VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Autonomous attendance mobile application based on face recognition or voice recognition

By
Hoàng Minh – ITITIU19029

A thesis submitted to the School of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Bachelor of Information Technology/Computer Science/Computer Engineering

Ho Chi Minh City, Vietnam
Year 2023

Autonomous attendance mobile application based on face recognition or voice recognition

APPROVED BY:

_________________________________,
Doctor Ly Tu Nga

____
(*Typed Committee name here*)

____
(*Typed Committee name here*)

____
(*Typed Committee name here*)

____
(*Typed Committee name here*)

THESIS COMMITTEE
(Whichever applies)

ACKNOWLEDGMENTS

Embarking on this thesis journey has been an enriching experience, and I owe my gratitude to the wonderful people and institutions who made it possible. Completing a project on "Autonomous Attendance Mobile Application Based on Face Recognition or Voice Recognition" would not have been half as rewarding without the support and encouragement I received along the way.

First, a huge shoutout to my supervisor, Mrs. Ly Tu Nga, as the guiding star in this academic galaxy. Your insights and encouragement have made navigating the complexities of this topic a much smoother ride.

I want to extend my appreciation to the entire faculty of the Information Technology Department for fostering an environment of curiosity and learning. The tip of the hat to the International University for providing not just a campus but also a home for exploration.

A big virtual round of applause for the participants of my study-you made the research process not just insightful but downright enjoyable. Your willingness to share your experiences has added a layer of depth to this work that would not have been possible otherwise.

To my family, the real supporters: Your unwavering support and occasional nudges to take breaks kept me sane. Thank you for your consideration.

This thesis is a culmination of shared efforts, wisdom, and moments of inspiration. It made this academic endeavor not just a task, but a rewarding journey.

Cheers to the end of one chapter and the beginning of another!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In Kazakhstani universities, the process of checking student attendance is a crucial aspect of evaluating their performance and determining their final grades. Currently, the traditional method of manual attendance is time-consuming and prone to error [1]. According to the previous section, attendance information should be gathered in the classroom to confirm the presence of students and teachers. However, this is not an efficient way, since it takes up a significant amount of time to call out each student's name and mark their attendance. Moreover, this manual approach also leaves room for students to forge their attendance, either by marking their friends as present or by proxying for someone else. Consequently, there is a need for an automated system that simplifies the process of checking attendance and eliminates fraud. To overcome these drawbacks, the proposed system uses an autonomous attendance-checking Android application. This application utilizes various technologies to enhance the accuracy and convenience of attendance management [2]. Android, designed for low-power devices, provides a perfect platform for developing such applications [1]. Firebase provides multiple functions to help complete the application. In conclusion, the proposed autonomous attendance-checking Android application utilizes Firebase Authentication, Firebase Cloud Messaging, Firebase Machine Learning, TensorFlow Lite model for face recognition, Speech Recognizer class for voice recognition, File class for storing serialized files, and Firebase Realtime Database. Overall, the application leverages the power of Firebase services and machine-learning technologies to create a secure and efficient attendance-checking system. In conclusion, the project aims to create an autonomous attendance-checking Android application that utilizes various Firebase services such as authentication, cloud messaging, and a real-time database.

Keywords: Face Detection, Face Recognition, Firebase Authentication, Firebase Realtime Database, Firebase Cloud Messaging, Firebase Machine Learning, TensorFlow Lite, Voice Recognition.

CHAPTER 1

INTRODUCTION

1.1. Background

Face recognition is essential in everyday life because it allows us to recognize family, friends, and familiar people. We might not have realized that various processes have been used to recognize human faces. Human intelligence enables the reception and evaluation of information during the recognition process. We obtain information through images transmitted to our eyes, primarily the retina, in the form of light. Light is a type of electromagnetic wave emitted from a source onto an object and projected into human perception. Robinson-Riegler, G., and Robinson-Riegler, B. [3] stated that following visual processing by the human visual system, we classify the shape, size, contour, and texture of the item to interpret the information. The examined data were compared to other representations of objects or faces in human memory for recognition. In reality, developing an automated system capable of recognizing faces in the same way as humans is a difficult problem. However, a large amount of memory is required to distinguish between diverse looks. For example, at universities, there are many students of all races and genders; it is difficult to remember every face of an individual without making mistakes. Facial recognition systems employ computers with nearly infinite memory, high processing speed, and power to circumvent human limitations.

The human face is a distinct expression for individual individuals. Thus, face recognition is described as a biometric approach in which an individual is identified by comparing a real-time captured image to photographs previously recorded in that person's database [4].

Face recognition systems are becoming increasingly popular owing to their ease of use and superior performance. Face recognition is used by airport security and FBI to monitor criminals, missing children, and narcotic activities [5]. Facebook, a famous social networking website, uses facial recognition to allow users to tag friends in photos for entertainment [6]. Intel allows customers access to online accounts using facial recognition [7]. Apple allows consumers to unlock their iPhone X using facial recognition [8].

Facial recognition research began in 1960. Woody Bledsoe, Helen Chan Wolf, and Charles Bisson developed a system that required the administrator to identify eyes, ears, nose, and mouth from photographs. The distances and ratios between the identified features and common reference points were computed and compared. Goldstein, Harmon, and Lesk expanded the experiments in 1970 by automating recognition using other variables, such as hair color and lip thickness. Kirby and Sirovich proposed principle component analysis (PCA) in 1988 as a solution to the face recognition challenge.

This thesis addresses these issues through the development of an autonomous mobile application that leverages the precision of face or voice recognition, thereby transforming conventional paradigms of attendance management.

1.2. Problem Statement

Traditional methods of attendance tracking, such as manual registers and card-based systems, pose challenges in terms of accuracy, reliability, and susceptibility to fraudulent practices. This thesis aims to address these challenges by proposing an autonomous mobile application that leverages the precision of face or voice recognition to redefine the landscape of attendance management.

Zhao et al. [9] published an article outlining the issues associated with face identification. One of the challenges in facial recognition is distinguishing between known and unfamiliar photos. Furthermore, the article provided by Pooja et al. [10] discovered that the training procedure for a face recognition student attendance system was sluggish and time-consuming. Furthermore, the article provided by Priyanka Wagh et al. [11] said that varied lighting and head positions are frequently the issues that might decrease the effectiveness of a facial recognition-based student attendance system.

Attendance tracking is not merely a logistical task but a critical aspect of institutional governance. The inefficiencies of traditional methods consume valuable time and compromise the accuracy of attendance records, thereby influencing the decision-making processes. The need for a comprehensive solution becomes apparent in educational institutions, corporate settings, and other sectors, where attendance plays a pivotal role.

1.3. Scope and Objectives

Planning clear and smart objectives for the application is necessary to dertermine all the steps needed to build an Androind application. The objectives are presented in the below table:

Table 1.1: Objectives for the application

Objective	Content
1	Build an Android application for teachers and students
2	Provide face recognition or voice recognition into the system
3	Provide a method to send notification for the Teacher

Objective 1: Build an Android application for teachers and students: Creating an "autonomous attendance mobile application based on face recognition or voice recognition" requires some basic knowledge of how a mobile application works as well as a mobile application lifecycle. A complete application should have an authentication system for the user to log in and a database to store all the information. For these two requirements, the Firebase Authentication and Firebase Realtime Database will solve them easily. Therefore, the main goal of this thesis is to create a completely autonomous attendance mobile application. Therefore, some basic user functions are listed below:

Student (User)

1. Register Face
2. Register Voice

3. Check In Using Face Recognition
4. Check In Using Voice Recognition

Teacher (User)

1. Request Check In Using Face Recognition
2. Request Check In Using Voice Recognition
3. View Attendance List

Objective 2: Provide face recognition and voice recognition into the system: To provide a face recognition model into the system, two steps are required: using a Firebase Machine Learning face detection API to detect faces and the TensorFlow Lite model to recognize faces. Regarding voice recognition, using an API written by Java, which is a SpeechRecognizer, is essential.

Objective 3: Provide a method to send notifications to the Teacher: Requiring students to check attendance is a simple job that is usually done manully. To avoid the teacher having to speak aloud in the classroom, they can press a button in the application to notify their students to check using Firebase Cloud Messaging.

1.4. Assumption and Solution

As previously mentioned, the system is divided into two types of users: teachers and students. Therefore, two different menu screens should be designed and implemented to avoid the teacher going to the Student Menu Screen. Using email formats such as "@student.hcmiu.edu.vn" and "@hcmiu.edu.vn" to identify the Student and the Teacher would be a solution to this problem.

In addition, Firebase provides many useful APIs such as Firebase Authentication, Firebase Realtime Database, Firebase Machine Learning, and Firebase Cloud Messaging, which are used in many applications worldwide and will solve almost all thesis problems.

1.5. Structure of thesis

The proposition contains six chapters, each of which offers and clarifies different perspectives on the issue.

● Chapter 1: INTRODUCTION

- Provide a short and clear introduction to why and how the application was created.

● Chapter 2: LITERATURE REVIEW

- Presenting the existing literature related to the application.

● Chapter 3: METHODOLOGY

- Outlining the approach for designing and developing applications.

● Chapter 4: IMPLEMENT AND RESULTS

- Providing the implementation details and the results of the application

● Chapter 5: DISCUSSION AND EVALUATION

- Analyzing and discussing the results presented in the previous chapter.

● Chapter 6: CONCLUSION AND FUTURE WORK

- Summarizing the key findings of the study and draws conclusions about the application's impact and contributions

CHAPTER 2

LITERATURE REVIEW

2.1. Student Attendance System

Arun Katara et al. [13] identified the limitations of the Radio Frequency Identification (RFID) card, fingerprint, and iris recognition systems. Because of its ease, an RFID card program was developed, and the user regularly supported their friends by verifying whether they had their friend's identification card. The fingerprint method is effective but inefficient because the verification procedure takes time, requiring the user to align and perform each verification one at a time; it occasionally fails many times before the user checks. Nonetheless, the human face is exposed to facial recognition but includes fewer details than the iris. Iris recognition systems with additional details may compromise user privacy. Voice recognition technology is available, although it is less accurate than the other approaches. Hence, a system that combines face and voice recognition is suggested for use in student attendance system solutions to enhance accuracy.

2.2. Definition of Terms and History

2.2.1. Face detection

Face detection[15] is a computer technology used in various applications to identify human faces in digital photos. Face detection refers to the psychological process through which humans locate and focus on faces in a visual environment.

Face detection can be considered as a subset of object-class detection. Object-class detection involves determining the locations and sizes of all the objects in a picture belonging to a specific class. Examples include the upper torsos, pedestrians, and automobiles. Face detection answers two questions: 1. Do the collected photos or videos contain human faces? 2. Where is the face ?

Face-detection algorithms have been designed to detect human faces from the front. This is similar to image detection in that the image of a person is matched bit by bit. This image matches the image stored in the database. Any modifications to the database's facial features invalidate the matching procedure.

Facial recognition software employs numerous techniques, each with its own advantages and disadvantages.

Viola-Jones algorithm. This approach involves training a model to recognize faces. Although this framework is still popular for real-time face recognition, it is difficult to distinguish faces that are covered or not properly positioned.

Knowledge- or rule-based. These techniques characterize faces according to these principles. However, it is difficult to develop clearly defined knowledge-based norms is difficult.

Feature-based or feature-invariant. These techniques identify facial traits using an individual's eyes or nose. Light and noise may have negative effects on this.

Template matching. This technique uses previously stored standard facial traits or templates to compare photos and identify faces. Nevertheless, this method has trouble correcting variances in terms of shape, stance, and proportion.

Appearance-based. This technique determines pertinent face picture attributes by applying machine learning and statistical analysis. Adapting appearance-based approaches to changes in illumination and orientation can provide challenges.

Convolutional neural network-based. Designed to analyze pixel input, a convolutional neural network (CNN) is a form of deep learning artificial neural network (ANN) used in image recognition and processing. CNN frameworks can use suggestions from region-based CNN, or R-CNN, to locate and categorize objects in pictures. Similar to other places like the pixelated area of the eye, these recommendations concentrate on specific portions or regions within the image. The R-CNN recognizes that it has discovered a match if this eye area matches the other eye regions. CNNs, however, have the potential to "overfit," which occurs when they match noisy areas in the training set but not the anticipated patterns of face characteristics.

Single-shot detector (SSD). SSD just needs one camera shot to recognize numerous objects in the picture, but region proposal network-based techniques like R-CNN require two camera shots—one to produce region proposals and another to detect the item of each proposal. SSD is hence quicker than R-CNN. SSD, however, has trouble identifying faces that are little or farther away from the camera.

Face detection is used in marketing, photography, lip reading, facial motion capture, facial identification, and emotional interference.

2.2.2. Face recognition system

Systems for matching human faces from digital photos or video frames to a facial database are known as facial recognition systems [16]. Identifying and quantifying face characteristics from a picture, this type of technology is widely used to verify individuals through ID verification services.

In the 1960s, computer applications marked the beginning of the development of analog systems. Since their introduction, facial recognition technologies have been seen in a growing number of smartphones and other devices, including robots. face recognition software is categorized as biometrics as it uses computerized face recognition to measure an individual's physiological traits. Even though face recognition technologies are not as accurate as iris recognition, fingerprint image acquisition, palm recognition, and voice recognition, they are nonetheless used in biometric applications. Because of its contactless method, it is frequently employed. Advanced human-computer interaction, video surveillance, law enforcement, passenger screening, hiring and housing choices, and self-image indexing are just a few of the applications that have made use of facial recognition technology.

2.2.3. Voice recognition

A branch of computer science and computational linguistics known as voice recognition [17] creates tools and techniques to let computers understand spoken language and convert it into text. These are sometimes referred to as speech-to-text

(STT), automated speech recognition (ASR), and computer speech recognition. It integrates research and information from the fields of languages, science, and computer engineering.

"Solly" (or "enrollment") is a need for several speech recognition systems, when a speaker reads distinct text or vocabulary into the system. By examining each individual's unique speech, the system may improve the accuracy of voice recognition by fine-tuning it. Systems referred to as "speaker-independent" do not require training. The so-called "speaker-dependent" training systems.

Applications for voice recognition include voice user interfaces for standard structured documentation (like project reports), speaker characteristics identification, speech-to-text processing (like word processors or texting), voice-to-text dialing (like "call my girlfriend"), call routing (like "I want to call my girlfriend"), domotic device control, keyword searches (like "find a YouTube video named Baby Shark"), simple data entry (like "enter a phone number"), and aircraft (direct voice input).

2.2.4. Firebase

Google offers a range of backend cloud computing services and application development platforms under Firebase Inc. [18]. Databases, services, authentication, and integration for several applications—including JavaScript, Node.js, Android, iOS, Java, Unity, PHP, and C++—are hosted on it.

James Tamplin and Andrew Lee launched Envolve, a firm that gave rise to Firebase in 2011. Developers were able to include live chat features into their websites by using an API that Evolve supplied. Last name. Tamplin and Lee found out that the chat service was being exploited to send application data other than chat messages after releasing it. Envolve is used by developers to allow users to synchronize real-time application data, such as game state. Tamplin and Lee made the decision to divide the real-time infrastructure supporting the chat system from the chat system itself. In 2011, they established Firebase as a stand-alone business, and in April 2012, they went public.

2.2.5. TensorFlow

A free open-source software library for AI and machine learning is called TensorFlow[23]. Although it may be applied to many different tasks, deep neural network training and inference are among its best uses.

The Google Brain team developed TensorFlow for use in internal research and production. Under the Apache License 2.0, the first version was made available in 2015. In September 2019, TensorFlow 2.0 was released by Google.

Numerous programming languages, including as Python, JavaScript, C++, and Java, are supported by TensorFlow. It is appropriate for a variety of applications across several sectors due to its versatility.

Google Brain's second-generation technology is called TensorFlow. February 11, 2017 saw the release of version 1.0.0.Unlike the standard version, which is available for 64-bit Linux, macOS, Windows, and mobile computing platforms like Android and iOS, TensorFlow can run on multiple CPUs and GPUs.

Because of its flexible design, computing may be easily deployed across a variety of platforms (CPUs, GPUs, and TPUs), including PCs, server clusters, mobile devices, and edge devices.

Stateful dataflow graphs were used for TensorFlow computations. The actions that these neural networks perform on multidimensional data arrays, or tensors, are the source of TensorFlow. In June 2016, Jeff Dean disclosed at the Google I/O Conference that just five of the 1,500 GitHub projects mentioning TensorFlow were from Google.

At a conference in December 2017, engineers from Google, Cisco, RedHat, CoreOS, and CaiCloud first introduced Kubeflow. TensorFlow can be deployed and operated on Kubernetes thanks to Kubeflow. TensorFlow.js version 1.0 for JavaScript machine learning was published by Google in March 2018. In January 2019, TensorFlow 2.0 was revealed by Google. Official access to it began in September 2019. Google released TensorFlow Graphics in May 2019 as a deep-learning framework for computer graphics.

2.3. Similar existing Application and Feature

2.3.1. Face Recognition Based Attendance System

The Face Recognition Based Attendance System [14] is a face recognition attendance system written in Python and OpenCV as the open-source computer vision library.

2.3.2. Mobile Attendance Checking System on Android Platform for Kazakhstani University

The mobile Attendance Checking System on Android Platform for Kazakhstani University [1] is a mobile system that checks student attendance based on the Android OS.

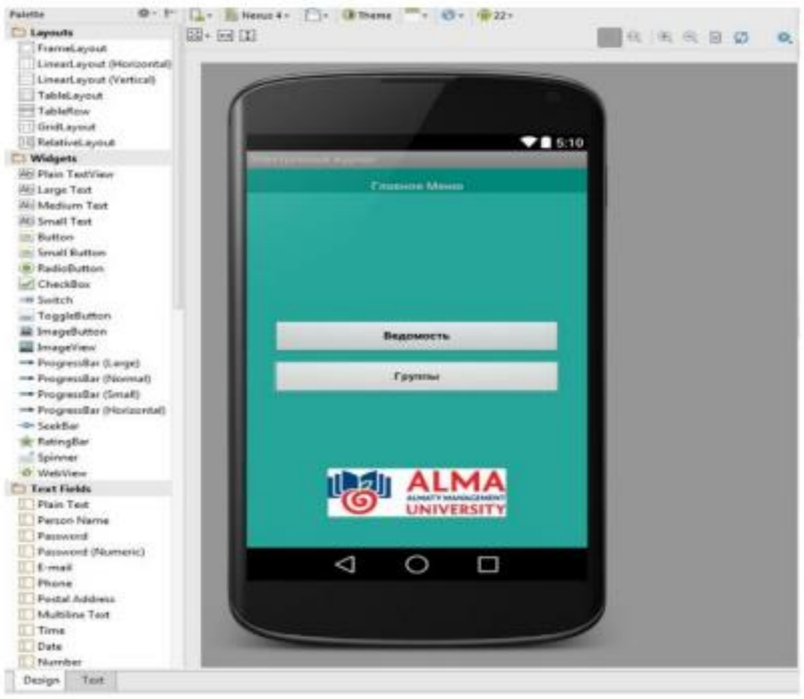


Figure 2.1: The start of the application [1]

CHAPTER 3

METHODOLOGY

3.1. Overview

The application is first divided into two types of users. The first will be the student, and the second will be the teacher, each with a different function. In the very beginning, when you open the application, the login screen is displayed, and you can log in to your existing account or go to the registration screen to create a new account. The email address defines the account’s type of user. For Students, checking in will be the main function, while on the other hand, requesting checking attendance will be the main function of the Teacher. Thus, the processes for both types are as follows:

Only teachers and students are the main users of the system, as follows:

Student: After logging in by student email, the student menu screen was displayed. On this screen, users can register their faces in the face recognition database and their answers using their voice in the voice recognition database. In addition, they can check in by using their face in the face recognition part or by using their answer in the voice recognition part. After checking in, the student's name, student identification number, date, and time were stored in the cloud database.

Teacher: After logging in using a teacher email, the teacher menu screen is displayed. On this screen, users can request their students to check their attendance by pressing the checking attendance button. Users can choose between face and voice recognition based on their requirement for accuracy. Users can also view the data stored in the cloud database.

The types of accounts will now be easily defined and can be upgraded into an optimized solution. The system checks whether the email that users type into the register contains “@student.hcmiu.edu.vn,” the accounts of the users will be defined as Student type, and if it contains “@hcmiu.edu.vn, ” accounts will be defined as the teacher type. All information is stored in the Firebase Database so that the admin can modify it if there are any problems.

The system was built as shown below:

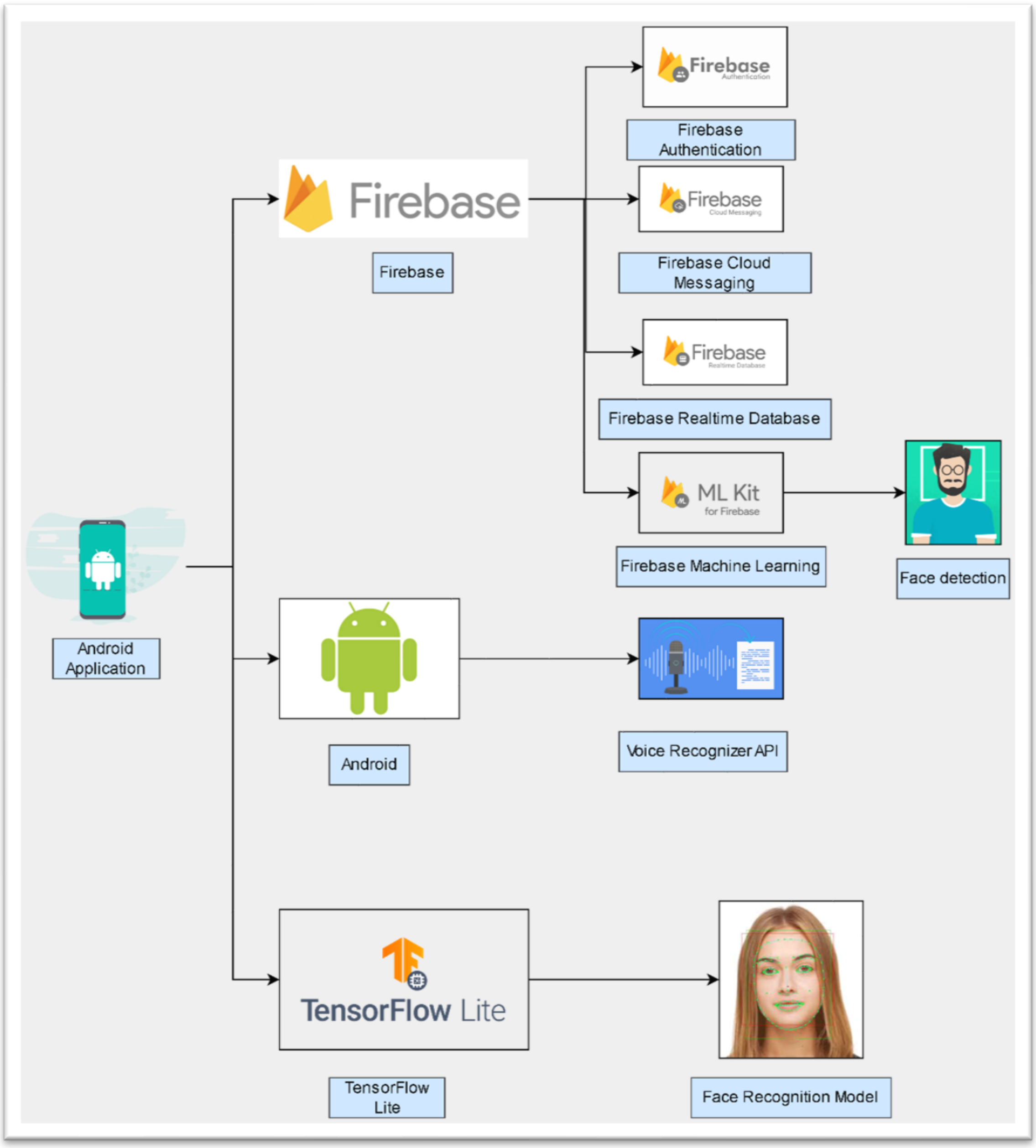


Figure 3.1: System overview diagram

3.2. User requirement analysis

The application is first divided into two types of users. The first will be the student, and the second will be the teacher, each with a different function. In the very beginning, when you open the application, the login screen is displayed, and you can log in to your existing account or go to the registration screen to create a new account. The email address defines the account's type of user. For Students, checking in will be the main function, while on the other hand, requesting checking attendance will be the main function of the Teacher. Thus, the processes for both types are as follows:

Only teachers and students are the main users of the system, as follows:

Student: After logging in via student email, the student menu screen is displayed. On this screen, users can register their faces in the face recognition database and their answers using their voice in the voice recognition database. In addition, they can check in by using their face in the face recognition part or by using their answer in the voice recognition part. After checking in, the student's name, student identification number, date, and time were stored in the cloud database.

