MAIDE: A MOTHER'S AIDE IN RECOGNIZING AND ANALYZING BABY CRIES USING MACHINE LEARNING

A Capstone Project

presented to the Faculty of the College of Computer Studies, Misamis University, Ozamiz City, Philippines

In Partial Fulfillment

of the Requirements for the degree of

Bachelor of Science in Information Technology

Jessa Abadilla

Kylah M. Ostia

Novermber 2024

GRAMMARLY MAY 2025-Revised-Capstone-Paper-Super-Updated-PRINT-FINAL-NEW-MAide-Capstone-Paper-1-1 (2) (1) (wecompress.com)

by one device only

General metrics

63,621	8,989	739	35 min 57 sec	1 hr. 9 min
characters	words	sentences	reading time	speaking time

Writing Issues

No issues found

Plagiarism

113
sources

3% of your text matches 113 sources on the web or in archives of academic publications

Unique Words

17%

Measures vocabulary diversity by calculating the unique words percentage of words used only once in your document

46%

rare words

Rare Words

Measures depth of vocabulary by identifying words that are not among the 5,000 most common English words.

Word Length

5.4

Measures average word length

characters per word

Sentence Length

12.2

Measures average sentence length

words per sentence

GRAMMARLY MAY 2025-Revised-Capstone-Paper-Super-Updated-PRINT-FINAL-NEW-MAide-Capstone-Paper-1-1 (2) (1) (wecompress.com)

Misamis University
Ozamiz City, Philippines 7200
Tel. No. +63 88 521 0387 local 117/Telefax No. +63 88 521 5817

COLLEGE OF COMPUTER STUDIES

CERTIFICATE OF PANEL APPROVAL

This Capstone entitled "**MAide: A Mother's Aide in Recognizing and Analyzing Baby Cries using Deep Learning**" developed by Jessa Abadilla, Kylah M. Ostia in partial fulfillment for the degree of Bachelor of Science in Information Technology has been examined and is recommended for acceptance and approval.

MARKDY Y. ORONG, DIT
Adviser

Approved by the committee on oral examination with a grade of _____.

Date: _____

ROSECLAREMATH A. CARORO, DIT
Chairman

FLORENCE JEAN B. TALIRONGAN, DIT
Member

HIDEAR TALIRONGAN, DIT
Member

This Capstone Project is accepted in partial fulfillment of the requirements for the degree of Bachelor of Science in Information Technology.

ROSECLAREMATH A. CARORO, DIT
Dean, College of Computer Studies

ABSTRACT

Recognizing and analyzing baby cries is crucial for identifying specific needs such as hunger, discomfort, belly pain, burping, or sleepiness. This study developed a tool, MAide, to address these challenges of recognizing baby cries and interpreting their meaning. The system integrated a real-time detection device that captures and processes baby cries, transmitting data to a mobile app. This device was developed using the Agile methodology and the Scrum framework, where modeling was used, Teachable Machine Learning, a platform built on TensorFlow, which implements a pre-trained Convolutional Neural Network (CNN) to classify and analyze baby cries based on a labeled dataset. The dataset utilized in this research comprises samples from Kaggle, GitHub, Splann, and Donate-a-Cry database.

Nevertheless, the researchers had to examine these datasets closely to guarantee their accuracy and dependability. The MAide allows mothers to register devices and add their baby's information to detect baby cries and receive instant notifications about their baby's needs. This solution enhanced caregiving by enabling timely responses, reducing stress for new parents, and improving confidence in infant care. The system was evaluated and received high functionality, accuracy, and usability ratings. The MAide platform demonstrated that the IoT-enabled system significantly enhanced infant care by recognizing and analyzing baby cries, providing mothers with a reliable and efficient tool for understanding and promptly responding to their baby's needs.

Keywords: Baby Cry Recognition, Convolutional Neural Network, IoT, Teachable Machine Learning, TensorFlow

DEDICATION

This capstone project is dedicated to our family and friends, whose tireless support and affection have been the backbone of our journey. Our family has always been there for us, providing unconditional love, guidance, and encouragement. They have been our biggest source of courage, reminding us to trust in ourselves even when the challenges appeared overwhelming. Their sacrifices and unwavering faith in our abilities have motivated us to strive for excellence, and for that we are forever thankful. We would not have come to this point without their never-ending trust in our abilities and constant encouragement. Foremost, our extend heartfelt thanks to the Almighty God, the source of strength and wisdom that empowered us to initiate and successfully complete this endeavor.

To those individuals whose company has made our journey all the more memorable. You have been here to share in both the challenges and the successes, involving late-night study sessions, numerous discussions, and moments of doubt. Your joy, advice, and unwavering trust in our achievement have helped us get through difficult times. The friendships we've established have provided us with reassurance and inspiration, assuring us that we are never alone on this journey. This project is a testament to the love, patience, and support we've received from our family and friends. Thank you for taking this journey with us. We dedicate this success to all of you.

- Researchers

ACKNOWLEDGMENT

We, the researchers would like to begin by sincerely thanking God, the Almighty, for all of his opportunities, wisdom, and favors, which made it possible for us to complete this capstone study. We are extremely thankful to the following people for their invaluable contributions, support, and knowledge, which greatly expanded our perspective and helped bring this study to come to pass. With God's direction, we found motivation in the verse, "For with God, nothing shall be impossible."

We are extremely grateful to the research committee, Dr. Roseclaremath A. Caroro, Dr. Hidear A. Talirongan, and Dr. Florence Jean Talirongan, for their approval and guidance in helping us complete this capstone project. We also acknowledge Dr. Markdy Y. Orong, our research adviser, for his support and guidance during this work.

We also want to express our deepest appreciation to our parents, whose constant financial, spiritual, and emotional support kept us going during the duration of the entire capstone project. Their words of encouragement raised our spirits and given us the determination to finish what we had started. We sincerely appreciate their tireless patience and continuous motivation, which helped us along the way.

We also like to express their gratitude to all the guardian who generously agreed to be interviewed and to pediatrician Dr. Lasaca, who made time for our interview in spite of his hectic schedule. This capstone project would not have been possible without your amazing knowledge and insights, for which we are extremely grateful.

We also want to express our gratitude to our close friends, who have encouraged us to achieve success in our research before the deadline, supported us throughout difficult times, and lifted our spirits whenever we needed it. Lastly, we would want to thank everyone who helped us out, whether quietly or overtly. I sincerely appreciate your encouragement, which greatly helped us finish this capstone project.We pray for your continuing good health and many blessings and give our sincere best wishes to each and every one of you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The experienced parents can differentiate baby crying based on experience and training. Nevertheless, hearing a baby cry is challenging for young parents and beginner childcare workers. Crying is an inherent action used by infants as the very beginning method of communication to communicate with their parents or caretakers. The vibration of the vocal cords, which are regulated by only the Central Nervous System (CNS), creates cry vocalizations related to the expression of adults (Kolandaisamy et al., 2022).

The rising awareness of how infant cries correlate with developmental and health markers has fueled advancements in cry analysis tools. Artificial intelligence and machine learning have recently gained traction for detecting and interpreting baby cries, particularly in healthcare and parenting tech. These technologies aim to identify patterns in cries to assess pain levels, hunger, or other emotional states, assisting parents and_healthcare providers (Kolandaisamy et al., 2022; HealthyChildren.org, 2024).

Globally, studies emphasize the need for accessible tools to parents, especially in low income regions with limited access to pediatric care. In countries like Japan, research into cry patterns has contributed to developing early warning systems for medical conditions. Similarly, European initiatives focus on cry analysis to assist parents in understanding emotional and physical cues, reducing parental stress, and misdiagnosis of colic or discomfort (Barr, 2020). In the Philippines, parenting programs and workshops have incorporated traditional childcare practices alongside modern insights into infant cries. Local studies highlight the cultural norms surrounding infant care, where extended family often assists in interpreting baby cries. Despite this, many resttime parents express the need for better education on understanding cries, suggesting a gap in accessible parenting resources tailored to Filipino contexts (PEPS, 2024).

All babies communicate their needs and discomforts through different cries and sounds from birth. Learning to interpret these cues accurately is critical for parents to understand and meet their baby's needs. Researchers have identified distinct categories of cries signaling hunger, discomfort, sickness, and other states. The Dunstan Baby Language theory proposes universal cries babies use to communicate specific needs like hunger or tiredness (George, A. H., & George, A. S., 2023).

Hiring a caregiver to watch infants is an option when parents are busy, or the nursery is an alternative solution. However, it is not always affordable for parents to hire a babysitter to monitor their babies (Alam et al., 2023). Recent advancements in mobile technology have enabled the development of applications that assist parents in understanding their infants' needs through cry analysis. Instead of relying on IoT devices, our system focuses on a mobile application that utilizes audio recognition algorithms to interpret baby cries. This transition emphasizes the convenience and accessibility of smartphone applications, allowing parents to monitor their baby's needs in real-time without additional hardware.

Research has shown that machine learning techniques can differentiate between various types of baby cries, helping caregivers respond promptly to their infants' needs (Yang et al., 2022). These studies utilize deep learning algorithms and audio feature extraction methods to classify cries associated with hunger, discomfort, or pain. Additionally, the mobile app environment enables the integration of user-friendly interfaces, enhancing parental engagement and understanding of their baby's requirements (Reyes-Galaviz & Arch-Tirado, 2022).

Crying is an important means infants convey their intentions to their parents. It is supposed that around the age of two months, it is possible to gradually distinguish between the different emotions in an infant's cry based on its sound. However, it is generally difficult to understand the emotion that infants wish to express through crying (the causes of crying), particularly for those who do not have enough experience in childcare. Therefore, they may fail to satisfy the infant's wants. In medical science, it is often observed that there are some differences in the expression of emotions through a cry between an infant with cerebral disorders or autism spectrum disorders and a non-disordered infant (Matsunaga et al., 2020).

Cry serves as the primary means of communication for infants. Experts (parents and childcare specialists) can distinguish infant cries through training and experience. However, it is difficult for new parents and inexperienced childcare workers to interpret infant cries (Kuo, 2019). Baby cry sound recognition enables timely notification and allows parents to monitor when their baby is crying remotely. Thus, automated recognition of crying sound patterns has become a popular research topic in developing innovative baby monitoring systems and analyzing various crying sound patterns in different contexts such as hunger, sleepiness, pain, and so on (Manikanta, Soman & Manikandan, 2019).

Several studies have been conducted on the acoustic analysis of an infant's cry from the viewpoint of emotion detection. Infants' pain is one of the traditional research topics for the detection of emotions. In recent years, "hunger" and "sleepiness" have also been studied (Matsunaga et al., 2020). Diagnosing a baby's feelings poses a challenge for doctors and parents because babies cannot explain their feelings through expression or speech. Understanding babies' emotions and associated expressions during different sensations, such as hunger, pain, etc., is complicated. All communication and feelings are propagated in infancy through cry speech, a natural phenomenon (Yasin et al., 2022).

The brains of babies work similarly to those of adults when exposed to identical painful stimuli. To express their feelings, babies "speak" by crying. For example, the feeling of pain is usually expressed by babies by making a sound similar to "ouch". Similarly, when the baby cannot read the environment but knows something is wrong, a unique means of communication is required. However, understanding a baby's emotions and needs is challenging, requiring an understanding of cries showing only minor differences. The cry is a dynamic and advanced signal with a unique characteristic that alerts parents to the need for action (Yasin et al., 2022).

Conceptual Framework

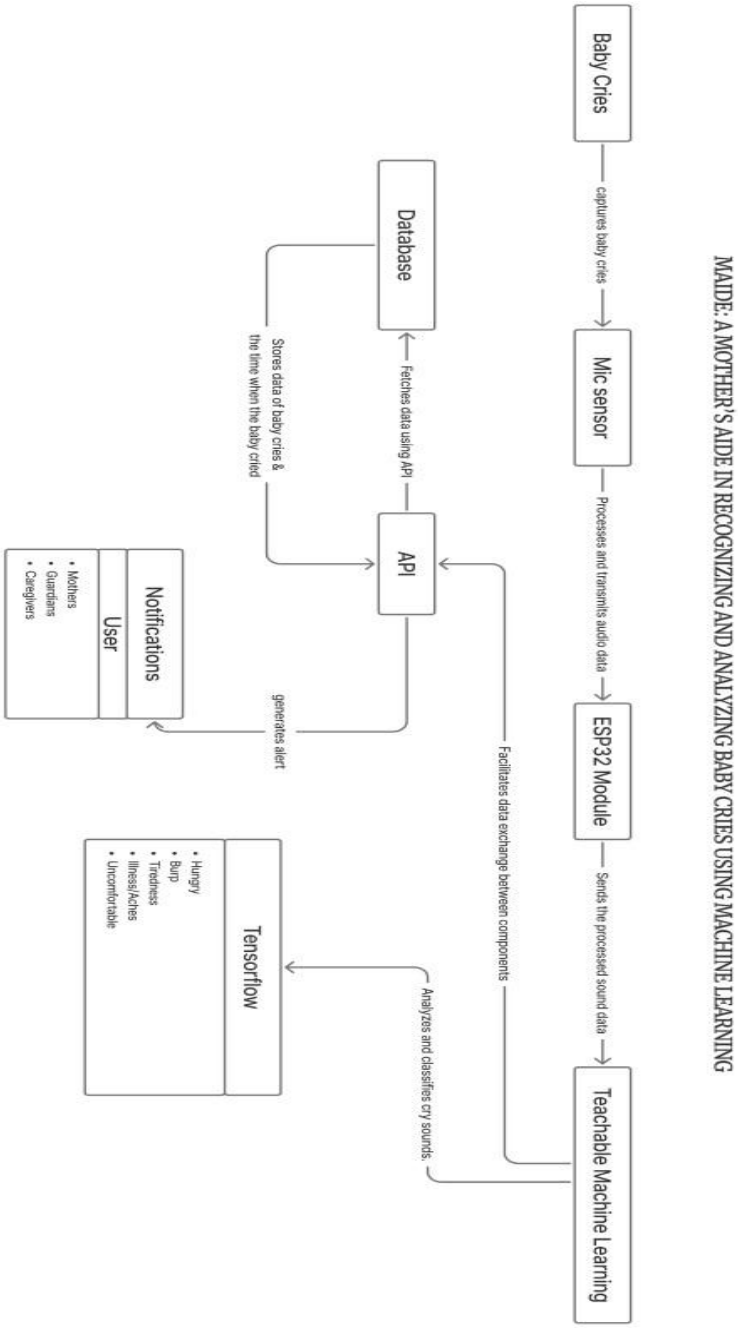

Figure 1 MAide: A Mother's Aide in Recognizing and Analyzing Baby Cries Using Machine Learning Conceptual Framework

This system focuses on recognizing and analyzing infant cries to help parents better understand their babies. Named "MAide: A Mother's Aide in Analyzing and Recognizing Baby Cries using Deep Learning," this system utilizes machine learning techniques to classify various baby cries.

