DogAlone: Your Dog's Caretaker

*Note: Sub-titles are not captured in Xplore and should not be used

Junhyeong Choi
dept. Automotive Engineering
Hanyang University
Seoul, Korea
james200074@ hanyang.ac.kr

Junhyeong Byun
dept. Mechanical Engineering
Hanyang University
Seoul, Korea
jhjs1202@hanyang.ac.kr

Dongryul Lee
dept. Sport science
Hanyang University
Seoul, Korea
ga0212kimi@naver.com

Seoyeon Kim
dept. Tourism
Hanyang University
Seoul, Korea
2001seoyeon@gmail.com

Chaeyeon Jun
dept. Information System
Hanyang University
Seoul, Korea
oandand03@gmail.com

***Abstract*— Dogs left alone at home face many uncomfortable situations. This study focuses on developing a system that analyzes a dog's vocalizations to understand their emotional state and provides a corresponding smart home solution. By using AI-based voice recognition technology, various types of crying and barking are analyzed, enabling immediate responses even when the owner is not at home.**

Keywords*—*dog care, AI-based vocalization analysis, emotion detection, smart home integration

ROLE ASSIGNMENTS

Name	**Role & Description**	
	Role	***Description***
Dongryul Lee	Design & Front-end Developer	As a designer, he uses figma tool to establish the direction of the project, ensuring that tasks are completed on time, while facilitating communication between team members. He also creates and manages the user interface(UI) as a front-end developer.
Junhyeong Choi	Project Manager & Back-end Developer & AI Developer	He manages server-side logic, including handling database interactions, user authentication.
Junhyeong Byun	Project Manager & Back-end Developer & AI Developer	He designs and implements machine learning models to solve specific project challenges, ensuring high accuracy and efficiency. He preprocesses data, optimizes algorithms, and collaborates with the team to integrate AI solutions into the overall system. He also monitors the performance of AI models and fine-tunes them as necessary to meet project goals.
Seoyeon Kim	Document & Back-end Developer & AI Developer	She manages server-side logic, including handling database interactions, user authentication.
Chaeyeon Jun	Document & Front-end Developer	She works on converting ideas and designs into interactive elements through React Native. She adapts the technical aspects of front end to the user's needs to build an attractive app experience that meets their expectations.

I. INTRODUCTION

A. Motivation

In modern households, dogs are becoming an important member of the family beyond just pet. Raising a dog requires constant attention and affection, but in a busy modern society, it is often necessary to leave the dog alone at home. To solve this problem, our team devised a system that analyzes a dog's crying to understand its meaning and provides a smart home solution accordingly.

B. Problem Statement

As dogs become increasingly integrated into family life, their emotional well-being has garnered significant attention. However, when left alone for extended periods, many dogs experience stress, anxiety, and loneliness, which can lead to destructive behaviors and negatively impact their health.

Moreover, owners may not be able to observe or respond to their dog's distress signals in real-time, leading to a lack of effective communication between pets and their owners. This disconnect can result in prolonged periods of anxiety for the dog and increased worry for the owner. Thus, there is a pressing need for an innovative solution that not only recognizes a dog's emotional state through its vocalizations but also enables immediate and appropriate responses to alleviate their distress.

C. Research on Related Software

i) Furbo
Camera allows owners to monitor their pets remotely. It features two-way audio and a treat-tossing mechanism, enabling owners to engage with their dogs.

ii) Whistle
GPS-enabled health and activity tracker for dogs monitors fitness levels and location. It offers valuable insights into a dog's health.

iii) 헬로코코
An unused smartphone can be used as a CCTV. It offers real-time monitoring, two-way video communication, and features sound and motion detection with recording capabilities.

iv) 도기보기
It provides real-time video viewing, AI behavior detection with recording for behavior analysis, and music playback to alleviate anxiety and stress.

