

치매 환자 발화패턴 학습 AI를 이용한 위험 상황 감지 및 실종 예방 도우미

Early detector of missing situation using AI
to learn speech patterns for dementia patients

수제지능팀

2201658 황혜진
2104651 오지현
2201639 전희주
2201624 옥지원

팀원 소개

[캡스톤디자인 졸업작품 B01]

헤아Ring : 치매 환자 발화패턴 학습 AI를 이용한 위험 상황 감지 및 실종 예방 도우미



황혜진 (2201658)



- 팀장 + AI 전담
- 위험상황 판단 AI 개발 및 딥러닝 서버 구축
- 로그인, 회원가입 App 개발



전희주 (2201639)



- App 개발 전담
- 서버와의 통신을 위한 API 연동
- UIUX 설계



오지현 (2104651)



- 백엔드 및 서버 전담
- DB설계 및 관리 등 백엔드 전체 개발
- 웹서버 구축



옥지원 (2201624)



- 하드웨어 전담
- 라즈베리 파이4 전체 개발
- 클라이언트와 서버간 통신 기능 개발

목차

1 / 프로젝트 개요

- 01 기획 배경
- 02 프로젝트 목표
- 03 벤치마킹 및 차별성

2 / 시스템 분석 및 설계

- 01 서비스/사용자 시나리오
- 02 시스템 아키텍처
- 03 개발 환경
- 04 알고리즘 시나리오
- 05 App 구성도
- 06 DB 설계

3 / 프로젝트 구현

핵심

- 01 로그인 및 회원가입
- 02 메인화면
- 03 위험감지 화면
 - └ 위험상황 판단 AI
- 04 GPS 화면
- 05 이전기록 다시보기 화면
- 06 개발 요약

4 / 1차 심사 피드백 반영사항

- 01 피드백 내용 정리
- 02 현재 한계점
- 03 개선 사항
- 04 새로 개발된 사항
- 05 추후 연구과제

5 / 실제 시연

프로젝트 개요

- 기획 배경
- 프로젝트 목표
- 벤치마킹 및 차별성

01 기획 배경

(1) 치매 환자의 위험한 배회

✓ 치매환자 실종신고 매년 반복

치매 환자 관련 실종 신고 접수 건수	
2019년	1247건
2020년	1162건
2021년	1186건
2022년	1297건
2023년	1289건

✓ 범죄타겟화 : 증거 및 증언 수집의 어려움



“ 현재까지 조사 결과, 할머니가 넘어지는 장면 등은 포착되었지만 폭행을 의심할 만한 상황은 파악되지 않은 것으로 전해졌다. ”

(2) 활용도 낮은 실종문자

✓ 실종 경보 문자 활용의 문제점

■ 효과 있다는데, 활용도는 왜 낮을까?

실종경보문자의 효과성을 알면서도, 왜 많이 활용하지 않느냐는 질문에 경찰청은 다음과 같이 답했습니다.

1. 실종경보문자는 문자가 효과적인 상황일 경우에만 사용한다.
2. 실종자가 발견될 가능성이 높은 지역에만 발송한다.

일선 경찰관들의 반응도 비슷했습니다. 문자 발송의 필요성이 크지 않다는 반응이었습니다.



(3) 보호자의 돌봄 부담

✓ 보호자 돌봄 부담 증가

실제 치매환자 보호자들은 환자를 돌봄에 있어 경제적인 부담뿐만 아니라 **신체적, 정서적 측면에서의 부담**도 함께 느끼고 있다.

- 배회로 인해 얻는 가족의 스트레스 ↑

면서 사회와 단절이 되고, 치매노인 혼자 나갔다가 다치면 어 떡하나 하는 걱정으로 신경을 쓰는 등 신체적, 정신적으로 많은 스트레스를 경험하였다. **치매노인의 배회는 위험을 예측 할 수 없는 속성** 때문에 24시간 지속적인 관찰이 필요하며 이는 치매노인을 돌보는 가족의 부담과 스트레스를 더 심하게 하며[7], 치매노인을 돌보는 것은 가족에게 있어 혈압상승[23] 등의 신체적인 영향뿐만 아니라 우울증에 걸릴 가능성이 높음 [24] 등의 **정신적인 건강에도 영향을 미친다**는 선행연구 결과를 고려했을 때 치매노인을 돌보는 가족의 배회로 인한 고통을 완화시키고 신체적·정신적 건강관리를 위한 노력이 더욱 필요할 것으로 사료된다.

- 논문 :『가족 돌봄 제공자의 치매노인 배회관리 경험』

02 프로젝트 목표

인지 장애로 판단이 어려운 치매 환자의 마음을
스스로 헤아려 각종 위험 상황에 대비하고
실종 자체를 미연에 방지하는 알림 서비스

“ 헤아Ring ”



02 서비스 소개



치매 환자

언제 나타날 지 모르는 섬망 증세와
그에 따른 배회에 대한 **심리적 안전망이 필요하다.**



단말기

치매 환자가 사용하는 단말기로
치매환자가 외출 시 **치매 환자의 안전을 살피는 역할을 한다.**

(주요기능 : 조간녹음진행, GPS 값 추출)



보호자

치매환자의 보호자로
치매 환자의 언제 나타날 지 모르는 섬망 증세와 배회 때문에
개인의 시간이 항상 부족한 상태이다.



App ([헤아Ring App Service](#))

보호자는 App을 통해 치매 환자의 **상태를 확인할 수 있다.**
따라서 보호자는 조금이나마 개인의 시간을 확보할 수 있으며
보호자의 간병 스트레스를 완화하는 역할을 한다.

(주요기능 : 치매 환자 외출 알림, 위험 감지 알림, 현재 위치 확인 등)

03 벤치마킹 및 차별성

● 기존 서비스의 문제점

- 위치 추적 중심의 기능**
실종 예방에만 초점이 맞춰져 있어 GPS 기능만 제공한다.
- 높은 가격대**
20~60만원 정도의 고가의 가격으로 보급률 또한 3% 정도로 낮다.
- 기존 서비스 기기의 한계**
외부요인 및 섬망 상태는 감지하지 못하는 한계가 있다.

기존 치매 환자 배회 감지기와의 차별점				
종류	스마트지킴이	소지형	깔창형	매트형
이미지				
장점	생체정보를 활용한 고도화된 기능 제공	저렴한 가격, 용이한 휴대성	착용 거부감이 적음	압력센서를 통한 사용자의 움직임 감지 낙상예방 가능
단점	비싼 가격대 (20만원), 신체 정보로 감지할 수 있는 위험에 한계 존재.	GPS 기능만 제공		비싼 가격대 (30~60만원), 외출 이후에는 대응불가
'헤아Ring'의 차별점	<ul style="list-style-type: none"> 낮은 가격대 제공 (10만원 미만) 발화를 통한 섬망 상태, 외부 위험 요소 감지 	<ul style="list-style-type: none"> 발화 분석을 통해 실종 뿐만 아니라 위험 상황까지 탐지 가능 위험/실종 발생 시 보호자에게 즉각적인 알림 송수신 녹음 파일 확인으로 위험 상황에 직접 대응 가능 	<ul style="list-style-type: none"> 낮은 가격대 제공 (10만원 미만) 외출 이후의 위험 상황 대비 가능 	

03 벤치마킹 및 차별성

● ‘헤아Ring’의 차별성

■ 음성 기반 위험 상황 판단 AI 도입

딥러닝 기반의 음성 분석 시스템을 통해
섬망 및 추가적인 외부 위험 상황을 탐지할 수 있도록 설계

- **섬망 상태 대비**

섬망 상태에서 보이는 비정상적인 발화 데이터 학습

- **외부 요인 위험 판단 가능**

발화 데이터를 바탕으로 배회 시 외부 요인 위험 요소 데이터 학습

[AI 모델 학습 데이터 유형 및 적용 범위]

데이터 유형	세부 분류	설명
배회 형태 (섬망 상태)	주변의 도움 요청, 길 잃음, 사고 위험, 무언가를 찾음	치매 환자가 길을 잃거나, 특정 물건/장소를 찾으려 하며 무맥락적인 발화를 할 가능성이 높음.
위급 상황	신고 요청, 응급 의료, 도움 요청	긴급 상황에서 환자 스스로 도움을 요청하거나 신고의 필요성을 발화하는 경우.
외부 요인	간힘, 화재, 재해 강력범죄(폭력/절도/강도), 낙상, 강제추행(성범죄)	치매 환자가 외부 환경으로 인해 직접적인 위협을 받는 상황.

03 벤치마킹 및 차별성

● ‘헤아Ring’의 차별성

■ 실시간 알림 (외출, 귀가, 위험감지)

- 환자의 발화가 위험상황으로 판단될 시 즉시 보호자에게 위치, 시간, 위험 요인을 포함한 알림 전송
- 보호자가 **신속한 대처**를 할 수 있게 지원

■ 합리적인 가격

‘스마트 지킴이’····· 20만원 이상
‘매트형 감지기’····· 30 ~ 60만원

- 약 8만원 정도의 단가로 사용자들의 **경제적 부담 완화** 가능
- 대량 생산의 가능성을 염두해두면 더 낮은 단가로 지원 가능

■ 농촌 지역에서도 효과적인 대응 가능

- 실종경보문자와 같이 인구밀도에 영향을 받지 않아, 인적이 드문 농촌 지역에서도 효과적으로 대응이 가능하여 **안정적인 서비스를 제공**한다.

■ 위험 상황 데이터 저장

- 환자가 범죄와 같은 상황에 처했을 때 데이터 저장을 통해 **법적 증거 자료로 사용**될 수 있다.
- 데이터가 누적되면, 치매환자의 행동 패턴을 이해하고 **예방적 조치를 취하는 의료 자료로 사용**할 수 있다.

시스템 분석 및 설계

- 서비스/사용자 시나리오
- 시스템 아키텍처
- 개발 환경
- 알고리즘 시나리오
- App 구성도
- DB 설계

01 서비스 시나리오



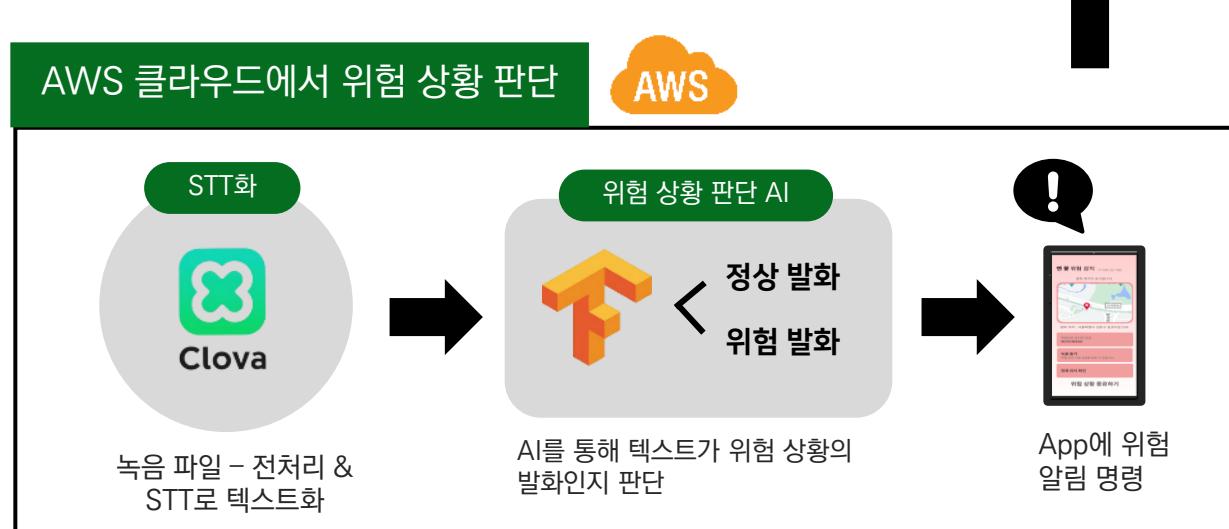
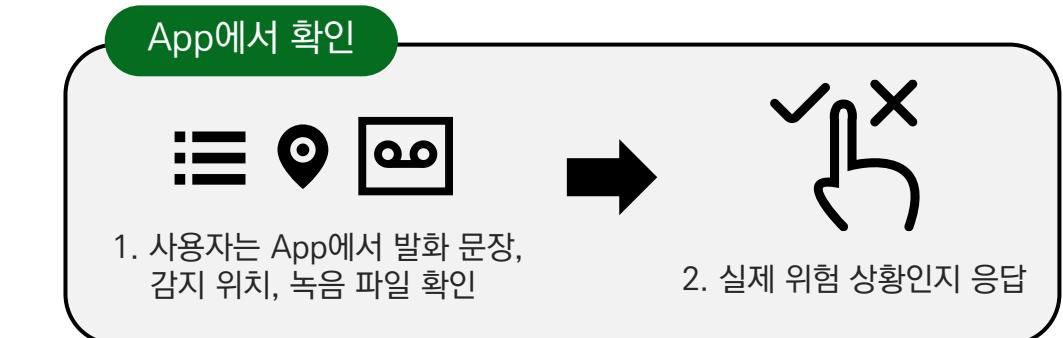
환자

