JOURNAL OF MODERN AND VERSATILE RESEARCH APPROACH (JMVRA)
© 2026 JMVRA | Volume 4, Issue 10, October 2026 | ISSN: 2984-8911 | JMVRA.ORG

AI-POWERED REAL-TIME FACE RECOGNITION ATTENDANCE SYSTEM

With Automated Excel-Compatible Report Generation

[1]Abhilash Telkar Shivkumar, [2]Akshay Narayan Naik, [3]Dhanush Kumar M R, [4]Gagandeep K L
[1,2,3,4]Undergraduate Student
[1,2,3,4]Department of Artificial Intelligence and Machine Learning,
[1,2,3,4]Moodlakatte Institute of Technology Kundapura (VTU Belagavi), Udupi, Karnataka, India
Corresponding author: Abhilash Telkar Shivkumar, Email: [ADD EMAIL]

Abstract : Background: Manual classroom attendance is slow, interrupts teaching and is open to proxy marking. Methods: This paper presents an AI-powered, real-time face-recognition attendance system. A browser sends camera frames to a FastAPI backend, which forwards them to an isolated machine-learning microservice. SCRFD detects faces, five-point alignment is applied, and ArcFace (w600k_r50) produces 512-dimensional L2-normalised embeddings. The backend matches these against an enrolled gallery in PostgreSQL with pgvector using cosine similarity, applies a threshold and quality check, and records attendance at most once per student per session. Access is secured with Argon2id, JWT and role-based access control, and a randomised head-movement challenge with proxy alerts adds a lightweight anti-spoofing layer. Results: The system passed 73 automated tests (43 ML, 30 backend), and gallery search takes about 5.9 ms for 1,500 vectors and 19.7 ms for 5,000 vectors. Recognition accuracy (FAR, FRR, Rank-1) is not yet measured because the available dataset has no identity labels. Conclusion: The system is a secure, modular and testable prototype; identity-labelled field calibration is the main next step.

Index Terms - Face recognition, ArcFace, SCRFD, pgvector, attendance automation, role-based access control.

I. INTRODUCTION

Attendance is a basic administrative requirement in academic institutions. It is used to monitor student participation, to meet minimum-attendance rules for examinations and to maintain institutional records. In most classrooms it is still taken by roll call or on paper registers, which consumes lecture time, interrupts teaching and is easy to falsify through proxy attendance, where one student answers for another. RFID and ID-card systems automate recording but can be shared between students, and fingerprint scanners need physical contact, create queues at the door and raise hygiene concerns in shared rooms. Face recognition is contactless and works with an ordinary camera, which makes it attractive for classrooms.

Recent advances in deep learning have made face recognition practical on commodity hardware. Detectors such as SCRFD [1] output face boxes together with five facial landmarks that allow faces to be aligned to a standard pose, and margin-based embedding networks such as ArcFace [2] convert an aligned face into a compact vector so that faces of the same person lie close together. Combined with a vector-capable database such as PostgreSQL with the pgvector extension, these components can be assembled into a modular and secure attendance system. Many prototype systems, however, combine detection, recognition and storage in a single script, lack proper authentication, do not clearly handle unknown faces, duplicate marking or biometric privacy, and report accuracy without identity-aware evaluation.

This work therefore aims to design and implement a secure, real-time face-recognition attendance system for classrooms. Its objectives are to (i) detect faces and five landmarks and align faces to a canonical pose, (ii) generate 512-D L2-normalised ArcFace embeddings and match them by cosine similarity, (iii) store embeddings in PostgreSQL with pgvector, (iv) manage attendance sessions (CREATED, ACTIVE, COMPLETED) and prevent duplicate attendance, (v) provide a React web interface with role-based access, (vi) protect biometric data and accounts, and (vii) validate the system with automated tests and benchmarks while reporting accuracy honestly. The scope is classroom attendance using a browser camera, covering

student registration with 3 to 5 face samples, live recognition, session management and CSV reports. Trained presentation-attack detection, multi-camera deployment and full threshold calibration with identity-labelled data are left as future work.