Figure 1 presents the conceptual framework of MAide. The MAide project is designed to assist mothers and guardians in recognizing and understanding their baby's cries using a microphone and machine learning technology. The process begins when the microphone captures the baby's cry in real time. This audio is processed by a machine learning model built with TensorFlow, which analyzes and classifies the cry based on the Dunstan Baby Language framework into categories such as hunger, need to burp, sleepiness, belly pain, and discomfort.

The classification process enables accurate identification of the baby's needs. An API connects the system to a database, where patterns in the classified cries are stored and tracked. The system generates real-time notifications to inform caregivers about the most likely reason for the baby's cry. This approach empowers mothers and guardians to address their baby's needs promptly, enhancing caregiving through the integration of machine learning and IoT technology.

The study aims to develop and implement MAide, a machine learning-based system designed to recognize and analyze infant cries. This system will thereby assist parents in understanding and responding to their infants' needs more effectively.

The specific objectives of the study are the following:

- Detects baby cries using an audio sensor
- Detect, recognize, and classify different types of baby cries (e.g., hunger, discomfort, sleepiness, burping, and belly pain) based on an existing dataset of labeled audio recordings.
- Allow mothers to register their devices easily and receive real-time notifications.
- Allow mothers to share the device with others for collaborative monitoring and care.
- An alert notification system should be applied to notify parents or guardians.
- Allow mothers to add, edit, and remove notes on the notification.
- Evaluate the system's performance in recognizing and analyzing baby cries, focusing on accuracy, responsiveness, and user satisfaction.

The development of MAide promises several significant benefits. By utilizing machine learning for cry analysis, MAide is expected to offer a more precise interpretation of various infant cries than traditional methods, thus enhancing accuracy. This advanced technology also has the potential to reduce parents' stress by providing real-time, data-driven insights that alleviate the uncertainty often associated with infant care. Furthermore, accurately recognizing infant cries can facilitate timely and appropriate responses to their needs, significantly improving their comfort and overall well-being.

This study offers real-time monitoring and identification of various baby cries in indoor residents, providing important information about the needs and emotional states of the baby. The following are the benefits of this study:

Mothers. The device helps mothers better understand and meet their baby's needs by giving them instant feedback on their baby's cries. This strengthens the mother-child connection and may lessen maternal stress by improving maternal response, as mothers can swiftly distinguish between cries for hunger, discomfort, tiredness, tummy pain, or burping.

Caregivers. The system provides trustworthy information on what each sort of cry most likely indicates to help caretakers who might not be intimately familiar with the baby's particular needs. This lowers the possibility of misunderstandings and enables caregivers to respond appropriately and promptly to the baby's needs even when the mother is not present.

Guardians. The method offers essential support to guardians who might not provide the child with regular care in comprehending the infant's wants. This is particularly helpful when the guardians are temporarily in charge of the child because it enables them to keep an eye on the child and react appropriately, which benefits the child's overall well-being.

Other household members. Ensuring that other family members or siblings can comprehend the meaning behind the baby's cry makes the home safer and more nurturing for the child. This can improve the safety and care given in the home and raise awareness of the baby's needs.

Despite technological advancements, there is a notable gap in automated systems designed to assist parents in interpreting and responding to infant cries effectively. Existing solutions often lack the precision required for accurate cry analysis and recognition, which can result in ineffective or delayed responses to an infant's needs. This inadequacy underscores the need for a more reliable and objective method of analyzing and categorizing baby cries. The scope of the MAide project includes several key

components. The project will begin with collecting a diverse dataset of infant cries, capturing various types and conditions. Such sounds can be considered primitive words and are classified according to the "Dunstan Baby Language" (DBL).

According to the DBL, baby cries are considered universal, meaning parents can understand them worldwide. Priscilla Dunstan, the creator of this theory, identified universal sounds that babies make within the last few months of life, regardless of language or culture. These sounds include: "Neh" – Hunger, "Owh" – Sleepiness, "Heh" – Discomfort, "Eairh" – Lower gas, and "Eh" – Burp. The MAide project aims to revolutionize how parents or caregivers understand and respond to infant needs by developing a system that interprets different baby cries. This is inspired by the Dunstan Baby Language, which asserts that newborns produce specific sounds, or "pre-cry reflexes," that correspond to their immediate needs, such as hunger, burping, belly pain, discomfort, or sleepiness. By leveraging this theory, the MAide project created a device that recognizes and analyzes these five (5) classifications of cries.

In households with more than one baby, it is recommended that parents acquire a separate microphone for each baby. The system is designed to function with one microphone per baby, requiring each child to have a dedicated device for optimal performance. This is essential to ensure that each child's cries and needs are accurately identified and addressed, preventing any potential overlap or confusion resulting from using a single device for multiple children.

Additionally, the system is designed to recognize only four classes of baby cries for infants aged 0 to 8 months: hunger, sleepiness, need to burp, belly pain, and discomfort. If the baby has a hoarse voice, the system may not recognize the cry accurately. The system's minimum requirements are Android versions 7 through 14 to ensure optimal performance and compatibility, with quality contingent on the phone's capabilities.

RESEARCH METHODOLOGY

Agile, specifically through the Scrum framework, facilitates the development of the MAide: A Mother's Aide in Analyzing and Recognizing Baby Cries using Machine Learning project by ensuring iterative progress and enhanced collaboration. Scrum's structure of sequential sprints provides clear timelines and allows the researchers to deliver incremental updates, focusing on each feature or component step by step. This iterative approach prioritizes functionalities from the product backlog, such as real-time audio detection, cry classification, and notification systems, ensuring that essential features are operational in the development process. Regular sprint reviews enable continuous feedback and adjustments, ensuring the system aligns with user needs, achieves high accuracy in cry recognition, and remains reliable for caregivers.

The development of MAide adhered to the Software Development Life Cycle (SDLC) process, which encompasses several key phases, such as planning, analysis, design, testing, building, deployment, and maintenance. One commonly employed methodology for software development is Scrum, an agile framework primarily utilized in managing intricate projects, especially in technology, engineering, and product management domains.

In the Scrum process, teams operate within defined sprint cycles, which are time-constrained intervals during which they complete a specific amount of work. As part of this process, the product backlog, which comprises a list of components essential for the product, undergoes regular updates. Sprint prioritization is determined by the importance of each item on the backlog, with the most critical items receiving attention first. The product backlog serves as a visual representation of pending work and a tool for translating user stories into functional code. This approach enables developers to introduce new features systematically, modify existing ones, and address bugs in a well-organized and prioritized manner (Sedano et al., 2019).

Product and sprint Backlog

In Agile and Scrum approaches, the Product and Sprint Backlogs are essential components that support teams' productivity and organization. The Product Backlog is a list of features, repairs, and enhancements that must be completed and arranged according to priority. This list is frequently updated to consider user input, shifting priorities, and market demands. The Sprint Backlog is a smaller, time-bound selection of tasks from the Product Backlog scheduled to be finished in a sprint. When combined, these backlogs assist the researchers in concentrating on completing the most important tasks and adapting to changes to give users the best possible experience.

The finished product backlog is a list of all the tasks and user stories that must be performed before the project is considered complete. The product backlog is an flexible, prioritized list that includes features, enhancements, and fixes for a product. It acts as the primary source of requirements for the development team, reflecting the changing scope of work. The backlog is frequently updated and refined based on feedback, shifting priorities, and market needs, ensuring the team consistently focuses on the most valuable and relevant tasks to maximize value for endusers.

Table 1 Product-Sprint Backlog of the Design User Interface (Client: Parents/Guardians)

User's Stories	Product	Sprint
1. User's usage authorization	1.a Develop a login page for the mobile application	1.a.1 Create a user registration page
		1.a.2 Create a login page
2. User-friendly interactive interfaces for the homeowners	2.a Design User Interface	2.a.1 Design Home Tab for Mothers
		2.a.2 Design Add Device UI
		2.a.3 Design View Device UI

		2.a.4 Design Edit Device UI
		2.a.5 Design Delete Device UI
		2.a.6 Design Add and Assign Baby UI
		2.a.7 Design View Baby and Remove assigned Device
		2.a.8 Design Update Baby UI
		2.a.9 Design Delete Baby UI
		2.a.10 Design Profile Settings UI
		2.a.11 Design Device Sharing UI
		2.a.12 Design “acknowledge” UI
		2.a.13 Design Add and View Note UI
		2.a.14 Design Delete Note UI
3. User’s data storage	3.a Database design	3.a.1 Design ERD
		3.a.2 Convert ERD into physical database
4. User’s Device Configuration	4.a Configure IoT Device	4.a.1 Design Prototype
5. Developing Users Backend Functionality	5.a Backend Development	5.a.1 Design and implement baby cry recognition through notification management
		5.a.2 Design and implement account management function

		5.a.4 Design and implement device management function
		5.a.5 Design and implement baby management function

Table 1 outlines the key sprints and tasks involved in developing the system for recognizing and analyzing baby cries, focusing on various functional areas necessary for its implementation. These tasks are organized into eight main categories: User's usage authorization, Accurate Cry Classification, Real-time Notifications for Mothers, User's data storage, System Testing and Feedback, Configuration, and Developing User's Backend Functionality. Each category includes specific sub-tasks aimed at addressing different features of the system, such as detecting baby cries, analyzing their patterns, creating an easy-to-use interface for parents, storing and managing data, and developing the backend functionality needed for categorizing cry types and behaviors. This structured approach ensures that each system component is developed systematically, with clear milestones for each project phase.

Figures 2 and 3 illustrate the sprints planned for the product backlog, focusing on the initial development tasks for the mobile application. Key priorities include creating essential user authentication features, beginning with the development of a user registration page, followed by the creation of a login page. These tasks are crucial for establishing secure access and managing user accounts within the app.

Product 1.a: Develop a login page for the mobile application
Sprint 1.a.1: Create a user registration page

The mobile application's registration screen (Figure 2) is intended to be an intuitive and accessible entry point for new users, particularly mothers looking to effectively monitor their babies. This screen is an important part of the user onboarding process, allowing them to create an account and start using the app's features. The design prioritizes usability to provide a streamlined experience for all new registrants.

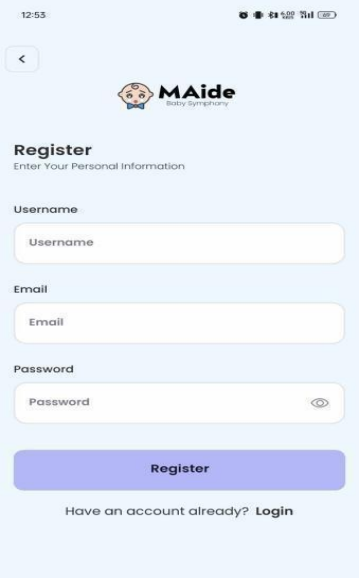

Figure 2 Registration Page

Sprint 1.a.2: Create a login page

Figure 3 shows the login screen of the mobile application, designed as part of the user onboarding process. This screen enables returning users to securely access their accounts, supporting the core functionality of the water management system.

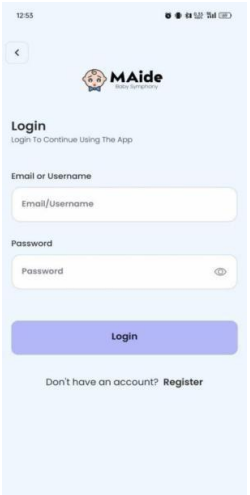

Figure 3: Login Page

Figure 3 showcases the "Login" section, which contains input fields for "Username" and "Password." The password field includes a visibility toggle icon to improve usability, and a prominent "Login" button enables users to access their accounts quickly and securely.

Product 2.a: Design User Interface
Sprint 2.a.1: Design Home Tab for Mothers

Figure 4 displays the home tab explicitly designed for mothers, providing a user-friendly dashboard to view recent notifications, baby profiles, and connected devices. This home tab includes real-time updates, informing mothers about their baby's status and any systemgenerated alerts. Key features include an overview of recent activity, quick access to each baby's profile, and device management tools, enabling mothers to monitor and manage multiple babies and devices from one central location. This design ensures that essential information is readily available to mothers, supporting better tracking and response to their baby's needs.

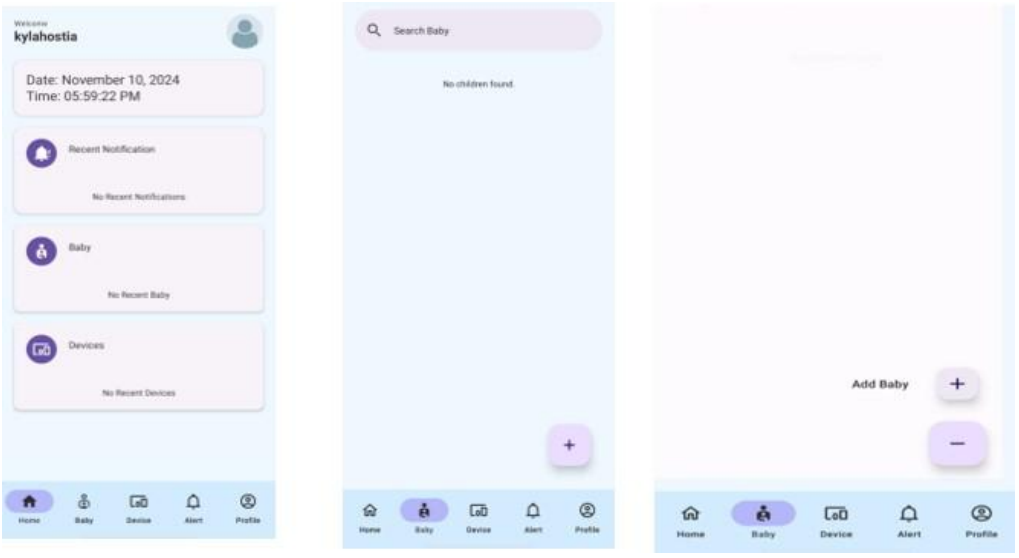

Figure 4: Design Home Tab

Figure 4 displays the user's home tab, offering a comprehensive overview of devices and babies. Figures 4 to 8 highlight the user interface designed for efficient device management, illustrating various components and functionalities that enable users to monitor babies effectively.

Sprint 2a.2: Design Add Device UI

In the figure provided, it is essential to incorporate an "Add Device" feature, allowing users to input crucial information about the device, such as the device ID. This functionality lets the system establish a personalized connection between the monitoring device and the baby's profile. By associating each device with a specific baby, the system can provide more tailored responses and accurately track the baby's cry patterns and needs.

