Analysis of Attendance System in Practicum Room Using Fingerprint Sensor Integrated with Android Application and Database

Prayogo Jati[1], Mozart Rafi Kurniawan Tampubolon.[1]

[1]Jurusan Teknik Elektro, Prodi Teknik Rekayasa Elektronika, Politeknik Negeri Batam, Batam, Indonesia

Email: prayogojati01@gmail.com
Email: mozartrafi@stegripe.org

Received on 23-12-2025 | *Revised on 23-12-2025* | *Accepted on 23-12-2025*

Student attendance recording in the laboratory is a crucial component in education that often faces challenges such as data manipulation and time-consuming processes. This research aims to design and implement an automatic attendance system based on fingerprint sensors integrated with an Android application and a database. The system is designed using the ESP32 as the microcontroller, Raspberry PI as the host machine to run Docker, and the R307 fingerprint sensor for identification. Attendance data captured by the fingerprint sensor will be sent to the server running on the Raspberry PI via the ESP32. This server, hosted in a Docker container, will store the attendance data in a MariaDB or MySQL database, enabling real-time access and monitoring through an Android application. By leveraging fingerprint sensor technology, this system can enhance the accuracy and efficiency of attendance recording while reducing data manipulation. The testing results indicate that the system is capable of handling data traffic and transactions between devices, Android applications, and the server with an average latency of 0.071 seconds during service initialization, 0.335 seconds for API access from clients, and 0.541 seconds for web-based access, with a total system deployment time of approximately 60.6 seconds. In addition, interviews with students and lecturers show a 100% satisfaction rate, indicating that the system is considered more effective and efficient than conventional manual attendance methods.

Keywords: Automatic attendance system, R307 fingerprint sensor, Android application, MariaDB database, Student attendance.

I. INTRODUCTION

In the world of education, especially in practical laboratories, recording student attendance is a crucial component that faces many challenges. Manual attendance systems, which generally rely on signatures, have a number of weaknesses such as susceptibility to data manipulation, loss of documents, and time-consuming processes [1]. This has a negative impact on attendance evaluation, suboptimal academic management, and potential unfairness in assessment [2].

Along with technological developments in the digital age, various innovative solutions have emerged to overcome these problems. One promising technology is the use of fingerprint sensors for attendance systems. Fingerprint sensors offer high accuracy and security because each individual has a unique fingerprint pattern that is difficult to forge [3]. With the application of fingerprint sensors, attendance identification can be done quickly, precisely, and with minimal errors [4].

Given the increasingly advanced global conditions, this study proposes an automatic attendance system using fingerprint sensors integrated with a database. This system can improve the accuracy and efficiency of recording student attendance in practical laboratories. In addition, this system allows real-time data monitoring through an Android application, which minimizes the potential for data manipulation [5].

Manual attendance systems often hinder effective and accountable academic management. Therefore, innovative solutions are needed to address these challenges and usher education into a more modern, accurate, and efficient era. Fingerprint sensor technology can overcome various problems that often arise in manual attendance systems, while also providing users with easy access to check their attendance anytime and anywhere through an Android application [6].

Overall, the background of this research is to introduce an innovative solution that can overcome the weaknesses of traditional attendance systems. Thus, this research offers a more modern, accurate, and efficient way of managing student attendance data in the practicum room [7].

II. METHOD

A. System Design

The design of the fingerprint attendance system integrated with the application and database consists of two main components, namely the Android application and the MariaDB database system. The Android application will have several key features to facilitate the recording and monitoring of student attendance. The block diagram of the system can be seen in Figure 1.

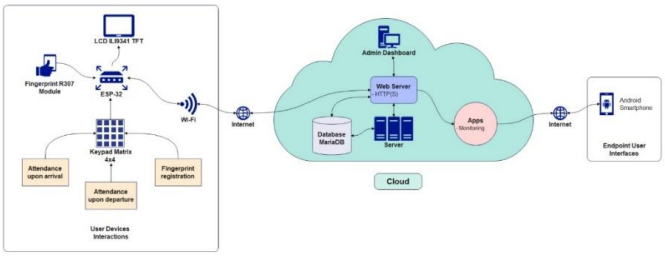

Figure 1. Overall system architecture of the fingerprint-based attendance system is integrated with an Android application and a cloud database.

Figure 1 illustrates the overall architecture of the fingerprint attendance system integrated with the Android application and cloud-based database. This diagram shows how the attendance device on the user side, consisting of ESP32, R307 fingerprint sensor, keypad, and TFT LCD, functions as the starting point for collecting student attendance data. The data obtained from the registration and attendance processes is then sent via an internet connection to the central server, so that the entire attendance recording process no longer depends on local storage on the device [8].

First, students will register their fingerprints on a fingerprint sensor connected to a microcontroller. The registered fingerprint data will be stored in a database and used to automatically record attendance [9]. The database will store the profiles of all students, and attendance records will be separated by date to facilitate monitoring and analysis.

This Android application will have the main features of today's attendance and information on student attendance status for that day [10]. Users who log in will default to guest status and can select the "Today's Attendance" menu to see a summary of attendance for that day. In this menu, users can filter student searches by name, student ID number, class, and attendance status [11].

The "Admin Login" menu is specifically for registered admin accounts. Admins will have access to manage attendance data and student information. This process is done by connecting the "Admin Menu" page through a web server to communicate with the database so that student data can be retrieved and managed on that page.

The integration of fingerprint sensors with the database begins with the collection of student fingerprint data through sensors that capture fingerprint patterns [12]. This data is then processed to extract minutiae points, which are converted into unique digital templates and stored in the database along with student information. During the verification or identification stage, students place their fingers on the sensor to generate new templates that are compared with existing templates in the database using matching algorithms such as minutiae-based matching or pattern matching. During this process, fingerprint templates are retrieved from the database for matching, and an Android application or other software serves as a user interface that communicates with a central server to manage data. The data type used is a random ID based on the output of these minutiae points.

