FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 9 No. 12, December (Special Issue), 2025, pp 590 – 598
DOI: https://doi.org/10.33003/fjs-2025-0912-4243

DEVELOPMENT OF A SCALABLE AND REAL-TIME BIOMETRIC-BASED ATTENDANCE MONITORING SYSTEM

***[1]Kazeem Sodiq, [2]Adetunji Olusogo, [1]Saliu Abolanle, [3]Oyerinde Elisha, [1]Salau Solomon, [1]Ojosipe Adebisi, [1]Ishola Pelumi, [1]Adenuga Nofisat, [1]Uwah Wisdom and [1]Akinfe Olayinka**

[1]Department of Computer Engineering, Yaba College of Technology Lagos, Nigeria
[2]Department of Computer Science, Olabisi Onabanjo University, Ago-Iwoye, Nigeria
[3]Department of Computer Technology, Yaba College of Technology, Ojo, Nigeria

*Corresponding authors' email: kazeem.sodiq@yabatech.edu.ng

ABSTRACT

This research work introduces a scalable and real-time student attendance monitoring system designed to improve accountability and academic discipline through automation and surveillance. Traditional attendance methods like manual roll calls and RFID are often inefficient and prone to manipulation; in contrast, this system uses Face API JS a JavaScript-based deep learning facial recognition library to detect and identify students and lecturers via live video streams captured through webcams. Lecturers are the only authorized personnel to initiate attendance sessions, while students can access their attendance records through a secure web interface. The system incorporates Socket.IO for real-time, bidirectional communication between clients and the server, allowing for seamless updates and efficient attendance coordination. Administrators can monitor live classroom activities using mobile devices or built-in Raspberry Pi modules. Experimental evaluation across varying lighting and classroom conditions yielded an average face recognition accuracy of 95.7%, with a false acceptance rate of 2.1% and average detection latency under 1.2 seconds, confirming the system's robustness and responsiveness. In conclusion, the proposed solution offers an automated, non-intrusive, and transparent attendance tracking mechanism that significantly enhances institutional oversight, reduces human error, and ensures both student presence and lecturer participation.

Keywords: Automation and Surveillance, Facial Recognition System, Real-Time Attendance Monitoring, Raspberry Pi Integration, Socket IO Communication

INTRODUCTION

Attendance management is a critical aspect of higher education, as it ensures students remain actively engaged in their courses and provides institutions with reliable data for academic accountability. Traditional approaches such as roll calls, paper registers, and RFID cards, though widely used, are inefficient, time-consuming, and vulnerable to manipulation (Smith & Jones, 2022). Practices like proxy attendance compromise data integrity, while manual processes burden lecturers and consume valuable teaching time (Brown, 2020).

Technological interventions such as RFID and biometric systems have attempted to address these challenges. RFID-based solutions, however, still allow proxy attendance through card sharing. Biometric systems like fingerprint or iris scanning improve security but face issues of hygiene, sensor malfunctions, high cost, and user resistance, particularly in post-pandemic contexts. Consequently, institutions continue to seek cost-effective, scalable, and contactless alternatives.

Facial recognition technology offers a promising solution by automating attendance marking through live video feeds, eliminating the need for physical tokens or contact-based verification. System can pre-register by uploading facial images to a database; webcams can then authenticate attendance in real time, while administrative login safeguards against misuse. Raspberry Pi integration for live classroom monitoring further enhances transparency and efficiency (Johnson, 2022).

Deep learning approaches, such as FaceNet by Schroff et al. (2015), significantly improved recognition accuracy and now underpin many smart attendance applications. Hybrid methods combining traditional and deep learning techniques have further increased robustness against lighting, pose, and quality variations (Huda Al-Nayyef, 2024).

These advancements demonstrate a clear shift toward intelligent, web-based, and real-time attendance systems. By leveraging facial recognition and classroom surveillance, such systems can reduce administrative workload, minimize manipulation, and provide reliable data for both lecturers and administrators. This research contributes to this growing field by developing a scalable, real-time web-based attendance management system integrating face recognition technology, thus addressing the persistent limitations of traditional and semi-automated methods.

MATERIALS AND METHODS

System Requirement and Tools

The system is implemented using several integrated tools and components. High-resolution cameras capture students' facial data, while a Raspberry Pi processes these images using deep learning models for recognition. A 7-inch display screen provides real-time classroom visuals, and administrators can monitor activities remotely through their mobile phones. The software architecture combines Swagger for API documentation, Next.js for an efficient frontend interface, and NestJS for a structured backend built on Node.js. MySQL serves as the database for storing and organizing system records, while Amazon S3 offers secure cloud storage for large data files. The entire solution is hosted on a Virtual Private Server (VPS), ensuring dedicated resources and reliable system performance.

Conceptual Framework

The development of an automated student attendance system using facial recognition is grounded in multiple conceptual domains, including computer vision, real-time

communication, and biometric security. This section explores the foundational theories and technologies that inform the design and implementation of the proposed system. The figure 1 shows a conceptual framework of the entire system.