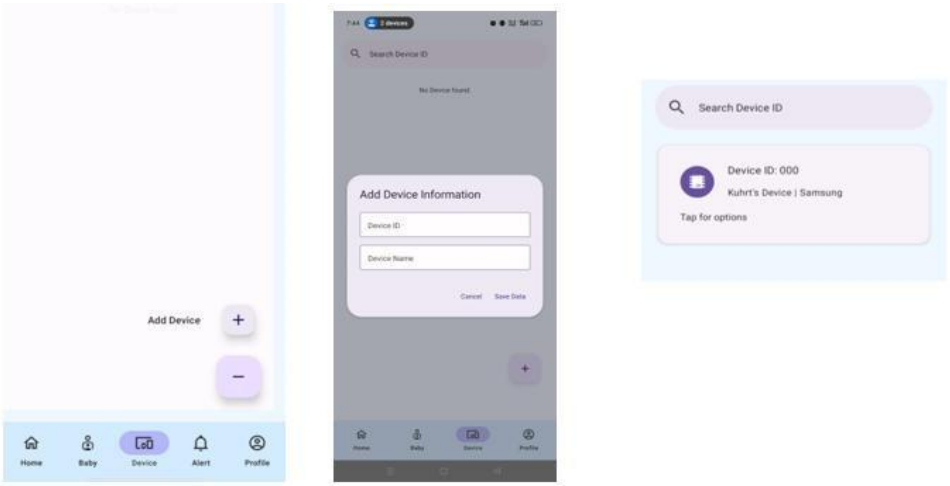

Figure 5: Design Add Device

Figure 5 shows the initial steps for adding a device to the system. First, users should view the homepage, then click the "Device" button in the navigation bar. Next, they should click the "Add Device" option to open the form. Here, users can enter the device information, such as name, location, and unique identifier. Finally, they should click the "Save" Button to store the device data in the system.

Sprint 2.a.3: Design View Device UI

Figure 6 depicts the device information feature, which allows users to easily view the details of a particular device registered in the system. This feature displays important information such as the device ID, name, and the precise date and time the device was added. It also includes a section with information about the child associated with the device, such as their name, age, and gender. This feature enables users to easily monitor and manage the connection between each device and the baby it is assigned to, promoting accurate tracking and personalized device usage.

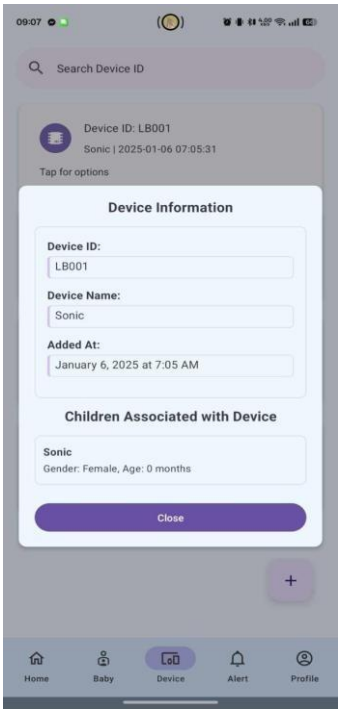

Figure 6: View Device Information

Sprint 2.a.4: Design Edit Device UI

Figure 7 depicts the device modification feature, which is a core functionality that allows users to update the information of an existing monitoring device within the system. This allows for greater flexibility and accuracy in managing connected devices, ensuring that the monitoring system is up to date with the user's current requirements and device configurations.

Figure 7: Design Update Device

Sprint 2.a.5: Design Delete Device UI

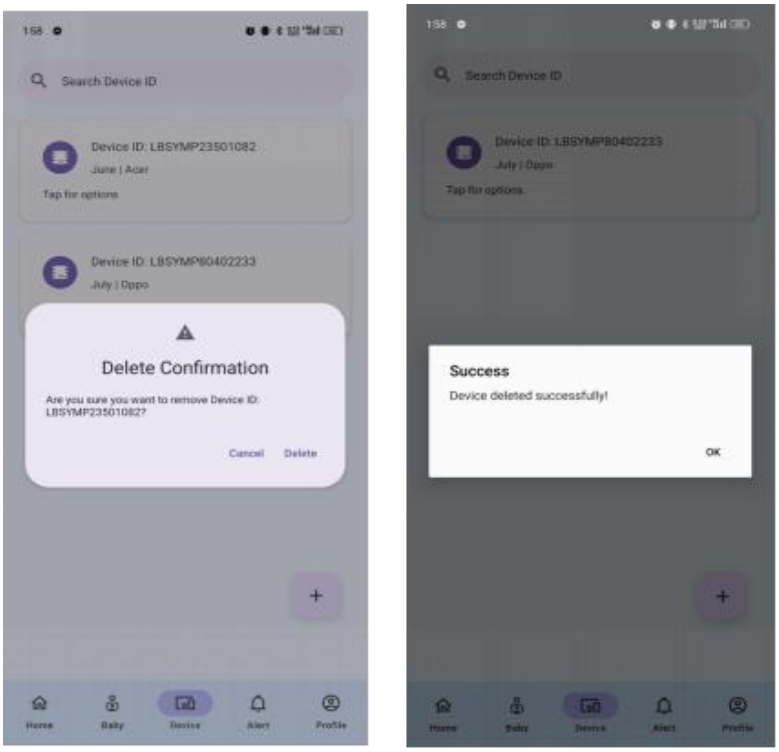

Figure 8: Design Delete Device

Figure 8 depicts a critical feature: users' ability to remove a device from the monitoring system. This feature gives users control over their connected devices, allowing them to manage their inventory of monitored items by easily disconnecting devices that are no longer in use or required for monitoring.

Sprint 2.1.6: Add and Assign Baby

Figure 9 shows a multifaceted feature that allows users to add new baby profiles while also managing existing ones within the monitoring system. This includes features like entering baby information (such as a nickname, age, and gender), assigning devices to specific babies for monitoring, and managing these assignments. The screenshots show the workflow from entering a baby's information to successfully assigning a device, demonstrating the user's ability to keep an organized and effective monitoring environment for multiple children.

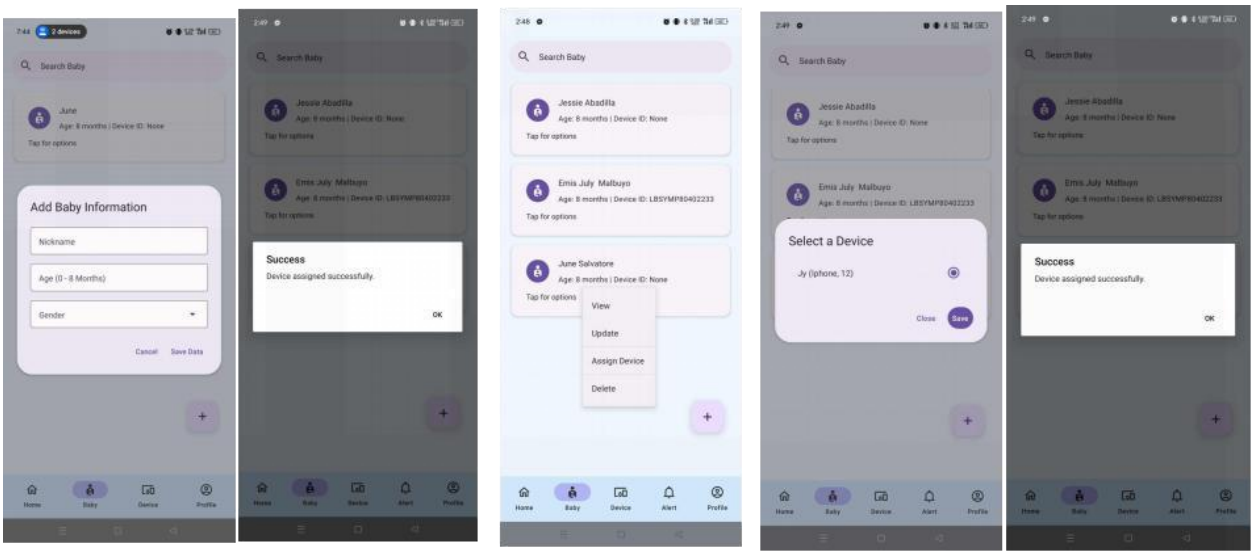

Figure 9: Add and Assign Baby

Figure 9 showcases a feature that enables users to add a baby and assign it to a specific device. This process begins with an "Add Baby" button that, when clicked, opens a form where users can input relevant details such as the baby's name, age in months, and other identifying information. Once the

baby is added, the user can assign the baby to a registered device from a drop-down list or selection menu. This ensures that each device can be associated with a specific child, making it easier for users to manage device interactions and track usage. The section is designed for user convenience, streamlining the process of adding and linking babies to devices for better organization and oversight.

Sprint 2.a.7: View Baby and Remove Assigned Device

Figure 10 depicts a key feature that enables users to view detailed information about a specific baby and manage the associated devices. Specifically, it demonstrates the ability to view the baby's profile information and, more importantly, to remove a device assigned to that baby. This provides users with granular control over each child's monitoring setup, allowing for accurate and flexible device management as needed.

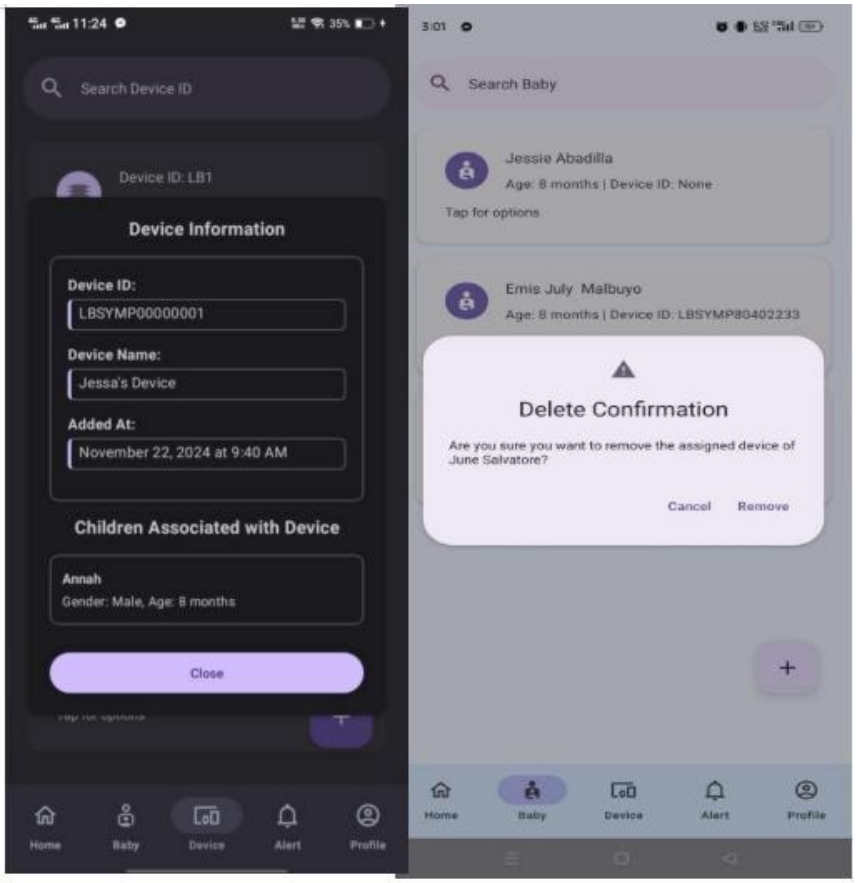

Figure 10: View Baby and Remove Assigned Device

Sprint 2.a.8: Update Baby UI

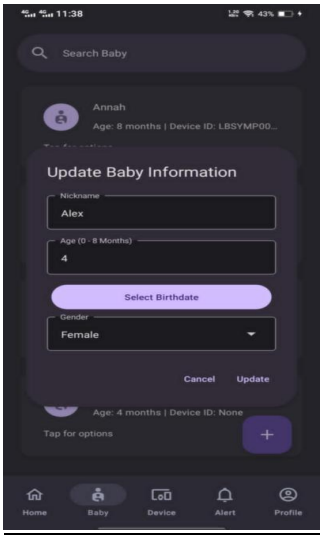

Figure11: Update Baby

Figure 11 shows a feature that allows users to update a baby's information. Users are directed to an editable view displaying the baby's existing details by selecting a baby's profile from the main list. Within this view, users can make changes to any of the available fields. For instance, they can correct spelling errors, update the date of birth, or add new notes related to the baby's growth or medical history. The interface typically includes form fields that are easy to navigate and modify.

Sprint 2.a.9: Delete Baby UI

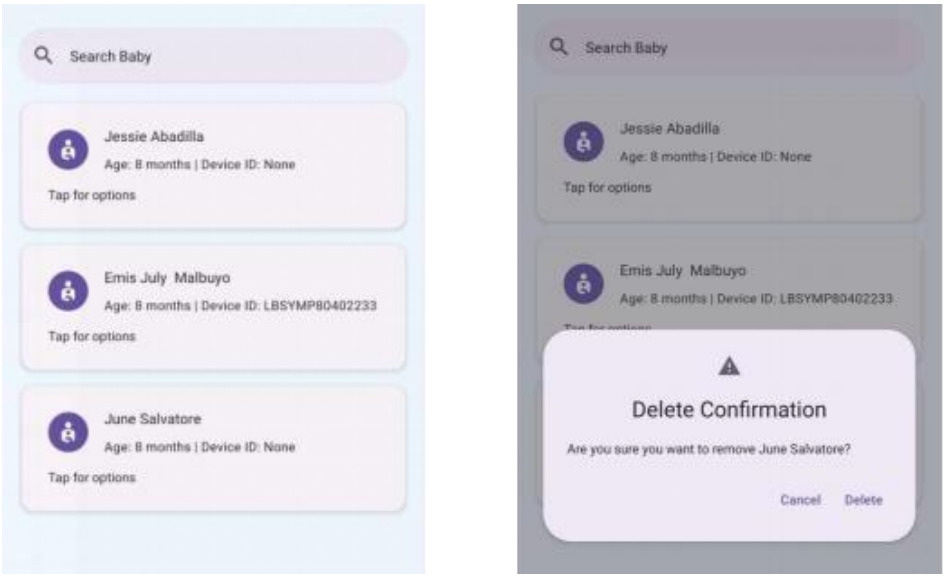

Figure 12: Delete Baby UI

Figure 12 represents the feature that allows users to delete a specific baby. In addition to viewing this information, users can delete the baby's profile through a marked "Delete" Button or icon. This action may prompt a confirmation popup to prevent accidental deletions and ensure users know the consequences before finalizing the action. This confirmation step typically includes a message such as, "Are you sure you want to remove (name of the baby)?"

Sprint 2.a.10: Design Profile Settings UI

Figure 13 shows the profile settings where users can update their information.

