

Android Face Recognition Based Attendance Management Application

Asma A. Shaikh¹, Tanmay Dilip Khot², Akshat Rakesh Vhora³,
Swarup Vikas Jagtap⁴

^{1,2,3,4}Artificial Intelligence & Data science, ADCET, Ashta, India

Abstract

This paper presents an Android-based Attendance Management System leveraging facial recognition to automate and streamline attendance tracking. By integrating real-time face detection via the mobile camera, the system minimizes manual intervention and reduces proxy risks. It offers a dual-mode mechanism: automatic attendance through facial recognition and manual fallback via roll number or student ID. The system employs deep learning models (e.g., FaceNet) for accurate face recognition, with robust error handling for conditions like unrecognized faces or low light. Attendance data is securely stored in a cloud database (Firebase), ensuring scalability, reliability, and easy reporting. The modular architecture, comprising image capture, preprocessing, detection, and backend integration, allows future enhancements such as multi-face detection or broader mobile integration. Experimental results confirm the system's accuracy, reliability, and operational efficiency under varied conditions. Beyond academic use, this approach holds promise for workplaces, events, and secure access systems, demonstrating the potential of machine learning to automate administrative processes, improve user experience, and reduce human error.

Keywords: Facial Recognition, Secure Data Storage, Accuracy, Android-based, Automatic Attendance

1. Introduction

In modern educational institutions and organizations, efficient attendance management is crucial for maintaining accurate records and ensuring fair participation. Traditional attendance systems, whether manual sign-ins or basic digital registers, often fall short due to issues like human error, proxy attendance, and the significant time investment required. These shortcomings highlight the urgent need for a more advanced, automated solution that can enhance both efficiency and reliability.

This project introduces a smart Android-based Attendance Management System powered by facial recognition technology. By leveraging the capabilities of real-time face detection through a mobile device's camera, the system eliminates the need for manual attendance marking and significantly reduces the chances of proxy entries. The core aim is to create a system that not only automates attendance tracking but also ensures security, scalability, and ease of use.

The application offers two modes for flexibility: automatic mode where facial recognition marks attendance seamlessly, and a manual fallback mode where users can input attendance using student IDs or roll numbers when needed. The system securely stores attendance data and provides instant visual feedback to users, confirming successful detection and submission.

The project integrates machine learning techniques with mobile app development to deliver a robust, intuitive

solution. This system is particularly beneficial for educational settings, where managing large student groups can be challenging, but it is also adaptable for corporate or organizational use. By combining automation, security, and a user friendly interface, the proposed solution promises to transform traditional attendance practices into a modern experience

2. Literature Review

The efficiency and accuracy of attendance systems have been widely studied in academic and industrial research. Traditional attendance methods, including manual roll calls and RFID-based tracking, suffer from limitations such as human error, proxy attendance, and time inefficiencies. The literature reviewed below provides insight into the advancements in automated attendance solutions using biometric recognition and artificial intelligence.

Over the years, significant progress has been made in face recognition and attendance systems. Notable works like FaceNet (Schroff et al., 2015) and ResNet (He et al., 2016) introduced deep learning models that greatly improve recognition accuracy but come with high computational demands. MobileNetV2 (Sandler et al., 2018) and TensorFlow Lite (2017) focused on optimizing models for mobile devices, balancing performance with efficiency, although they face trade-offs in accuracy or limited support for complex tasks. Meanwhile, Kumar and Sharma (2017) proposed an automated attendance system using facial ID, emphasizing preprocessing to handle variations in lighting and facial features.

Further research explored cloud-based solutions (Singh Das, 2019), leveraging platforms like Firebase for real-time data handling, and IoT-based monitoring systems (Zhang Lee, 2018) that integrate biometrics for better tracking. Proxy detection using facial recognition (Patel Mehta, 2018) addressed classroom fraud, though limited to specific environments. Comparative studies (Kaur Verma, 2018) evaluated the performance of classic vs. deep learning methods, highlighting the need to adapt to environmental conditions. More recently, edge AI models (Reddy Krishnan, 2020) assessed the deployment efficiency on mobile devices, though resource limitations remain a challenge. Together, these studies lay a strong foundation for designing modern, efficient, and scalable attendance systems.

