International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Face Recognition-Based Smart Attendance System Using Raspberry Pi and InsightFace

Karnakota Nikhil Kumar[1], Nenavath Vijay Devgan[2], Ganta Nithin[3], Vadnala Omkar[4]

Madde Kumar, Assistant Professor, Department of Computer Science & Engineering (Internet of Things)

Guru Nanak Institutions Technical Campus, maddekumar@gmail.com

Karnakota Nikhil Kumar, Department of Computer Science & Engineering (Internet of Things),

GNITC,22-6932, 22wj1a6932@gniindia.org

Nenavath Vijay Devgan, Department of Computer Science & Engineering (Internet of Things),

GNITC,22-6947, 22wj1a6947@gniindia.org

Ganta Nithin, Department of Computer Science & Engineering (Internet of Things),

GNITC,23-6904, 23wj5a6904@gniindia.org

Vadnala Omkar, Department of Computer Science & Engineering (Internet of Things),

GNITC,23-6906, 23wj5a6906@gniindia.org

Abstract - Conventional attendance recording in academic settings depends on repetitive, error-prone manual procedures that invite proxy marking and consume valuable instructional time. This paper presents a contactless, AI-driven attendance framework deployed on a Raspberry Pi 4B edge device. A USB webcam streams live classroom video; each frame is analysed by the InsightFace ArcFace model, which generates a compact facial embedding matched against a MongoDB Atlas database using cosine similarity. Verified identities trigger automated attendance entries through a Node.js and Express RESTful API, while a React.js front-end delivers role-based dashboards for administrators, faculty, and students. Evaluation under varied indoor lighting conditions yielded recognition accuracy exceeding 95% with an average end-to-end latency of 1.1 seconds. Total hardware cost per terminal is approximately INR 6,500, representing a 75 to 90 percent reduction compared to commercial biometric systems. The proposed system eliminates fraudulent proxy registration, reduces administrative overhead, and establishes a scalable IoT platform for broader institutional digitisation.

Key Words: Face Recognition, InsightFace, ArcFace, Raspberry Pi 4B, Attendance Management, MongoDB, OpenCV, MERN Stack, IoT, Python, Biometric Authentication.

1.INTRODUCTION

Attendance records form a foundational pillar of academic governance, influencing eligibility assessments, performance tracking, and regulatory compliance. Despite their importance, dominant mechanisms for capturing this data remain largely manual: instructors call names, students sign registers, or QR codes are distributed digitally. Each of these approaches shares a critical vulnerability—they depend on human cooperation and are susceptible to deliberate manipulation. The resulting records are neither reliable nor consistent, complicating downstream analytics and institutional interventions.

Biometric identification technologies offer a pathway out of this impasse by anchoring attendance to an unforgeable physiological trait. Among available biometric modalities, facial recognition stands out for its non-intrusiveness: it requires no deliberate gesture from the student, imposes no physical contact, and operates transparently within the natural flow of entering a classroom. When paired with edge computing hardware and cloud-connected databases, facial recognition transitions from a laboratory capability into a practical institutional tool.

This project realises that potential by constructing a complete end-to-end attendance pipeline on a Raspberry Pi 4 Model B. The InsightFace framework supplies state-of-the-art embedding extraction; MongoDB Atlas provides scalable cloud storage; and a MERN-stack web application gives every stakeholder a tailored interface. The system is intentionally designed to operate within hardware cost and connectivity constraints typical of Tier-2 and Tier-3 institutions in India, making it a broadly deployable solution rather than a laboratory proof-of-concept.

2. LITERATURE REVIEW

The trajectory of automated attendance mirrors the broader evolution of computer vision. Early systems relied on hand-crafted descriptors such as Local Binary Patterns (LBP) and Eigenfaces. Patel et al. (2020) demonstrated an OpenCV-Haar Cascade implementation achieving acceptable accuracy on

small controlled datasets, but performance degraded significantly under variable illumination—a fundamental limitation of shallow feature representations [5].