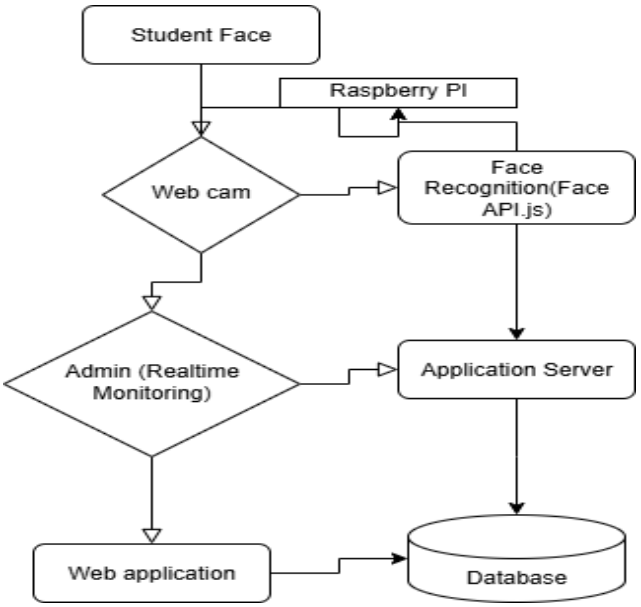

Figure 1: Conceptual Framework

Biometric authentication refers to the automated recognition of individuals based on their physiological or behavioral characteristics. Among various biometric modalities, facial recognition is widely adopted for its non-intrusive nature, ease of deployment, and public acceptance (Jain et al., 2004). Facial recognition systems involve two key processes: face detection and face identification.

In this research, Face API JS, a client-side JavaScript library built on TensorFlow.js, was used for both face detection and recognition. It performs feature extraction and compares facial embeddings using pre-trained deep learning models. This approach ensures that all processing occurs on the user's device, enhancing privacy and reducing server load.

Although the research uses Face API JS for performance and real-time capabilities, its architectural principles are influenced by FaceNet, especially the use of embeddings for comparing facial identities during attendance marking. The figure 2 presents a attendance monitoring system.

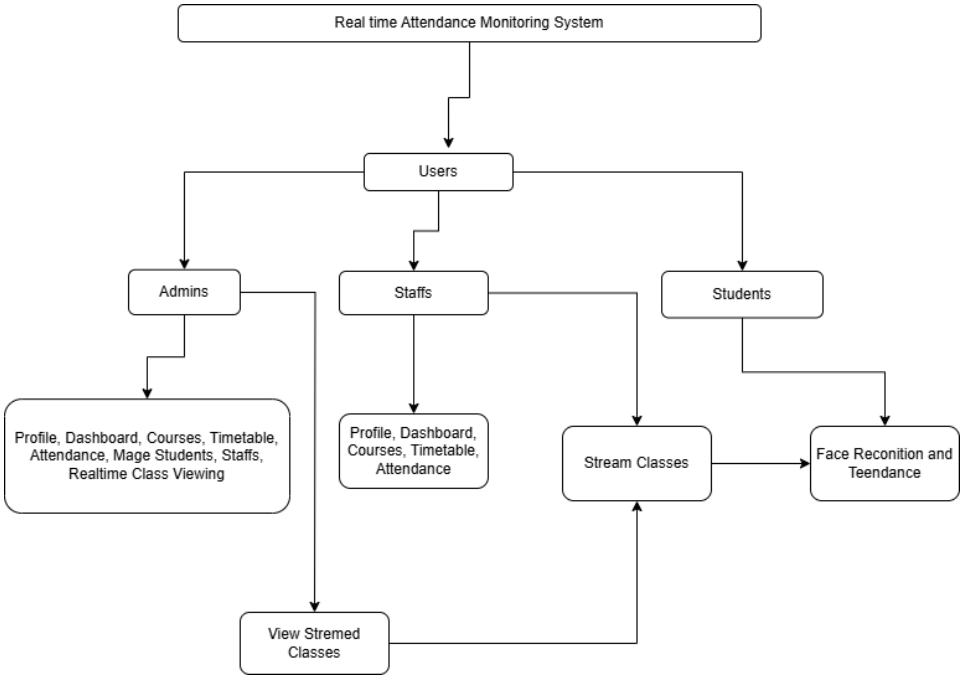

Figure 2: Block Diagram of the Student Attendance System

In a multi-user attendance system, real-time synchronization between devices is essential. Socket.IO, a JavaScript library that enables WebSocket-based communication, was used in this system to provide bidirectional, low-latency data exchange between the web application and classroom devices. This allows lecturers to initiate attendance, receive face recognition results in real-time, and allow administrators to monitor live class activity from remote devices. Socket.IO

supports fallback mechanisms like HTTP polling, making it resilient to poor network conditions (Zaidman et al., 2016).

Implementation

Hardware Implementation

The hardware implementation of the smart, real-time -based student attendance system as in figure 3 consists of a combination of key components designed to work together seamlessly. At the core of the system is a camera module which is responsible for capturing live facial images of students as they enter the classroom. This camera is strategically positioned to ensure clear visibility of each student's face, and it must support high-resolution imaging to improve the accuracy of facial detection. The captured images are sent to the Raspberry Pi. The Raspberry Pi interfaces with the camera, handles basic image acquisition tasks, and run lightweight face detection software to filter frames before forwarding them. This helps reduce the volume of data transmitted and speeds up the recognition process. The block diagram (figure 3) presents the connection of the physical part of the smart attendance management system.

