

# Smart home Solution for Dogs Left Alone at Home

## DogAlone: Your Dog's Caretaker

2024 Hanyang University Software Engineering Project  
Team APET: 김서연 변준형 이동률 전채연 최준형

A

# 목차

## 01 개발 배경

문제 상황  
해결 방안

## 02 앱 소개

전체 AI 텍쳐  
실시간 반려견 소리 분석  
스마트홈 솔루션 추천  
울음소리 리포트  
LG 스마트홈 기기 제어  
반려견 식사 관리

## 03 시연 영상



## 04 AI 텍쳐 구조

AI 학습 과정  
데이터 전송 과정  
전체 연동

## 05 결론

느낀 점  
향후 계획

## 집에 혼자 남은 반려견을 위한 대책 부족

현대 가정에서 반려견은 단순한 애완동물을 넘어 가족의 중요한 구성원으로 자리 잡았습니다.

그러나 많은 반려견이 집에 홀로 남는 시간이 길어지면서 적절한 돌봄을 받지 못하는 경우가 많습니다. 혼자 있는 동안 반려견은 외로움을 느끼며, 심한 경우 분리불안, 과도한 짖음, 배변 실수 등의 문제를 겪을 수 있습니다.

집을 비운 동안 외로워할 반려견을 생각하며 죄책감을 느끼는 보호자들이 많지만, 이를 효과적으로 해결할 방법은 아직 부족한 실정입니다.

302만  
마리

2022년 우리나라 반려견 수

75  
%

반려견을 집에 혼자 남겨두는  
경우가 있다고 응답한 비율

5 시간 22 분

반려견이 집에 혼자 있는 시간 평균

# 반려견을 위한 스마트홈 솔루션

이러한 문제를 해결하고자, 우리 팀은 반려견의 울음소리를 AI로 분석하여 감정 상태를 파악하고, 적절한 스마트홈 솔루션을 제공하는 어플리케이션을 구상하였습니다.

- 논문 'Communication in Dogs'에 따르면, 개들의 언어는 주파수 폭과 특정 음향 규칙에 따라 맥락적 정보를 제공
- 개의 울음소리를 이미지 형태로 학습시켜 개가 무슨 감정을 표현하고 있는지 파악하는 AI를 개발
- 논문과 수의사의 처방을 참고하여 각 감정에 필요한 스마트홈 솔루션을 추천