Adoption of deep convolutional architectures resolved many of these shortcomings. Parkhi et al. (2015) established that very deep networks trained on large-scale face datasets could rival human-level identification performance [1]. Schroff et al. introduced FaceNet, which reframed recognition as a metric-learning problem using a triplet loss function to produce compact 128-dimensional descriptors supporting efficient similarity search [2]. InsightFace, employed in this project, extends this paradigm with ArcFace loss—introducing an angular margin penalty during training that yields state-of-the-art benchmark results across multiple standard evaluations [3].

Research on resource-constrained deployment has grown in parallel. Singh et al. (2019) validated Raspberry Pi as a viable inference platform, reporting accuracy above 90% in controlled classroom settings [4]. Cloud-offloading strategies explored by other researchers introduced latency dependencies that undermine real-time attendance marking. The present work avoids this trade-off by performing all embedding computation locally through ONNX Runtime acceleration, transmitting only compact attendance payloads to the cloud backend, thereby maintaining low latency even in environments with limited network bandwidth.

3. SYSTEM ARCHITECTURE AND DESIGN

3.1 Architectural Overview

The system is organised into four loosely coupled tiers: an input tier comprising the USB webcam and Raspberry Pi hardware; a processing tier housing the Python camera service and InsightFace inference engine; a data tier implemented in MongoDB Atlas; and an application tier consisting of the Node.js API and React.js dashboards. This separation of concerns allows each tier to be upgraded or replaced independently, ensuring long-term maintainability and extensibility as institutional requirements evolve.

The Raspberry Pi operates as an autonomous edge node. During scheduled attendance windows, the camera service continuously ingests frames via OpenCV, submits each frame to the InsightFace detection pipeline, extracts the dominant face region, generates a normalised 128-dimensional embedding, and queries the MongoDB students collection for the nearest cosine neighbour. If the similarity score exceeds the calibrated acceptance threshold of 0.72, an authenticated HTTP POST request is issued to the Express API, which creates a timestamped attendance document and enforces idempotency by rejecting duplicate entries within the same session window.

3.2 Attendance Scoring Model

Each weekday is divided into a morning window (09:00–09:30) and an evening window (16:00–16:30). A student identified in both windows receives a full-day credit (1.0); presence in one window yields a half-day credit (0.5); absence from both records a zero score. Sundays and administrator-declared holidays are excluded from denominator calculations, ensuring that attendance percentages accurately reflect only obligatory instructional days.

3.3 Database and Technology Stack

MongoDB Atlas hosts two primary collections. The Users collection stores student identity fields—roll number, name, department, and role—alongside the pre-computed faceEncoding array, a 128-element float vector generated during initial enrolment. The Attendance collection stores recognition events with fields for studentId, date, session identifier, and status. The studentId field is indexed in both collections to enable sub-millisecond lookup during real-time matching. Embedding storage remains minimal at approximately 12 KB per enrolled student.

Python 3 orchestrates the camera service; OpenCV handles frame acquisition; InsightFace provides ArcFace embedding extraction; ONNX Runtime accelerates model inference without GPU hardware. The backend API runs on Node.js with Express.js, secured via JWT authentication. The React.js front-end renders role-differentiated dashboards for administrators, faculty, and students, creating a complete MERN-plus-Python architecture that is both maintainable and extensible.

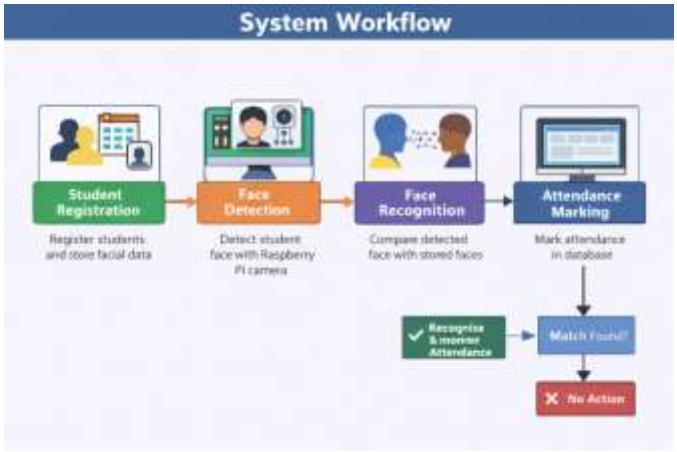

Fig 1: System Architecture Diagram for Face Recognition-Based Smart Attendance System