Minutiae points themselves are specific features in fingerprint patterns that are used in fingerprint recognition technology for identification and verification. Minutiae points include "bifurcations," which are points where a single fingerprint ridge branches into two or more, and "ridge endings," which are points where a fingerprint ridge ends or is interrupted. In addition, there are other features such as dots (small points), islands (isolated short ridges), lakes (ridges that form circles), and crossovers (two ridges that cross each other). In the fingerprint recognition process, the sensor captures the fingerprint pattern and the feature extraction algorithm identifies and records the location and orientation of these minutiae points. The resulting digital template is used to match new fingerprints with those already in the database, ensuring accurate and unique identification [13].

B. Electrical Design

The electrical design of the Smart Workspace Attendance System device uses ESP32 as the main microcontroller. One of the important and necessary features for creating a fingerprint attendance device on this microcontroller is the Wi-Fi feature. This feature is needed so that the device can connect to the MariaDB database via the internet. The electrical wiring diagram for the fingerprint device in this project can be seen in Figure 2.

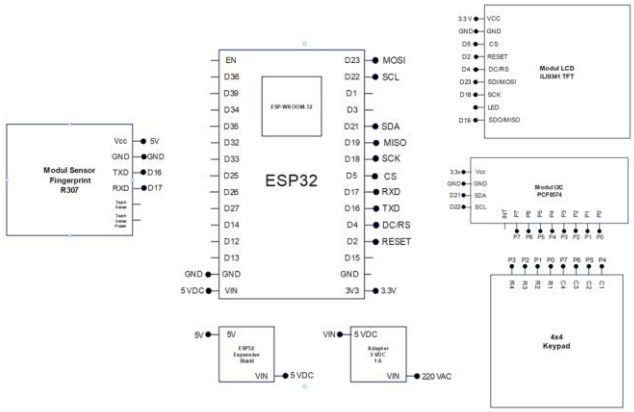

Figure 2. Electrical wiring diagram of the fingerprint attendance device based on the ESP32 microcontroller.

The ESP32 microcontroller receives input voltage from an adapter that has a voltage of 5 VDC and 1 A. The adapter will be connected to a special ESP32 extension module first. This is done so that modules that require GND pins and voltages of both 3.3 Volts and 5 Volts have sufficient terminals as a voltage source.

The ILI9341 TFT LCD module is used as the main interface module that displays the User Interface (UI) to users. This module requires a voltage of 3.3 volts to function properly. On this module, users can see the menu options for clocking in, clocking out, and registering student fingerprints based on commands from the ESP32 to the LCD module.

The 4x4 keypad module is one of the modules that provides input from the user to the ESP32 microcontroller. This module is used to provide input for the menu selection, confirmation, back, name, student ID number, and class features. This module is connected to the PCF8574 I2C module, which requires a voltage of 3.3 volts. Then, the I2C PCF8574 module is

connected to Arduino via the SDA and SCL pins to run I2C serial communication. The main purpose of using the I2C PCF8574 module is to reduce the number of pins from the keypad module to the ESP32. This reduction is significant because it can reduce the number of pins required from eight to only two that will go to the ESP32.

The last module that acts as an input provider for the Smart Workspace Attendance System fingerprint device is the R307 fingerprint module. This module requires a voltage source of 5 Volts to function properly. This module works by capturing the fingerprint patterns of each individual and storing them in the module's internal storage in the form of fingerprint templates. These fingerprint templates are then assigned an ID and sent to the ESP32 for processing in the attendance process.

C. Mechanical Design

The mechanical design of the Smart Workspace Attendance System device was created with user convenience in mind. The fingerprint sensor, keypad module, and TFT LCD are located at the front of the device for easy access by users. The keypad module is on the left and the TFT LCD is above the keypad module. In addition, the fingerprint sensor is placed on the right side of the front to make it easier for users to scan their fingerprints using their right hand. The mechanical design of the Smart Workspace Attendance System device can be seen in Figure 3.

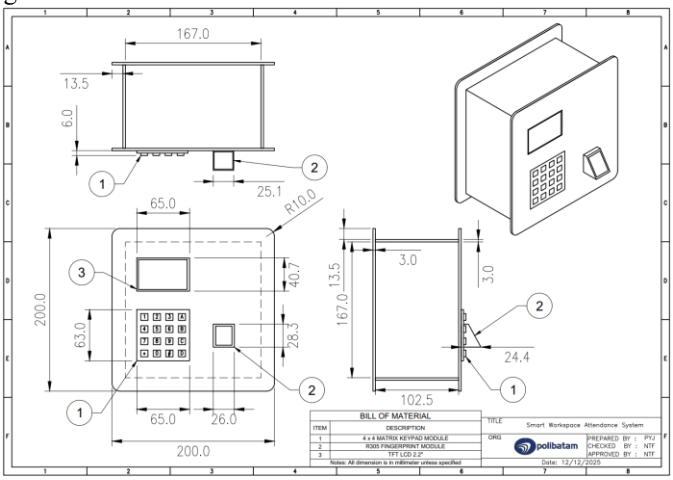

Figure 3. Mechanical design and physical layout of the Smart Workspace Attendance System device.

The main dimensions of the Smart Workspace Attendance System device are 200 mm x 108.5 mm x 200 mm. The device is designed to be easy to carry due to its small size. This makes maintenance of the device easy to perform.

D. Software Design

In implementing the features that have been designed previously in the design process, software design involves several important stages. The stages of software design are as follows.