II. REQUIREMENT

A. Introduction

Upon launching the app, users are greeted with a series of dedicated pages that comprehensively summarize the various functions available within DogAlone. These pages serve as an essential introduction, guiding both new and returning users through the innovative features that make DogAlone an indispensable tool for pet care.

i) Intuitive Navigation
The initial pages are crafted to provide a seamless navigation experience. Users can easily scroll through a visually appealing layout that highlights each feature's purpose and functionality.

ii) Detailed Function Descriptions
Each page presents a concise yet informative overview of specific functions. These descriptions not only explain how the features work but also illustrate their benefits, empowering users to understand the value of each function in creating a supportive environment for their pets.

iii) Visual Aids and Examples
To enhance comprehension, the pages include visual aids—such as icons and illustrations—that represent each function. This visual representation makes the information more accessible and engaging, encouraging users to explore the app further.

iv) Call to Action Buttons
At the conclusion of these informative pages, users will find clear and inviting buttons for "Sign Up" and "Login." This user-friendly design ensures that interested users can quickly transition from learning about the app’s features to actively engaging with the platform.

B. Sign Up

The registration form is streamlined and intuitive, requiring only essential information to get started. Users are typically prompted to provide their name, email address, and a secure password. This minimalistic approach *ensures that the sign-up process is quick and hassle-free.*

C. Login

Registered users can easily log into the DogAlone application using their email address and password. This secure login process grants them access to their personalized dog profile, where they can view and manage important information about their pet. Additionally, users can configure and adjust their smart home settings tailored to their dog’s needs. By logging in, users can monitor their dog's emotional states, receive notifications, and ensure a comfortable environment, all from the convenience of their mobile device. This streamlined login experience enhances user engagement and allows for quick access to essential features that promote their pet's well-being.

D. Information Input

The application allows users to input and manage their dog's profile with ease. Users can enter essential details, including the dog's name, breed, and age, which help personalize the experience. Additionally, there is a section for specific characteristics, such as health conditions, dietary preferences, and behavioral traits. This comprehensive profile not only assists in tailoring the app’s features to meet the unique needs of each dog but also enables users to track important information over time.

E. Voice Recognition

The system can effectively receive the dog's crying sounds through an external microphone. This advanced voice recognition technology captures and analyzes the distinct vocalizations of the dog, such as barking, whining, or whimpering. By processing these sounds, the system can identify the dog’s emotional state and respond appropriately.

F. *Crying Analysis*

The application leverages AI technology to analyze various types of dog crying, including different vocalizations such as whining, barking, and howling. By examining these sounds, the system can determine the dog’s emotional state, whether it be anxiety, loneliness, or discomfort.

Furthermore, the app records data at user-specified intervals, allowing pet owners to monitor their dog's crying patterns over time. This data is visualized in user-friendly charts and graphs, making it easy to identify trends in crying behavior across different time periods. By offering both analysis and visualization, the application equips pet owners with valuable insights into their dog’s emotional well-being, empowering them to take proactive steps in enhancing their pet’s quality of life.

G. *Solutions based on different conditions*

The system continuously monitors the dog's status and communicates this information to the user while seamlessly controlling connected smart home devices to enhance the pet's well-being.

i) Anxiety
When the system detects signs of anxiety in the dog, it takes proactive measures to create a calming environment. It can play soothing music specifically designed to reduce stress, stream comforting videos through the smart TV, and adjust the lighting to softer tones, all of which contribute to a more relaxed atmosphere for the pet.

ii) Loneliness
To combat feelings of loneliness, the system can initiate a voice call with the user via connected speakers. This allows the owner to provide reassurance through their voice, bridging the gap between them and their pet. Additionally, it can activate the treat robot from PetCare, delivering treats or engaging the dog with interactive play to alleviate feelings of isolation.

iii) Hunger
In situations where the dog may be hungry, the system operates the automatic feeder based on the pet’s established meal intake schedule. This ensures that the dog receives its meals at the right times, promoting a healthy eating routine and preventing hunger-related stress.

iv) Pain/Discomfort
If the system detects signs of pain or discomfort, it promptly sends notifications to the owner’s smartphone, ensuring they are informed about their pet's condition. Furthermore, it can assist in finding nearby veterinary clinics, providing vital information to ensure that the pet receives timely medical attention if needed.