- 치매 환자
- 단말기 이용



사용자

- 환자의 보호자
- App 이용



01 사용자 시나리오

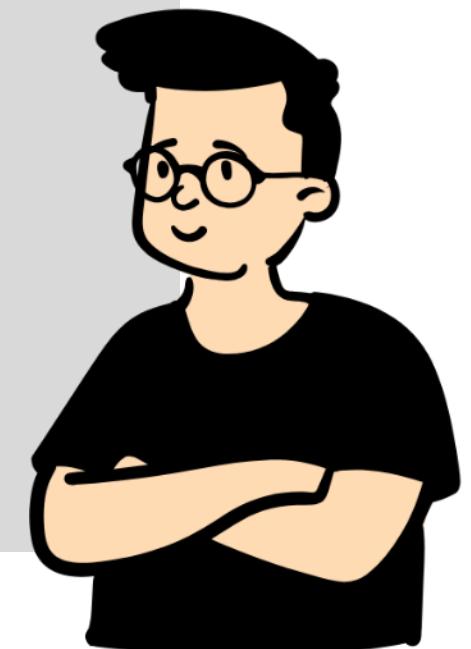
70대 노인 A씨



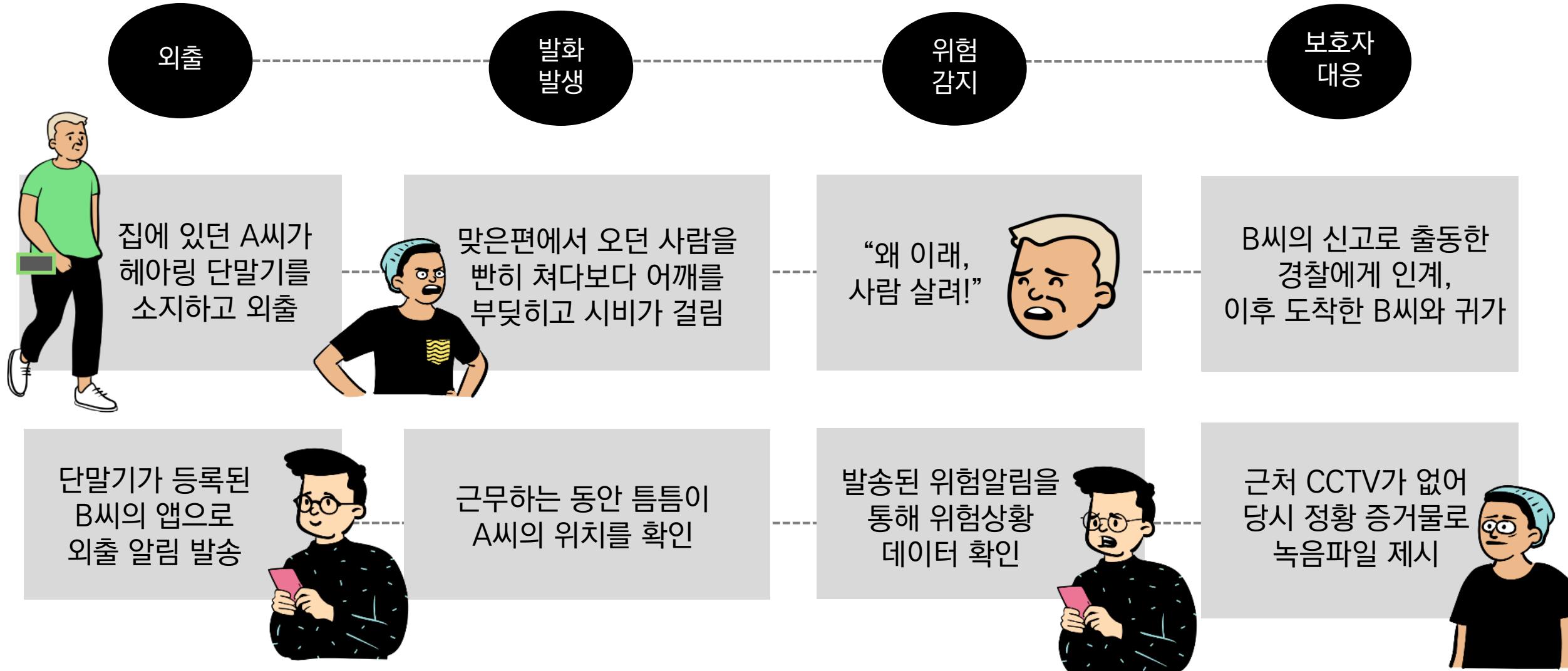
- 경증 치매 환자
- 단독 외출 가능
- 종종 섬망 증세를 보임

50대 직장인 B씨

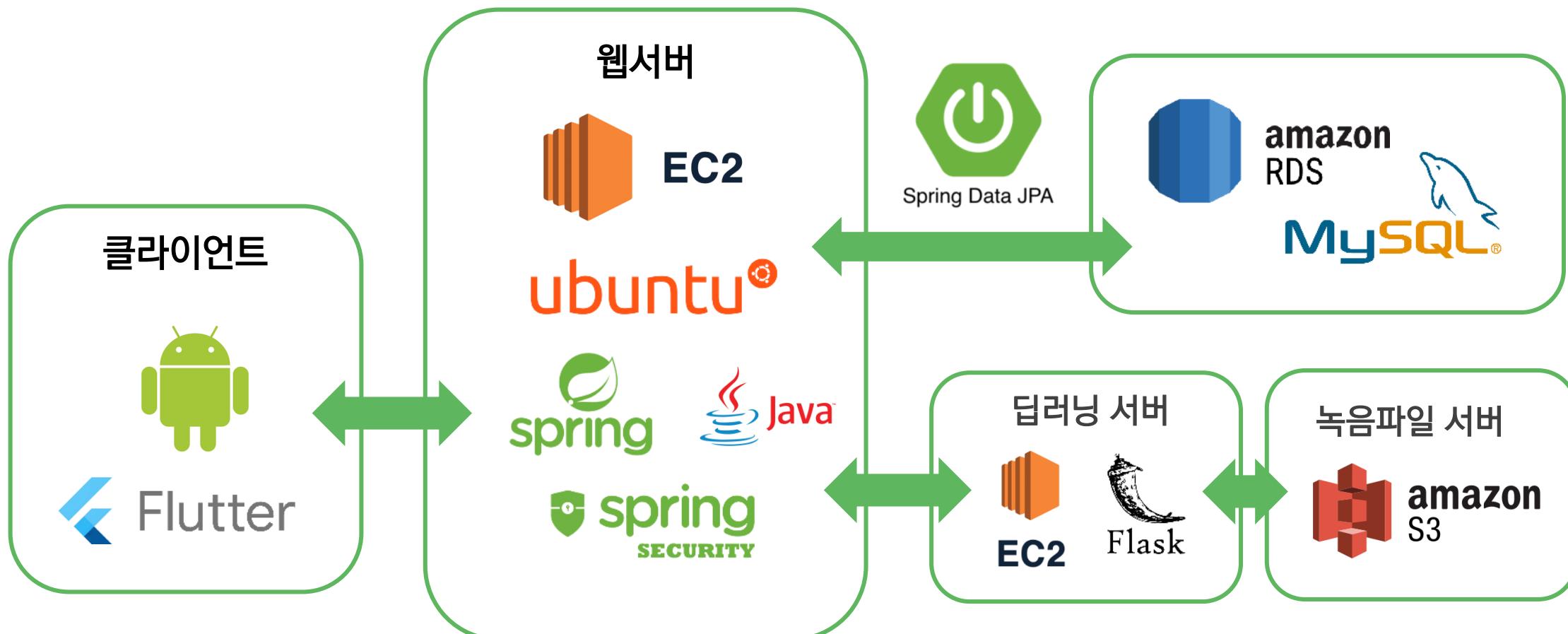
- A씨의 자녀
- A씨를 부양 중
- 최근 인근 동네의
치안 문제 염려 중



01 사용자 시나리오



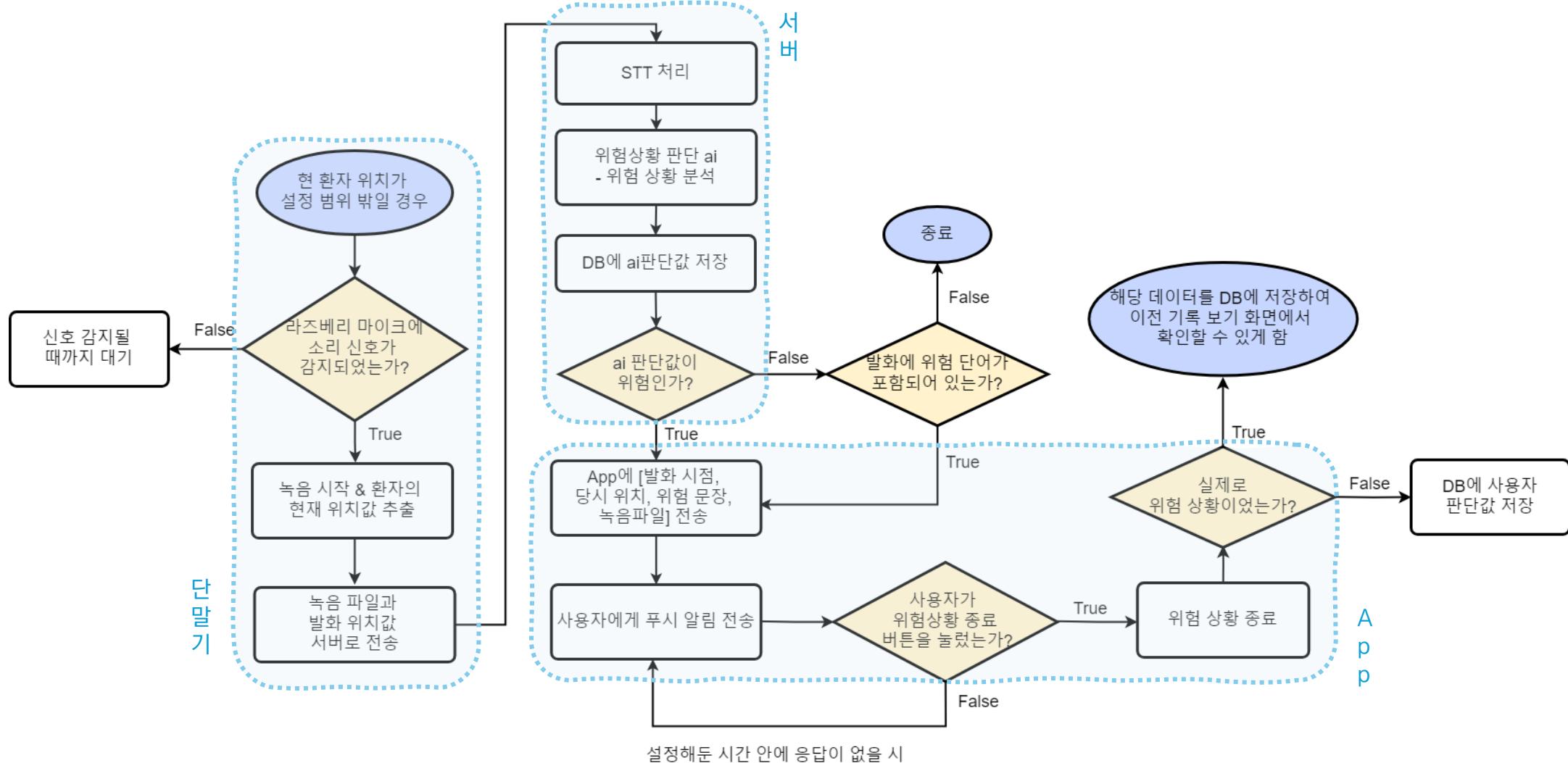
02 시스템 아키텍처



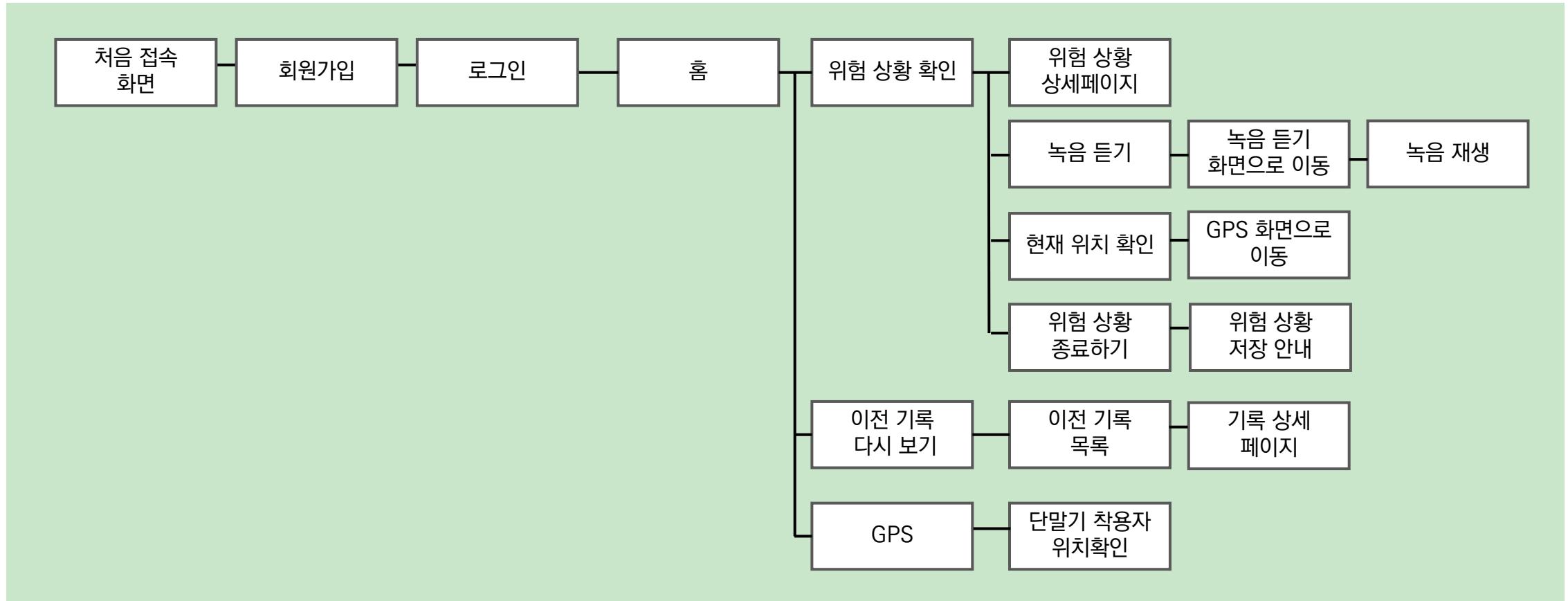
03 개발 환경

구분		상세내용
S/W 개발환경	OS	window 11, Linux, Ubuntu
	개발환경	Pycharm, Colab, IntelliJ IDEA, 안드로이드 스튜디오, Linux
	개발도구	Maven, Spring Boot, Flutter, Firebase
	개발언어	Python, Java, Dart
	기타사항	서버 – AWS 인공지능 모델 – PyTorch, HuggingFace
H/W 구성장비	디바이스	Usb 마이크, 라즈베리파이4, Gps모듈, 충전모듈, 쿨링팬
	통신	무선 랜 wifi, MQTT, WebSocket
	언어, 도구	Vscode, Python
프로젝트 관리	의사소통	Notion, Discord, Slack, Jira