1) User Experience (UX) Design

Designing an Android application interface that is user-friendly and easy to use by students or lecturers. The interface design will have a main feature, namely “Today's Attendance”. This feature allows users to see a recap of today's attendance. Users can also use the search engine to filter search categories such as name, student ID number, class, and attendance status.

2) User Interface Design

Designing an intuitive user experience to ensure that the attendance process runs smoothly. Users, who are defaulted as guests, will be able to easily select the menu they need without confusion. The attendance process will be designed to be easy so that students can easily register their fingerprints on the fingerprint sensor connected to the Android application. Lecturers have access as administrators and can manage student attendance data efficiently.

3) Module Development

Develop modules such as student fingerprint registration, fingerprint attendance, student attendance monitoring, student status search engine, and login for administrators or lecturers. The fingerprint registration module will allow students to register their fingerprints on a sensor connected to a microcontroller, and this data will be stored in a database for attendance purposes. The fingerprint attendance module allows students to check in and out using a fingerprint device connected to the database. The student monitoring module allows Android app users to monitor student attendance directly through the app. The student status search module allows users to specifically search for a student's status, whether they have checked in or out, and can search for students by name, student ID number, and class. The login module ensures that only registered users can access features such as managing student attendance records and changing data. The admin feature is specifically for lecturers who need student attendance data for academic assessment and auditing purposes.

The programming flowchart on the ESP32 microcontroller explains the algorithm on the fingerprint device. This flowchart outlines how the programming on the ESP32 works. The following is the ESP32 programming flowchart in Figure 4.

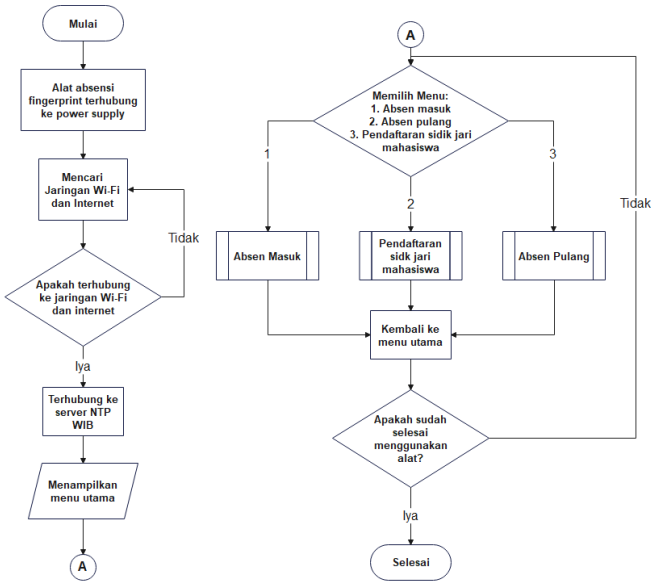

Figure 4. Flowchart of the ESP32 program for the fingerprint attendance device.

The workflow of the fingerprint device in the Smart

Workspace Attendance System can be seen in the flowchart above. The process begins with the device connecting to a power supply, namely an adapter with a voltage of 5 volts and a current of 1 ampere. Then, the program on the ESP32 microcontroller will attempt to connect to the Wi-Fi network that has been registered in the program and connect to the internet. If the Wi-Fi and internet search process fails, the program will continue to search for the Wi-Fi network and internet. If successful, the program will connect to the NTP server to obtain the time in the western part of Indonesia (WIB, GMT +7). This time will later be used to determine when students arrive and leave. After that, the LCD module will display the main menu of the program. The flowchart for the main menu can be seen in Figure 5.

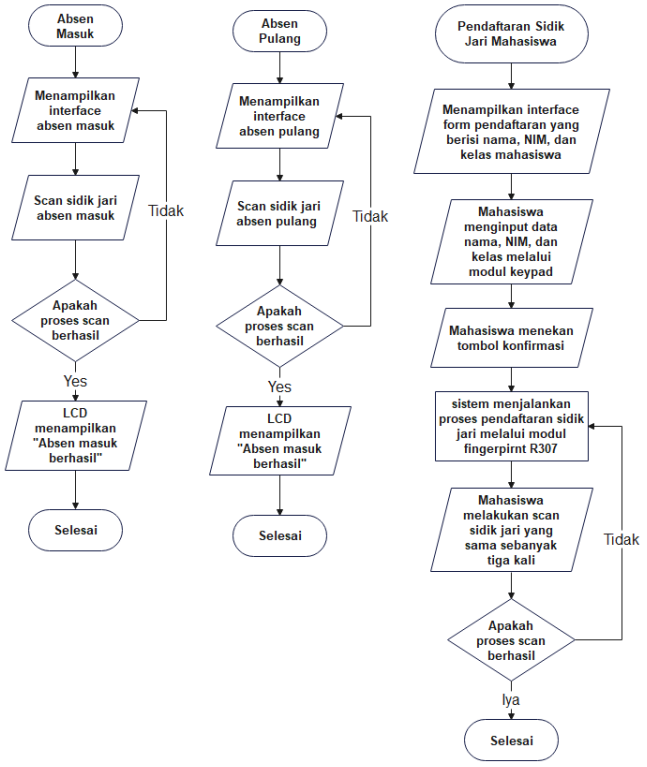

Figure 5. Main menu flowchart of the fingerprint attendance system.

The first option on the main menu is the attendance check-in option. On this menu, students can check in by scanning their fingerprints on the R307 fingerprint module. If successful, the LCD will display a message indicating that the attendance check-in was successful. If unsuccessful, the system will return to the attendance check-in interface. The same applies to the second option on the main menu, which is attendance check-out. The process for checking out is the same as checking in, except that the data sent to the database regarding when students check out is specific to checking out, not checking in.