H. *Integration with LG U+PetCare products*

The application seamlessly integrates with a range of dog-related devices, including treat robots, automatic feeders, and smart pet care systems like 맘카. These devices are intelligently activated based on the dog’s current situation, ensuring that the pet receives timely attention and care. Moreover, the user has the convenience of remotely controlling these devices through the app. This functionality allows pet owners to monitor their dog’s behavior in real time and respond quickly to their needs, whether they are at home or away. With just a few taps on their mobile device, users can dispense food, provide treats, or activate other devices to ensure their furry companion feels loved and cared for, regardless of the distance.

III. DEVELOPMENT ENVIRONMENT

All development languages and frameworks use the most recent versions as of November 1, the start date of development, considering stability and compatibility.

A. *Frontend*

i) Platform Choice: **Windows 10**
The platform for front-end development has been set to Windows. Windows offers an intuitive UI and a wide range of development tools that enhance work efficiency. It also provides excellent compatibility with programs needed for web design and UI development. With Microsoft’s technical support, developers can rely on a stable and trustworthy development environment.

ii) Programming Language and Frameworks

· Language**: JavaScript**
Recently, a variety of JavaScript-based frameworks and libraries have been developed, making it easier to create complex web applications. JavaScript handles the dynamic aspects of development, providing the advantage of directly enhancing user experience.

· Framework**: React Native (10.8.2)**
React Native is a JavaScript framework for building mobile applications that run on both iOS and Android. It has the advantage of converting the written UI code into corresponding native elements based on the platform. Additionally, React Native operates separately from the main UI thread, allowing it to maintain fast performance without compromising the app's capabilities.

iii) Supplementary Tool for Development

· Design Collaboration Tool: **Figma Online**
Figma is a tool that facilitates smooth collaboration between designers and developers in software development projects, particularly for UI/UX design. By using Figma, the process of converting designs into code becomes more efficient, thereby increasing development productivity.

iv) Developer’s computer resource

Name	**Computer resource**	
Chaeyeon Jun	Operating System	Microsoft Windows 11 Home 23H2
	CPU	Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz

Name	Computer resource	
	RAM	8GB
	GPU	Intel(R) UHD Graphics 620
Dongryul Lee	Operating System	Microsoft Windows 11 Home
	CPU	13th Gen intel(R) Core(TM) i5-1340p CPU @ 1.90Ghz
	RAM	16G
	GPU	Intel(R) Xe Graphics

B. *Backend*

i) Platform Choice: **Windows 10,11**
The platform for back-end development has been set to Windows. Windows, a widely used operating system, offers a user-friendly interface and robust support for various development tools. It is compatible with a range of software applications and provides strong support for Microsoft technologies, making it suitable for web and desktop application development.

ii) Programming Language and Frameworks

- Language**: JavaScript (Node.js 20.8.0)**

Node.js is a powerful runtime environment for building scalable network applications, making it suitable for API servers and data processing tasks. JavaScript, commonly used in front-end development, enables seamless integration between the front end and back end.

- Framework**: Express.js 4.18.2**

Express.js is a minimal and flexible framework for Node.js web applications that offers a robust set of features for building RESTful APIs. It facilitates efficient communication between server and client applications.

- Database: **MySQL 8.0.34**

MySQL is a widely used relational database management system that effectively manages user data and AI analysis results.

- HTTP Client: **Axios 1.4.0**

Axios is a promise-based HTTP client used to perform requests between the front end and back end, facilitating data transmission and enhancing interaction between APIs and the user interface.

iii) Developer's computer resource

Name	Computer resource	
Seoyeon Kim	Operating System	Microsoft Windows 11 Pro
	CPU	11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz
	RAM	16.0GB
	GPU	Intel(R) Iris(R) Xe Graphics
Junhyeong Byun	Operating System	Microsoft Windows 10 Enterprise
	CPU	Intel(R) Core(TM) i7-8565U @ 1.80GH

Name	Computer resource	
	RAM	16GB
	GPU	Intel(R) UHD Graphics 620 의 Mirage Driver
Junhyeong Choi	Operating System	Microsoft Windows 11
	CPU	Intel(R) Core(TM) i5-8500U CPU @ 3.00GHz
	RAM	16GB
	GPU	NVIDIA GeForce GTX 1060 3GB

C. *AI*

i) Platform Choice: **Linux Ubuntu 22.04 LTS**
The operating system for AI development has been set to Linux. Due to its compatibility with open-source libraries and tools, Linux is widely used for AI model development. The Linux development environment offers stability and customization options. To use Linux on Windows, the plan is to utilize WSL (Windows Subsystem for Linux), which allows the use of Linux commands and tools in the Windows environment without requiring a separate reboot.

ii) Programming Language and Frameworks

- Language **: Python 3.12.0**

Python is the most widely used programming language in the fields of AI and machine learning, offering powerful data processing libraries and intuitive syntax that make it convenient for both researchers and developers. Python integrates well with various AI frameworks and libraries (e.g., TensorFlow, PyTorch), making it suitable for model development and experimentation, thus it has been chosen as the development language.

