WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

NextGen Attend: A Smart Attendance Management System

Mr. B. Varun Kumar*[1], Akalya M*[2], Anusree K*[3], Divya S*[4]

[1-4]Department of Information Technology, Sri Shakthi Institute of Engineering and Technology, Coimbatore, Tamil Nadu, India
Email: akalyam25it@srishakthi.ac.in

ABSTRACT

Attendance recording is a routine activity in educational institutions, yet manual roll calls and conventional attendance tools can consume classroom time and may be vulnerable to proxy attendance, recording mistakes, and fragmented record management. This paper presents NextGen Attend, a smart attendance management application designed to support classroom attendance through a webcam-based computer vision workflow. The system combines a React-based user interface, Python FastAPI backend services, MediaPipe-based face detection, and MySQL data management. Its workflow includes student registration, live face detection, recognition against registered student information, a head-movement-based liveness check, automatic attendance marking, unknown-face handling, and spoken confirmation through the Web Speech API. A bulk-registration module supports importing student details and photographs from a prepared PDF, alongside separate upload options. Faculty and student interfaces provide role-oriented access to attendance operations and records. The paper describes the architecture, processing workflow, implementation components, and development observations of the prototype. Since controlled evaluation metrics have not been supplied, the manuscript does not claim measured recognition accuracy, spoof-resistance rates, or time savings. These outcomes require systematic testing under varied classroom conditions. The work demonstrates a practical integration of browser-based interaction, backend services, face-processing components, and database persistence for a college attendance workflow.

Keywords: Smart Attendance, Face Detection, Face Recognition, Liveness Detection, Computer Vision, FastAPI, MediaPipe, MySQL, Web Speech API

Manuscript draft prepared in the structure of the supplied model paper. Author metadata, final test results and journal-specific requirements must be verified before submission.

1. INTRODUCTION

Attendance is used by educational institutions to monitor student participation, maintain academic records, and support administrative decisions. In many classrooms, attendance is still recorded by calling student names, passing a register, or using systems that require repeated manual input. These practices may interrupt teaching and create opportunities for proxy attendance. Maintaining records across separate files or applications can also make retrieval and reconciliation difficult.

Computer vision offers a way to automate part of this process by detecting faces in camera frames and comparing detected identities with registered student information. However, an attendance application must do more than detect a face: it must manage registration, distinguish known and unknown people, avoid repeated marking within a session, preserve attendance records, and provide feedback to users. A camera-based workflow also introduces concerns about lighting, pose, privacy, consent, and presentation attacks.

NextGen Attend is developed as an AI-powered attendance management application for a college environment. It brings together a role-based interface, live camera processing, attendance marking, student data management, and voice feedback. The project uses React for the frontend, FastAPI for backend services, MediaPipe for face detection, and MySQL for persistent data. A bulk registration feature supports importing student details and photographs from a PDF document, with separate upload workflows also retained.

1.1. Objectives

The objectives of NextGen Attend are: (i) to provide a webcam-based interface for classroom attendance; (ii) to detect one or more faces in a live frame; (iii) to identify registered students through a recognition workflow; (iv) to apply a head-movement liveness check before marking attendance; (v) to mark attendance automatically and provide spoken confirmation; (vi) to flag faces that do not match registered identities; (vii) to support bulk registration from student details and photographs; and (viii) to maintain student and attendance information in a MySQL database.

1.2. Related Work

Viola and Jones introduced a rapid object-detection framework based on integral images, Haar-like features, and a cascade classifier, establishing an influential approach to face detection [1]. Modern face-recognition systems often represent a face image as an embedding and compare embeddings to determine identity. Schroff, Kalenichenko, and Philbin introduced FaceNet, which learns a compact Euclidean embedding for face verification, recognition, and clustering [2]. These methods inform the distinction between locating a face and determining whose face it is.

Liveness detection is an additional concern because face matching alone may accept a photograph or replay if the system does not assess whether the subject is physically present. Research on face anti-spoofing describes presentation attacks and the need to evaluate methods under different capture conditions [3]. NextGen Attend uses a simple head-movement check as a prototype liveness measure; it should not be interpreted as a comprehensive anti-spoofing solution.