The rest of the paper is organised as follows. Section II reviews related work and the research gap, Section III gives the requirements and architecture, Section IV describes the implementation, Section V presents testing and validation, Section VI discusses the results, and Section VII concludes with limitations and future work.

II. Literature Review

A. Existing attendance approaches

Attendance approaches range from paper registers to deep-learning pipelines (Table 1). Manual and card-based methods are time-consuming, error-prone and vulnerable to proxy attendance when cards are shared. Fingerprint and other contact biometrics are more accurate and secure but require touching a sensor, with hygiene concerns, sensor maintenance costs and failures with dirty or wet fingers. Classical face-recognition prototypes use Haar Cascade detection with Local Binary Patterns Histograms (LBPH) and record attendance automatically, but are sensitive to lighting, appearance change, face angle and photograph spoofing. Deep-learning pipelines such as InsightFace [3] and FaceNet-style models compare face embeddings using similarity measures and stay accurate under varied lighting, angles and expressions, at the cost of more computation and careful dataset preparation.

Table 1: comparison of attendance approaches

Approach	Advantages	Disadvantages
Manual roll call	Traditional and widely used	Time-consuming; human error and proxy marking
Card-based (RFID / smart card)	Faster than manual methods	Extra hardware and card upkeep; cards can be shared
Fingerprint / contact biometrics	Better accuracy and security than manual	Needs touch (hygiene); sensor cost; fails with dirty or wet fingers
Classical face (Haar + LBPH)	Contactless; automatic recording	Sensitive to lighting, pose, appearance change and photo spoofing
Deep-learning (InsightFace, FaceNet-style)	High accuracy under varied lighting, angles and expressions	Needs compute, dataset preparation and suitable hardware

B. Related research

Recent work from 2021 to 2025 applies these methods to classroom attendance directly. Singh [4] built an eigenface-based system with OpenCV, Trivedi et al. [5] used the Viola-Jones detector for real-time marking, and Jyoti et al. [6] designed an automated smart attendance system based on facial detection and identification. A 2022 study combined a CNN with a Haar classifier and also tested face-mask images [7], and Rao [8] proposed AttenFace, which runs recognition for each class in parallel and can connect to platforms such as Moodle. Deep models are reported to give stronger results: an MTCNN-based system outperformed a Haar-cascade CNN method [9], a Siamese network reached 94.4% accuracy on a university student dataset [10], a CNN-ArcFace system reported 98% accuracy in a Nigerian university setting [11], and an end-to-end system using classroom surveillance cameras was described in 2025 [12]. Kamil [13] reported up to 100% accuracy with deep learning outperforming machine-learning methods, a smart-library system reported 100% accuracy for single faces and 95% for faces turned sideways [14], and Al-Nayyef [15] and Thalor and Gaikwad [16] also used deep learning for attendance marking.

Liveness detection has received less attention. Santoso et al. [17] combined liveness detection with deep learning and reported 87% recognition accuracy on 1,300 photographs, and a benchmark of Dlib, ArcFace, SFace and GhostFaceNet on about 13,000 classroom images found Dlib best in top-1 accuracy (87.93%) and speed [18]. Such figures come from each study's own dataset and protocol, so the best model depends on deployment conditions.

C. Research gap

Existing student attendance systems often (i) tightly couple ML models with business logic, making them hard to maintain, (ii) lack role-based access control over attendance records and biometric data, (iii) do not enforce database-level constraints against duplicate marking in a session, (iv) report accuracy without identity-aware evaluation, and (v) give limited attention to liveness. The proposed system addresses these by separating the frontend, backend and a stateless ML service with enforced communication boundaries; by using Argon2id [19], JWT [20] and RBAC without ever returning password hashes or raw embeddings; by enforcing one attendance record per student per session through a database unique constraint; by providing an evaluation and threshold-optimisation framework that refuses to fabricate metrics without identity-labelled data; and by adding a randomised head-movement challenge, stating plainly that it is not a trained presentation-attack detector.