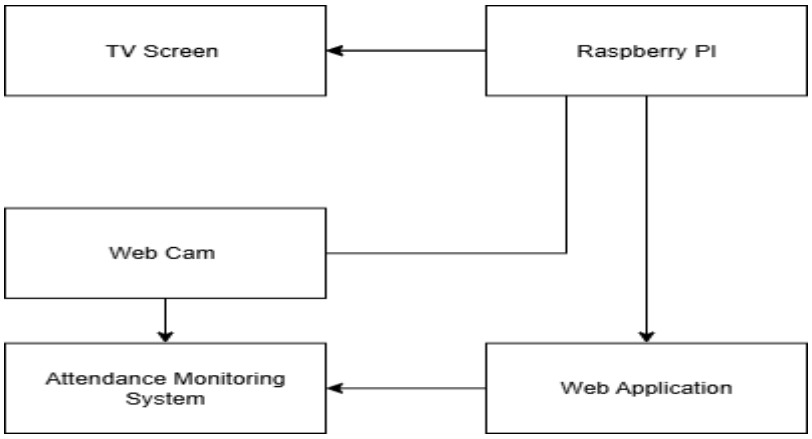

Figure 3: Block diagram of the connection of the physical part of the system

The Raspberry Pi connects to the internet through the Wi-Fi module, allowing it to transmit the facial data to a central server or cloud-hosted processing unit in real time. The server is equipped with deep learning models for advanced facial recognition, feature extraction, and matching. This offloading of intensive computation to the server enhances performance and reduces the computational load on the local device. The display screen is connected to the Raspberry Pi to provide real-time visual feedback, showing the detected faces or recognition results, which is useful for monitoring or debugging. The power supply unit ensures uninterrupted operation of the Raspberry Pi, particularly in classrooms with unstable power sources. The central server maintains the attendance database and hosts a web-based administrative dashboard, allowing authorized personnel to monitor attendance logs in real time. This integration between hardware components and cloud-based processing forms a robust, scalable, and efficient solution for automated attendance tracking. The figure 4 presents the physical part of the smart attendance management system.

Figure 4: Physical part of the system screen, raspberry Pi, webcam

Software Implementation

The software implementation of the smart, real-time student attendance system is centered on a modular architecture designed for scalability, performance, and secure data management. The system's frontend is developed using Next.js, a modern web framework known for its speed and static-site generation capabilities, which allows for a responsive and lightweight user interface. This enables administrators and students to interact seamlessly with attendance data, access live video streams, and view historical records through dynamic and server-rendered pages.

At the backend, the system leverages NestJS, a TypeScript-based server-side framework modeled after Angular's architecture. NestJS provides a highly testable and maintainable environment for developing RESTful APIs, handling authentication, processing attendance logs, and integrating facial recognition services. Its modular design allows separation of concerns across services such as user

authentication, attendance tracking, and media file handling. For data storage, MySQL is employed as the relational database management system. It efficiently stores structured data such as student profiles, facial embeddings, attendance logs, and role-based access controls. MySQL's robust indexing and relational mapping features support real-time querying and reporting across large datasets.

Media files, including facial images and video streams captured for attendance verification, are offloaded to Amazon S3 (Simple Storage Service). This cloud-based object storage service ensures high durability and accessibility, enabling the system to efficiently store and retrieve large amounts of biometric data and real-time footage without burdening the application server. The entire platform is deployed on a Virtual Private Server (VPS), which hosts the NestJS backend, the Next.js frontend, and the MySQL database. A VPS offers a cost-effective and scalable environment with root access, allowing full control over server configurations, system updates, and the deployment of security protocols such as HTTPS and firewalls. The figure 5 shows the application schema connection and database relationships.

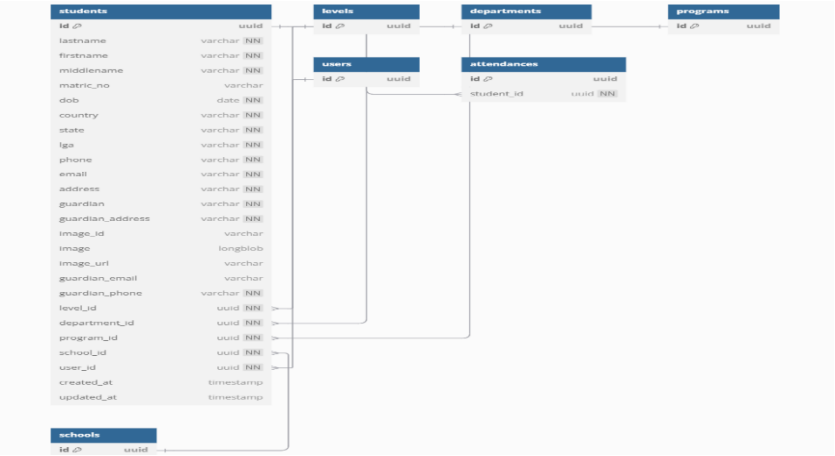

Figure 5: Application Schema, Connections and Relationship

During facial recognition, when a student attempts to check in, their image is captured via a browser interface, transmitted securely to the backend, and compared with stored embeddings in the database using deep learning algorithms. If a match is found, attendance is automatically marked and logged. This recognition process involves services like TensorFlow.js or ONNX Runtime, integrated via the backend for real-time prediction. The system supports role-based dashboards for administrators to monitor attendance metrics, view live surveillance feeds, and download reports. Notifications for absenteeism or system anomalies are sent via APIs integrated with email or SMS gateways, ensuring timely alerts and operational efficiency.