- Framework: **PyTorch 2.0.0, TensorFlow Serving 2.13.0, Flask 2.3.3, FastAPI 0.104.0**

PyTorch is a deep learning framework designed for training and optimizing emotion analysis models. It offers dynamic computation graphs, making it easier to experiment with complex neural network architectures and advantageous for rapid prototyping.
TensorFlow Serving is a system for deploying trained models to enable real-time predictions. This framework supports efficient model serving, allowing smooth processing of inference requests in production environments.
Flask and FastAPI are lightweight web frameworks for providing AI models as REST APIs. Flask is known for its simplicity and flexibility, while FastAPI supports high performance and automatic OpenAPI documentation generation, making it easier to build robust APIs.

iii) Developer's computer resource

Name	Computer resource	
Seoyeon Kim	Operating System	Microsoft Windows 11 Pro
	CPU	11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz
	RAM	16.0GB

Name	Computer resource	
	GPU	Intel(R) Iris(R) Xe Graphics
Junhyeong Byun	Operating System	Microsoft Windows 10 Enterprise
	CPU	Intel(R) Core(TM) i7-8565U @ 1.80GH
	RAM	16GB
	GPU	Intel(R) UHD Graphics 620 의 Mirage Driver
Junhyeong Choi	Operating System	Microsoft Windows 11
	CPU	Intel(R) Core(TM) i5-8500U CPU @ 3.00GHz
	RAM	16GB
	GPU	NVIDIA GeForce GTX 1060 3GB

D. *Cost Estimation*

Total estimate cost is $250

	Cost Category	Estimate Cost
Software Costs	AI Analysis Libraries (TensorFlow/PyTorch)	Free (open source)
	Mobile App Development Platform (React Native/Flutter)	Free (open source)
	Server and API Development (Node.js/Express.js)	Free (open source or free tier)
	Database (MySQL/Firebase)	Free (open source)
	Visualization Libraries (D3.js/Chart.js)	Free (open source)
Hardware Costs	Microphone and Sensors	$50
	LG U+ Pet Care Devices (if purchasing)	$200

E. *Software in use*

Research continues in the development of AI for communication with various animals, including dogs. This often involves using AI models that analyze human speech.

i) Wav2Vec2 Dog Barking Sound Analysis
The AI Research Institute at the University of Michigan utilized the AI model 'Wav2Vec2' to analyze dog barking sounds. They analyzed vocalization data from 74 dogs of various breeds, ages, and genders (42 Chihuahuas, 21 French Poodles, and 11 Schnauzers) to identify the emotions, breeds, and genders of the dogs based on their barking. The database includes barking sounds recorded in various situations, such as playing with their owner, encountering strangers, and reacting to doorbells. As a result, 'Wav2Vec2' achieved an accuracy of 62% in identifying the breed and emotions of dogs, 69% accuracy in identifying their gender, and 50% accuracy in identifying a specific dog among several. This suggests that sounds and patterns derived from human speech could potentially serve as a foundation for understanding animals.

ii) Convolutional Neural Network (CNN)
CNNs are powerful for image data and are suitable for learning audio data transformed into 2D image formats, such as spectrograms. CNN layers can be used to extract frequency or voice features for classification tasks. For example, one can use the Librosa library to extract MFCCs or chroma features from audio data, convert these features into 2D images, and then provide them as input to a CNN.

iii) Recurrent Neural Network (RNN) - LSTM, GRU
LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) are specialized for handling time-series data, making them advantageous for learning temporal information such as sound flow. They can effectively learn continuous sequences, like frequency features or MFCCs. A common structure is to extract local features of spectrograms using CNNs and then learn the temporal flow through RNNs.

iv) CRNN (Convolutional Recurrent Neural Network)
CRNN combines the advantages of CNNs and RNNs, using CNNs to extract frequency and voice features from audio data converted into image formats and RNNs to learn the temporal flow. The CRNN model is designed to process spatial features of audio data in the CNN block and temporal data in the RNN block. This approach enables complex tasks such as emotion prediction or sound classification.

