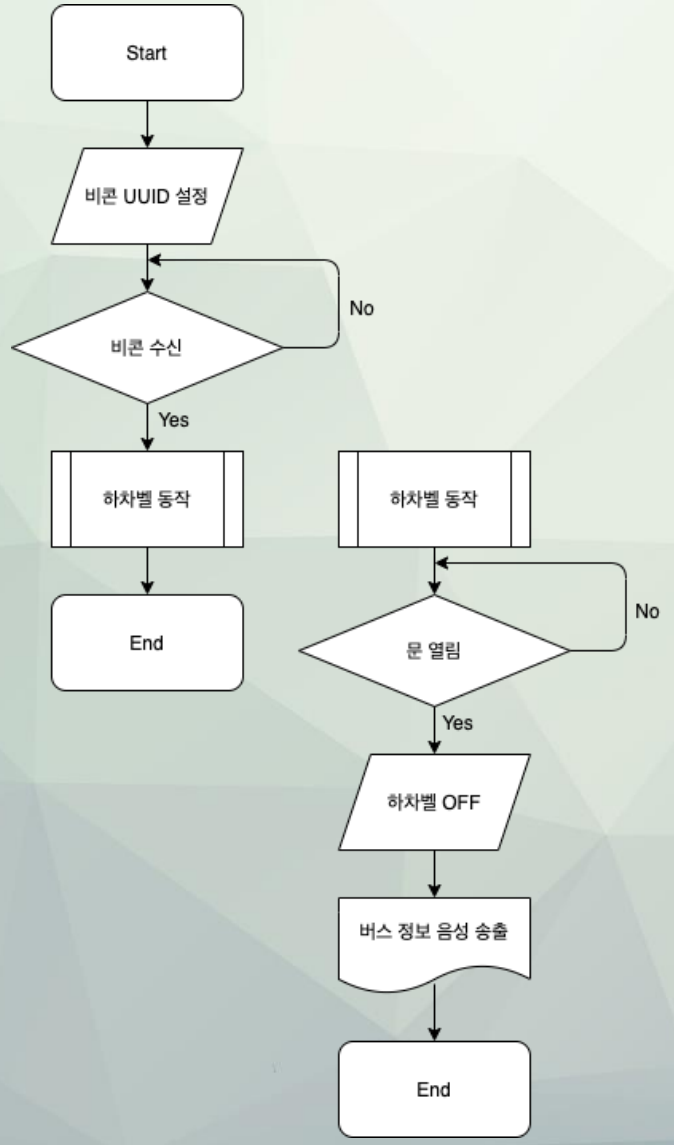


5 시스템 모듈 상세설계

5.1 하드웨어 모듈 상세설계
5.2 소프트웨어 모듈 상세설계 (서버)
5.3 소프트웨어 모듈 상세설계 (앱)