1.3. Problem Statement

The project addresses four practical challenges: manual attendance is time-consuming; proxy attendance may occur; fragmented records complicate management; and registration of a class can require repetitive data entry. The proposed application combines face-based identification, a liveness step, automatic marking, database-backed records, and bulk registration. The effectiveness of each measure depends on correct implementation and validation.

2. SYSTEM ARCHITECTURE

2.1. Overview

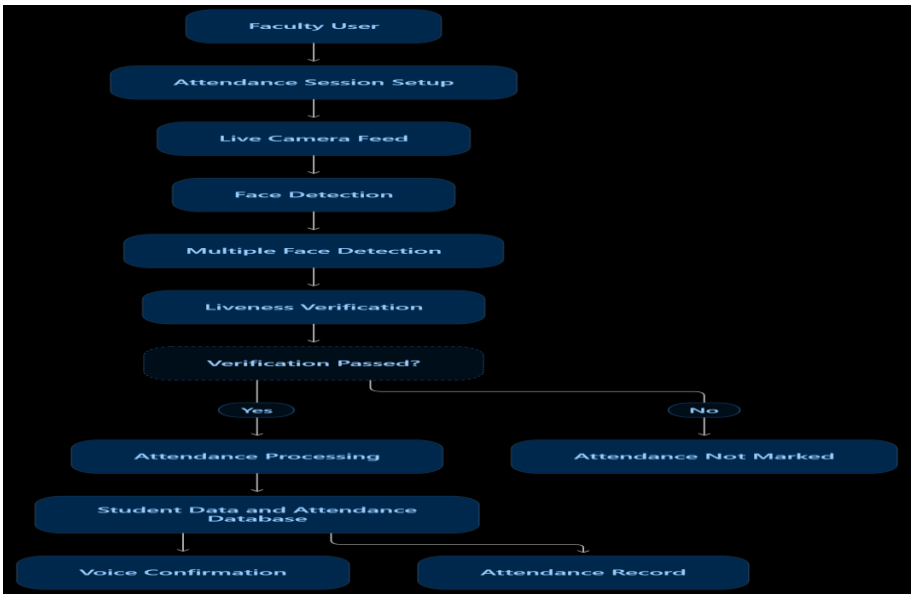
NextGen Attend follows a layered application architecture. The presentation layer contains the React user interface, including role selection, login, faculty and student dashboards, attendance setup, and live attendance. The application layer is implemented using Python and FastAPI and coordinates registration, attendance requests, and database operations. The computer-vision component processes camera frames, detects faces, and supports the recognition and liveness workflow. The data layer uses MySQL to store student and attendance records

Layer	**Technology / component**	**Purpose**
Presentation	React.js, JSX, CSS	Role-oriented screens, camera view, dashboards and feedback
Backend	Python, FastAPI	API handling, validation and attendance workflow coordination
Vision	MediaPipe Tasks Vision; recognition workflow	Face detection, face matching and movement-based liveness check
Database	MySQL	Persistent student details and attendance records
Browser feedback	Web Speech API	Spoken confirmation after successful marking
Registration	PDF and separate upload workflow	Bulk student details and photograph registration

2.2. System Architecture Diagram

USER / BROWSER	**APPLICATION SERVICES**	**DATA / PROCESSING**
Faculty or Student React UI Login / Dashboard Live camera	FastAPI backend Request validation Attendance coordination Registration service	MySQL Student records Attendance sessions Attendance entries
Camera frame →	Face detection → Recognition → Liveness decision	Registered identity comparison
Voice confirmation ←	Mark attendance only when workflow conditions pass	Persist attendance record

Fig. 1. Editable three-layer architecture and interaction summary of NextGen Attend.



The frontend sends requests to backend endpoints through the application interface. Backend validation is responsible for checking the request and applying the relevant business rules before database updates. Face detection identifies candidate face regions; recognition associates a detected face with a registered identity when a match is available. A liveness decision is considered before attendance is marked. The backend stores the resulting attendance event, while the frontend displays status and may speak a confirmation.

2.3. Technology Stack

The technology selection reflects the project's web-based implementation. React supports reusable UI components; FastAPI provides Python-based API services; MediaPipe Tasks Vision provides the face-detection component; and MySQL provides relational persistence. The Web Speech API supplies browser speech synthesis for confirmation. The precise recognition model, matching threshold, API routes, and final deployment configuration should be documented from the current source code before journal submission.

