

ATTENDAI: A MULTI-FACTOR STUDENT ATTENDANCE SYSTEM

Arsath Mohamed M¹, Gokul M², Rajamanikandan R³, Mr.M.Premkumar [Assistant professor]⁴

Department of Artificial Intelligence and Data Science, Sengunthar Engineering College, Tamil Nadu, India

Abstract— Manual attendance recording consumes classroom time and may permit proxy attendance. Face-recognition-based attendance systems can automate identity matching, but their reliability may be affected by illumination, image quality, and presentation attacks. This paper presents ATTENDAI, a webbased student attendance system designed to combine face recognition, voice verification, and eye-blink liveness detection in a staged verification workflow. The proposed implementation uses Python and Flask for application services, OpenCV with Haar Cascade face detection and Local Binary Patterns Histograms (LBPH) for face processing, Mel-Frequency Cepstral Coefficients (MFCC) for voice features, and facial landmarks with the Eye Aspect Ratio (EAR) for blink detection. Attendance is recorded only after the configured verification stages succeed. The system also supports Admin, Staff, and Student roles, departmentwise management, attendance reporting, and a four-hour session rule intended to prevent duplicate entries while retaining historical records. This paper describes the motivation, system architecture, principal modules, session logic, and an evaluation plan. Quantitative performance results are not claimed because they must be obtained from controlled testing of the completed implementation.

Keywords - Student attendance, face recognition, LBPH, voice verification, MFCC, eye-blink detection, EAR, Flask, multifactor authentication.

I. INTRODUCTION

Attendance is an important administrative record in educational institutions. Paper registers and roll-call methods require staff time, make record consolidation difficult, and may allow one student to respond for another. Digital attendance systems can automate record creation and retrieval, but reliable identity verification remains essential. Face recognition provides a contactless method of matching a captured face with an enrolled identity. However, performance may vary with illumination, camera position, facial expression, and image quality. A face-only system may also be unable to determine whether the person in front of the camera is live. These issues motivate the use of additional verification signals and explicit attendance-session rules.

ATTENDAI is designed as a web-based application that combines face recognition, voice verification, and eye-blink liveness detection. Face recognition identifies a candidate student, voice verification compares a live voice sample with the registered profile, and blink detection checks for an eyestate sequence that indicates a live response. Attendance is recorded only after the required stages pass. The application also provides role-based access, department-wise organization, and report export. The contributions of this project are:

- a staged multifactor attendance workflow;
- integration of face, voice, and blink verification

- a four-hour session mechanism for duplicate prevention while preserving historical records
- role-based access and attendance reporting in a web application.

II. RELATED WORK

Salim, Olanrewaju, and Balogun presented a class attendance management system using face recognition [1]. Their prototype used a Raspberry Pi camera, Haar-feature cascade detection, Local Binary Patterns, a database, and a web-based attendance interface. The authors reported a 95% F1-score for a dataset of eleven students and identified illumination sensitivity as a limitation. This work is the principal base paper for ATTENDAI. Viola and Jones introduced a boosted cascade of simple features for rapid object detection, a method widely associated with Haar-like feature cascades [2]. Ahonen, Hadid, and Pietikainen investigated face description using Local Binary Patterns, which encode local texture information for face recognition [3]. These approaches provide a lightweight foundation for face detection and recognition in resource-constrained applications.

Mel-Frequency Cepstral Coefficients are widely used to represent short-term spectral characteristics of speech [4]. Nearest-neighbour classification, including K-Nearest Neighbours (KNN), provides a simple approach for comparing feature vectors [5]. In ATTENDAI, MFCC features and a voice matching component are used as part of the voice-verification stage. Facial landmarks and the Eye

Aspect Ratio are used to estimate whether the eyes transition from open to closed and back to open. Unlike the base paper's face-based workflow, ATTENDAI proposes additional voice verification, blink-based liveness checking, a four-hour session rule, and role-based application access. These additions are design extensions and require independent evaluation.

Problem Statement And Objectives

The problem addressed is the need for an attendance system that reduces manual work and strengthens identity verification beyond a single stage. A face-recognition system may fail under changing lighting and may not provide explicit liveness evidence. Repeated attendance submissions also need to be controlled without deleting historical records.

The objectives are to:

- detect and recognize registered students using a camera;
- verify a live voice sample against the registered voice profile
- detect an eye-blink sequence as a liveness step
- record attendance only after all required verification stages succeed
- provide Admin, Staff, and Student access roles
- organize students and reports by department; and
- prevent duplicate entries within a four-hour session while retaining older records.