The third option on the main menu is the student fingerprint registration option. On this menu, students will register their personal data and fingerprints to be able to perform the attendance check-in and check-out processes on options one and two on the main menu. The data entered by students is their name, student ID number, and class on the data entry form. After that, students will be asked to scan their fingerprints on the same finger three times. If the fingerprint scanning process is successful, the student has successfully registered their fingerprints. The name, student ID number, and class data of each student will be stored in the database based on the student's fingerprint template ID. That way, each student will have individual data that is different from other students.

E. Android Application Design

The design of the Android application can be seen in the flowchart in Figure 6. When the application is opened, it will display a splash screen. After that, the application will display a page for monitoring today's attendance.

Figure 6. General flowchart of the Android attendance application.

Next, the application will attempt to connect to the internet network on the smartphone [14]. If it fails, it will continue to search for an internet network. However, if it succeeds, the application will connect to the database to retrieve today's attendance data. This process can be seen in the flowchart in Figure 7.

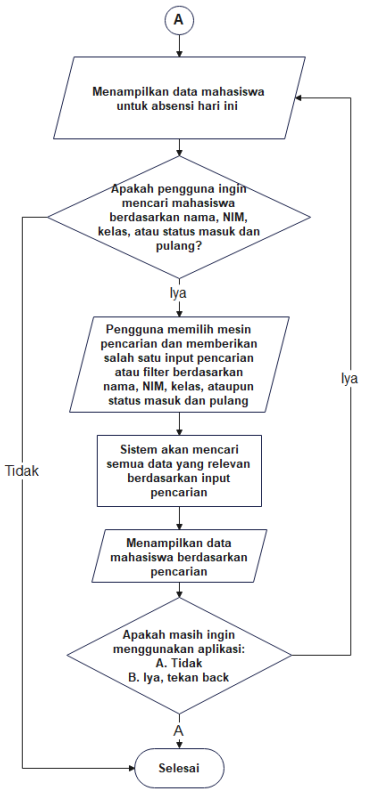

Figure 7. Flowchart of the data retrieval process for today's attendance in the Android application.

The Android application will request today's attendance data from the database. After that, the attendance data will be displayed in the form of box icons. Information about student attendance will be displayed in these boxes. This information includes the student's name, student ID number, class, and time.

F. Database Design

The development of a MariaDB database to support Android applications involves steps designed to ensure data efficiency and security. The database design stages are as follows.

1) Planning

This stage includes determining the data requirements to be stored in the database, such as student information, fingerprint data, and attendance records.

2) Database Schema Design

This step involves designing the structure of tables and determining the relationships between tables that will be used to store and manage attendance data efficiently.

3) Implementation

Implementation is carried out by creating tables and relationships that have been designed previously in MariaDB, ensuring that the database structure is in accordance with the application requirements [15]. The database connected to the web server uses private networking to obtain today's attendance data. Meanwhile, the Android application only uses the internet connected to the domain to obtain today's attendance data.

4) Integration with Fingerprint System

This stage includes connecting the database to the fingerprint sensor device used for student attendance registration and verification. In addition, integration is also carried out with the Android application to synchronize attendance data in real-time.

The process of installing and setting up Git and Docker-based applications is illustrated in this flowchart. First, before starting the process, users must ensure that Git is already installed; if not, they must install Git first. Once installed, users can clone the database project image with Git, then fill in the ".env" file or perform the configuration.

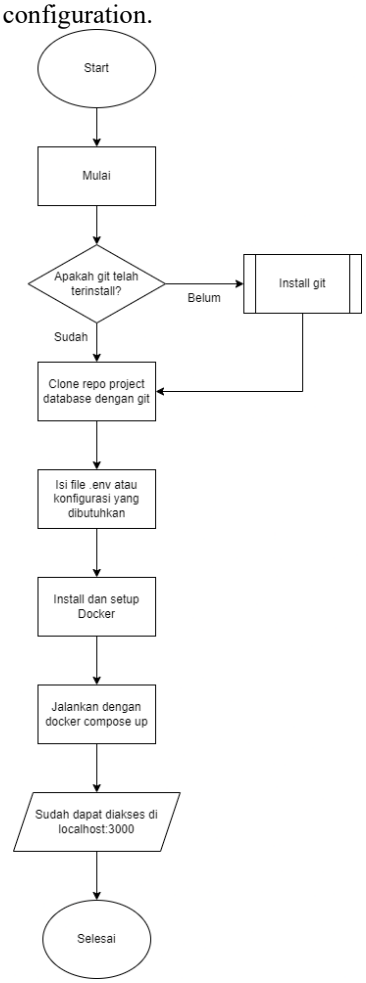

Figure 8. Flowchart of Git and Docker-based application installation and deployment process.

After installing and configuring Docker, the command "docker compose up -d" is used to run the project. The application can be accessed via a web browser at the address "localhost:3000", which indicates that the application is running correctly. Once the installation and configuration process is complete, users can use the application as needed. Details are provided in Figure 8.

G. Testing

The testing conducted on the Smart Workspace Attendance System project was to determine how efficient the time required to deploy and access data on the attendance system, which we can refer to as a cloud instance, was. This involved the timing of data transmission/reception from the fingerprint attendance device to the database and web server/Android application. Additionally, the researcher also conducted interviews with students regarding the web server and Android application.

These interviews were conducted to determine whether the Android application and database system created could help students and lecturers in facilitating the student attendance process. The following is an explanation of the testing techniques used.