System Process

The smart, real-time student attendance system operates by integrating both hardware and software components to ensure efficient, automated attendance logging and live monitoring. The process begins with the initialization of hardware such as cameras and the activation of core software services, including Next.js for the frontend interface, NestJS for backend processing, MySQL for data storage, and Amazon S3 for file management. These components communicate securely through a Virtual Private Server (VPS), ensuring system stability and data security. Meanwhile, the step by step initialization process includes:

The following are pages of student registration form, lecturer and student dashboard, Student Select Course List, Lecturer Assigned Courses, Student Attendance, Student image upload, Lecturer view student list, and Admin Dashboard.

Lecturer can view the list of student on their dashboard after student have successfully marked their attendance as seen in figure 6.

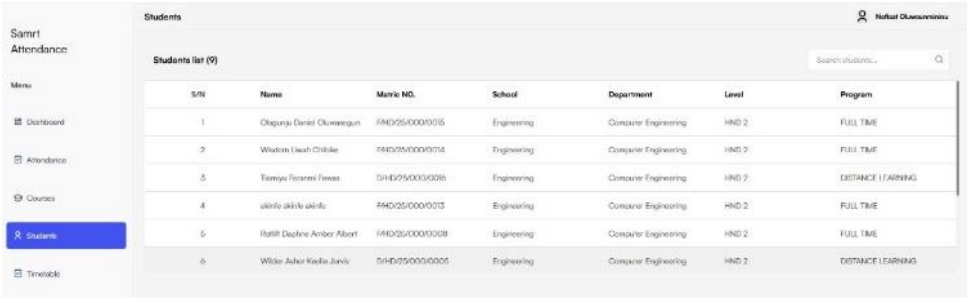

Figure 6: Lecturer View of Marked Student Attendance

The admin permission also gives access to creating, adding and deleting courses, can view all attendance irrespective of courses and view all registered staff and students in school as presented in figure 7.

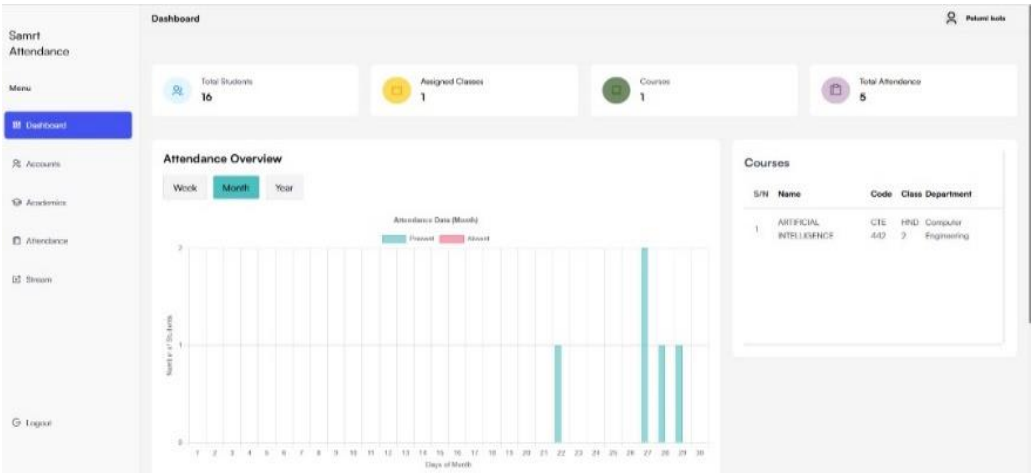

Figure 7: Admin View of All Attendance

Face Recognition

The smart attendance system leverages face-api.js, a JavaScript library built on top of TensorFlow.js, to perform real-time facial recognition directly within the browser. This system was designed to solve the challenges associated with manual attendance recording in academic institutions, specifically targeting accuracy, speed, and automation. The facial recognition feature is one of the core modules of the system and operates entirely on the frontend of the application, which makes it lightweight, fast, and easy to deploy without any dependency on server-side machine learning frameworks.

In this research, face-api.js was used to identify and verify student identities from a live webcam stream. The attendance session begins when a lecturer starts a class on the application, and at that point, the system initiates a process that fetches all registered students offering the course from the backend database. Each of these student records includes pre-captured facial data which was obtained during the student's onboarding process. This facial data is stored in the form of face descriptors numerical encodings that represent the unique features of a person's face. Once fetched, these descriptors are loaded into memory on the frontend, making them readily available for comparison.