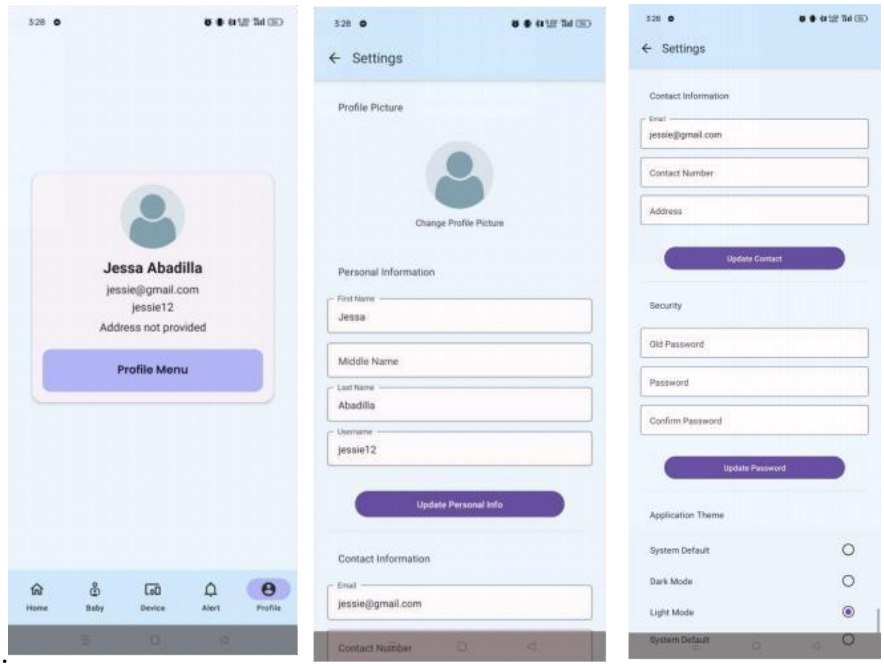

Figure 13: Profile Settings

Figure 13 showcases a feature that allows users to update their mother's or user's information. By selecting the relevant profile, users are directed to an editable view displaying personal details such as name, contact information, and security settings.

Scrolling down within this section reveals additional fields where users can update their contact information, such as phone numbers and email addresses, ensuring accurate and up-todate communication details. The security settings area allows users to manage password updates or adjust security preferences for enhanced account protection.

Additionally, users will find an application theme section, enabling interface customization. Here, users can choose from:

- “System Default": Aligns the app's appearance with the device's default theme.
- “Dark Mode": A darker color scheme for reduced eye strain, ideal for low-light settings.
- “Light Mode": A brighter color scheme suited for well-lit environments.

Sprint 2.a.11: Design Device Sharing UI

Figure 14 presents a feature that allows users to share a device with others for receiving notifications. Upon selection, the option to share the device with specific users will be displayed, enabling them to receive relevant alerts.

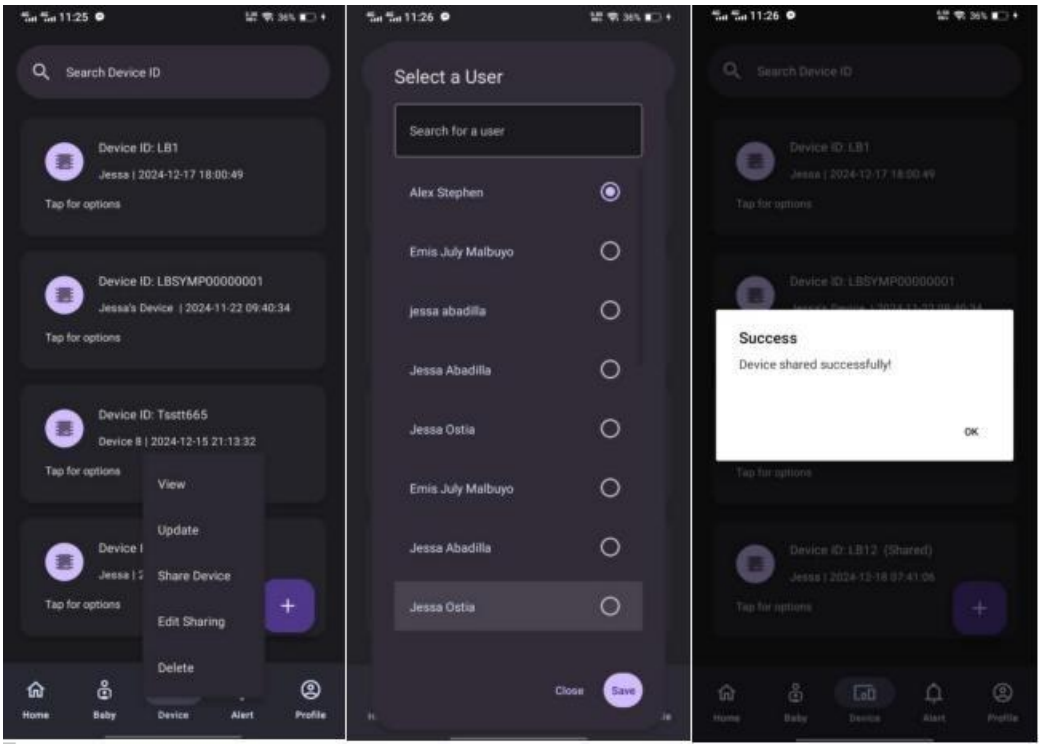

Figure 14: Device Sharing

Figure 14 presents a feature allowing users to share a device to receive notifications. The option to share the device with specific users will be displayed upon selection, enabling them to receive relevant alerts. The shared device can only acknowledge the notification.

Sprint 2.a.12: Design "acknowledge" UI

Figure 15 presents a feature for acknowledging a notification that indicates that our system detects and recognizes a baby cry.

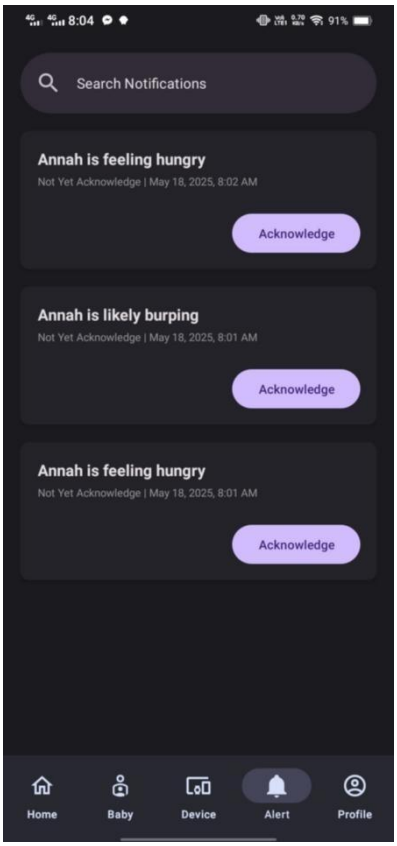

Figure 15: Design "acknowledge" UI

Sprint 2.a.13: Design add , view, delete note UI

Figure 16 demonstrates the add and view note feature integrated into our monitoring system. This functionality provides users with a convenient way to add personalized notes directly related to specific notifications. For instance, if a notification alerts a mother about a baby's temperature fluctuation, she can add a note detailing actions taken, observations, or reminders for follow-up. This feature allows users to annotate critical events, providing context and a personal record that enhances the utility and comprehensiveness of the monitoring system. It transforms a simple alert into a trackable event with user-added insights.

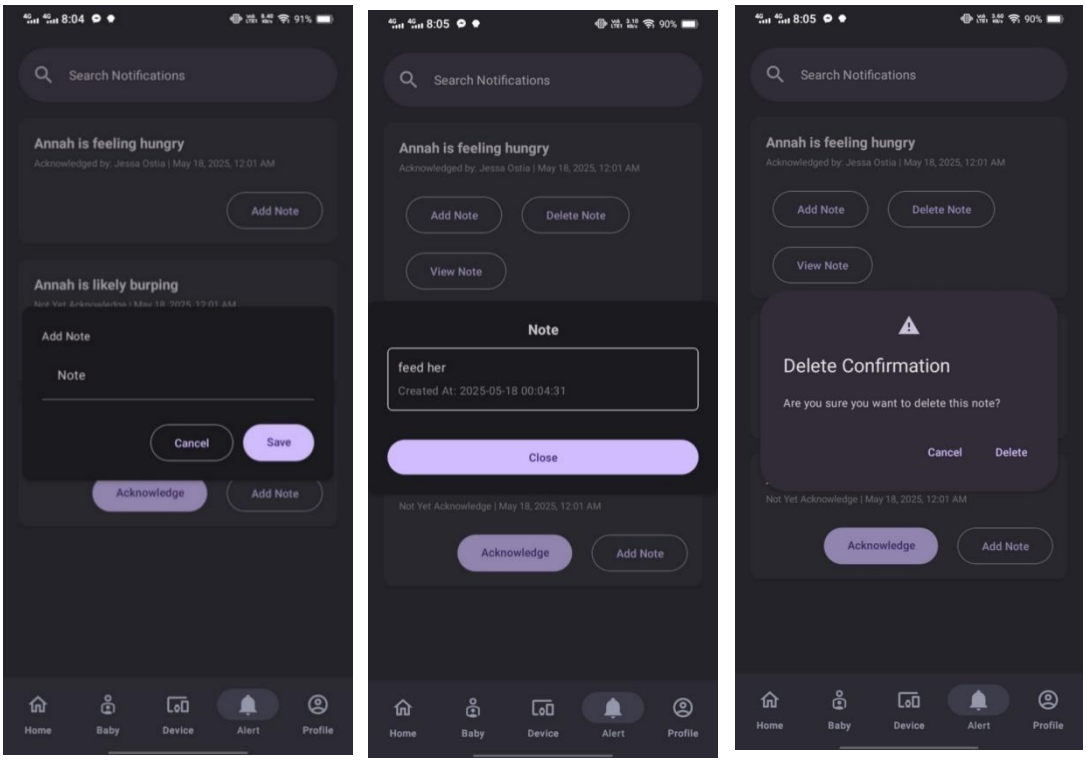

Figure 16: Design Add, View, Delete Note UI

Product 3: Database Design

Sprint 3.a.1: Design ERD

The entity-relationship (ER) database model used in this study outlines the database structure for the proposed system. The design of the entities and their attributes was informed by data collected from interviews and previous research. This database model ensures that all essential fields and parameters are properly represented and stored, supporting the functional requirements of the system.

Figure 17: ERD

Figure 17 is the Entity-Relationship Diagram (ERD) that shows three main entities: baby, users, and devices, with a clear focus on database normalization and referential integrity. The baby table represents children in the system and includes attributes like id (primary key), descriptive fields (nickname, gender, months), and foreign keys (userid and device_id) to establish relationships with the users and devices tables. A one-to-one relationship exists between the baby and devices tables, where each baby is associated with a unique device for tracking or monitoring. Additionally, there is a many-to-one relationship between the baby and users' tables, indicating that multiple babies can belong to a single user, such as a parent or guardian. The users table stores user information, including credentials (username, password), contact details, and profile metadata. The devices table contains device-specific information like a unique id, creation timestamp, and an optional foreign key (userid) to indicate device ownership. Overall, the diagram effectively enforces referential integrity through foreign keys, ensuring consistency and flexibility. While the design is normalized and well-structured, improvements like

explicitly marking cardinalities on relationship lines or optimizing field sizes for real-world data could enhance the schema further.

Sprint 3.a.2: Convert ERD into physical database

Figure 18 displays the physical database structure for the system, as shown in phpMyAdmin. This structure organizes data that supports various system functionalities, ensuring effective data management and operations.

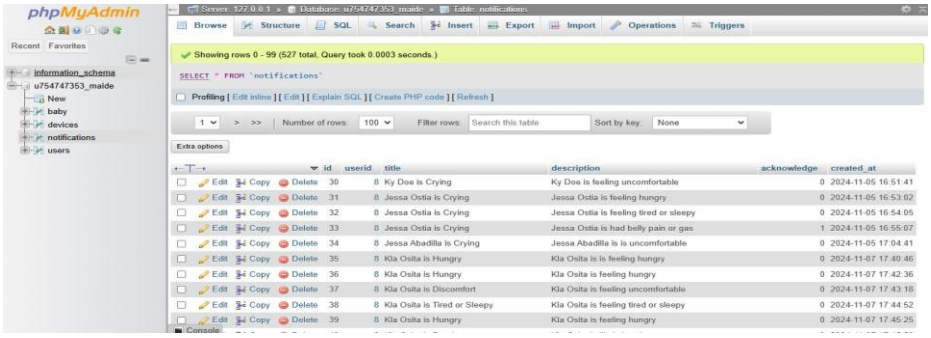

Figure 18: Physical Database

Figure 18 displays the physical database structure for the system, shown in phpMyAdmin. The database, named u754747353_jessakyla, consists of 3 tables, including baby, devices, notifications, user tables.

Product 4: Configure IoT Device

Sprint 4.a.1: Design Prototype

Figure 19 shows the hardware prototype of the MAide system, designed to recognize baby cries in real-world conditions. The system utilizes Teachable Machine Learning for classification. Once classified, the system transmits the data to a mobile app, providing mothers with real-time notifications. Integrated with IoT technology, the hardware prototype ensures accurate and timely baby cry recognition, enhancing infant care and parental responsiveness.

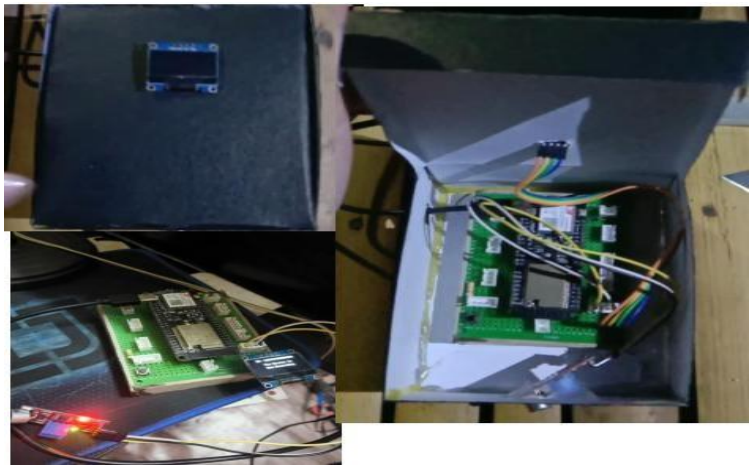

Figure 19: Hardware Prototype

The image above showcases the hardware prototype of the MAide system, designed to recognize and process baby cries. The setup includes a microcontroller, likely an ESP32, mounted on a prototyping board, which acts as the central processing unit. It is connected to an OLED display that provides real-time feedback or system status updates. Additionally, there is a microphone module with indicator LEDs, used to capture audio input for analysis. The wiring demonstrates integration of various components, ensuring communication between sensors, the microcontroller, and output devices. This prototype is compact, making it suitable for testing the system's functionality in practical scenarios.