F. *Task distribution*

Task distribution will be detailed in the next phase, which is the design phase.

IV. SPECIFICATIONS

A. *Introduction: App planning and Design*

Using Figma, the app design will be completed with a focus on user experience and functionality. The design will be intuitive, making it easy for first-time users to navigate the app. The placement, size, and color of buttons will be appropriately arranged to enhance user convenience. Icons and illustrations will be actively utilized to improve information delivery. A concise and consistent UI will be created to minimize user confusion. The main tasks include the following:

i) Creating a Flowchart

- A chart depicting the overall flow of the app, detailing what screens appear when certain buttons are pressed, to assist developers in implementation.
- The flow will be designed considering all functionalities to provide the highest possible user experience.

ii) Designing Screens by Functionality

- Conceptualizing all potential screens for each functionality.
- Establishing the layout for buttons and images, determining their appropriate placement.

iii) Finalizing Detailed Design

- Analyzing examples from various apps to contemplate appropriate design choices.
- Adopting a single design to ensure uniformity across all screens.

B. *Sign Up*

Implement the user information input, validation, and secure data storage features in conjunction with the backend, and provide a welcome screen with a brief introduction on how to use the app upon successful registration.

i) User Information Input

- UI Design: Create a user information input screen that includes fields for name, email, and password.
- Implementation: Use React Native to build the input fields, managing the entered information through state management. Validate input using regex to check the email format and password strength, displaying error messages for invalid entries.
- Backend Integration: Set up an API using Node.js and Express to store user information in the database, employing a hashing library to securely store passwords.

ii) Authentication Process

- UI Design: Provide a "Send Verification Email" button and display a notification upon clicking it to inform the user that the verification email has been sent.
- Implementation: Utilize the Nodemailer library on the backend to send the verification email. Include a link in the email that, when clicked, authenticates the user using JWT (JSON Web Token).

iii) Registration Completion Screen

- UI Design: Display a registration success message along with a brief explanation of how to use the app.
- Implementation: Redirect users to a welcome screen after registration, maintaining the registration completion status through state management.
- Additional Features: Implement a tutorial module using React Native's useState for a simple onboarding experience, leveraging a UI library to enhance user-friendliness.

C. Login

Implement a login feature that remembers user information, providing user authentication through email and password, and maintaining login status with JWT tokens. The system will also include a password reset feature via email verification for setting a new password, and enhance user security through a login persistence option and automatic logout functionality.

i) Login Screen Composition

- UI Design: Create a login screen that includes fields for email and password, a 'Login' button, and a 'Forgot Password' link.
- Implementation: Build input fields using React Native and manage the input values through state management. Use Axios to send a login request to the backend.
- Backend Integration: Develop a user authentication API using Node.js and Express, generating a JWT token upon successful login and sending it to the client. The client will store this token in local storage or session storage to maintain the login state.

ii) Password Recovery and Reset

- UI Design: Include an email input field and a 'Reset Password' button to allow users to reset their password.
- Implementation: Create an API to handle password reset requests, sending a reset link to the user's email. Use Nodemailer on the backend to send the email, including a one-time token in the link that expires after a set time.
- Password Reset Screen Composition: Redirect users to a screen where they can enter a new password when they click the link in the email. Encrypt the new password using bcrypt before storing it.

iii) Login Persistence Feature

- UI Design: Add a 'Keep Me Logged In' checkbox to allow users to opt for persistent login.
- Implementation: When users select the 'Keep Me Logged In' option, the client will store the JWT token in local storage, setting a longer expiration time for the token to maintain the login state.
- Security Considerations: Implement automatic logout after a period of inactivity. This can be achieved on the client side using setTimeout or setInterval, or by configuring token expiration times on the backend.