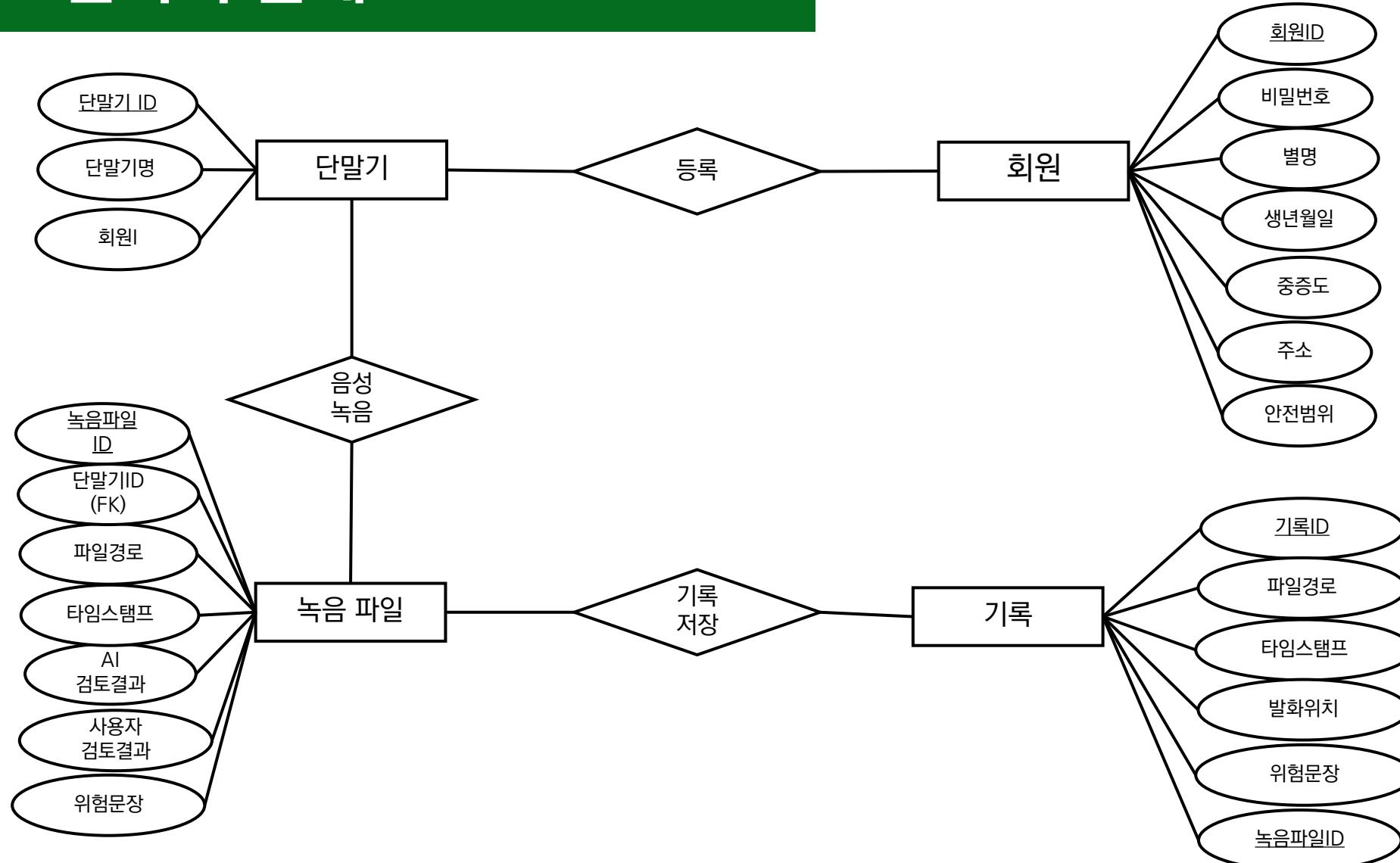
04 알고리즘 시나리오



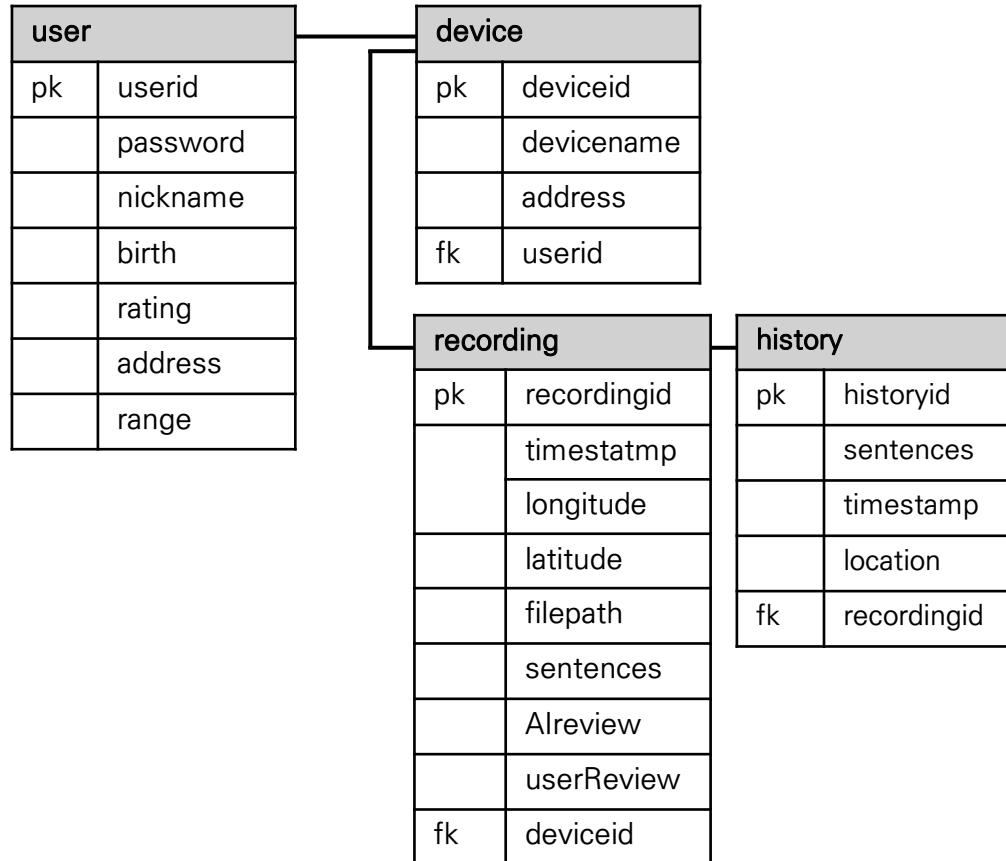
05 App 구성도



06 DB - 엔티티 관계도



06 DB - 테이블 정의서



〈recording〉 테이블

순번	필드명	데이터타입	기본값	PK/FK
1	recordingid	int		PK
2	timestatmp	char		
3	longitude	long		
4	latitude	long		
5	filepath	varchar		
6	sentences	varchar		
7	Alreview	bool	0	
8	userReview	bool	0	
9	deviceid	int		FK

〈histiory〉 테이블

순번	필드명	데이터타입	기본값	PK/FK
1	historyid	int		PK
2	sentences	int		
3	timestamp	char		
4	location	varchar		
5	recordingid	int		FK

(이하생략)

프로젝트 구현

- 로그인 및 회원가입
- 메인 화면
- 위험감지 화면
- GPS 화면
- 이전기록 다시보기 화면
- 개발 요약

01 로그인 및 회원가입



[처음 접속 화면]

[회원가입 - 계정]

[회원가입 - 돌봄 대상자 정보 입력]

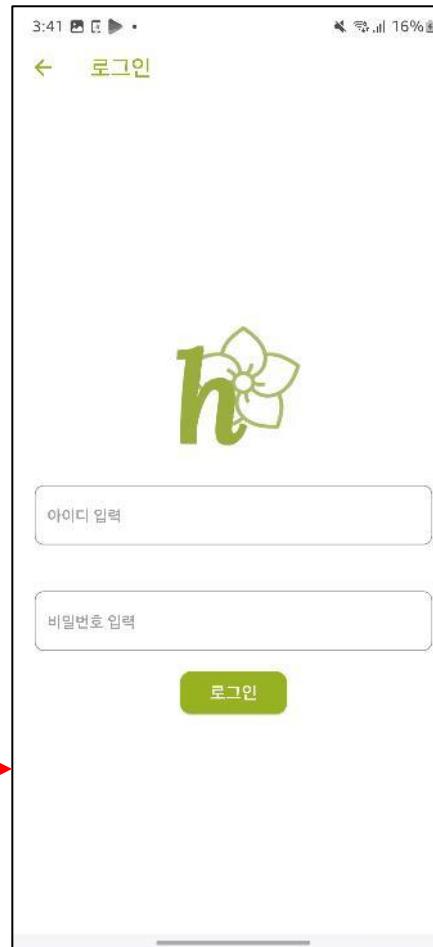
이전

다음

이전

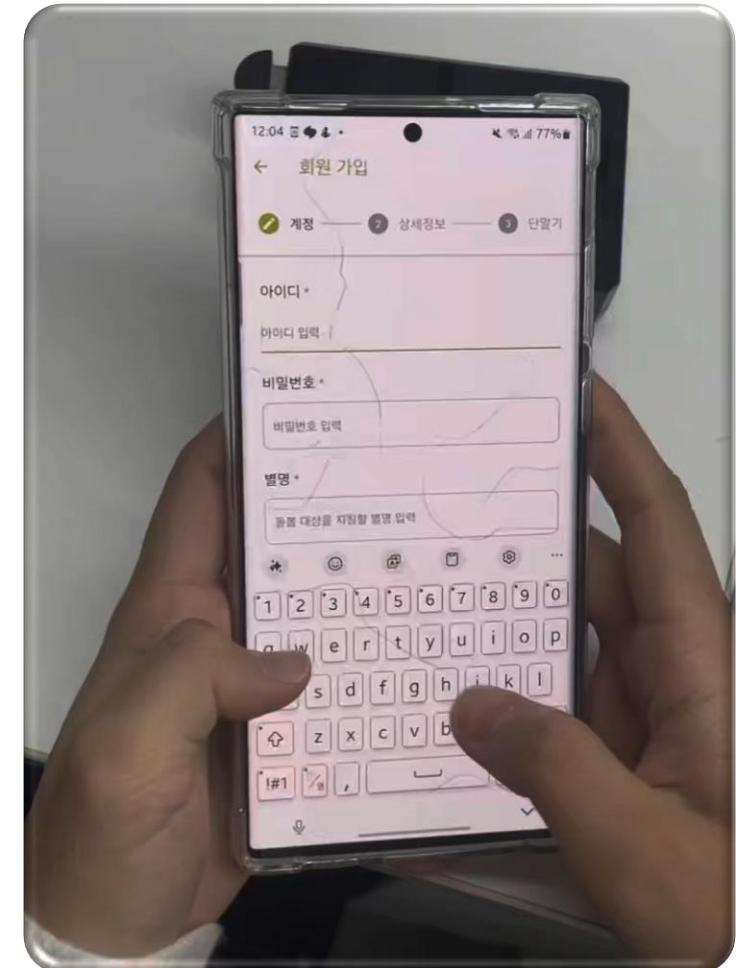
다음

01 로그인 및 회원가입



[회원가입 – 단말기 등록]

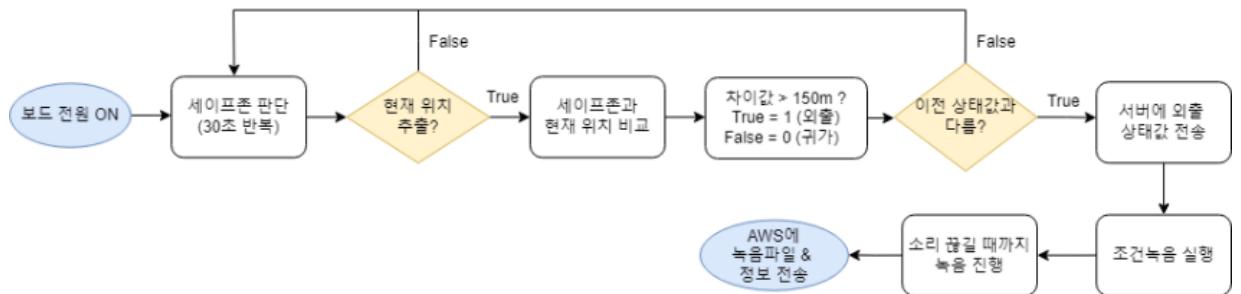
▶ 실제 시연 (단말기 등록 : 14초)



[로그인 화면]

02 메인화면

(1) 단말기 : 위치 추적 및 세이프존 여부 확인



● 외출 상태

```
(myenv) pi@raspberrypi:~/handmade $ python3 restart4.py
Current GPS location: (37.57302727371158, 126.9773387671158)
==is_outside_safe_zone RUN ==> 0.19847629164222613
==send_status_to_server RUN== 1
Request error: Expecting value: line 1 column 1 (char 0)
Outside Safe Zone. Recording will start.
```

서버에 외출 상태값(1) 전송

현재 위치 추출

현재 위치와
Safe존 사이의 거리
(0.15km ↑)

● 귀가 상태

```
Current GPS location: (37.570779965679, 126.97742358341758)
==is_outside_safe_zone RUN ==> 0.051116778981657174
==send_status_to_server RUN== 0
```

(2) 서버 : 세이프존 여부에 따른 데이터베이스 저장

● 기기 상태 업데이트 API

Curl

```
curl -X 'PUT' \
'http://localhost:8080/api/devices/device/2/status?deviceStatus=0' \
-H 'accept: application/json;charset=UTF-8' \
-H 'Content-Type: application/json;charset=UTF-8' \
-d 'true'
```

Request URL

```
http://localhost:8080/api/devices/device/2/status?deviceStatus=0
```

Server response

Code	Details
200	Response body can't parse JSON. Raw result: 외출 상태가 업데이트되었습니다.

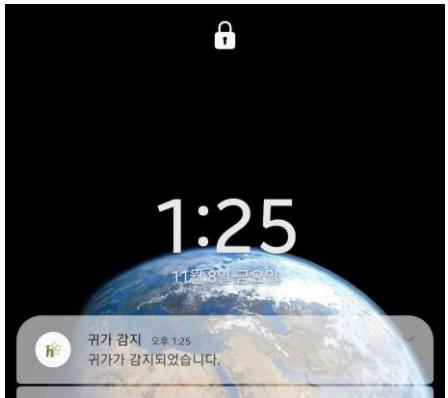
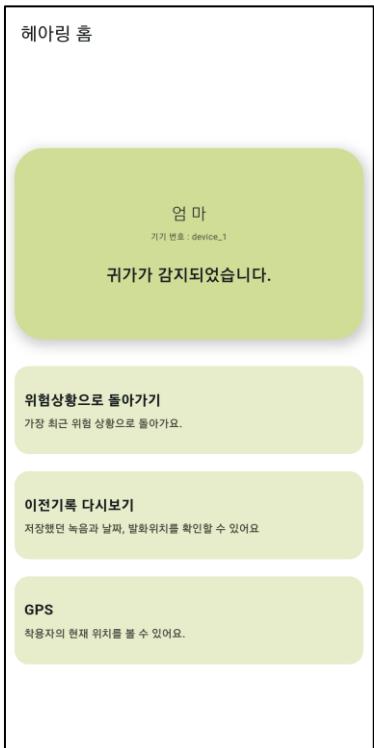
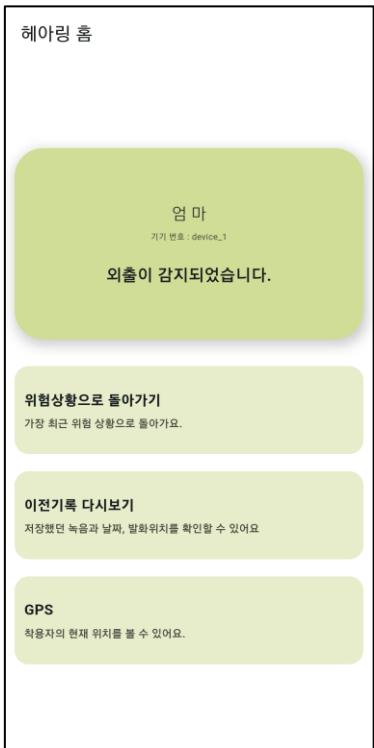
● Device 테이블

device_id	device_address	device_name	device_num	device_status
2	서울시 용산구	엄마	804c0513-ec99-4e83-ab82-606d2cc85372	1
*	HULL	HULL	HULL	HULL

device_id	device_address	device_name	device_num	device_status
2	서울시 용산구	엄마	804c0513-ec99-4e83-ab82-606d2cc85372	0
*	HULL	HULL	HULL	HULL

02 메인화면

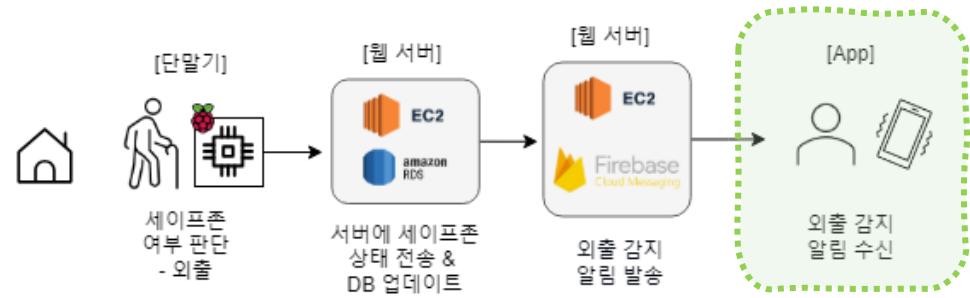
(3) App : 외출 감지 알림 수신 (시연)



[외출 감지]

[귀가 감지]

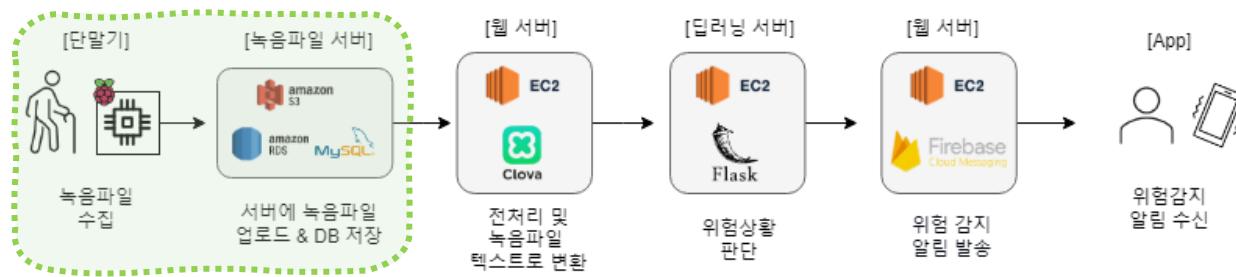
[외출 및 귀가 알림]



▶ 외출 감지 알림 + 메인화면 시연



03 위험 감지 화면



(1) 단말기 : 소리 감지 및 녹음 파일 저장

● 소리 감지 및 녹음 파일로 저장

```

Listening...
Sound detected. Starting recording.
Silence detected. Stopping recording.
Saved helpsign4444.wav and helpsign4444.mp3
helpsign4444.wav and helpsign4444.mp3 successfully created.
helpsign4444.mp3 file S3 upload complete with metadata.
Local files helpsign4444.wav and helpsign4444.mp3 deleted.