Teacher: After logging using a teacher email, the teacher menu screen is displayed. On this screen, users can request their students to check their attendance by pressing the checking attendance button. Users can choose between face and voice recognition based on their requirement for accuracy. Users can also view the data stored in the cloud database.

The types of accounts will now be easily defined and can be upgraded into an optimized solution. The system checks whether the email that users type into the register contains "@student.hcmiu.edu. vn," the accounts of the users will be defined as Student type, and if it contains "@hcmiu.edu. vn, " the accounts will be defined as the teacher type. All information is stored in the Firebase Database so that the admin can modify it if there are any problems.

3.3. Use Case and Analysis

3.3.1. Use Case and Description

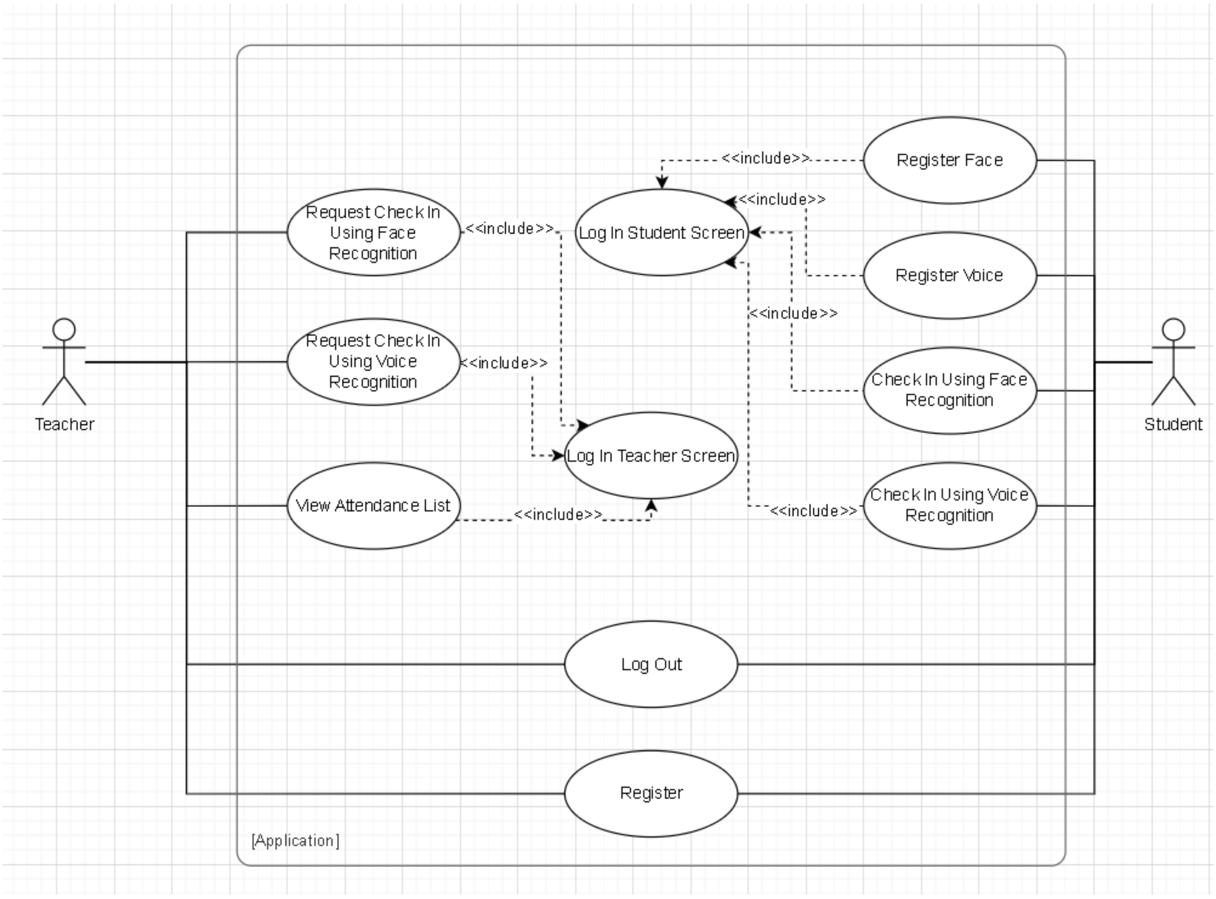


Figure 3.2: Use Case diagram for the application

Table 3.1: Description for "Register Face" use case

Name	UC-1: Register Face
Actor	Student
Preconditions	(none)
Trigger	Student wants to register their face into the database
Postconditions	
Main Scenario	1. Student presses a "Register Face" button.

	2. Navigate to Register Face Screen, Student chooses between capturing a new image or choosing an existing image from their device’s library. 3. Student adds their full name. 4. Student presses “Register.” 5. System shows a text “Your face is fully registered” .
Extensions	(none)

Table 3.2: Description for “Register Voice” use case

Name	UC-2: Register Voice
Actor	Student
Preconditions	(none)
Trigger	Student wants to register their answer for the security question
Postconditions	(none)
Main Scenario	1. Student presses a “Register Voice” button. 2. Navigate to Register Voice Screen, Student presses “Select a question” and a Question menu drops down. 3. Student chooses a question that they like. 4. Student presses “Start Recording” button. 5. Student says their answer aloud. 6. Student presses “Stop Recording” button. -- If the answer is not right, Student can go back to Step 4 again. 7. Student presses “Register.” 8. The system shows a text “Your answer is successfully registered”
Extensions	(none)

Table 3.3: Description for “Check In Using Face Recognition” use case

Name	UC-3: Check In Using Face Recognition
Actor	Student
Preconditions	(none)
Trigger	Teacher requests Student to check in using face recognition
Postconditions	(none)
Main Scenario	1. Student presses “Face Recognize” 2. Navigate to Face Recognize Screen, Student chooses between the front or back camera of their device. 3. Student put their face into the camera. -- If their face exists in the system database, go to Step 4. 4. The system shows a text “You have checked attendance.”
Extensions	(none)

Table 3.4: Description for “Check In Using Voice Recognition” use case

Name	UC-4: Check In Using Voice Recognition
Actor	Student
Preconditions	(none)
Trigger	Teacher requests Student to check in using voice recognition
Postconditions	(none)
Main Scenario	1. Student press “Voice Recognize” 2. Student press “Select a question” and a Question menu drops down. 3. Student choose the question that they registered before. 4. Press “Start Recording” button. 5. Student says their answer aloud. 6. Press “Stop Recording” button.

	-- If the answer is not right, Student can go back to Step 3 again. 7. Press "Recognize." -- If the answer and the question match, go to Step 9. If they do not match, go to Step 1. 8. The system shows a text "You have checked attendance" 9. The system pushes their check-in record into Firebase Realtime Database. 10. The system will go back to the Student Menu screen.
Extensions	(none)

Table 3.5: Description for "Log Out" use case

Name	UC-5: Log Out
Actor	Student/ Teacher
Preconditions	(none)
Trigger	When Student/Teacher wants to log out their account
Postconditions	(none)
Main Scenario	1. Press "Log out" button. 2. The system will go back to the Log In Screen.
Extensions	(none)

Table 3.6: Description for "Request Check In Using Face Recognition" use case

Name	UC-6: Request Check In Using Face Recognition
Actor	Teacher
Preconditions	(none)
Trigger	Teacher wants to request their Students to do a check in using face recognition

Postconditions	(none)
Main Scenario	1. Press “Check Face Attendance” 2. The system will send a notification to Student’s device to notify them.
Extensions	(none)

Table 3.7: Description for “Request Check In Using Face Recognition” use case

Name	UC-7: Request Check In Using Voice Recognition
Actor	Teacher
Preconditions	(none)
Trigger	Teacher wants to request their Students to do a check in using voice recognition
Postconditions	(none)
Main Scenario	1. Press “Check Voice Recognition” 2. The system will send a notification to Student’s device to notify them.
Extensions	(none)

Table 3.8: Description for “View Attendance List” use case

Name	UC-8: View Attendance List
Actor	Teacher
Preconditions	(none)
Trigger	Teacher wants to view attendance list
Postconditions	(none)
Main Scenario	1. Press “Attendance List” 2. The system will show up a new screen that includes information about Student’s check-in data.
Extensions	(none)

Table 3.9: Description for “Log In Teacher Menu Screen” use case

Name	UC-9: Log In Teacher Menu Screen
Actor	Teacher
Preconditions	(none)
Trigger	Teacher wants to log their account in
Postconditions	(none)
Main Scenario	1. 1. Teacher enters their username and password. -- Verify if the username and the password are correct and are identified as Teacher. If they are not, the system shows a message “Failed to Login.” If they are, go to step 2. 2. The system shows a message “Logged In,” then goes to the Teacher Menu Screen.
Extensions	(none)

Table 3.10: Description for “Log In Student Menu Screen” use case

Name	UC-10: Log In Student Menu Screen
Actor	Student
Preconditions	(none)
Trigger	Student wants to log their account in
Postconditions	(none)
Main Scenario	1. Student enters their username and password. - Verify if the username and the password are correct and are identified as Student. If they are not, the system shows a message “Failed to Login.” If they are, go to step 2. 2. The system shows a message “Logged In,” then goes to the Student Menu Screen.
Extensions	(none)

Table 3.11: Description for “Register” use case

Name	UC-11: Register
Actor	User
Preconditions	(none)
Trigger	Student wants to log their account in
Postconditions	(none)
Main Scenario	1. User presses “REGISTER” button. 2. System will navigate to Register Screen. 3. User will enter their Full Name, Email Address, Password, and Phone Number. 4. User will press “Create Account” button. 5. System will push their information into FireBase Authentication.
Extensions	(none)

3.3.2. Sequence Diagram

UC-1: Register Face

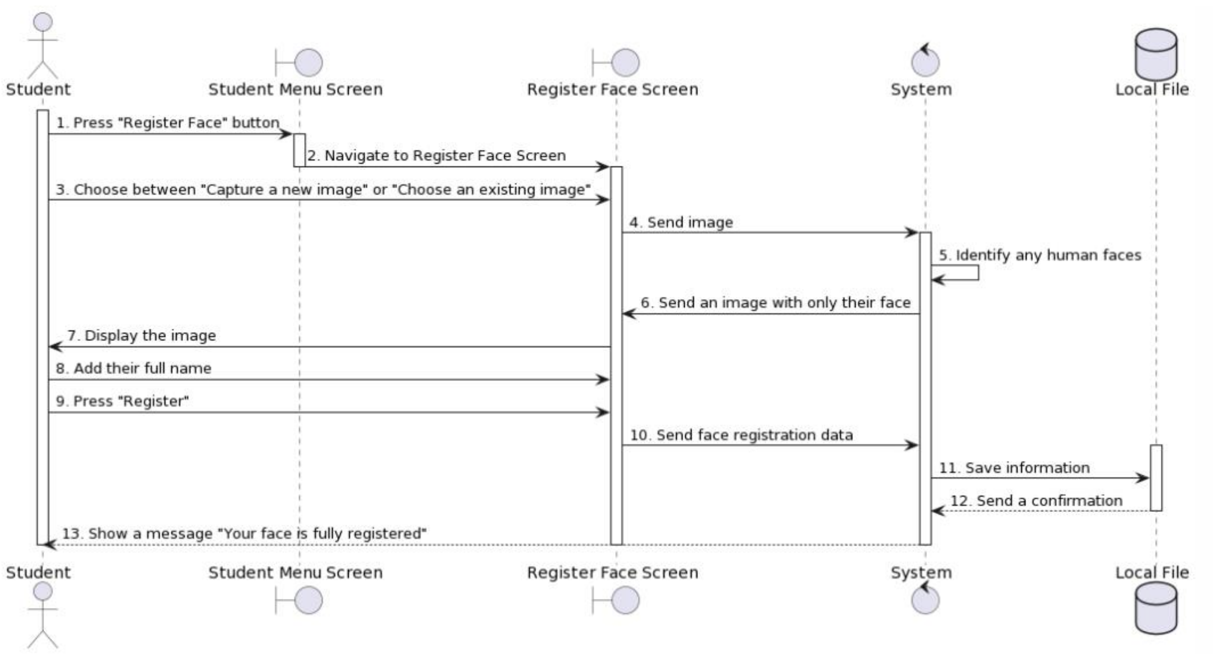

Student
Student Menu Screen
Register Face Screen
System
Local File
1. Press "Register Face" button
2. Navigate to Register Face Screen
3. Choose between "Capture a new image" or "Choose an existing image"
4. Send image
5. Identify any human faces
6. Send an image with only their face
7. Display the image
8. Add their full name
9. Press "Register"
10. Send face registration data
11. Save information
12. Send a confirmation
13. Show a message "Your face is fully registered"

Figure 3.3: Sequence diagram for “Register Face”

Figure 3.3 shows that this use case can only be used by students. In the Student Menu Screen, Student will press a “Register Face” button to start a face registeration process. There are two buttons that allow students to choose whether they want to capture a new image or choose an existing one. After that, the image is sent to the system to locate any human face in the image and show it back to the student. There will be a box for the student to enter their full name and press the “Register” button so that the system can save the data into their device’s local file.

UC-2: Register Voice

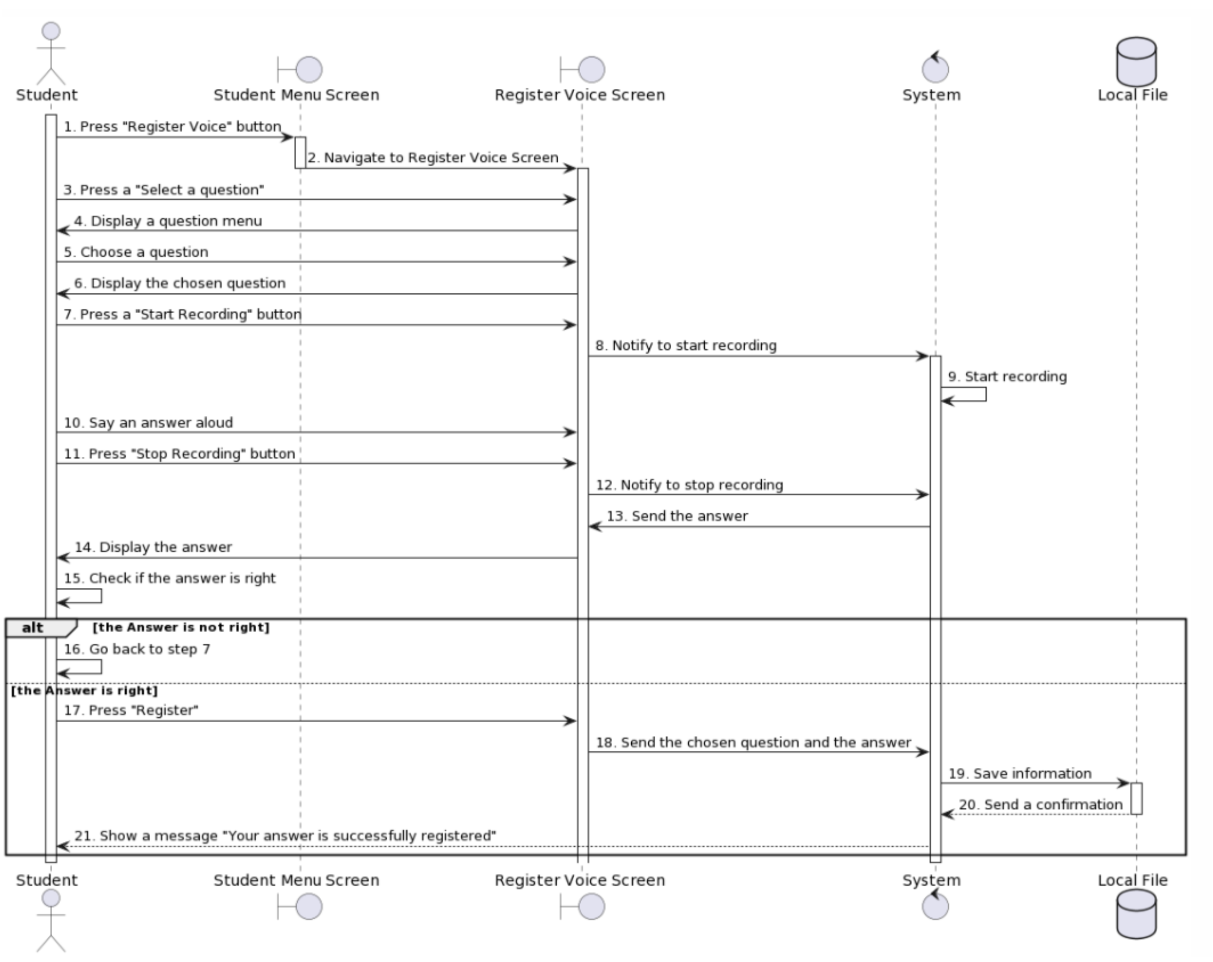

Student
Student Menu Screen
Register Voice Screen
System
Local File
1. Press "Register Voice" button
2. Navigate to Register Voice Screen
3. Press a "Select a question"
4. Display a question menu
5. Choose a question
6. Display the chosen question
7. Press a "Start Recording" button
8. Notify to start recording
9. Start recording
10. Say an answer aloud
11. Press "Stop Recording" button
12. Notify to stop recording
13. Send the answer
14. Display the answer
15. Check if the answer is right
alt
[the Answer is not right]
16. Go back to step 7
[the Answer is right]
17. Press "Register"
18. Send the chosen question and the answer
19. Save information
20. Send a confirmation
21. Show a message "Your answer is successfully registered"
Student
Student Menu Screen
Register Voice Screen
System
Local File

Figure 3.4: Sequence diagram for “Register Voice”

Figure 3.4 shows that this use case can only be used by the student. There will be a drop-down menu for students to select a question they like. After chosing the question, Student presses the “Start Recording” button to start to record. The student will say an answer to that question aloud. Subsequently, the student presses the “Stop Recording” button and if the answer showed in the screen is correct, Student presses the “Register” button. The system will save the question and answer in a local file and then show a message “Your answer is successfully registered.”

UC-3: Check In Using Face Recognition

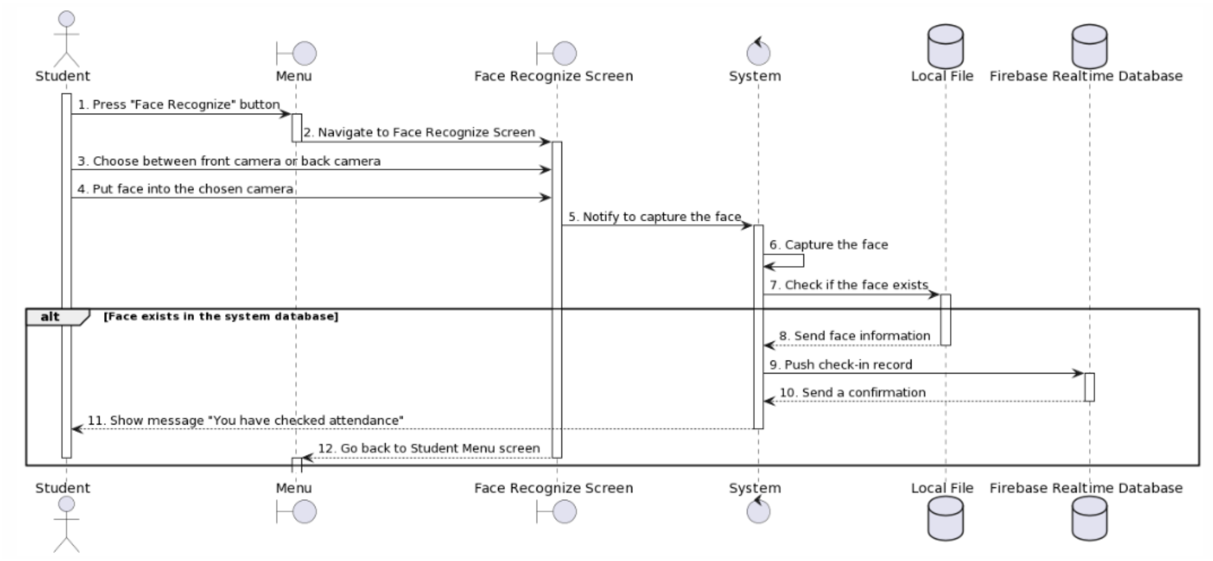

Student
Menu
Face Recognize Screen
System
Local File
Firebase Realtime Database
1. Press "Face Recognize" button
2. Navigate to Face Recognize Screen
3. Choose between front camera or back camera
4. Put face into the chosen camera
5. Notify to capture the face
6. Capture the face
7. Check if the face exists
alt
[Face exists in the system database]
8. Send face information
9. Push check-in record
10. Send a confirmation
11. Show message "You have checked attendance"
12. Go back to Student Menu screen

Figure 3.5: Sequence diagram for “Check In Using Face Recognition”

Figure 3.5 shows that this use case can only be used by the student. There will be a “Face Recognize” button on the Student Menu Screen. After pressing it, the student will be redirected to the Face Recognize Screen, which will open the camera on their device. The student will put their face into the camera, and it will be captured and sent to the system to check if the face exists in the system database. If it does, the system will push a check in record to the Firebase Realtime Database system, show a message “You have checked attendance” to the student, and redirect them back to the Student Menu Screen.