# 주파수

# 소리 길이

# 빈도 수

## DogAlone의 주요 기능



실시간 반려견 소리 분석



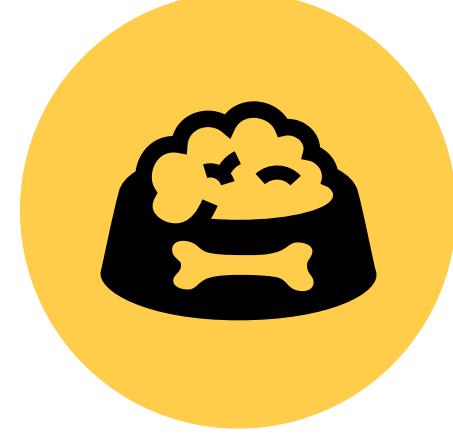
스마트홈 솔루션 추천



울음소리 리포트

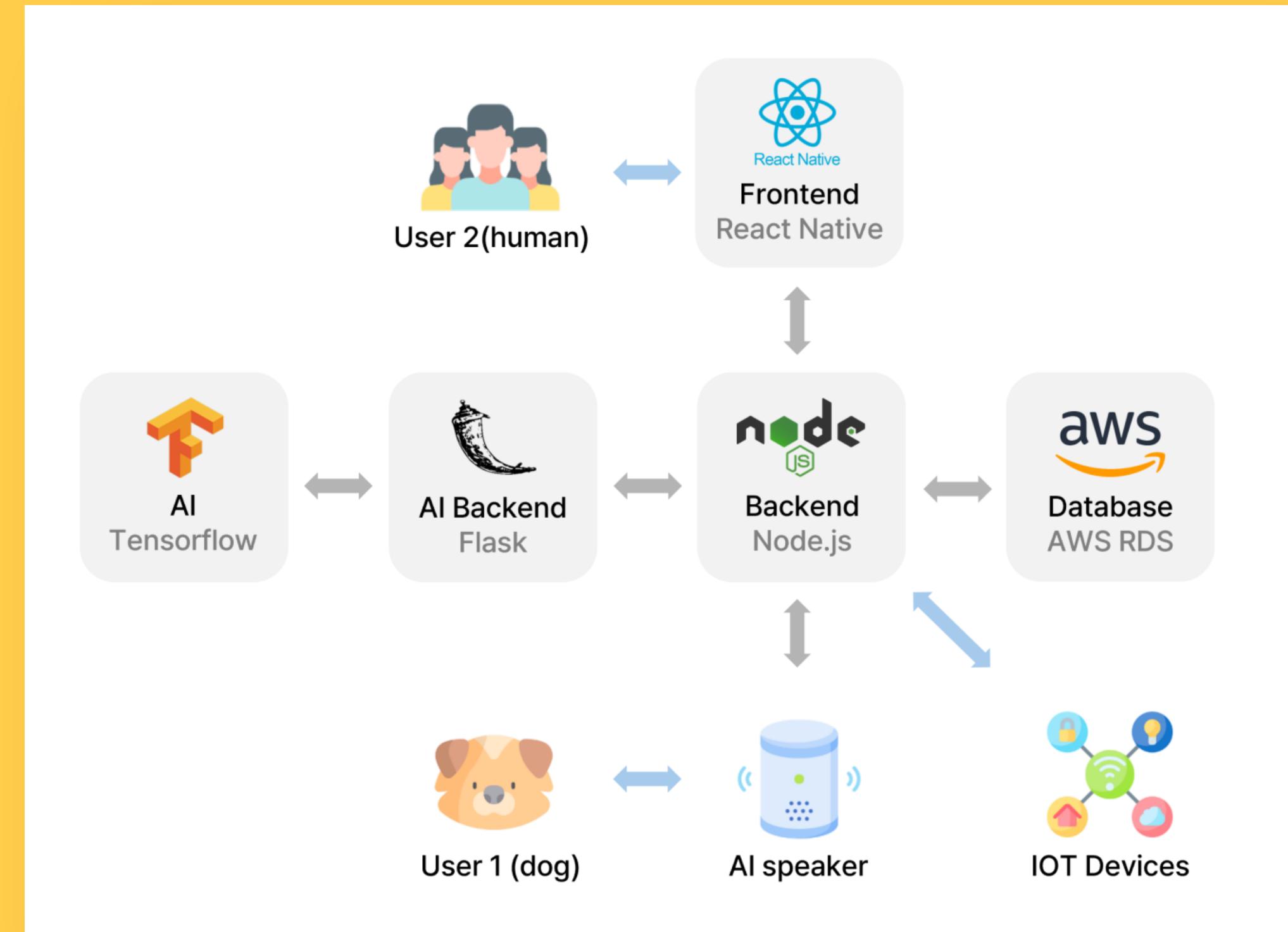


LG 스마트홈 기기 제어



반려견 식사 관리

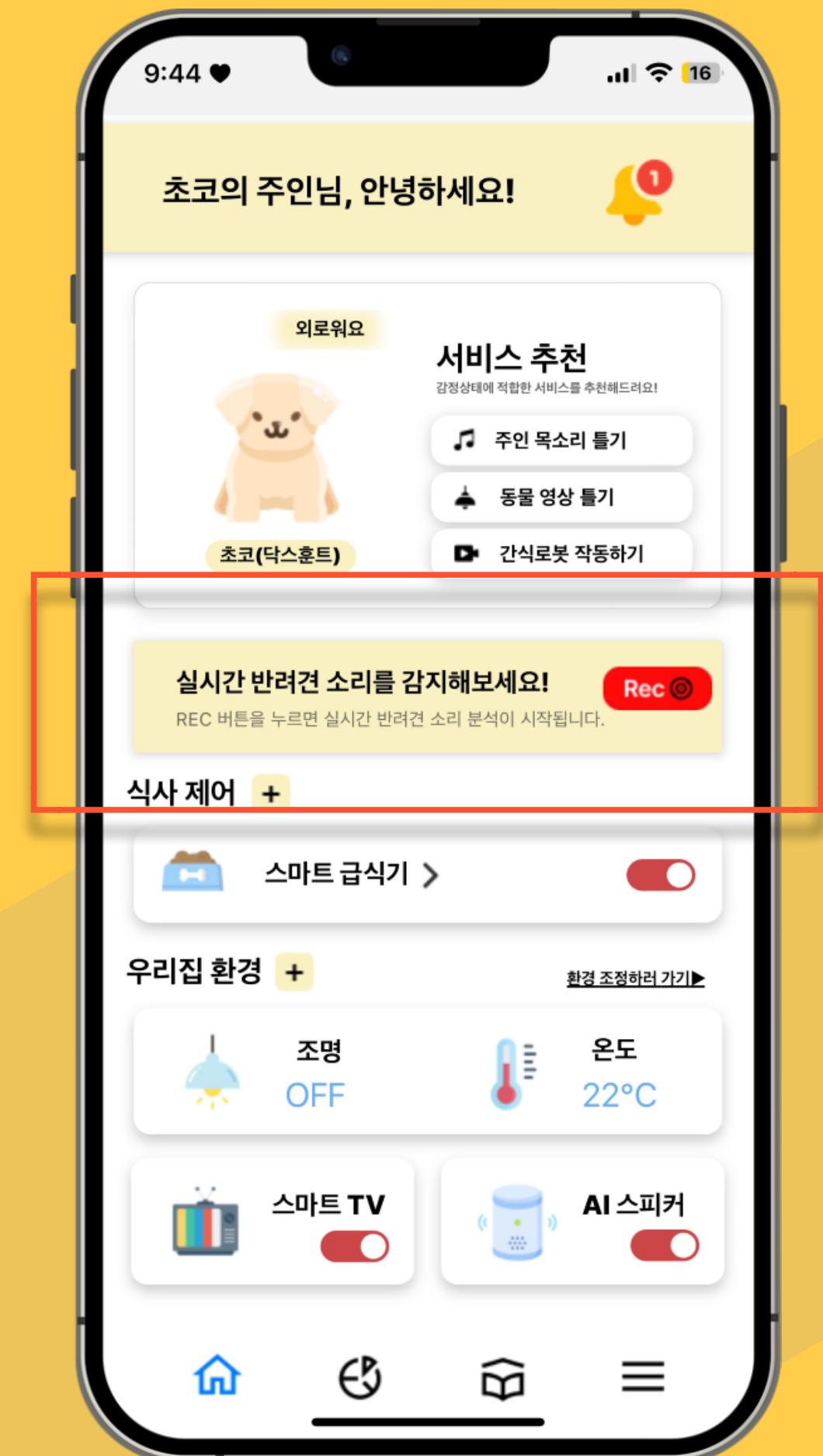
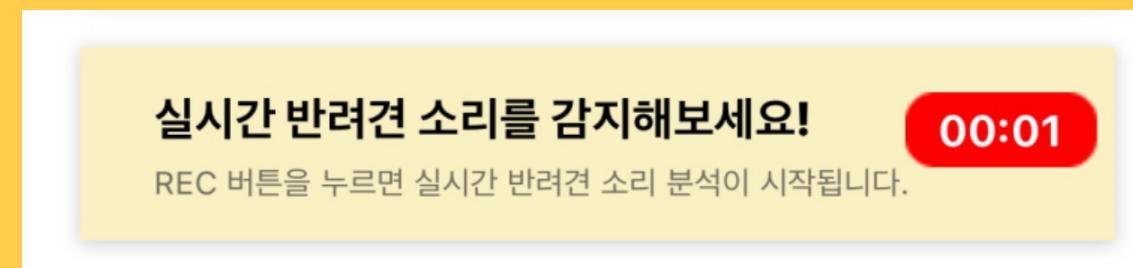
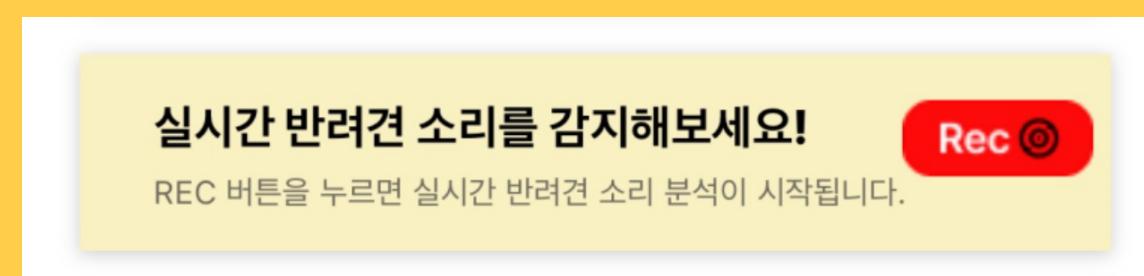
# 전체 애플리케이션 구조