D. Information Input

Create a basic information input screen where users can view and enter pet information at a glance. The inputted information will be sent to the server via Axios and stored in MySQL. After saving the information, users will be able to review and modify it.

i) Basic Information Input Screen Composition

- UI Design: Create input fields for user information such as name, contact information, and pet details.
- Implementation: Use React Native to build each input field and manage the state with useState or data. Ensure the state updates in real-time as the input values change.

ii) Information Saving and Transmission

- UI Design: Add a 'Save' button to allow users to save all the information they have entered.
- Implementation: Use Axios.post to send the user information to the server. Display a notification message saying 'Information has been saved' upon successful transmission.
- Database Integration: On the backend, create an API that saves the user information in MySQL. Use SQL queries to ensure the data is properly stored in the database.

iii) Completion Screen

- UI Design: After all information is saved, display a completion message along with a summary of the information for users to review.
- Implementation: Manage the completion message using useState or data, and use conditional rendering to show the summary screen once the save operation is complete. Provide a "Edit" button to allow users to modify their information again.

E. Voice Recognition

When the user presses the microphone button, voice recognition starts, and recording begins for voice analysis. The recorded audio file will be sent to the server, and the emotional analysis results will be received.

i) Microphone Input Functionality

- UI Design: Add a microphone button for recording the dog's bark, and configure it so that when the user presses the button, voice recording begins.
- Implementation: Use the react-native-audio-recorder-player library in React Native to implement the recording feature. Add an animation or display a message like 'Listening...' on the screen to indicate that voice recognition is in progress, helping users understand the current state. Prepare the recorded audio file for transmission to the server.

ii) Sending Audio File to Server

- Implementation: Once the recording is complete, wrap the file in a JavaScript FormData object and send it to the Node.js server. Use the axios library to handle asynchronous file uploads.

iii) Processing Audio File and Emotion Analysis

- File Reception and Storage: The Node.js server will use the Multer library to store the audio file sent from the client and proceed with emotion analysis.
- Using AI Emotion Analysis Model: Input the received audio file into an AI model to predict the emotional state of the dog's bark. Deploy a CNN-LSTM-based model developed with PyTorch or TensorFlow, and build an API using TensorFlow Serving to connect with the Node.js server for real-time emotion analysis results.

iv) Displaying Emotion Analysis Results

- Result Data Handling: The server will send the emotion analysis results back to the React Native client, indicating the emotional state of the dog's bark (e.g., "Happy," "Anxious," "Alert").
- Implementation: Manage the analysis results received from the server with useState, and visualize the analyzed emotions to the user through notifications or pop-ups.

F. Crying Analysis

Based on the dog's barking data, we will develop an AI model to analyze emotional states and control smart home devices. Utilizing PyTorch and TensorFlow, we will train and deploy the model, building a system with optimal performance for audio data preprocessing and emotion analysis. The main tasks are outlined as follows:

i) Data Collection and Preprocessing

- Collecting Dog Barking Data: Gather barking audio data from various situations. Utilize public datasets when necessary, and apply data augmentation techniques (e.g., adding noise, voice modulation) to increase the volume of data if it is insufficient.
- Audio Data Preprocessing: Use Librosa to transform the barking data into a format suitable for analysis. Extract frequency features, Mel-frequency cepstral coefficients (MFCCs), and chroma features to create the characteristics that will be used for model training.

ii) Emotion Analysis Model Design and Training

- Model Design: Design a classification model to predict emotions from dog barking. Build a deep learning model that accepts audio data as input, using a CNN-LSTM architecture to classify emotions such as anxiety, happiness, and stress as a multi-class classification model.
- Model Training and Optimization: Train the model using PyTorch and optimize its performance through hyperparameter tuning. After training, deploy the model with TensorFlow Serving and connect it to a real-time prediction API.

iii) Model Performance Evaluation and Improvement

- Model Evaluation Metrics: Use metrics such as Precision, Recall, and F1 Score to evaluate the model's performance.
- Performance Optimization: Adjust hyperparameters to maintain optimal performance as needed. For instance, manage dropout rates or change optimizers to prevent overfitting and enhance generalization.

iv) API Server Construction and Integration

- Real-time Emotion Analysis API Development: Build a RESTful API using Flask or FastAPI to return the analyzed emotional states for use by both the frontend and backend.
- Model Deployment and Maintenance: Manage the deployed model using TensorFlow Serving, monitoring performance and updating the model as necessary.