UC-4: Check In Using Voice Recognition

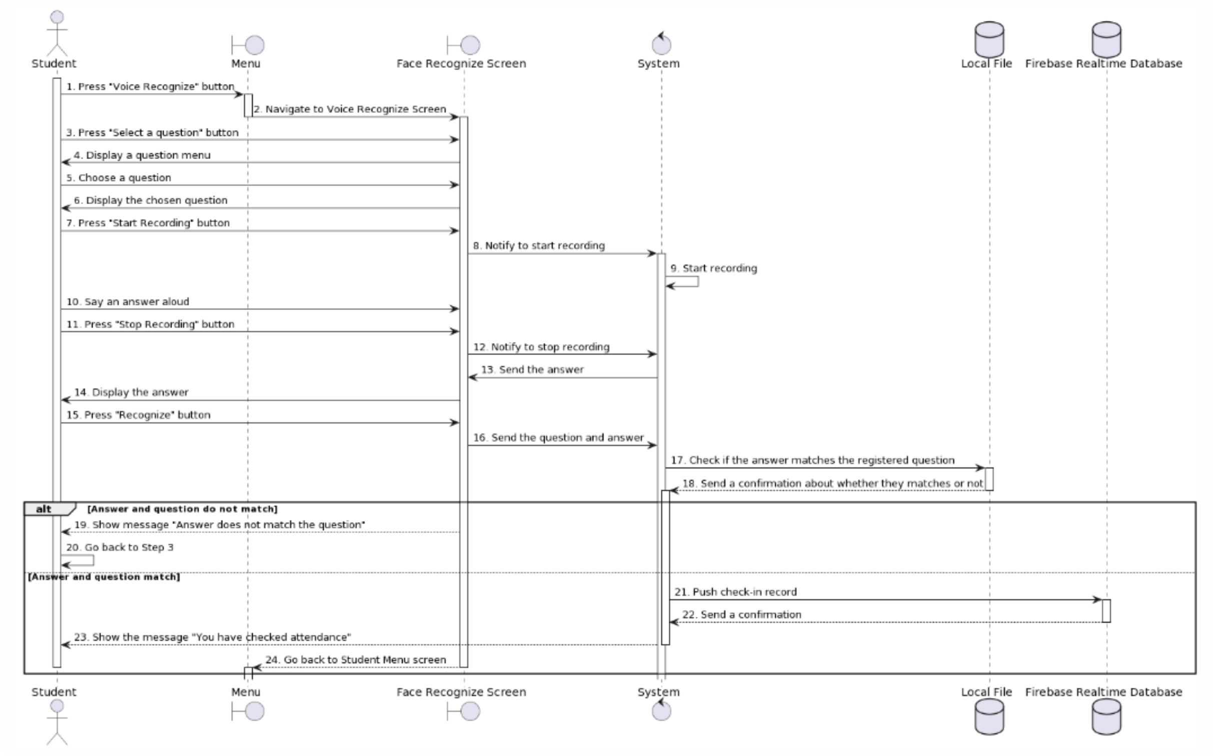

Student
Menu
Face Recognize Screen
System
Local File
Firebase Realtime Database
1. Press "Voice Recognize" button
2. Navigate to Voice Recognize Screen
3. Press "Select a question" button
4. Display a question menu
5. Choose a question
6. Display the chosen question
7. Press "Start Recording" button
8. Notify to start recording
9. Start recording
10. Say an answer aloud
11. Press "Stop Recording" button
12. Notify to stop recording
13. Send the answer
14. Display the answer
15. Press "Recognize" button
16. Send the question and answer
17. Check if the answer matches the registered question
18. Send a confirmation about whether they matches or not
alt
[Answer and question do not match]
19. Show message "Answer does not match the question"
20. Go back to Step 3
[Answer and question match]
21. Push check-in record
22. Send a confirmation
23. Show the message "You have checked attendance"
24. Go back to Student Menu screen

Figure 3.6: Sequence diagram for "Check In Using Voice Recognition"

Figure 3.6 shows that this use case can only be used by the student. There will be a "Voice Recognize" button in the Student Menu Screen. After pressing it, Student will be redirected to Voice Recognize Screen, which will have a dropdown menu for question, a field and a "Start Recording" button for answer. Students choose the question that they have registered before and then press the "Start Recording" button. Now, System will record everything Student said and stop when Student press a "Stop Recording" button. Students can check if the answer is exactly what they have said, and if it does not, they can record it again by pressing the "Start Recording" button one more time. However, if it does, the student can press a "Recognize" button, and the question and answer will be sent to the system to check whether they match each other. If they do not match, System will show a message "Answer does not match question", but if it does, the system will push a check in record to the Firebase Realtime Database, show a message "You have checked attendance" to the student, and redirect them back to the Student Menu Screen.

UC-5: Log Out

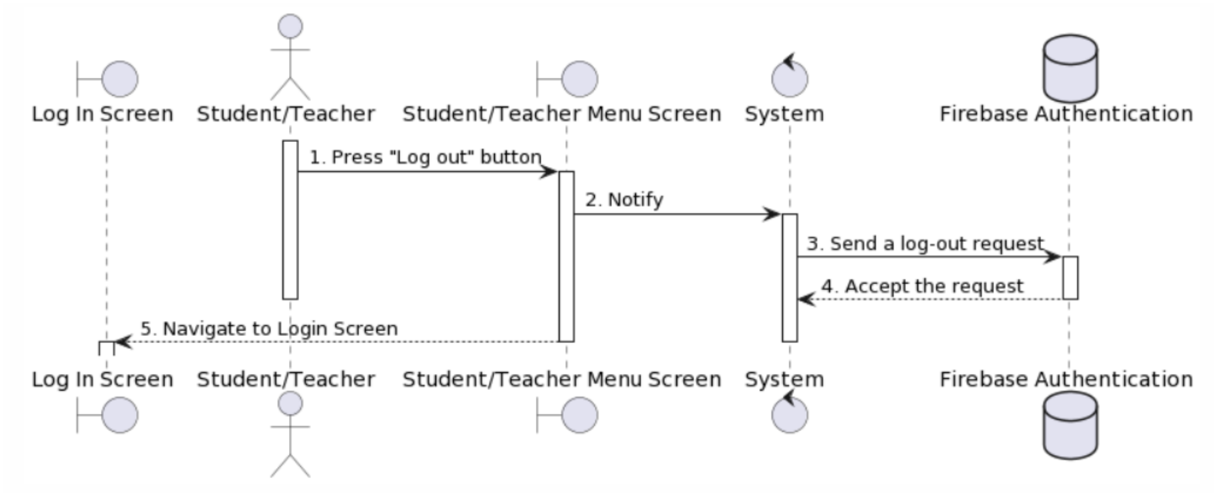

Log In Screen
Student/Teacher
Student/Teacher Menu Screen
System
Firebase Authentication
1. Press "Log out" button
2. Notify
3. Send a log-out request
4. Accept the request
5. Navigate to Login Screen

Figure 3.7: Sequence diagram for “Log Out”

Figure 3.7 shows that this use case can be used only by the Student and Teacher (or user of the application). In the User Screen (Student Menu Screen or Teacher Menu Screen), User will press a “Log Out” button, the log out information will be sent to System and System will send it to Firebase Authentication, which will accept the request to log out. The system then navigates the User to Login Screen.

UC-6: Request Check In Using Face Recognition

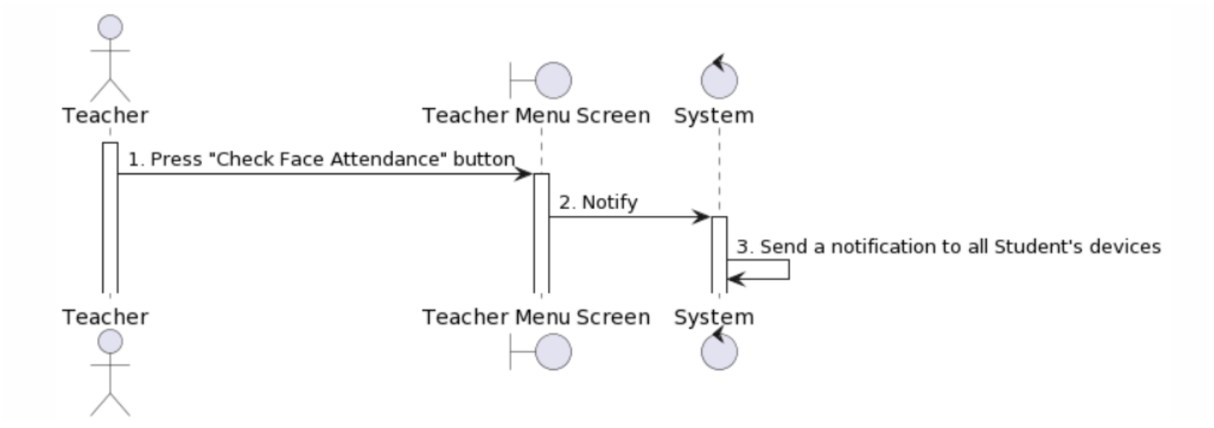

Teacher
Teacher Menu Screen
System
1. Press "Check Face Attendance" button
2. Notify
3. Send a notification to all Student's devices

Figure 3.8: Sequence diagram for “Request Check In Using Face Recognition”

Figure 3.8 shows that this use case can only be used by the teacher. There is a “Check Face Attendance” button in the Teacher Menu Screen. After pressing, the system will be notified and send a notification to all students.

UC-7: Request Check In Using Voice Recognition

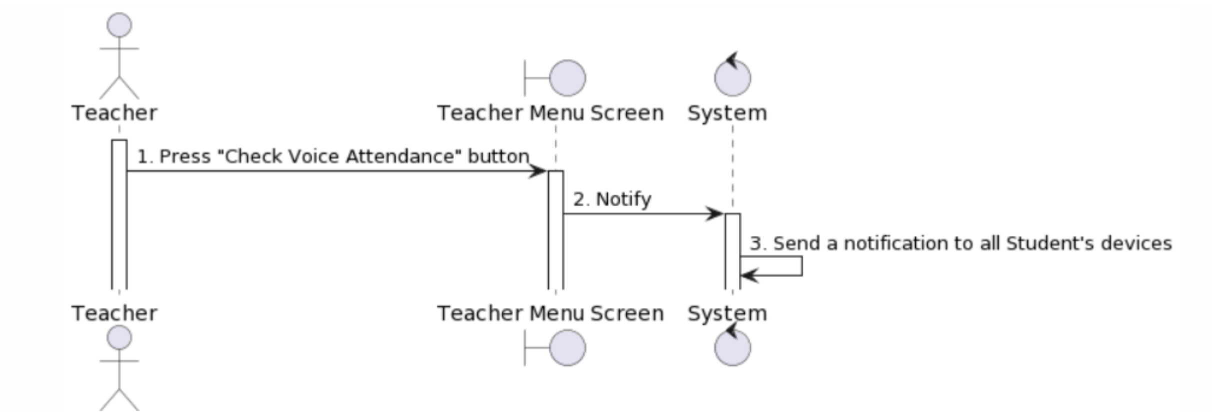

Teacher
Teacher Menu Screen
System
1. Press "Check Voice Attendance" button
2. Notify
3. Send a notification to all Student's devices
Teacher
Teacher Menu Screen
System

Figure 3.9: Sequence diagram for “Request Check In Using Voice Recognition”

Figure 3.9 shows that this use case can only be used by the teacher. There is a “Check Voice Attendance” button in the Teacher Menu Screen. After pressing, the system will be notified and send a notification to all students.

UC-8: View Attendance List

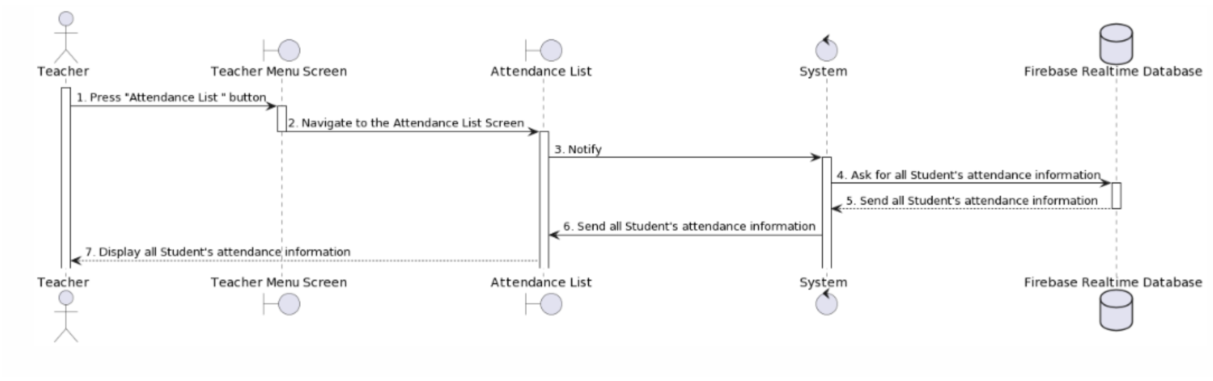

Teacher
Teacher Menu Screen
Attendance List
System
Firebase Realtime Database
1. Press "Attendance List " button
2. Navigate to the Attendance List Screen
3. Notify
4. Ask for all Student's attendance information
5. Send all Student's attendance information
6. Send all Student's attendance information
7. Display all Student's attendance information
Teacher
Teacher Menu Screen
Attendance List
System
Firebase Realtime Database

Figure 3.10: Sequence diagram for “View Attendance List”

Figure 3.10 shows that this use case can only be used by the teacher. There will be a “Attendance List” button in the Teacher Menu Screen. After pressing it, the teacher will be redirected to the Attendance List Screen. The system will be notified and send a request to the Firebase Realtime Database for student attendance information. After receiving the information, the system shows it on the Attendance List Screen.

UC-9: Log In Teacher Menu Screen

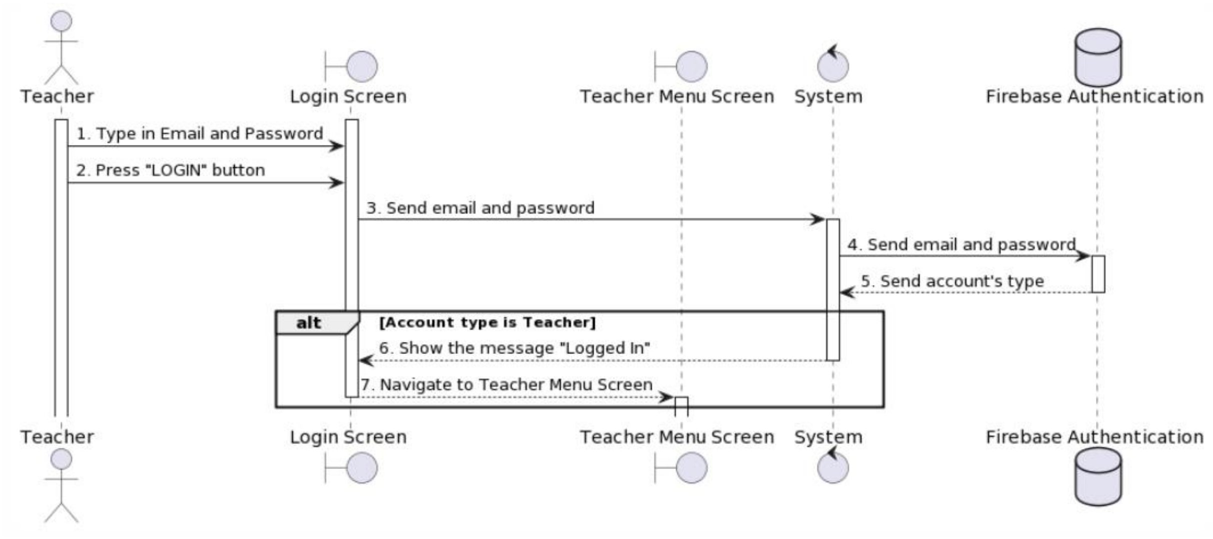

Teacher
Login Screen
Teacher Menu Screen
System
Firebase Authentication
1. Type in Email and Password
2. Press "LOGIN" button
3. Send email and password
4. Send email and password
5. Send account's type
alt
[Account type is Teacher]
6. Show the message "Logged In"
7. Navigate to Teacher Menu Screen

Figure 3.11: Sequence diagram for “Log In Teacher Screen”

Figure 3.11 shows that this use case can only be used by the teacher. Teacher will type in their email address and password then press a “LOGIN” button. This information is sent to and checked by the system. If the email address and password belong to Teacher, System will show a message “Logged In” and navigate Teacher to Teacher Menu Screen.

UC-10: Log In Student Menu Screen

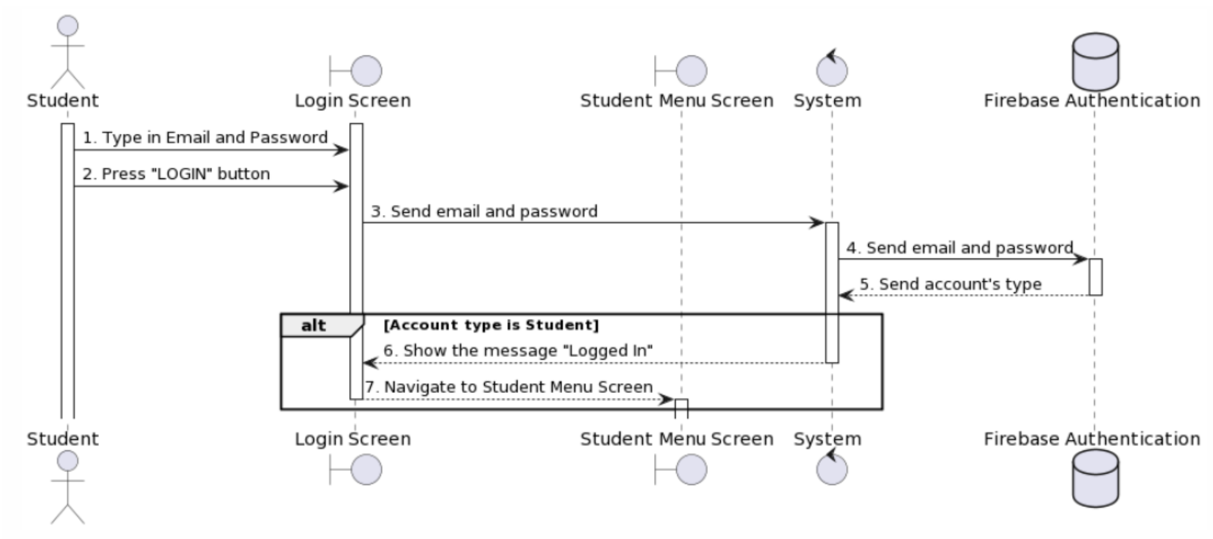

Student
Login Screen
Student Menu Screen
System
Firebase Authentication
1. Type in Email and Password
2. Press "LOGIN" button
3. Send email and password
4. Send email and password
5. Send account's type
alt
[Account type is Student]
6. Show the message "Logged In"
7. Navigate to Student Menu Screen

Figure 3.12: Sequence diagram for “Log In Student Screen”

Figure 3.12 shows that this case can only be used by the student. Student will type in their email address and password then press a "LOGIN" button. This information is sent to and checked by the system. If the email address and password belong to Student, System will show a message "Logged In" and navigate the student to the Student Menu Screen.

UC-11: Register

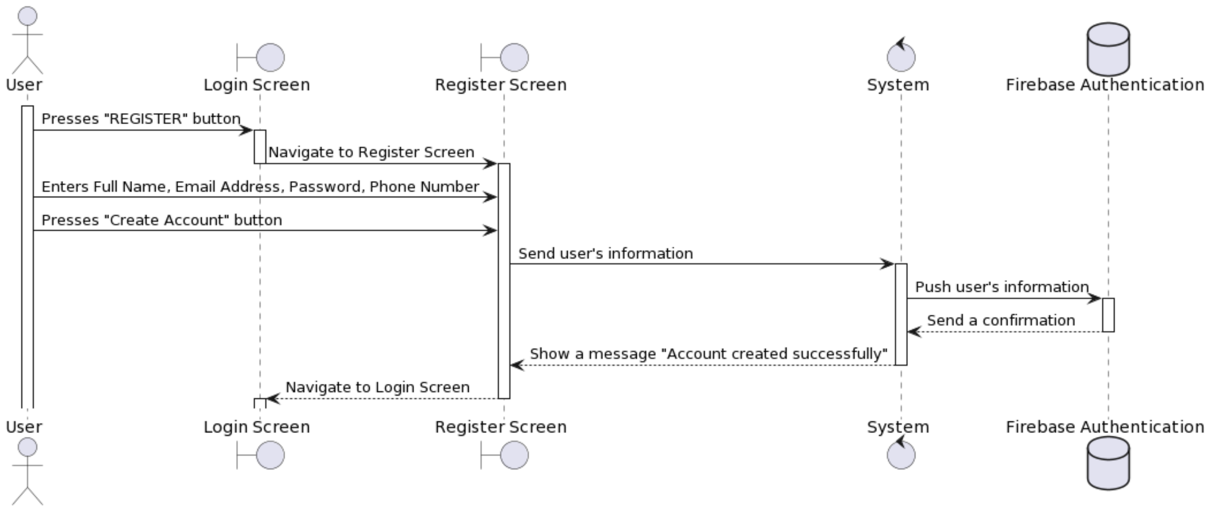

User
Login Screen
Register Screen
System
Firebase Authentication
Presses "REGISTER" button
Navigate to Register Screen
Enters Full Name, Email Address, Password, Phone Number
Presses "Create Account" button
Send user's information
Push user's information
Send a confirmation
Show a message "Account created successfully"
Navigate to Login Screen

Figure 3.13: Sequence diagram for "Register"

Figure 3.13 shows that this use case can be used by anyone who uses the application. In the Login Screen, User will press a "Register" button and they will be redirected to Register Screen which has 4 boxes such as "Full Name," "Email Address," "Password," and "Phone Number." After filling all necessary information, User presses a "Create Account" button and all their information will be stored and handled by Firebase Authentication. The system will show the message "Account created successfully" to the user.