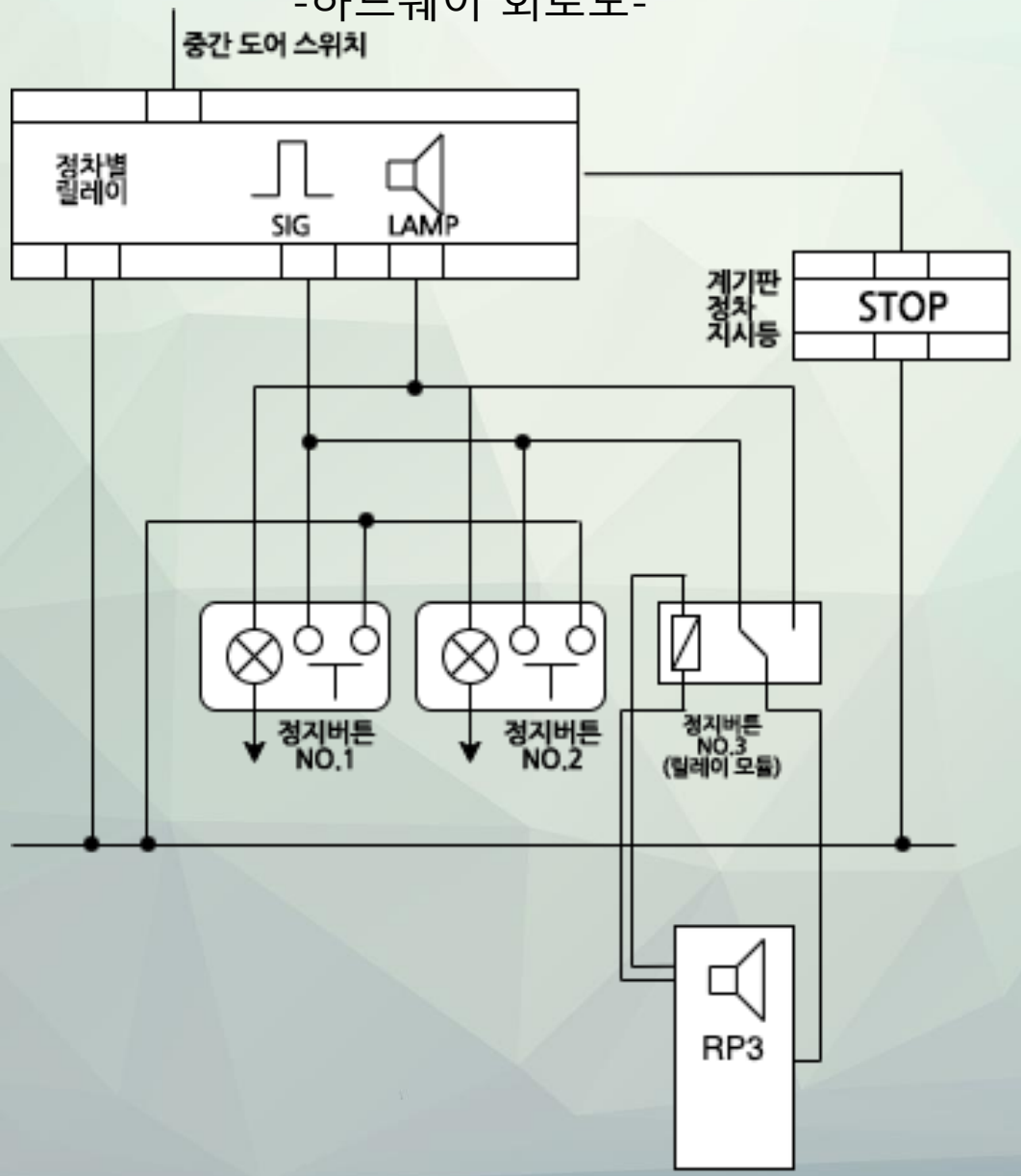
5.1 하드웨어 모듈 상세설계

-하드웨어 동작 흐름도-



5.1 하드웨어 모듈 상세설계

-하드웨어 회로도-



5.1 하드웨어 모듈 상세설계

iBeacon prerfix (9 bytes)	Proximity UUID (16 bytes)	Major (2 bytes)	Minor (2 bytes)	TX power (2 bytes)

Proximity UUID (16 bytes)

- 범용 고유 아이디 UUID는 WeTayo의 고유 식별자
- 모든 버스의 iBeacon UUID는 모두 동일하게 부여

Major (2 bytes)

- Major 번호는 지역을 구분하는데 사용
- ex) 서울시 : 00, 시흥시 : 11

Minor (2 bytes)

- Minor 번호는 해당 지역의 버스를 구분하는데 사용
- ex) 버스1 : 00, 버스2 : 01, … 버스n : FF

5.2 소프트웨어 모듈 상세설계 (서버)

-Github action & CodeDeploy가 실행하는 모듈-

Deploy.yml	VM에 어떤 OS를 설치할지와 build 명령, S3/Codedeploy에 upload 명령을 정의한 파일	
Appspec.yml	Run_new_was.sh	현재 실행되고 있는 port번호 조회후 다른 port번호로 새로운 서버(was) 실행
	Health_check.sh	새로 실행한 WAS가 실행될때 까지 check하는 모듈
	Switch.sh	Nginx에 새로 띄운 서버의 포트로 스위칭하는 모듈

5.2 소프트웨어 모듈 상세설계 (서버)

-DB TABLE-

RIDE	
ID	AUTO_INCREASEMENT
STATION_ID	정류장 ID
ROUTE_ID	노선 ID

ROUTE_STATION	
STATION_ID	정류장 ID
ROUTE_ID	노선 ID
UP_DOWN	버스 방향 (정, 역)
STATION_ORDER	정류장 순서
STATION_NAME	정류장 이름
ROUTE_NUMBER	노선 번호

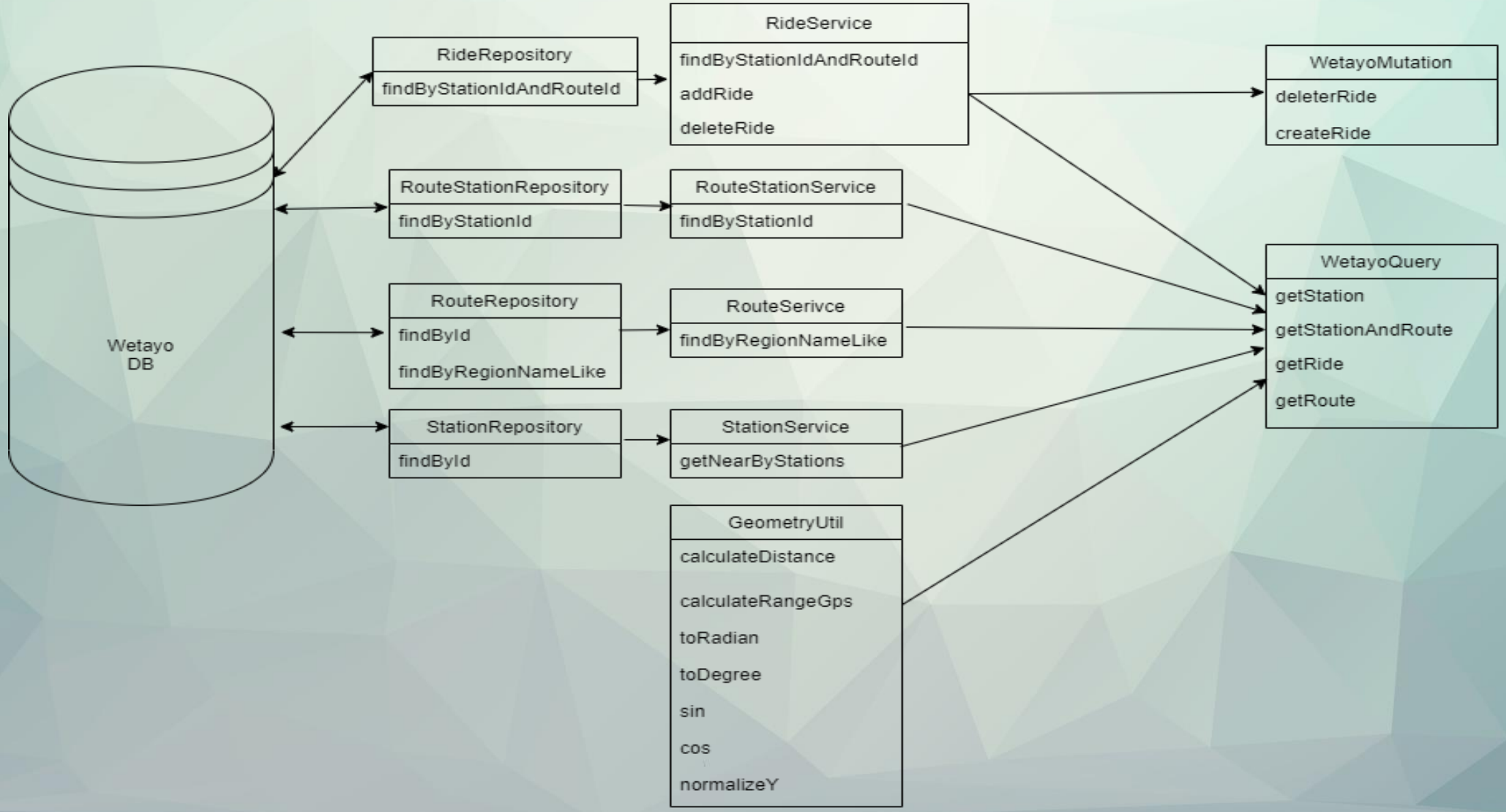
STATION	
STATION_ID	정류장 ID
STATION_NAME	정류장 이름
CENTER_ID	중앙차로 ID
CENTER_YN	중앙역 여부
REGION_NAME	지역 이름
MOBILE_NUMBER	정류장 고유 MOBILE NUMBER
GPS	GPS (위도, 경도) 정보
DISTRICT_CODE	관할 지역