2.4. Database Design

The database stores student identity and attendance data as related records. The student entity includes an identifier, name, roll number, department, year, section, photograph path, and recognition-related data where implemented. Attendance entries associate a student with an attendance session and date/time. A unique roll number and relational constraints can help prevent duplicate registration and maintain referential consistency. The final schema and constraints should be reproduced from the deployed database rather than inferred from this conceptual description.

3. METHODOLOGY

3.1. Student Registration and Bulk Import

Registration establishes the student records used by the attendance workflow. NextGen Attend supports a bulk-registration process in which a prepared PDF contains student details and photographs. The application extracts or processes the supplied information, associates photographs with student records, and registers the resulting entries. Separate student-details and photo-upload options are also retained to accommodate alternate input workflows. Input validation is important because incomplete names, duplicate identifiers, missing photographs, or mismatched photo-to-student associations can affect.

3.2. Live Face Detection

During a live attendance session, the browser obtains camera frames and the vision component detects face regions. MediaPipe Tasks Vision is used for face detection. Detected bounding boxes are drawn on the live view to provide visual feedback. Multiple faces may be detected in the same frame. Detection only locates a face; it does not by itself establish the identity of the person.

3.3. Face Recognition and Identity Matching

A detected face is passed through the project's recognition workflow and compared with registered student information. Conceptually, a recognition system may compare a face representation with enrolled representations and accept a candidate only when the similarity criterion is met. The specific embedding model and matching rule must be confirmed against the final code. An unrecognized face is treated as unknown rather than assigned to a student by guesswork.

3.4. Liveness Verification

To reduce simple photo-based proxy attempts, the current prototype uses head movement as a liveness cue for a single detected face. A change in face position is measured across successive frames and compared with a movement threshold. When movement is sufficient, the workflow proceeds; otherwise, attendance marking is withheld. This is a basic challenge-like check and may be affected by camera placement, natural movement, occlusion, and video replay. It is not equivalent to a trained anti-spoofing model.

3.5. Automatic Attendance and Voice Feedback

After the required identity and liveness conditions are satisfied, the application marks the eligible student as present. A session should prevent repeated marking of the same student during the same attendance period. The browser's speech synthesis capability can announce a successful event, for example, “Attendance marked successfully for [student name].” Voice feedback informs the user but is not itself evidence that a database transaction succeeded; the application should speak only after the backend confirms persistence.

3.6. Processing Workflow

Step	Operation	Decision / output
1	Start session and open camera	Camera permission and stream available
2	Detect face(s) in frame	No face / one face / multiple faces
3	Recognize detected identity	Registered identity or unknown
4	Apply liveness check as configured	Pass / wait / reject
5	Check session duplicate	Already marked / eligible
6	Send attendance request	Backend validates and persists
7	Display and speak confirmation	Success or error feedback

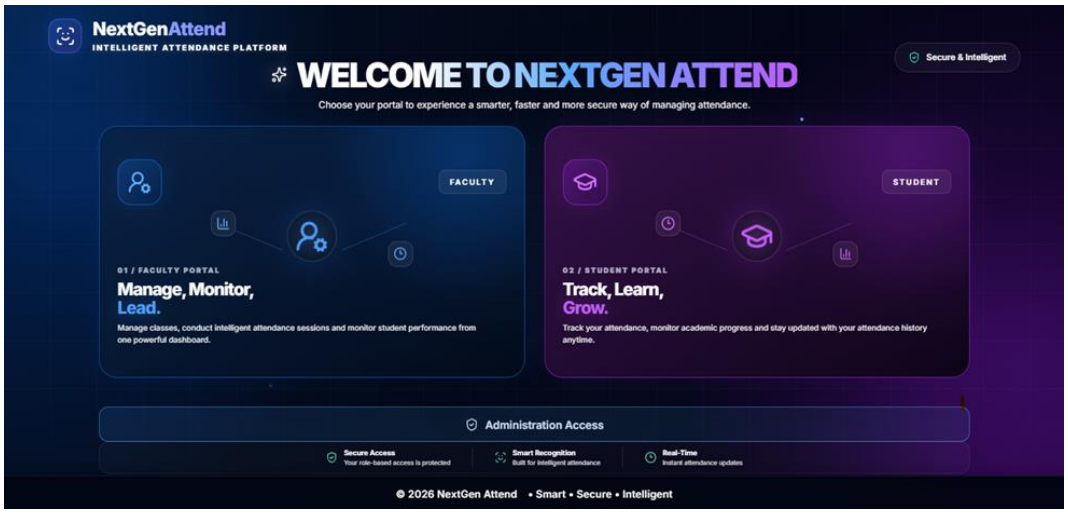


Fig. 2. Editable attendance workflow summary.