3.3.3. System Workflow

Register Face

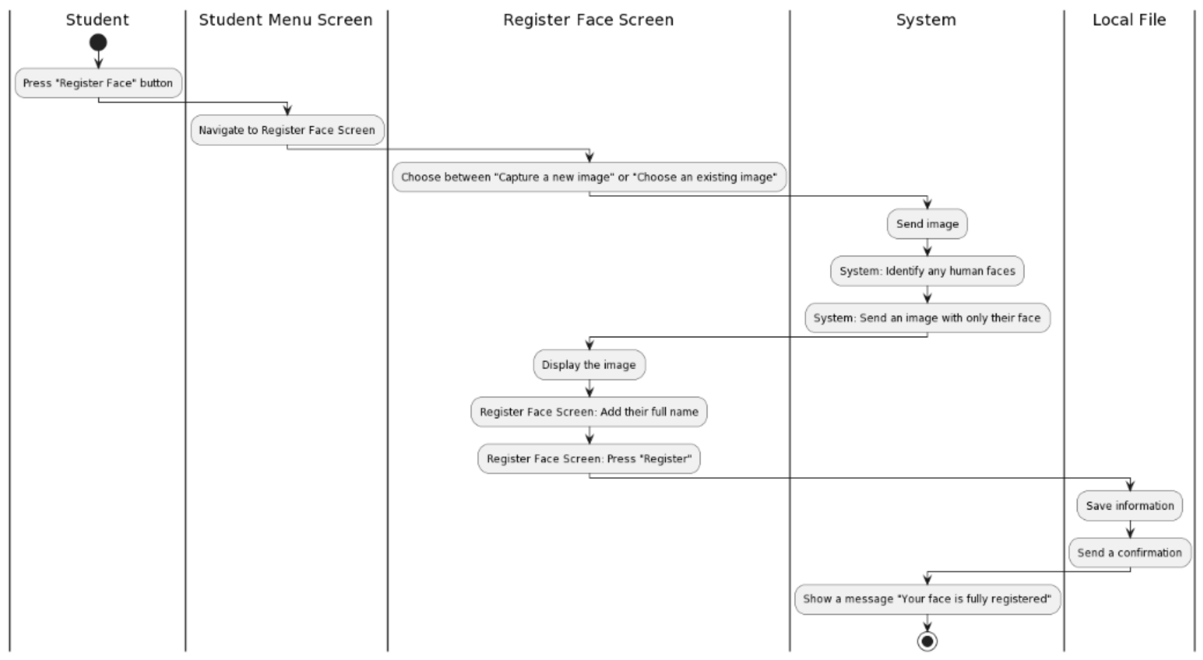

Student
Student Menu Screen
Register Face Screen
System
Local File
Press "Register Face" button
Navigate to Register Face Screen
Choose between "Capture a new image" or "Choose an existing image"
Send image
System: Identify any human faces
System: Send an image with only their face
Display the image
Register Face Screen: Add their full name
Register Face Screen: Press "Register"
Save information
Send a confirmation
Show a message "Your face is fully registered"

Figure 3.14: Workflow diagram for “Register Face” sequence diagram

The function is designed to allow the Student to register their face into the system as follows: the Student will press the “Register Face” button in the “Student Menu Screen” to navigate to the “Register Face” Screen. At that point, the student inputs an image by capturing a new image or choosing an existing one. The image is then sent to and handled by the system. After detecting any human face, the system sends the face image to the Student and requests them to input their full name. Subsequently, the system saves the information into a local file.

Register Voice

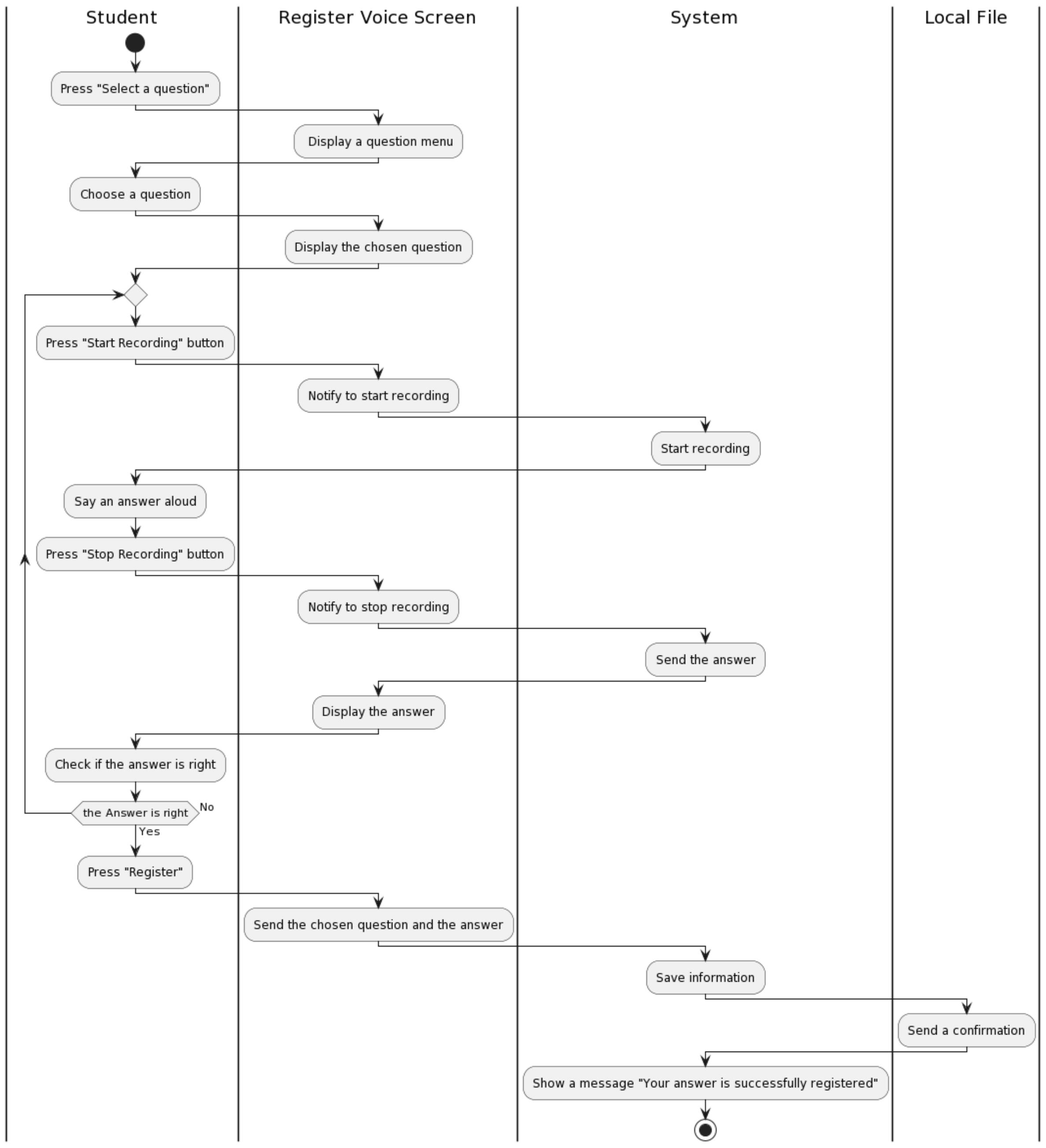
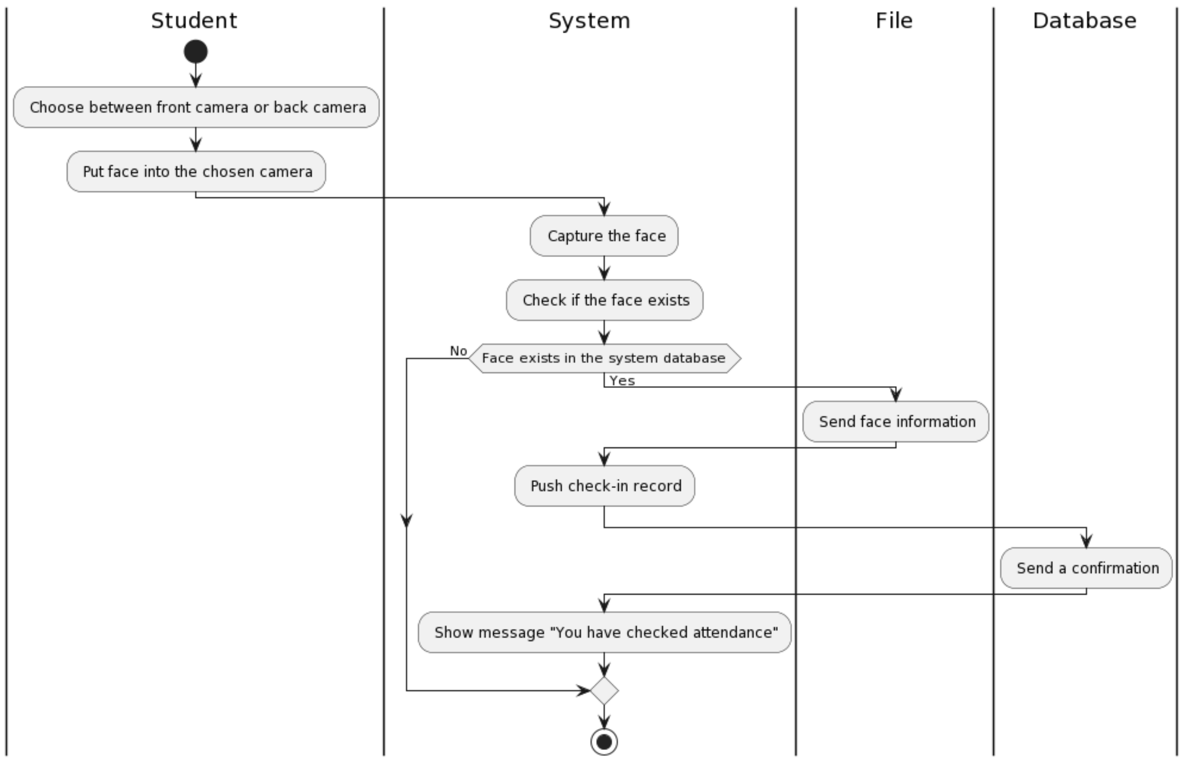

Student
Register Voice Screen
System
Local File
Press "Select a question"
Display a question menu
Choose a question
Display the chosen question
Press "Start Recording" button
Notify to start recording
Start recording
Say an answer aloud
Press "Stop Recording" button
Notify to stop recording
Send the answer
Display the answer
Check if the answer is right
the Answer is right
No
Yes
Press "Register"
Send the chosen question and the answer
Save information
Send a confirmation
Show a message "Your answer is successfully registered"

Figure 3.15: Workflow diagram for “Register Voice” sequence diagram

The function is designed to allow the Student to register their face into the system as follows: the Student will press the “Register Voice” button in the “Student Menu Screen” to navigate to the “Register Voice” Screen. At that point, the Student chooses a question and records an answer. After the “Register” button pressed, the system will save the question and answer into a local file.

Check In Using Face Recognition



Student
System
File
Database
Choose between front camera or back camera
Put face into the chosen camera
Capture the face
Check if the face exists
No
Face exists in the system database
Yes
Send face information
Push check-in record
Send a confirmation
Show message "You have checked attendance"

Figure 3.16: Workflow diagram for “Check In Using Face Recognition” sequence diagram

The function is designed to allow the Student to register their face in the system as follows: the student inputs their face into the front camera or back camera of their device. After recognizing a human face, the system checks it in.

Check In Using Voice Recognition

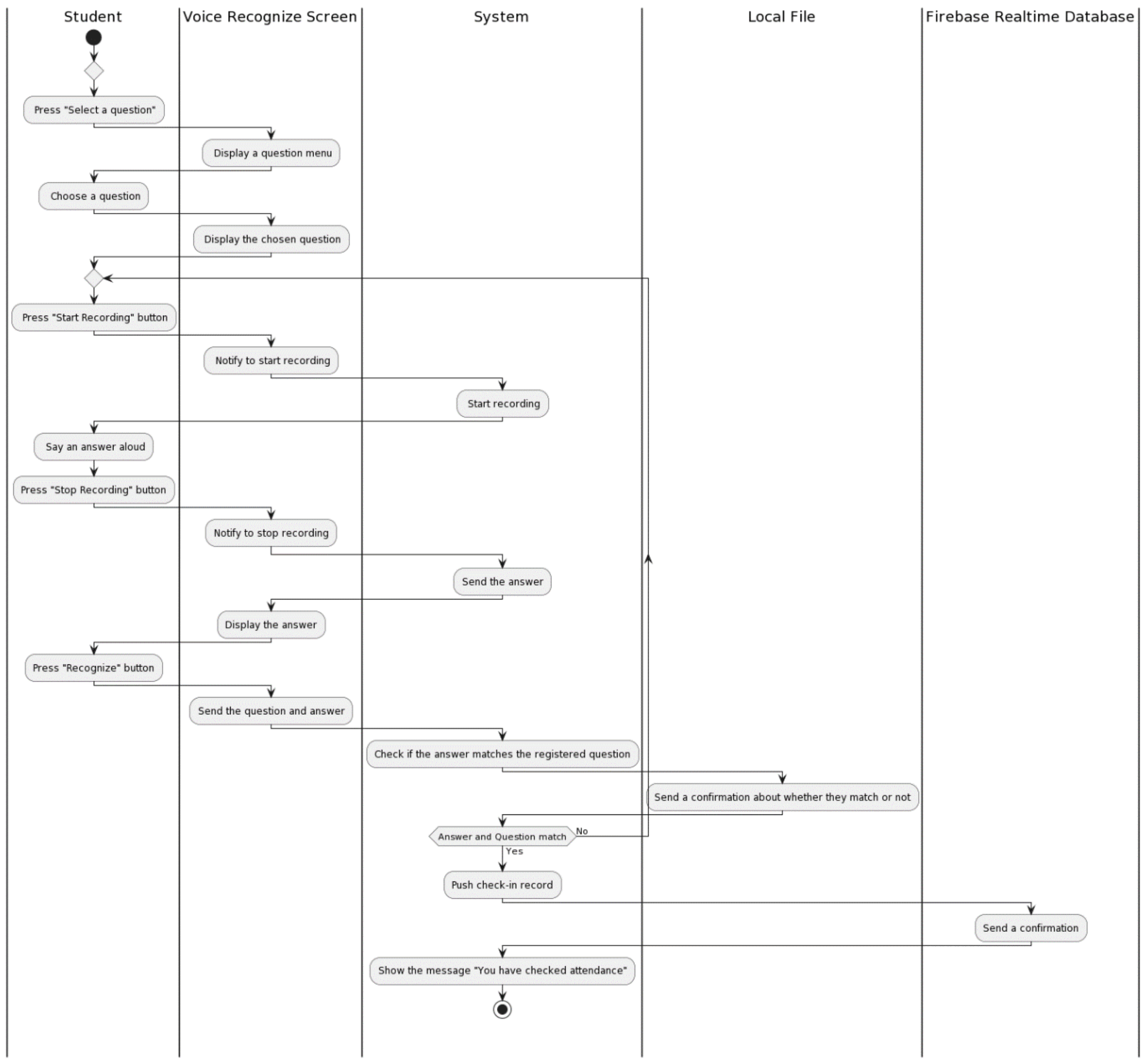

Student
Voice Recognize Screen
System
Local File
Firebase Realtime Database
Press "Select a question"
Display a question menu
Choose a question
Display the chosen question
Press "Start Recording" button
Notify to start recording
Start recording
Say an answer aloud
Press "Stop Recording" button
Notify to stop recording
Send the answer
Display the answer
Press "Recognize" button
Send the question and answer
Check if the answer matches the registered question
Send a confirmation about whether they match or not
Answer and Question match
No
Yes
Push check-in record
Send a confirmation
Show the message "You have checked attendance"

Figure 3.17: Workflow diagram for “Check In Using Voice Recognition” sequence diagram

The function is designed to allow the Student to register their face in the system as follows: the Student chooses a question and inputs their answer to that question. If the question is not registered, the system requests that the Student choose another one. After the question-and-answer match, the system checks the Student in.

Log Out

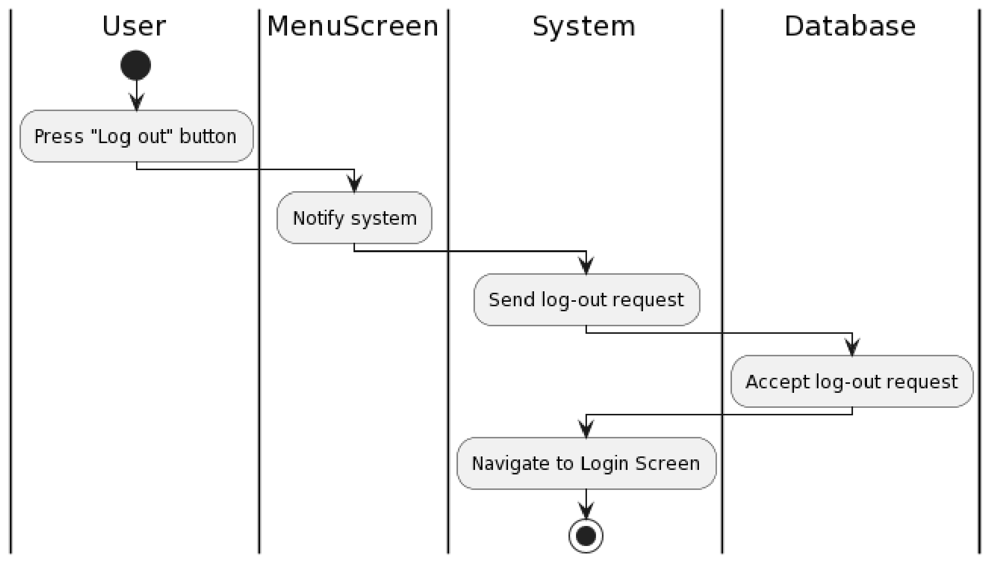
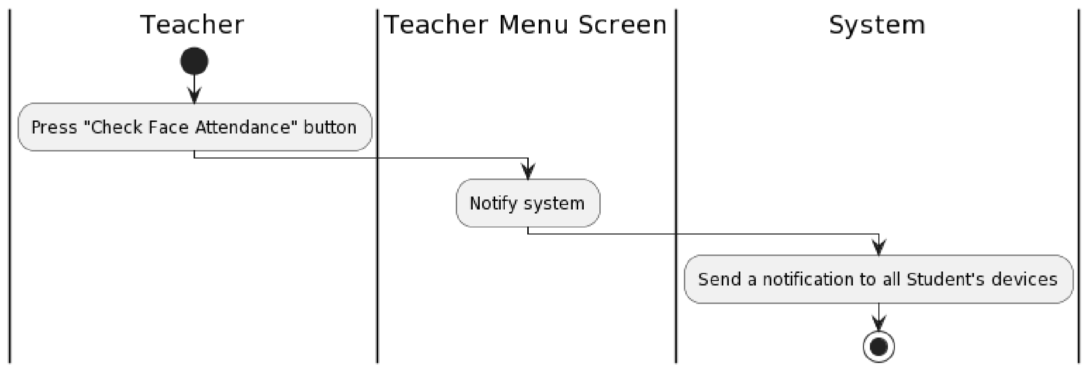


Figure 3.18: Workflow diagram for "Log Out" sequence diagram

The function is designed to allow the User to log out of the system by pressing the "Log out" button on the Menu Screen.

Request Check In Using Face Recognition



Teacher
Teacher Menu Screen
System
Press "Check Face Attendance" button
Notify system
Send a notification to all Student's devices

Figure 3.19: Workflow diagram for "Request Check In Using Face Recognition" sequence diagram

The function is designed to allow the Teacher to request their student to check in using face recognition. After pressing the "Check Face Attendance" button, a push notification will be sent to the Student's devices.

Request Check In Using Voice Recognition

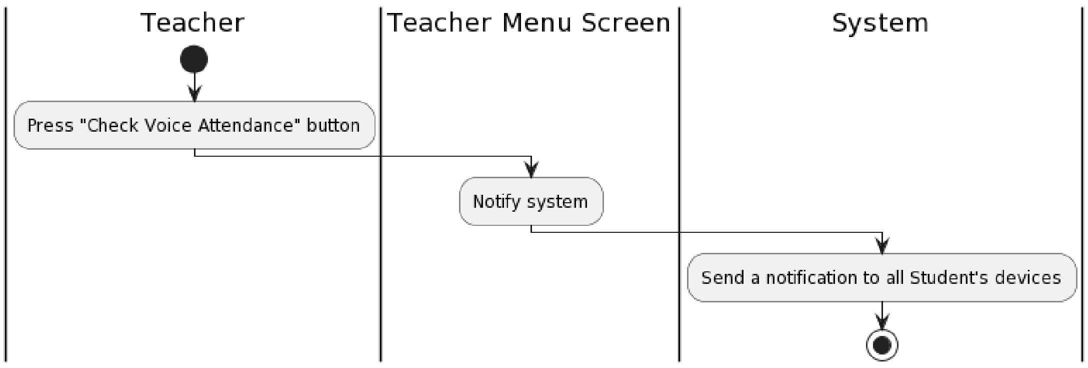

Teacher
Teacher Menu Screen
System
Press "Check Voice Attendance" button
Notify system
Send a notification to all Student's devices

Figure 3.20: Workflow diagram for "Request Check In Using Voice Recognition" sequence diagram

The function is designed to allow the Teacher to request their student to check in using voice recognition. After pressing the "Check Voice Attendance" button, a push notification will be sent to the Student's devices.

View Attendance List

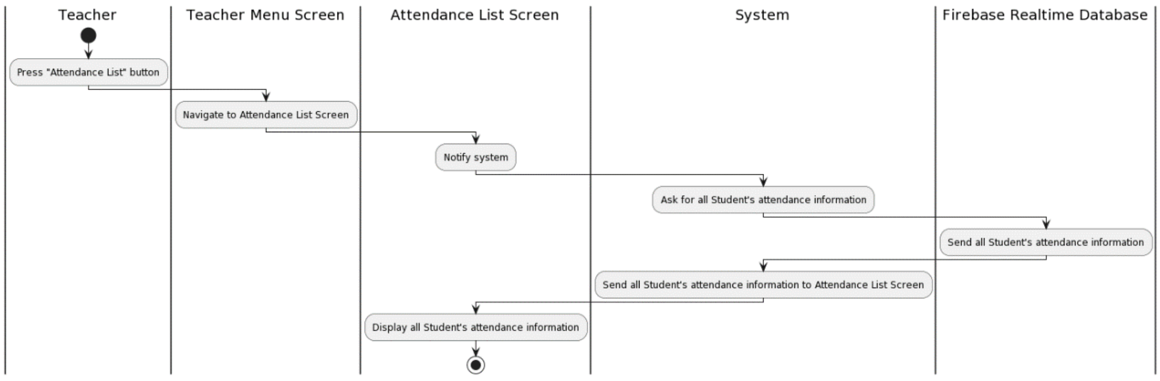

Teacher
Teacher Menu Screen
Attendance List Screen
System
Firebase Realtime Database
Press "Attendance List" button
Navigate to Attendance List Screen
Notify system
Ask for all Student's attendance information
Send all Student's attendance information
Send all Student's attendance information to Attendance List Screen
Display all Student's attendance information

Figure 3.21: Workflow diagram for "View Attendance List" sequence diagram

The function is designed to allow the Teacher to view the current attendance list. After pressing the "Attendance List" button, the Teacher will be navigated to the "Attendance List" Screen, which displays all information.