5.2 소프트웨어 모듈 상세설계 (서버)

-DB TABLE-

ROUTE			
ROUTE_ID	노선 ID	UP_FIRST_TIME	기점 첫차 시간
ROUTE_NUMBER	노선 번호	UP_LAST_TIME	기점 막차 시간
ROUTE_TP	노선 유형	DOWN_FIRST_TIME	종점 첫차 시간
START_STATION_ID	기점 정류소 ID	DOWN_LAST_TIME	종점 막차 시간
START_STATION_NAME	기점 정류소 이름	PEEK_ALLOC	최소 배차 시간
START_STATION_NUMBER	기점 정류소 번호	NPEEK_ALLOC	최대 배차 시간
END_STATION_ID	종점 정류소 ID	COMPANY_ID	운수업체 ID
END_STATION_NAME	종점 정류소 이름	COMPANY_NAME	운수업체 이름
END_STATION_NUMBER	종점 정류소 번호	TEL_NUMBER	운수업체 전화번호
REGION_NAME	지역 이름	DISTRICT_CODE	관할 지역

5.2 소프트웨어 모듈 상세설계 (서버)



5.2 소프트웨어 모듈 상세설계 (서버)

-서버 시스템 모듈 설명-

클래스 명	메서드 명	기능
Wetayo Query	getStation(Double x, Double y, Double distance)	요청 GPS 반경 distance만큼의 정류장 조회
	getStationAndRoute(Double x, Double y, Double distance)	요정 GPS반경 distance만큼의 정류장 정보와 그에 속한 버스 정보 조회
	getRide(Integer stationId,Integer routeId)	특정 노선의 특정 정류장의 사용자 탑승 희망 여부 조회
	getRoutes(String regionName)	특정 지역에 해당하는 노선들 모두 조회
Wetayo Mutation	createRide(Integer stationId, Integer routeId)	특정 노선의 특정 정류장에 탑승 희망 생성
	deleteRide(Integer stationId, Integer routeId)	특정 노선의 특정 정류장에 탑승 희망 생성

5.2 소프트웨어 모듈 상세설계 (서버)

-서버 시스템 모듈 설명-

클래스 명	메서드 명	기능
Ride Service	findByStationIdAndRouteId(Integer stationId,Integer routeId)	요청한 복합키인 StationId와 RouteId가 존재하는지 repository를 통해 확인
	addRide(Integer stationId, Integer routeId)	사용자가 탑승을 희망하면 테이블에 컬럼 insert
	deleteRide(Integer stationId, Integer routeId)	사용자가 탑승을 했거나, 버스가 지나가면 테이블에 컬럼 delete
Station Service	getNearByStations(Double gpsX, Double gpsY, Double distance)	요청한 GPS, distance를 repository를 통해 컬럼 조회
Route Service	findByRegionNameLike(String regionName)	지역이름을 통해 노선정보들 조회
Route Station Service	findByStationId(Integer id)	노선에 속한 정류장 테이블에서 stationId를 통해 컬럼 조회