III. METHODOLOGY

A. Functional requirements

The system must support a complete pipeline from image capture in the browser to attendance recorded within an active session. The browser captures JPEG frames through the getUserMedia API at a controlled interval (2.5 s by default), and registration requires three to five samples per student whose quality (blur, face size, pose) is checked before storage. SCRFD detects all visible faces and five landmarks (two eyes, nose tip, two mouth corners), and an affine transformation aligns each face to a reference pose. ArcFace converts each aligned face into a 512-dimensional L2-normalised embedding, and the model version is stored with every embedding. The backend compares the query embedding with the enrolled gallery using cosine similarity and the pgvector <=> operator, converts distance to a similarity score and compares it with a predefined threshold; if no identity meets it, the face is classified UNKNOWN.

Administrators create, start and stop sessions for specific classes, subjects and semesters, and attendance is recorded only while a session is ACTIVE and at most once per student. Each processed face yields one of six outcomes: **MATCHED** (identified), **ALREADY_MARKED** (already recorded in this session), **UNKNOWN** (not recognised), **LOW_QUALITY** (unsuitable image), **NO_FACE** (no face detected) and **NO_ACTIVE_SESSION**. The interface shows live results and a session roster, offers reports by date, class, semester, subject and session, and exports records as CSV. Secure login and RBAC with two roles are required: ADMIN manages students, enrolment, sessions, users and system metrics, while USER can scan faces and view sessions and records. A randomised head-movement challenge must be passed before attendance is recorded, and proxy-detection evidence and alerts are restricted to administrators.

B. Non-functional requirements

The system must apply strict thresholding that favours a low false-accept rate, provide near real-time inference and fast gallery search, keep the frontend, backend and ML service loosely coupled, hash passwords, use short-lived tokens, retain no raw images and sanitise API responses. Code must be modular and covered by automated tests, with configuration through .env and no hard-coded secrets, and the system must start with one-click scripts on Windows.

C. System architecture

The system follows a microservices-inspired architecture of four components (Table 2, Fig. 1). Communication is restricted to three paths: frontend to backend, backend to database, and backend to ML service. Direct frontend-to-ML communication and direct database access by the ML service are forbidden, which improves security, maintainability and separation of responsibilities.

Table 2: system components, ports and responsibilities

Component	Port	Responsibility
React 19 frontend (Vite)	5173	UI for Admin and User; REST only to the backend
FastAPI backend	8000	Auth, RBAC, sessions, attendance, matching, persistence
ML service (FastAPI)	8001	Stateless detection, alignment, embedding
PostgreSQL 16 + pgvector	5432	Users, students, embeddings, sessions, attendance

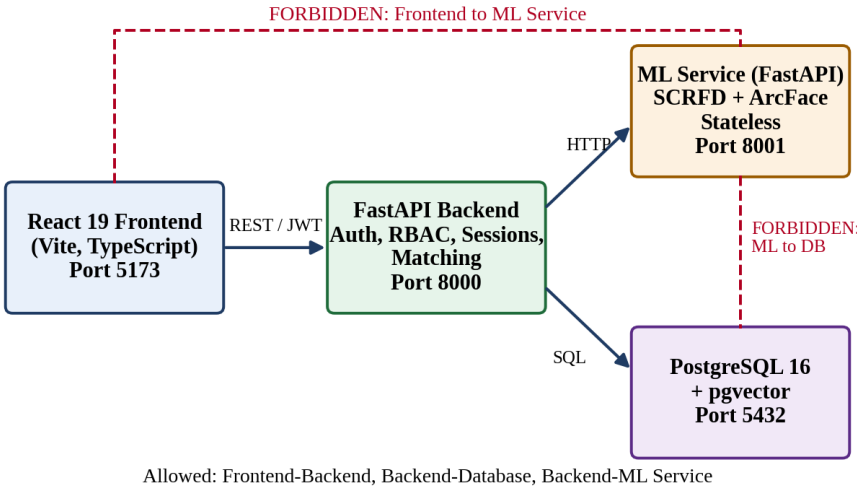


Fig. 1: system architecture and allowed communication paths

Ethical considerations: face data are used only for enrolment and attendance. Raw images are not retained after processing; only irreversible 512-dimensional embeddings are stored, access is restricted by RBAC, and proxy evidence is visible only to administrators. Enrolment must follow the informed consent of each student [ADD CONSENT PROCEDURE].