```

● 서버(AWS S3)에 녹음 파일 업로드된 모습

<input type="checkbox"/>	helpsign4444	mp3
	.mp3	
2024. 11. 8.		35.1KB
pm 1:09:29		PM KST

메타데이터 (4)		
유형	키	값
시스템 정의	Content-Type	binary/octet-stream
사용자 정의	x-amz-meta-latitude	37.57302727371158
사용자 정의	x-amz-meta-longitude	126.9773387671158
사용자 정의	x-amz-meta-recording_time	2024-11-08 13:09:14 KST

위치 & 시간도 함께 서버에 업로드됨

▶ 소리 감지될 시 녹음 시연



03 위험 감지 화면



(2) 서버 : 녹음파일 STT화 → 위험상황 판단 → 위험 감지 알림 발송

● 녹음파일 업로드시 AWS Lambda 로직 수행

녹음 파일에서 메타데이터 [경도/위도/time/파일경로] 추출
→ DB에 recording 레코드 추가

해당 recording으로 Clova Speech API 호출 → STT 진행

텍스트 → 위험상황 판단 ai호출
→ 위험여부 판단

```
(base) ubuntu@ip-172-31-10-13:~/Desktop $ curl -d '{"sentences": ["여기가 어디지?", "이쪽 골목이 아니었나?", "길을 못 찾겠네."]}' -H "Content-Type: application/json" -X "POST" http://127.0.0.1:5000/predict
{"isDangerous": "True", "sentences": [{"prediction": "danger", "text": "여기가 어디지?"}, {"prediction": "danger", "text": "이쪽 골목이 아니었나?"}, {"prediction": "danger", "text": "길을 못 찾겠네."}]}
```

[DB 값 업데이트]

recording_id	ai_review	filepath	latitude	longitude	recording_status	text
2	1	13://ha...	126.977338...	37.57302...	위험 감지	여기가 어디지?
created_at	user_review	device_id				
2024.10.29 15:40	0	2				

→ Recording 테이블 : ai_review에 위험 감지값(1) 업데이트
→ 위험 감지 알림 App에 전송

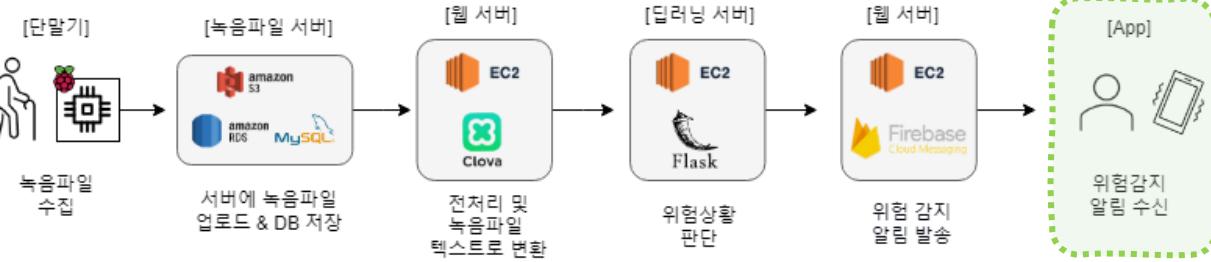
검토 대기

처리 중

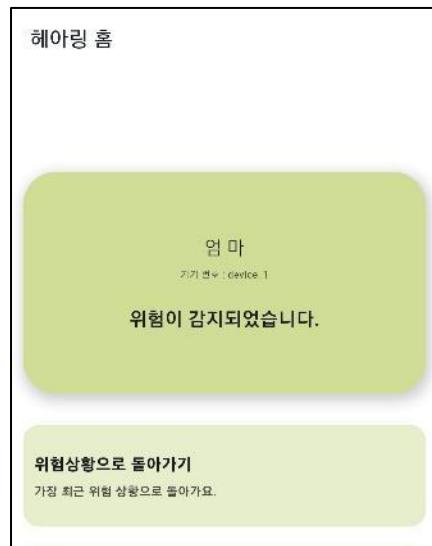
위험 감지

알림
발송

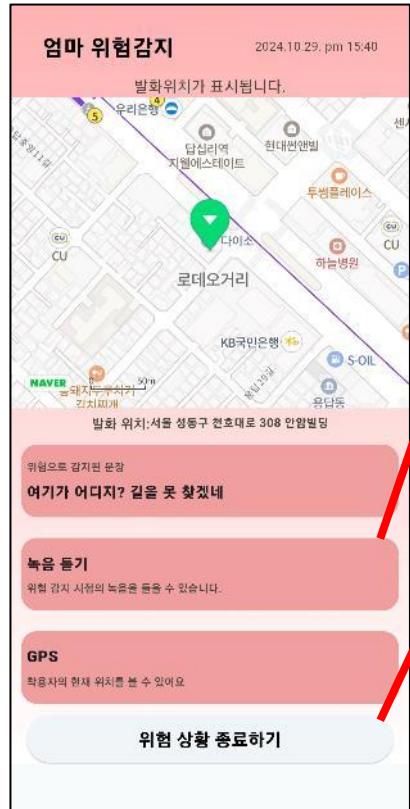
03 위험 감지 화면



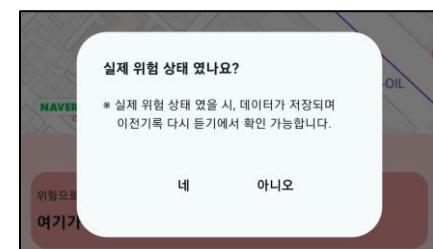
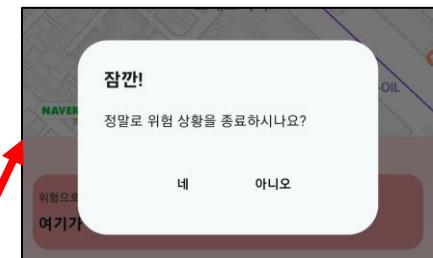
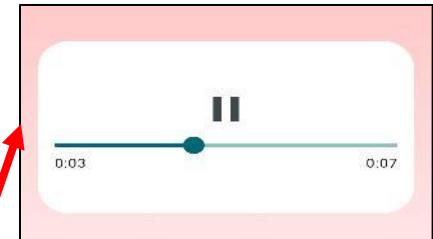
(3) App : 위험 감지 화면 (시연)



[위험감지 알림 및
메인화면]



[위험 감지 화면]



[위험상황 종료]

▶ 위험 감지 알림 + 화면 시연



03-1 위험 상황 판단 AI

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT _{BASE}	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

위험 상황 판단의 근거

위험 상황을 판단하는 과정은 위험 상황 데이터의 패턴과 Bert 모델의 구조에 기반하여 이루어짐. 다만 이 과정은 학습 데이터에 매우 의존적이기에 데이터 구축 과정에서 그 품질을 보증하고자 함.

데이터 패턴

위험 상황 발화 특징

위험 상황의 발화는 특정 언어 패턴, 단어 선택, 문맥적인 힌트가 포함될 수 있음

길잃음

내가 치매약을 먹는데 집 가는 길이 기억이 안 나. 나 길 찾는 것 좀 도와줄래요?

도움요청

엠뷸런스 좀 불러줘요. 너무 아파서 움직이질 못 하겠어.



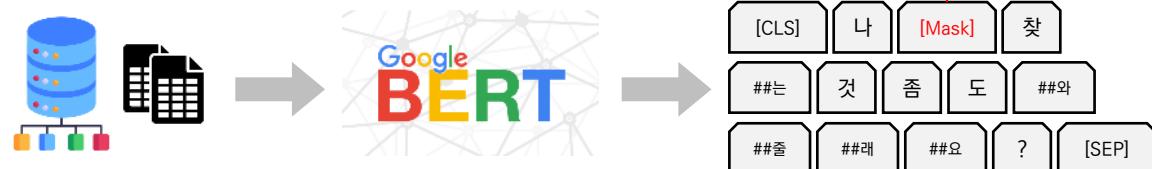
화재

할머니 건물에 불이 났어요. 어서 대피해야 해요.

모델 구조

Bert 모델은 이러한 특징을 인식할 수 있음

Bert는 대규모 텍스트 데이터셋으로 다양한 언어 패턴을 학습. 이 과정에서 문장의 문맥, 어휘, 구조를 이해.



학습 데이터

꼼꼼한 데이터 검수 작업을 통해 데이터의 품질 보증

따라서 자료조사를 통한 위험 상황 세분화 작업과 꼼꼼한 데이터 선별, 검수 작업을 거쳐 데이터의 명확성을 확보하고자 함.

세분화

논문을 통해 치매 노인에 특화된 위험 상황 유형을 파악 + 데이터 추출 (길잃음, 무언갈 찾음, 사고위험, 주변의 도움)

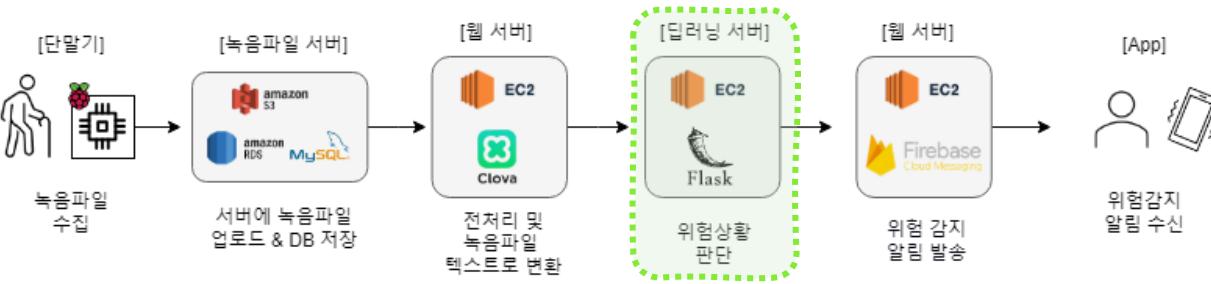
정제

부적합 데이터 선별 - 위험 상황의 대표성을 띠는 데이터만 추출

검수

추가 검수 작업을 거쳐 위험 상황에 부합하는지 확인

03-1 위험 상황 판단 AI



(1) 발화 데이터셋 구축

01 데이터 수집

정상상황 발화 데이터 수집 😊

- AI HUB : 자유대화 음성 (노인남여)
- 깃허브 인공 데이터 : 일상 발화 데이터

위험상황 발화 데이터 수집 🤔

- AI HUB : 위급상황 음성/음향 데이터
- AI HUB : 위급상황 음성/음향 데이터 – 119지능형 신고 접수 음성 인식 데이터
- 치매안심센터 : ‘배회 모의 훈련’ 동영상
- 치매 노인 돌봄제공자의 배회 관리 논문

- 논문**
- 『가족 돌봄 제공자의 치매노인 배회관리 경험』 천홍진·송준아(2015). 노인간호학회지 제17권 제3호
 - 『요양기관 간호사가 인식하는 치매환자의 배회 사정 경험 구조』 김선희(2021). 고려대학교 대학원 박사학위 논문
 - 『재가 치매노인 가족을 위한 배회관리 지침 개발』 천홍진(2015). 고려대학교 대학원 석사학위 논문

02 데이터 전처리



수작업 정제 : 데이터 선별 방법



- 치매환자(노인)과 직접적으로 관련된 내용으로 선택
⇒ 젊은 사람이 술 먹고 쓰러져있네. 여기. (X)
⇒ 어제 코로나 백신 주사를 맞고 숨을 못 쉬겠어요. (X)
⇒ 어르신이 지금 쓰러져셨거든요. 갑자기. (O)
⇒ 노인네가 지금 거시기 하는데요. 숨을 못 쉬고. (O)

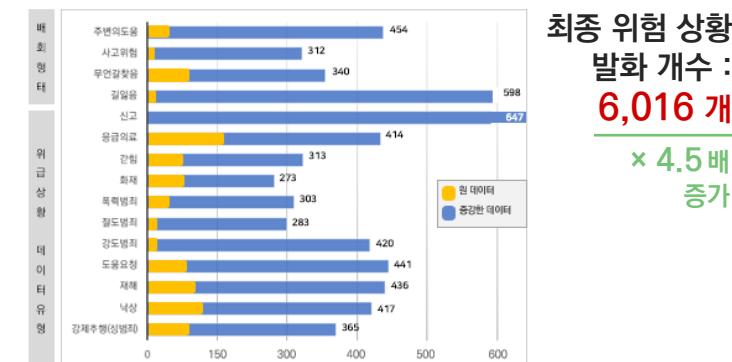
- 위험 상황을 확실히 드러내는 발화로 선택
⇒ 물이 들어 오고 있어요. (X)
⇒ 여기 산사태로 사람이 흙에 파묻혔어요. 도와주세요. (O)

수작업 증강

- 데이터의 길이가 짧은 것이 多
- 원 데이터를 가이드라인 삼아 ChatGPT & 수작업 증강 진행

분류	원 데이터	수작업으로 증강한 데이터
낙상	으 팔목이야	팔목이 성치 않아.
낙상		팔이 너무 아파.
낙상		넘어진 곳이 크게 부으셨어.
낙상	못 일어나겠어	여기서 한 발자국도 못 움직이겠어.
낙상		움직일 수가 없어요.

라이브러리 증강



03-1 위험 상황 판단 AI

(2) KoBert 모델 선택

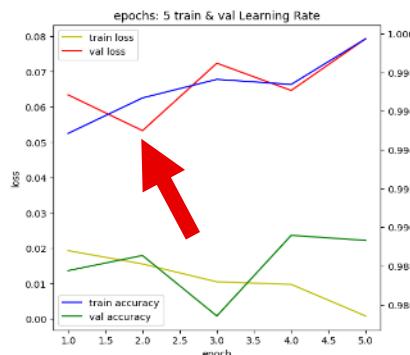


- Bert 모델을 파인튜닝하여 위험 상황과 정상 상황의 발화를 분류.
- 한국어로 훈련된 SKT의 'KoBert' 모델 선택해 파인튜닝 진행.