Log In Teacher Menu Screen

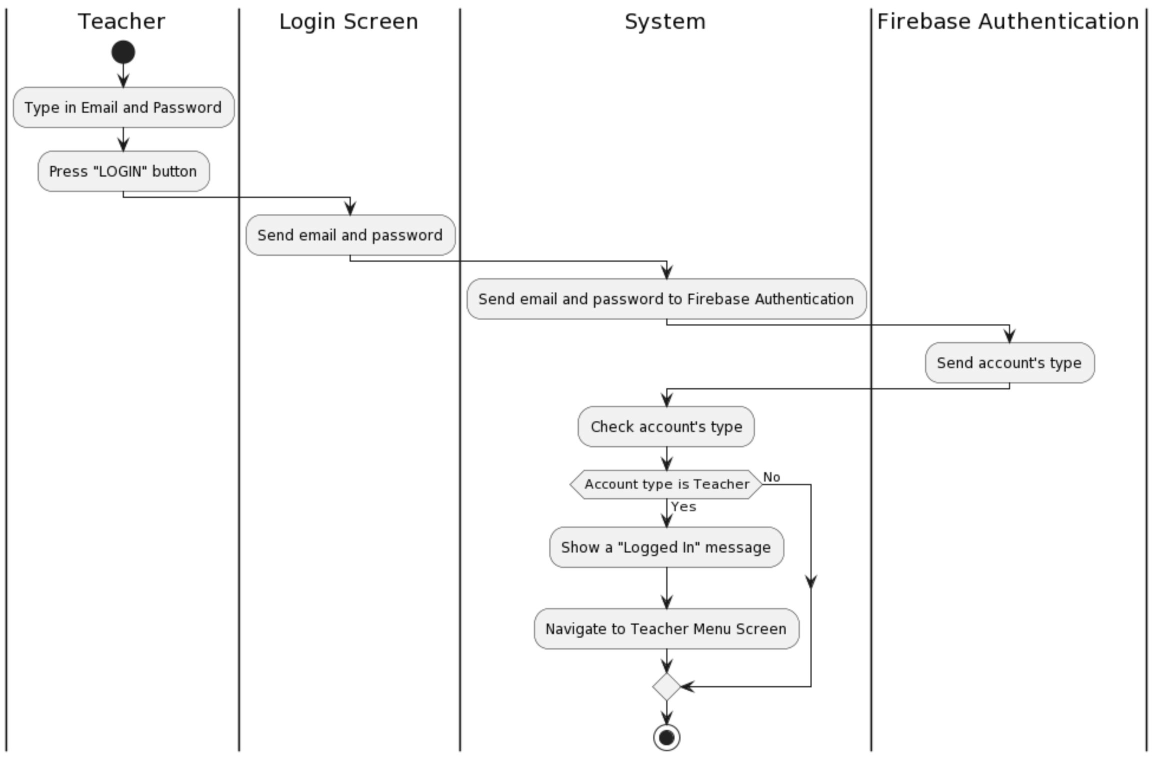


Figure 3.22: Workflow diagram for "Log In Teacher Menu Screen" sequence diagram

The function is designed to allow the Teacher to log in to the system using their Teacher email. After inputing their Teacher email and password, the system will navigate them to the "Teacher Menu" Screen.

Log In Student Menu Screen

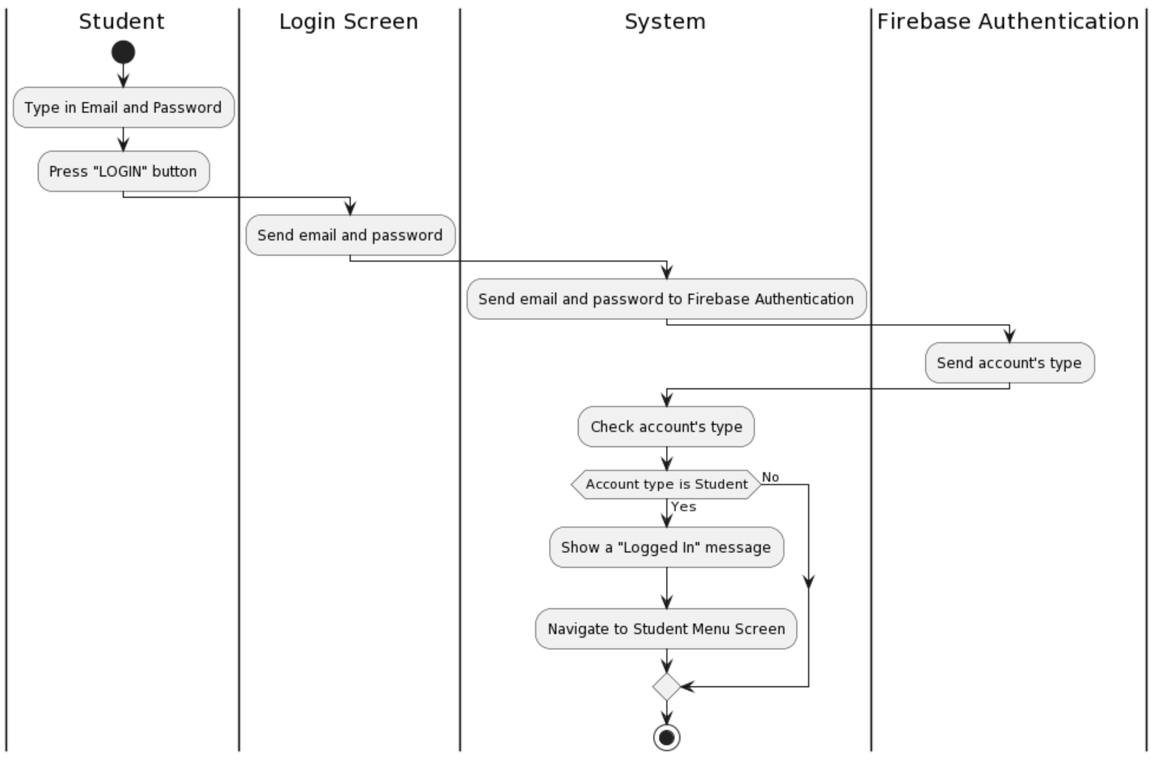


Figure 3.23: Workflow diagram for “Log In Student Menu Screen” sequence diagram

The function is designed to allow the Student to log in to the system using their Student email. After inputting their Student emails and passwords, the system navigates them to the "Student Menu” Screen.

Register

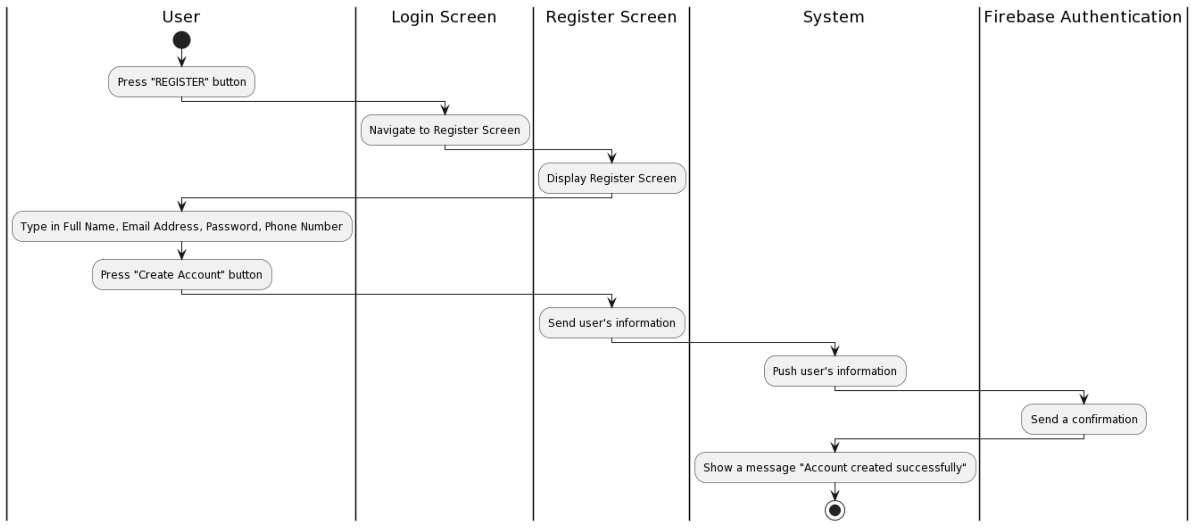


Figure 3.24: Workflow diagram for "Register" sequence diagram

This function is designed to allow the User to register a new account using their email address. After fulfilling all the required fields and pressing the "Create Account" button, the system checks the user's email address to identify the type (Student or Teacher) of account. Subsequently, the system saves the new account into the Firebase Authentication database and navigates it to the correct "Menu" Screen.

3.3.4. ERD Diagram

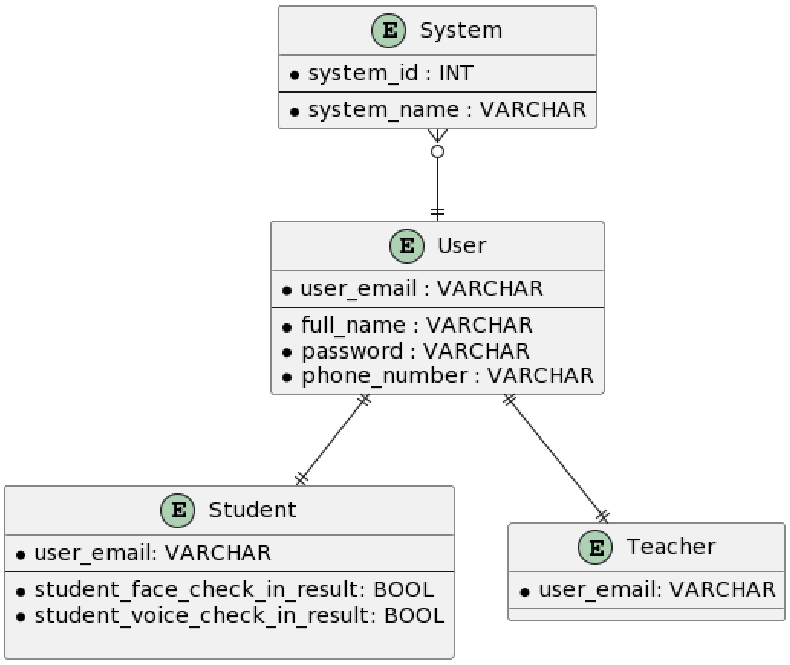


Figure 3.25: ERD diagram for the application

The System entity will have two fields, system_id and system_name, to link the system to Firebase. The User entity has four main fields: user_email, full_name, password, and phone_number. Based on user_email, the User will be divided into two different types: Teacher and Student, which have two other special fields: student_face_check_in_result and student_voice_check_in_result.

3.3.5. Class Diagram

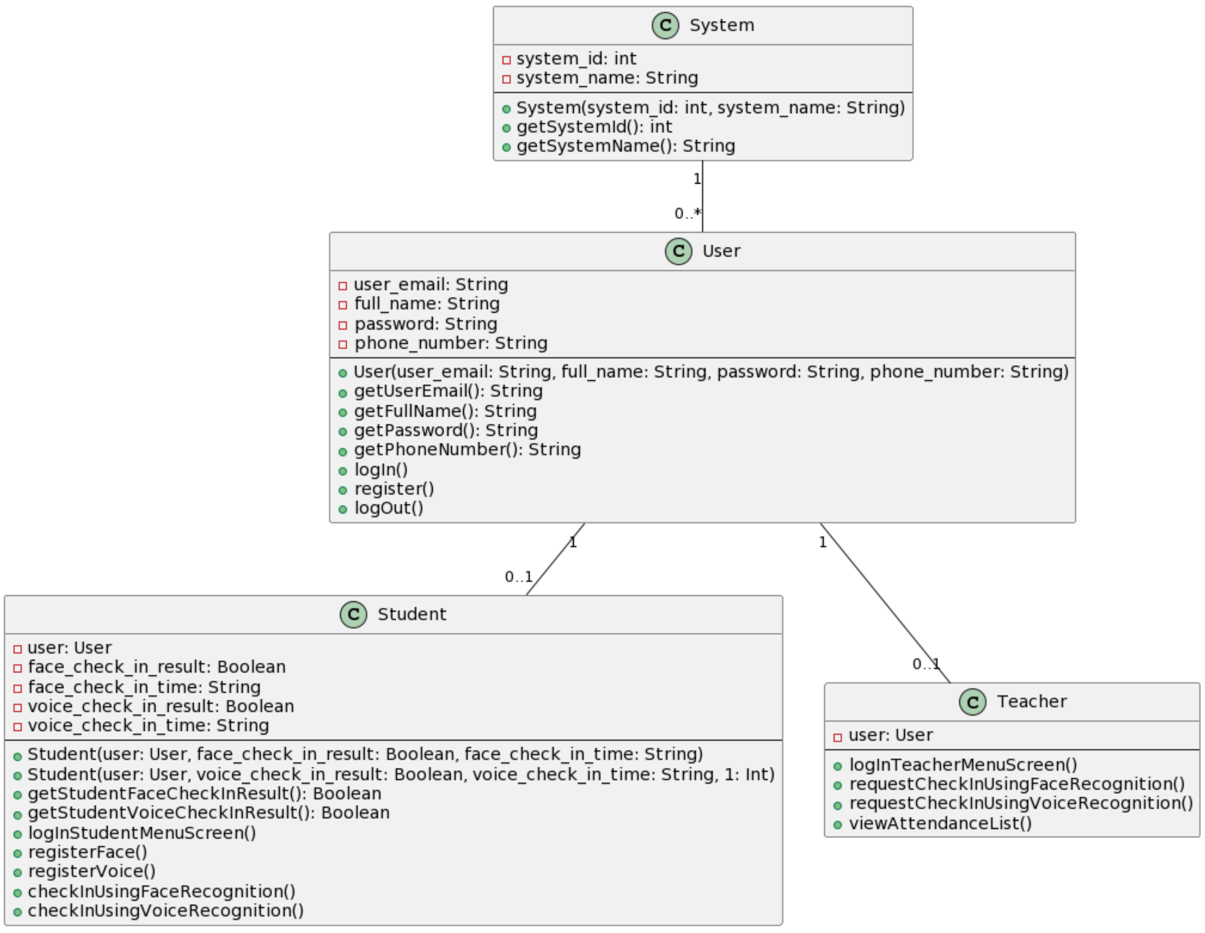

System
system_id: int
system_name: String
System(system_id: int, system_name: String)
getSystemId(): int
getSystemName(): String
1
0..*
User
user_email: String
full_name: String
password: String
phone_number: String
User(user_email: String, full_name: String, password: String, phone_number: String)
getUserEmail(): String
getFullName(): String
getPassword(): String
getPhoneNumber(): String
logIn()
register()
logOut()
1
0..1
1
0..1
Student
user: User
face_check_in_result: Boolean
face_check_in_time: String
voice_check_in_result: Boolean
voice_check_in_time: String
Student(user: User, face_check_in_result: Boolean, face_check_in_time: String)
Student(user: User, voice_check_in_result: Boolean, voice_check_in_time: String, 1: Int)
getStudentFaceCheckInResult(): Boolean
getStudentVoiceCheckInResult(): Boolean
logInStudentMenuScreen()
registerFace()
registerVoice()
checkInUsingFaceRecognition()
checkInUsingVoiceRecognition()
Teacher
user: User
logInTeacherMenuScreen()
requestCheckInUsingFaceRecognition()
requestCheckInUsingVoiceRecognition()
viewAttendanceList()

Figure 3.26: Class diagram for the application

This system, described in Figure 3.26, has the following functions: the User can log in, log out, and register a new account; the Student can register their face, register their voice, check in using face recognition, check in using voice recognition; and the Teacher can request their Student to check in using face or voice recognition, and view the attendance list.

3.3.6. User Interface Design

These below figures are the mockup user interface which have been designed for the application

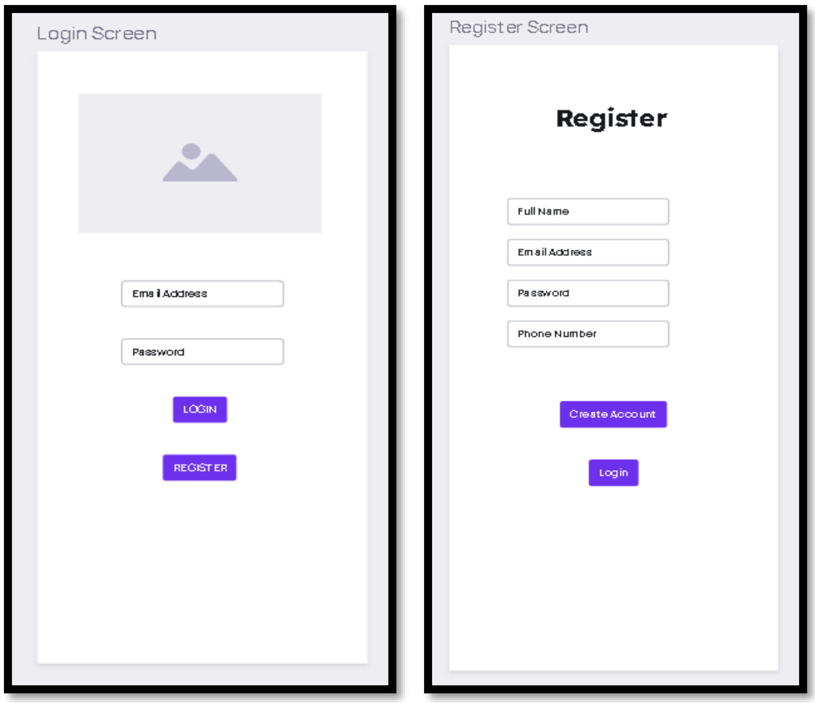


Figure 3.27: User Interface Design for Login and Register Screen

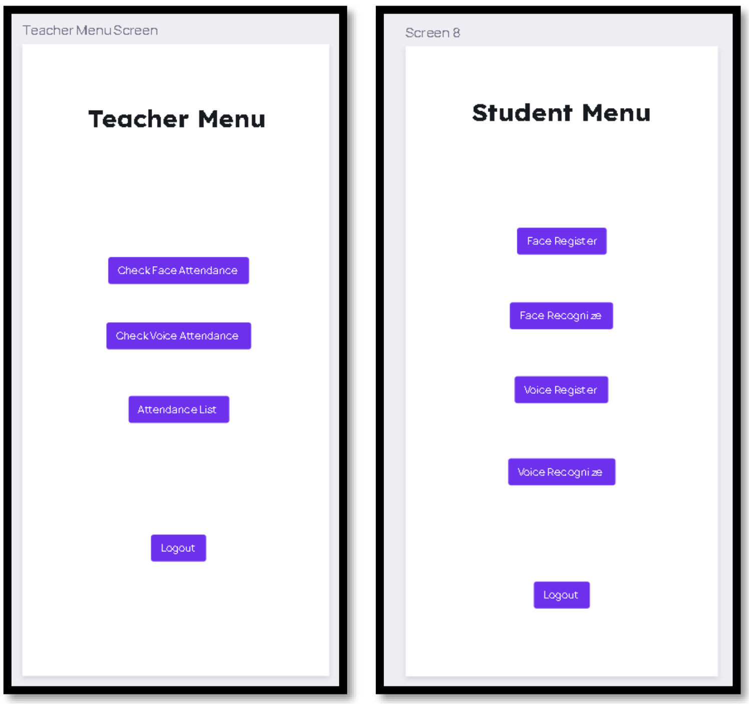


Figure 3.28: User Interface Design for Teacher Menu and Student Menu Screen

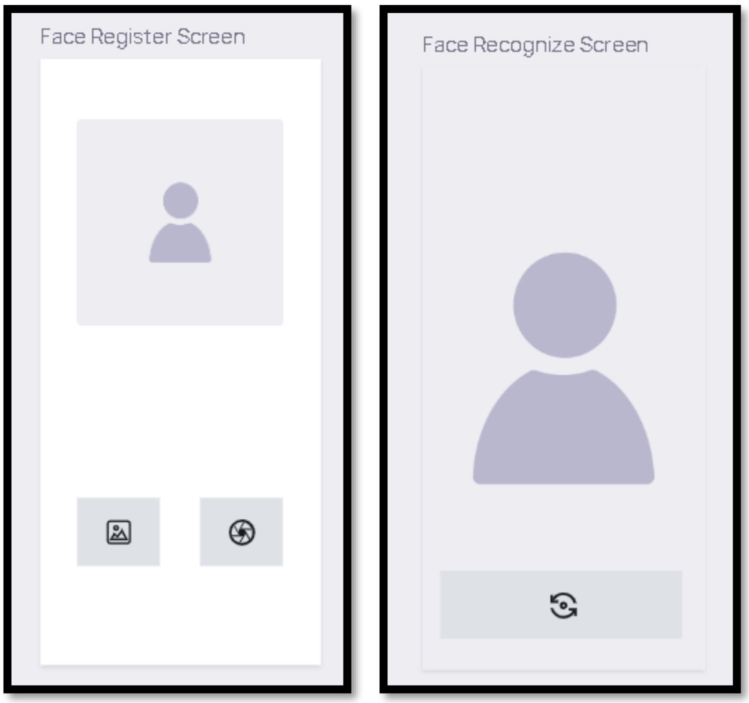


Figure 3.29: User Interface Design for Face Register and Face Recognize Screen

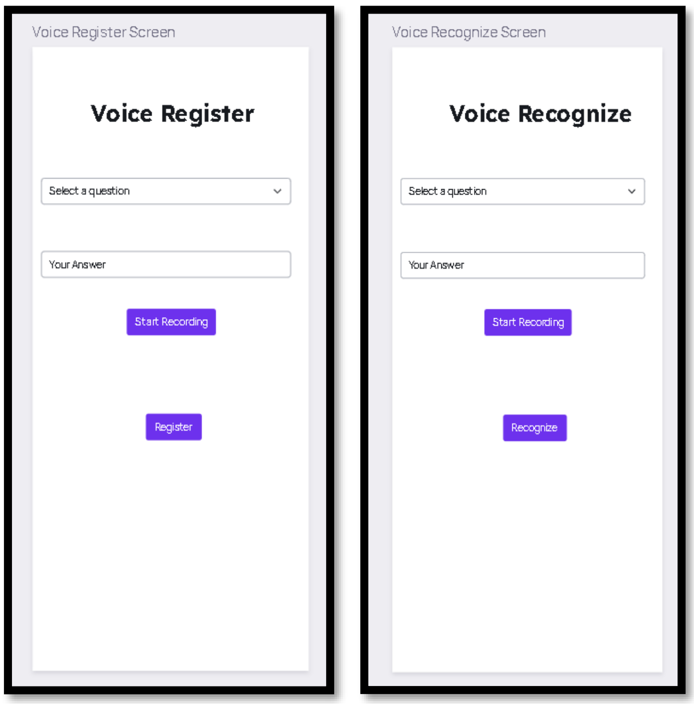


Figure 3.30: User Interface Design for Voice Register and Voice Recognize Screen

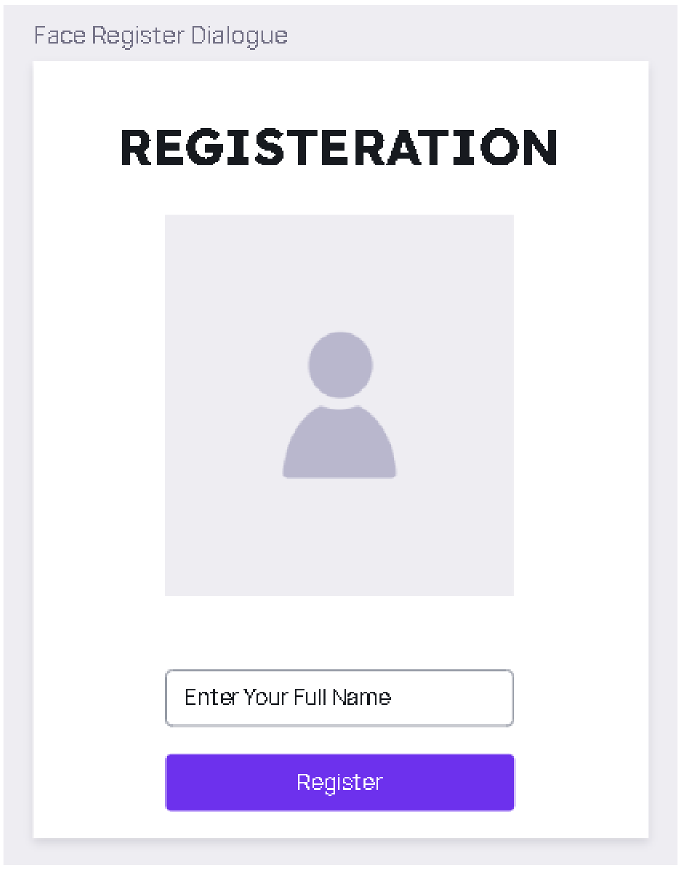


Figure 3.31: User Interface Design for Face Register Dialogue

CHAPTER 4

IMPLEMENT AND RESULTS

4.1. Implement

All critical tasks, such as developing and testing an Android application, XML files, and Java files, are completed on personal computers. All user information data are stored in the Firebase Realtime Database and may be collected and updated later. Additionally, Firebase Authentication is utilized in the registration and login sections, such as registering new users, updating existing users, and deleting users. Furthermore, Firebase Machine Learning can be utilized to detect human faces in a picture.