3. System Architecture

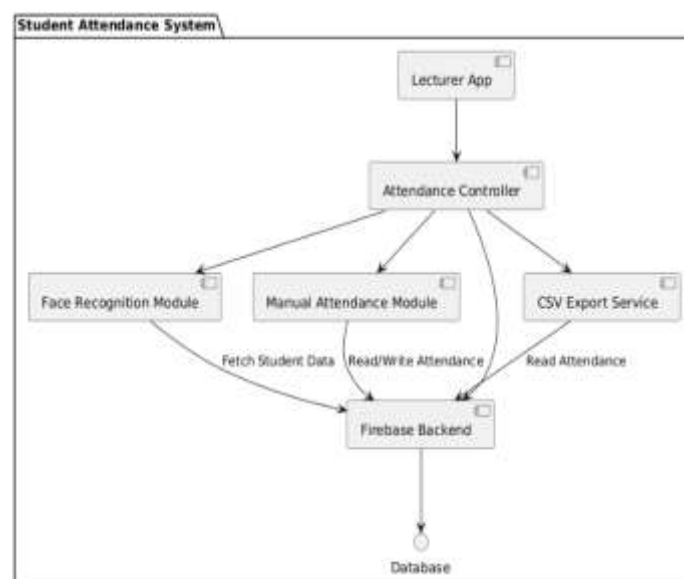


Fig 1. System Architecture

The Android-based Attendance Management System is designed following object-oriented principles to ensure modularity, scalability, and maintainability. The system is composed of several key components, each handling a distinct responsibility within the attendance process.

Further research explored cloud-based solutions (Singh Das, 2019), leveraging platforms like Firebase for real-time data handling, and IoT-based monitoring systems (Zhang Lee, 2018) that integrate biometrics for better tracking. Proxy detection using facial recognition (Patel Mehta, 2018) addressed classroom fraud, though limited to specific environments. Comparative studies (Kaur Verma, 2018) evaluated the performance of classic vs. deep learning methods, highlighting the need to adapt to environmental conditions. More recently, edge AI models (Reddy Krishnan, 2020) assessed the deployment efficiency on mobile devices, though resource limitations remain a challenge. Together, these studies lay a strong foundation for designing modern, efficient, and scalable attendance systems.

Teacher Module: This module allows teachers to log in securely, select specific lectures, mark student attendance, and export attendance reports for review or record-keeping. Core methods include `login()` for authentication, `selectLecture()` to choose the relevant session, `markAttendance()` to update student attendance, and `exportAttendance()` to generate attendance records.

Student Module: This module represents student profiles, including essential details such as `studentID`, `name`, `rollNumber`, `course`, and `faceData` used for facial recognition.

Lecture Module: This component manages lecture details such as the subject, date, and time. It provides methods like `getLectureDetails()` to retrieve relevant lecture information.

Attendance Module: Responsible for tracking each student's attendance status within a lecture, this module includes methods like `markAttendance()` to update status and `getAttendanceStatus()` to retrieve the current attendance record for a student.

Face Recognition Module: This critical component handles the facial recognition tasks using pre-trained machine learning models. Important methods include `trainModel()` to prepare the facial recognition system and `recognizeFace()` to identify students during attendance marking.

Database Design: The system relies on a cloud-integrated relational database to store all data related to teachers, students, lectures, and attendance records. This integration ensures real-time data access, secure storage, and easy scalability as the institution's needs grow.

4. Methodology

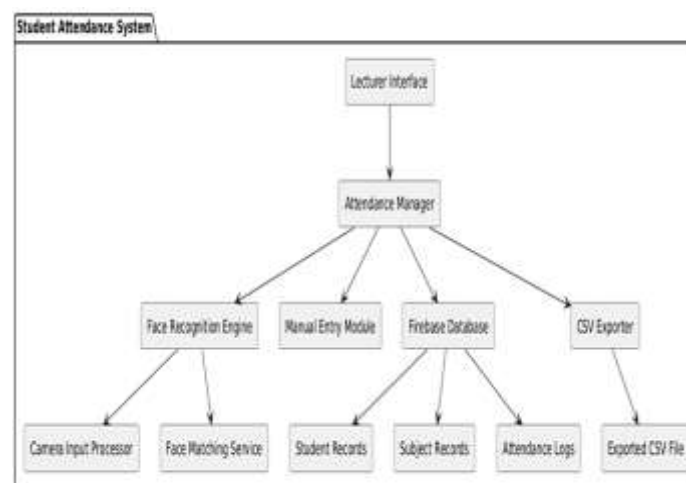


Fig 2. Methodology

Login Page: The login page serves as the initial gateway to the application. It is intricately connected to a database that securely stores user information, including IDs and passwords. Based on the user's type (whether regular user or admin), the features available within the application are dynamically defined. This ensures that users only have access to functionalities relevant to their roles.