3.7. Recognition and Attendance Logic

A general identity-matching workflow can be expressed as a comparison between a query representation and enrolled representations. Let q denote the query face representation and r_i a registered representation. A distance-based matcher may select the nearest enrolled candidate:

$$d(q, r_i) = ||q - r_i||_2$$

3.8. Attendance Record Integrity

Attendance persistence should be tied to a session and student identity. A conceptual uniqueness rule is (student_id, session_id), which can prevent duplicate entries for the same student in a session. The backend should validate identity, session state, and duplicate status before writing a record. If the database operation fails, the interface should report failure and avoid issuing a successful voice announcement. Final implementation details must be checked against the actual MySQL schema and API behavior.

3.9. Security and Privacy Considerations

The system handles student photographs, identity information, and attendance history. Access should therefore be limited to authorized roles, credentials should be stored using secure password hashing, and protected API operations should validate the authenticated user. Database credentials should not be embedded in frontend code. Camera access should be requested only for the attendance workflow, and the institution should define retention, deletion, consent, and access policies for biometric-related data. These are recommended controls; the final manuscript should state only controls that have been implemented and tested.

4. IMPLEMENTATION

4.1. User Interface

The React interface provides role selection and login, followed by faculty and student dashboards. Faculty-facing attendance screens support setup and live camera operation. Student-facing screens are intended to provide access to the student's relevant attendance information. The live attendance layout presents the camera view alongside an attendance status panel. Interface behavior depends on browser camera permission and device availability.

4.2. Backend Services

FastAPI acts as the service layer between the frontend and MySQL. It receives application requests, validates data, coordinates registration and attendance actions, and returns structured responses. The backend separates client presentation from data persistence, making it possible to update the UI without directly exposing database operations in the browser.

4.3. Database Integration

MySQL stores student details and attendance records. The database connection is configured in the backend environment. Persistent data enables attendance history to be retrieved after a session and supports dashboard summaries. Database connectivity, schema constraints, duplicate handling, and recovery from failed writes should be included in the final test record.

5. RESULTS AND DISCUSSION

The current development records describe an integrated application workflow rather than a completed controlled accuracy study. The implemented project components include a role-based React interface, camera-based face detection, a recognition and liveness workflow, attendance marking, speech confirmation, bulk student registration, and backend/database integration. The following observations are therefore functional-development observations; they are not quantitative performance results.

5.1. Functional Observations

Module	**Observed / intended function**	**Evidence needed for publication**
Live camera	Camera stream and face boxes are displayed	Screenshots and device/browser details
Face handling	Multiple detections and unknown faces are handled	Test cases and recognition outcomes
Liveness	Head movement used as a basic cue	Attack and genuine-user test protocol

Module	Observed / intended function	Evidence needed for publication
Attendance	Eligible identity is marked automatically	Database records and duplicate test
Voice feedback	Success message is spoken	Browser test and persistence confirmation
Bulk registration	PDF-based and separate upload workflows	Sample input, imported count, error report
Database	Student and attendance records persist	Schema, query checks, restart/retrieval test

5.2. Evaluation Plan

A journal-ready evaluation should use a consented and appropriately governed test set representing realistic classroom variation. Recognition evaluation should report the number of enrolled identities, genuine and impostor comparisons, false acceptance rate, false rejection rate, and overall identification performance. Liveness evaluation should separately test genuine users and presentation attacks, including printed photographs and replayed video, and report attack acceptance and genuine-user rejection. Operational evaluation should measure end-to-end attendance latency, successful database writes, duplicate prevention, and registration errors. Test conditions should include different lighting, face poses, camera distances, and group sizes.