정상상황 발화 13,776개 + 위험상황 발화 6,016개 \Rightarrow 총 19,792개

✓ 하이퍼 파라미터 결정

- 데이터셋 분할 : 준비된 데이터셋에서 학습 6 : 검증 2 : 테스트 2로 분할
- 에포크 수 : 5번의 학습 중 3번부터 과적합이 발생 \Rightarrow 2번으로 결정



- 최적화 함수 설정 : 가장 많이 쓰는 알고리즘인 AdamW로 설정
- 손실 함수 설정 : CrossEntropy로 설정

(3) 모델 훈련

✓ 과적합 방지 위한 모델 튜닝

- 조기 종료 함수 도입 : val_loss값을 추적
- 초기 학습률 감소 : $3e-5$
- 드롭아웃 비율 증가 : 0.7

훈련 진행 결과

loss 0.10과 accuracy 0.98로 좋은 성능을 보임

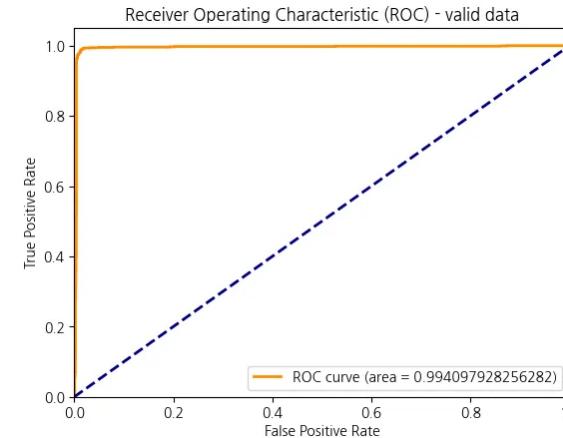
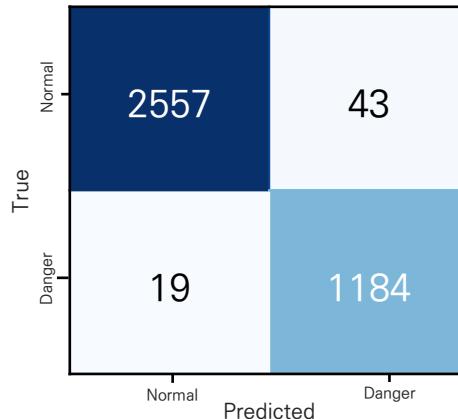
훈련 시간	train_loss	val_loss	val_accuracy
10.12분	0.0127	0.1023	0.9845

✓ 문장 직접 넣어 Test해보기

- input : 할머니 위험해요! \Rightarrow logits: [0.000254 0.999746] 위험
- input : 딸네 집에 갈라고 했는데 여기가 어디지? \Rightarrow logits: [0.0002588 0.9997411] 위험
- input : 요즘에 임영웅인가 좋더라. \Rightarrow logits: [0.9998001 0.0001998] 정상
- input : 말복이니까 닭 먹었지 어제는. \Rightarrow logits: [0.9997994 0.0002006] 정상

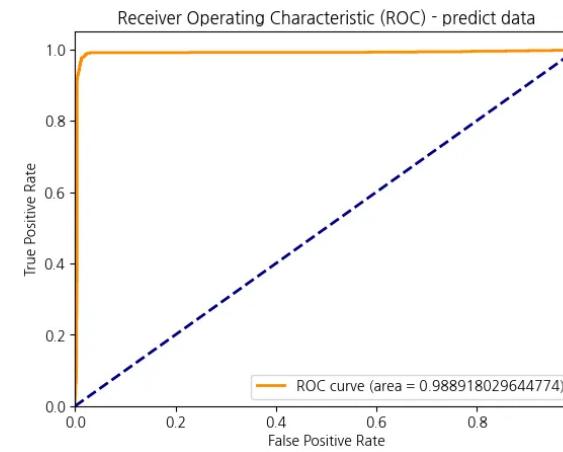
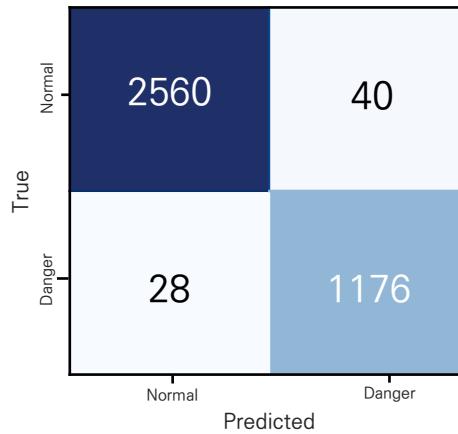
03-1 위험 상황 판단 AI

검증 데이터



평가지표	validation
accuracy	0.98
recall	0.98
precision	0.96
f1-score	0.97

테스트 데이터



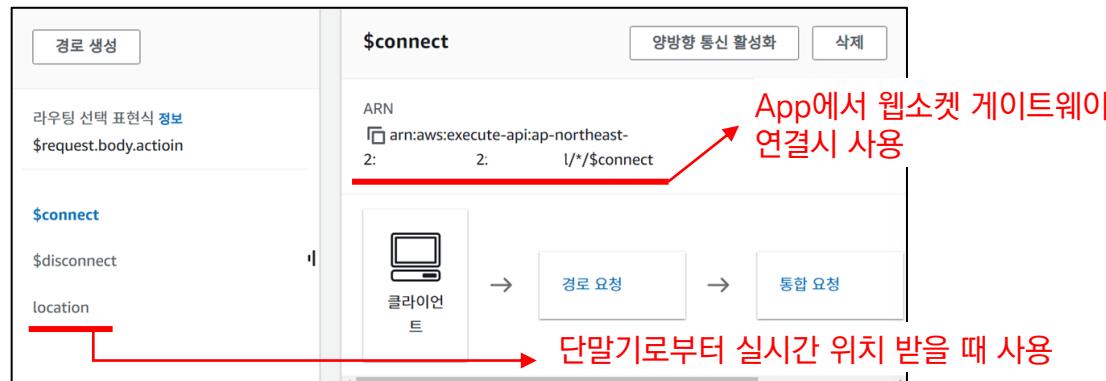
평가지표	test
accuracy	0.98
recall	0.98
precision	0.97
f1-score	0.97

위험 데이터에 과적합된 이슈가 발생하여 정상 발화를 추가해 해결 완료하였음. 현재 모델은 좋은 성능을 보이고 있음.

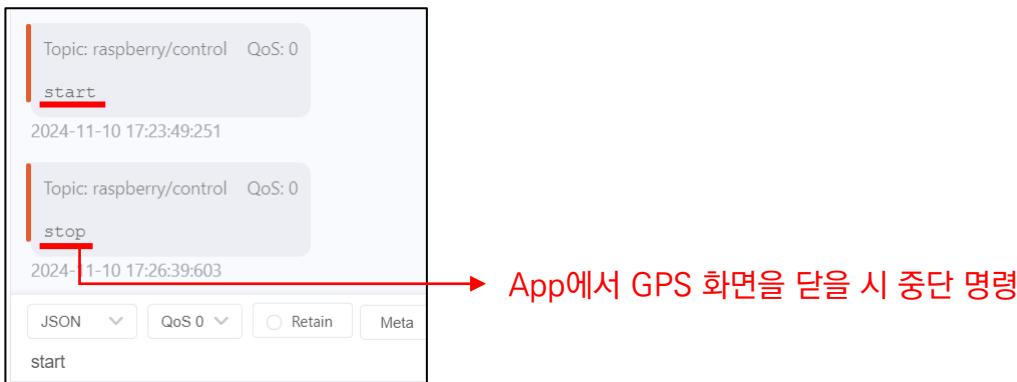
04 GPS 화면

(1) 서버 : App에서 GPS 화면이 열리면 웹소켓 통신 시작

● AWS Gateway – 웹소켓 통신 시작



● MQTT 통신 – 브로커 서버에 시작/중단 명령 게시



(2) 단말기 : MQTT 통해 명령 수신 + 실시간 위치값 전송

- MQTT 브로커 서버에 게시된 명령 수신
- 웹소켓 서버로 실시간 위치값 전송

● start 명령 수신 : 위치데이터를 1초마다 전송

```
Received MQTT command: start
Starting to send location data...
Connecting to WebSocket at wss://... 서버주소 ...
Connected to WebSocket
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:50:21'}
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:50:22'}
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:50:23'}
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:50:24'}
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:50:25'}
```

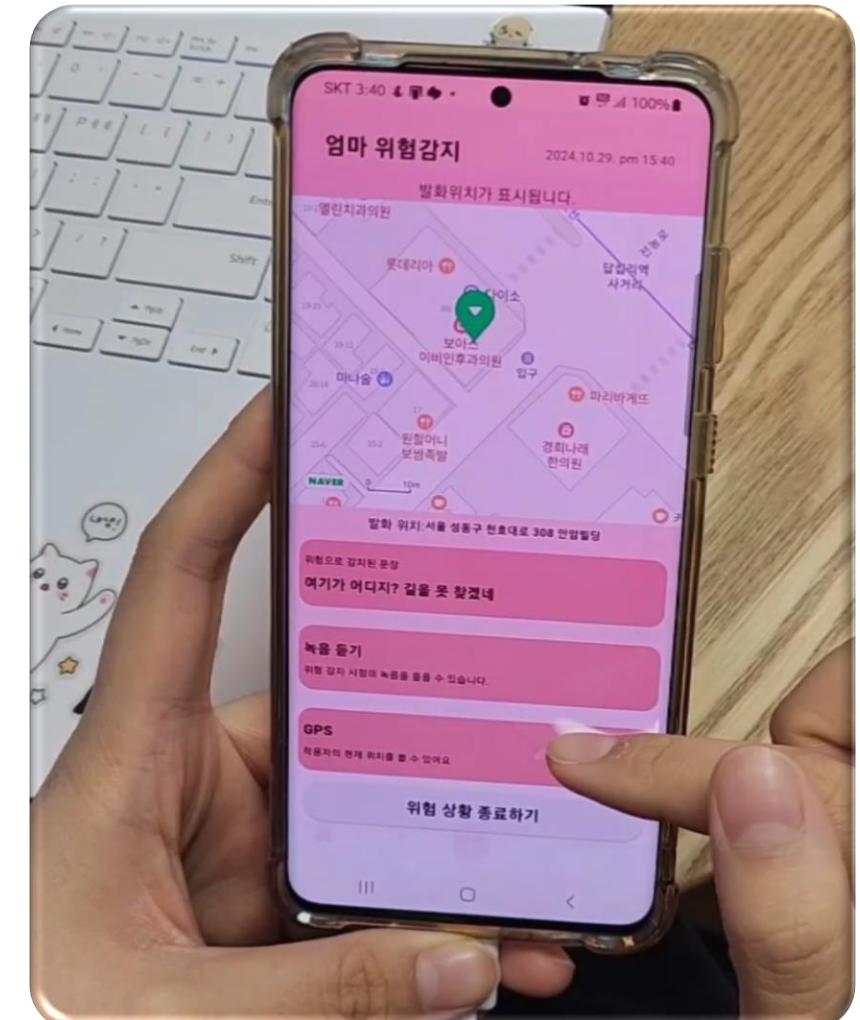
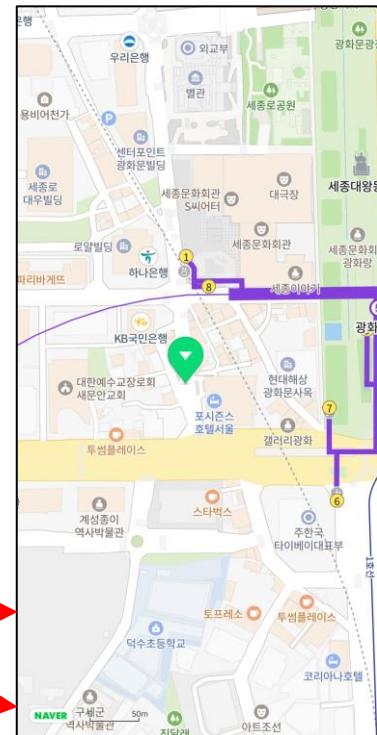
● stop 명령 수신 : 위치 데이터 전송 중단

```
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:51:00'}
Sending location data: {"latitude": 37.57302727371158, "longitude": 126.9773387671158, "timestamp": '2024-11-10 22:51:01'}
Received MQTT command: stop
Stopping data transmission.
```

04 GPS 화면

▶ GPS 화면 시연

(3) App : GPS 화면 시연



[메인 화면]

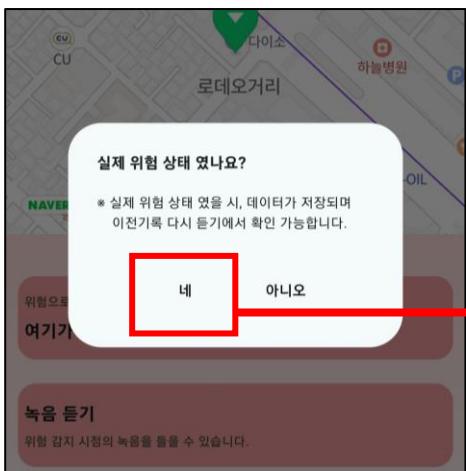
[위험 감지 화면]

[GPS 화면]