# 실시간 반려견 소리 분석

## AI의 실시간 감정분석

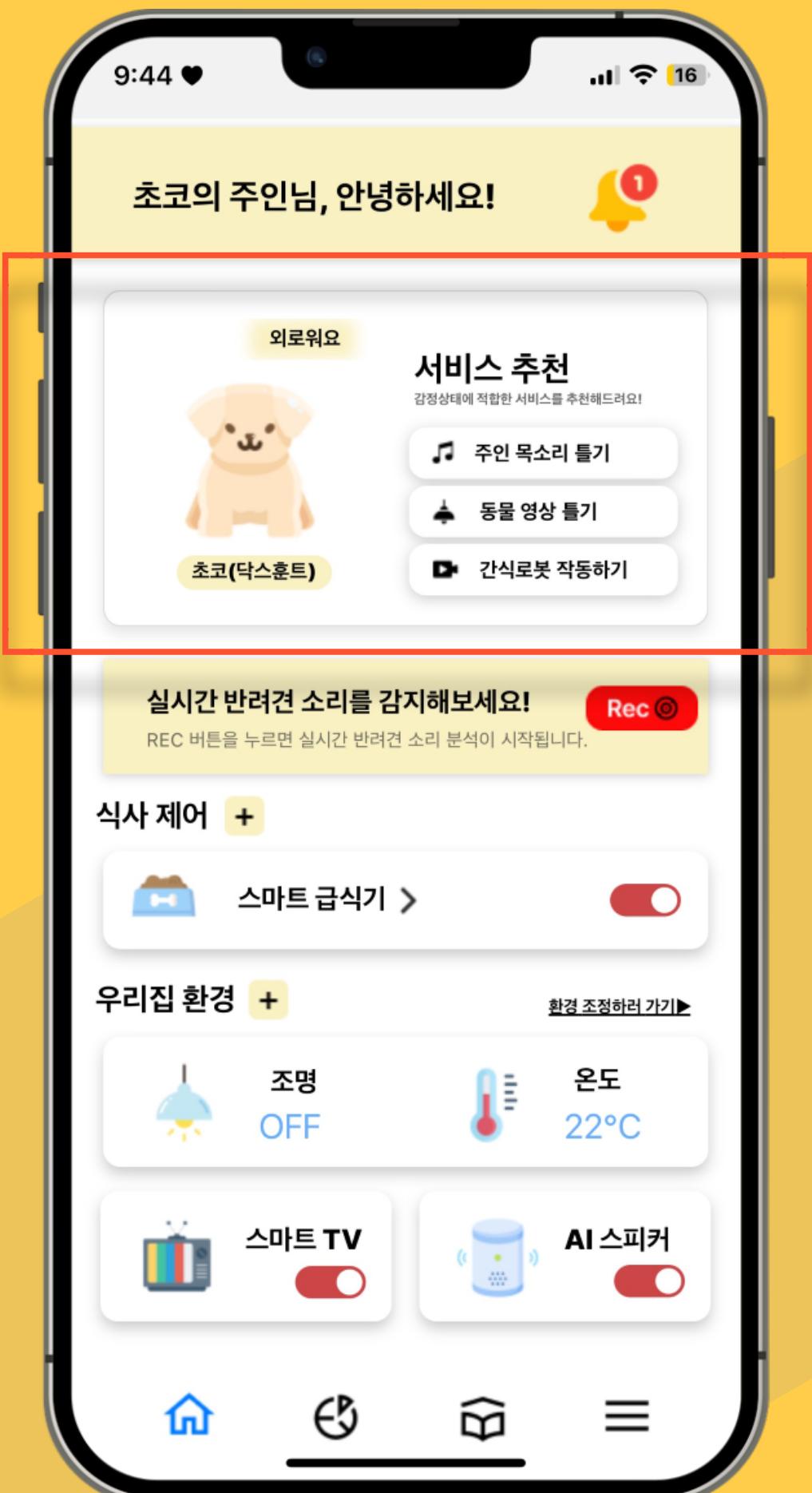
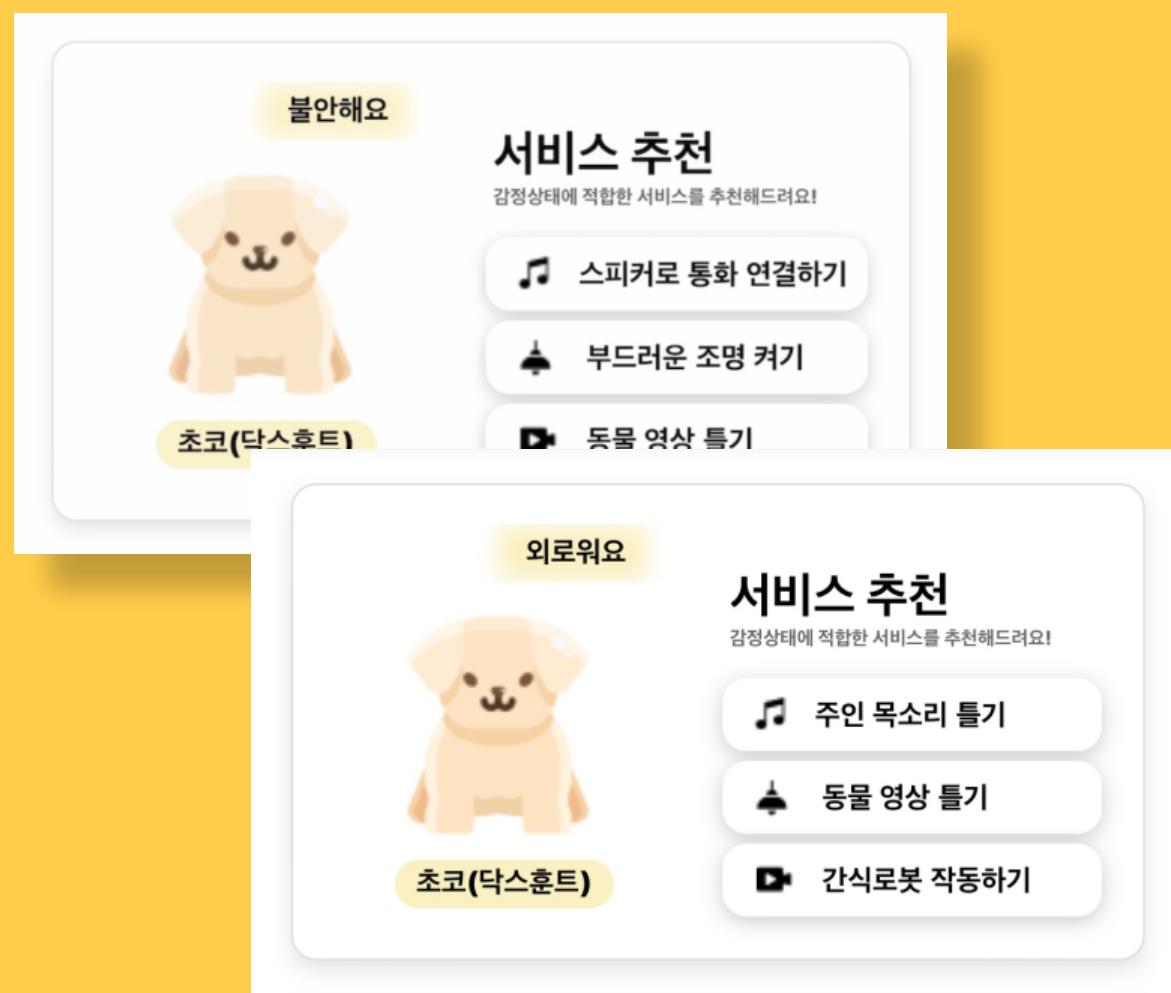
녹음 버튼을 누르면 AI 스피커를 통한 실시간 소리 전송이 시작되어, 반려견 소리를 AI로 분석하고 감정분석 결과를 사용자가 확인할 수 있게 합니다.