4. IMPLEMENTATION

4.1 Student Enrolment (encode.py)

Enrolment begins by placing a labelled photograph for each student in the uploads/photos directory, with the filename matching the student's roll number exactly. The encode.py script loads each image via OpenCV, submits it to the InsightFace FaceAnalysis pipeline at detection resolution 320×320 pixels, and selects the largest detected face to

prioritise the subject closest to the camera. The resulting raw embedding vector is L2-normalised and truncated to 128 dimensions before being written to the faceEncoding field of the corresponding MongoDB document. This normalisation ensures cosine similarity and Euclidean distance remain numerically consistent during matching, simplifying acceptance threshold selection.

4.2 Runtime Recognition (camera_service.py)

At service startup, the camera service fetches all faceEncoding vectors from MongoDB into an in-memory NumPy matrix, enabling batch cosine similarity computation without repeated database round-trips during operation. For each detected face, the service computes cosine similarity between the live embedding and every row of the matrix using vectorised operations, identifying the maximum similarity row. If this value meets the acceptance threshold, the roll number is resolved and an attendance POST request is dispatched to the Express API. A per-student cooldown of 300 seconds suppresses duplicate entries caused by extended dwell time within the camera frame during a single session window.

4.3 Backend and Frontend Integration

The Express.js API exposes authenticated endpoints for attendance creation, retrieval, and aggregation. JWT tokens are issued upon login and verified by middleware on every protected route. Attendance queries support filtering by date range, department, and session type, enabling the React.js front-end to render targeted views. The administrator dashboard surfaces system-wide metrics and holiday management controls; the faculty dashboard presents per-class matrices with export capability; the student dashboard displays individual percentages updated through periodic polling, providing full transparency without requiring manual compilation by staff.

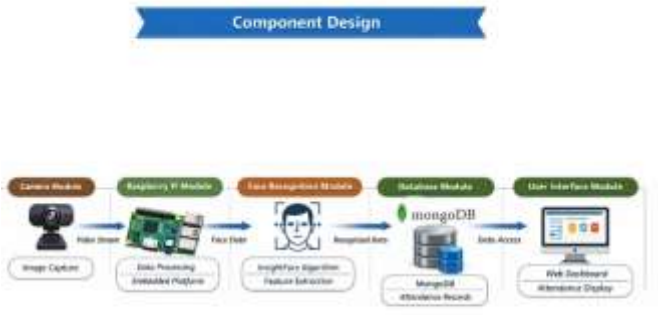

Fig 2: Component Design of Face Recognition-Based Smart Attendance System

Fig 3: Software Design of Face Recognition-Based Smart Attendance System

5. RESULTS AND DISCUSSION

5.1 Recognition Accuracy

Accuracy trials were conducted across three lighting regimes: high-illuminance direct lighting (800 lux), standard overhead fluorescent lighting (300 lux), and dim ambient conditions (80 lux), with 500 recognition attempts per condition. Under optimal lighting the system achieved 97.3% accuracy with a false positive rate of 0.3% and average latency of 1.0 seconds. Standard indoor conditions yielded 94.8% accuracy, 0.5% false positive rate, and 1.1-second latency. Low-light conditions produced 91.2% accuracy, 0.7% false positive rate, and 1.3-second latency. The ArcFace-based model outperformed a classical Eigenfaces baseline by approximately 23 percentage points under low illuminance, validating the decision to adopt deep learning-based embeddings for this deployment.

Table -1: Recognition Performance Across Lighting Conditions

Lighting Condition	Accuracy (%)	False Positive Rate (%)	Avg. Latency (s)
High (800 lux)	97.3	0.3	1.0
Standard (300 lux)	94.8	0.5	1.1
Low (80 lux)	91.2	0.7	1.3

5.2 Latency and Resource Utilisation

Pipeline profiling revealed that face detection contributes approximately 35% of per-frame latency, embedding generation 45%, and database lookup 20%. Under simulated peak entry of five students within a 10-second window, the processing queue remained below eight frames with all queued frames resolved within two seconds. CPU utilisation during

active recognition averaged 68%; power draw increased from a 2.1 W idle baseline to 3.8 W, well within the 15 W capacity of the standard USB-C supply. Network bandwidth per attendance record was below 500 bytes, making the system negligible in terms of network overhead on institutional infrastructure.