IV. Implementation

A. Registration and processing layer

During registration the administrator enters the student's name, register number, class and semester, and the system captures three to five images from different angles. Each image undergoes quality validation, SCRFD detection and landmark alignment, and ArcFace (w600k_r50, run through InsightFace and ONNX Runtime) produces an L2-normalised 512-D embedding that is stored in PostgreSQL with the student ID, model version and quality score. Unit tests verify that embeddings are normalised, finite and deterministic. The quality module checks blur, image size, pose and lighting, at about 0.19 ms per face; low-quality images are rejected during registration, and unsuitable faces receive the LOW_QUALITY status during attendance.

B. Live recognition and gallery matching

For live attendance the browser sends a frame to the backend endpoint /api/attendance/recognize, which forwards it to the ML service (/api/v1/embed). The backend then matches each embedding against the gallery with pgvector cosine distance (Fig. 2). Because embeddings are L2-normalised, cosine similarity reduces to a dot product:

$$s(q, g) = (q \cdot g) / (\|q\| \|g\|) = q \cdot g, \text{ since } \|q\| = \|g\| = 1 \qquad (1)$$

The student is identified only when the best similarity satisfies the configured threshold; otherwise the face is UNKNOWN, is never enrolled automatically and cannot create an attendance record. The backend then checks that a session is active and that attendance is not already recorded, and saves the record transactionally with student ID, session ID, timestamp and similarity score.

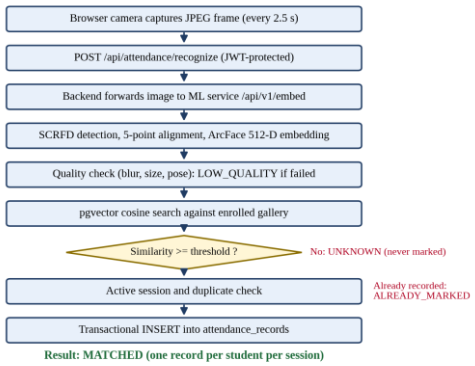


Fig. 2: operational flow of live attendance recognition

C. Session engine and duplicate prevention

Sessions follow a strict state machine (CREATED, ACTIVE, COMPLETED), and attendance can be recorded only in the ACTIVE state. Duplicates are blocked at three levels: frontend state validation, a backend query for existing records, and a database unique constraint on (session_id, student_id). If two concurrent requests try to mark the same student, the database rejects the second, and the backend catches the IntegrityError and returns ALREADY_MARKED with HTTP 200. The database holds the tables users, students, face_embeddings, sessions and attendance_records.

D. Liveness challenge and proxy detection

A recognised student must complete a randomised head-movement challenge before attendance is recorded. The implementation uses changes in the face detector's bounding box as a lightweight motion signal; it is not a trained presentation-attack detector and cannot reliably prevent all photograph or video-replay attacks. The challenge state is kept in backend process memory, so a single worker or sticky routing is required until shared storage is added. Proxy evidence and security alerts are visible only to administrators.

E. Authentication, RBAC and reporting

Passwords are hashed with Argon2id (about 51.6 ms per hash), and JSON Web Tokens use HS256 with 30-minute access tokens and 24-hour refresh tokens; a logout endpoint ends the session. Requests from USER accounts to administrator-only routes return HTTP 403. An authenticated reports API exports attendance as UTF-8 CSV, which opens directly in a spreadsheet, and an authenticated WebSocket channel broadcasts recognition outcomes and proxy alerts without transmitting raw video.

F. Technology stack and deployment

The backend uses Python, FastAPI, SQLAlchemy (asynchronous driver), Alembic migrations, Pydantic settings, Argon2id and JWT [21]. The ML service uses InsightFace (SCRFD and ArcFace), ONNX Runtime and NumPy, and an evaluation framework (evaluator, metrics, trials, threshold optimiser) supports future calibration. The frontend uses React 19, TypeScript, Vite and Tailwind CSS with WebRTC camera access. The system runs natively on Windows without Docker: start_system.bat checks Python, Node.js and PostgreSQL, applies migrations, starts the three services and polls their health endpoints, while stop_system.bat and restart_system.bat stop or restart them.