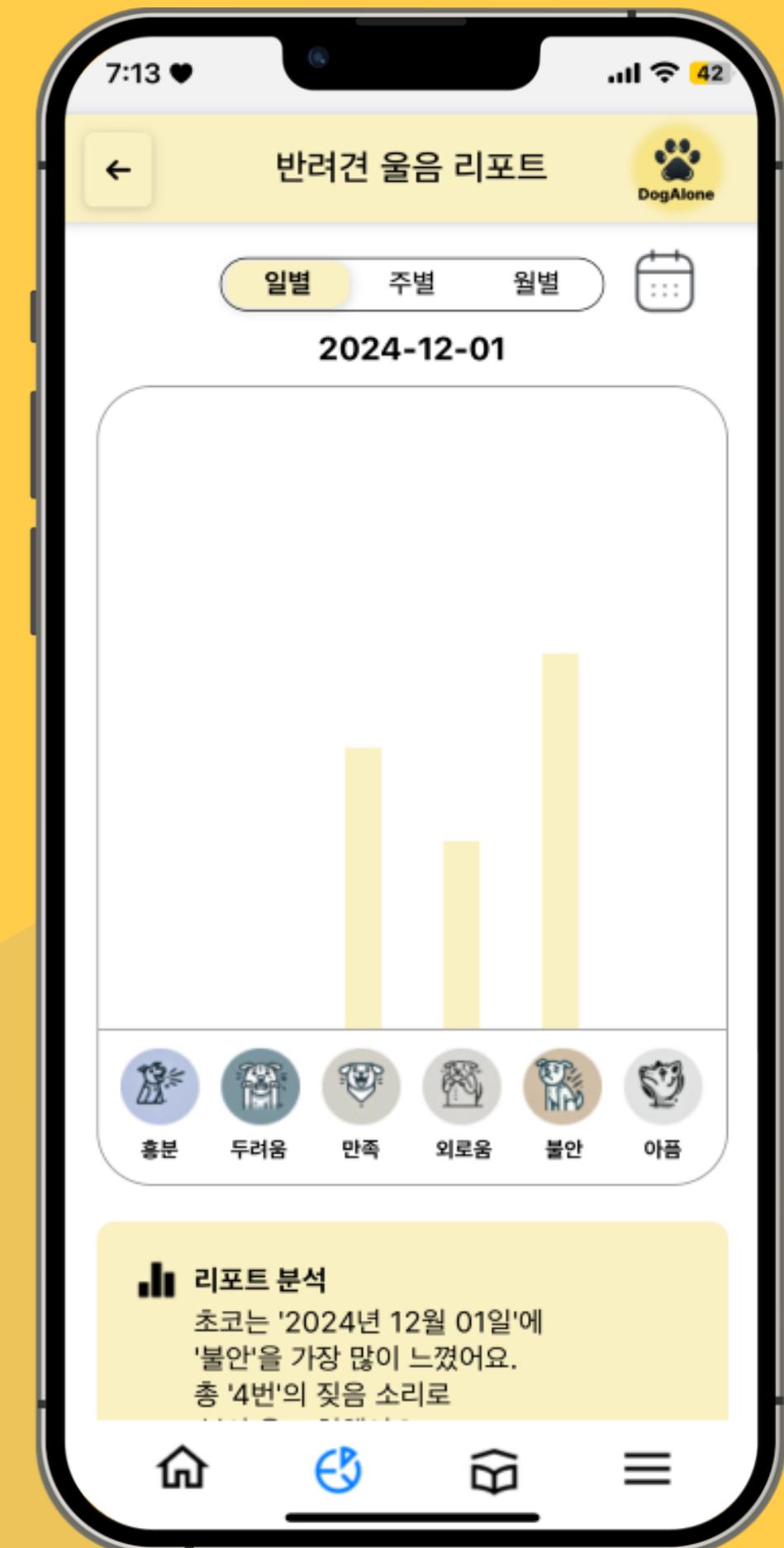
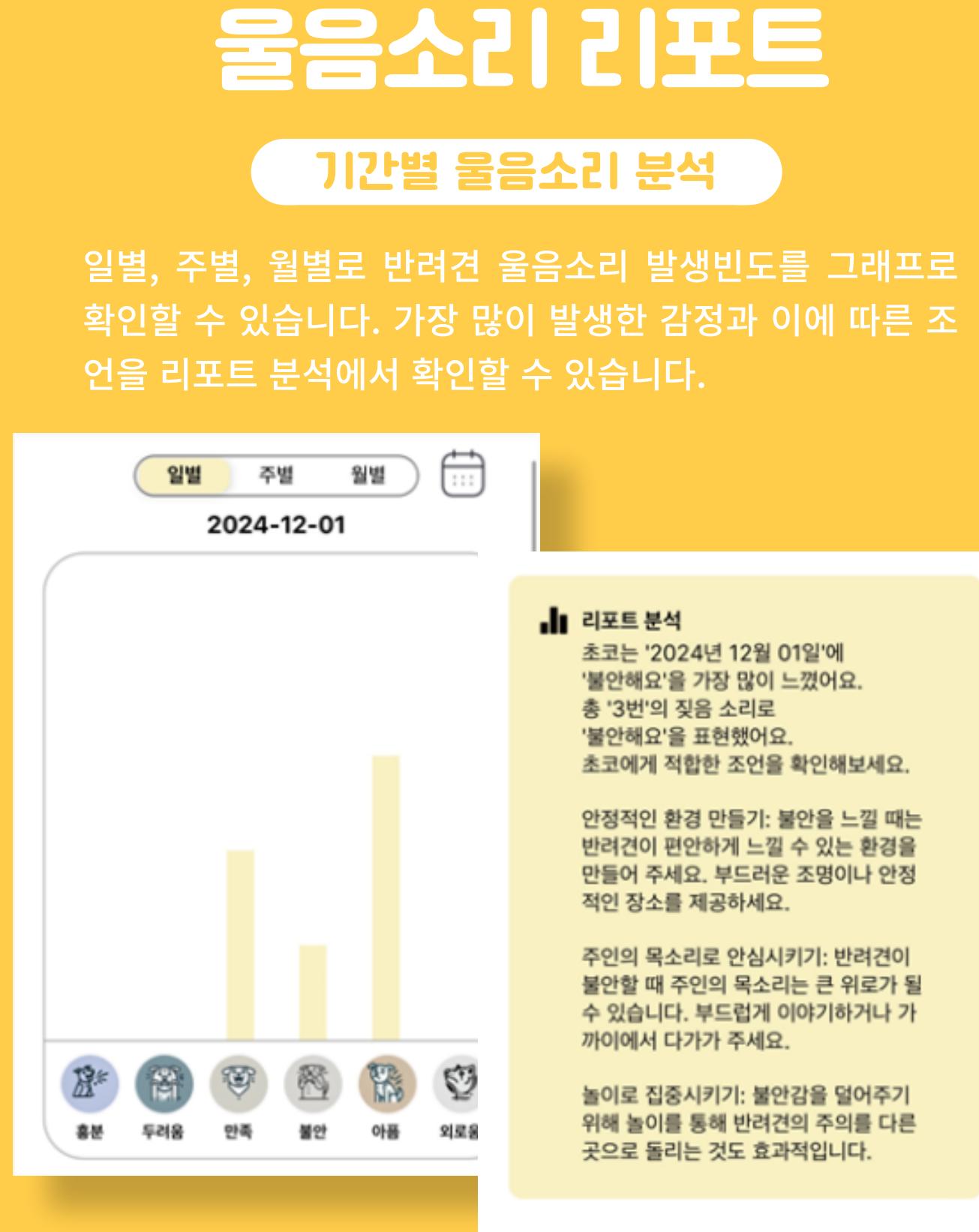


# 스마트홈 솔루션 추천

## 감정별 스마트홈 솔루션

전문가의 의견을 참고하여, 각 감정별 추천 스마트홈 솔루션을 제공합니다. 버튼을 클릭하면 스마트홈 솔루션이 시행되어 반려견이 안정을 되찾도록 도와줍니다.