The program will have two types of users, teachers and students, and it will provide some basic features. For example, a teacher may primarily request to check attendance and examine the attendance list, while a student can register their face and voice in the device's local files and utilize them to check attendance. Consequently, the system has the following features:

Table 4.1: List of functions of the application

No.	Features	Details
1	Different login permissions for Teacher and Student	Accounts will be assigned different permissions by the attributes such as isTeacher or isStudent to distinguish which account is will be navigated to Student/Teacher Menu Screen.
2	Register face	Student can input their face into their device’s local file.

3	Register voice	Student can input their question and voice anwser into their device's local file.
4	Check in using face recognition	Student can input their face and System will compare it to the saved data to check them in.
5	Check in using voice recognition	Student can input the question and their voice answer and System will compare it to the saved data to check them in.
5	Request face/voice check attendance	Teacher can make a request to all Student to check their attendance using their face/voice.
6	View attendance list	Teacher can view a student attendance list.
7	Register	User can register a new account using their email address.

4.1.1. Implementation

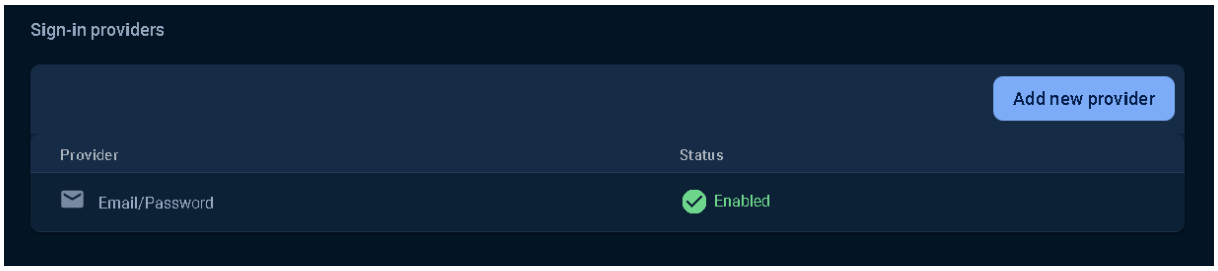
4.1.1.1. Setting up

First, Android Studio must be installed to begin developing an Android application. In addition, to connect the application to Firebase, a Firebase account and new project in the console must be created. Then, a project ID is generated based on the project's name. The process of linking the Firebase project and Android application is described in a guide document [25].

Firebase Authentication

Firebase Authentication is a backend service that provides a simple method for authenticating users in the application. In this application, an email address is an authentication credential for the user, and it is passed to the Firebase Authentication SDK to be verified.

The addition of Firebase Authentication to the Android Application was demonstrated in [26].



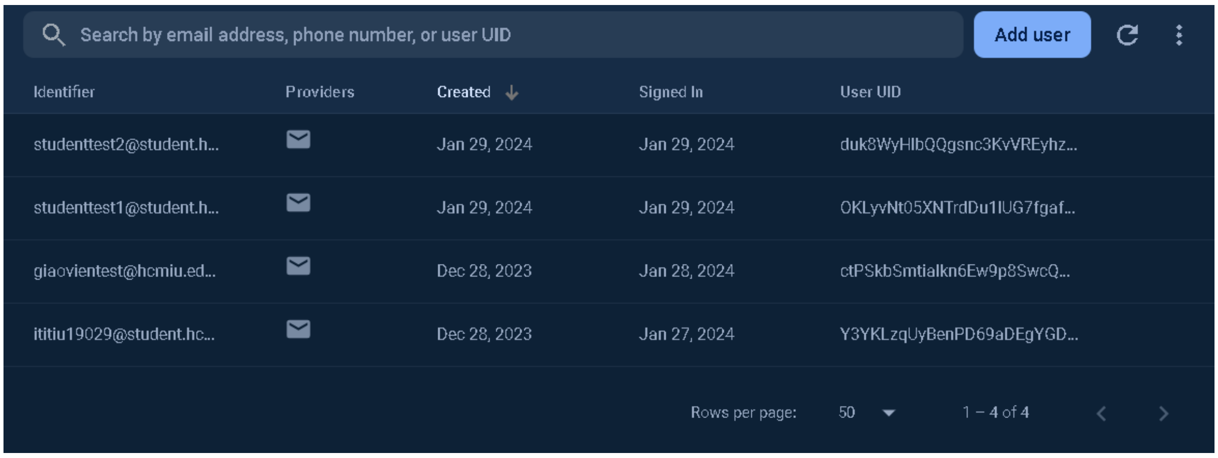
Sign-in providers

Add new provider

Provider	Status
Email/Password	Enabled

Figure 4.1: Firebase Authentication Sign-in Providers

Figure 4.1 shows that the email/password method is used for the application, and another method can be added.



Search by email address, phone number, or user UID

Add user

Identifier	Providers	Created	Signed In	User UID
studenttest2@student.h...	✉	Jan 29, 2024	Jan 29, 2024	duk8WyHlbQQgsnc3KvVREyhz...
studenttest1@student.h...	✉	Jan 29, 2024	Jan 29, 2024	OKLyvNt05XNTrdDu1lUG7fgaf...
giaovientest@hcmiu.ed...	✉	Dec 28, 2023	Jan 28, 2024	ctPSkbSmtialkn6Ew9p8SwcQ...
ititiu19029@student.hc...	✉	Dec 28, 2023	Jan 27, 2024	Y3YKLzqUyBenPD69aDEgYGD...

Rows per page: 50 1 – 4 of 4

Figure 4.2: Firebase Authentication User Management

The above figure shows how user information is stored and managed. The user's UID is automatically generated by Firebase Authentication. Furthermore, adding new users can be done manually by the administrator or owner of the application. The figure below shows three methods for managing an account: reset password, disabled account, and deleted account.

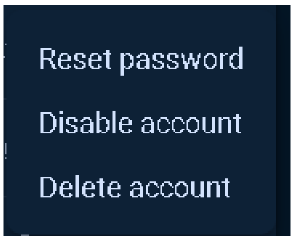


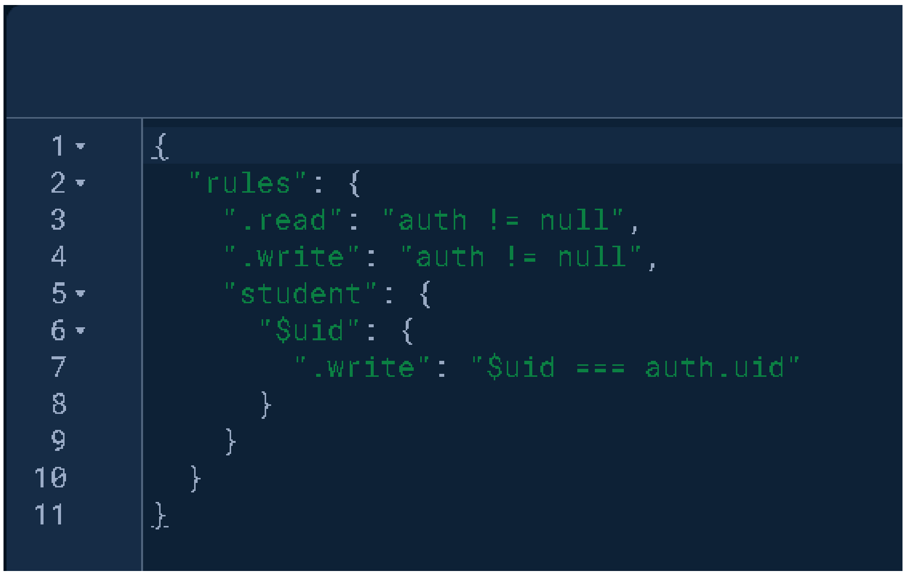
Figure 4.3: Firebase Authentication Account Management

Firebase Realtime Database

Firebase Realtime Database is a NoSQL cloud-hosted database that will store and sync all data from users in realtime.

To add the Firebase Realtime Database to the application, some steps are required, as shown in [27].

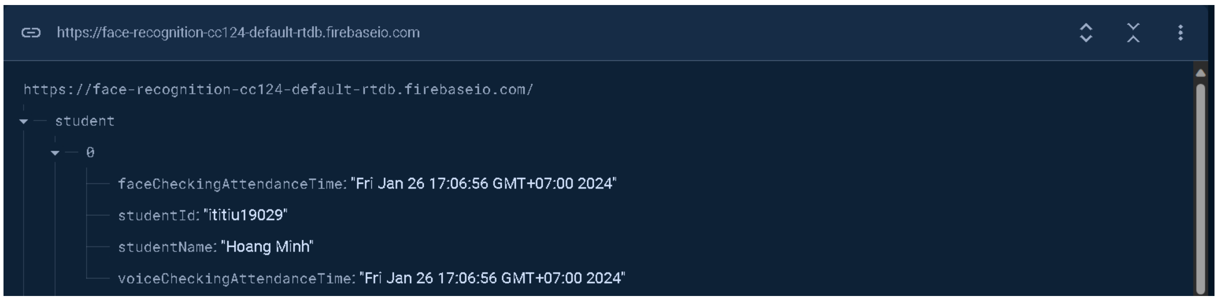
Figure 4.4 shows the rules for how the data are stored in the database. In this application, only students can write new data in the user-identification field.



```
{
  "rules": {
    ".read": "auth != null",
    ".write": "auth != null",
    "student": {
      "$uid": {
        ".write": "$uid === auth.uid"
      }
    }
  }
}
```

Figure 4.4: Firebase Realtime Database Rules

Figure 4.5 shows how a student's information is stored in the database. All data will be stored in JSON and key-value formats.



```
https://face-recognition-cc124-default-rtdb.firebaseio.com

https://face-recognition-cc124-default-rtdb.firebaseio.com/
▾ — student
    ▾ — 0
        — faceCheckingAttendanceTime: "Fri Jan 26 17:06:56 GMT+07:00 2024"
        — studentId: "ititiu19029"
        — studentName: "Hoang Minh"
        — voiceCheckingAttendanceTime: "Fri Jan 26 17:06:56 GMT+07:00 2024"
```

Figure 4.5: Database of Firebase Realtime Database

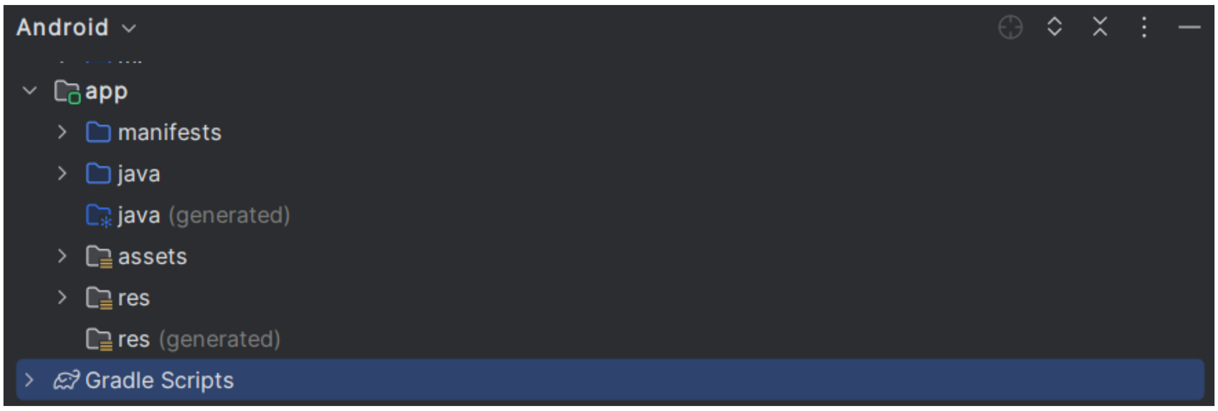
Firebase Machine Learning

Firebase Machine Learning is a mobile SDK that brings Google's machine learning expertise to Android and Apple apps in a powerful yet easy-to-use package. The face detection API of the Machine Learning Kit is the main feature used in the application, and with it, the application can detect faces in an image.

Adding Machine Learning Kit face detection API to the application is guided by [28].

4.1.1.2. Code

The code structure is divided into different types and is clearly distinguished so that the code of the application can be easily imagined.

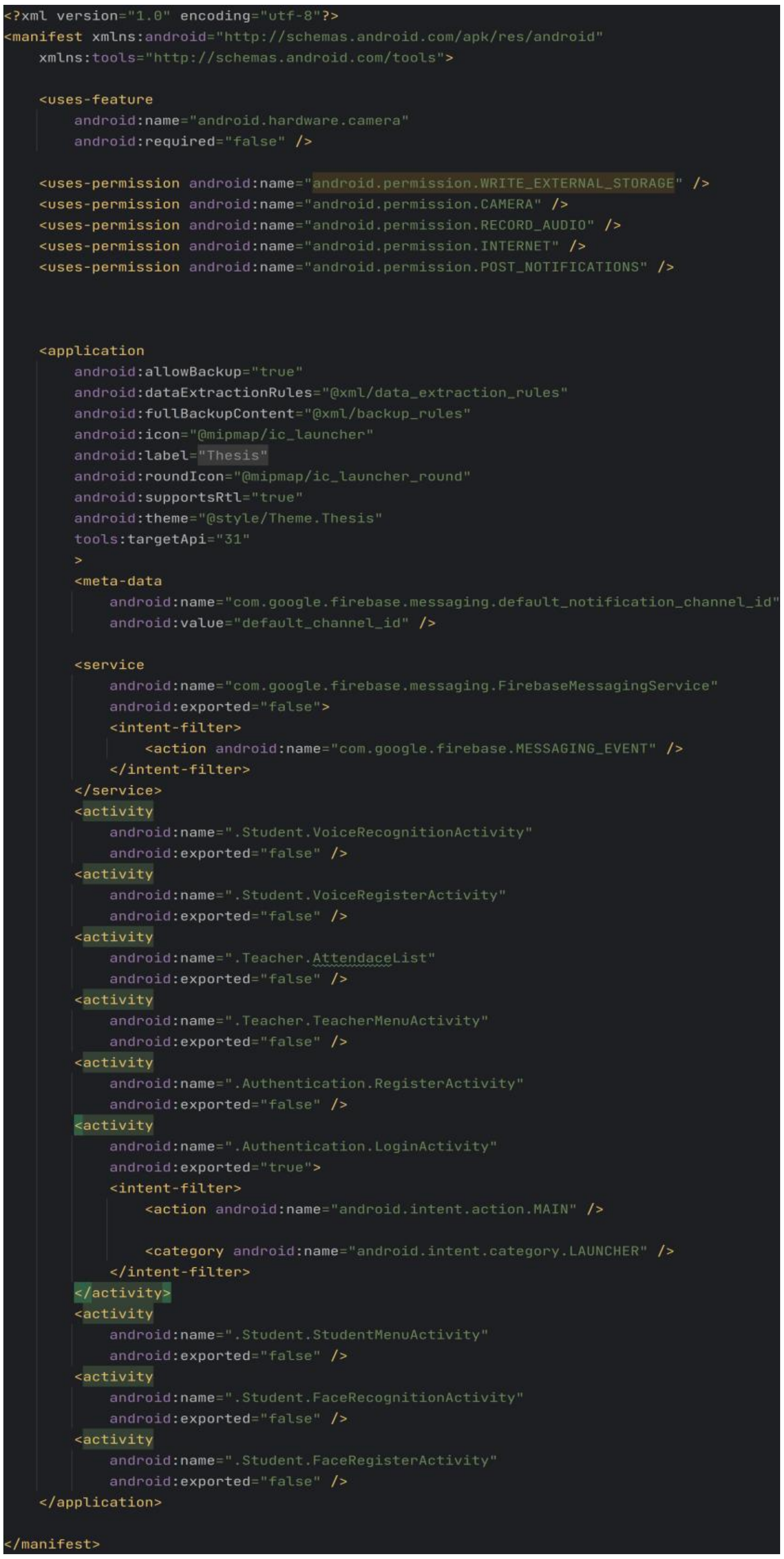


```
Android ⌄
⌄ app
    > manifests
    > java
      java (generated)
    > assets
    > res
      res (generated)
> Gradle Scripts
```

Figure 4.6: Structure of the application

Manifests

The manifests directory stores the AndroidManifest.xml, which is required for any Android application. In this file, everything about user permissions, application information, services, and the structure of activities will be registered.

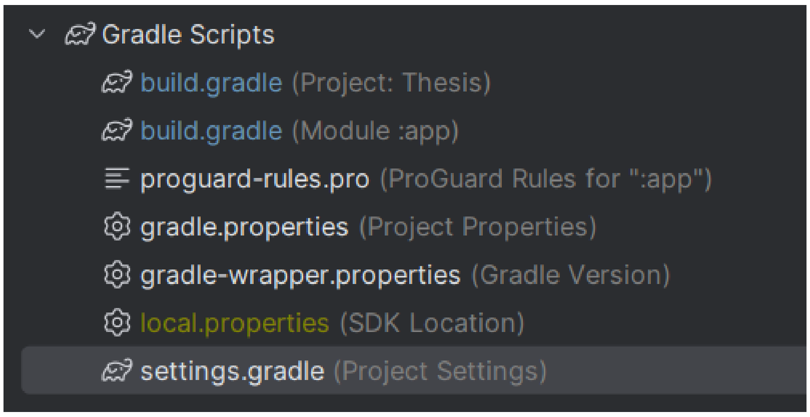


```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-feature
        android:name="android.hardware.camera"
        android:required="false" />

    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.POST_NOTIFICATIONS" />



    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="Thesis"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Thesis"
        tools:targetApi="31"
        >
        <meta-data
            android:name="com.google.firebase.messaging.default_notification_channel_id"
            android:value="default_channel_id" />

        <service
            android:name="com.google.firebase.messaging.FirebaseMessagingService"
            android:exported="false">
            <intent-filter>
                <action android:name="com.google.firebase.MESSAGING_EVENT" />
            </intent-filter>
        </service>
        <activity
            android:name=".Student.VoiceRecognitionActivity"
            android:exported="false" />
        <activity
            android:name=".Student.VoiceRegisterActivity"
            android:exported="false" />
        <activity
            android:name=".Teacher.AttendaceList"
            android:exported="false" />
        <activity
            android:name=".Teacher.TeacherMenuActivity"
            android:exported="false" />
        <activity
            android:name=".Authentication.RegisterActivity"
            android:exported="false" />
        <activity
            android:name=".Authentication.LoginActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".Student.StudentMenuActivity"
            android:exported="false" />
        <activity
            android:name=".Student.FaceRecognitionActivity"
            android:exported="false" />
        <activity
            android:name=".Student.FaceRegisterActivity"
            android:exported="false" />
    </application>

</manifest>
```

Figure 4.7: AndroidManifest.xml file

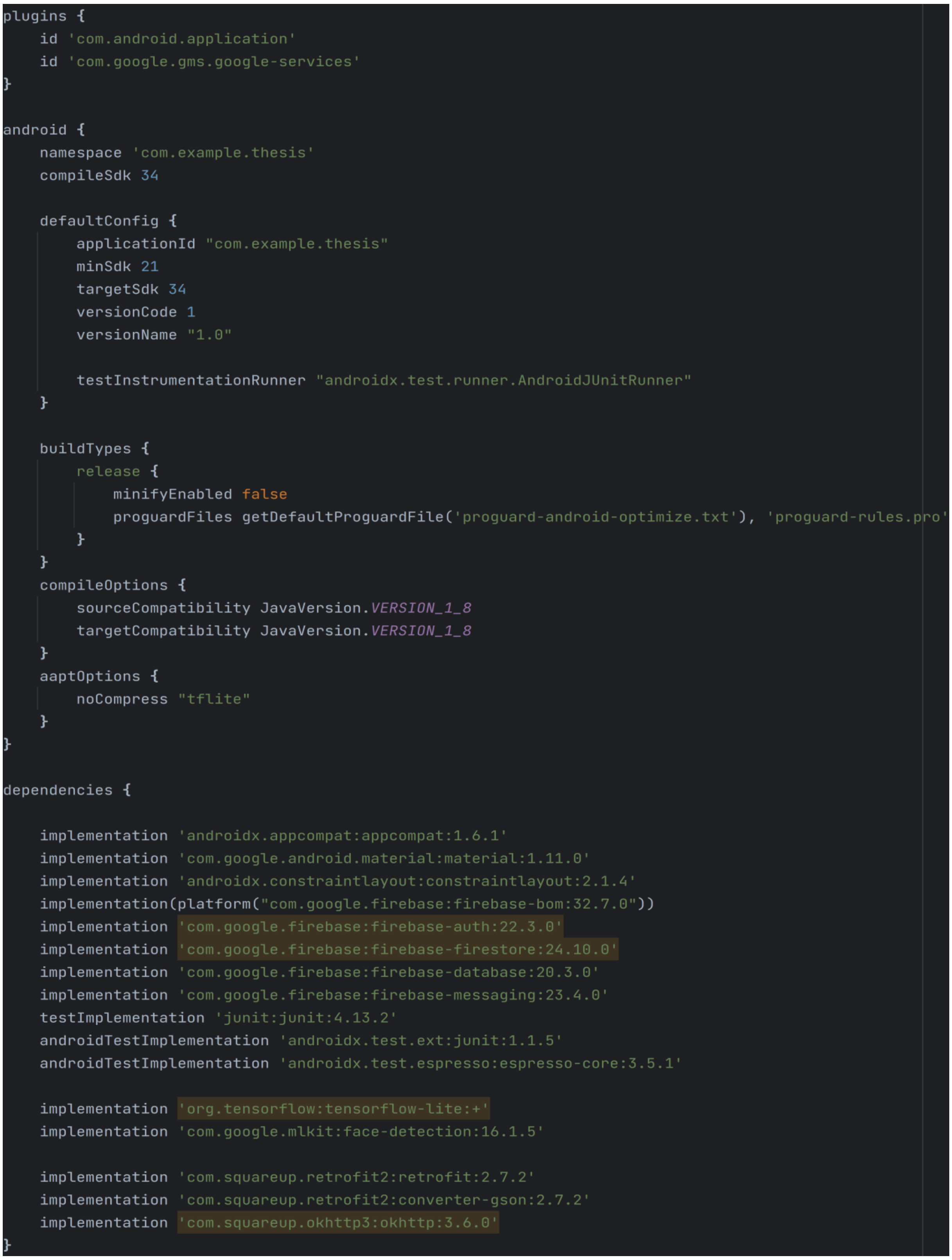
Gradle Scripts

Gradle Scripts are written in Groovy DSL and are required for every Android Application. Some figures show the main files of Gradle Scripts.