Product 5: Backend Development

Sprint 5.a.1: Design and implement baby cry recognition through notification management

Figure 20 illustrates a feature that allows users to view notifications identifying the meaning behind a baby's cry

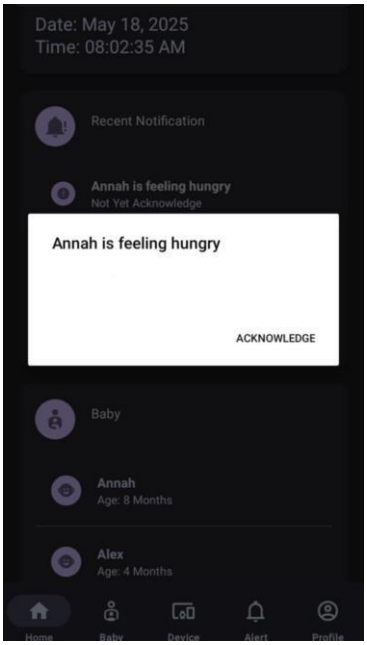

Figure 20: Baby Cry Notification

Figure 20 showcases the feature that is designed to provide parents or caregivers with timely and informative alerts to help them respond promptly to the baby's needs.

The notification system analyzes the baby's cry and categorizes it into specific types, such as hunger, discomfort, sleepiness, or pain. When a cry is detected, users receive a real-time notification that includes a brief description, such as "(Name of the baby) is feeling uncomfortable" or "(Name of the baby) is experiencing belly pain". This helps users understand the potential reason behind the baby's crying without delay.

Sprint 5.a.2: Design and account management function

Figure 21 shows the account management feature, which was designed and implemented to allow users to personalize and maintain their account settings. Users can use this interface to change their profile picture, update personal information such as their first, middle, and last names, and change their username and contact information, which includes their email address and phone number.

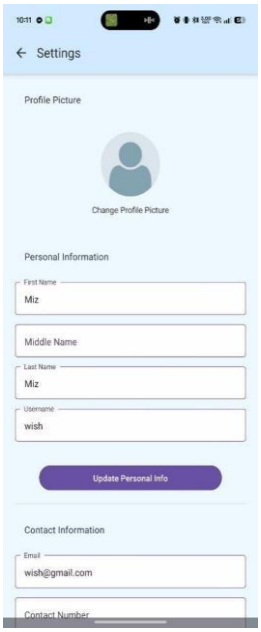

Figure 21: Account management

Sprint 5.a.3: Design and implement device management function

Figure 22 shows the feature that was carefully designed and implemented to give users complete control over their registered devices. This interface allows users to manage the devices they have added to the system, assign a specific baby to a specific device.

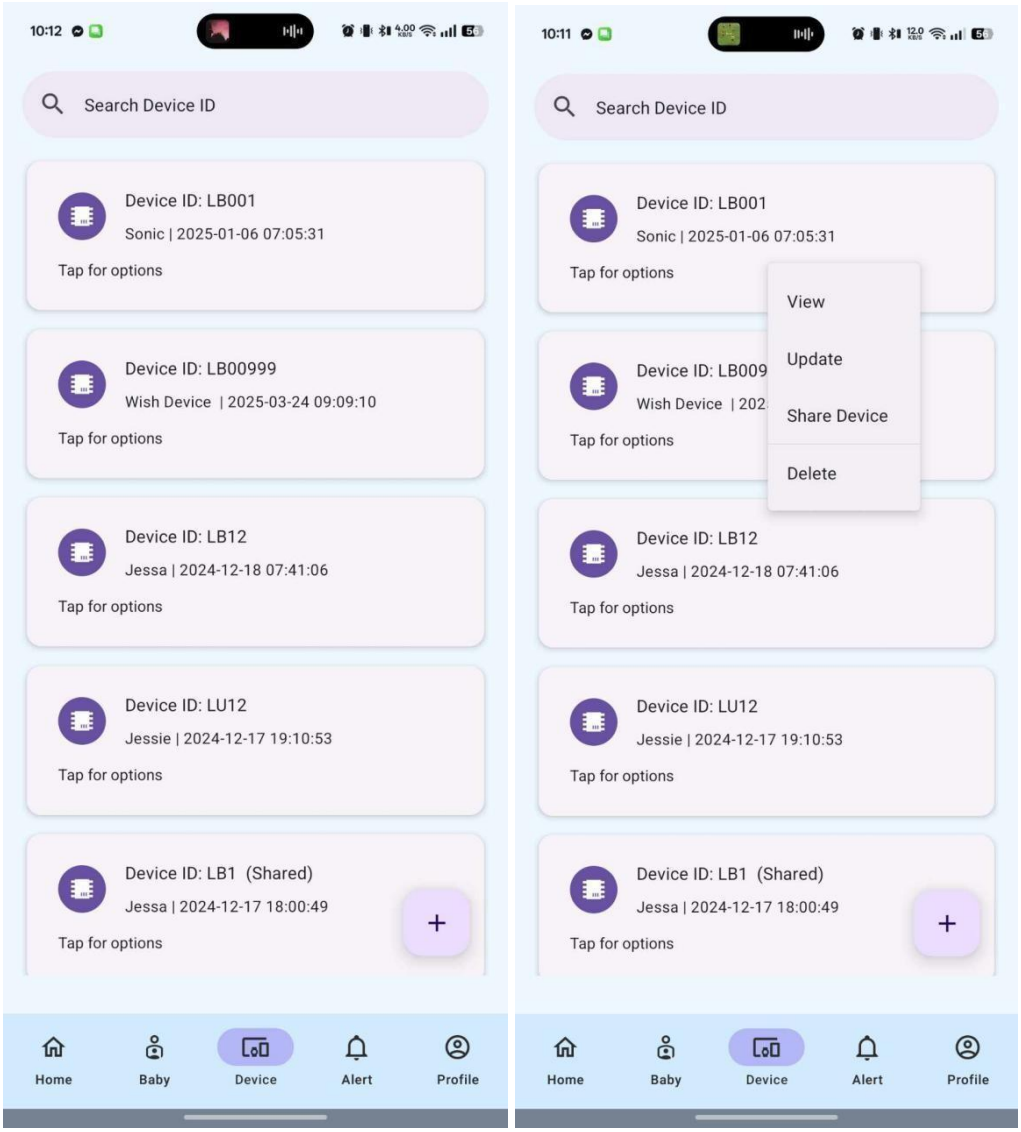

Figure 22: Device management

Sprint 5.a.4: Design and implement baby management function

Figure 23 depicts a feature that was specifically designed and implemented to assist users, particularly mothers—in managing the information about the babies they have added to the system. This feature allows users to quickly view detailed baby profiles, update or correct any existing information, delete entries as needed, and assign a specific device to a specific baby for more accurate monitoring and data tracking. This allows for personalized and organized baby management within the application.

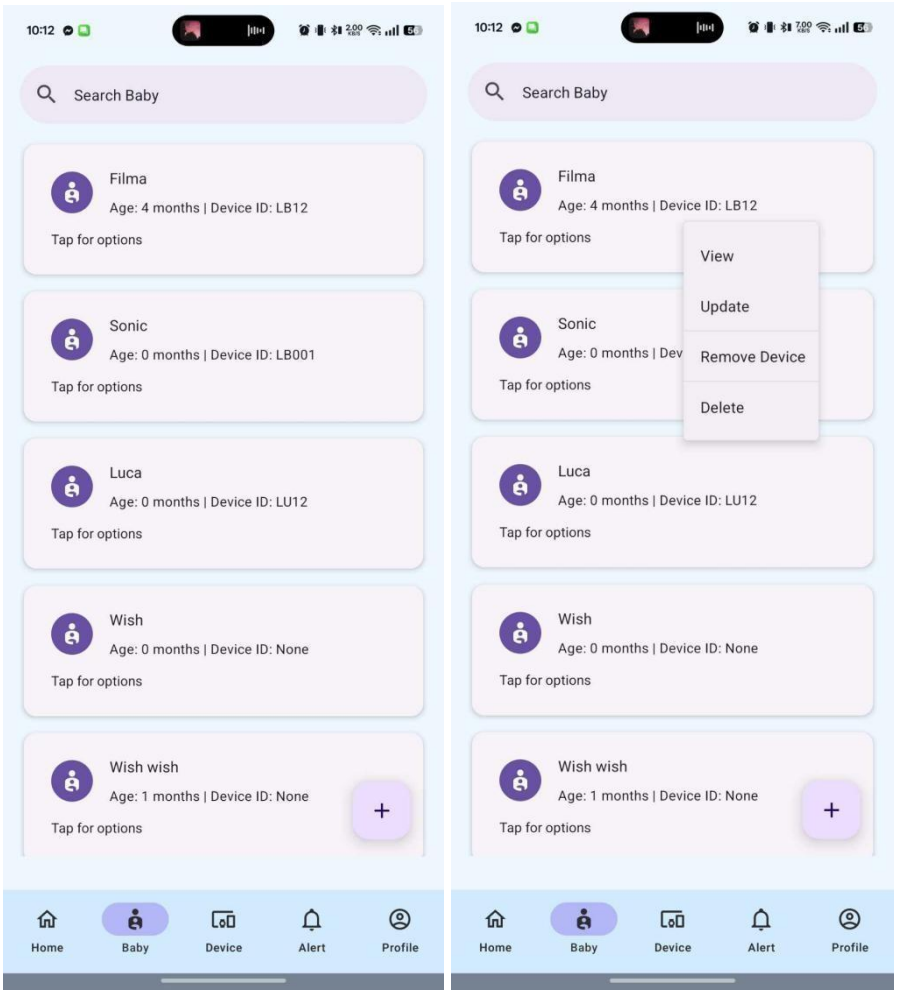

Figure 23: Baby management

System Requirements

The System Requirements section details the critical hardware and software specifications essential for the successful development, deployment, and operation of the system. This section provides a structured overview of the components and tools required for both development and user implementation, ensuring functionality, compatibility, and reliability across diverse platforms. By defining these requirements, this section establishes a foundational infrastructure, guiding the development process and optimizing user experience throughout the system's operational lifecycle.

Development Requirements

The Development Requirements outline the essential hardware and software components necessary to build and operate the system effectively. This section specifies each component and tool, creating a foundation for understanding the system's functional capabilities and supporting infrastructure. Hardware requirements include critical devices for monitoring and control, while software requirements cover programming environments, IoT platforms, and databases essential for data processing and connectivity. Together, these elements ensure reliable and efficient system performance, supporting the achievement of its intended operational goals.

Figure 24: ESP 32

The ESP32, created by Espressif Systems, is a low-cost System on a Chip (SoC) renowned for its power efficiency. It incorporates Wi -Fi and Bluetooth capabilities, including Bluetooth Low Energy (BLE), making it the perfect choice for IoT apps that require reliable , low - power wireless connectivity. The ESP32 in the MAide system interprets audio signals that are picked up by the microphone and sends the information over Wi-Fi to the machine learning model. It is essential to the operation of the system because of its smooth connectivity, which guarantees real-time cry analysis and immediate notifications.

Figure 25: LCD

An LCD (Liquid Crystal Display) device is commonly used in electronic projects to display data, such as text or numbers. It has a simple design with a rectangular screen and pin connectors, facilitating integration with microcontrollers and other devices. The LCD in the MAide system shows the device's code, making it simple for users to connect their mobile application to the system. This guarantees smooth communication between the user's app and the IoT device, allowing for data synchronization and real-time notifications.

Figure 26: Sound Sensor

An audio sensor is frequently used in electronic projects to identify and quantify sound levels. It has a microphone or piezoelectric component that transforms sounds into electrical signals, which can be manipulated by microcontrollers or other devices. A sound sensor, with its simple design and easy integration via pin connectors, is perfect for applications requiring audio detection, such as voice or noise recognition systems. The audio sensor, which records the baby cries in real time, is an essential component of the MAide system. The ESP32 microcontroller processes the electrical

impulses that the sensor transforms from crying sounds. The machine learning algorithm analyzes these signals to categorize the scream into distinct groups, such as tiredness, discomfort, or hunger. This part is necessary to guarantee precise audio detection, which allows the system to identify and react to the baby's requirements quickly.

Hardware Requirements

The hardware requirements section outlines the key components necessary to support the system's functionality, focusing on the ESP32 microcontroller and an audio sensor for detecting baby cries. The ESP32 microcontroller is integral to the system, providing reliable IoT connectivity for data processing and communication. The audio sensor, designed to detect and recognize baby cries, serves as the system's primary input device for sound recognition. Together, these components form the core infrastructure of the system, enabling efficient sound detection and processing. Figures 18 and 19 provide a detailed overview of these critical hardware elements and their roles in achieving the project’s objectives.

Software Requirements

The application must be compatible with devices running Android versions 7 to 14. This requirement ensures that the application can leverage the latest features, security enhancements, and performance improvements provided by these operating systems. By focusing on these versions, the application is positioned to deliver an optimal user experience, enabling users to interact with the system seamlessly across a diverse range of modern smartphones. The mobile application must be compatible with Android, ensuring it takes advantage of the latest features, security updates, and performance enhancements offered by these platforms. By focusing on responsive design and compatibility with popular devices, the application is optimized to provide a smooth and consistent user experience across smartphones and tablets. This approach ensures that users can interact with the system effectively, regardless of their device.

Implementation Requirements

The Implementation Requirements specify the essential smartphone specifications and capabilities to access the system effectively through a mobile application. Ensuring compatibility and functionality across various smartphone models is crucial for providing a seamless user experience.

This section outlines the necessary hardware and software to enable efficient access to the system's features and real-time functionality updates. Meeting these requirements ensures the mobile application operates at optimal performance levels, enhancing usability and user engagement.

RESULTS AND DISCUSSIONS

Many parents rely on traditional methods, such as observation and guesswork, to understand the reasons behind their baby's cries. These methods can be stressful and timeconsuming, often leading to delayed responses and increased anxiety for caregivers. While advancements in technology have improved various aspects of infant care, there is still a gap in real-time tools that assist parents in decoding their baby's needs efficiently.

Researchers created the MAide system to address these challenges, which uses audio recognition and analysis to interpret children 's cries and provides real-time alerts about possible interpretations. From a labeled dataset, the system can identify infant cries with an audio sensor and categorize various types of cries, such as hunger, discomfort, sleepiness, burping, and abdominal pain. Additionally, mothers can efficiently register their devices and receive immediate alerts, ensuring timely responses to their needs. Additionally, the system allows shared devices for collaborative monitoring, allowing multiple caregivers to stay on top of each other.

In addition, a notification alert system was implemented to immediately inform parents or guardians when a child's crying is detected. Users can also add, edit, and delete notes in notifications for better healthcare management. The system was evaluated on accuracy, responsiveness, and user satisfaction, and feedback from respondents, which included 19 mothers, seven computer experts, and one computer expert, indicated that MAide significantly improved their ability to respond promptly to their children's needs. The evaluation findings showed that the system effectively achieved its goals by providing a reliable, effective, and user-friendly tool for childcare.

Detection, Recognition, Classification of Baby Cries

Figure 27 presents the "Recognizing Baby Cry" feature, designed to assist mothers in understanding the meanings behind their baby's cries.