There were some challenges encountered during the implementation, especially in ensuring consistent lighting and camera quality, which are crucial for accurate face detection and recognition. Students were advised to face the camera directly and ensure their faces were well-lit for optimal results. Additionally, the system had to handle multiple students appearing in the video frame at once. To manage this, face-api.js processed faces sequentially, prioritizing the closest or most clearly visible face, and identifying others as resources allowed as in figure 8.

.

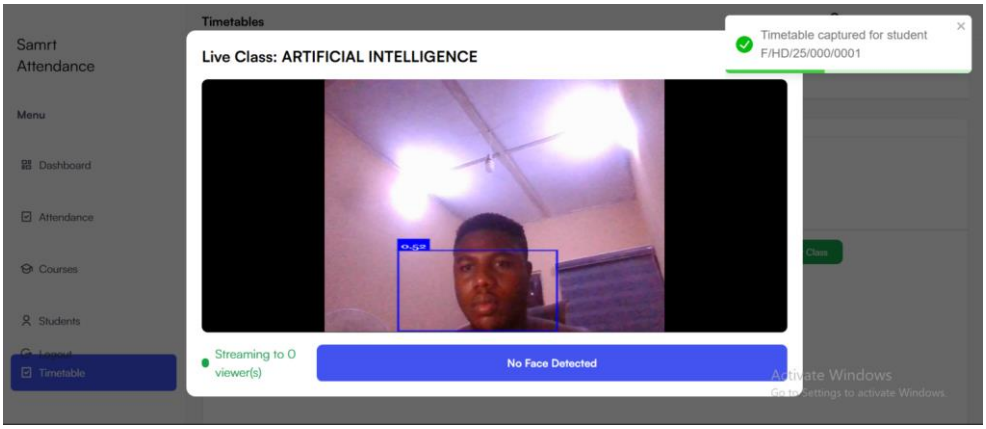

Figure 8: Face Api Detecting face in our application

Face-api.js served as the facial recognition engine at the heart of the smart attendance system. By combining real-time video capture with pre-stored facial descriptors and running the comparison entirely on the frontend, this research work was able to build a reliable, fast, and secure attendance solution. The system demonstrates how modern browser-based machine learning tools can be applied to solve real-world problems in education without the need for complex infrastructure or expensive hardware. It not only streamlined the attendance process but also showcased the practical power of client-side AI technologies.

Deployment and Hosting Architecture of the Smart Attendance System

To ensure reliability, performance, and scalability of the smart attendance system, the deployment was done using a multi-platform cloud infrastructure. The frontend was deployed on Vercel, the backend on Render, and the database was hosted on Aiven.io using MySQL. This setup separates concerns effectively and takes advantage of the strengths of each platform. Table 1 presents deployment structure of the system.

Table 1: Deployment Architecture

Component	Platform	Technology Used	Purpose
Frontend	Vercel	Next.js (React)	Client interface for students, lecturers
Backend	Render	Node.js (NestJS/Express)	API endpoints, authentication, logic
Database	Aiven.io	MySQL	Persistent relational data store

RESULTS AND DISCUSSION

The testing phase of the deep learning-based student attendance system revealed a remarkably high face detection accuracy of 99%, particularly when students maintain direct eye contact with the camera. This performance underscores the effectiveness of the system under ideal conditions and aligns with the observations of Nguyen and Kim (2022), who reported that convolutional neural networks (CNNs) demonstrate optimal recognition capabilities when users present a frontal facial pose. Despite the system's high accuracy, its performance deteriorates when students deviate from a direct camera gaze. In such instances, face detection accuracy drops considerably, resulting in failed attendance registration. This limitation corroborates findings from Hussain et al. (2023), who highlighted that off-angle facial positioning impairs the extraction of identifiable facial features, thereby diminishing recognition accuracy.

These outcomes highlight the importance of user compliance in ensuring the system's reliability. Specifically, students are required to face the camera directly to achieve successful and consistent attendance logging. The table 2 presents the performance evaluation to the face recognition attendance system.

Table 2: Performance Evaluation of the Face Recognition Attendance System

Condition	Face Detection Accuracy	Impact on Attendance	Author
Frontal face (looking into camera)	99%	Successful attendance registration	Nguyen & Kim (2022)
Off-angle face (looking away)	Significantly reduced	Failed or inaccurate registration	Hussain et al. (2023)

Comparative Analysis: Face-Api.js Vs. Other Face Recognition Technologies

When building a smart attendance system, selecting the right facial recognition technology is crucial for performance, scalability, data privacy, and ease of integration. In this research work, where the system runs entirely in the browser, recognizes students from a video stream, and matches them to pre-enrolled facial data, face-api.js presents itself as a highly suitable choice. However, to appreciate its strengths and limitations, it is essential to compare it with other notable face recognition frameworks such as OpenCV with Dlib, DeepFace, and Amazon Rekognition.