Dashboard: The dashboard serves as the central hub presenting the application's key features. Depending on the user type—regular user or admin—the dashboard customizes its display. For regular users, it showcases a streamlined set of essential features, ensuring a clean and focused interface. On the other hand, administrators are granted visibility to all features, allowing them to oversee and manage the application comprehensively.

Camera: The camera section is a pivotal component that facilitates the connection and utilization of the USB camera attached to the microscope. This functionality allows users to view the microscope's output directly on the screen. Users are empowered to capture high-quality images, providing a visual record of their observations. This feature enhances the documentation process, enabling users to capture and store images seamlessly.

5. Implementation

For the implementation of our Simple Desktop System, we opted for Python as the primary programming language due to its simplicity and versatility. Leveraging the Tkinter library, we designed a straightforward graphical user interface (GUI) featuring prominently displayed buttons for core functionalities. Upon launching the application, users are greeted with a clean main window housing buttons labeled "File Manager," "Task Manager," and "Communicator." These buttons serve as entry points to the respective modules of the desktop system. Clicking on the "File Manager" button prompts the system to initiate the file management module, enabling users to navigate, organize, and manipulate their files and directories. Similarly, selecting the "Task Manager" button activates the task management module, allowing users to create, prioritize, and monitor their tasks effectively. Lastly, clicking on the "Communicator" button triggers the communication module, facilitating seamless interaction through messaging features.