V. Testing and Validation

A. Testing strategy

Component-level unit tests check alignment output shapes, detector behaviour on valid and empty images, embedding normalisation and determinism, quality scoring, cosine similarity, matching across multiple enrolments and the recognition-model wrapper. Integrated tests cover the ML service endpoints, student registration and enrolment, the full session lifecycle, duplicate-attendance prevention, and authentication and RBAC. The audited face2comics dataset (20,003 images of 512 × 512 pixels, about 10,000 real and 10,000 comic faces) has no identity labels, so it is used only for detector benchmarking and quality-score calibration; it cannot support identity-based FAR/FRR or threshold calibration. The evaluation framework also includes data-leakage checks.

B. Test cases and outcomes

In total, 73 automated tests were executed and all passed (Table 3). Frontend cases were run manually against the running system with three enrolled students: enrolment appeared in the student directory as ACTIVE, a live scan marked the student PRESENT with the time, a repeat scan returned ALREADY_MARKED (match score 0.718) without creating a record, the reports page showed 4 sessions, 8 present records and 3 enrolled students, the CSV export opened in a spreadsheet, and the diagnostics page showed the backend and ML service ONLINE. End-to-end cases also confirmed that attendance is recorded only in an ACTIVE session, that concurrent duplicates leave one record, that a USER account receives HTTP 403 on admin routes, and that UNKNOWN or LOW_QUALITY faces create no record.

Table 3: summary of automated test suites

Area	Suite	Tests	Result
ML pipeline	alignment, detection, embeddings, evaluation, matcher, ML service, quality, recognition model, similarity	43	passed
Attendance and sessions	backend/tests/test_attendance.py	8	passed
Auth and RBAC	backend/tests/test_auth_rbac.py	14	passed
Students, enrolment, health	backend/tests/test_backend.py	8	passed
Total		**73**	**73 passed, 0 failed, 0 skipped**

C. What has not been measured

Recognition accuracy is not yet measured. FAR, FRR, TAR@FAR (0.01, 0.001), ROC-AUC and Rank-1/Rank-5 are recorded as NOT MEASURED because they need real, consented, identity-aware, multi-session data. The development threshold is used only for functional testing (the repository documents 0.60 in the final validation and 0.5 as the configurable value in the attendance phase). Cross-browser and multi-device behaviour, end-to-end latency under load and pilot testing with human subjects were not recorded.

VI. Results and Discussion

A. Benchmark results

Benchmarks were run with scripts/benchmark_phase_12.py (Python 3.13, AMD64); mean execution times are given in Table 4 and Fig. 3. Embedding checks gave a dimension of 512, an L2 norm of 1.000000, a self-similarity of 1.000000 and a random cross-similarity of about -0.0026 (std 0.045), showing well-separated vectors in the embedding space.

Table 4: mean execution time of measured operations

Operation	Mean time
Cosine similarity (512-D)	0.0038 ms
Gallery search, 300 students / 1,500 vectors	5.86 ms
Gallery search, 1,000 students / 5,000 vectors	19.74 ms
Face quality scoring	0.19 ms
Argon2id hash / verify	51.57 ms / 50.95 ms
JWT creation	0.024 ms

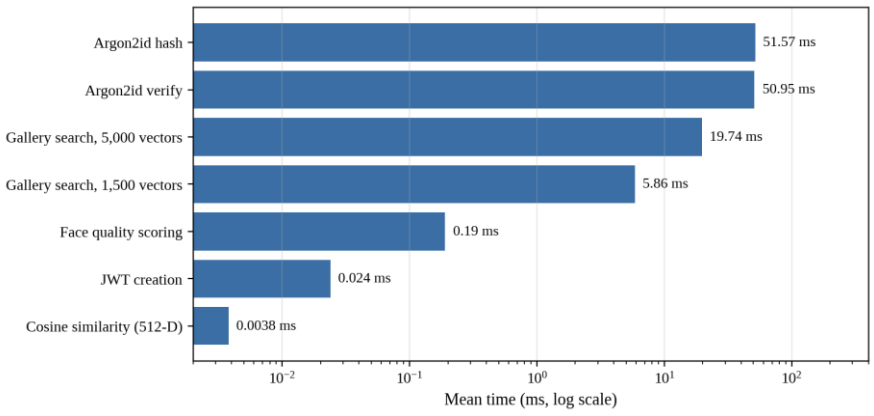


Fig. 3: mean execution time of measured operations (log scale)