Gradle Scripts
build.gradle (Project: Thesis)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)

Figure 4.8: Structure of Gradle Scripts directory

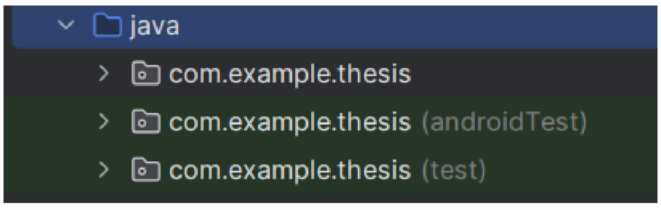


```
plugins {
    id 'com.android.application'
    id 'com.google.gms.google-services'
}

android {
    namespace 'com.example.thesis'
    compileSdk 34

    defaultConfig {
        applicationId "com.example.thesis"
        minSdk 21
        targetSdk 34
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    aaptOptions {
        noCompress "tflite"
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.6.1'
    implementation 'com.google.android.material:material:1.11.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation(platform("com.google.firebase:firebase-bom:32.7.0"))
    implementation 'com.google.firebase:firebase-auth:22.3.0'
    implementation 'com.google.firebase:firebase-firestore:24.10.0'
    implementation 'com.google.firebase:firebase-database:20.3.0'
    implementation 'com.google.firebase:firebase-messaging:23.4.0'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'

    implementation 'org.tensorflow:tensorflow-lite:+'
    implementation 'com.google.mlkit:face-detection:16.1.5'

    implementation 'com.squareup.retrofit2:retrofit:2.7.2'
    implementation 'com.squareup.retrofit2:converter-gson:2.7.2'
    implementation 'com.squareup.okhttp3:okhttp:3.6.0'
}
```

Figure 4.9: build.gradle (Module :app) file

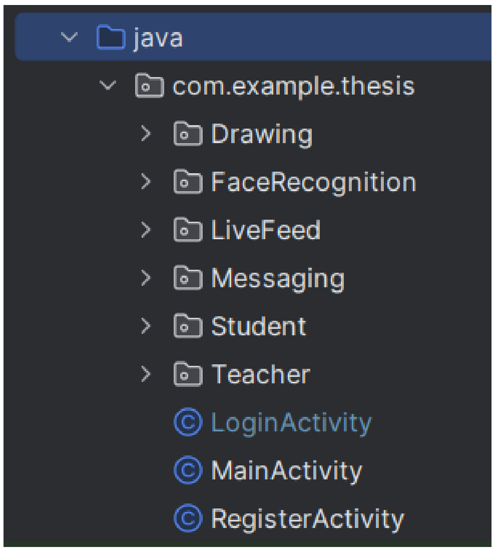
Figure 4.9 shows how a build.gradle (Module :app) file is written. There is information about minSdk, which is the lowest version of Android of a device that can run the application, and some libraries have been implemented, such as TensorFlow Lite, Firebase Authentication, and Firebase Database.

Java



```
java
  com.example.thesis
  com.example.thesis (androidTest)
  com.example.thesis (test)
```

Figure 4.10: Structure of java directory



```
java
  com.example.thesis
    Drawing
    FaceRecognition
    LiveFeed
    Messaging
    Student
    Teacher
    LoginActivity
    MainActivity
    RegisterActivity
```

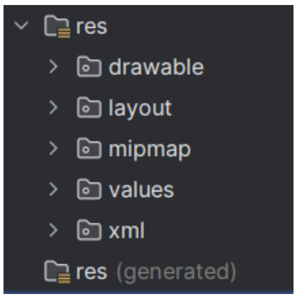
Figure 4.11: Structure of com.example.thesis directory

Figure 4.10 shows the structure of the Java directory, which includes three other directories. While two out of three directories are used for testing the system, the first one stores all Java files.

It is divided into seven small directories: Drawing for drawing a rectangle around the detected face, Face Recognition for handling the TensorFlow Model, Live Feed for detecting faces and drawing in real time, messaging for sending notifications, Student, Teacher and Authentication for registration and login.

res

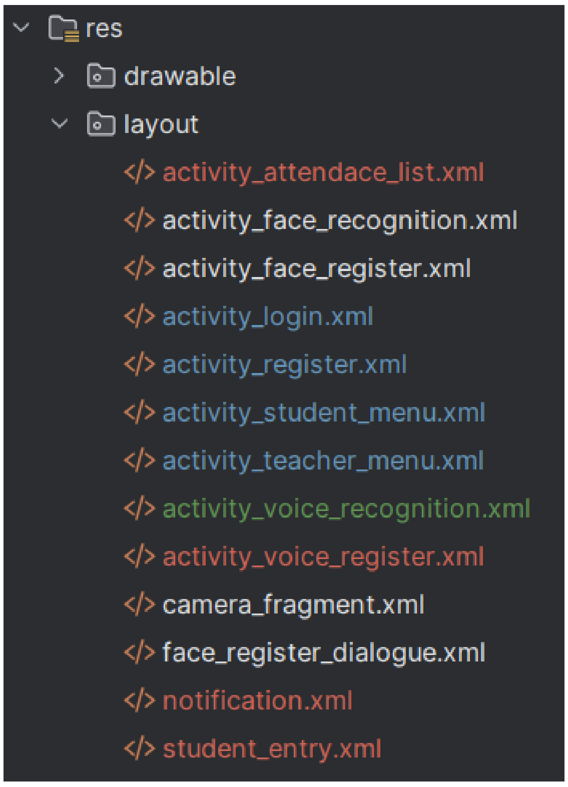
res is a directory that stores all xml files such as icons, colors, strings, and user interface files. Here, is the structure of the res directory.



```
res
  drawable
  layout
  mipmap
  values
  xml
res (generated)
```

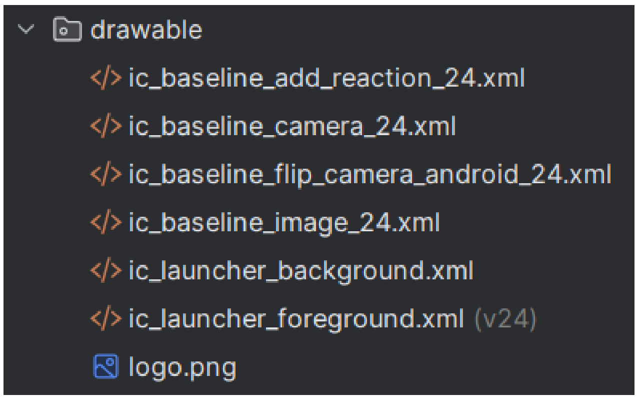
Figure 4.12: Structure of res directory

To make it easy to understand, Figure 4.13 will shows layout files which are all the user interface files such as Login Screen, Register Screen, Teacher Menu Screen, Student Menu Screen, Face Register Screen, Face Recognize Screen, Voice Register Screen, Voice Recognize Screen, and Face Register Dialogue. Furthermore, Figure 4.14 shows all icons and images used in the application.



```
res
  drawable
  layout
    activity_attendace_list.xml
    activity_face_recognition.xml
    activity_face_register.xml
    activity_login.xml
    activity_register.xml
    activity_student_menu.xml
    activity_teacher_menu.xml
    activity_voice_recognition.xml
    activity_voice_register.xml
    camera_fragment.xml
    face_register_dialogue.xml
    notification.xml
    student_entry.xml
```

Figure 4.13: Structure of layout directory



```
drawable
    ic_baseline_add_reaction_24.xml
    ic_baseline_camera_24.xml
    ic_baseline_flip_camera_android_24.xml
    ic_baseline_image_24.xml
    ic_launcher_background.xml
    ic_launcher_foreground.xml (v24)
    logo.png
```

Figure 4.14: Structure of drawable directory

4.2. Results

To run the entire project, any devices, such as virtual devices and physical devices, need to be created or paired. The guides for creating an Android virtual device and connecting a physical device are demonstrated in [29] and [30], respectively. In this study, a virtual device, Pixel 4a, using Android 11, was used to install the application.

4.2.1. Login Screen

Figure 4.15 shows the Login Screen, which has two text boxes: Email Address and Password, and how the system responds to incorrect email addresses and passwords; a message “Failed to Login” is shown.

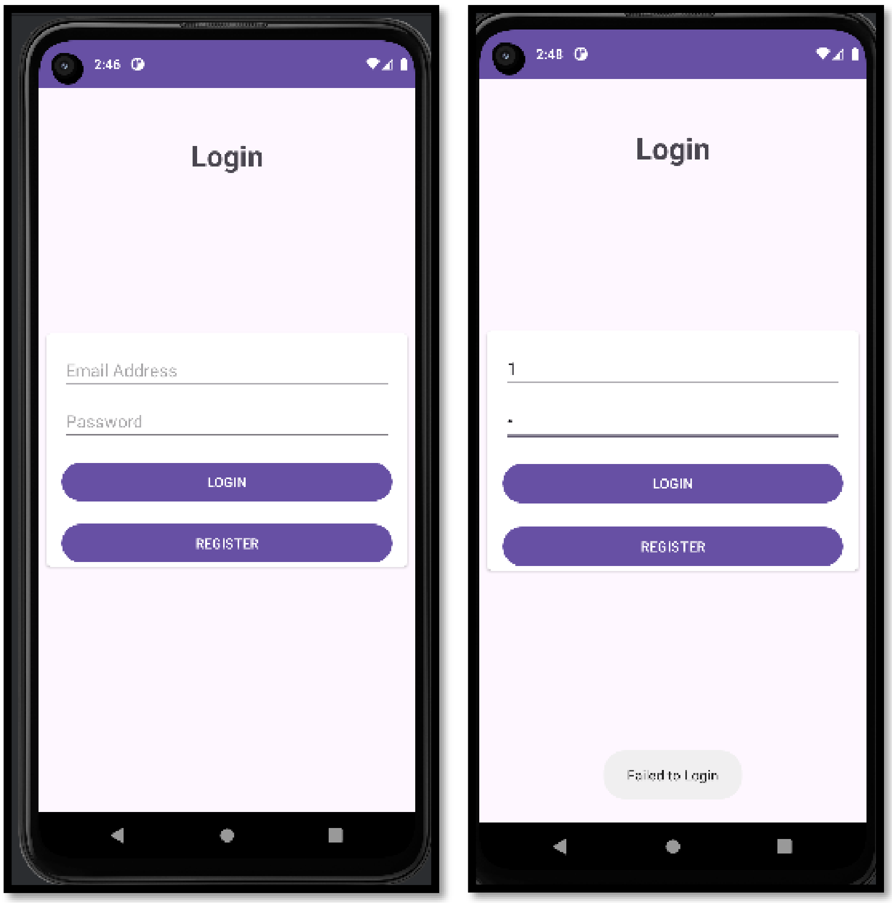


Figure 4.15: Login Screen

4.2.2. Register Screen

Figure 4.16 shows the Register Screen of the application, which has four text boxes for users to access their information. If the users press the “Create Account” button without fulfilling all boxes, error messages and icons will appear.

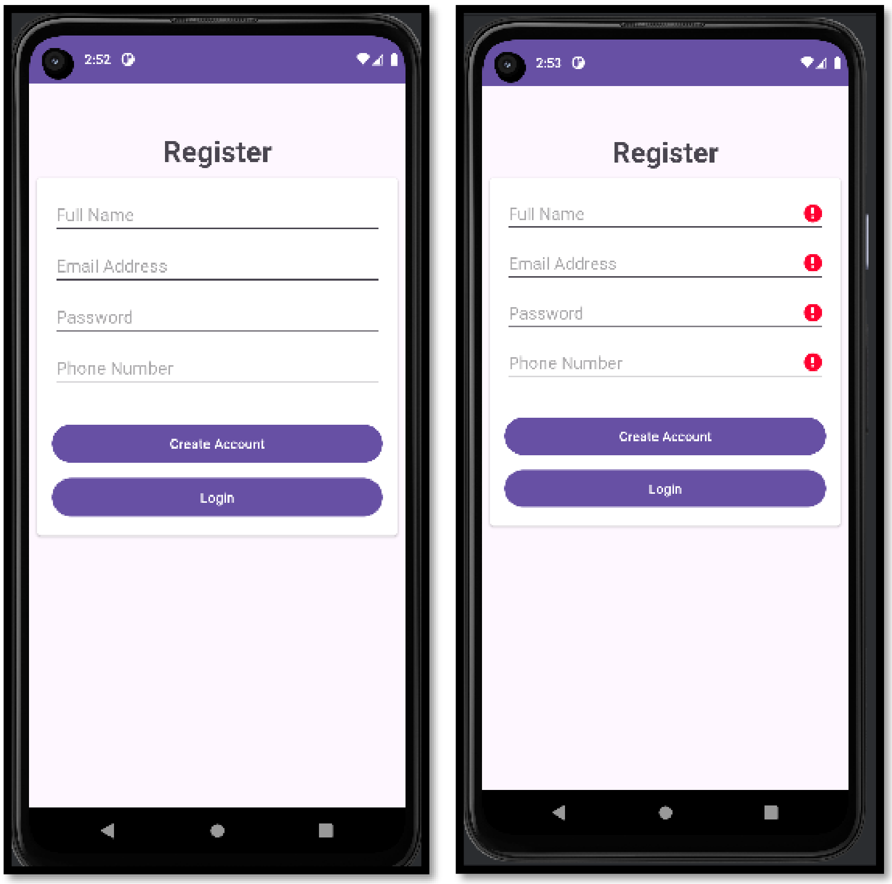
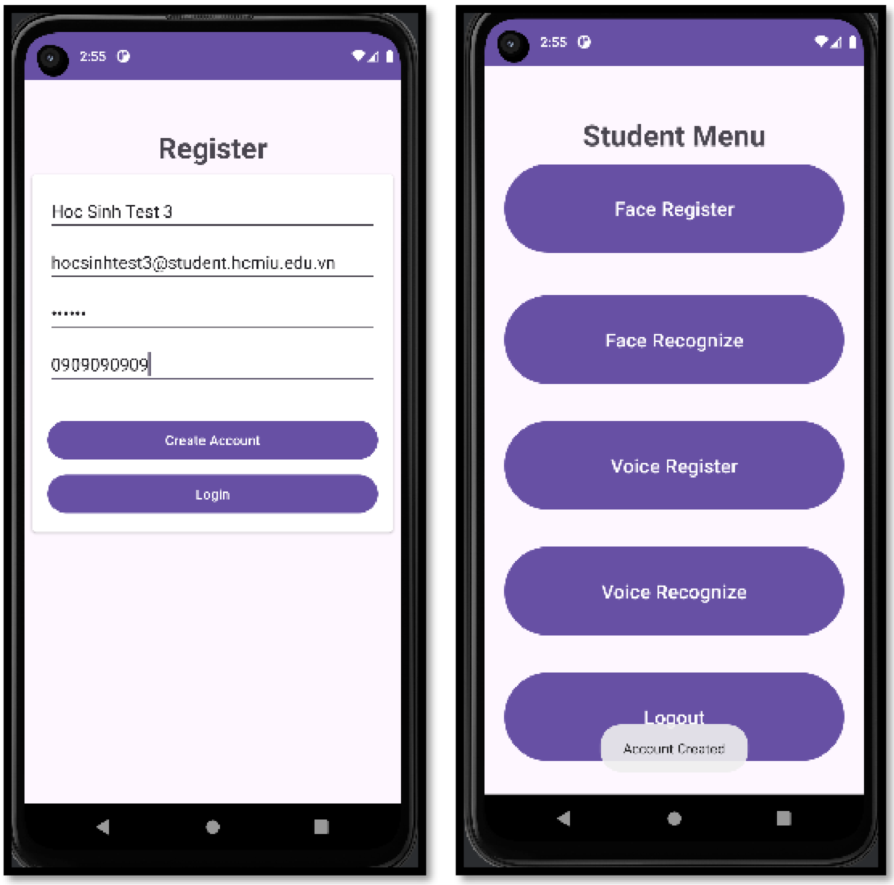


Figure 4.16: Register Screen

After entering all the required information, as shown in Figure 4.17 and 4.18, the system saves user information into the Firebase Authentication database and identifies the email address. If it belongs to a student, the system will navigate to the Student Menu Screen; otherwise, the Teacher Menu Screen will show up. In this application, any email ends with "@student.hcmiu.edu.vn" and "@hcmiu.edu.vn" will be considered as the Student and Teacher types, respectively. Any other type of email can be registered, but users cannot use it to log in.



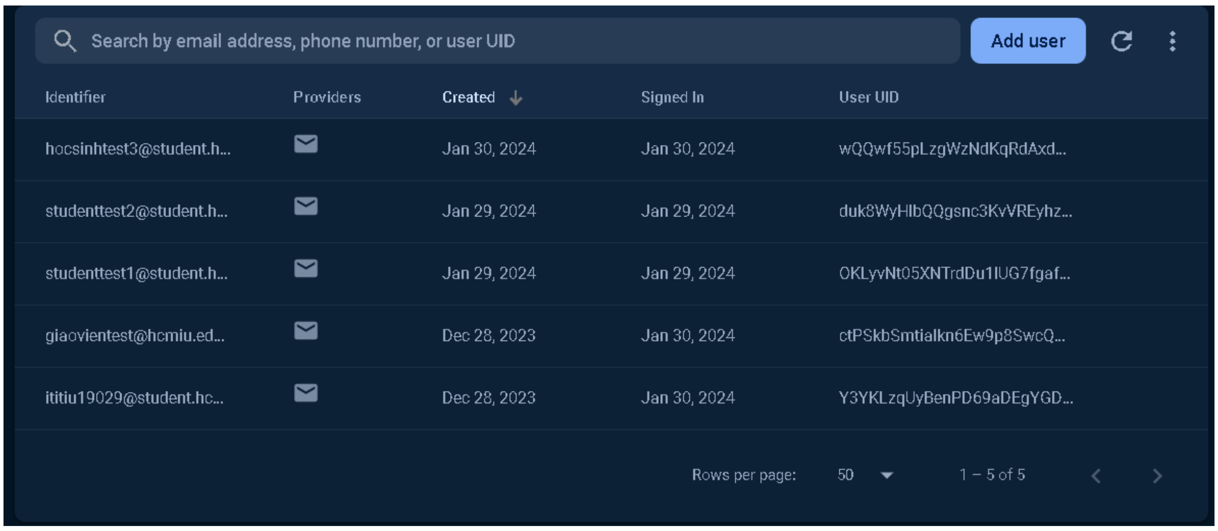
Hoc Sinh Test 3

hocsinhtest3@student.hcmiu.edu.vn

••••••

0909090909

Figure 4.17: The register form with full information



Identifier	Providers	Created	Signed In	User UID
hocsinhtest3@student.h...	✉	Jan 30, 2024	Jan 30, 2024	wQQwf55pLzgWzNdKqRdAxd...
studenttest2@student.h...	✉	Jan 29, 2024	Jan 29, 2024	duk8WyHlbQQgsnc3KvVREyhz...
studenttest1@student.h...	✉	Jan 29, 2024	Jan 29, 2024	OKLyvNt05XNTrdDu1lUG7fgaf...
giaovientest@hcmiu.ed...	✉	Dec 28, 2023	Jan 30, 2024	ctPSkbSmtialkn6Ew9p8SwcQ...
ititiu19029@student.hc...	✉	Dec 28, 2023	Jan 30, 2024	Y3YKLzqUyBenPD69aDEgYGD...

Rows per page: 50 1 – 5 of 5

Figure 4.18: Firebase Authentication database has been updated

4.2.3. Student Menu Screen

After logging into the application using a student account, the Student Menu Screen will be displayed. Figure 4.19 shows that it has five buttons: Face Register, Face Recognize, Voice Register, Voice Recognize, and Logout.

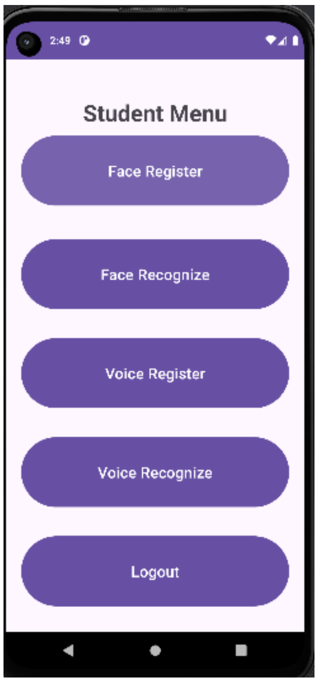


Figure 4.19: Student Menu Screen

Face Register Screen

After pressing the Face Register button on the Student Menu Screen, the Face Register Screen will be displayed. In this screen, there will be two buttons “Choosing an existing image from the library” and “Capturing a new image” that are represented by two icons.

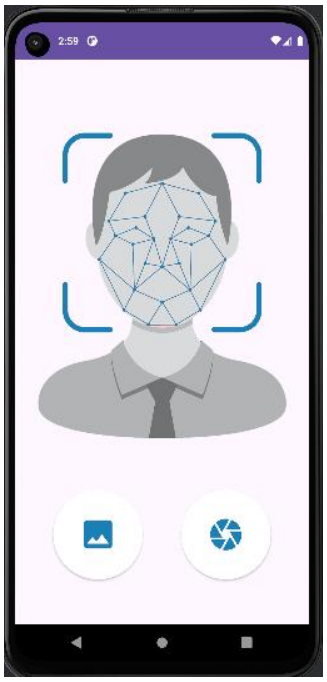


Figure 4.20: Face Register Screen

After choosing or capturing any images, they will be displayed in the middle of the screen, and the system detects any human faces in that image.

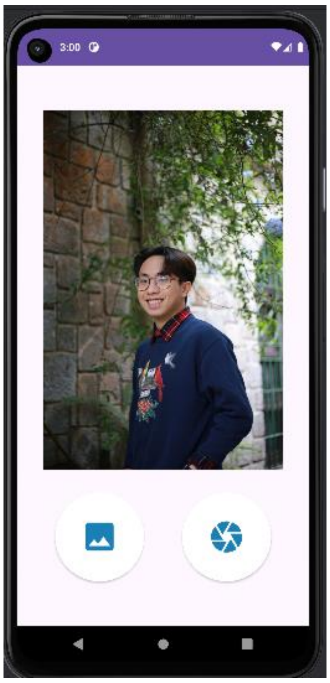


Figure 4.21: After choosing or capturing an image

After detecting a human face, the system shows the Face Register Dialogue. The user types their full name and presses the register button. If there is nothing in the "Enter Name" text box, an error message appears after the Register button is pressed.