III. PROPOSED SYSTEM ARCHITECTURE

ATTENDAI is organized into four logical layers: presentation, application service, verification, and data management. The presentation layer provides login, registration, camera preview, dashboards, and reports. The Flask application layer handles requests, checks permissions, coordinates verification stages, and returns status information. The verification layer contains face detection and recognition, voice feature extraction and matching, and blink detection. The data layer stores student details, user accounts, biometric profile references, attendance entries, and session identifiers.

The workflow begins when an authorized user opens the attendance page. The camera captures frames and Haar Cascade detects a face region. The LBPH recognizer compares the detected face with registered data. The voice stage then compares a live audio sample with the registered voice profile. After successful voice verification, blink detection checks for an open–closed–open eye sequence. If all configured checks pass, the application checks whether the student has already marked attendance in the current four-hour session. A new record is inserted only if no duplicate

exists

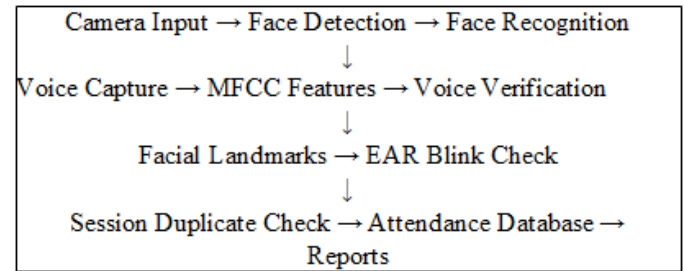


Fig. 1. Proposed ATTENDAI verification workflow

The interface should display clear states such as “Face Recognized,” “Voice Verified,” “Blink Detected,” and “Attendance Marked.” A failed stage must stop the workflow and return a specific status. Camera or microphone errors should not leave the page indefinitely loading.

IV. FACE RECOGNITION MODULE

Face detection identifies the region of interest in each camera frame. Haar Cascade is used as the detector in the current design. The detected face is resized and converted to grayscale before recognition. LBPH represents local texture patterns and compares the input face with the registered face model. In a basic Local Binary Pattern calculation, each neighbouring pixel is compared with a central pixel. The comparisons form a binary code, and codes are summarized in histograms over facial regions. The resulting representation is compared with enrolled samples to estimate the closest identity. LBPH is suitable for a small software prototype because it can run on ordinary CPU hardware. However, it remains sensitive to illumination, camera position, expression, and image quality. Uncertain matches should be rejected rather than automatically accepted. During testing, logs should distinguish face-detection failure from recognition failure. Useful diagnostic information includes whether a face was detected, whether the model was loaded, the predicted student identifier, and the confidence or distance value. The recognition threshold must be calibrated using a separate test dataset.

V. VOICE VERIFICATION MODULE

Voice verification provides a second identity signal. During registration, the student supplies multiple voice samples; the current project design collects three samples and processes them to create a voice profile. During attendance, a new sample is captured and compared with the registered profile.

MFCCs represent short-term spectral characteristics of speech and are commonly used in speech-processing systems [4]. A

matching component compares the live representation with the registered profile. The project design refers to KNN-based voice classification or verification, but the threshold and measured performance must be established experimentally. The system should handle quiet audio, clipping, background noise, and microphone-access errors.

A voice mismatch must stop the workflow before attendance is marked. The voice module should return a structured match status and, where available, a confidence or distance value. Voice recordings and derived profiles should be protected and retained according to an explicit institutional policy.

VI. EYE-BLINK LIVENESS DETECTION

Eye-blink detection is used as a liveness step after face and voice verification. Facial landmarks identify points around the left and right eyes. The Eye Aspect Ratio (EAR) is calculated from distances between selected eye landmarks. A simplified EAR expression is

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

where $p_1, \dots, p_6$ represent ordered eye landmarks. The ratio generally decreases when the eye closes and rises when it opens.

The implementation should use a sequence-based state machine rather than relying on one frame. It can require the closed-eye condition to persist for a small number of consecutive frames, then latch success when the eyes reopen. EAR thresholds should be calibrated with camera data because face shape, camera position, and landmark quality affect measured values.

Temporary face-recognition uncertainty during a blink should not automatically erase the closed-eye state after face and voice have already been verified. The verified identity can be retained for the short liveness step, with explicit rules for face loss or multiple faces. This reduces the possibility of missing a blink because the detector resets on one imperfect frame. Blink detection is a basic liveness signal and should not be described as protection against every possible presentation attack.

VII. FOUR-HOUR ATTENDANCE SESSION

The session mechanism divides a day into six fixed windows: 00:00–03:59, 04:00–07:59, 08:00–11:59, 12:00–15:59, 16:00–19:59, and 20:00–23:59. Each window receives a session identifier. A student may have at most one accepted

attendance entry per session, while entries from earlier sessions remain available for reports.