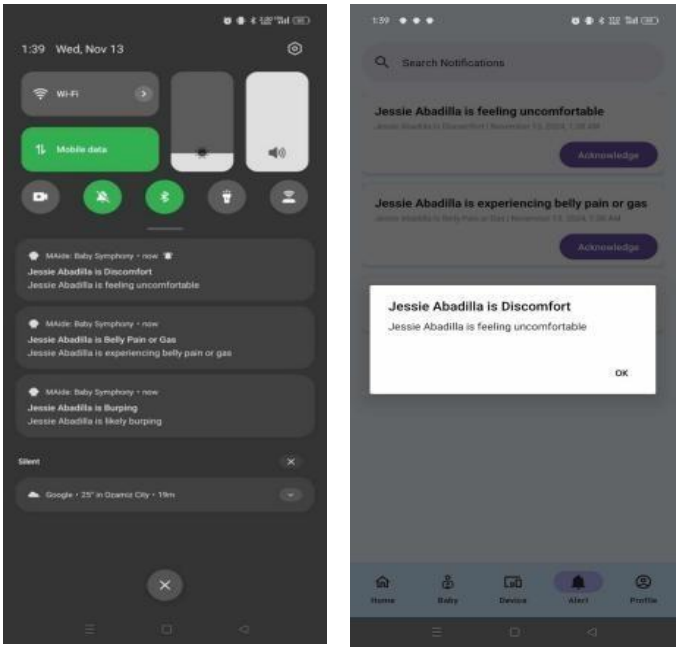

Figure 27: Baby Cry Recognition

Figure 27 illustrates the notification system for the "Recognizing Baby Cry" feature within the MAide app. The left screenshot shows the phone's notification panel, where real-time notifications from the app are displayed. These notifications inform the caregiver about various interpretations of the baby's cry, such as "Jessie Abadilla is feeling uncomfortable," "Jessie Abadilla is experiencing belly pain or gas," "Jessie Abadilla is burping," and "Jessie Abadilla is likely hungry." This lets caregivers quickly understand their baby's needs without opening the app, enabling prompt attention and action. The exemplary screenshot shows the "Notifications" section within the app, where a specific notification is expanded in a popup that states, "Jessie Abadilla is feeling uncomfortable," accompanied by an "OK" button for acknowledgement. This feature provides caregivers with more detailed information and allows them to interact with or dismiss notifications as necessary.

System Evaluation Results

Evaluating the system is essential to ensure that it meets its intended objectives. Researchers must assess the system's performance, focusing on functionality, usability, and reliability, before it is officially launched. The system assessment was conducted with a total of 27 respondents, consisting of 19 mothers and caregivers, 65.6%, 7 IT Students which is 31.2%, and 1 IT Professional which is 3.1% out of the total number of respondents.

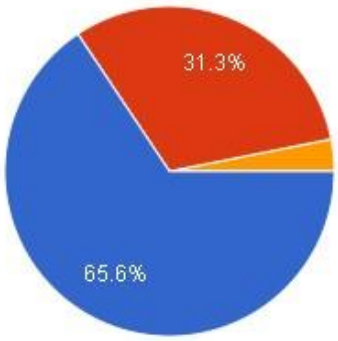

Figure 28: Respondents' Percentage

Functionality Assessment

Functionality refers to the ability of the system to produce precise outcomes and execute its designated operations. Table 2 presents the results of the evaluation.

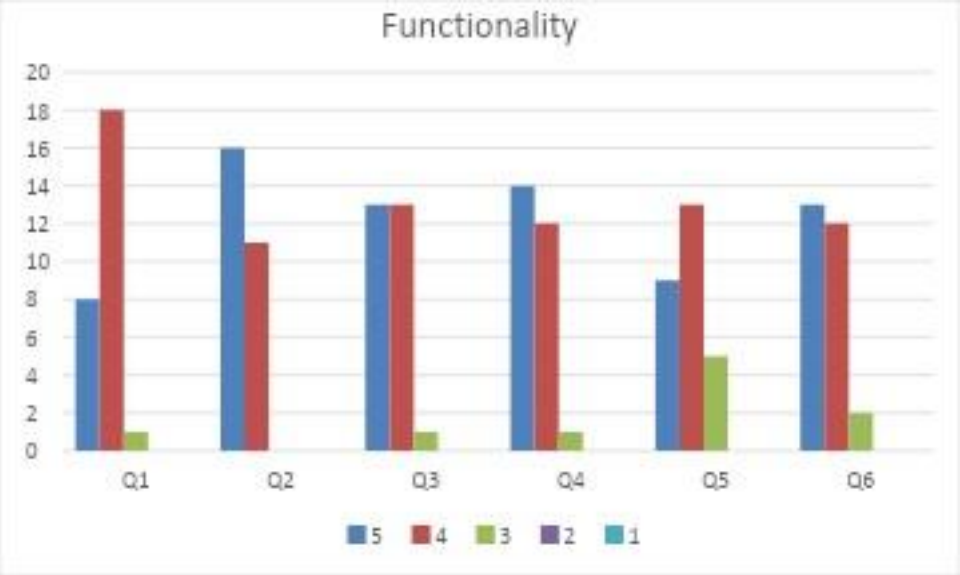

Figure 29: Bar Graph of User Responses in the Functionality Assessment

Figure 29 presents a bar chart depicting users' responses to the system's functionality assessment. For Q1, most respondents (18) rated the system positively, 8 gave moderate scores, and only 1 gave a low rating. In Q2, 16 users gave favorable feedback, with 11 offering moderate ratings. Q3 saw an even split between positive and moderate responses, with 13 votes and one low rating. Q4 followed a similar pattern with 14 positive, 12 moderate, and one low rating. Q5 showed more variation, with nine positive responses, 13 moderates, and 5 low ratings, indicating some disparity in users' views. Lastly, Q6 received 13 positive ratings, 12 moderate, and two low responses. Overall, the data suggests that while most users rated the system positively or moderately across the questions, there were very few low ratings, indicating a generally favorable perception of the system's functionality.

Table 2: Functionality Assessment Result

Statement	Mean	Interpretation
The mobile app allows secure login with a username and password.	3.9	Functional
Users can register their baby's name in the mobile application for a personalized experience.	4.19	Functional

Users can easily connect devices to their baby's profile for realtime monitoring of the baby's cries.	4.19	Functional
Users receive instant notifications when their baby is crying, allowing for timely responses.	4.16	Functional
Users can search for their registered baby's name quickly and easily.	4.22	Very Functional
Users can also view a history of notifications related to the baby's cries, providing guardians with an overview of past alerts.	4.29	Very Functional
The application provides accurate analysis of the baby's cries, helping caregivers understand their needs better.	4.19	Functional
Average Weighted Mean	4.16	Functional

The overall mean of 4.16, interpreted as Functional, indicates that users generally believe the mobile application is effective and reliable in carrying out its intended tasks. This suggests that the app's core features meet user expectations and function properly, resulting in a positive user experience. Among all the features evaluated, the highest mean is 4.29, which corresponds to the statement, "Users can also view a history of notifications related to the baby's cries, providing guardians with an overview of past alerts." This was interpreted as Very Functional and most likely received the highest score because users highly value the ability to review previous alerts. This feature provides additional convenience and insight, allowing caregivers to track cry patterns and respond more effectively, thereby improving the caregiving experience.

On the other hand, the feature with the lowest mean is 3.9, which is still interpreted as Functional, and refers to the statement, "The mobile app allows secure login with a username and password." Although this function is critical for privacy and security, it may not be perceived as highly impactful in comparison to baby monitoring-related features. The slightly lower rating may reflect user concerns about ease of access, login speed, or a preference for alternative authentication methods. Overall, while all features are rated positively, the differences in mean scores show which aspects of the app users value the most and which could benefit from minor improvements.

Usability Assessment

Usability refers to the proficiency in which a particular user can utilize the system. Table 3 shows the results of the evaluation.

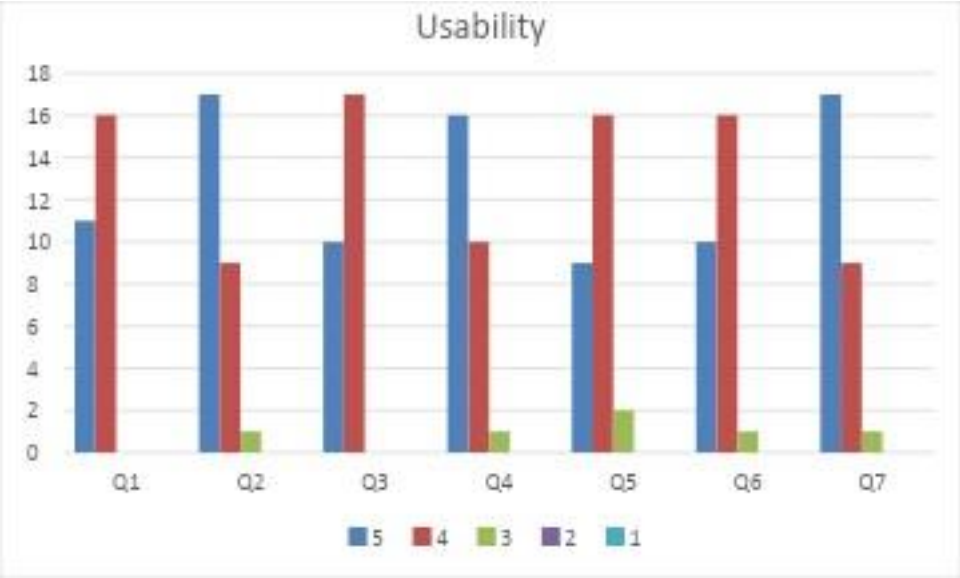

Figure 30: Bar Graph of User Responses in the Usability Assessment

Table 3: Usability Assessment Result

Statement	Mean	Interpretation
The system's interface is intuitive and easy to navigate, requiring minimal training for first-time users.	4.06	Usable
The process of adding devices to associate with the baby's profile is straightforward, promoting seamless connectivity for real-time monitoring.	4.03	Usable
Instant notifications alert users when their baby is crying, ensuring timely responses and fostering a user-friendly experience.	4.06	Usable
The application responds promptly to user actions, contributing to a seamless and efficient user experience.	4.29	Very Usable
The application allows users to view a history of notifications related to the baby's cries, which enhances usability by providing guardians with an overview of past alerts and helping them monitor patterns over time.	4.29	Very Usable

Overall, the application provides accurate analysis of the baby's cries, helping caregivers understand their needs better and making the experience more effective.	4.35	Very Usable
Average Weighted Mean	4.18	Usable

The analysis shows that users generally find the app "Usable," with an average score of 4.18. Key features such as intuitive interface, easy baby registration, straightforward device addition, and prompt notifications contribute to a positive user experience. Users particularly valued the app's quick response time and the ability to view a history of notifications, rated as "Very Usable." Features like error prompt for incorrect inputs and accurate analysis of baby cries enhance usability and effectiveness. Overall, the app is seen as efficient and supportive in helping caregivers monitor and understand their baby's needs.

Reliability Assessment

Reliability refers to the capabilities of the system to handle system errors and the ability to perform its required functions under conditions for accuracy. Table 4 shows the results of the evaluation.

Table 4: Reliability Assessment Result

Statement	Mean	Interpretation
The system consistently captures and analyzes the baby's cries with accuracy, ensuring reliable data input for effective monitoring and responsive caregiving.	4.16	Reliable
The text within the application is clear, easy to read, and understandable for all users.	4.03	Reliable
Instructions provided by the system are straightforward and easy to follow, enhancing user experience.	4.16	Very Reliable
Buttons and options on the screen are easily accessible and intuitive to use.	4.22	Very Reliable
The color scheme and design of the application are comfortable for long periods of use, reducing eye strain.	4.32	Very Reliable

The system provides clear feedback during the image upload and cry analysis processes, guiding users with messages such as "Your audio is being processed, please wait," which enhances the user experience and minimizes confusion.	4.25	Very Reliable
Users can easily understand how to interact with the application, making it user-friendly for all caregivers.	4.19	Reliable
The system is user-friendly for everyone (mothers, guardians, and caregivers).	4.19	Reliable
Average Weighted Mean	4.19	Reliable

The analysis indicates that users find the system "Reliable" overall, with an average score of 4.19. Key features such as accurate cry analysis, clear and readable Text, straightforward instructions, and intuitive buttons contribute to a positive user experience. The app's color scheme and design, which minimize eye strain and provide clear feedback messages during processes, were rated especially highly as "Very Reliable." Additionally, users appreciated the app's ease of use, making it accessible to various caregivers. Overall, the system effectively supports reliable and user-friendly caregiving.

CONCLUSIONS AND RECOMMENDATIONS

Detecting and interpreting baby cries remains a challenging task for parents, often leading to uncertainty and stress when trying to determine the cause of their infant's discomfort. Traditional methods of gauging a baby's needs rely heavily on parental intuition and experience, sometimes resulting in delayed responses or missed signs of underlying issues. To address this, there is a growing demand for a more effective and automated solution to help parents understand their baby's cries more accurately.

In response to this need, the researchers developed a mobile-based application, MAide, to recognize and analyze baby cries in real-time. The system detects baby cries using an audio sensor. It classifies them, such as hunger, burping, discomfort, belly pain, and sleepiness, based on a labeled dataset and the principles of the Dunstan Baby Language. The app provides real-time notifications to parents, ensuring a timely and informed response. This aligns with the study's objectives, particularly detecting, recognizing, and classifying different baby cries to improve infant care.

The MAide system allows mothers to register their devices easily and share the device with others for collaborative monitoring, ensuring that multiple caregivers stay informed. Integrating an alert notification system ensures that parents and guardians receive immediate updates when a baby's cry is detected. Additionally, the system allows users to add, edit, and remove notes on notifications, enhancing caregiving flexibility. These features fulfill the study's objectives of realtime alert notifications and efficient user interaction. The system was tested and assessed based on accuracy, responsiveness, and user satisfaction to evaluate its performance. Results demonstrated that MAide significantly improved caregivers' ability to recognize and respond to baby cries, validating the objective of evaluating the system's effectiveness. The user-friendly interface and comprehensive user guide further ensured ease of use, meeting the need for an accessible and intuitive caregiving tool.

The system was tested and assessed based on accuracy, responsiveness, and user satisfaction to evaluate its performance. Results demonstrated that MAide significantly improved caregivers' ability to

recognize and respond to baby cries, validating the objective of evaluating the system's effectiveness. The user-friendly interface and comprehensive user guide further ensured ease of use, meeting the need for an accessible and intuitive caregiving tool.

Recognizing Multiple Cry Signals

A key recommendation for this study is to integrate a feature that can identify multiple underlying causes in a single baby cry. For instance, the app could recognize when a baby is uncomfortable due to hunger or when the discomfort is caused by belly gas. Enhancing the app's ability to detect and differentiate between these combined signals would provide more accurate notifications to caregivers. This improvement would enable quicker, more precise responses, ultimately supporting better infant care and more efficient caregiving.