05 이전기록 다시보기 화면

(1) 서버 : 실제 위험 상황일시 이전 기록 데이터로 저장

● 기록으로 저장 API



● History 테이블에 위험 데이터가 저장된 모습

(2) 서버 : App에서 이전기록 화면 열릴 시 데이터 조회

● 이전기록 전체조회 API

```
Curl
curl -X 'GET' \
  'http://localhost:8080/api/histories/device/2' \
  -H 'accept: application/json; charset=UTF-8'

Request URL
http://localhost:8080/api/histories/device/2

Server response

Code Details
200 Response body
[
  {
    "historyId": 4,
    "location": "서울특별시 종로구 세종대로 23길",
    "text": "여기 가 이디지? 이쪽이 우리집이 분명 있는데",
    "timestamp": "2024. 8. 20. am 11:14:28 AM KST"
  }
]

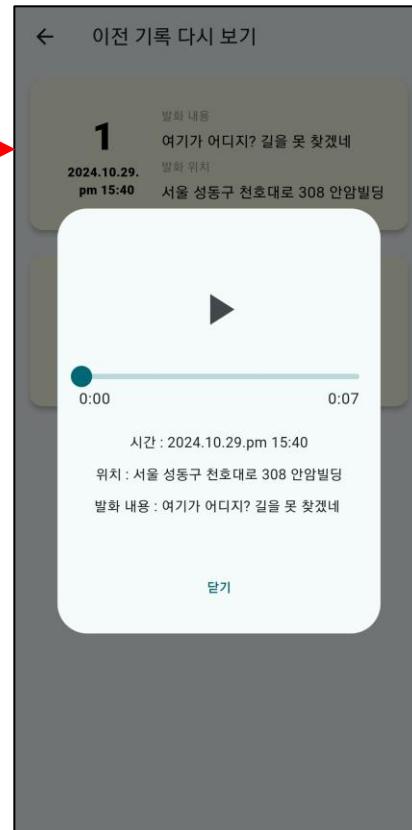
Response headers
connection: keep-alive
content-type: application/json; charset=UTF-8
date: Fri, 08 Nov 2024 15:54:06 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

● 이전기록 세부조회 API

GET /api1/histories/{historyId} 특정 기록 조회	
ID로 특정 기록을 조회	
Parameters	
Name	Description
historyId <small>required</small> <small>integer (\$int64) (path)</small>	<input type="text" value="historyId"/>
Responses	
Code	Description
200	성공적으로 조회됨
Media type	
<input checked="" type="checkbox"/> application/json; charset=UTF-8	
Controls Accept header.	
Example Value Schema	
<pre>{ "historyId": 0, "timestamp": "string", "location": "string", "risk": "string", "text": "string", "device": "string" }</pre>	

05 이전기록 다시보기 화면

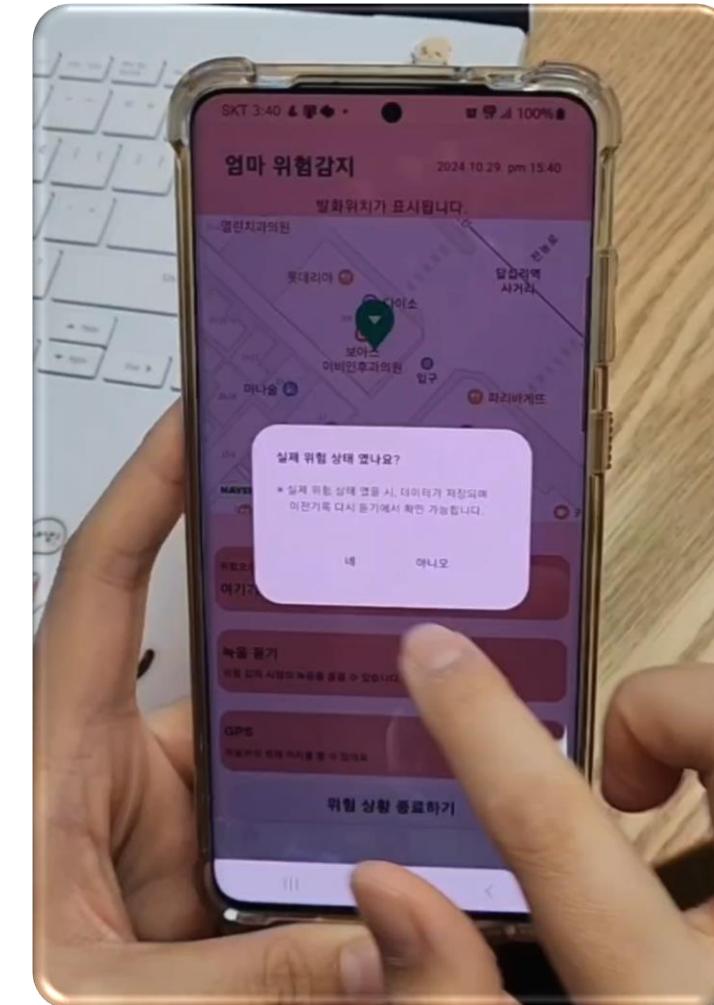
(3) App : 이전기록 다시보기 화면 (시연)



[이전기록 - 전체조회]

[이전기록 - 세부조회]

▶ 이전기록 다시보기 화면 시연



06 개발 요약

애자일 방법론 기반 전체 개발 진행도

BN

총 14개 API 中 14개 개발 완료 (100%)

- 기기관리 4개, 기록 관리 2개, 실시간위치 3개, 알림 읽음 처리 1개, 위험상황 데이터 확인 4개

FN

총 8개 화면 中 8개 개발 완료 (100%)

- 처음 접속 화면, 로그인, 회원가입, 메인화면, 위험감지 화면, 이전기록 화면, GPS 화면

HW

총 6개 기능 中 6개 개발 완료 (100%)

- 조건부 녹음, WebSocket 양방향 통신, 서버와의 데이터 송수신, MQTT 프로토콜을 통한 통신, GPS 현재 위치 추출, 세이프존 여부 판단

AI,
통합

위험 상황 판단 AI 개발 완료 (상세 테스트 진행 중 80%)

통합 진행 중 (90%), 모듈 테스트 진행 중 (17/21)

차수	화면	대분류	우선 순위	기능	구현	담당	상태
삭제	위험감지	HW	=	단말기 조립 (제로보드)	단말기	지원	완료
추가	위험감지	HW	중	라즈베리 파이 4 - 충전 모듈 조립	단말기	지원	30%
추가	위험감지	HW	중	라즈베리 파이 4 - 쿨링팬 조립	단말기	지원	30%
1차	위험감지	HW	상	소리 감지 - 조건 녹음 코드	단말기	지원	완료
추가	위험감지	HW	상	조건 녹음 코드 - 외출해서 작동 Test	단말기	지원	완료
1차	위험감지	HW	상	단말기에서 발화 당시 위치 추출	단말기	지원	완료
1차	GPS	HW	상	단말기에서 현재 위치 추출 (조 간격)	단말기	지원	완료
1차	GPS	HW	상	단말기에서 현재 위치 비교 (외출 감지)	단말기	지원	완료
1차	GPS	HW	상	외출 감지 - 조건 녹음 코드 실행	단말기	지원	완료
1차	서버	상		외출 감지 - 서버에 세이프존 여부(0,1)를 보냄	서버	지원	완료
1차	위험감지	HW	상	단말기 --> 서버 녹음파일 데이터 수신	서버	지원	완료
1차	위험감지	HW	중	단말기 외형 디자인	디자인	지원	완료
1차	위험감지	HW	중	단말기 착용법 고안	디자인	지원	완료
1차	위험감지	BN	상	녹음파일 위험데이터로 변환 처리	로직	지현	완료
1차	위험감지	BN	상	클로바 API 호출 및 결과 저장	API	지현	완료
1차	위험감지	BN	상	AI API 호출 및 결과(위험감지/삭제대기) 저장	API	지현	완료
1차	위험감지	BN	상	위험감지 PUSH 전송 처리	API	지현	완료
1차	위험감지	BN	상	서버 배포	Merge	지현	완료
1차	위험감지	BN	하	보안 설정	Merge	지현	완료
1차	위험감지	FN	상	위험 감지 화면	화면	회주	완료
1차	위험감지	FN	상	위험 감지 화면	화면	회주	완료
1차	위험감지	FN	중	화면 안 텍스트	화면	회주	완료
1차	위험감지	FN	상	녹음듣기 버튼 - 녹음 재생	화면	회주	완료
1차	위험감지	FN	상	위험 상황 데이터 받기 및 배치	Merge	회주	90%
1차	위험감지	FN	상	위험 상황 데이터 저장요청	Merge	회주	90%
1차	위험감지	FN	상	위험 상황 데이터 삭제요청	Merge	회주	90%
1차	위험감지	FN	상	App <--> 서버 연동	Merge	회주	90%
1차	위험감지	FN	상	네이버 지도 API 사용	화면	회주	완료
추가	위험감지	FN	중	좌표 값 리버스 지오코딩 진행	화면	회주	완료
1차	위험감지	FN	상	현재 위치 버튼 - GPS 화면 전환	화면	회주	완료
1차	위험감지	FN	상	위험상황 종료 버튼 - 팝업 제공	화면	회주	완료
1차	위험감지	FN	상	위험상황 종료 버튼 - 상황 종료	화면	회주	완료
1차	위험감지	FN	상	위험상황 PUSH 수신 작업	TEST	회주	90%
1차	위험감지	AI	상	위험상황 판단 AI 모델	AI	혜진	완료
추가	위험감지	AI	중	위험상황 판단 AI - 과적합 해결 (위험발화 구분문제)	AI	혜진	완료
추가	위험감지	AI	상	위험상황 판단 AI - 과적합 해결 (정상발화 구분문제)	AI	혜진	완료
1차	위험감지	서버	상	위험상황 판단 AI-서버배포	서버	혜진	완료
1차	위험감지	서버	상	위험 판단 AI-API 개발	API	혜진	완료

[위험감지화면 개발해야 될 기능 목록]

06 1차 심사 이후 통합 테스트

Pass, Fail로 분류하며 통합 테스트 진행 (완료)

회면	분류	항목	11.20 (수)	11.22 (금)	11.23 (토)
				Pass	Pass
	단말기	단말기 전원 ON	X	O	O
	단말기	단말기 코드 자동 실행	X	X	O
<외출 감지 - 메인화면>					
	단말기	단말기를 착용하고 밖으로 외출	X	O	O
	서버	단말기 --> 서버로 '외출 상태 업데이트' 확인	O	O	O
	App	외출 감지 알림 수신	O	O	O
	단말기	단말기에서 자동 녹음 실행 + 단말기 발화	O	O	O
		└ 소리 잘 끊기는지 확인	X	O	O
		└ 녹음 파일 확인	O	O	O
	App	메인화면에서 '외출이 감지되었습니다' 확인	O	O	O
<귀가 감지 - 메인화면>					
	단말기	단말기를 착용하고 집/실내로 들어가기	O	O	O
	서버	단말기 --> 서버로 '귀가 상태 업데이트' 확인	O	O	O
	App	귀가 감지 알림 수신	O	O	O
<정상 감지 - 단말기에서 절실향으로 발화>					
	단말기	단말기 --> 서버로 녹음 파일 업로드 확인	O	O	O
	서버	Lambda 호출 : 녹음 파일에서 메타데이터 추출 & 엔드포인트 호출	O	O	O
	서버	STT 결과 / AI 결과 도출 확인	O	O	O
	AI	AI 판단 결과 정확도 체크	X	X	O
	서버	DB 값 확인	O	O	O
	App	위험 알림 안 가는 거 확인	X	X	O
<위험 감지 - 단말기에서 위험 상황으로 발화>					
	단말기	단말기 --> 서버로 녹음 파일 업로드 확인	O	O	O

회면	분류	항목	11.20 (수)	11.22 (금)	11.23 (토)
				Pass	Pass
	App	위험 알림 안 가는 거 확인	X	X	O
<위험 감지 - 단말기에서 위험 상황으로 발화>					
	단말기	단말기 --> 서버로 녹음 파일 업로드 확인	O	O	O
	서버	Lambda 호출 : 녹음 파일에서 메타데이터 추출 & 엔드포인트 호출	O	O	O
	서버	STT 결과 / AI 결과 도출 확인	O	O	O
	AI	AI 판단 결과 정확도 체크	O	X	O
<위험 상황 O>					
	App	위험 알림 수신 확인	O	O	O
<알림 읽음 X>					
	서버	알림 안 읽음 시 --> 알림 재전송 5분 확인 (5분 * 3)	O	O	O
<알림 읽음 O>					
	서버	알림 읽음 처리 확인	O	O	O
	App	알림 목록 확인	O	X	O
	App	알림 목록 --> 위험 감지 화면 내의 데이터 배치 확인 (타임스탬프/위치/녹음)	X	O	O
<유저 - 실제 위험 상태 X>					
		유저판단 업데이트 엔드포인트 호출되는지 확인	O	O	O
		기록으로 저장 안 되는 거 확인	O	O	O
<유저 - 실제 위험 상태 O>					
	App	위험 종료 버튼 누르기	O	O	O
	서버	위험 상황 종료 --> 기록 저장 엔드포인트 호출되는지 확인	O	O	O
	서버	DB 기록 저장되는지 확인	O	O	O
<GPS화면>					
	App	메인화면 들어가서도 잘 실행되는지 확인	O	O	O
	App	위험감지화면 - 현재위치 확인 화면 들어가기	O	O	O

1차 심사 피드백 반영사항

- 피드백 내용 정리
- 현재 한계점
- 개선사항
- 새로 개발된 사항
- 추후 연구과제

01 피드백 내용 정리

〈벤치마킹〉



- 기존 제품과의 차별성을 명확히 제시해줄 것

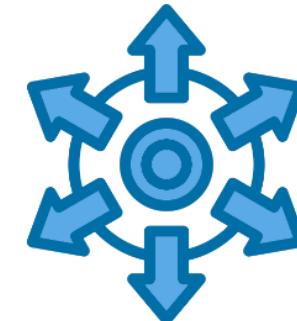


〈하드웨어〉



- 몸에서 떼어놓는 상황 대비
- 배터리 작동 시간?
- 소형화?
- 네트워크 연결 방식?
- 구현 불가한 부분 제시

〈확장 범위〉



- 치매환자 외에 헤아Ring을 적용할 수 있는 대상이 또 있는가?

02 현재 한계점 (반영 불가 요소)

● 외출 시 단말기를 몸에서 떼어놓는 상황 대비

문제점

- 환자가 외출 시 단말기를 풀어버리면 어떡할 것인가?

개선 방안 :

- 벨트 및 단말기에 센서를 달아 환자의 몸에서 떨어지면 보호자에게 “단말기 부착 해제 알림”을 보냄



- 압력 센서 : 벨트가 몸에 고정되어 있는지 감지
- 가속도 센서 : 벨트가 갑작스럽게 빠지는 비정상적 움직임 감지

● GPS 센서의 한계점

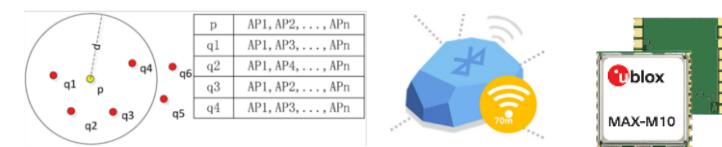
GPS 센서 성능 문제 :

- 예산 초과 ⇒ 저가형 GPS 센서 사용
- 흐린 날씨, 실내 환경에서 위치 감지가 어려움



개선 방안 :

- 보조 위치 추적 기술 도입
 - 비콘 : 요양병원 등 실내에 비콘을 설치해 위치 데이터를 수집
 - WiFi 기반 : AP 통해 실내에서도 위치를 보조적으로 추적
- GNSS 모듈 추가 (러시아, 중국 위성 등을 지원)



하지만 현재 단말기에 대한 추가 개발은 비용과 시간 상의 한계가 존재

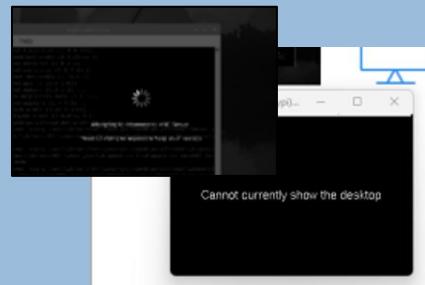
02 현재 한계점 (반영 불가 요소)

● 단말기 소형화

〈재료 구입 + 조립〉



〈문제점 발생〉



- VNC 강제 종료
- dpkg 설치 에러 문제
- VS Code 설치 불가능
- 패키지 설치 불가능

〈이슈 공유 + 해결 기간〉

이를 해결하기 위해 시도한 방법:

1. 네트워크 재설정: 네트워크 연결 상태를 재구성함.
2. KeepAlive 파라미터 설정: SSH 연결 유지를 위해 파라미터 값을 조정
3. 불필요한 서비스 비활성화: sudo systemctl disable 명령어를 실행
4. 충전 모듈 분리 후 재개방: 충전 모듈을 분리하고 개별로 재개
5. 겉면 판리 비활성화: 라즈베리파이 제로의 겉면 판리 기능을 비활성화하였음.
6. OS 재설치 후 재부팅: Lite 32Bit 및 64Bit 버전, 그리고 Raspberry OS 32Bit 및 64Bit, Legacy 32Bit를 모두 적용하여 재부팅
7. SSH 환경에서 PuTTY를 사용하여 dpkg 설치 시도

```
# 설정 후 ssh 재시작  
sudo systemctl restart ssh  
  
결과 => sudo dpkg --configure -a 명령어 입력 후 끊기는건.. 처음보단 조금 늦춰진건 제각원  
하지만 끊기는건 똑같다 + 끊기는 느낌이 불규칙해짐. 언제 끊길지 예상이 안간다  
  
3. 소프트웨어 최적화  
a. 불필요한 서비스 비활성화 sudo systemctl disable 명령어 사용  
=> 달라진게 전혀 없다..  
  