The duplicate check should use the combination of student identifier and session identifier. Checking only the date would prevent a student from recording attendance in later sessions on the same day. Deleting the entire attendance table every four hours would destroy historical data and is not the intended behaviour. Session boundaries should use the institution's local time and should be tested at exact transition points.

Attendance insertion should be treated as a database operation whose result is checked. The application reports success only after the database confirms that a new record was inserted. If the student already has a record in the current session, the system returns a duplicate status and preserves the existing record. A database-level uniqueness constraint on the student-session pair can provide additional protection against simultaneous duplicate requests.

VIII. DATA FLOW AND DATABASE DESIGN

The attendance process depends on consistent identifiers across registration, verification, and reporting. Each student should have a stable student identifier that is not generated from a biometric measurement. The student profile stores basic institutional details and references to enrolled biometric data. Attendance records should store the student identifier, date and time, session identifier, and final verification status. Keeping profile data separate from attendance transactions makes it easier to update student details without changing historical records. A typical data flow starts with an authenticated request to the attendance page. The application checks the user's role and initializes the camera and microphone components. Captured frames and audio samples are passed to the relevant verification modules; they should not be treated as attendance records by themselves. The modules return structured results to the application service. Only after the required checks return successful statuses does the service calculate the active session and attempt a database insertion. The database result is then converted into a success, duplicate, or error response for the browser.

The database should enforce important rules instead of relying only on interface logic. In particular, a unique constraint on the pair of student identifier and session identifier prevents two simultaneous requests from creating duplicate attendance rows. Transactions should be used when multiple related fields are inserted. Database errors must be logged for administrators while the user receives a clear message that

attendance could not be saved. This distinction prevents the interface from reporting success when a write has failed.

IX. ATTENDANCE DECISION PROCEDURE

The decision procedure should be deterministic and auditable. First, the application confirms that the current user is allowed to use the attendance function. Second, the face module returns a candidate identity and a match status. Third, the voice module verifies that the live sample is consistent with that candidate's registered profile. Fourth, the liveness module detects the expected blink sequence. Finally, the session service checks for an existing record and writes the new attendance entry if permitted.

The following pseudocode summarizes the intended control flow:

Input: student candidate, face status, voice status, blink status, current time

- If face status is not verified, reject the attempt.
- If voice status is not verified, reject the attempt.
- If blink status is not verified, reject the attempt.
- Compute the four-hour session identifier.
- If a record exists for this student and session, return duplicate status.
- Otherwise insert attendance and confirm the database result.
- Return success only after the insert is committed.

This order makes the final attendance decision independent of intermediate user-interface messages. Each stage should return an explicit status, and the application should record the reason for rejection without exposing sensitive biometric information to unauthorized users. A retry should restart only the failed stage when safe; it should not bypass a required verification step.

1. User Roles And Reporting

ATTENDAI separates access into Admin, Staff, and Student roles. Admin users manage the application, including users and departments. Staff users access attendance information for permitted departments or classes. Students use the attendance flow and view the records made available to them. Role checks must be enforced by backend routes; hiding a button in the interface alone is not a security control. Attendance management includes viewing student and class records and exporting supported formats. Reports should contain fields such as student identifier, name, date, time, department, and session identifier. Export functions should respect user

permissions and avoid exposing records belonging to unrelated departments.

2. Security, Privacy, And Reliability

biometric information requires careful handling because a voice sample or face image is more sensitive than an ordinary attendance field. Access to registration and biometric management functions should be restricted to authorized roles. The system should collect only the data required for its stated purpose, protect database files and backups, and avoid placing raw biometric samples in application logs. If raw samples are retained for model updates, the institution should define a retention period and an appropriate consent process.

Authentication and authorization are separate responsibilities. Login confirms the account identity, while authorization determines which operations that account may perform. Every protected server route should verify the current role and permitted department or class. A student should not be able to retrieve another student's records by changing an identifier in a request. Administrative operations such as deleting profiles, changing departments, or exporting class-wide records should be auditable.

Reliability also depends on handling device and network failures. Camera permission denial, microphone unavailability, a missing recognition model, or a database connection error should produce a visible recovery message rather than an endless loading state. Camera streaming should have a controlled lifecycle so that repeated page visits do not leave multiple capture loops running. When the browser disconnects, the server should release resources where appropriate. Logs should identify the failed component and time while excluding unnecessary personal data.