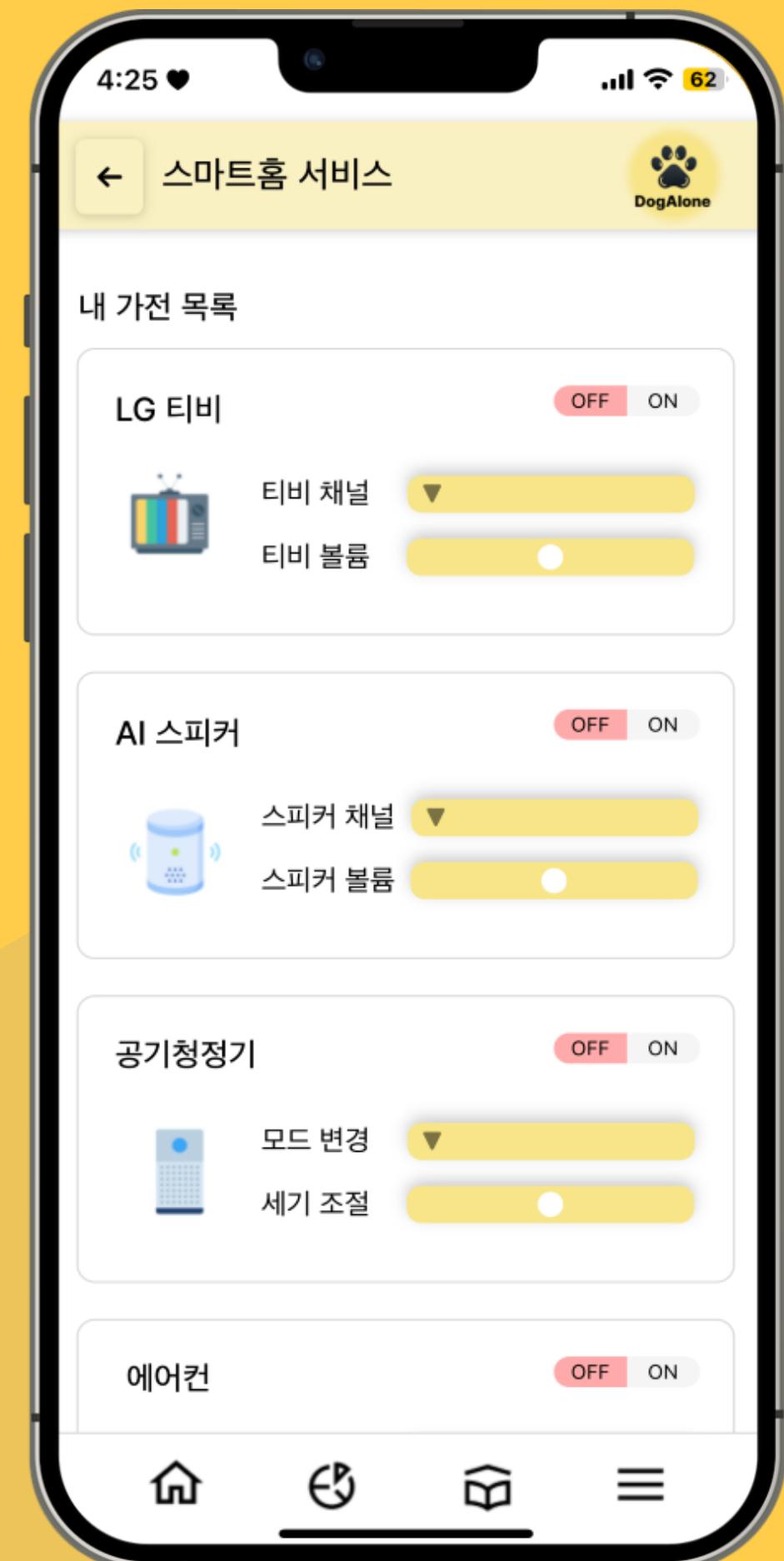
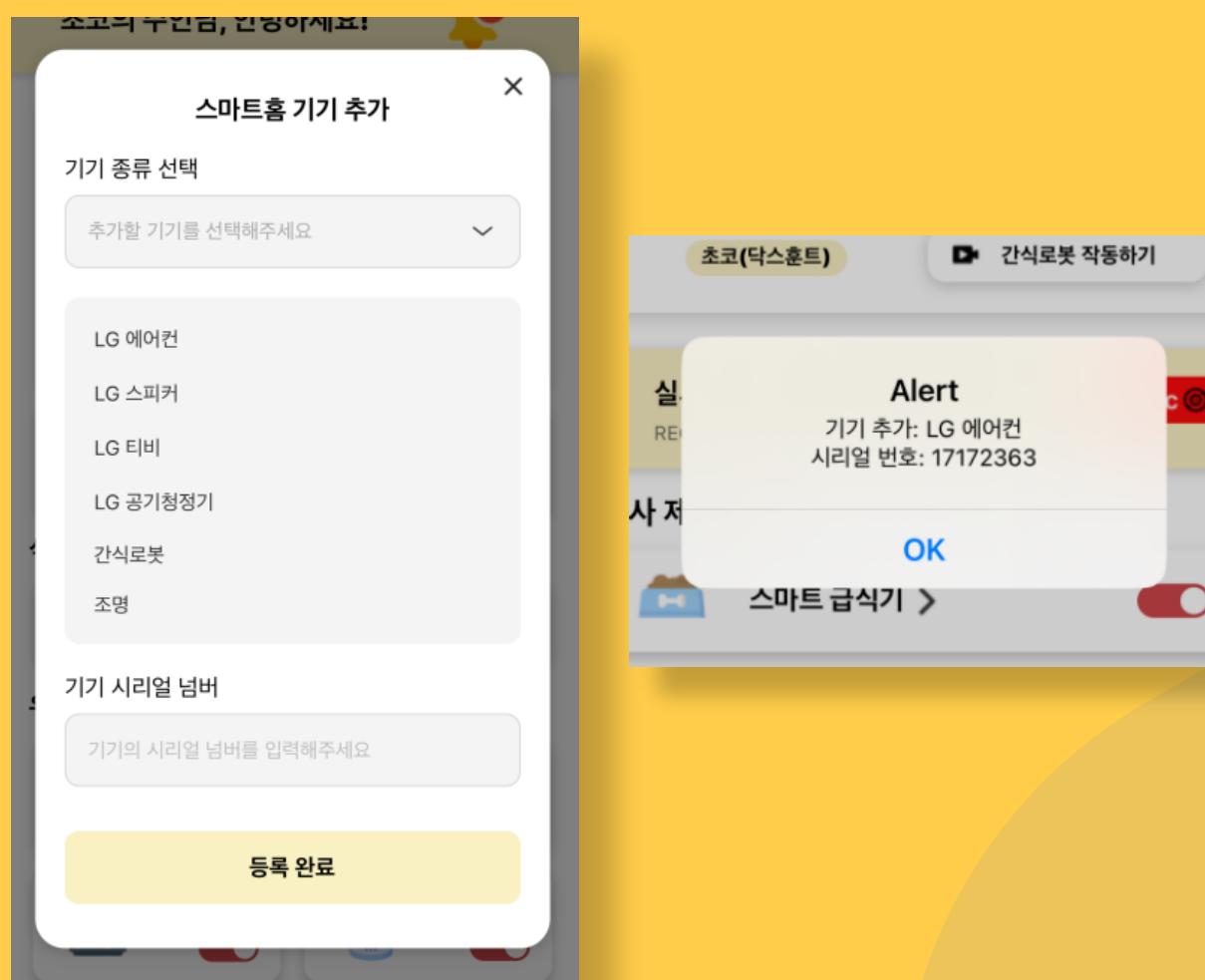