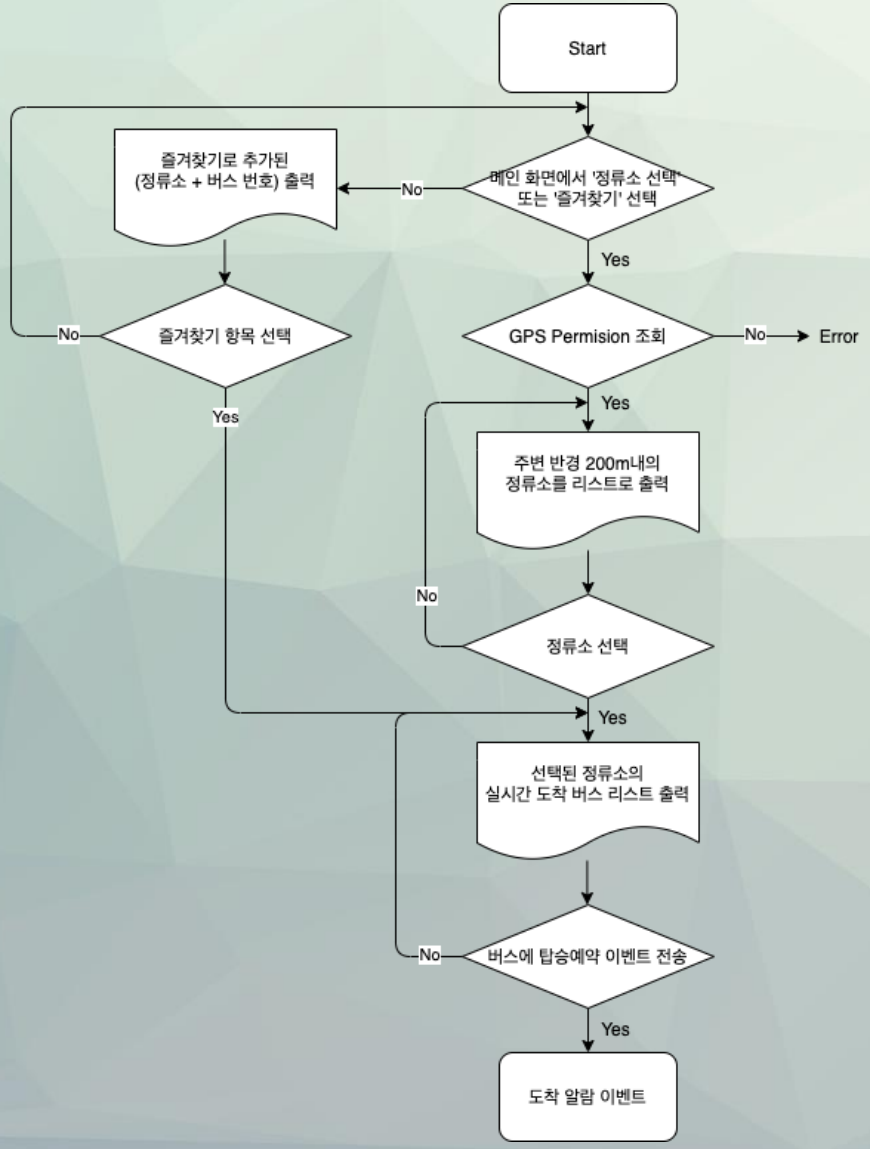
5.2 소프트웨어 모듈 상세설계 (서버)

-서버 시스템 모듈 설명-

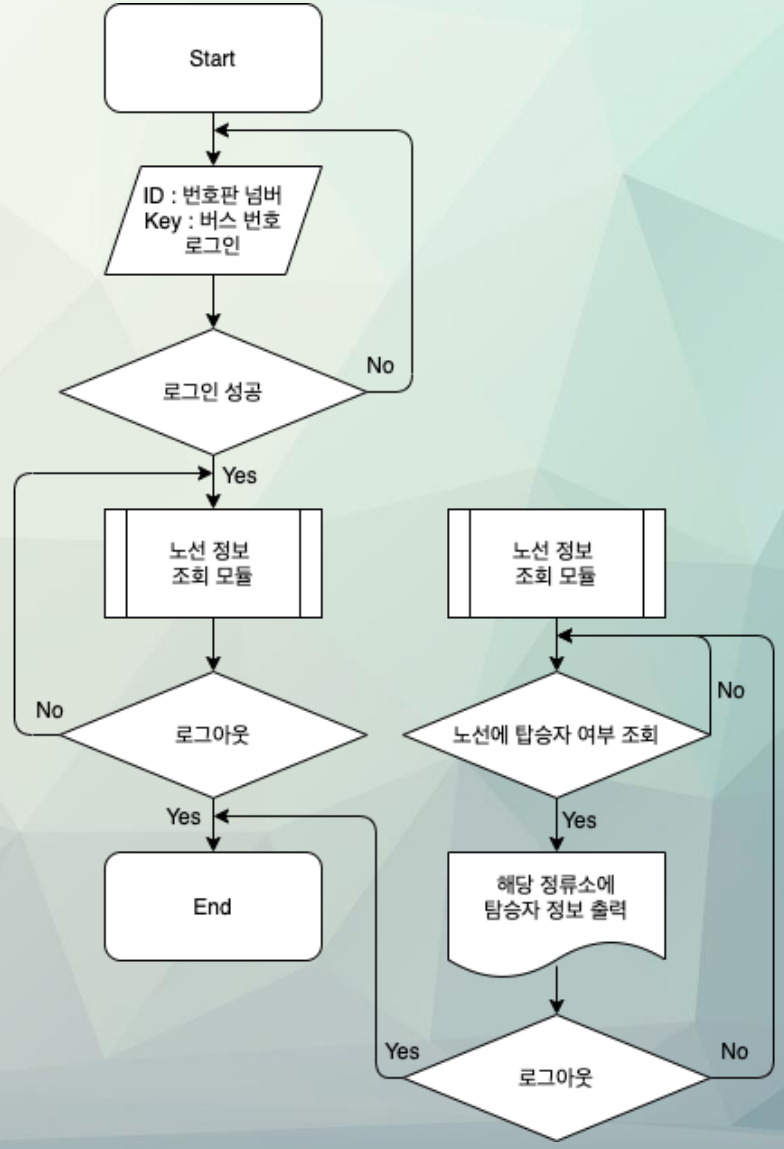
클래스 명	메서드 명	기능
Geometry Util	calculateRangeGps(Double baseX, Double baseY, Double distance,Double bearing)	요청 gps의 distance만큼 bearing(목적지 방향)으로 떨어진 gps좌표를 계산
	calculateDistance(*double* lat1,*double* lon1, *double* lat2, *double* lon2)	두 gps좌표간의 거리를 계산
	toRadian(Double coordinate)	각도(degree)를 radian기반으로 바꿔주는 함수
	toDegree(Double coordinate)	Radian값을 각도(degree)로 바꿔주는 함수
	sin(Double coordinate)	Radian을 기반으로 sin값 계산
	cos(Double coordinate)	Radian을 기반으로 cos값 계산
	normalizeY(Double gpsY)	계산시 원이기에 -360~360 범위의 값을 경도범위에 맞게 ~180~180에 맞게 정규화