1) Speed Testing when Deploying Cloud Instance Programs

Speed testing when deploying (building and activating) a cloud instance program is done by collecting the average value between runtime trials. The data collected is the processing speed listed on the page that records the gap between the processing and ready conditions/statuses. The following is the equation used to find the average (mean) of the collected data.

$$Mean = \frac{\sum Xi}{n} \quad (1)$$

Explanation:
Mean = Average
∑Xi = Sum of each value in the data
n = Number of data

2) Speed Testing when Accessing Cloud Instance Programs by Android Tools/Applications

Speed testing when accessing programs from cloud instances (databases & APIs) by Android tools/applications is done by collecting the average data transmission values. The data collected is the processing speed listed on the page that records the traffic gap between the client (Android tool/application) and the cloud instance. The following is the equation used to find the average (mean) of the collected data.

$$Mean = \frac{\sum Xi}{n} \quad (1)$$

Explanation:
Mean = Average
∑Xi = Sum of each value in the data
n = Number of data

3) Speed Testing when Accessing Cloud Instance Programs by Users

Speed testing when accessing programs from cloud instances (web servers) by users who typically use web browsers is conducted by collecting average data transmission values. The data collected is the processing speed listed on the page that records the gap between client (user) traffic and cloud instances. The following is the equation used to find the mean of the collected data.

$$Mean = \frac{\sum Xi}{n} \quad (1)$$

Explanation:
Mean = Average
∑Xi = Sum of each value in the data
n = Number of data

4) Interviews with Students Regarding Android Applications and Database Systems

In the testing conducted through interviews, the researcher asked questions about how effective the attendance system was in facilitating the student attendance process. The data from the questionnaire was collected and processed to determine the effectiveness of the attendance system.

A parenthetical statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). The serial comma is preferred: "A, B, and C" instead of "A, B and C."

III. RESULT AND DISCUSSION

Based on the design created in Chapter 3, the fingerprint attendance device was constructed in accordance with the electrical and mechanical designs. From the design process, the researcher carried out the assembly process by placing each electrical module in an easily accessible location. The results of the assembly of the device can be seen in Figure 9.

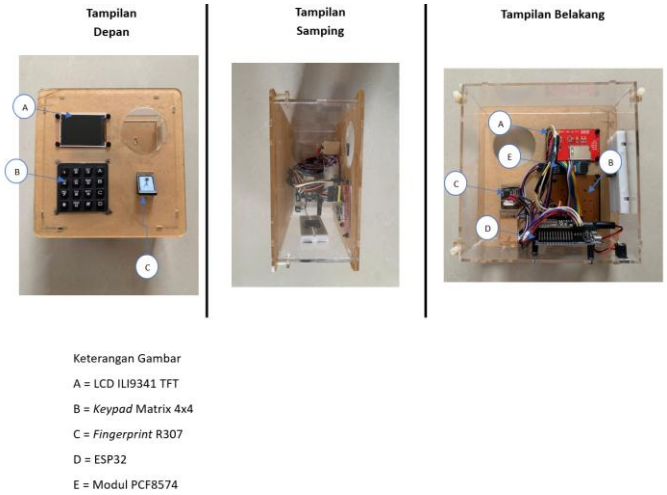

Figure 9. Final assembled prototype of the fingerprint attendance device.

During the assembly process, researchers installed the ILI3941 LCD module, 4x4 keypad, and R307 fingerprint sensor on the front of the device. Meanwhile, the I2C and ESP32 modules were placed inside the device to protect them from interference that could cause the wiring between modules to disconnect.

An Android application is used as a tool to monitor the attendance of each student. Lecturers and students can find out which students have checked in and out. In addition, this application also shows the expected time for attendance.

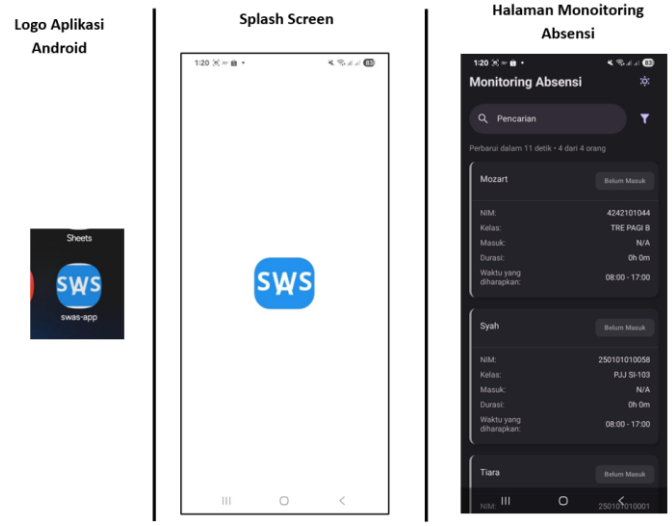

Figure 10. Splash screen and initial interface of the Android attendance application.

The Android application interface can be seen in Figure 10. When the application is opened, it will display a splash screen. After that, the application will display today's attendance monitoring page. On this page, students can use the search engine or filter feature to search for students based on name, class, and attendance status.

In addition, the researcher designed a main dashboard display for the web server application that displays the title "Create Table" to add database tables, for example, "data_mahasiswa" which contains sub-data such as name, student ID number, class, study program, entry-exit timestamp, etc. This data is placed in "Columns." This allows administrators or lecturers to manage, monitor, edit, and even delete data. This web server dashboard can be seen in Figure 11.

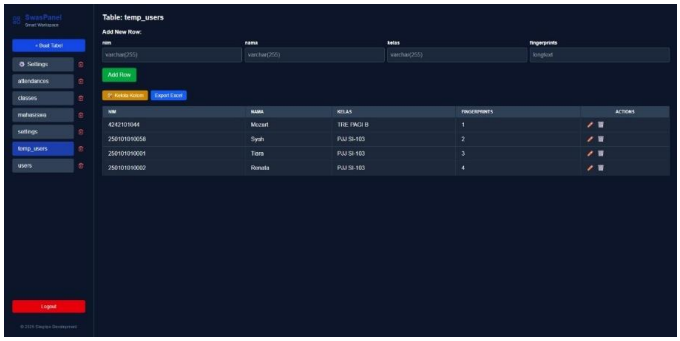

Figure 11. Web server dashboard interface for database and student data management.

