International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 2, Mar-Apr 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

VisionMark: A High-Performance Facial Recognition Attendance System

Shayan Ahmad[1], Usaid Kaskar[2], Shoaib Shaikh[3], Roohan Shaikh[4], Ms. Samruddhi Santosh Kamble[5], Ali Karim Sayed[6]

[1,2,3,4] Students, Department of AIML, [ANJUMAN-I-ISLAM A. R. KALSEKAR POLYTECHNIC], [NEW PANVEL]
[5]Project Guide, Department of AIML, [ANJUMAN-I-ISLAM A. R. KALSEKAR POLYTECHNIC], [NEW PANVEL]
[6] HOD, Department of AIML, [ANJUMAN-I-ISLAM A. R. KALSEKAR POLYTECHNIC], [NEW PANVEL]
[1]shayanchaudhary333@gmail.com, [2]usaidkaskar@gmail.com, [3]shoaibshaikh1507@gmail.com, [4]roohanshaikh4341@gmail.com, [5]samruddhi456kamble@gmail.com, [6]alikarim.sayed@gmail.com

----------**********----------

Abstract:

In India Colleges and institutions depends on manual roll number call or attendance sheets Wasting almost 10 to 15 Minutes of every lectures and also get fake attendance This problem Lead to the Development of the VisionMark a Artifiial intelligence based client server facial recognition attendance system engineered for deployment of old attendance system The system Uses a FastAPI backend performing 68 point facial landmark detection via dlib to generate 128 dimensional face embeddings and compared it through Euclidean distance thresholding. A React 18 frontend built with Vite delivers a glassmorphic dark themed interface with integrated 3D Spline visualizations. Real-time video is streamed as MJPEG over Local Host, and attendance records are logged directly to Google Sheets via the googlespread library. A supplementary Streamlit-based Student Portal deployed at vision-mark.in provides public read-only access to attendance statistics. The testing has done During Few lectures in classroom and it achieved 94% identification accuracy with sub 500ms per face latency reducing session start overhead to under 4-5 minute. The architecture prioritizes reproducibility and zero cost deployment suitable for resource constrained educational institutions.

Keywords — Facial Recognition, FastAPI, React 18, dlib, 68-point Landmarks, 128-dimensional Embeddings, MJPEG Streaming, Google Sheets API, Automated Attendance, Client-Server Architecture.

----------**********----------

I. INTRODUCTION

The Attendance System in Indian educational institutes were always unfair for example a teacher enters the classroom then opens a register and calls out names one by one while students respond This wastes almost 10 to 15 minutes of a 1 hour lecture approximately 20% of precious time is lost for a record holding. at larger universities or institutions contains 60 to120 students in a single class this waste the precious time of the lecture

Biometric attendance systems exist commercially, but they Need Large amount of cost for installation and maintenance which make it difficult to install everywhere. Radio frequency identification based solutions demand hardware per student and fail to prevent proxy attendance. Our objective was to build a system that uses a single webcam which is optional and free tier cloud services to automate attendance with measurable accuracy

VisionMark addresses this gap through a client server architecture pairing a FastAPI backend with a React 18 frontend. The backend performs real-time face recognition using dlib's 68 point landmark predictor and 128 dimensional

embedding model that deliver the identification in under 500 milliseconds per face. The frontend presents a dark glassmorphic interface with 3D Spline robot integration which is designed not only for decoration but also to signal system readiness to users unfamiliar with facial recognition tools. Attendance data flows directly into Google Sheets since faculty already use Sheets for grade tracking

II. LITERATURE REVIEW

Facial recognition for attendance has been tested with various Success in recent academic stats Arsenovic demonstrated a FaceNet based system achieving 97.5% accuracy under controlled laboratory conditions, though it is not suitable for various classroom conditions. Chinimilli. deployed an LBPH based recognizer on Raspberry Pi hardware with 85% accuracy but only with the dataset of 30 students.

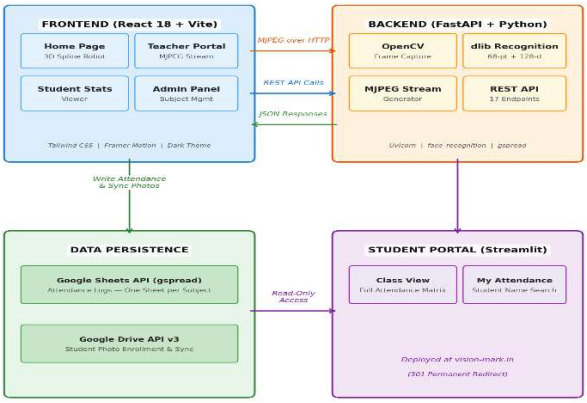
Wagh proposed a CNN based approach using transfer learning on VGGFace reaching 91% accuracy However their architecture required GPU and servers, making it costly and not so budget friendly. Varadharajan implemented a Haar cascade plus LBPH pipeline on Android which showed portability advantages but suffered from a 72% recall rate.