5.3 소프트웨어 모듈 상세설계 (앱)

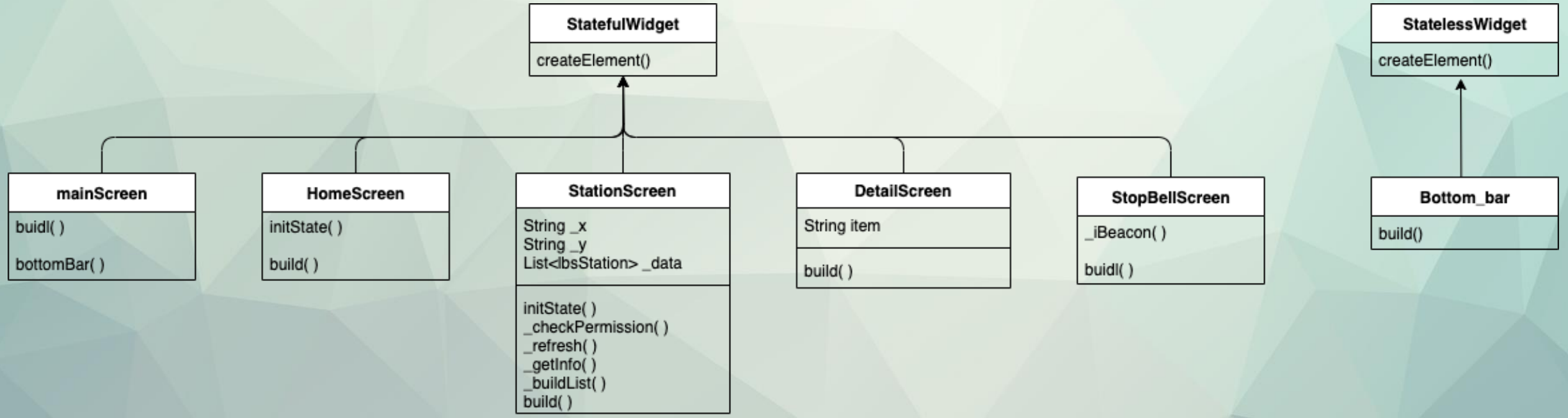
-사용자 앱 동작 흐름도-



-버스기사 앱 동작 흐름도-



5.2 소프트웨어 모듈 상세설계 (앱)

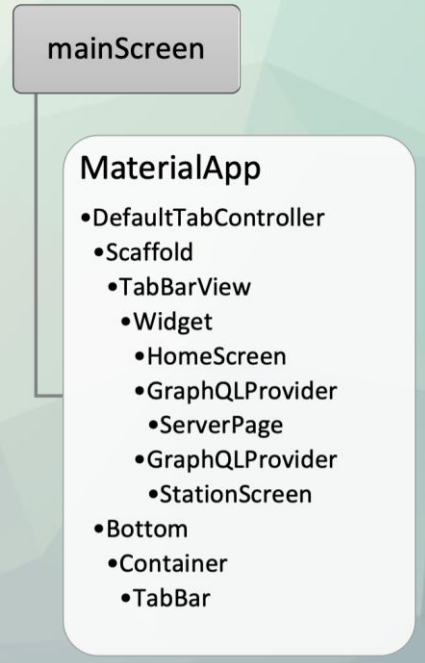


5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명	메서드 명	기능
mainScreen	Widget build(BuildContext context)	사용자 앱의 3가지 탭을 구성하는 뼈대를 역할하는 위젯

메서드	
형식	Widget build(BuildContext context)
리턴 값	Widget
설명	사용자 앱의 3가지 탭을 구성하는 뼈대를 역할하는 위젯

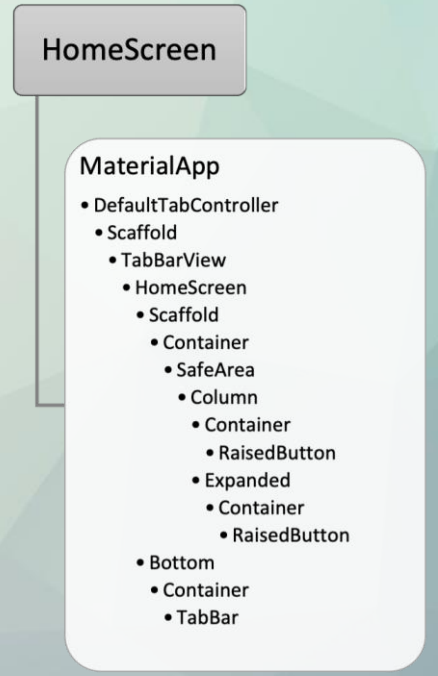


5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명	메서드 명	기능
HomeScreen	Widget build(BuildContext context)	사용자 앱 실행 시 보이는 가장 첫 번째 화면을 구성하는 함수

메서드	
형식	Widget build(BuildContext context)
리턴 값	Widget
설명	사용자 앱 실행시 보이는 가장 첫 번째 화면을 구성하는 함수



5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명	메서드 명	기능
StationScreen	_checkPermission() async	GPS 권한을 디바이스에 부여하기 위한 비동기 함수
	_refresh() async	페이지가 새로고침 될 때마다 현재 나의 GPS 값(위도 경도)를 조회하기 위한 비동기 함수
	Widget build(BuildContext context)	' 정류소 선택’ 탭의 위젯을 빌드하는 함수
	Widget _buildList(BuildContext context, QueryResult rresult)	요청한 쿼리 결과에 따른 리스트뷰 위젯을 빌드하는 함수

5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명x	메서드 명	기능
StationScreen	_checkPermission() async	GPS 권한을 디바이스에 부여하기 위한 비동기 함수