In addition, administrators or lecturers can view students' attendance records. This feature can be found in the attendance section of the web server dashboard menu. The student attendance dashboard can be seen in Figure 12.

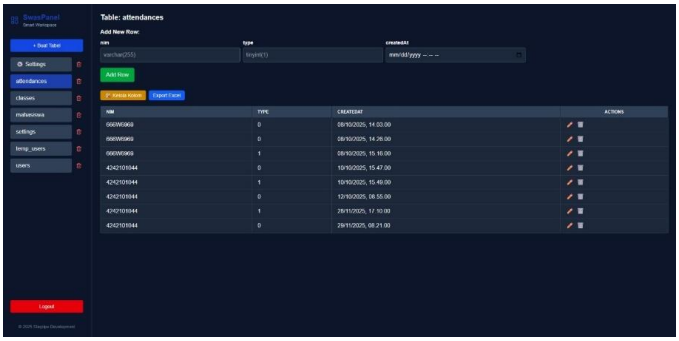

Figure 12. Student attendance record dashboard on the web server.

Thus, through the dashboard in Figure 11 and Figure 12, administrators or lecturers can manage student attendance data. This management can be used as a benchmark for student assessment. In addition, student attendance data during practical can be used by the campus for auditing purposes.

The researcher has obtained and recorded the following data as accurately as possible using the tools and materials available during the testing listed, to get the maximum potential from the software that has been designed and deployed. The following is a summary of information for setting up the entire environment in this cloud instance.

Figure 13. Overview of Docker build performance, including build duration, cache utilization, and parallel execution during system deployment.

TABLE I
BUILD TIMING FROM DOCKER WEB SERVER, API, AND DATABASE

No.	Category	Value
1	Cached Steps	8
2	Non-Cached Steps	11
3	Total Steps	19
4	Cached Usage	8/11
5	Dominant Activity	Executions
6	Parallel Execution	Almost all are parallel
	Accumulated Time	46,2 second
	MariaDB Image Pull	12,8 second
	Total Build Time (real time)	60,6 second

Table 1 shows the build and deployment process on the web server, API, and database using Docker. The total real-time build time was 60.6 seconds, with overall execution and parallel processing activities almost entirely demonstrating that the container architecture used was able to speed up the service initialization process. The MariaDB image pull time of 12.8 seconds is an important factor because it is directly related to

the readiness of the database as the center of attendance data storage. This shows that the database system can be activated quickly and is ready to accept connections from APIs and Android applications.

Speed when Deploying Cloud Instance Programs In the testing conducted through interviews, the researcher asked questions about how effective the attendance system was in facilitating the student attendance process.

1) Speed when Deploying Cloud Instance Programs

The testing did not stop there; there was also a post-production stage, which involved real-time actions and experiences gained when operating tools and software in production/stable conditions. The researcher conducted five experiments to determine the delay or how fast/slow the tools and Android applications were when trying to access or even change the information/content of the database.

The following is the timing data required for tools and Android applications to access the API from the web server. The researcher used the basic calculations just discussed to obtain the mean response time.

TABLE 2
LATENCY/DELAY WHEN STARTING UP THE WEB SERVER, API, AND DATABASE

No.	Execution Step	Expected Time	Time/Result
1	Runtime ke-1	0,05 second	0,066 second
2	Runtime ke-2	0,05 second	0,065 second
3	Runtime ke-3	0,05 second	0,078 second
4	Runtime ke-4	0,05 second	0,079 second
5	Runtime ke-5	0,05 second	0,068 second
	Average (Second)		0,071 second

$$Mean = \frac{\sum Xi}{n} \quad (1)$$
$$= \frac{(0{,}066 + 0{,}065 + 0{,}078 + 0{,}079 + 0{,}068)}{5}$$
$$= 0{,}071\ detik$$

In Table 2 above, it can be seen that the latency when turning on the web server, API, and database is on average 0.071 seconds. This average data is still very close to the expected time. This data indicates that the back-end service initialization process runs stably and consistently in each runtime experiment. This speed is important because it is a key component before the fingerprint tool and Android application can perform data transactions to the database. Therefore, the smaller the initial delay, the faster the system will be ready for use in real-world conditions.

2) Speed when Accessing Cloud Instance Programs by Android Tools/Applications

The researchers conducted five experiments to determine the delay or how fast/slow Android tools and applications are when trying to access or even change database information/content.

The following data shows the time required by Android devices and applications to access the API from the web server. The researchers used the basic calculation discussed above to obtain the average response time (mean). The following is a description of the standardized calculation method.

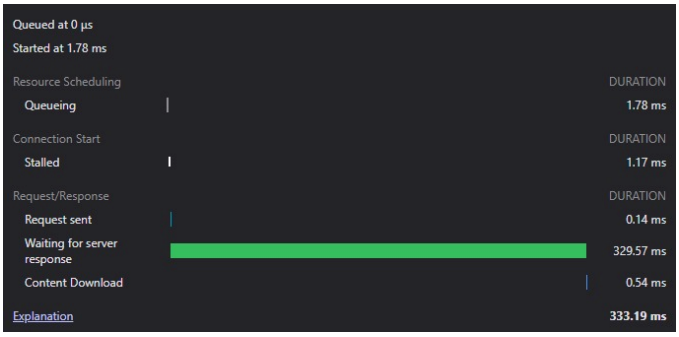

Figure 14. API access runtime load speed benchmark of 1st attempt by the module.