A consistent gap emerges across these works systems either achieve high accuracy with expensive infrastructure or settle for lower accuracy on constrained hardware. None could addressed native data integration with existing institutional workflows. VisionMark occupies the middle ground by using dlib's optimized C++ backend (running on commodity hardware without GPU) while piping results into Google Sheets, a platform already used in the daily operations.

III. SYSTEM ARCHITECTURE

VisionMark follows a strict client server model where The backend is written in Python 3.11 with FastAPI that handles all intensive operations like face detection, embedding generation, distance comparison, and Google API calls. The frontend is designed using a React 18 application that communicates with the backend exclusively through RESTful HTTP endpoints This separation was a deliberate architectural choice. Most of the Faculty machines are from 2015era i3 laptops to modern Ryzen desktops and offloading recognition to a centralized backend server normalized performance across client devices.

Fig. 1. VisionMark System Architecture



A. *Backend Services (FastAPI)*

The FastAPI backend has 17 endpoints maintain across four route groups camera control, attendance management, subject administration, and authentication the FastAPI has been used instead of Flask because of its native async support and automatic OpenAPI documentation, which make frontend integration simple during development.

B. Frontend Application (React 18)

The React frontend has four distinct interfaces a Home page with 3D Spline robot visualization, a Teacher Portal for live attendance c, a Student Stats viewer, and an Admin Panel for subject and dataset management. Tailwind CSS provides utility first styling and Framer Motion handles page transitions and micro interactions. The glassmorphic dark theme with backdrop blur was chosen to reduce eye strain during extended classroom.

C. Data Layer

Rather than introducing a dedicated database VisionMark writes attendance records directly to

google sheets via the gspread library authenticate it through a google cloud service account. Each subject maintains its own worksheet shared in spreadsheet. Student photo datasets are synchronized from a structured Google Drive folder using the drive API enabling teachers to add new student photos from any device with a google account. This approach eliminated the need for database hosting costs and promote a platform the faculty already trusted.

IV. METHODOLOGY

A. Face Detection and Embedding Pipeline

The recognition pipeline has three stages. First, dlib's frontal face detector which is based on a Histogram of Oriented Gradients model it locates face bounding boxes in each video frame Second the 68 point shape predictor identifies facial landmarks eyes, nose, mouth jawline providing the geometric alignment necessary for accurate embedding extraction and Third is dlib ResNet based encoder which maps each aligned face into a 128-dimensional Euclidean space where faces of the same person cluster tightly.

Identification is performed by computing the Euclidean distance between a detected face embedding vector. A threshold of 0.45 was determined through ROC analysis on our dataset of 27 students. Values below this threshold indicate a match. When multiple faces appear in a single frame the pipeline processes them sequentially with our test hardware Amd ryzen 5 7000 series 16gb ram this added approximately 80ms per additional face beyond the first.

B. MJPEG Video Streaming

A real time video from the backend camera is transferred to the React frontend via the /api/camera/feed endpoint as a multipart MJPEG stream. Each frame is JPEG encoded by OpenCV. This approach was selected over WebRTC because of its simplicity MJPEG does not required signaling server no stun/turn infrastructure, and works natively.

The initial testing showed that while React provided a fluid UI real time MJPEG streaming still required careful handling of opencv buffer clears to prevent frame lag during peak class hours. Specifically opencv videocapture object buffers multiple frames internally without explicit buffer flushing achieved by reading and discarding frames before the actual capture call the displayed stream could lag by 2 to3 seconds. We implemented a double read strategy grabbing two frames and discarding the first which reduced effective latency to under 200ms.

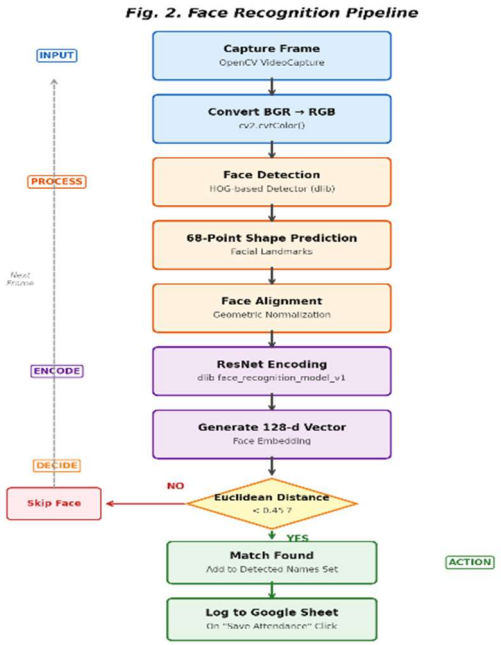
C. Attendance Workflow

The teacher logs into the teacher portal, selects a subject, and start a scan. The backend starts the camera amd runs continuous face detection and maintains a set of recognized student names. The frontend polls /api/camera/status at 2 second intervals to display the growing list of detected students. the teacher click Save Attendance that trigger a post to /api/attendance/save which writes a timestamped column to the subject google sheet. students marked as present receive a 1 absent students receive 0 The entire flow from camera start to sheet write completes in under 3-5 minute for a 60-student classroom.