메서드	
형식	_checkPermission() async
리턴 값	X
설명	GPS 권한을 디바이스에 부여하기 위한 비동기 함수
예시	X

5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명	메서드 명	기능
StationScreen	_refresh() async	페이지가 새로고침 될 때마다 현재 나의 GPS 값(위도 경도)를 조회하기 위한 비동기 함수

메서드	
형식	_refresh() async
리턴 값	X
설명	페이지가 새로고침 될 때마다 현재 나의 GPS 값(위도 경도)를 조회하기 위한 비동기 함수
예시	X

5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명	메서드 명	기능
StationScreen	Widget build(BuildContext context)	' 정류소 선택' 탭의 위젯을 빌드하는 함수
	Widget _buildList(BuildContext context, QueryResult result)	요청한 쿼리 결과에 따른 리스트뷰 위젯을 빌드하는 함수

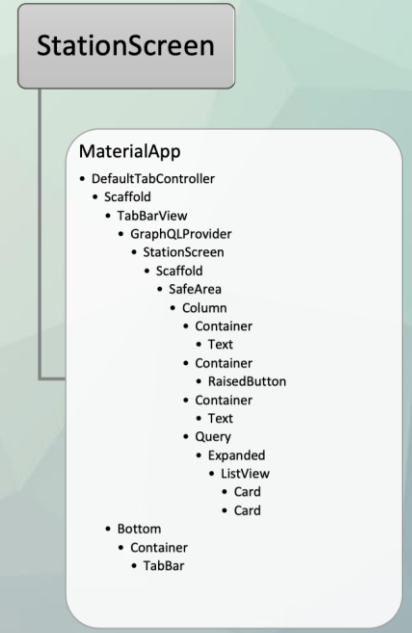
메서드	
형식	Widget build(BuildContext context)
리턴 값	Widget
설명	' 정류소 선택' 탭의 위젯을 빌드하는 함수

5.2 소프트웨어 모듈 상세설계 (서버)

-앱 시스템 모듈 설명-

클래스 명	메서드 명	기능
StationScreen	Widget build(BuildContext context)	' 정류소 선택’ 탭의 위젯을 빌드하는 함수
	Widget _buildList(BuildContext context, QueryResult result)	요청한 쿼리 결과에 따른 리스트뷰 위젯을 빌드하는 함수

메서드	
형식	Widget _buildList(BuildContext context, QueryResult result)
리턴 값	Widget
설명	요청한 쿼리 결과에 따른 리스트뷰 위젯을 빌드하는 함수



6 시스템 환경 및 개발 방법

6.1 하드웨어 개발 환경
6.2 소프트웨어 개발 환경
6.3 시스템 개발 방법

6.1 하드웨어 개발 환경

라즈베리파이 3B+

- CPU : 1.4GHz Cortex-A53
- 1GB LPDDR2 SDRAM
- Bluetooth 4.2
- 무선 LAN
- 5V/2.5A DC 전원 입력

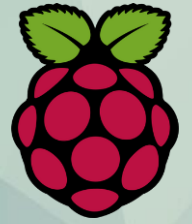
릴레이 모듈

- 입력 전압 : 5v
- 1채널

외부 차량용 스피커

- 출력 전원 : 3w
- 크기 : 80*57*45mm
- 임피던스 : 80hm

6.2 소프트웨어 개발 환경



Rasbian

- 개발사 : 라즈베리파이 재단
- 사용 언어 : C/C++
- 버전 : 1.4

VSCode

- 개발사 : Microsoft
- 사용 언어 : Flutter - Dart
- Flutter 버전 : 1.17

Intellij

- 개발사 : Anaconda
- 사용 언어 : Java
- 버전 : 20.2

AWS EC2

- 개발사 : Amazo
T2.micro
- 최대 30GB
- 램 1GB
- Ubuntu Linux 20.04

MySQL

- 개발사 : Oracle
- 사용 언어 : SQL
- 버전 : 8

6.2 소프트웨어 개발 환경

네이버 Clova TTS

- 개발사 : 네이버
- 입력 : 텍스트
- 요금 : 1000자당 4원

공공 API

- 개발사 : 경기도
- 출력 : XML, JSON

Github

- 개발사 : Microsoft
- 사용 언어 : Git
- 목적 : 버전관리, Issu 관리

Zenhub

Kanban 이용
- 목적 : 프로세스 관리

6.3 시스템 개발 방법

GitHub 주소

https://github.com/gowoonsori/weTaYo

강석원

https://github.com/kssssw

박형근

https://github.com/kssssw

이지수

https://github.com/leejisu9068

홍의성

https://github.com/gowoonsori

6.3 시스템 개발 방법

하드웨어 (라즈베리파이 3B+)
라즈베리파의 BLE기능을 활성화
스피커는 외부전력을 연결해 버스 외부에서 버스 정보를 출력
마그네틱 도어센서 사용해 문 열림을 감지, 감지에 따른 스피커 동작
TeamViewer을 사용해 원격으로 접속하여 개발할 수 있는 환경 구축
센서 동작 응용프로그램은 python을 사용해 작성

6.3 시스템 개발 방법

서버 (AWS ec2)
AWS ec2를 사용하여 서버 구축
AWS의 ec2의 개발 OS는 우분투/리눅스
Java Spring boot를 사용해 서버 개발
DB는 mysql8을 사용
공공데이터의 정적인 데이터(정류소 위치, 버스 노선 정보)는 DB에 저장
공공데이터의 동적인 데이터(실시간 버스 위치)는 실시간 공공 API를 이용
좌표간 거리 계산 알고리즘 (하버 사인 공식, 구면 코사인 법칙)을 이용