REFERENCES

Alam, H., Burhan, M., Gillani, A., Haq, I. U., Arshed, M. A., Shafi, M., & Ahmad, S. (2023). IoT Based Smart Baby Monitoring System with Emotion Recognition Using Machine Learning. *Wireless Communications and Mobile Computing, 2023.* https://doi.org/10.1155/2023/1175450

Bouchet, H., Plat, A., Levréro, F., Reby, D., Patural, H., & Mathevon, N. (2020). Baby cry recognition is independent of motherhood but improved by experience and exposure. *Proceedings of the Royal Society B: Biological Sciences, 287*(1921). https://doi.org/10.1098/rspb.2019.2499

Bratan, C. A., Gheorghe, M., Ispas, I., Franti, E., Dascalu, M., Stoicescu, S. M., ... & Nastase, L. (2021, October). Dunstan baby language classification with CNN. In *2021 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)* (pp. 167– 171). IEEE.

Chang, C. M., Chen, H. Y., Chen, H. C., & Lee, C. C. (2020, December). Sensing with contexts: Crying reason classification for infant care center with environmental fusion. In *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)* (pp. 314–318). IEEE.

Chang, C. Y., & Tsai, L. Y. (2019). A CNN-based method for infant cry detection and recognition. In *Web, Artificial Intelligence and Network Applications: Proceedings of the Workshops of the 33rd International Conference on Advanced Information Networking and Applications (WAINA-2019) 33* (pp. 786–792). Springer International Publishing.

Cohen, R., Ruinskiy, D., Zickfeld, J., IJzerman, H., & Lavner, Y. (2020). Baby cry detection: deep learning and classical approaches. *Development and analysis...* [Incomplete reference]

Dewi, S. P., Prasasti, A. L., & Irawan, B. (2019, July). The study of baby crying analysis using MFCC and LFCC in different classification methods. In *2019 IEEE International Conference on Signals and Systems (ICSigSys)* (pp. 18–23). IEEE.

Dewi, S. P., Prasasti, A. L., & Irawan, B. (2019, November). Analysis of LFCC feature extraction in baby crying classification using KNN. In *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)* (pp. 86–91). IEEE.

Felipe, G. Z., Aguiar, R. L., Costa, Y. M., Silla, C. N., Brahnam, S., Nanni, L., & McMurtrey, S. (2019, June). Identification of infants' cry motivation using spectrograms. In *2019 International Conference on Systems, Signals and Image Processing (IWSSIP)* (pp. 181–186). IEEE.

George, A. H., & George, A. S. (2023). Decoding Infant Communication: Understanding the Meaning Behind Baby's Cries. *Partners Universal Innovative Research Publication, 1*(1), 1–14.

Hovan George, A. S., & George, A. S. (2023). Decoding Infant Communication: Understanding the Meaning Behind Baby's Cries. https://doi.org/10.5281/zenodo.8432700

Ji, C., Basodi, S., Xiao, X., & Pan, Y. (2020, September). Infant sound classification on multistage CNNs with hybrid features and prior knowledge. In *International Conference on AI and Mobile Services* (pp. 3–16). Springer International Publishing.

Liang, Y. C., Wijaya, I., Yang, M. T., Cuevas Juarez, J. R., & Chang, H. T. (2022). Deep Learning for Infant Cry Recognition. *International Journal of Environmental Research and Public Health, 19*(10). https://doi.org/10.3390/ijerph19106311

Liu, L., Li, W., Wu, X., & Zhou, B. X. (2019). Infant cry language analysis and recognition: an experimental approach. *IEEE/CAA Journal of Automatica Sinica, 6*(3), 778–788.

Mahmoud, A. M., Swilem, S. M., Alqarni, A. S., & Haron, F. (2020, December). Infant cry classification using semi-supervised k-nearest neighbor approach. In *2020 13th International Conference on Developments in eSystems Engineering (DeSE)* (pp. 305– 310). IEEE.

Manikanta, K., Soman, K. P., & Sabarimalai Manikandan, M. (2019, December 1). Deep Learning Based Effective Baby Crying Recognition Method under Indoor Background Sound Environments. In *CSITSS 2019 - 2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution, Proceedings.* https://doi.org/10.1109/CSITSS47250.2019.9031058

Maghfira, T. N., Basaruddin, T., & Krisnadhi, A. (2020). Infant cry classification using CNN -

RNN. *Journal of Physics: Conference Series, 1528*(1). https://doi.org/10.1088/17426596/1528/1/012019

Novamizanti, L., Prasasti, A. L., & Utama, B. S. (2020, December). Study of linear discriminant analysis to identify baby cry based on DWT and MFCC. In *IOP Conference Series: Materials Science and Engineering, 982*(1), 012009. IOP Publishing.

Prasasti, A. L., Novamizanti, L., & Razik, M. I. (2019, November). Identification of baby cry with discrete wavelet transform, mel frequency cepstral coefficient and principal component analysis. In *Journal of Physics: Conference Series, 1367*(1), 012061. IOP Publishing.

Sharma, A., & Malhotra, D. (2020, August). Speech recognition based IICC-intelligent infant cry classifier. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)* (pp. 992–998). IEEE.

Sriraam, N., Tejaswini, S., & Chavan, A. A. (2020). Development of Portable Medical Electronic Device for Infant Cry Recognition: A Primitive Experimental Study. In *Data Analytics in Medicine: Concepts, Methodologies, Tools, and Applications* (pp. 288– 297). IGI Global.

Teeravajanadet, K., Siwilai, N., Thanaselanggul, K., Ponsiricharoenphan, N., Tungjitkusolmun, S., & Phasukkit, P. (2019, November). An infant cry recognition based on convolutional neural network method. In *2019 12th Biomedical Engineering International Conference (BMEiCON)* (pp. 1–4). IEEE.

Tuduce, R. I., Rusu, M. S., Horia, C. U. C. U., & Burileanu, C. (2019, July). Automated baby cry classification on a hospital-acquired baby cry database. In *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)* (pp. 343–346). IEEE.

Witteman, J., Van IJzendoorn, M. H., Rilling, J. K., Bos, P. A., Schiller, N. O., & BakermansKranenburg, M. J. (2019). Towards a neural model of infant cry perception. *Neuroscience & Biobehavioral Reviews, 99*, 23–32.

Yasin, S., Draz, U., Ali, T., Shahid, K., Abid, A., Bibi, R., Irfan, M., Huneif, M. A., Almedhesh, S. A., Alqahtani, S. M., Abdulwahab, A., Alzahrani, M. J., Alshehri, D. B., Abdullah, A. A., & Rahman, S. (2022). Automated Speech Recognition System to Detect Babies' Feelings through Feature Analysis. *Computers, Materials and Continua, 73*(2), 4349–

4367. https://doi.org/10.32604/cmc.2022.028251

Yu, X., Wang, L., Zhao, X., Lu, C., Long, X., & Chen, W. (2019, November). An investigation into audio features and DTW algorithms for infant cry classification. In *Proceedings of the 2019 6th International Conference on Biomedical and Bioinformatics Engineering* (pp. 54–59).

APPENDICES

Appendix A: Survey Letter

MAide: A Mother's Aide in Recognizing and Analyzing Baby Cries using Deep Learning

Greetings!

Good day, Ma'am/Sir. Under the Bachelor of Science in Information Technology domain, we, the students of Misamis University, would like to ask permission to evaluate the functionality, reliability, and usability issues on our developed web application entitled "**MAide: A Mother's Aide in Recognizing and Analyzing Baby Cries using Deep Learning**
" Utilizing and answering this survey, we will gain knowledge regarding the effectiveness of our system.

This is in partial fulfillment of the requirements for the degree, Bachelor of Science in Information Technology.

We understand that we are carrying on huge responsibilities. Please remember that all information provided will be kept confidential and only used for academic purposes. We hope that this request will merit your approval. Your participation in this study is highly appreciated. Thank you very much, and God bless!

Sincerely yours,
The Researchers

Appendix B: System Evaluation Instrument

Directions. Please rate the system's functionality, reliability, and usability using the rating scale below. You should check (✓) the number that best represents your system evaluation for each statement.

Functionality Assessment of the User's Side

Scale	Range	Responses	Interpretation
5	4.24 – 5.00	Strongly Agree	Very Functional
4	3.43 – 4.23	Agree	Somewhat Functional
3	2.62 – 3.42	Neutral	Functional
2	1.81 - 2.61	Disagree	Less Functional
1	1.00 - 1.80	Strongly Disagree	Not Functional

Statement	5	4	3	2	1
The system efficiently captures and analyzes the baby's cries using the mobile application.					
Users can register their baby's name in the mobile application for a personalized experience.					
Users can easily connect devices to their baby's profile for realtime monitoring of the baby's cries.					
Users receive instant notifications when their baby is crying, allowing for timely responses.					
Users can search for their registered baby's name quickly and easily.					
Users can also view a history of notifications related to the baby's cries, providing guardians with an overview of					

past alerts.					
It includes a simple, user-friendly interface that requires minimal training to navigate effectively.					
The application provides accurate analysis of the baby's cries, helping caregivers understand their needs better.					
Average Weighted Mean					

Usability Assessment of the User's Side

Scale	Range	Responses	Interpretation
5	4.24 – 5.00	Strongly Agree	Very Usable
4	3.43 – 4.23	Agree	Somewhat Usable
3	2.62 – 3.42	Neutral	Usable
2	1.81 - 2.61	Disagree	Less Usable
1	1.00 - 1.80	Strongly Disagree	Not Usable

Statement	5	4	3	2	1
The system's interface is intuitive and easy to navigate, requiring minimal training for first-time users.					
Users can register their baby's name effortlessly, enhancing ease of use within the application.					
The process of adding devices to associate with the baby's profile is straightforward, promoting seamless connectivity for real-time monitoring.					
Instant notifications alert users when their baby is crying, ensuring timely responses and fostering a user-friendly experience.					
Users can quickly search for their registered baby's name, streamlining navigation and improving usability.					
The application responds promptly to user actions, contributing to a seamless and efficient user experience.					

The application allows users to view a history of notifications related to the baby's cries, which enhances usability by providing guardians with an overview of past alerts and helping them monitor patterns over time.					
The system offers error messages or prompts when users upload incorrect or incomplete inputs, improving usability.					
Overall, the application provides accurate analysis of the baby's cries, helping caregivers understand their needs better and making the experience more effective.					
Average Weighted Mean					

Reliability Assessment of the User's Side

Scale	Range	Responses	Interpretation
5	4.24 – 5.00	Strongly Agree	Very Reliable
4	3.43 – 4.23	Agree	Somewhat Reliable
3	2.62 – 3.42	Neutral	Reliable
2	1.81 - 2.61	Disagree	Less Reliable
1	1.00 - 1.80	Strongly Disagree	Not Reliable

Statement	5	4	3	2	1
The system consistently captures and analyzes the baby's cries with accuracy, ensuring reliable data input for effective monitoring and responsive caregiving.					
The cry analysis algorithm reliably identifies and categorizes the baby's cries, providing relevant insights for caregivers.					
The text within the application is clear, easy to read, and understandable for all users.					
Instructions provided by the system are straightforward and easy to follow, enhancing user experience.					
Buttons and options on the screen are easily accessible and intuitive to use.					
The color scheme and design of the application are comfortable for long periods of use, reducing eye strain.					

The system provides clear feedback during the image upload and cry analysis processes, guiding users with messages such as "Your audio is being processed, please wait," which enhances the user experience and minimizes confusion.					
Users can easily understand how to interact with the application, making it user-friendly for all caregivers.					
The system is user-friendly for everyone (mothers, guardians, and caregivers).					
Average Weighted Mean					

Appendix C: Program Code

index.txs

```
import React, { useEffect } from "react"; import { View, Text, Image,
ScrollView } from "react-native"; import { useTheme } from
"@/hooks/useAppTheme"; import CustomButton from
"@/components/CustomButton"; import { Images } from
"@/constants";
import { SafeAreaView } from "react-native-safe-area-context";
import { router } from "expo-router"; import { Button } from "react-
native-paper";
import AsyncStorage from "@react-native-async-storage/async-storage";

export default function App() {   const
{ currentTheme } = useTheme();   useEffect(()
=> {
   const checkLoginStatus = async () => {       const userId =
await AsyncStorage.getItem('user_id');       if (userId) {
       router.replace("/home");
     }
   };

   checkLoginStatus();
 }, []);
 return (
   <SafeAreaView       className="h-full"
    style={{ backgroundColor: currentTheme.background }}
   >
     <ScrollView         contentContainerStyle={{
        height: "100%",
       }}       >
       <View className="flex items-center justify-center w-full h-full px-4">
         <Image
           source={currentTheme.applogo}         className="w-[130px]
   h-[84px]"          resizeMode="contain"
         />

         <Image
           source={Images.landing_image}             className="max-w-[380px]
w-full h-[298px]"
           resizeMode="contain"
```

```
        />

        <View className="relative mt-5">
          <Text
            className="text-3xl font-bold text-center"
            style={{ color: currentTheme.textColor }}
          >
            Discover Endless{"\n"}
            Possibilities with <Text className="text-[#426ae1]">MAide</Text>
</Text>
        </View>

        <Text
          className="text-sm text-center font-pregular mt-7"
style={{ color: currentTheme.textColor }}
        >
          Where Compassion Meets Technology: Embark on a Journey of Care and
Innovation with MAide, the app that identifies your baby's cry.          </Text>

        <Button
          onPress={() => router.push("/sign-in")}
className="w-full mt-10"
style={{ borderRadius: 12 }}            contentStyle={{
            backgroundColor: "#B3B7FA",
            borderRadius: 12,
            minHeight: 62,
          }}
        >
          <Text
   style={{            color:
   "#161622",
              fontFamily: "Poppins-SemiBold",
             fontSize: 18,
            }}
          >
            Login to Continue
          </Text>
        </Button>
      </View>
    </ScrollView>
   </SafeAreaView>
  );
}
```

+not.found.txs

```
import { Link, Stack } from 'expo-router';
import { StyleSheet } from 'react-native';

import { ThemedText } from '@/components/ThemedText';
import { ThemedView } from '@/components/ThemedView';

export default function NotFoundScreen() {   return (
   <>
     <Stack.Screen options={{ title: 'Oops!' }} />
     <ThemedView style={styles.container}>
       <ThemedText type="title">This screen doesn't exist.</ThemedText>
       <Link href="/" style={styles.link}>
         <ThemedText type="link">Go to home screen!</ThemedText>
       </Link>
     </ThemedView>
   </>
  );
}

const styles = StyleSheet.create({
  container: {     flex: 1,
    alignItems: 'center', justifyContent:
    'center', padding: 20,
  }, link: {   marginTop:
  15,
    paddingVertical: 15,
  },
});
```