Environment and Integration Approach

Face-api.js runs entirely in the browser, leveraging Web APIs like getUserMedia() for video streaming and TensorFlow.js under the hood for inference. This in-browser architecture makes it distinct from other technologies. It does not require a backend for processing, which simplifies deployment and ensures that facial data never leaves the user's device—an excellent advantage in academic settings where student privacy is paramount.

In contrast, OpenCV with Dlib and DeepFace are typically deployed in backend environments. They are written in C++ and Python respectively, and rely on server-side processing. This setup offers greater computational power but introduces latency, server costs, and potential privacy concerns.

Amazon Rekognition, a commercial cloud-based solution, exposes facial recognition as an API. While it's highly scalable and extremely fast due to AWS's GPU infrastructure, it comes with cost and privacy trade-offs. Images must be sent over the internet to AWS servers, which may not be acceptable for institutions dealing with sensitive biometric data like student faces.

Performance and Accuracy

From a performance standpoint, face-api.js is surprisingly capable for real-time face detection and recognition in classroom scenarios. It provides various pre-trained models like SSD MobileNet and TinyFaceDetector, which are optimized for speed on devices like laptops or tablets. While it may not match the absolute accuracy of server-side or cloud-hosted models, its precision is sufficient for recognizing students in typical indoor environments with controlled lighting. OpenCV with Dlib provides more flexibility and higher accuracy, especially when configured with GPU acceleration or CNN-based detectors. It's often used in academic and research environments where precise facial landmark detection is required. DeepFace, on the other hand, is a high-accuracy Python-based library that integrates multiple backend models such as VGG-Face, Facenet, and ArcFace. It can outperform face-api.js in terms of recognition accuracy, especially in datasets with varied lighting, occlusions, and expressions. Amazon Rekognition leverages deep learning models trained on vast datasets. It offers very high accuracy with features like emotion detection, face comparison, and even celebrity recognition. However, this level of performance is bundled with a price tag and dependency on cloud infrastructure.

Data Privacy and Offline Capability

One of the most critical differentiators is data privacy. face-api.js excels here by allowing the entire facial recognition pipeline to run in the browser. Student face descriptors (128D vectors) are stored securely in the system, and matching is done locally using Euclidean distance comparisons. There's no risk of student images being exposed to third parties or stored in the cloud. This approach aligns with privacy standards such as GDPR and FERPA, which are relevant in academic environments.

In contrast, OpenCV with Dlib and DeepFace can be deployed in a privacy-respecting way, but this depends on how the backend is configured. If the server is compromised or not secured, student data is at risk. Amazon Rekognition, despite its robustness, requires sending facial images over the internet, which introduces legal and ethical concerns—especially in education.

Ease of Use and Customization

From a development perspective, face-api.js is incredibly developer-friendly. It requires minimal setup just import it into a browser-based application and start detecting faces with

a few lines of code. Developers have full control over thresholds, detection intervals, and matching logic, which is particularly beneficial in a custom system like this research work.

In comparison, OpenCV with Dlib requires native builds and often complex setup steps, especially for real-time applications. DeepFace is relatively easier to use in Python environments but requires model downloads, backend setup, and knowledge of facial embeddings and verification thresholds. Amazon Rekognition is simple to integrate via SDKs but offers limited control over its internal algorithms, making it somewhat of a black box.

Cost and Scalability

face-api.js is open-source and free, which is perfect for academic institutions or developers working on budget-sensitive projects. Since it operates on the client-side, there are no server or cloud hosting costs. However, scalability is limited to the power of the user's browser and device.

OpenCV, Dlib, and DeepFace are also open-source, but they incur costs in the form of server infrastructure and possibly GPU hardware for large-scale deployments. Amazon Rekognition is a commercial service with a pay-as-you-go model. While it offers high scalability out of the box, the cost can escalate quickly with frequent usage, making it less viable for schools or universities operating on limited budgets.

Low-Light Performance

Interestingly, the system demonstrated a strong capacity to detect faces even under low-light conditions, indicating a high level of robustness and adaptability in suboptimal environments. This performance can be attributed to the diverse training dataset, which incorporated images captured under a wide range of illumination scenarios. As a result, the model effectively generalized to different lighting settings, enabling reliable face detection across typical classroom conditions. These findings are consistent with the work of Wang et al. (2022), who observed that deep learning-based facial recognition systems can maintain accuracy in low-light environments through the enhancement of image contrast and edge features. The use of convolutional neural networks (CNNs) equipped with illumination-invariant filters contributes significantly to this capability.

Although extremely dark conditions still pose challenges to detection, the system-maintained functionality under standard classroom lighting and during minor power outages, without significant degradation in performance. This suggests that deep learning models trained with illumination-variant datasets are well-suited for deployment in real-world educational settings, where lighting can often be inconsistent. The table 3 presents the performance evaluation under different lighting conditions.