6.3 시스템 개발 방법

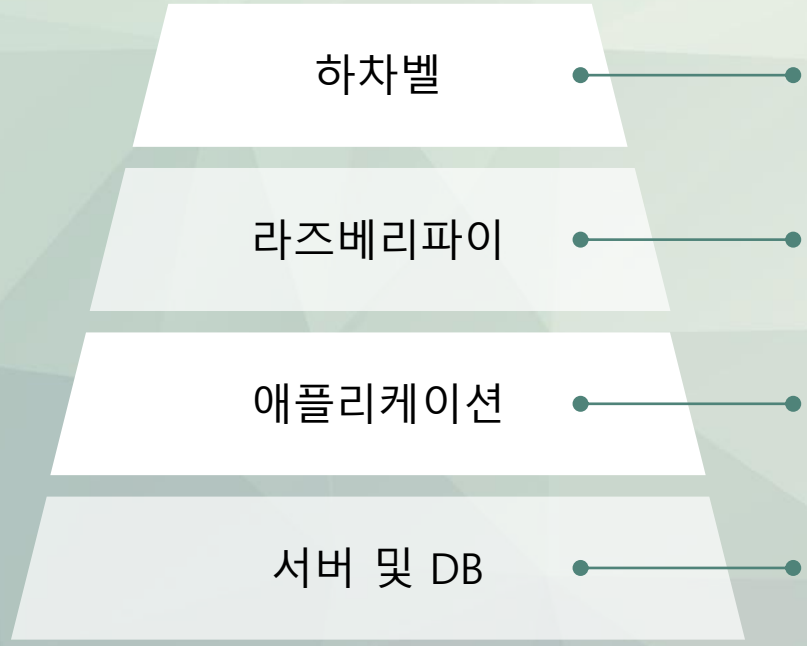
애플리케이션 (Flutter)
Flutter을 이용하여 IOS와 Android 하이브리드 애플리케이션 개발
네이버 Clova TTS 음성 합성 기술 API를 사용한 음성 송출
사용자 앱은 시각장애인의 편의가 적용된 UI/UX를 반영
무선 하차벨을 동작시키는 과정에서 Beacon을 사용
Flutter은 Dart 언어를 사용하고 개발 환경은 Visual Studio Code를 사용

DevOps
Zenhub를 이용하여 효율적으로 프로세스를 관리
애자일 방식을 사용해 Kanban 이용
Github로 Version과 Issue를 관리