+html.txs

```
import { ScrollViewStyleReset } from 'expo-router/html'; import { type
PropsWithChildren } from 'react';
```

```
/**
*         This file is web-only and used to configure the root HTML for every web page during static rendering.
*         The contents of this function only run in Node.js environments and do not have access to the DOM or browser APIs.
 */
export default function Root({ children }: PropsWithChildren) {   return (
   <html lang="en">
     <head>
       <meta charSet="utf-8" />
       <meta httpEquiv="X-UA-Compatible" content="IE=edge" />
       <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />

       {/*
         Disable body scrolling on web. This makes ScrollView components work closer to how they do on native.
         However, body scrolling is often nice to have for mobile web. If you want to enable it, remove this line.
       */}
       <ScrollViewStyleReset />

       {/* Using raw CSS styles as an escape-hatch to ensure the background color never flickers in dark-mode. */}
      <style dangerouslySetInnerHTML={{ __html: responsiveBackground }} />
       {/* Add any additional <head> elements that you want globally available on web... */}     </head>
     <body>{children}</body>
   </html>
  );
}

const responsiveBackground = ` body {
  background-color: #fff;
}
@media (prefers-color-scheme: dark) {   body {
    background-color: #000;
  }
}`;
```

[…unmatched].txs

```
import { Redirect } from "expo-router";
```

```
export default function AuthSessionRedirect() {   return
<Redirect href="/(auth)/sign-in" />;
}
```

layout.txs

```
import React, { useEffect } from "react"; import
{ useFonts } from "expo-font"; import { SplashScreen,
Stack } from "expo-router";
import { useSafeAreaInsets } from "react-native-safe-area-context";
import { ThemeProvider } from "@/context/ThemeContext"; // Adjust the path if necessary import
{ GoogleAuthProvider } from "@/context/GoogleAuthContext";
import { Alert, BackHandler } from "react-native"; import
{ UserProvider } from "../context/UserContext";
// Prevent the splash screen from auto-hiding before asset loading is complete.
// SplashScreen.preventAutoHideAsync();
const RootLayout = () => {   const insets =
useSafeAreaInsets();   const [fontsLoaded,
error] = useFonts({
    "Poppins-Black": require("../assets/fonts/Poppins-Black.ttf"),
    "Poppins-Bold": require("../assets/fonts/Poppins-Bold.ttf"),
    "Poppins-ExtraBold": require("../assets/fonts/Poppins-ExtraBold.ttf"),
    "Poppins-ExtraLight": require("../assets/fonts/Poppins-ExtraLight.ttf"),
    "Poppins-Light": require("../assets/fonts/Poppins-Light.ttf"),
    "Poppins-Medium": require("../assets/fonts/Poppins-Medium.ttf"),
    "Poppins-Regular": require("../assets/fonts/Poppins-Regular.ttf"),
    "Poppins-SemiBold": require("../assets/fonts/Poppins-SemiBold.ttf"),
    "Poppins-Thin": require("../assets/fonts/Poppins-Thin.ttf"),
  });

  useEffect(() => {
    if (error) throw error;

    if (fontsLoaded) {       SplashScreen.hideAsync();
    }
  }, [fontsLoaded, error]);

  useEffect(() => {
```

```
   // Handle back button press     const
backAction = () => {
     Alert.alert("Exit App", "Are you sure you want to exit the app?", [
       {                text: "Cancel",
onPress: () => null,
         style: "cancel"
       },
       { text: "YES", onPress: () => BackHandler.exitApp() }
     ]);
     return true; // Prevent default back action
   };

  const backHandler = BackHandler.addEventListener(
    "hardwareBackPress",    backAction
 );

   // Cleanup the event listener on unmount
   return () => backHandler.remove();
 }, []);

 if (!fontsLoaded) {     return
null;
 }

 return (
   <UserProvider>
     <ThemeProvider>
       <GoogleAuthProvider>
         <Stack>
           <Stack.Screen name="(tabs)" options={{ headerShown: false }} />
           <Stack.Screen name="(auth)" options={{ headerShown: false }} />
           <Stack.Screen name="index" options={{ headerShown: false }} />
           <Stack.Screen
name="[...unmatched]"
options={{ headerShown: false }}
           />
         </Stack>
       </GoogleAuthProvider>
     </ThemeProvider>

   </UserProvider>

 );
```

};

export default RootLayout;

- **Authentication**

layout.tsx

```
import { ThemeProvider } from "@/context/ThemeContext"; import
{ Redirect, Stack } from "expo-router";
import { StatusBar } from "expo-status-bar";

const AuthLayout = () => {

  return (

    <Stack>        <Stack.Screen
name="sign-in"
options={{           headerShown:
false,
        }}
      />
      <Stack.Screen          name="sign-
up"
options={{           headerShown: false,
        }}
      />
      <Stack.Screen
name="forgotpassword"
options={{           headerShown: false,
        }}
      />
      <Stack.Screen
name="otpverification"
options={{           headerShown: false,
        }}
      />
      <Stack.Screen
name="newpassword"          options={{
         headerShown: false,
        }}
```

```
      />
    </Stack>


  );
};

export default AuthLayout;
```

forgotpassword.tsx

```
import { View, Text, ScrollView, Image, Dimensions } from 'react-native' import
React, { useState } from 'react' import { SafeAreaView } from 'react-native-safe-
area-context' import { useTheme } from '@/hooks/useAppTheme'; import
{ Images } from '@/constants'; import FormField from
'@/components/FormField'; import { Button, IconButton } from 'react-native-
paper'; import { useNavigation } from '@react-navigation/native'; import
{ router } from 'expo-router';

const ForgotPassword = () => {     const
{ currentTheme } = useTheme();
   const [form, setForm] = useState({         email:
"",
   });

   const [errors, setErrors] = useState({
      email: "",
   });

   const handleEmailChange = (text: string) =>
{        setForm({ ...form, email: text });
      if (text !== "") {
         setErrors((prevErrors) => ({ ...prevErrors, email: "" }));
      }
   };
   const navigation = useNavigation();
   return (
      <SafeAreaView            className="h-
full"
         style={{ backgroundColor: currentTheme.background }}
      >
         <ScrollView>
```

```
          <View
              className="w-full flex justify-center h-full px-4 my-6"
style={{
                  minHeight: Dimensions.get("window").height - 200,
              }}
           >
              <IconButton
icon="chevron-left"
                  size={24}
                  onPress={() => navigation.goBack()}
iconColor={`${currentTheme.textColor}`}
                  style={{ backgroundColor: currentTheme.background, borderColor: "#dbdfe3",
borderRadius: 10, zIndex: 10 }}
                  className="absolute top-0  border"
              />
              <View className="items-center">
                  <Image
                      source={Images.forgotpasswordimage}
                      resizeMode="contain"
className="h-[330px] mb-7"
                  />
              </View>
              <View className="">
                  <Text
                      className="text-2xl font-semibold mt-10 mb-2 font-psemibold "
style={{ color: currentTheme.textColor }}
                  >
                      Forgot Password?
                  </Text>
<Text
                      className="font-plight"
                      style={{ color: currentTheme.textColor }}
                  >
                      Don't worry! It occurs. Please enter the email address linked with your account
</Text>
              </View>                  <FormField
title="Email"                      value={form.email}
handleChangeText={handleEmailChange}
                  otherStyles="my-6"
                  placeholder="Enter your email"
                  error={errors.email}
              />
              <Button
```

```
                onPress={() => {router.replace('/otpverification')}}
className="w-full pb-5"                    style={{ borderRadius: 12 }}
contentStyle={{
                    backgroundColor: "#B3B7FA",
                    borderRadius: 12,
minHeight: 62,
                }}
              >                  <Text
style={{                       color:
"#161622",
                      fontFamily: "Poppins-SemiBold",
                      fontSize: 18,
                    }}
                  >
                    Send Code
                  </Text>
                </Button>
              </View>
            </ScrollView>
          </SafeAreaView>
    )
}

export default ForgotPassword
```

newpassword.tsx

```
import { View, Text, ScrollView, Image, Dimensions } from 'react-native' import
React, { useState } from 'react' import { SafeAreaView } from 'react-native-safe-
area-context' import { useTheme } from '@/hooks/useAppTheme'; import
{ Images } from '@/constants'; import FormField from
'@/components/FormField';
import { Button, Dialog, IconButton, Portal } from 'react-native-paper'; import { useNavigation } from
'@react-navigation/native'; import { router } from 'expo-router';

const ForgotPassword = () => {
    const [visible, setVisible] = useState(false);

    const showDialog = () => setVisible(true);
```

```
   const hideDialog = () => setVisible(false);     const
{ currentTheme } = useTheme();
   const [form, setForm] = useState({         email:
"",
   });

   const [errors, setErrors] = useState({
      email: "",
   });

   const handleEmailChange = (text: string) =>
{         setForm({ ...form, email: text });
      if (text !== "") {
         setErrors((prevErrors) => ({ ...prevErrors, email: "" }));
      }
   };
   const navigation = useNavigation();
   return (
      <SafeAreaView            className="h-
full"
         style={{ backgroundColor: currentTheme.background }}
      >
         <ScrollView>
<View
               className="w-full flex justify-center h-full px-4 my-6"
style={{
                  minHeight: Dimensions.get("window").height - 200,
               }}
            >
               <IconButton
 icon="chevron-left"                    size={24}
                  onPress={() => navigation.goBack()}
  iconColor={`${currentTheme.textColor}`}
                  style={{ backgroundColor: currentTheme.background, borderColor: "#dbdfe3",
borderRadius: 10, zIndex: 10 }}
                  className="absolute top-0  border"
               />
               <View className="items-center">
                  <Image
                     source={Images.resetpasswordImage}
                     resizeMode="contain"
                     className="h-[140px]"
                  />
```

```
            </View>
            <View className="">
              <Text
                 className="text-2xl font-semibold mt-10 mb-2 font-psemibold "
style={{ color: currentTheme.textColor }}
              >
                 Create New Password
              </Text>
<Text
                 className="font-plight"
                 style={{ color: currentTheme.textColor }}
              >
                 Your new password must be unique from previously used.
              </Text>
            </View>
<FormField                    title="New
Password"
value={form.email}
              handleChangeText={handleEmailChange}
              otherStyles="my-6"
placeholder="Enter password"
              error={errors.email}
            />
            <FormField
              title="Confim Password"
value={form.email}
handleChangeText={handleEmailChange}
              otherStyles="mb-[28px]"
              placeholder="Enter confirm password"
error={errors.email}
            />
            <Button
              onPress={showDialog}
              className="w-full pb-5"
style={{ borderRadius: 12 }}
              contentStyle={{
                 backgroundColor: "#B3B7FA",
                 borderRadius: 12,
minHeight: 62,
              }}
            >
              <Text
style={{
```

```
                        color: "#161622",
                        fontFamily: "Poppins-SemiBold",
                        fontSize: 18,
                      }}
                    >
                      Reset Password
                    </Text>
                  </Button>
                </View>
              </ScrollView>
              <Dialog visible={visible} onDismiss={hideDialog} style={{ backgroundColor: '#eff8ff' }}>
                <Dialog.Title>Reset Successful</Dialog.Title>
                <Dialog.Content>
                  <Text>Your password has been successfully reset.</Text>
                </Dialog.Content>
                <Dialog.Actions>
                  <Button onPress={() => { router.replace('/sign-in') }}>OK</Button>
</Dialog.Actions>
              </Dialog>

            </SafeAreaView>
        )
}

export default ForgotPassword
```

otpverification.tsx

```
import { View, Text, ScrollView, Image, Dimensions, TextInput } from 'react-native' import React,
{ useRef, useState } from 'react'
import { SafeAreaView } from 'react-native-safe-area-context'
import { useTheme } from '@/hooks/useAppTheme'; import
{ Images } from '@/constants'; import FormField from
'@/components/FormField'; import { Button, IconButton } from
'react-native-paper'; import { useNavigation } from '@react-
navigation/native'; import { OTPInput } from 'react-native-otp-
component';
import { router } from 'expo-router';

const OTPVerification = () => {
    const { currentTheme } = useTheme();

    const navigation = useNavigation();
```

```
   const [codes, setCodes] = useState<string[] | undefined>(Array(6).fill("")); const
refs = Array(6)
      .fill(null)
      .map(() => useRef<TextInput>(null));

   const [errorMessages, setErrorMessages] = useState<string[]>();

   const onChangeCode = (text: string, index: number) =>
{       if (text.length > 1) {           setErrorMessages(undefined);
const newCodes = text.split("");           setCodes(newCodes);
refs[5]!.current?.focus();
         return;
      }
      setErrorMessages(undefined);
```

Misamis University
Ozamiz City
COLLEGE OF COMPUTER STUDIES

MU-CCS-028/28June2024

GRAMMARIAN'S CERTIFICATE

Date: May 21, 2025

This is to certify that the undersigned, **Markdy Y. Orong, DIT** has reviewed and went through all the pages of the manuscript of the Thesis/Capstone Project entitled **MAide: A Mother's Aide in Recognizing and Analyzing Baby Cries using Deep Learning"** as against the set of structural rules that govern the composition of sentences, phrases, and words in the English language.

Signed:

MARKDY Y. ORONG, DIT
Grammarian

Conformer:

JESSA ABADILLA

KYLAH M. OSTIA

CURRICULUM VITAE

KYLAH M. OSTIA
Bachelor of Science in Information Technology, Misamis University, Ozamiz City 7200, Misamis Occidental, Mindanao, Philippines
Email: kykyostia11@gmail.com

PERSONAL INFORMATION

Age : 22
Birthdate : July 11, 2002
Citizenship: Filipino

Civil Status: Single

EDUCATIONAL BACKGROUND

Primary

- ***Misamis University*****,** H.T. Feliciano St., Ozamiz City, Misamis Occidental, 2012-2013

Secondary

- ***Misamis University*****,** H.T. Feliciano St., Ozamiz City, Misamis Occidental, Science, Technology, Engineering, and Mathematics (STEM) 2019-2020

Tertiary

- ***Misamis University*****,** H.T. Feliciano St., Ozamiz City, Misamis Occidental, Bachelor of Science in Information Technology (BSIT), 2020-Present

SKILLS

- UI Designer
- Frontend Designer

ORGANIZATION

- **Member,** Philippines Society of Information Technology Students (PISTS), 2020 up to present.

CURRICULUM VITAE

JESSA ABADILLA
Bachelor of Science in Information Technology, Misamis University, Ozamiz City 7200, Misamis Occidental, Mindanao, Philippines
Email: jeessaabadilla@gmail.com

PERSONAL INFORMATION

Age	**:** 21
Birthdate	**:** December 24, 2002
Citizenship	**:** Filipino
Civil Status	**:** Single

EDUCATIONAL BACKGROUND

Primary

- ***Ozamiz City Central School***, Ozamiz City, Misamis Occidental, 2012-2013

Secondary

- ***Ozamiz City National High School***, Ozamiz City, Misamis Occidental 2014-2018
- ***Misamis University*****,** H.T. Feliciano St., Ozamiz City, Misamis Occidental, Science, Technology, Engineering, and Mathematics (STEM) 2019-2020

Tertiary

- ***Misamis University*****,** H.T. Feliciano St., Ozamiz City, Misamis Occidental, Bachelor of Science in Information Technology (BSIT), 2020-Present

SKILLS

- Graphic Design
- Front-end Developer

ORGANIZATION

- **Member,** Philippines Society of Information Technology Students (PSITS), 2020 up to present