TABLE 3
LATENCY/DELAY WHEN ACCESSING THE API FROM THE CLIENT (ANDROID DEVICE/APPLICATION)

No.	Execution Step	Expected Time	Time/Result
1	Runtime ke-1	0,05 second	0,333 second
2	Runtime ke-2	0,05 second	0,335 second
3	Runtime ke-3	0,05 second	0,334 second
4	Runtime ke-4	0,05 second	0,333 second
5	Runtime ke-5	0,05 second	0,335 second
	Average (Second)		0,335 second

$$Mean = \frac{\sum Xi}{n} \quad (2)$$
$$= \frac{(0{,}333 + 0{,}335 + 0{,}334 + 0{,}333 + 0{,}335)}{5}$$
$$= 0{,}335\ detik$$

In Table 3 above, you can see the latency when the fingerprint device and Android application access the API to communicate with the database. The average response time obtained is 0.335 seconds. This value is influenced by the process of sending data through the internet network and the processing of requests by the API before the data is stored and retrieved from the database. Although this value is greater than the internal server latency, the time is still considered safe for a real-time attendance system. Thus, the attendance recording process remains responsive for users on both the device and Android application sides.

3) Speed Testing when Accessing Cloud Instance Programs by Users

After observing and verifying the production environment, it is now time to access the results directly as users who will act as lecturers or students. The following is the data on the time required for users to access the front-end of the web server. Researchers still use the basic mean formula to obtain the average response time of the site framework. The following is a description of the standardized calculation method.

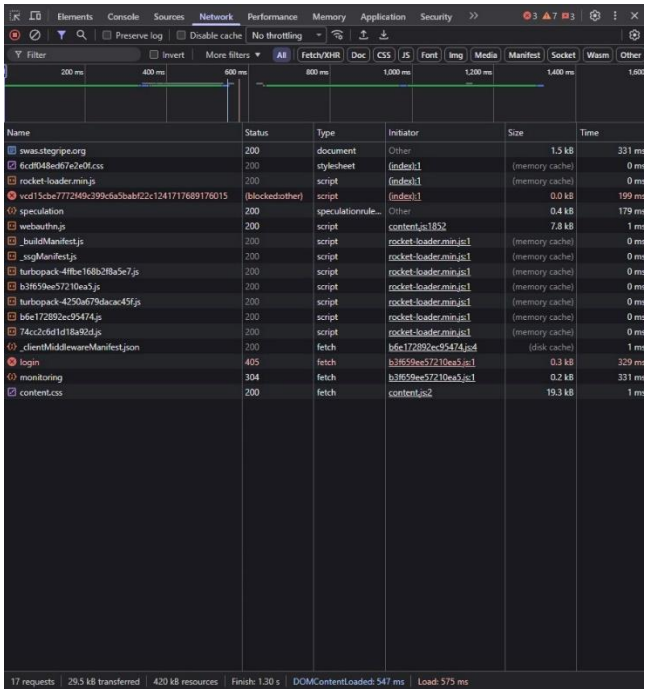

Figure 15. Front-page (GUI) access runtime load speed benchmark of 1st attempt by the user.

TABLE 4
LATENCY/DELAY WHEN ACCESSING A WEB SERVER VIA A WEB BROWSER FROM A CLIENT (USER)

No.	Execution Step	Expected Time	Time/Result
1	Runtime ke-1	0,75 second	0,575 second
2	Runtime ke-2	0,75 second	0,547 second
3	Runtime ke-3	0,75 second	0,526 second
4	Runtime ke-4	0,75 second	0,515 second
5	Runtime ke-5	0,75 second	0,542 second
	Average (Second)		**0,541 second**

$$Mean = \frac{\sum Xi}{n} \quad (3)$$
$$= \frac{(0{,}575 + 0{,}547 + 0{,}526 + 0{,}515 + 0{,}542)}{5}$$
$$= 0{,}541\ detik$$

Based on Table 4, the average delay for users accessing the web server through a browser is 0.541 seconds. This time is faster than expected. This indicates that the web interface connected to the database is capable of displaying attendance data quite easily and quickly. This response is important for administrators or lecturers when monitoring, managing, or checking student attendance records because all data is taken directly from a centralized database.

4) Results of Interviews with Students Regarding Android Applications and Database Systems

In the testing conducted through interviews, the researcher asked questions about how effective the attendance system was in facilitating the student attendance process. The questionnaire was the result of interviews conducted with fifteen students. The questionnaire data was collected and processed to determine the effectiveness of the attendance system.

TABLE 5
INTERVIEW RESULTS

No.	Execution Step	Poor(%)	Adequate(%)	Good(%)
1	Does the fingerprint attendance device simplify the attendance process?	0%	0%	100%
2	Does the database system connecting the application to the database function properly?	0%	0%	100%
3	Is the fingerprint attendance device superior to the current manual attendance process?	0%	0%	100%

The interview results in Table 5 show very positive responses from all respondents. From these results, it was found that 100% of the 30 students rated the fingerprint device, database system, and Android application as working well and more effective than manual attendance. This data is an important non-technical indicator because it proves that the integration between the device, API, and database is not only successful in terms of the system but also directly beneficial to users. Thus, the attendance system that was developed is very well received by end users.

B. Discussion

Based on the test results, the fingerprint-based attendance system integrated with the Android application and database showed quite good performance. From a technical perspective, the deployment time for cloud instances using Docker was relatively fast and stable, and the API access latency by tools and Android applications was still within an acceptable range for real-time use. This shows that the system architecture used, from ESP32 and web servers to databases, is well integrated and supports the attendance process without significant obstacles. Compared to manual attendance systems, this system is clearly more efficient because it reduces recording time and minimizes data input errors. The following is an illustration of the implementation of the interrelationship between

When compared to previous studies, the results of this study are in line with biometric-based studies which state that fingerprint attendance is more accurate and difficult to manipulate than manual or RFID methods. The main difference in this study lies in the use of an Android application as a medium for real-time attendance monitoring, which provides easy access for lecturers and students. In addition, the interview results showed a very positive response from students, with all respondents rating this system as better and more helpful than manual attendance.