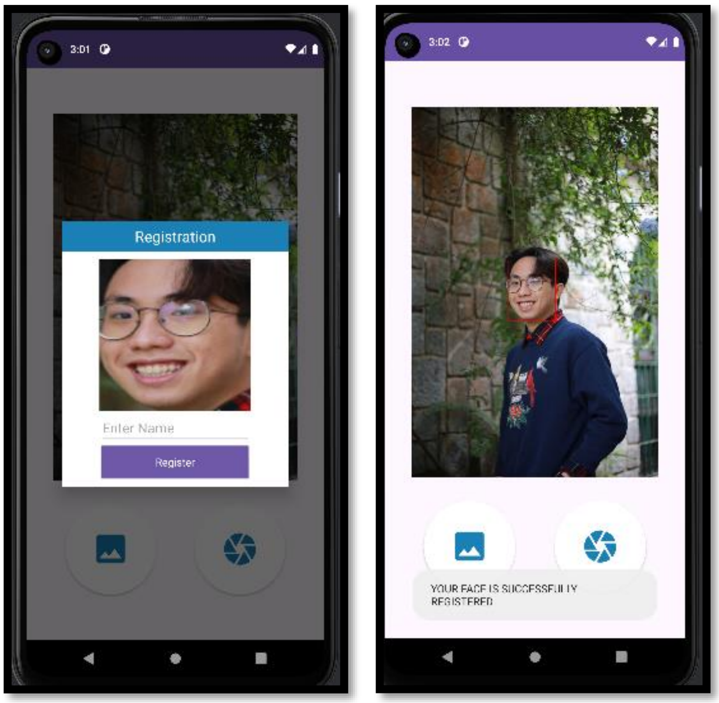


Figure 4.22: Face Register Dialogue

After successful registration, a "Your face is successfully registered" message will appear.

Face Recognize Screen

After pressing the Face Recognize button on the Student Menu Screen, the Face Recognize Screen is displayed. On this screen, there will be a huge image view representing the camera view and a button to switch from the back and front camera.

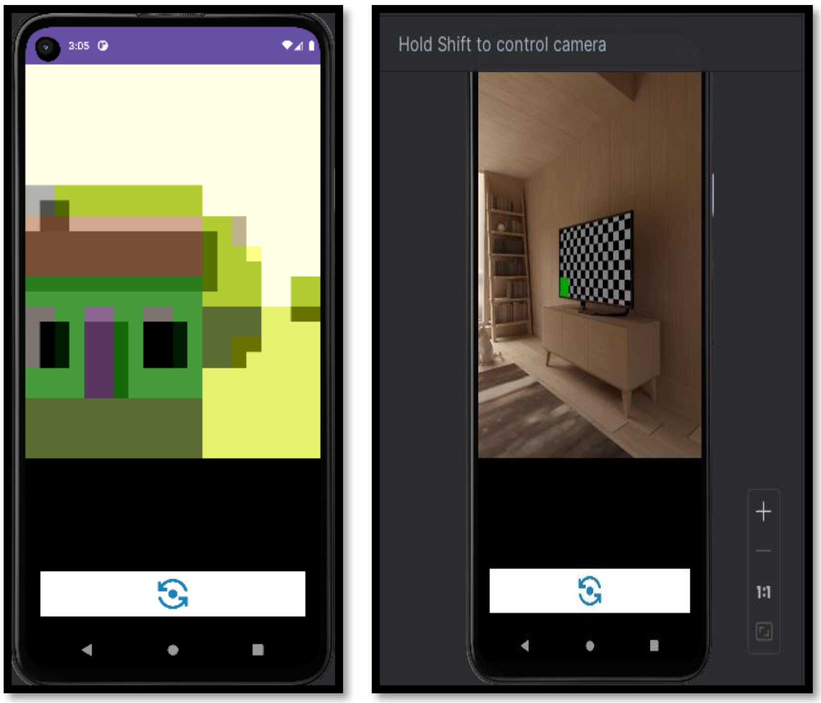


Figure 4.23: Face Recognize Screen

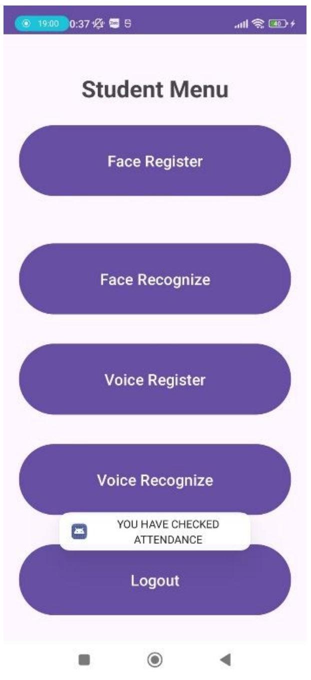


Figure 4.24: Check In Using Face Recognition successfully

Figure 4.24 shows that a check using face recognition was successfully demonstrated. This process was performed in the Redmi Note 9S device because the virtual device had no camera. After recognizing a face, the system automatically checks the student and navigates to the Student Menu Screen.

Voice Register Screen

After pressing the Voice Register button on the Student Menu Screen, the Voice Register Screen is displayed. On this screen, there will be a dropdown question menu, a text box that cannot be clicked, and two buttons, which are Start/Stop Recording and Register. The text box is coded as not being able to be clicked, so it prevents users from entering the answer manually.

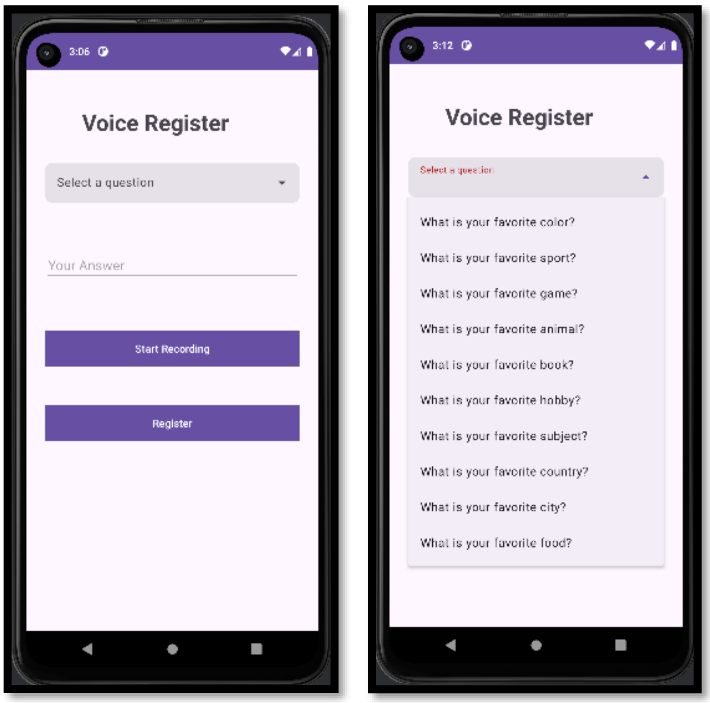

3:06
Voice Register
Select a question
Your Answer
Start Recording
Register
3:12
Voice Register
Select a question
What is your favorite color?
What is your favorite sport?
What is your favorite game?
What is your favorite animal?
What is your favorite book?
What is your favorite hobby?
What is your favorite subject?
What is your favorite country?
What is your favorite city?
What is your favorite food?

Figure 4.25: Voice Register Screen

If users press the Register button before choosing a question and recording an answer, two different error messages will appear to remind them.

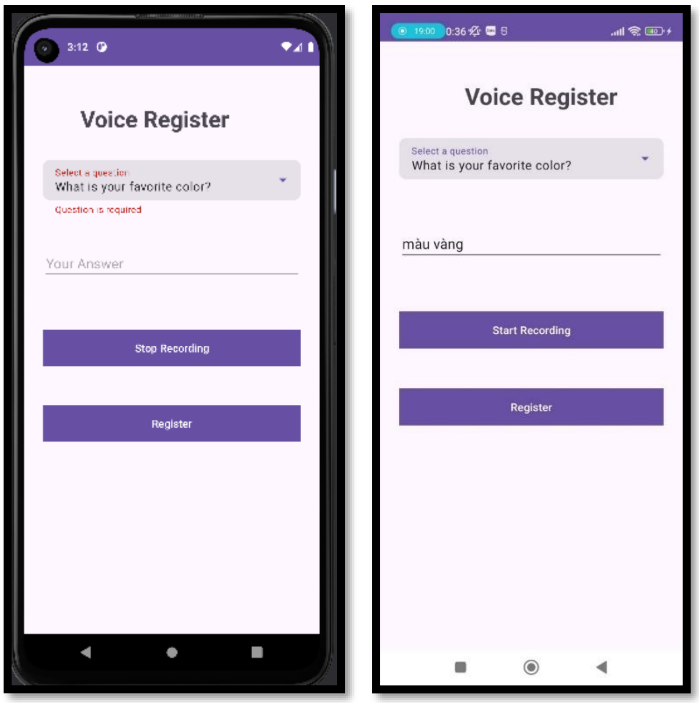


Figure 4.26: “Start Recording” button pressed

Figure 4.26 shows that after pressing the Start Recording button, the system will record everything that the users say aloud and turn it into their answer after the Stop Recording button is pressed. When the Register button is pressed, the chosen question and answer are stored in a local file. Figure 4.27 shows that the answers were recorded using this system.

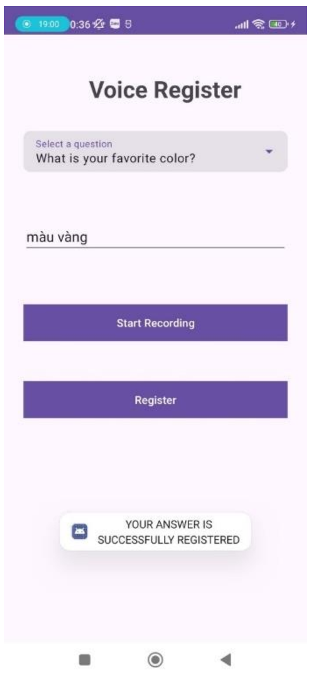

Voice Register
Select a question
What is your favorite color?
màu vàng
Start Recording
Register
YOUR ANSWER IS SUCCESSFULLY REGISTERED

Figure 4.27: Registering Succefully

Voice Recognize Screen

After pressing the Voice Recognize button on the Student Menu Screen, the Voice Recognize Screen is displayed. This screen will be the same as the Voice Register Screen, except that the Register button will be changed to the Recognize button. After pressing the recognize button, the system checks whether the questions and answers match. If they match, the check-in record is pushed into the Firebase Realtime Database.

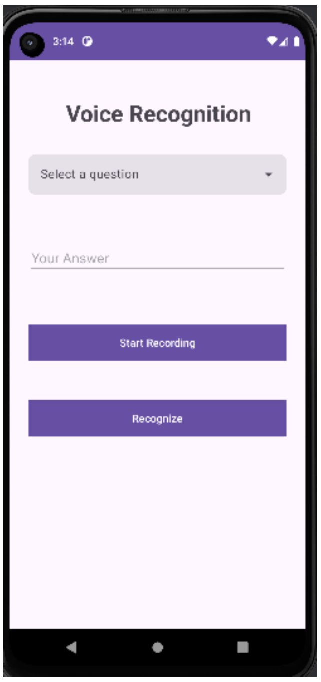
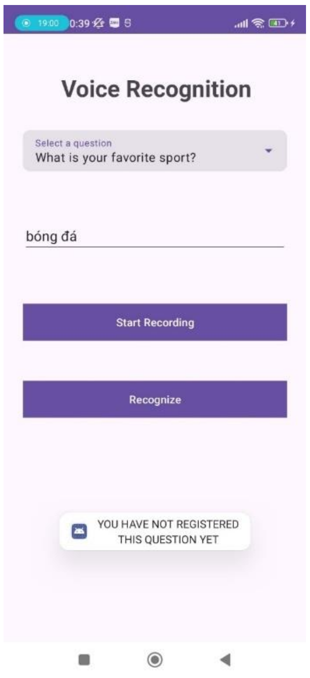


Figure 4.28: Voice Recognize Screen



What is your favorite sport?

bóng đá

YOU HAVE NOT REGISTERED THIS QUESTION YET

Figure 4.29: Choosing a question which is not registered

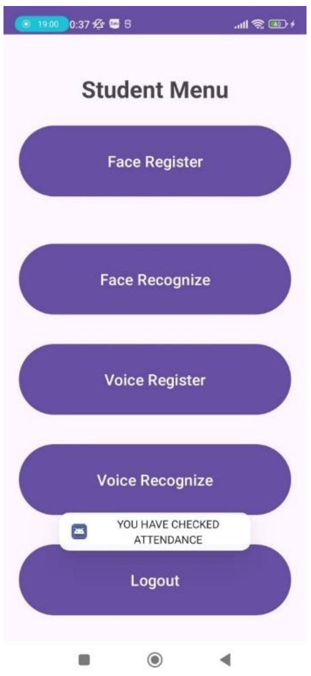


Figure 4.30: Check in successfully

Figure 4.30 shows the successful check-in process after question-and-answer matching.

4.2.4. Teacher Menu Screen

After logging into the application using a teacher account, the Teacher Menu Screen is displayed. Figure 4.31 shows that it has four buttons: Check Face Attendance, Check Voice Attendance, Attendance List, and Logout.

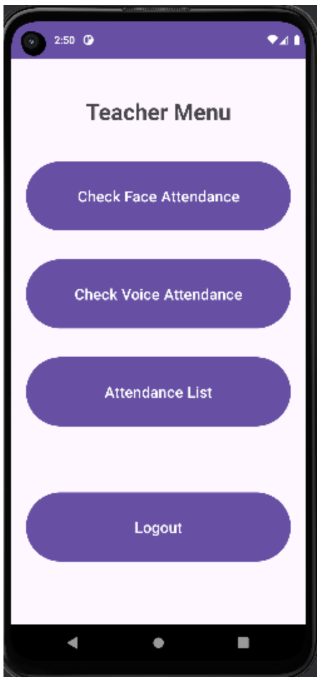


Figure 4.31: Teacher Menu Screen

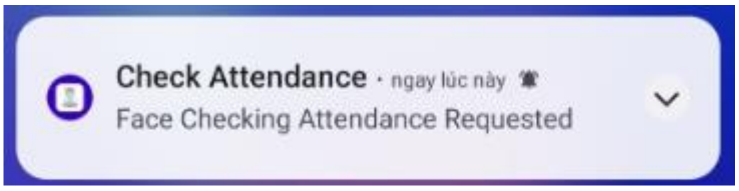

Check Attendance · ngay lúc này
Face Checking Attendance Requested

Figure 4.32: Push notification after "Check Face Attendance" button pressed

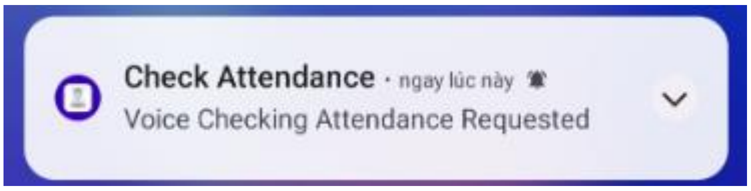

Check Attendance · ngay lúc này
Voice Checking Attendance Requested

Figure 4.33: Push notification after "Check Voice Attendance" button pressed

Figure 4.51 and 4.52 show the push notifications on the Student devices after the Teacher pressed the "Check Face Attendance" and "Check Voice Attendance" buttons, respectively.

CHAPTER 5

DISCUSSION AND EVALUATION

5.1. Discussion

All the initial objectives mentioned have been met with a complete demonstration version, such as an Android application that allows users to register, log in, and check attendance. In addition, the different screens and functions of the Teacher and the Student provide a modern and reasonable approach.

5.2. Comparison

5.2.1. Face Recognition Based Attendance System

The difference is that Python and OpenCV are used, whereas Java and TensorFlow Lite are used in this thesis. The advantage of this thesis is that Android devices are becoming increasingly common and useful for checking attendance. Furthermore, this thesis system has an authentication system, a database, and a function for the teacher to send notifications to students.

5.2.2. Mobile Attendance Checking System on Android Platform for Kazakhstani University

The Mobile Attendance Checking System for Kazakhstani University has an old design, is outdated, and does not provide authentication to users. As mentioned above, this thesis takes a much more modern and reasonable approach. Furthermore, this thesis project combines face recognition and voice recognition to achieve higher accuracy.

5.3. Evaluation

In this project, checking attendance using face and voice recognition was successfully applied and returned fairly good performance. In addition, different screens and functions for the Teacher and the Student increase the security and interaction of the application.

However, building more functions, such as viewing grades or homework, will also be an essential task that needs to be done to upgrade the application.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1. Conclusion

The autonomous attendance mobile application based on face recognition or voice recognition has been successfully created and has some important functions such as Register Face, Register Voice, Check In Using Face Recognition, and Check In Using Voice Recognition. This project provides an opportunity to learn how to create an Android application, how to use Firebase, and how to implement machine learning to solve real-world problems. In addition, the application also enhances knowledge about the user interface and user experience design, an Android application lifecycle, and how to create a system from the beginning.

6.2. Future work

Although the application has been successfully created, there are still many shortcomings that need to be addressed.

The first involves upgrading the user interface and the user experience design. A more user-friendly and beautiful design should be created to replace old designs.

The second is upgrading more useful functions such as viewing grades, homework, or session materials to help the Teacher and the Student to have a better application for the study environment.

REFERENCES

1. Saparkhojayev, N., Shakhov, E., & Mailybayev, Y. (2016). Mobile Attendance Checking System on Android Platform for Kazakhstani University. Journal of Physics: Conference Series, 710, 1.
2. Salac, D.M.V.(2018). PRESENT: An Android-based class attendance monitoring system using face recognition technology. International Journal of Computing Sciences Research, 2(3), 102-115.
3. Robinson-Riegler, B., Robinson-Riegler, G. (2016). Cognitive psychology: Applying the science of the mind.
4. Gillis, A. S. (2022, December). Definition: Facial recognition. TechTarget. Retrieved from https://www.techtarget.com/searchenterpriseai/definition/facial-recognition.
5. Silk, R. (2017). Biometrics: Facial recognition tech coming to an airport near you. Travel Weekly. Retrieved from https://www.travelweekly.com/Travel-News/Airline-News/Biometrics-Facial-recognition-tech-coming-airport-near-you.
6. Fussell, S. (2018). Facebook's new face recognition features: What we do (and don't) know. Gizmodo. Retrieved from https://gizmodo.com/facebooks-new-face-recognition-features-what-we-do-an-1823359911.
7. Reichert, C. (2017). Intel demos 5G facial-recognition payment technology. ZDNet. Retrieved from https://www.zdnet.com/article/intel-demos-5g-facial-recognition-payment-technology/.
8. deAgonia, M. (2017, November 1). Apple's Face ID [The iPhone X's facial recognition tech] explained. Computerworld. Retrieved from https://www.computerworld.com/article/3235140/apples-face-id-the-iphone-xs-facial-recognition-tech-explained.html.
9. Zhao, W., Chellappa, R., Phillips, P. J., & Rosenfeld, A. (2003). Face recognition: A literature survey. ACM Computing Surveys, 35(4), 399–458. https://doi.org/10.1145/954339.954342.

10. Pooja G.R, et al. (2010). Automated attendance system using image processing. In Proceedings of the 1st International Conference on Innovations in Computing & Networking (ICICN16) (p. 363). Published in the International Journal of Advanced Networking & Applications (IJANA), ISSN: 0975-0282.
11. Wagh, P., Thakare, R., Chaudhari, J., & Patil, S. (2015). Attendance system based on face recognition using eigen face and PCA algorithms. In 2015 International Conference on Green Computing and Internet of Things (ICGCIoT). DOI: 10.1109/ICGCIoT.2015.7380478
12. P. A. K. M. S. V. K. M. A. P. Z. M. N. D. B. M. C. J. B. (2017). Attendance System Using Face Recognition and Class Monitoring System. International Journal on Recent and Innovation Trends in Computing and Communication, 5(2), 273–276. https://doi.org/10.17762/ijritcc.v5i2.214
13. Katara, A., Kolhe, S. V., Zilpe, A. P., Bhele, N. D., Bele, C. J. (2017). Attendance System Using Face Recognition and Class Monitoring System. International Journal on Recent and Innovation Trends in Computing and Communication, 5(2), 273-276.
14. Kumar, Y. V. S. S. Avinash, Venkata Lakshmi, CH., Harish, G. B., & Khan, Pattan Abdulla. (2020-21). Face Recognition Based Attendance System [Bachelor of Technology, Electronics and Communication Engineering, Project report]. Anil Neerukonda Institute of Technology and Sciences.
15. Barney, N. (2023). Definition: face detection. TechTarget. Retrieved from https://www.techtarget.com/searchenterpriseai/definition/face-detection.
16. Wikipedia. (n.d.). Facial recognition system. Retrieved from https://en.wikipedia.org/wiki/Facial_recognition_system.
17. Wikipedia. (n.d.). Speech recognition. Retrieved from https://en.wikipedia.org/wiki/Speech_recognition.
18. Wikipedia. (n.d.). Firebase. Retrieved from https://en.wikipedia.org/wiki/Firebase.
19. Firebase. (n.d.). Firebase Authentication. Retrieved from https://firebase.google.com/docs/auth.

20. Firebase. (n.d.). Firebase Realtime Database. Retrieved from https://firebase.google.com/docs/database.

21. Firebase. (n.d.). Firebase Machine Learning. Retrieved from https://firebase.google.com/docs/ml.

22. Firebase. (n.d.). Firebase Cloud Messaging. Retrieved from https://firebase.google.com/docs/cloud-messaging.

23. Wikipedia. (n.d.). TensorFlow. Retrieved from https://en.wikipedia.org/wiki/TensorFlow.

24. Wikipedia. (n.d.). FaceNet. Retrieved from https://en.wikipedia.org/wiki/FaceNet.

25. Firebase |. (n.d.). Add Firebase to your Android project. Retrieved January 31, 2024, from https://firebase.google.com/docs/android/setup?authuser=1&hl=en.

26. Firebase |. (n.d.). Get Started with Firebase Authentication on Android. Retrieved January 31, 2024, from https://firebase.google.com/docs/auth/android/start?authuser=1.

27. Firebase |. (n.d.). Connect your App to Firebase. Retrieved from https://firebase.google.com/docs/database/android/start?authuser=1.

28. Firebase |. (n.d.). Detect Faces with ML Kit on Android. Retrieved January 31, 2024, from https://firebase.google.com/docs/ml-kit/android/detect-faces?authuser=1.

29. Firebase |. (n.d.). Create and manage virtual devices. Retrieved January 31, 2024, from https://developer.android.com/studio/run/managing-avds.

30. Firebase |. (n.d.). Run apps on a hardware device. Retrieved January 31, 2024, from https://developer.android.com/studio/run/device.

31. Kim, J.H., Kim, H., & Lee, Y.S. (July 2001). A novel method using edge detection for signal extraction from cDNA microarray image analysis. Experimental and Molecular Medicine, 33(2), 83-88. DOI: 10.1038/emm.2001.15.

APPENDIX