5.3. Discussion

The project's primary contribution is the integration of several attendance-related tasks into one application workflow. Face detection supports locating people in camera frames, recognition supports identity assignment, liveness introduces an additional check, and backend persistence makes attendance records available for later retrieval. Bulk registration addresses a separate operational burden by reducing repeated student entry. The components should be evaluated individually as well as end-to-end, since errors in enrollment, detection, matching, liveness, or persistence can affect the final attendance outcome.

6. LIMITATIONS

Several limitations should be considered when interpreting the prototype. First, face recognition performance may vary with illumination, camera quality, pose, occlusion, and changes in appearance. Second, head movement is a limited liveness signal and can be bypassed by more sophisticated presentation attacks; it should not be described as robust anti-spoofing without dedicated testing. Third, camera-based systems may produce false matches or fail to recognize a student, requiring a transparent manual correction or fallback procedure.

Bulk registration depends on the quality and consistency of the source PDF. Incorrect extraction, mismatched photographs, duplicate student identifiers, or incomplete records can propagate into recognition and attendance operations. MySQL persistence and backend availability are also necessary for reliable operation. Browser speech synthesis may differ across devices and should not be treated as a required success criterion.

Finally, the project handles sensitive student data. Institutional authorization, informed notice, data minimization, access control, secure storage, retention limits, and a process for correcting attendance are essential for responsible deployment. The current manuscript does not claim that a formal privacy assessment or institutional deployment audit has been completed.

7. CONCLUSION AND FUTURE SCOPE

NextGen Attend presents a web-based smart attendance management workflow that combines face detection, a recognition process, movement-based liveness checking, automatic attendance marking, voice confirmation, bulk student registration, and database-backed record management. The architecture uses React for the user interface, FastAPI for backend services, MediaPipe for face detection, and MySQL for persistent data. The project is intended to reduce repetitive attendance work and organize attendance operations within a single application.

The development illustrates how computer vision and web technologies can be integrated into an educational attendance application. However, the reliability and suitability of the system must be established through controlled testing rather than inferred from feature implementation. Future work includes evaluation across realistic classroom conditions, stronger presentation-attack detection, calibrated recognition thresholds, robust duplicate prevention, audit trails, secure data-retention controls, and a documented manual correction process. Performance results and final database/API details should be added after validation against the completed system.

ACKNOWLEDGEMENTS

The authors acknowledge the guidance and support provided by the faculty and Sri Shakthi Institute of Engineering and Technology during the development of the NextGen Attend project. Faculty name and acknowledgement wording should be confirmed before submission.

REFERENCES

- P. Viola and M. J. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” in Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2001, vol. 1, pp. I-511–I-518. doi: 10.1109/CVPR.2001.990517.\
- 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. doi: 10.1109/CVPR.2015.7298682.
- R. Ramachandra and C. Busch, “Presentation Attack Detection Methods for Face Recognition Systems: A Comprehensive Survey,” ACM Computing Surveys, vol. 50, no. 1, Article 8, pp. 1–37, 2017. doi: 10.1145/3038924.
- Google AI for Developers, “MediaPipe Face Detector,” MediaPipe documentation. Available: https://ai.google.dev/edge/mediapipe/solutions/vision/face_detector. Accessed: 3 Nov. 2026.
- FastAPI, “FastAPI Documentation.” Available: https://fastapi.tiangolo.com/. Accessed: 3 Nov. 2026.
- React, “React Documentation.” Available: https://react.dev/. Accessed: 3 Nov. 2026.
- Oracle, “MySQL 8.4 Reference Manual.” Available: https://dev.mysql.com/doc/. Accessed: 3 Nov. 2026.
- MDN Web Docs, “Web Speech API.” Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API. Accessed: 3 Nov. 2026.
- Agustiyar, A., Isnanto, R. R., Widodo, C. E., and Zaini, N. A. B. (2025). “A Systematic Literature Review of Face Recognition Based Attendance System: Methods, Trends, and Research Gaps.” This study reviews face recognition methods, feature extraction techniques, and research challenges in automated attendance systems. ResearchGate
- Google AI Edge (2025). “MediaPipe Face Detector Documentation.” This documentation explains the use of face detection models to locate faces in images and video streams, supporting the webcam-based face detection component of NextGen Attend.CCN+1