B. Sample output

Fig. 4 shows the live attendance page captured during testing with three enrolled students: the browser camera runs the auto-attendance loop, the session roster lists the student as PRESENT with the time of marking, and a repeat scan returns ALREADY_MARKED. These screenshots demonstrate functional behaviour and are not a measure of recognition accuracy.

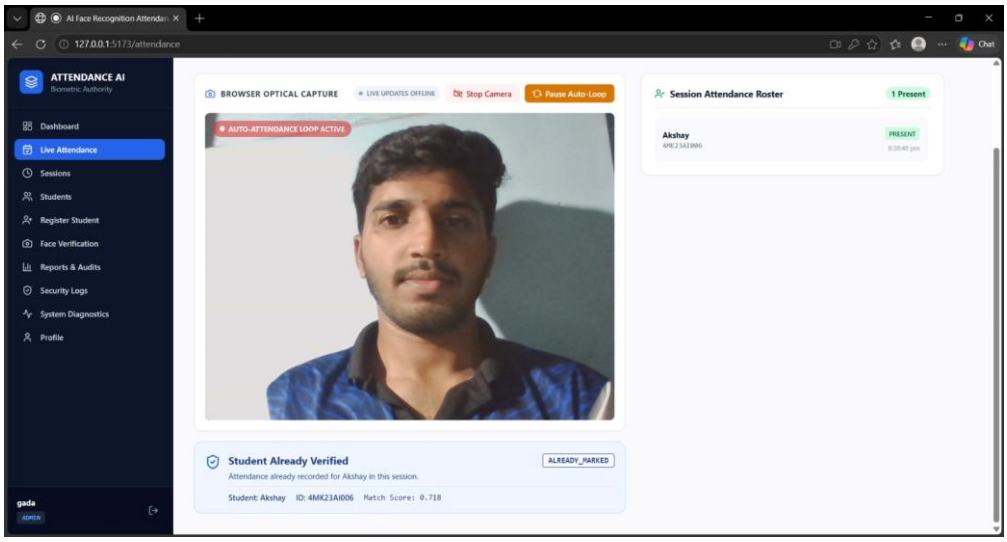


Fig. 4: live attendance page with session roster and ALREADY_MARKED result

C. Analysis and comparison

Matching cost is negligible compared with detection and embedding, so scaling to about 1,000 students stays well within real-time limits; the deliberately slow Argon2id hash affects only login. The main open issue is not speed but accuracy calibration, because without identity-labelled data the threshold cannot be justified statistically. The privacy design (no raw-image retention, irreversible 512-D vectors) and service separation reduce risk if one component is compromised.

Compared with the approaches in Table 1, the system combines contactless capture through a browser camera, a modern SCRFD and ArcFace pipeline, database-enforced duplicate prevention, role-based access control and fast gallery search (5.86 ms for 1,500 vectors, 19.74 ms for 5,000). Its proxy-attendance defence is a randomised head-movement challenge with alerts rather than a trained attack detector. The comparison is qualitative: accuracy for this system is not yet measured, and the figures reported by other studies come from their own datasets and protocols, so they should not be compared directly. The principal gap relative to published studies is the absence of identity-aware accuracy figures, which is the first item of future work.

VII. Conclusion and Future Work

This paper presented a complete, modular face-recognition attendance system with browser capture, an SCRFD and ArcFace pipeline, pgvector matching, session-based attendance with database-level duplicate prevention, secure authentication with RBAC, CSV reporting and one-click Windows deployment. All 73 automated tests pass and the benchmarks show real-time suitability.