# LG 스마트홈 기기 제어

## 스마트홈 서비스 지원

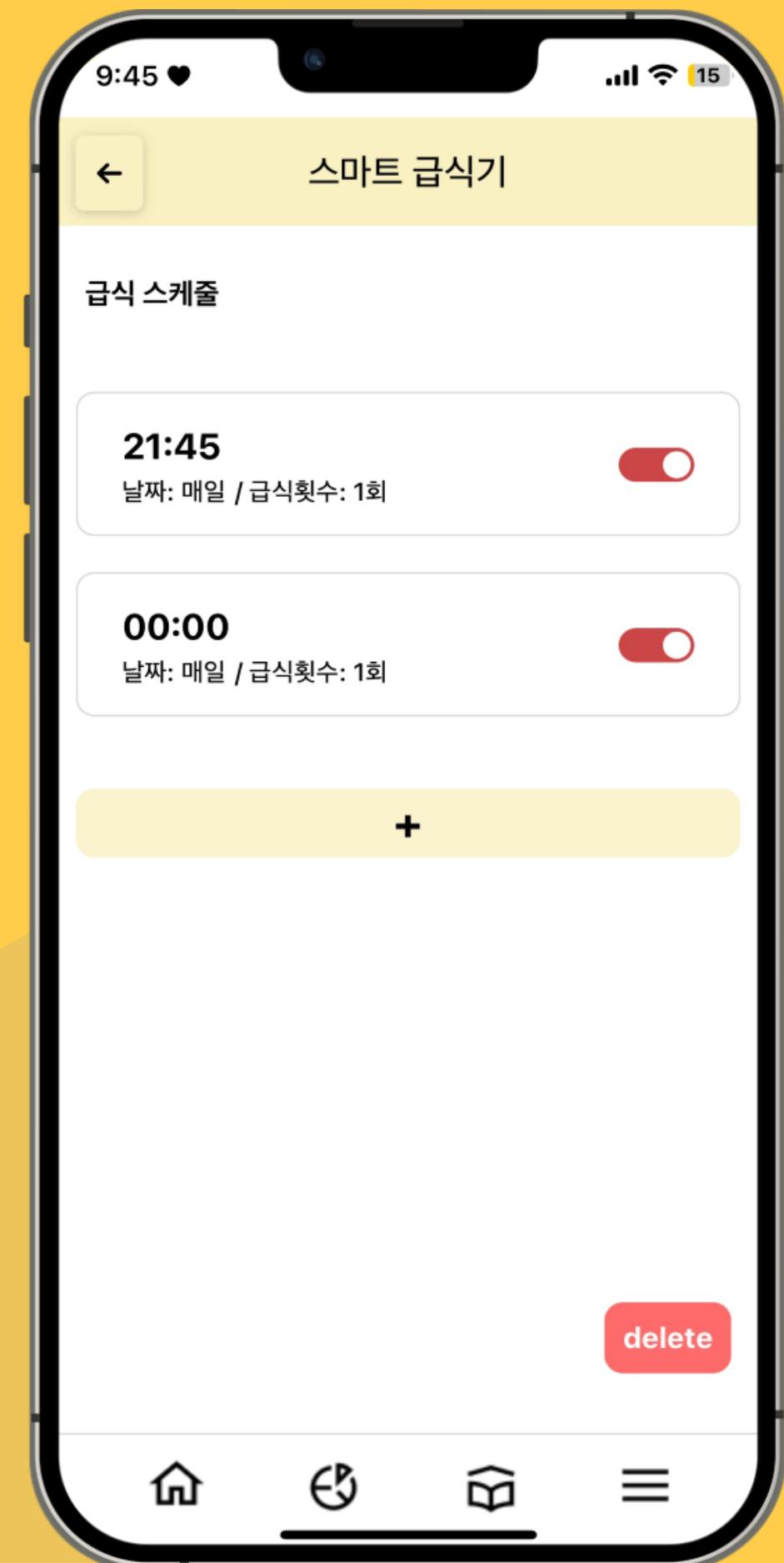
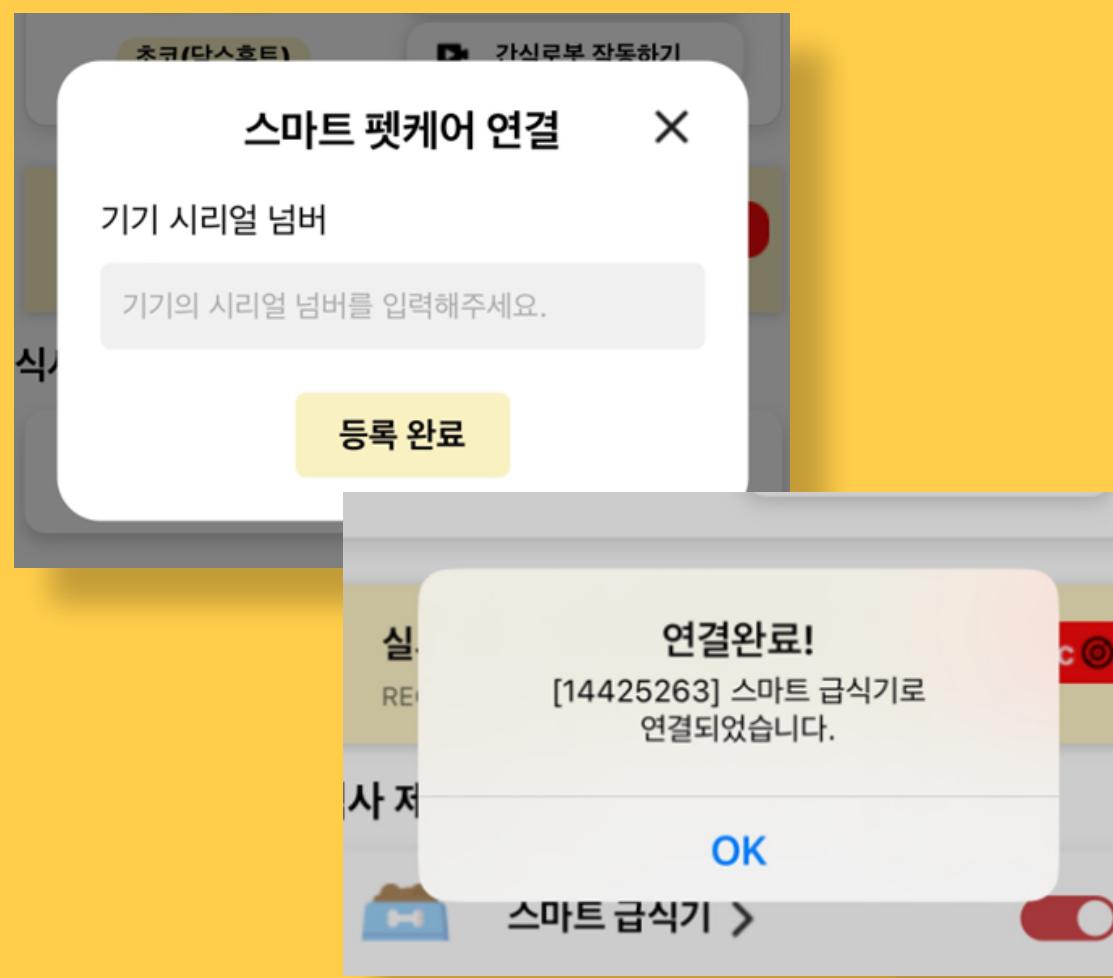
LG 스마트홈 기기를 앱에 등록하면, 어디서나 손쉽게 제어하고 관리할 수 있습니다. 외출 중에도 집 안 기기의 상태를 확인하거나 원하는 설정으로 조정할 수 있습니다.



# 반려견 식사 관리

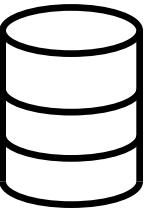
## 스마트 급식기 연동

스마트 급식기를 앱에 등록하면, 스마트 급식기가 특정 시간에 특정 양의 음식을 제공하게 설정할 수 있습니다. 급식 스케줄에 따라 식사 제공시, 푸쉬 알림을 보냅니다.



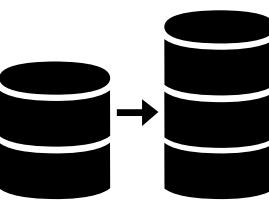


# AI 학습 과정



## 데이터셋 수집

Kaggle, 유튜브를 통해  
울음소리 데이터 600개를 수집



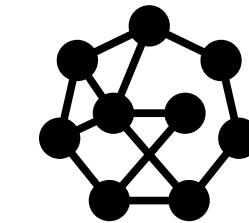
## 데이터 라벨링

논문을 참고하여 울음소리를  
주파수, 길이, 빈도에 따라 6가지로  
분류한 후, 데이터셋을 라벨링



## 스펙트로그램 변환

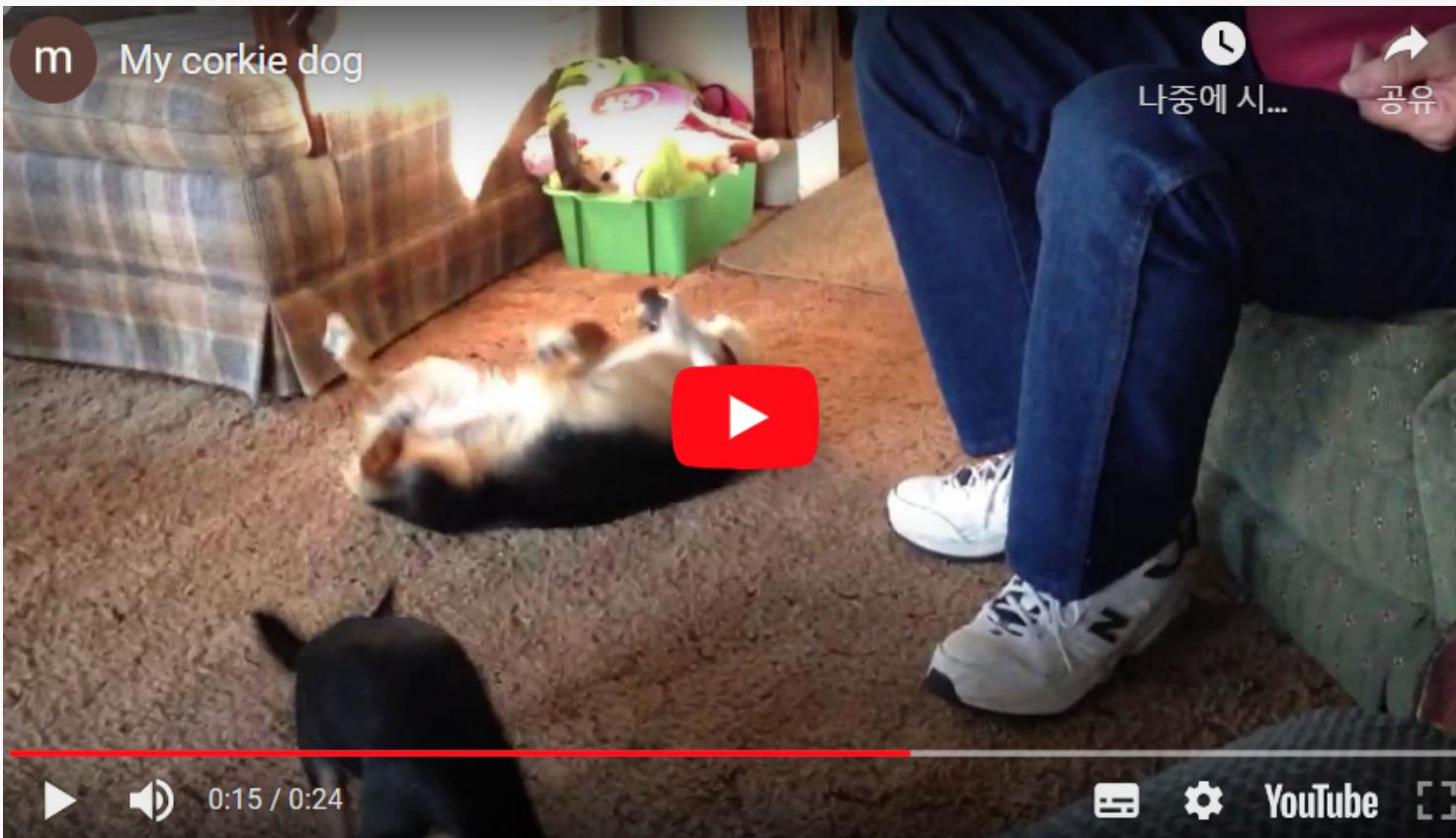
울음소리의 특징이 시각화되어  
학습 될 수 있도록 WAV 데이터를  
스펙트로그램(PNG)으로 변환