결과 코드:  
Removed /etc/systemd/system/multi-user.target.wants/ssh.service.
```

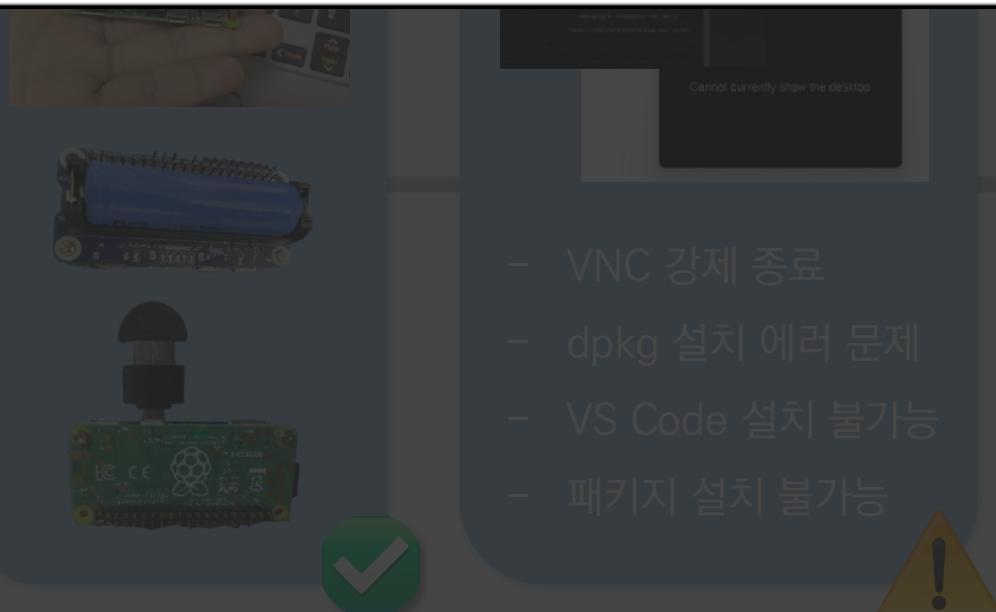
- 팀원들과 이슈 공유
- 이슈 해결을 위해 노력
해보는 시간 가짐



〈이슈 해결을 위해 계획 수립 + 모든 방법 시도〉

이를 해결하기 위해 시도한 방법:

1. 네트워크 재설정: 네트워크 연결 상태를 재구성함.
2. KeepAlive 파라미터 설정: SSH 연결 유지를 위해 파라미터 값을 조정
3. 불필요한 서비스 비활성화: sudo systemctl disable 명령어를 실행
4. 충전 모듈 분해 후 재개발: 충전 모듈을 분해하고 개발을 재개
5. 전력 관리 비활성화: 라즈베리파이 제로의 전력 관리 기능을 비활성화하였음.
6. OS 재설치 후 재부팅: Lite 32Bit 및 64Bit 버전, 그리고 Raspberry OS 32Bit 및 64Bit, Legacy 32Bit를 모두 적용하여 재부팅
7. SSH 환경에서 PuTTY를 사용하여 dpkg 설치 시도



- VNC 강제 종료
- dpkg 설치 에러 문제
- VS Code 설치 불가능
- 패키지 설치 불가능

```
# 설정 후 ssh 재시작  
sudo systemctl restart ssh
```

결과 ⇒ sudo dpkg --configure -a 명령어 입력 후 끊기는건.. 처음보단 조금 늦춰진건 체감됨
하지만 끊기는건 똑같다 + 끊기는 느낌이 불규칙해짐. 언제 끊길지 예상이 안간다

3. 소프트웨어 최적화

- a. 불필요한 서비스 비활성화 sudo systemctl disable 명령어 사용
⇒ 달라진게 전혀 없다 ..

결과 코드:

```
Removed /etc/systemd/system/multi-user.target.wants/ssh.service.
```

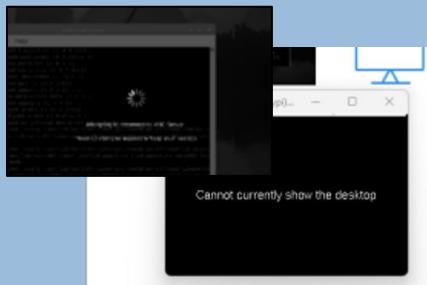
02 현재 한계점 (반영 불가 요소)

● 단말기 소형화

〈재료 구입 + 조립〉



〈문제점 발생〉



- VNC 강제 종료
- dpkg 설치 에러 문제
- VS Code 설치 불가능
- 패키지 설치 불가능

〈이슈 공유 + 해결 기간〉

이를 해결하기 위해 시도한 방법:

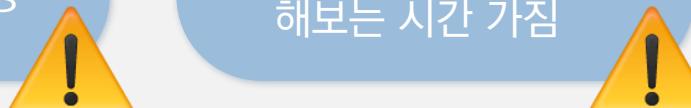
1. 네트워크 재설정: 네트워크 연결 상태를 재구성함.
2. KeepAlive 파라미터 설정: SSH 연결 유지를 위해 파라미터 값을 조정
3. 불필요한 서비스 비활성화: sudo systemctl disable 명령어로 실행
4. 충전 모듈 분해 후 재개발: 충전 모듈을 분해하고 개발을 재개
5. 전력 관리 비활성화: 라즈베리파이 제로의 전력 관리 기능을 비활성화하였음.
6. OS 재설치 후 재부팅: Lite 32Bit 및 64Bit 버전, 그리고 Raspberry OS 32Bit 및 64Bit, Legacy 32Bit를 모두 적용하여 재부팅
7. SSH 환경에서 PuTTY를 사용하여 dpkg 설치 시도

```
# 설정 후 ssh 재시작
sudo systemctl restart ssh

결과 => sudo dpkg -configure -a 명령어 입력 후 끊기는건.. 처음보단 조금 늦춰진건 제각될
하지만 끊기는건 똑같다 + 끊기는 느낌이 불규칙해짐. 언제 끊길지 예상이 안간다