Table 3: Performance under Different Lighting Conditions

Lighting Condition	Detection Accuracy	Performance Outcome	Source
Normal daylight / artificial lighting	High (Near 99%)	Accurate and consistent attendance registration	Proposed System
Low-light (dim classroom)	Moderately High	Successful face detection; minimal errors	Wang et al. (2022)
Power outage (partial darkness)	Acceptable	Minor reduction in accuracy, but system remained functional	Wang et al. (2022)
Extreme darkness	Low / Unreliable	Detection often fails; attendance cannot be marked reliably	-

Time Efficiency and Attendance Marking Speed

The developed smart attendance system significantly enhances efficiency by marking attendance in under 10 seconds, a remarkable improvement over traditional methods. Conventional manual approaches such as passing an attendance register around the classroom are often slow, disruptive, and prone to inefficiencies. These manual processes can lead to undesirable classroom behaviors, including disruptions, loss or damage of the register, and incomplete attendance records (Jain et al., 2004).

In contrast, the automated system utilizes facial recognition technology to uniquely identify and verify each student in real time. This ensures that only physically present students are able to mark their attendance, thus eliminating the possibility of proxy attendance—a common malpractice in educational institutions (Rahman et al., 2021). By automating the process, the system also minimizes human error, reduces administrative burden, and promotes a more transparent and secure attendance process.

Comparison with existing systems

The table 4 presents the comparison of the developed systems with existing systems. These systems were compared in terms of scalability, security and platforms supported.

Table 4: Comparison with existing systems

Source	Title	Scalability	Security	Platforms Supported	Comparison to Proposed
Kawaguchi & Shoji (2005)	Continuous Observation for Attendance	Low (manual CV)	Minimal	PC w/ camera only	Less scalable & secure, limited to legacy desktops.
Sharma & Srivastava (2019)	Smart Attendance with Face Recognition	Low	Basic (no auth layer)	Raspberry Pi	More cost-efficient, but this research work is more scalable and secure.
Suresh & Naresh (2020)	Deep Learning on Web Platform	High (CNN + Web)	High	Web	Similar approach; both support DL + web, comparable.
Praveen(2020)	ML vs DL Models	High	High (model-dependent)	Research/Sim only	This research work provide real-time deployment; this is academic benchmarking.

Source	Title	Scalability	Security	Platforms Supported	Comparison to Proposed
Ranaware et al.(2021)	RFID + Biometric + SMS	Moderate	Moderate–High	RFID scanners, Phones	Hardware-dependent, less scalable than this research work.
Bui & Nguyen(2021)	CNN + Haar Cascade Web App	Moderate	Moderate	Web	Similar, but less accurate than face-api.js or deep transfer learning.
Islam & Morsalin(2021)	Real-Time Face Recog. w/ RTSP	High (real-time)	Moderate	PC with Camera	Real-time performance match, this research work is more deployable.
Rao (2022)	AttenFace + Moodle	Medium	Moderate	Web + LMS	LMS integration is unique; this research work has more flexibility.
Seah(2023)	IoT Face Recog System	High	High	Web, GUI	Similar to ours, with great performance. Slightly better due to GUI integration.
Ennajar & Bouarifi(2024)	Transfer Learning for COVID Mask Detection	High	High	Web/Desktop	Focus on masked faces; similar in stack and approach.
Touzene et al.(2024)	Embedded DL Attendance	Medium	Moderate	Raspberry Pi	this research work is cloud-hosted, easier to scale.
Huda Al-Nayyef(2024)	Dual Face Recognition (HAAR+CNN)	Medium	Moderate–High	Desktop	Hybrid method; this research work have better UI integration and web support.
Schroff et al.(2015)	FaceNet	Very High	Very High	Research baseline	Foundation model;
Parkhi et al.(2015)	Deep Face Recognition	Very High	Very High	Academic baseline	Strong benchmark; this research work builds on such models for deployment.
Proposed (2025)	Modern Face Recognition Attendance	High (Cloud-ready)	High (Auth + HTTPS)	Web, Mobile, Desktop	Scalable, secure, flexible deployment, real-time use.

CONCLUSION

The development of a smart scalable student attendance system has significantly improved the accuracy, efficiency, and automation of attendance tracking in educational institutions. By leveraging deep learning for facial recognition, the system eliminates traditional challenges such as manual errors, unauthorized proxies, and time-consuming record-keeping (Jain et al., 2020). The integration of real-time processing ensures instant updates to attendance records, enhancing both transparency and institutional accountability (Rahman et al., 2021). However, the accuracy of the system is highly dependent on students positioning themselves correctly in front of the camera. Studies have shown that facial recognition models perform best when individuals look directly into the camera, minimizing misidentification and improving detection speed (Nguyen & Kim, 2022). Factors such as poor lighting, partial obstructions, and extreme facial angles can reduce accuracy, but with proper user compliance, the system remains highly effective. Future research will address challenges such as variations in facial expressions, changes in student appearance over time, and potential spoofing attacks exist. In addition to this, a mobile application will be developed to allow students to mark their attendance-this app will use IP location-based geofencing to ensure that students can only register attendance when physically present in the designated lecture hall, preventing remote check-ins as proposed by (Kumar & Sharma, 2021).

REFERENCES

Brown, T. (2020). Attendance management in higher education: Challenges and solutions. Educational Technology Publications.