## CNN 모델 학습

패턴 인식과 특징 추출에  
강점이 있어, 주로 이미지 분류에  
사용되는 CNN 모델 학습

## 데이터셋 수집



**Dog voice emotion dataset (Demo-Lite)**

Classification of Dog voice emotion

Data Card Code (0) Discussion (0) Suggestions (0)

**About Dataset**

The dataset comprises four classes: other animal bark, normal dog bark, dog grunting, and dog growling. It contains 163 audio files, with files allocated for training and 48 for testing. The class distributions are as follows: 36 training and 14 testing files for other animal bark, 23 training and 13 testing files for normal dog barks, 23 training and 11 testing files for dog grunts, and 23 training and 10 testing files for dog growls. The other animal bark (cat) and some normal dog bark samples come from the open-source cat-dog audio dataset on Kaggle, at 16KHz. Dog grunt and growl recordings were collected from a 1.5-year-old female dog using a voice recorder app and sampled at 44.1KHz.

- Kaggle에서 3가지 종류의 음성 데이터를 찾았지만, 데이터셋의 양과 다양성이 부족하다고 판단.
- 유튜브 영상에서 소리를 다양한 울음소리를 하나씩 추출해 데이터셋으로 만듦
- 다른 소리와 섞여있어 음질이 좋지 않은 영상이 많아, 이를 추가적으로 필터링하는데 많은 노력을 투자

# 데이터 라벨링

- 울음소리를 구분하고 분류하는 기준을 정하는 일에 어려움을 느낌
- 강아지 울음소리의 **음성학적 특성**을 먼저 공부



- 음성학 관련 논문과 연구를 참고하면서 **음조(pitch), 길이(duration), 빈도수(frequency)** 같은 음향학적 특징에 따라 의미가 달라진다는 점을 이해하고 이를 바탕으로 소리와 감정을 6가지로 세분화

	Pitch	Duration	Frequency	Emotion
Bark	높음-중간	짧음	반복적	흥분
Growl	낮음	길	반복적	두려움
Grunt	낮음	짧음	반복적X	만족
Whimper	높음	짧음	반복적	불안
Howl	낮음	길	반복적X	외로움
Yip	높음	짧음	반복적	아픔

소리의 미세한 변화가 의사소통에서 큰 의미를 가질 수 있다는 점을 새롭게 인식  
감정 분석의 정확성을 높이기 위해서는 **데이터 라벨링의 정교함과 다양한 소리 샘플**이 필요하다는 것을 이해

# 스펙트로그램 변환

6개의 울음소리별로 각 100개의 음성데이터를 수집 후,  
시간에 따른 신호의 주파수 성분을 시각적으로 나타낸 그래프인 스펙트로그램으로 변환



# CNN 모델 학습

스펙트로그램 이미지 데이터를 증강 및 전처리한 뒤, CNN 모델을 설계하고 다중 클래스 분류를 위해 학습시켜 강아지 울음소리 감정 분류 모델을 생성 및 저장

```
19/19 [=====] - 24s 1s/step - loss: 0.0757 - accuracy: 0.9733
Epoch 94/100
19/19 [=====] - 23s 1s/step - loss: 0.0848 - accuracy: 0.9650
Epoch 95/100
19/19 [=====] - 24s 1s/step - loss: 0.0757 - accuracy: 0.9733
Epoch 95/100
19/19 [=====] - 24s 1s/step - loss: 0.0757 - accuracy: 0.9733
19/19 [=====] - 24s 1s/step - loss: 0.0757 - accuracy: 0.9733
Epoch 96/100
Epoch 96/100
19/19 [=====] - 25s 1s/step - loss: 0.0724 - accuracy: 0.9633
Epoch 97/100
19/19 [=====] - 23s 1s/step - loss: 0.0869 - accuracy: 0.9667
19/19 [=====] - 23s 1s/step - loss: 0.0869 - accuracy: 0.9667
Epoch 98/100
19/19 [=====] - 20s 1s/step - loss: 0.0643 - accuracy: 0.9750
Epoch 99/100
19/19 [=====] - 23s 1s/step - loss: 0.0913 - accuracy: 0.9700
Epoch 100/100
19/19 [=====] - 24s 1s/step - loss: 0.0748 - accuracy: 0.9667
```

데이터 양

총 600개

Epochs

100번

배치 크기

32

정확도

96.67%

# 데이터 전송 과정

오디오 서버와 AI 서버를 연결하여 소리데이터를 전송하고 분석결과를 받음

```
(NOBRIDGE) LOG Starting recording...
(NOBRIDGE) LOG Stopping recording...
(NOBRIDGE) LOG Recording saved at: file:///var/mobile/Containers/Data/Application/5E639968-2058-4944-9BFC-292A0F1E49A9/Library/Caches/ExponentExperienceData/@anonymous/MyNewProject1-5526b0b9-27c7-4f4c-a5ec-e91512e77b74/AV/recording-2D0E46D0-99E3-4926-B04C-3020606FC19B.wav
(NOBRIDGE) LOG Base64 데이터 길이: 901100
(NOBRIDGE) LOG Audio data sent to the server
```



```
Received audio data in Base64 format
Audio Saved!
WAV file saved to: C:\Users\SeoyeonKim\Documents\hanyangUni\4th_grade\1st_sem\SW_Eng\dev\link\backend\server\audio_data\audio_1733145208107.wav
File sent to AI server.
AI 분석 결과: { analyze_result: 'whimper', current_time: '2024-12-02 22:13:28' }
Sent data to DB and Client
Deleted file: C:\Users\SeoyeonKim\Documents\hanyangUni\4th_grade\1st_sem\SW_Eng\dev\link\backend\server\audio_data\audio_1733145208107.wav
```



```
Predicted Emotion: Whimper
Prediction Probabilities:
Bark: 0.00%
Growl: 0.00%
Grunt: 0.35%
Howl: 0.00%
Whimper: 99.62%
Yip: 0.04%
192.168.0.47 - - [02/Dec/2024 22:13:28] "POST /analyze_audio HTTP/1.1" 200 -
```



```
(NOBRIDGE) LOG Received aiResult data: {"analyze_result": "Whimper", "current_time": "2024-12-02 22:13:28"}
(NOBRIDGE) LOG Whimper
(NOBRIDGE) LOG Recommendation data: {"data": {"emotion": "Whimper", "id": 76, "recommendation": "동화 연결하기, 부드러운 조명 켜기, 동물 영상 틀기", "time": "2024-12-02T13:13:28.000Z"}}
```

## Frontend (React Native)

녹음이 시작되면, 데이터를 실시간으로 오디오 서버로 전송

## Backend (Node.js)

Base64 형태로 전송된 데이터를 wav 형태로 저장한 후, AI 서버로 전송  
AI 서버로 분석된 결과를 반환받고, 이를 프론트와 DB로 전송