X. IMPLEMENTATION TECHNOLOGIES

Python is used for processing and Flask for web routes and request handling. OpenCV provides camera access and image-processing functions. Haar Cascade and LBPH support face detection and recognition. The voice pipeline uses audio capture and MFCC feature extraction together with a matching component. Facial landmarks and EAR calculations support blink detection. A database module stores student, user, and attendance information, while HTML, CSS, and JavaScript provide the browser interface.

The system is intended to operate with a standard webcam and microphone rather than dedicated biometric hardware. Its modular structure allows the camera, verification, attendance, and reporting components to be tested separately. Package

versions, database schema, and operating-system dependencies should be recorded to improve reproducibility.

XI. TESTING AND EVALUATION PLAN

The system should be evaluated using a documented protocol. Face recognition should be tested with registered and unregistered students under different lighting and head positions. Voice verification should be tested with registered voices, nonregistered voices, background noise, and different recording levels. Blink detection should be tested with natural blinks, no-blink sequences, and brief landmark-loss events. End-to-end tests should verify that attendance is marked only after all required stages succeed.

The four-hour logic should be tested at session boundaries and with repeat submissions. Role-based access should be tested by attempting restricted routes as each role. Reports should be checked against database records. Useful metrics include face precision and recall, voice false-accept and false reject rates, blink detection success rate, end-to-end completion time, duplicate rejection rate, and application error rate. No numerical accuracy result is asserted in this paper because measured project test results have not been supplied. After testing, a results table should report sample counts, test conditions, successful and failed cases, and calculation methods. This avoids presenting unverified figures as experimental findings.

XII. OPERATIONAL CONSIDERATIONS AND LIMITATIONS

A classroom deployment introduces conditions that may not be represented by a small laboratory test. Students may stand at different distances from the camera, wear glasses, speak softly, or attend in rooms with fans and conversations in the background. Lighting may change between morning and afternoon sessions. These factors can affect the face, voice, and landmark modules independently, so the application should provide instructions about camera position, lighting, and microphone distance. The staged process can improve confidence in an attendance decision, but it also creates more opportunities for a legitimate student to be rejected because one sensor produces a poor sample. The user interface should allow a controlled retry and should provide a staff-assisted fallback procedure for accessibility needs or device failures. A fallback must be recorded distinctly rather than silently treating incomplete verification as a successful multi-factor check.

The current design uses relatively lightweight methods that can run on common computers. This is useful for a student project and may reduce hardware costs, but it does not establish production-grade security. LBPH can be sensitive to changes in appearance and illumination, MFCC features alone are not a complete guarantee of speaker identity, and a blink check is not resistant to every spoofing method. A real deployment should test attack scenarios, secure biometric storage, review applicable institutional policy, and obtain appropriate approval before collecting student biometric data.

XIII. DISCUSSION

The main design benefit of ATTENDAI is the use of multiple verification stages rather than treating face matching alone as sufficient. The workflow separates an identity match from the final attendance decision. The session identifier also separates attendance eligibility from historical record retention. Multi-factor verification introduces trade-offs. Voice capture can be affected by ambient noise and privacy concerns; blink detection requires a clear view of the eyes; and multiple stages can increase completion time. The application must communicate requirements clearly, recover from device errors, and never treat incomplete verification as successful. Performance and usability should therefore be measured together.

Error Handling And User Experience

A reliable attendance interface should make the current verification stage visible to the user. The page may show instructions such as positioning the face inside the camera frame, recording a short voice sample, and blinking naturally when prompted. Status messages should be concise and should distinguish a retryable issue from a final rejection. For example, “microphone not available” is a device problem, while “voice did not match” is a verification result. These cases should not be presented as if they were the same error. The browser and server must agree on the state of an attendance attempt. When a new attempt begins, the application should initialize the required flags and clear stale state from the previous student or session. Once the face and voice stages succeed, the verified candidate should be associated with the active attempt. The blink stage should have a bounded timeout, and the page should provide a retry or restart option if the required sequence is not detected. When an attempt is cancelled, the application should clear the pending state so that another user cannot inherit a previous user’s verification status.

Asynchronous camera and microphone operations need explicit success and failure handling. The interface should not

wait forever for a video frame or audio recording. If the stream stops, the application should show an error and allow the user to restart the camera. Server endpoints should return predictable response structures, including a status code and a short message. Diagnostic details can be written to protected logs, but internal paths, stack traces, biometric vectors, and other sensitive information should not be sent to the browser. Accessibility and fairness are important operational considerations. A user may be unable to perform a standard blink on demand, may have a speech difference, or may encounter poor sensor conditions. The institution should define a supervised alternative that preserves accountability and is clearly labelled as a manual or assisted attendance entry. Such an alternative should not be silently counted as a successful face-voiceblink verification, because doing so would make the system's records misleading.