Bui & Nguyen (2021) “Student Attendance System Using Face Recognition. Techs: Python, CNN, Haar Cascade, ORL dataset. https://doi.org/10.1007/978-981-33-6307-6_98

Ennajar, S., & Bouarifi, W. (2024). Deep Transfer Learning Approach for Student Attendance System During the COVID-19 Pandemic. *Journal of Computer Science, 20*(3), 229–238. https://doi.org/10.3844/jcssp.2024.229.238

Hussain, S., Nefti-Meziani, S., & Atyabi, A. (2023). Pose-invariant face recognition: Challenges and solutions. IEEE Transactions on Biometrics, Behavior, and Identity Science, 5(1), 1-15. https://doi.org/10.1109/TBIOM.2023.3245678.

Huda Al-Nayyef (2024) Iraq introduced dual-method attendance using HAAR+LBPH and HOG+CNN, addressing image preprocessing challenges for low-quality data. Techs: Haar cascade, LBPH, HOG, CNN, image filters. http://dx.doi.org/10.3390/engproc2025084039

Islam, M. & Morsalin, S. (2021). Real-Time Face Recognition Based Smart Attendance System. 2021 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST). https://doi.org/10.1109/ICREST51555.2021.9331149.

Jain, A. K., Ross, A., & Prabhakar, S. (2004). An introduction to biometric recognition. IEEE Transactions on Circuits and Systems for Video Technology, 14(1), 4-20. https://doi.org/10.1109/TCSVT.2003.818349

Johnson, L. (2022). Biometric systems in education: Trends and future directions. Journal of Educational Technology

Systems, 50(3), 345-360. https://doi.org/10.1177/00472395221093645.

Kawaguchi, Y & Shoji, T. (2005) "Face Recognition-based Lecture Attendance System", used continuous observation to enhance face-detection accuracy in lecture halls. Techs: continuous observation, face recognition. https://www.researchgate.net/publication/241608617_Face_Recognition-based_Lecture_Attendance_System

Kumar, P., & Sharma, R. (2021). Geofencing for attendance systems: A mobile-based approach. International Journal of Mobile Computing, 12(2), 78-92. https://doi.org/10.1016/j.ijmc.2021.04.003

Nguyen, T., & Kim, H. (2022). Deep learning for facial recognition: Advances and challenges. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(5), 2001-2018. https://doi.org/10.1109/TPAMI.2022.3167890

Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep Face Recognition. Proceedings of the British Machine Vision Conference (BMVC). https://doi.org/10.5244/C.29.41

Praveen, K., Mamata, G., Sujata C., Bijay K., Rakesh S. and Suneeta S. (2020) "Student Attendance System Based on Face Recognition and Machine Learning", implemented SVM, Decision Tree, CNN, VGG19, and ResNet50, achieving 96–97% accuracy; stored output in Excel. Techs: SVM, Decision Tree, CNN, VGG19, ResNet50. https://ceur-ws.org/Vol-3283/Paper48.pdf

Rao, A. (2022) AttenFace: A Real-Time Attendance System Using Face Recognition. In Proceedings of the 2022 IEEE 6th Conference on Information and Communication Technology (CICT) . https://doi.org/10.1109/CICT56698.2022.9998001

Rahman, M. M., Islam, M. S., & Hossain, M. A. (2021). Automated attendance systems in education: A review. Journal of Information Technology Education: Research, 20, 123-145. https://doi.org/10.28945/4765.

Ranaware, A. (2021). Smart Attendance System. *International Research Journal of Engineering and Technology (IRJET)*, 8(5), 3177–3180. https://www.irjet.net/archives/V9/i3/IRJET-V9I3177.pdf.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR.2015.7298682

Seah, J. (2023). Smart attendance system with IoT-based face recognition using deep learning approach. Journal of Artificial Intelligence and Technology, 3(2), 45-60. https://doi.org/10.1016/j.jait.2023.02.003

Sharma, R., & Srivastava, S. (2019). *Smart Attendance System Using Face Recognition*. 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), 1–4. https://doi.org/10.1109/IoT-SIU.2019.8777560

Suresh, A., & Naresh, G. (2020). A Deep Learning Model for Student Attendance System Using Facial Recognition. *Procedia Computer Science*, 167, 2544–2553. https://doi.org/10.1016/j.procs.2020.03.312

Touzene A., Abed Abdeljalil W., Slimane L.(2024) "An Embedded Intelligent System for Attendance Monitoring" integrates Raspberry Pi, Pi-Cam, and a web app; addresses resource constraints on embedded devices. Techs: Raspberry Pi, Pi-Cam, embedded DL, web interface. http://dx.doi.org/10.48550/arXiv.2406.13694

Wang, Y., Zhang, X., & Li, J. (2022). Illumination-invariant face recognition: A deep learning approach. IEEE Transactions on Image Processing, 31, 1234-1245. https://doi.org/10.1109/TIP.2022.3145678

Zaidman, A., Figueiredo, E., & Gross, H. G. (2016). Understanding WebSocket-based applications. IEEE Software, 33(3), 82–87. https://doi.org/10.1109/MS.2016.63

©2025 This is an Open Access article distributed under the terms of the Creative Commons Attribution 4.0 International license viewed via https://creativecommons.org/licenses/by/4.0/ which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is cited appropriately.