## AI Backend (Flask)

wav 데이터를 스펙트로그램으로 변환하여 학습된 AI로 분석  
6가지 감정 별로 유사도를 분석하고, 가장 높은 유사도를 보인 감정을 반환

## Frontend (React Native)

가장 유사도가 높은 감정과 추천시스템을 함께 표시

# 전체 연동

## 프론트, 백, AI의 연결

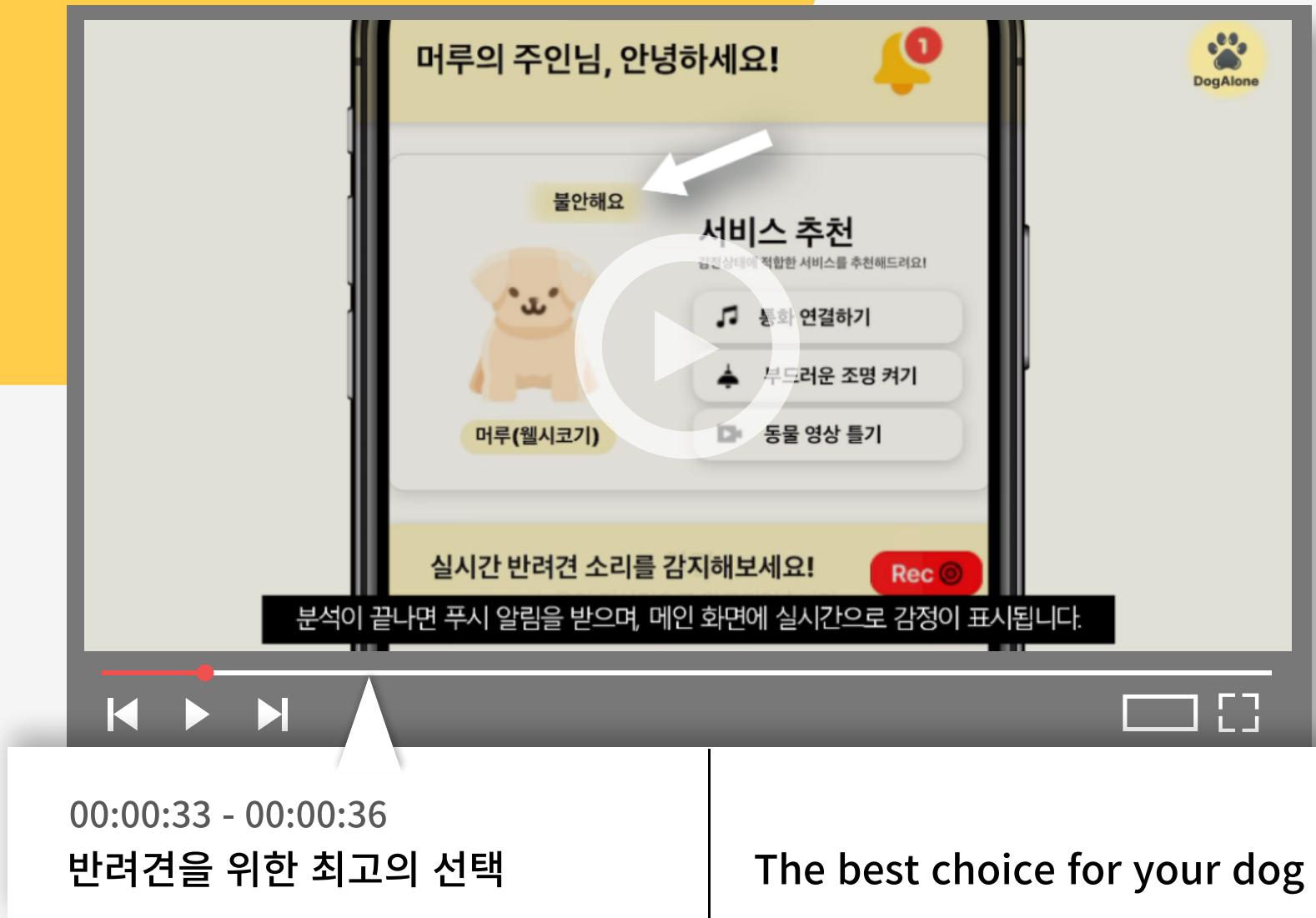
- 각 파트별로 개발을 마친 후 연동을 시작
- 연동 단계에 들어가니, 서로 생각했던 구현 방식과 전달하는 데이터 형식이 달라 수정하는데 오랜 시간이 걸림
- 특히 AI 모델과 Flask서버를 연결할 때, 모델이 학습된 환경과 서버가 실행되는 환경이 달라 AI모델을 다시 학습시켜야 했음.

```
jhjs1@LAPTOP-LH040HCL MINGW64 ~/Desktop/dog-alone/backend/server
$ node server.js
C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:5
    throw createErrorInternal(authOrCode, ...rest);
    ^
FirebaseError: Firebase: Error (auth/invalid-api-key).
    at createErrorInternal (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:5:14)
    at _assert (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:10:14)
    at Component.instanceFactory (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:15:14)
    at Provider.getOrInitializeService (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:20:14)
    at Provider.initialize (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:25:14)
    at initializeAuth (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:30:14)
    at getAuth (C:\Users\jhjs1\Desktop\dog-alone\backend\node_modules\@firebase\auth\dist\node\totp-7195c207.js:35:14)
```

```
(NOBRIDGE) LOG Bridgeless mode is enabled
(NoBridge) ERROR [AxiosError: Request failed with status code 404]
(NoBridge) WARN 🚔 React Native's New Architecture is always enabled in Expo
Go, but it is not explicitly enabled in your project app config. This may lead to
unexpected behavior when you create a production or development build. Set "new
(NoBridge) ERROR [AxiosError: Request failed with status code 404]
(NoBridge) WARN 🚔 React Native's New Architecture is always enabled in Expo
Go, but it is not explicitly enabled in your project app config. This may lead to
unexpected behavior when you create a production or development build. Set "new
ArchEnabled": true in your app.json.
Learn more: https://docs.expo.dev/guides/new-architecture/
(NoBridge) ERROR 일일 보고서 조회 중 오류 발생: [AxiosError: Network Error]
(NoBridge) ERROR 데이터 로드 실패: [AxiosError: Network Error]
(NoBridge) ERROR Failed to fetch recommendations: [AxiosError: Network Error]
```

다양한 디버깅 방법과 환경설정의 중요성을 배우게 됨  
 개발에 들어가기 전, 세부적인 구현 방식을 미리 명확히 정하는 것의 중요성을 실감

# 느낀 점



음성학과 AI 기술의 결합은 단순히 데이터를 처리하고 모델을 학습시키는 기술적 과정에 그치지 않고, 데이터를 깊이 이해하고 해석하는 학문적 접근이 필요하다는 점을 깨달은 프로젝트였습니다.

또한 감정을 분류하는 작업은 **윤리적 책임**도 동반된다는 점을 느꼈습니다. 잘못된 분석이나 판단이 강아지의 행동에 부적절한 영향을 미칠 가능성이 있기 때문에, 데이터 분석과 해석에 더욱 신중해야 함을 깨달았습니다.

## 향후 계획

- AI 모델 고도화** 더 다양한 울음소리를 수집하여 정확도 향상
- IOT** 스마트홈 솔루션 강화 및 추가기기 연동
- 클라우드 기반 시스템** 앱의 속도와 안정성을 높이는 동시에, 확장 가능성을 확보
- 앱 출시** 베타 버전으로 출시 후, 실제 사용자의 피드백 수집

**THANK YOU**

## CONTACT US



Github  
<https://github.com/pata1202/DogAlone-Your-Dog-s-Caretaker>



Youtube  
tbd

### **Seoyeon Kim**

Backend Developer  
AI Developer  
Document

### **Junhyeong Byun**

Backend Developer  
AI Developer  
Project Manager

### **Dongryul Lee**

Frontend Developer  
Design

### **Chaeyeon Jun**

Frontend Developer  
Document

### **Junhyeong Choi**

Backend Developer  
AI Developer  
Project Manager