Fig 3. User Login

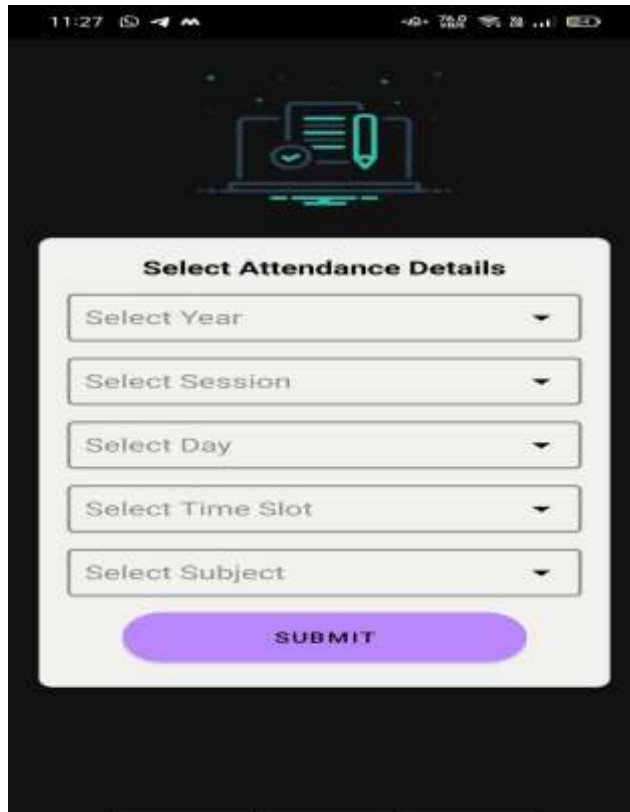


Fig 4. Attendance Details

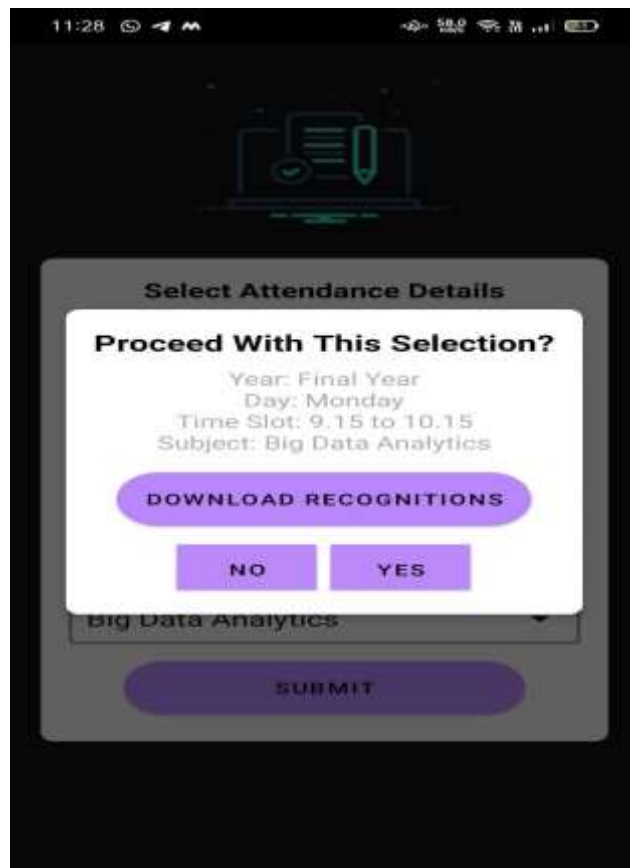


Fig 5. Recognitions for download

6. Conclusion

In conclusion, this report has outlined the development and implementation of simple microscopic image reporting software tailored to meet the needs of small businesses and individuals. At the core of this project was the recognition that microscope users, particularly those without extensive technical expertise, often face challenges in effectively documenting and sharing their research findings. To address this, we have created a user-friendly and intuitive software solution that streamlines the process of capturing microscopic images and generating comprehensive reports. Through a straightforward interface, users can easily upload their microscopic observations, input relevant data, and produce professional-looking reports with minimal effort. The software's accessibility and affordability make it an ideal solution for a wide range of microscope users, from small-scale researchers to quality control professionals. Going forward, we are committed to ensuring the continued relevance and effectiveness of this microscopic image reporting software. By continuously gathering user feedback and adapting the solution to evolving needs, we aim to solidify its position as an indispensable tool for simplifying the report generation process and empowering microscope users to effectively communicate their research and findings. This software represents a significant stride towards enhancing efficiency, accessibility, and the overall experience for those who rely on microscopic analysis as part of their daily operations.

7. Performance comparison

The following table presents a performance comparison of various state-of-the-art face verification models on the Labeled Faces in the Wild (LFW) benchmark. It lists models such as DeepFace, DeepID2+, SphereFace, CosFace, ArcFace, FaceNet, and MobileFaceNet, comparing them based on training data size, number of networks (#Net), model size, and LFW accuracy. Notably, ArcFace (LResNet100E-IR) achieves the highest LFW accuracy of 99.83%, while MobileFaceNet offers a competitive accuracy of around 99.55% with an impressively small model size of just 4.0 MB, making it highly efficient for mobile applications. This table highlights the trade-offs between model size and accuracy, emphasizing MobileFaceNet's balance of compactness and strong performance.

Method	Training Data	#Net	Model Size	LFW Acc.
Deep Face [31]	4M	3	-	97.35%
DeepFR [32]	2.6M	1	0.5GB	98.95%
DeepID2+ [33]	0.3M	25	-	99.47%
Center Face [34]	0.7M	1	105MB	99.28%
DCFL [35]	4.7M	1	-	99.55%
SphereFace [20]	0.49M	1	-	99.47%
CosFace [22]	5M	1	-	99.73%
ArcFace (LResNet100E-IR) [5]	3.8M	1	250MB	99.83%
FaceNet [18]	200M	1	30MB	99.63%
ArcFace (LMobileNetE) [5]	3.8M	1	112MB	99.50%
Light CNN-29 [14]	4M	1	50MB	99.33%
MobileID [17]	-	1	4.0MB	97.32%
ShiftFaceNet [15]	-	1	3.1MB	96.00%
MobileFaceNet	3.8M	1	4.0MB	99.55%
MobileFaceNet (112 × 96)	3.8M	1	4.0MB	99.53%
MobileFaceNet (96 × 96)	3.8M	1	4.0MB	99.52%

Fig 6. Performance Comparison

8. References

1. F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)}, 2015, pp. 815–823.
2. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)}, 2016, pp. 770–778.
3. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted residuals and linear bottlenecks,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510–4520.
4. TensorFlow Team, “TensorFlow Lite: Enabling machine learning on mobile devices,” 2017. [Online]. Available: <https://www.tensorflow.org/lite>
5. A. Kumar and S. Sharma, “Automated attendance system based on face recognition,” International Journal of Engineering and Technology, vol. 7, no. 2, pp. 298–302, 2017.
6. R. Singh and T. Das, “Cloud-based attendance management using image processing,” International Journal of Advanced Research in Computer Science, vol. 10, no. 5, pp. 12–18, 2019.
7. M. Zhang and H. Lee, “Attendance monitoring system via IoT,” in Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings), 2018, pp. 1–5.
8. J. Patel and V. Mehta, “Proxy detection in classroom attendance systems using face recognition,” Journal of Engineering Research and Applications, vol. 8, no. 4, pp. 1–5, 2018.
9. D. Kaur and N. Verma, “Comparative study of face detection algorithms,” International Journal of Computer Applications, vol. 179, no. 2, pp. 25–31, 2018.
10. G. Reddy and S. Krishnan, “Evaluating performance of edge AI models,” Journal of AI and Data Mining, vol. 5, no. 3, pp. 223–230, 2020