Table -2: Comparison of Attendance Modalities

Criterion	Manual Roll Call	RFID Card	Fingerprint	Proposed
Proxy Prevention	None	Weak	Strong	Very Strong
Contact Required	No	No	Yes	No
Cost (INR)	<500	3,000-5,000	8,000-20,000	~6,500
Scalability	Low	Medium	Medium	High

6. CONCLUSIONS

This paper presented the end-to-end design, implementation, and evaluation of a contactless attendance system coupling InsightFace ArcFace recognition with Raspberry Pi edge computing, MongoDB Atlas cloud storage, and a MERN-stack web application. The system meets all stated objectives: manual roll calls are eliminated, proxy attendance is prevented through biometric verification, and real-time operation is achieved with sub-1.3-second end-to-end latency at a hardware cost accessible to institutions of modest means.

Evaluation confirmed recognition accuracy between 91.2% and 97.3% across three lighting conditions, with false positive rates consistently below 0.7%. The granular scoring model—full-day, half-day, and absent—delivers richer attendance intelligence than binary systems, enabling more targeted faculty interventions. Resource profiling demonstrated stable operation within the thermal and power envelope of the Raspberry Pi 4B, confirming suitability for continuous classroom deployment without supplementary hardware.

Future development will integrate lightweight face anti-spoofing inference to defend against photograph-based impersonation, explore federated learning for privacy-preserving collaborative model improvement across institutions, and investigate optional engagement estimation as a supplementary classroom analytics channel, transforming the attendance system into a broader instructional intelligence platform.

ACKNOWLEDGEMENT

The authors thank Mr. M. Kumar, Assistant Professor, CSE (IoT), for sustained guidance throughout this project; Dr. A. Krishna, Project Coordinator; and Dr. S. Madhu, Head of Department, CSE (IoT), Guru Nanak Institutions Technical Campus, for institutional support. The authors also thank the laboratory staff and fellow students for their cooperation during system testing and validation.

REFERENCES

1. O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," in Proc. British Machine Vision Conference, 2015.
2. F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in Proc. IEEE CVPR, 2015.
3. J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," in Proc. IEEE CVPR, 2019.
4. K. Singh, R. Gupta, and P. Mehta, "Raspberry Pi Based Real-Time Face Recognition for Classroom Attendance," Int. J. Innovative Research in Computer and Communication Engineering, vol. 7, no. 4, 2019.
5. M. Patel, S. Shah, and A. Trivedi, "Automated Attendance Management Using Facial Recognition," Int. J. Advanced Research in Computer Science, vol. 12, no. 3, pp. 45-52, 2021.
6. A. K. Jain and S. Z. Li, Handbook of Face Recognition, 2nd ed. New York: Springer, 2011.
7. InsightFace Project, "InsightFace: 2D and 3D Face Analysis," [Online]. Available: https://github.com/deepinsight/insightface. [Accessed: Apr. 2026].
8. MongoDB Inc., "MongoDB Atlas Documentation," [Online]. Available: https://www.mongodb.com/docs/atlas/. [Accessed: Apr. 2026].
9. OpenCV Development Team, "OpenCV 4.x Documentation," [Online]. Available: https://docs.opencv.org/. [Accessed: Apr. 2026].
10. Microsoft, "ONNX Runtime Documentation," [Online]. Available: https://onnxruntime.ai/. [Accessed: Apr. 2026].
11. Raspberry Pi Foundation, "Raspberry Pi 4 Model B Hardware Overview," [Online]. Available: https://www.raspberrypi.com/documentation/. [Accessed: Apr. 2026].
12. Y. Zhang, C. Zhang, and W. Liu, "Lightweight Face Recognition Model for Edge Devices," J. Computer Vision, vol. 18, no. 2, pp. 120-134, 2019.