XIV. THREAT MODEL AND SAFEGUARDS

The system should consider both accidental errors and intentional misuse. Accidental cases include duplicate clicks, a page refresh during verification, device disconnection, a delayed database response, or two requests arriving at nearly the same time. Intentional cases may include attempts to use another student's account, submit a photograph to the camera, replay recorded audio, access another department's records, or call a protected endpoint directly without using the interface.

Several safeguards address these risks. Server-side role checks prevent interface-only restrictions from being bypassed. Database uniqueness constraints protect against simultaneous duplicate inserts. Short-lived attempt state and explicit reset rules reduce the risk of carrying a verified identity into another attempt. Rate limits and audit logs can help identify repeated failed attempts. Transport security should be used when the application is deployed beyond a trusted local development environment, and passwords should be stored using a suitable password-hashing mechanism rather than reversible plaintext. These controls reduce risk but do not guarantee complete prevention of spoofing. A basic blink sequence can be imitated by some presentation attacks, and ordinary MFCC-based feature matching should not be treated as equivalent to a modern speaker-verification model. Stronger liveness checks, challenge-response voice prompts, replay detection, and independent security testing are possible future improvements. Their effectiveness should be established by experiments rather than assumed from their presence in the architecture.

XV. EXPECTED OUTCOMES AND SUCCESS CRITERIA

The expected outcome is a working prototype that links the three verification stages to one attendance decision and maintains consistent records across repeated sessions. The system should be considered functionally successful only when positive and negative test cases behave as specified. For example, a recognized face alone must not create a record if voice verification fails; a valid face and voice must not create a record if the required blink sequence is absent; and a fully verified student must receive a duplicate response if attendance already exists for the active session.

Functional evaluation should be separated from biometric evaluation. Functional tests check navigation, registration, permissions, session calculation, record insertion, duplicate rejection, and export. Biometric tests measure how accurately each module accepts genuine users and rejects impostors under defined conditions. Reporting the two categories separately helps identify whether an error comes from recognition quality, application logic, or data storage.

For reproducibility, each experiment should document the number of participants, number of attempts, camera and microphone hardware, lighting and noise conditions, software versions, thresholds, and criteria for success. Results should include both the numerator and denominator behind every percentage. If the same samples are used to tune thresholds and evaluate performance, the reported result may be overly optimistic; a separate evaluation set is preferable. The system should also record the time required for each stage to identify bottlenecks that affect classroom usability.

Table I Suggested Evaluation Measures

Measure	Purpose
Face recognition rate	Proportion of test faces correctly identified under documented conditions.
Voice false acceptance	Impostor voice attempts incorrectly accepted by the voice stage.
Voice false rejection	Genuine registered voice attempts incorrectly rejected.
Blink detection rate	Proportion of required blink sequences correctly detected.
Duplicate rejection	Repeat attempts in one session correctly prevented from creating a second record.
End-to-end time	Time from starting verification to final success or rejection.

A suitable evaluation table can be completed after testing: These measures provide a transparent basis for future comparison with alternative face recognizers, speaker-

verification methods, or liveness techniques. They should be populated with measured values rather than estimates.

XVI. CONCLUSION AND FUTURE WORK

This paper presents the design of ATTENDAI, a webbased student attendance system combining face recognition, voice verification, and eye-blink liveness detection. It includes staged verification, role-based access, department-wise management, reporting, and a four-hour session mechanism. The design builds on the face-based architecture of the selected base paper while adding further verification and recordmanagement requirements.

Future work includes collecting a larger evaluation dataset, calibrating face and voice thresholds, measuring blinkdetection performance, strengthening biometric data protection, and testing in real classroom conditions. Other possible improvements include institutional single sign-on, cloud backup, mobile support, multilingual interfaces, and automated attendance alerts. These enhancements require appropriate privacy, consent, and security safeguards.

REFERENCES

1. O. A. R. Salim, R. F. Olanrewaju, and W. A. Balogun, "Class Attendance Management System Using Face Recognition," in Proc. 7th International Conference on Computer and Communication Engineering (ICCCE), 2018, pp. 93–96.
2. P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2001, pp. I-511–I-518.
3. T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037–2041, 2006.
4. S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 28, no. 4, pp. 357–366, 1980.
5. T. Cover and P. Hart, "Nearest neighbor pattern classification," IEEE Transactions on Information Theory, vol. 13, no. 1, pp. 21–27, 1967.
6. Python Software Foundation, "Python Documentation." [Online].
7. Available: <https://docs.python.org/3/>
8. OpenCV, "OpenCV Documentation." [Online]. Available: <https://docs.opencv.org/>