3. 소프트웨어 최적화
a. 불필요한 서비스 비활성화 sudo systemctl disable 명령어 사용
   => 딜레이가 전혀 없다..
결과 코드:
Removed /etc/systemd/system/multi-user.target.wants/ssh.service.
```

- 팀원들과 이슈 공유
- 이슈 해결을 위해 노력
- 해보는 시간 가짐



〈파이4 vs 제로2W 선택 회의 진행〉

라즈베리파이제로2W는 저전력 디바이스로 경량 작업에 적합하지만, 복잡한 작업에서는 성능 상이 제약이 발생하는 것으로 확인

라즈베리파이4는 고성능 CPU와 향상된 메모리 용량은 음성 녹음, 실시간 데이터 처리 및 서비스 통신과 같은 작업에서 제한 없이 개발 가능

라즈베리파이 4	라즈베리파이 제로 2W
성능	제한적 - 핵심적인 기능을 실행 하기 위한 패키지를 설치할 때 문제가 생김 - vs Code 설치 X(원격으로 사용해야함) - 웹서버 인식 부분에서 기본적인 기능 작동하는 것 확인.. 마이크 사용 가능(필요한 패키지 사용 불가) - 저전력 디바이스로 경량 작업에 적합하지만, 복잡한 작업에서는 성능상의 제약이 발생
남은 기능	- 기존에 개발한 기능이 있어서 남은 GPS 기능이 많이 없었음 - 패키지 설치 관련된 오류 해결 - 완성된 기능 코드 옮기기 - 패키지 오류에 따른 대체 패키지 사용 시 완성된 기능 수용 - GPS 기능 개발 - 서버 전송 & 통신 기능 개발
주정 개발 소요 시간	적음(통합 test까지 합계해서 10주까지 진행 가능)
선택 결론	○

〈파이4와 제로2W 선택 회의 진행〉

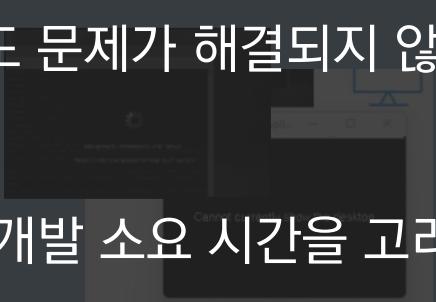
라즈베리파이제로2W는 저전력 디바이스로 경량 작업에 적합하지만, 복잡한 작업에서는 성능상의 제약이 발생하는 것으로 확인

라즈베리파이4는 고성능 CPU와 향상된 메모리 용량은 음성 녹음, 실시간 데이터 처리 및 서버 통신과 같은 작업에서 제한 없이 개발 가능

〈재료 구입 + 조립〉

- 해결 기간을 가졌음에도 문제가 해결되지 않았음
⇒ 긴급 회의 진행
- 성능, 남은 기능, 추정 개발 소요 시간을 고려하여
⇒ 라즈베리 파이4로 선택
- 결정 사항을 문서화하여, 결정 과정을 명료하게 기록

〈문제점 발생〉



〈이슈 공유 +

이 문제 해결하기 위해 시도한 방법:

1. 네트워크 재설정: 네트워크 연결 상태를 재구성
2. KeepAlive 파라미터 설정: SSH 연결 유지
3. 불필요한 서비스 비활성화: sudo systemctl disable
4. 주진 모듈 분체 후 재개방: 충전 모듈을 분체
5. 진피 전리 비활성화: 라즈베리파이 제로의 진피
6. OS 재설치 후 재부팅: Lite 32Bit 및 64Bit 버전 Legacy 32Bit를 모두 적용하여 재부팅
7. SSH 환경에서 PuTTY를 사용하여 dpkg 설치

설정 후 ssh 재시작
sudo systemctl restart ssh
결과 ⇒ sudo dpkg -configure-a 명령어 입력
하지만 끊기는건 똑같다 + 끊기는 느낌이 불규칙

3. 소프트웨어 최적화
a. 불필요한 서비스 비활성화 sudo systemctl disable

⇒ 딜라인에게 전해 있다 ..
결과 코드:
removed /etc/systemd/system/multi-user.target.wants/vncserver-x11.service

	라즈베리 파이 4	라즈베리파이 제로 2w
성능	<p>원활한 개발 환경을 제공</p> <p>- 보다 복잡한 연산과 멀티태스킹을 원활히 처리할 수 있다</p> <p>- 고성능 CPU와 향상된 메모리 용량은 음성 녹음, 실시간 데이터 처리 및 서버 통신과 같은 작업에서 제한 없이 개발 가능</p>	<p>제한적</p> <ul style="list-style-type: none">- 핵심적인 기능을 실행하기 위한 패키지들을 설치할 때 문제가 생김- vs Code 설치 X(원격으로 사용해야함)- 음성 인식 부분에서 기본적인 기능 작동하는 것 확인: 마이크 사용 가능(필요한 패키지 사용 불가)- 저전력 디바이스로 경량 작업에 적합하지만, 복잡한 작업에서는 성능상의 제약이 발생
남은 기능	<p>기존에 개발한 기능이 있어서 남은 GPS 기능이 많이 없었음</p>	<ul style="list-style-type: none">- 기존 환경 셋팅- 패키지 설치 관련된 오류 해결- 완성된 기능 코드 옮기기- 패키지 오류에 따른 대체 패키지 사용 시 완성된 기능 수정- GPS 기능 개발- 서버 전송 & 통신 기능 개발
추정 개발 소요 시간	<p>적음 (통합 test까지 합해서 10월까지 진행 가능)</p>	<p>많음 (적어도 2개월 추정 + 통합 test 기간 별도 필요)</p>
선택 결론	<p>○</p>	<p>!</p>

03 개선사항

● 보드 조립 완료 + 배터리 테스트



- 배터리 용량 : 18650
- 리튬 배터리2개: 4000mAh (4Ah)

- 충전 모듈 및 기타 부품 조립 완료
- 배터리 테스트 : 기능 돌아가는 상태에서 3시간 실행 가능

⇒ 현재 하드웨어 최적화 X, 최적화를 통해 더 장시간 실행될 수 있을 것으로 기대됨

● 단말기 네트워크 연결 방식

기존 배회감지기 네트워크 방식

- **통신사와 연계된** 공공 요금제를 활용
- 저렴한 비용으로 통신 서비스를 제공



프로토타입 단계 :

- 현재는 개발 담당자의 **모바일 핫스팟/WiFi** 이용해 연결

실용화 시 :

- 공공기관 지원 요금제 환경을 가정하여, **통신사와 협력하는** 방식으로 서비스를 최적화할 계획
 - 2024년 「치매정책 사업안내」

'2024년 「치매정책 사업안내」 지침 개정' 치매환자 지원을 확대합니다

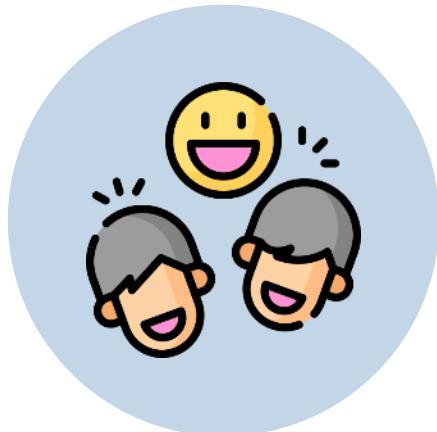
작성일 2024-01-31 18:55 | 조회수 12,542 | 담당자 장홍준 | 담당부서 노인건강과

'2024년 「치매정책 사업안내」 지침 개정' 치매환자 지원을 확대합니다

- 맞춤형 사례관리 서비스 전국 실시, 치매치료관리비 지원 확대 권고,
장애인 위한 치매검사 절차 마련 등 치매환자 지원 강화 -

03 개선사항

● 확장 가능성 (기대효과)



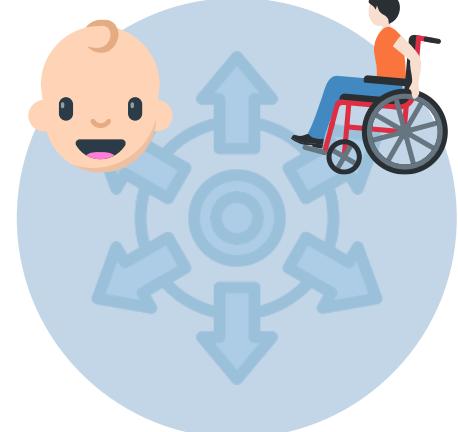
치매 돌봄 가족의
스트레스 완화 및
삶의 질 개선



실종 감소 효과 및
성범죄 등
각종 위험 상황 예방



실종 예방 통한
인력 및 자원 소모를
줄여 사회적 비용 절감



실종아동등*을
대상으로 확장가능

* 실종아동등: 납치, 유인, 유기, 사고, 가출, 길을 잃는 등의 사유로 인하여 보호자로부터 이탈된 당시 18세 미만인 아동, 지적·자폐성·정신장애인, 치매 환자의 통칭

03 개선사항

● 확장 가능성 (기대효과)

⇒ 데이터가 쌓일수록 제공 가능한 **다양한 기능**

#시각화 제공

- 주로 배회하는 시각
- 위험 감지가 많이 된 지역
- 워드 클라우딩
(위험 감지 문장)



#다중 분류 모델 변환

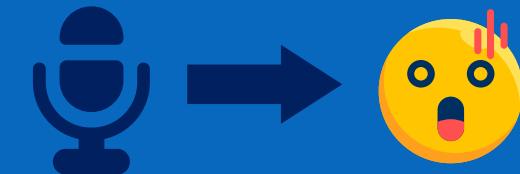
주변의 도움 요청, 길 잃음, 사고 위험, 무언가를 찾음

신고 요청, 응급 의료, 도움 요청

갇힘, 화재, 강력범죄(폭력/절도/강도), 재해, 낙상,
강제추행(성범죄)

어떤 위험 상황인지 분류하여 제공

#감성 분석 모델 추가

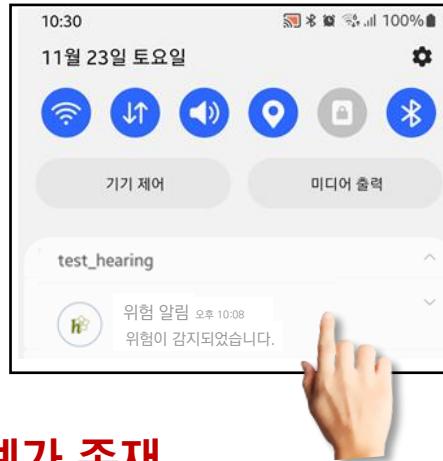


수집된 녹음 파일로,
치매환자의 감정까지 파악

04 새로 개발된 사항

● 알림 목록 화면

- 기존 [위험감지 화면] 접속 로직:
 - 알림 탭의 알림을 선택
 - 알림에 포함된 위험상황데이터 ID로 데이터 조회
 - 조회된 데이터가 화면에 출력



→ 알림 탭에서 사라진 알림은 확인하지 못하는 한계가 존재

- [알림 목록 화면] 개발하여 해결

목적: 알림 탭에서 확인하지 못한 이전 알림 내역 확인

- 실수로 알림 탭에서 삭제한 알림 확인 가능
- 이전에 수신한 알림(위험상황) 중에 읽지 않은 내역 확인하고 검토 가능



04 새로 개발된 사항

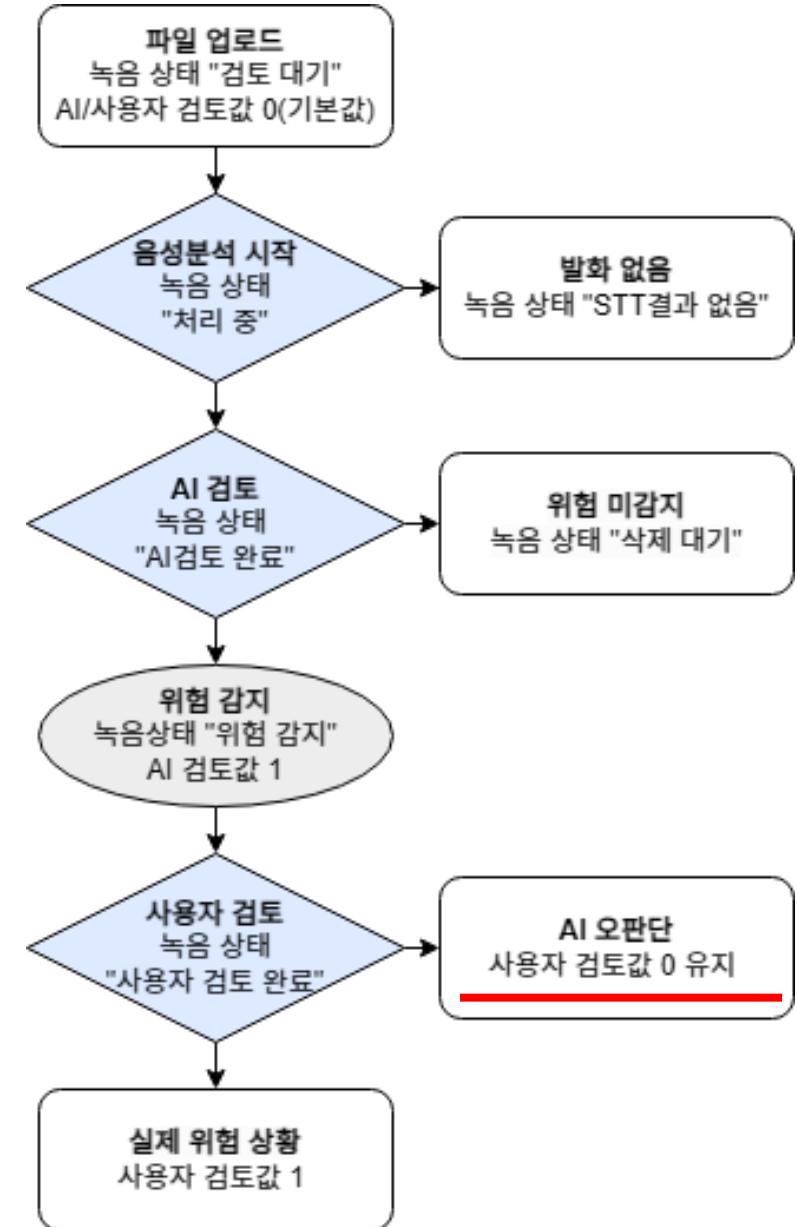
● AI 검토, 사용자 검토 확인 여부 추가

목적: AI의 오판단 구분

- 정상 상황 ⇒ 위험 상황으로 잘못 판단한 경우를 데이터화하기 위함
- recording 테이블에 ai_review / user_review 필드 추가

record_id	ai_review	user_review	recording_status	text
49	1	0	사용자 검토 완료	["굵어야 돼."]
50	1	1	사용자 검토 완료	["도와주세요.", "어디야..."]
63	0	0	STT 결과 없음	NULL

- 두 속성과 recording_status 속성값을 분석하여, 이후 AI 모델 학습에 활용 가능



04 새로 개발된 사항

● WebSocket 세션 관리의 서비스화

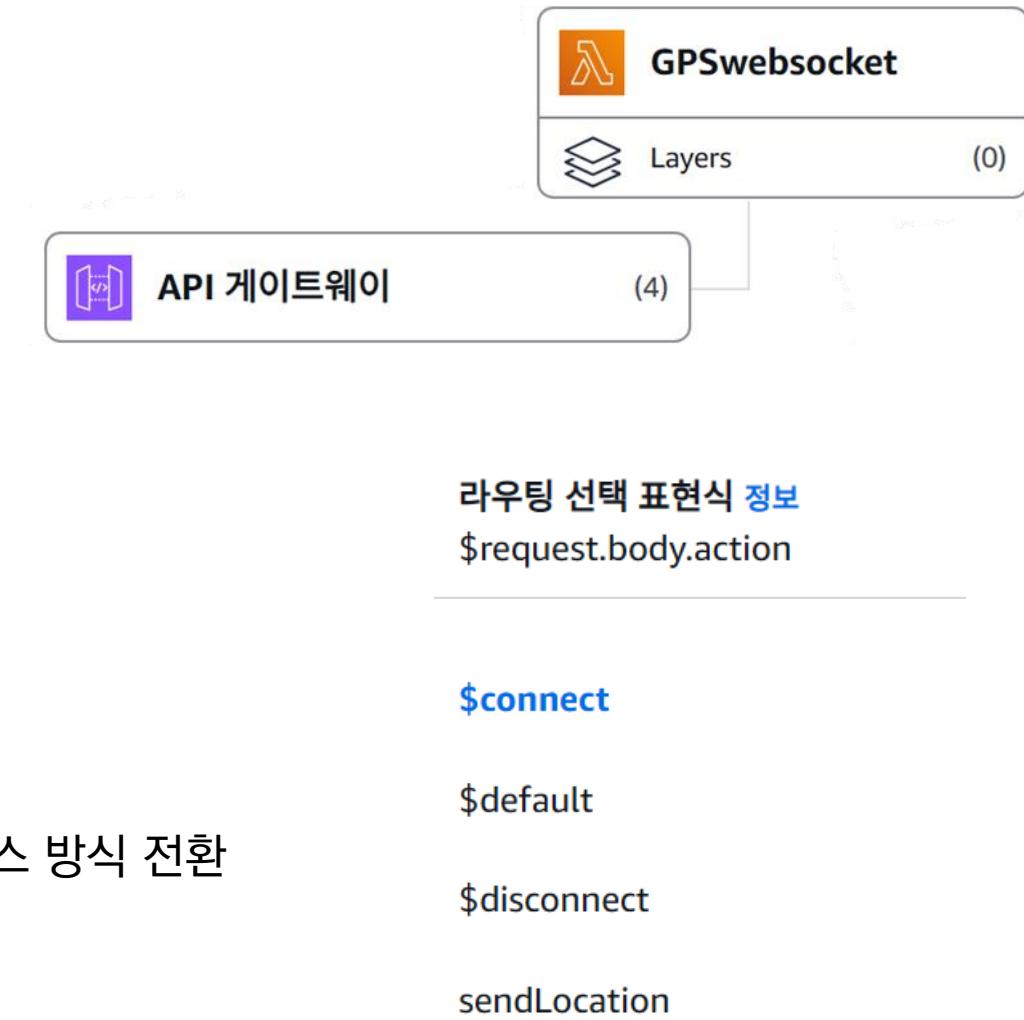
- 예산 한계 : 무료 티어 EC2 이용
→ 서버 과부하 발생 시, 실시간 위치 확인 기능 성능 문제 우려

- AWS Lambda 도입 (서비스)

목적: EC2 서버 부하 분산, 인프라 관리 단순화

- 기존 WebSocket API ⇒ AWS Gateway 변경
- 세션 및 메시지 처리 : 웹서버 담당 ⇒ Lambda 도입하여 서비스 방식 전환
- 서버 부하 분산, 서버 성능 + 비용 효율성 최적화

⇒ 세션 관리의 안정성 확보, 서버 비용 감소, 인프라 관리 단순화



05 추후 연구 과제

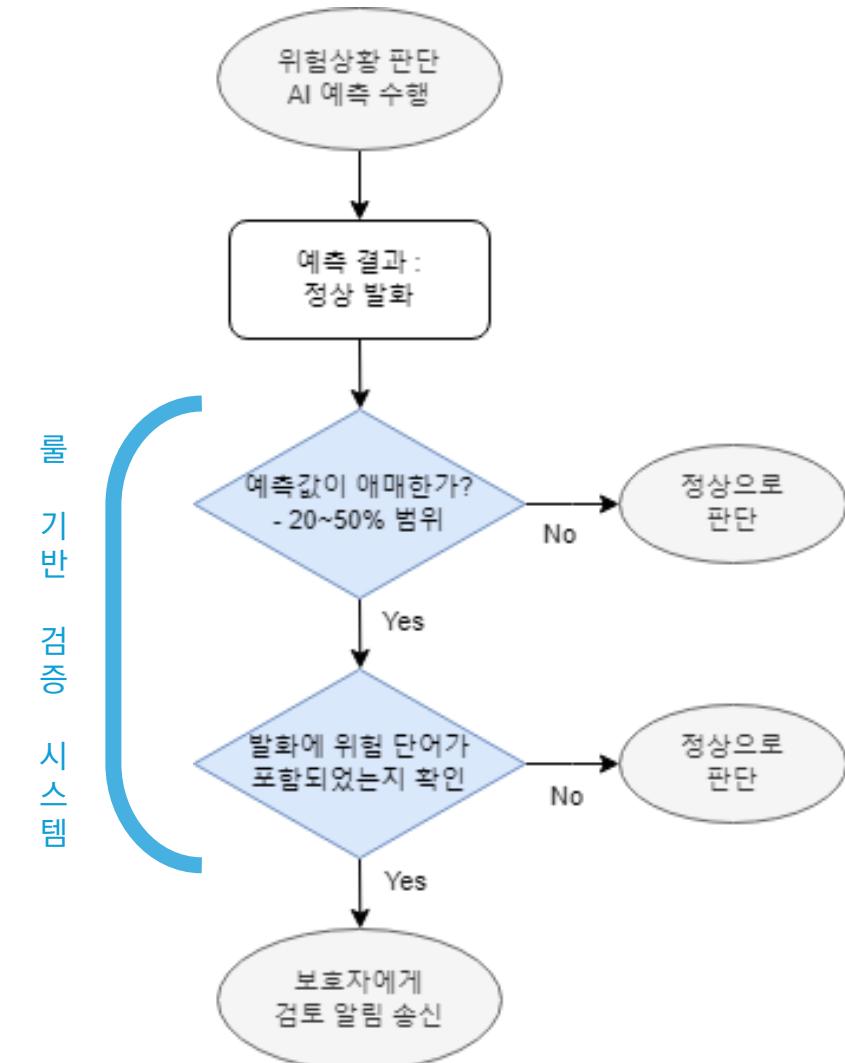
● ‘위험 단어’ 감지 로직 추가

- 위험 상황을 오판할 가능성 有
 - 잘못된 키워드를 인식하는 경우
 - 실제 위험 상황을 정상 발화로 오판하여 위험 상황을 놓치는 경우
- 로직 추가하여 해결

목적 : 위험상황 판단 AI 모델의 **판단 오류 최소화**

- AI 모델의 판단 확률이 애매한 범위 (20 ~ 50%)인 경우
- 를 기반 검증 시스템이 ‘위험 단어’가 포함된 발화를 보호자에게 알림
- 위험 단어 예시 :
('아파', '살려줘', '도움이 필요', '도와줘', '길을 잃었나봐', … 등)

⇒ AI 모델의 판단 오류를 보완하여 환자의 안전을 최대한으로 보장



05 추후 연구 과제

● 알림 채널 분리

목적 : 각 알림의 중요도 구분

- 각 알림은 **중요도에 따른 차별화된 처리 필요**
 - 외출/귀가 감지, 위험 감지 알림, 이후 추가될 위험단어 감지 알림
 - 사용자가 알림 중요도를 개별적으로 설정하는 기능 고려
- 활용 예시
 - 일반적으로 외출이 불가능한 **중증 환자**의 외출 알림 긴급
 - 혼자 거동 가능한 **경증 환자**의 외출 알림 보통

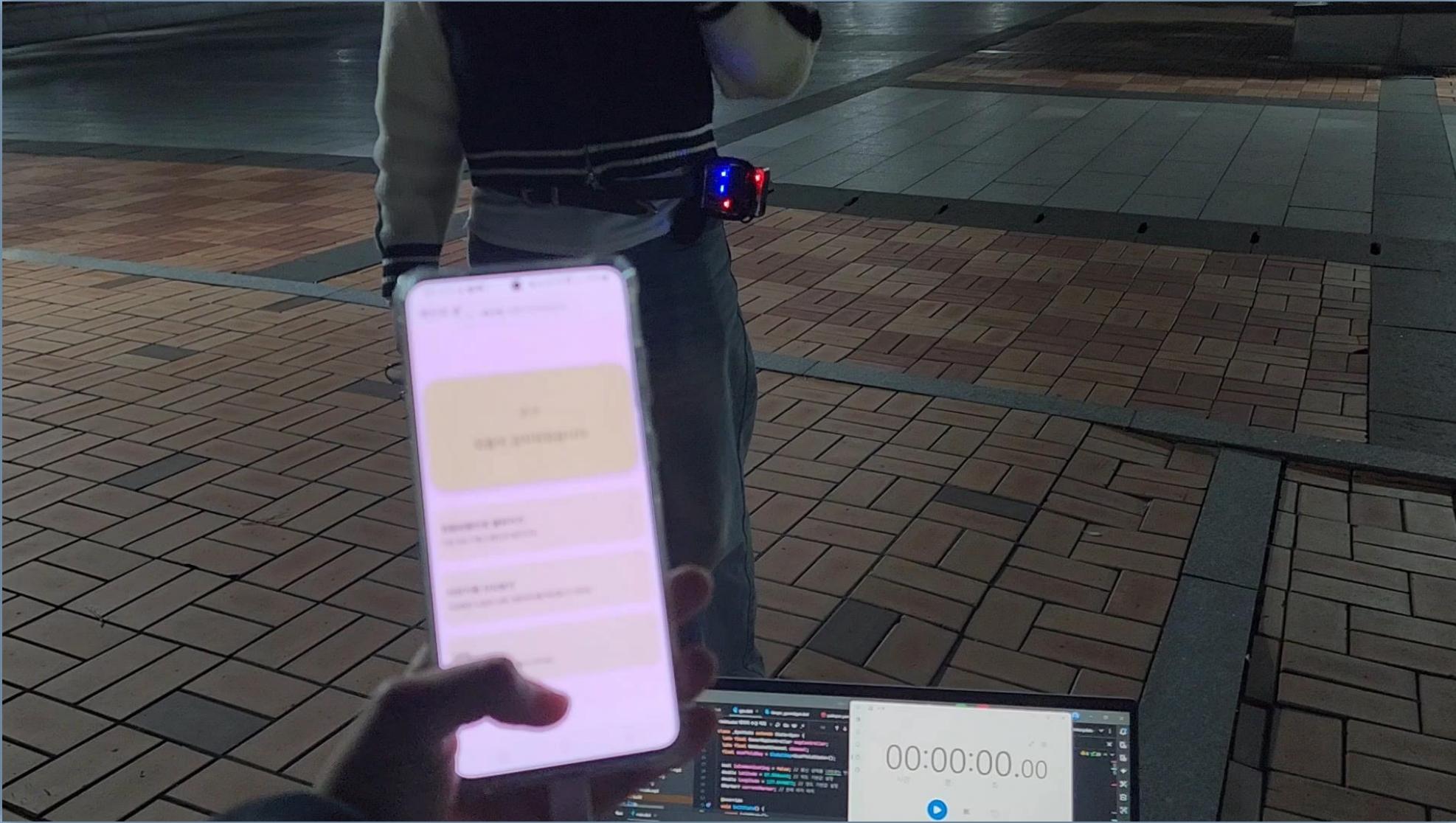
중요도	설명
긴급	알림음이 울리며 헤드업 알림으로 표시
높음	알림음 울림
보통	무음
낮음	무음, 상태표시줄에 표시되지 않음
없음	무음, 상태표시줄이나 화면 창에 표시되지 않음

〈FCM의 알림 중요도 수준〉

⇒ 사용자의 효율적인 알림 확인 유도, 사용자 맞춤형 알림 관리

실제 시연

01 GPS 화면



02 라이브 데모



App 실행 + 화면 공유

〈보호자〉

강의실 내



단말기 착용 & 발화 + 촬영 공유

〈치매환자〉

외부로 이동

감사합니다

수제지능팀
- 황혜진, 전희주, 오지현, 옥지원