This approach ensures a robust framework for analyzing dog barking sounds, contributing to better interaction with smart home systems based on the emotional states of pets.

G. Solutions based on different conditions

To facilitate seamless communication between the AI model and the frontend, we will build an API server using Node.js and Express. This server will utilize MySQL to store and manage AI analysis results and user data, integrating with LG U+ and LG ThinQ APIs for smart home control functionality. The main tasks are outlined as follows:

i) Server and Database Construction

- Node.js Server Setup: Construct an API server using Express.js to handle client requests. Implement a RESTful API that sends dog barking analysis requests to the AI model and returns the results to the frontend.
- MySQL Database Design: Design a MySQL database to efficiently store and manage AI analysis results and user information. Set up the database schema to ensure optimal data retrieval and storage.

ii) API Integration and Data Processing

- AI Analysis Request and Response API: Create an API endpoint that accepts barking audio files from the frontend, sends them to the AI model for analysis, and returns the results to the client.
- LG ThinQ and U+ API Integration: Integrate LG ThinQ API and LG U+ API for interaction with LG IoT devices. Implement logic that controls smart

home devices (e.g., lighting, music playback) based on emotional states detected by the AI model.

iii) Smart Home Control Logic Implementation

- Emotion-based Control Logic: Develop functionality to automatically control specific smart home devices based on the AI model's analysis results (e.g., anxiety, happiness). For instance, if the AI detects anxiety, it can trigger calming music playback and adjust lighting accordingly.
- Authentication Processing: Implement OAuth 2.0 authentication for using LG U+ and SKT NUGU APIs, enhancing security and user authentication. This allows users to access personalized smart home control features through their verified accounts.

iv) Data Communication with the Frontend

- Data Transmission via Axios: Use Axios for smooth data exchange between the frontend and backend. Handle requests from the client and ensure that the API server processes them appropriately.
- Error Handling and Security Enhancement: Implement exception handling for potential errors during data transmission, protecting user data and strengthening the security of the server and database.

By executing these tasks, we will create a robust API server that not only processes emotional analysis of dog barks but also integrates with smart home technologies, providing users with an enhanced interactive experience.

H. Integration with LG U+PetCare products

To enable users who own LG U+ pet care products to easily register these devices within our application, we will integrate the LG U+ pet care API for real-time monitoring and control. Here's a detailed outline of the implementation plan:

i) LG U+ Pet Care API Integration

- API Documentation Review: Examine the LG U+ pet care product API documentation to identify necessary endpoints and parameters, such as poduct status retrieval, control command transmission, user authentication endpoints.
- Authentication Setup: Configure the required authentication methods for using the LG U+ API, such as API keys or OAuth tokens, to ensure secure access to the API.

ii) API Call Functionality Implementation

- Axios Module Creation: Create a reusable Axios module for making API requests to the LG U+ pet care product endpoints. This module will facilitate sending requests to various endpoints, handling responses and errors consistently
  - Reusable API Request Module: Structure the request module to be adaptable for different API calls, ensuring efficient access to the LG U+ pet care product functionalities.

iii) Frontend Integration

- User Interface Development: Utilize React Native to design a user-friendly interface that includes display of product status (e.g., battery level, connectivity), control buttons for various functionalities (e.g., activating features, scheduling)
- Frontend and Backend Connection: Establish a seamless connection between the app's frontend and backend. Ensure that user interactions with the UI translate into commands sent to the backend, which in turn communicates with the LG U+ API.
- RESTful API Calls: Set up RESTful API calls from the app to send user commands to the backend, allowing for smooth interaction with the LG U+ pet care API.

iv) Real-time Status Updates and Feedback

- Real-time Updates via WebSocket or SSE: Implement WebSocket or Server-Sent Events (SSE) to push real-time status updates of the pet care products to the app. This enables users to see live data on their device's performance and status.
- User Feedback Mechanism: After a user issues a command, display success or failure notifications on the UI. This feedback loop will help users understand whether their control commands have been executed correctly, enhancing the overall user experience.