7 데모 환경 설계

7.1 데모 환경 구성
7.2 데모 순서

7.1 데모 환경 구성



하차벨의 전원 24V을 인가하고, 릴레이 모듈이 정상으로 동작하는지 점검

라즈베리파이의 무선 인터넷 연결 상태를 점검하고, 하차벨과 스피커가 정상 상태인지 점검

사용자용 애플리케이션이 설치된 디바이스와 버스기사용 애플리케이션이 설치된 디바이스를 각 1개씩 준비

AWS EC2 서버는 상시 동작중이며, 동작 상태와 GraphQL 프로세스 동작을 점검

7.2 데모 순서

1

전체 시나리오 동작 영상

- 전체 시나리오대로 동작하는 영상을 촬영
- 버스 또는 자차에 부착해 촬영

WE TAYO

2

애플리케이션 동작 직접 시연

- 사용자 & 버스기사 앱이 설치된 디바이스를 각 1대씩 준비해 동작 과정을 직접 시연

3

무선 하차벨 동작 직접 시연

- 사용자 앱에서 무선으로 비콘 통신으로 하차벨 동작을 직접 시연

4

동작 결과 웹으로 확인

- 동작으로 인한 모든 로그를 시각화 하여 웹을 통해 확인

8

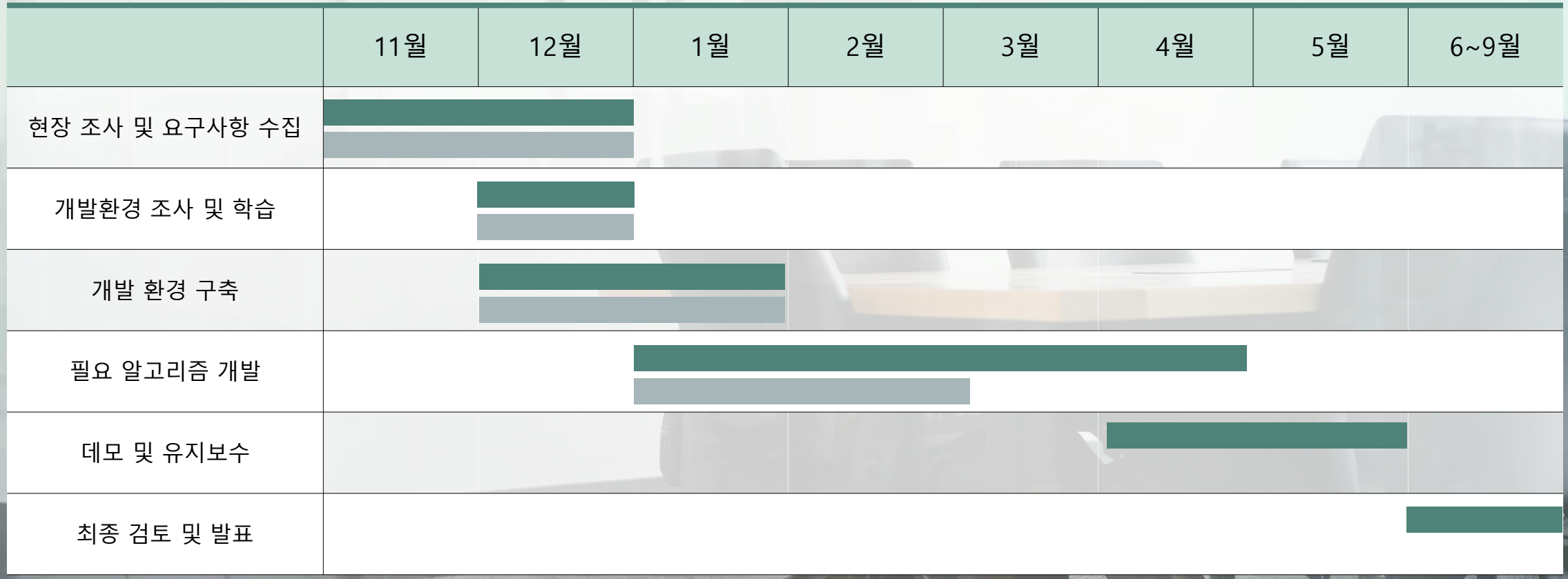
업무 분담 및 수행 일정

8.1 업무 분담
8.2 수행 일정
8.3 진행 상황

8.1 업무 분담

	강석원	박형근	이지수	홍의성
자료수집	- Flutter 조사 - 네이버 TTS Clova 조사 - GPS 오차 범위 조사 - 유사 사례 조사	- 유사 사례 조사 - 개발 방법 조사 및 목표 설정 - 하드웨어 및 모듈 조사 - Flutter 비콘 라이브러리 조사	- Flutter 조사 - 네이버 TTS Clova 조사 - Dart 언어 조사 - 유사 사례 조사	- AWS 자료 조사 - DB 조사 - 프로세스 관리 툴 조사 - 공공 데이터 조사 - GPS 오차 범위 조사
설계	IOS 앱과 서버 통신 설계	라즈베리파이 비콘 설정과 하차벨 회로 설계	Android 앱과 서버 통신 설계	AWS 서버 설계
구현	IOS 앱 구현	라즈베리파이 센서 연동 및 동작 구현	Android 앱 구현	AWS 서버 구축 및 통신 구현
테스트	Github로 수시로 Version 관리와 Issue관리를 함			

8.2 수행 일정


11월
12월
1월
2월
3월
4월
5월
6~9월
현장 조사 및 요구사항 수집
개발환경 조사 및 학습
개발 환경 구축
필요 알고리즘 개발
데모 및 유지보수
최종 검토 및 발표

8.3 진행 상황

	하드웨어	45%	소프트웨어(앱)	65%	서버	70%
전체 진행 상황 60%	버스 하차벨 회로 구현	90	사용자 앱 UI	90	무중단 자동 배포	100
	하차벨 회로 라즈베리파이 연결	10	(사용자 앱)GraphQL 통신 설계 및 구현	80	DB 업데이트 스케줄링	100
	비콘 송신 설계 및 구현	20	(사용자 앱)공공 API 통신 설계 및 구현	100	DB 설계	100
	스피커 설계 및 구현	50	(사용자 앱)비콘 통신 설계 및 구현	20	API 구현	100
	Clova TTS API 적용 및 구현	50	(사용자 앱)버스 탑승 예약 설계 및 구현	40	API 문서 작성	100
			버스기사 앱 UI	80	에러 처리	50
			(버스기사 앱)GraphQL 통신 설계 및 구현	40	SSL/TLS 기능	10
			대체 텍스트 구현	70	권한 인증	10

8.3 진행 상황

전체 진행 상황 60%	하드웨어	45%	소프트웨어(앱)	65%	서버	70%

하드웨어의 진행상황이 늦은 이유
- 데모환경을 구성하기 위한 실제 버스 하차벨 회로를 구하는데 시간이 걸림
- 라즈베리파이의 BLE를 활성화 하는 단계에서 보유중인 라즈베리파이의 블루투스 모듈 문제 발생

➔ 실제 대우버스에서 사용중인 하차벨 릴레이 회로를 구함
➔ 새로운 라즈베리파이를 구입함 라즈베리파이의 활성화가 계속 난제이면 비콘 모듈을 구입할 계획

소프트웨어(앱)의 진행 문제점
- 시각장애인 애플리케이션 사용지침을 참고하여 새롭게 디자인을 하는데 많은 시간이 소요됨
- 플러터 비콘 라이브러리의 코드 샘플이 많이 없어 찾는데 시간이 걸림

➔ 사용지침을 꼼꼼하게 정독하며 새로운 UI를 구성함
➔ 하드웨어의 구현이 완료하면 테스트를 반복하며 해결할 계획

서버의 진행상황
- 탑승 예약에 있어 서버DB에 아무나 접근할 수 없게 권한 인증을 처리해야함
- 사용자들의 사용성을 높이고자 더 좋은 방법을 구상 중

➔ 기사는 버스마다 고유 id를 이용하여 권한 인증 처리하는 방법 채택
➔ 사용자는 로그인 기능은 없애기로 하고 다른 방안을 생각 중

9 참고 문헌

9.1 참고 문헌

9.1 참고 문헌

Flutter 공식 문서 – https://flutter.dev/docs

CLOVA Speech Synthesis(CSS)

- https://apidocs.ncloud.com/ko/ai-naver/clova_speech_synthesis/

좌표간 거리계산

- https://pyxispub.uzuki.live/?p=1006

시각장애인 애플리케이션 사용 지침

- https://www.nld.go.kr/ableFront/new_standard_guide/accessibility.jsp

Amazon Elastic Compute Cloud – Linux 인스턴스용 사용 설명서

- AWS

GB302 버스정보 Open API 서비스 명세서 – 버스 노선 조회 서비스(SOAP)

- 경기도 버스정보 시스템

GB307 버스정보 Open API 서비스 명세서 – 버스 위치 정보 조회 서비스(SOAP)

- 경기도 버스정보 시스템

GB308 버스정보 Open API 서비스 명세서 – 버스 도착 정보 조회(SOAP)

- 경기도 버스정보 시스템

QnA

THANKS