IV. CONCLUSION

This study has successfully designed and implemented a fingerprint-based attendance monitoring system integrated with a centralized database and an Android application. The system is able to record student attendance accurately in real time, store the data securely in the database, and display attendance information through the Android application in a clear and structured manner. The test results show that the fingerprint sensor works reliably for user identification, while the communication between hardware, server, and mobile application runs as expected with minimal delay. Although the system performs well under normal conditions, limitations still exist, particularly in terms of dependency on network stability and sensor response under certain usage scenarios. Therefore, future work may focus on improving system scalability, enhancing data security mechanisms, and optimizing the mobile application interface to support broader deployment and long-term use in academic environments.

REFERENCES

[1] W. Dinasari, A. Budiman, and D. Ayu Megawaty, "SISTEM INFORMASI MANAJEMEN ABSENSI GURU BERBASIS MOBILE (STUDI KASUS : SD NEGERI 3 TANGKIT SERDANG)," *Jurnal Teknologi dan Sistem Informasi (JTSI)*, vol. 1, no. 2, pp. 50–57, 2020, [Online]. Available: http://jim.teknokrat.ac.id/index.php/JTSI

[2] S. Al Amin, M. A. Islam, and M. S. Islam, "A fingerprint based smart attendance and security system using IoT and ultrasonic sensor," in *2021 International Conference on Automation, Control and Mechatronics for Industry 4.0, ACMI 2021*, Institute of Electrical and Electronics Engineers Inc., Jul. 2021. doi: 10.1109/ACMI53878.2021.9528092.

[3] P.E.S. Dita and C. Bella, "RANCANG BANGUN KEAMANAN PINTU DENGAN SENSOR SIDIK JARI BERBASIS ARDUINO," *Jurnal Portal Data*, vol. 2, no. 2, 2022.

[4] V. M. Ekowati, A. S. Supriyanto, T. Miranti, and M. Machfud, "An Empirical Approach to Evaluate Employee Performance Using Finger Print Attendance," *Quality - Access to Success*, vol. 25, no. 199, pp. 57–64, 2024, doi: 10.47750/QAS/25.199.07.

[5] E. G. Abdulkadhim, "Design and Develop an Attendance System Based on Fingerprint and Arduino Board," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Mar. 2021. doi: 10.1088/1742-6596/1804/1/012011.

[6] M. Juansen and S. Simatupang, "Integrasi Mesin Absensi dan Pusher Notification pada Sistem Informasi Akademik Sekolah Untuk Monitoring Absensi Real-Time," *Journal of Computer System and Informatics (JoSYC)*, vol. 4, no. 4, pp. 1028–1035, Aug. 2023, doi: 10.47065/josyc.v4i4.3840.

[7] M. Rafai, S. Solikhun, and M. Safii, "PERANCANGAN ABSENSI QR CODE MAHASISWA BERBASIS WEBSITE PADA STIKOM TUNAS BANGSA PEMATANG SIANTAR MENGGUNAKAN METODE AGILE," *Jurnal Manajamen Informatika Jayakarta*, vol. 4, no. 1, p. 51, Feb. 2024, doi: 10.52362/jmijayakarta.v4i1.1303.

[8] S. Khan, A. Akram, and N. Usman, "Real Time Automatic Attendance System for Face Recognition Using Face API and OpenCV," *Wirel Pers Commun*, vol. 113, no. 1, pp. 469–480, Jul. 2020, doi: 10.1007/s11277-020-07224-2.

[9] M. K. Kah Wen, N. binti Ahmad, and S. H. binti Ruslan, "Arduino based outing and attendance system for boarding school students," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 20, no. 2, pp. 1053–1061, Nov. 2020, doi: 10.11591/ijeecs.v20.i2.pp1053-1061.

[10] A. Boza-Chua, L. Andrade-Arenas, and A. Roman-Gonzalez, "Mobile application for control and management of citizen security," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 2, pp. 1063–1074, Feb. 2023, doi: 10.11591/ijeecs.v29.i2.pp1063-1074.

[11] N. Yahya and S. S. Maidin, "Hybrid agile development phases: The practice in software projects as performed by software engineering team," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 3, pp. 1738–1749, Mar. 2023, doi: 10.11591/ijeecs.v29.i3.pp1738-1749.

[12] O. A. Akinola, S. O. Olopade, and A. S. Afolabi, "Development of mobile and desktop applications for a fingerprint-based attendance management system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 1, pp. 570–580, Oct. 2021, doi: 10.11591/ijeecs.v24.i1.pp570-580.

[13] X. Yin, Y. Zhu, and J. Hu, "3D Fingerprint Recognition based on Ridge-Valley-Guided 3D Reconstruction and 3D Topology Polymer Feature Extraction," *IEEE Trans Pattern Anal Mach Intell*, vol. 43, no. 3, pp. 1085–1091, Mar. 2021, doi: 10.1109/tpami.2019.2949299.

[14] B. Munthe, Herman, A. Arifin, B. S. Nugroho, and E. Fitriani, "Online Student Attendance System Using Android," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Jun. 2021. doi: 10.1088/1742-6596/1933/1/012048.

[15] A. S. M. Noor, A. F. C. Fauzi, A. A. C. Fauzi, N. Ahmad, and M. S. M. Arifin, "An automate failure recovery for synchronous distributed database system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 28, no. 2, pp. 973–979, Nov. 2022, doi: 10.11591/ijeecs.v28.i2.pp973-979.