The most important limitation is that recognition accuracy and the production threshold are not calibrated, which requires real, consented, multi-session data; the audited dataset lacks identity labels and differs strongly from real classroom conditions. Liveness relies on bounding-box motion only, and challenge state is kept in process memory, which limits deployment to a single worker or sticky routing. Future work will first collect a consented, identity-aware dataset and run the existing evaluator and threshold optimiser to obtain FAR, FRR, ROC and Rank-N metrics. A trained anti-spoofing model should then be added, with challenge state moved to shared storage such as Redis, and the system can be extended to multi-camera and whole-classroom group frames with small-face handling, FAISS or pgvector indexing for large galleries, a mobile or kiosk client, and richer analytics and absentee notifications.

Acknowledgment

The authors thank the Department of Artificial Intelligence and Machine Learning, Moodlakatte Institute of Technology Kundapura, for the support and resources provided for this work.

REFERENCES

[1] J. Guo et al., "Sample and Computation Redistribution for Efficient Face Detection (SCRFD)," ICLR, 2022.
[2] J. Deng et al., "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," CVPR, 2019.
[3] InsightFace project documentation.
[4] Y. Singh, "Automated attendance system using face recognition," Int. J. Sci. Res. Eng. Manag. (IJSREM), vol. 5, no. 7, Jul. 2021.
[5] A. Trivedi, C. M. Tripathi et al., "Face recognition based automated attendance management system," Int. J. Sci. Res. Sci. Technol., vol. 9, no. 1, pp. 261–268, 2022, doi: 10.32628/IJSRST229147.
[6] Jyoti L. et al., "Design of face recognition based effective automated smart attendance system," Indonesian J. Electr. Eng. Comput. Sci. (IJEECS), 2021.
[7] "Face recognition based attendance system using CNN and Haar classifier," J. Inf. Commun. Technol., vol. 16, no. 1, 2022, ILMA University.
[8] A. Rao, "AttenFace: A real time attendance system using face recognition," in Proc. IEEE Conf. Information and Communication Technology (CICT), 2022, arXiv:2211.07582.
[9] "Face recognition based attendance management system using deep learning method," in Proc. IEEE conference, Vellore, India, Feb. 2024, IEEE Xplore document 10508036.
[10] "Deep learning based attendance check system at FPT University," in Proc. ICIIT '24, ACM, 2024, pp. 272–281.
[11] H. Mikailu, F. N. Faseke et al., "Smart real-time attendance system for Nigerian universities," J. Inf. Organ. Sci., vol. 49, no. 1, 2025, doi: 10.31341/jios.49.1.8.
[12] "End-to-end automated student attendance recording system using surveillance camera," Revista Brasileira de Informática na Educação, pp. 371–393, 2025, doi: 10.5753/rbie.2025.5125.
[13] Kamil, "Effortless student attendance: A smart human-computer interactive system using real time facial recognition," An-Najah Univ. J. Res. A (Natural Sciences), vol. 39, 2025.
[14] "A proposed attendance check system in the smart academic library based on deep learning face recognition," Sci. J. Libraries, Documents Inf. Sci., vol. 6, no. 20, pp. 1–19, 2024.
[15] H. Al-Nayyef et al., "Advancing attendance: A facial recognition system empowered by deep learning techniques," J. Al-Qadisiyah Comput. Sci. Math., vol. 16, no. 1, 2024.
[16] M. A. Thalor and O. S. Gaikwad, "Facial recognition attendance monitoring system using deep learning techniques," Int. J. Integr. Sci. Technol., vol. 2, no. 1, 2024, doi: 10.59890/ijist.v2i1.1290.
[17] J. T. Santoso, E. Sediyono, K. D. Hartomo and I. Sembiring, "Optimizing attendance system: Integrating liveness detection and deep learning for reliable face recognition," Jurnal Informatika (JUITA), vol. 12, no. 2, pp. 207–216, 2024, doi: 10.30595/juita.v12i2.21738.
[18] "Comparative analysis of facial recognition based attendance system using machine learning," ITM Web Conf., vol. 81, art. 01018, 2026, doi: 10.1051/itmconf/20268101018.
[19] A. Biryukov et al., Argon2 password hashing specification.
[20] RFC 7519, JSON Web Token (JWT).
[21] FastAPI, SQLAlchemy, Alembic and pgvector documentation.

Copyright & License:

© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.