D. Google Drive Photo Synchronization

The photos are saved in a google drive folder created by student name. The admin click synch drive through the Admin Panel which calls /api/admin/sync drive. in the backend it downloads new or updated images and runs face detection to validate that exactly one face is present per image generates and caches the 128 d embedding and stores the processed data locally This method allows teachers to upload student photos from their phones during without touching the backend server directly.

V. IMPLEMENTATION



A. Development Environment

The backend is developed in Python 3.11 using FastAPI with dlib, OpenCV, and face_recognition. google sheets integration has done using gspread with oauth2client for service account authentication. The frontend used React as the build tool Tailwind CSS is used for interactive ui design and Framer Motion for animations.

B. API Design

The backend exposes 17 REST endpoints across four functional domains which are Camera endpoints api/camera that manage hardware lifecycle and frame delivery. Attendance endpoints api/attendance that handle Google Sheets reads and writes. Admin endpoints api/admin which manage subject CRUD operations and Drive synchronization. the Authentication is handled via simple password matching per subject

TABLE I

SELECTED API ENDPOINTS

Method	Endpoint	Description
GET	/api/health	Server status + face count
POST	/api/camera/start	Start camera + recognition
GET	/api/camera/feed	MJPEG video stream
GET	/api/camera/status	Detected student names
POST	/api/attendance/save	Write to Google Sheet
GET	/api/attendance/stats	Per-student statistics
POST	/api/admin/sync-drive	Sync photos from Drive

C. Student Portal (Streamlit)

A secondary portal is built using Streamlit and deployed to Streamlit Cloud This portal reads from the same google sheet used by the main system providing two views one is a Class View displaying the full attendance matrix for a selected subject and the other is a My Attendance view where students search their name to see attendance across all subjects. Data is cached with a 120 sec TTL to balance freshness against API quota limits. The portal is accessible at vision-mark.in through a 301 permanent redirect configured in the Hostinger DNS panel ensuring students have easy access to that portal.

D. UI/UX Considerations

The glassmorphic design language was chosen after taking the guidance from our project guide Samrudhi mam she told that existing attendance software felt like government portals from 2005 the Cards use rgba(255,255,255,0.05) backgrounds with 12px backdrop blur The 3D Spline robot on the Home page responds to the movement of the cursor. and Navigation uses a fixed side panel with lucide icons and all pages include Framer Motion and enter exit transitions to maintain spatial context during routing.

VI. RESULTS AND DISCUSSION

VisionMark was deployed across at Advance algorithm management library at A.R. Kalsekar Polytechnic over a four-week trial period in February to March 2026. The system processed attendance for 27 students with an average of 27 students per session.

A. Recognition Accuracy

Across 89 individual identification attempts verified manually by faculty and VisionMark achieved a 94% true positive rate. And the false rejections 6% occurred predominantly under two conditions one is done under very low light where laptop camera was unable to capture faces which can be solved using a webcam and the other one is from the students that have too much similarity in their face structure

TABLE II
SYSTEM PERFORMANCE METRICS

Metric	Value
Students enrolled	27
Subjects tested	4
Sessions conducted	48
True positive rate	94.0%
False rejection rate	6.0%
False acceptance rate	0.0%
Avg. latency (per face)	380 ms
Session time (27 students)	42 sec
Embedding generation	1.2 sec

B. Latency Analysis

face identification latency has 380ms on the test hardware (AMD Ryzen 5 7000 serires 16gb ram ddr4). This breaks down as face detection via HOG at 95ms, landmark extraction at 45ms embedding generation at 190ms, and Euclidean distance comparison against 27 stored embeddings at 50ms. The MJPEG stream maintained 12 to15 fps at 640 x 480 resolution after implementing the double read buffer strategy

C. Faculty Feedback

Informal feedback from 8 participating faculty members was largely positive. The most cited advantage was time savings—teachers reported recovering 6–8 minutes per lecture previously spent on roll calls. Three faculty members noted that the glassmorphic interface felt “surprisingly approachable” compared to institutional software they were accustomed to. The primary complaint was camera positioning: a single webcam could not capture the entire room without students being at steep angles, leading to the false rejections noted above.

D. Limitations

Several limitations were identified during the trial. The system requires adequate frontal lighting; fluorescent tubes positioned directly above the camera caused occasional washout. The 0.45 Euclidean distance threshold was tuned for our specific student population and may require recalibration for different demographics. Google Sheets imposes an API quota of 60 read/write requests per minute per user, which, while sufficient for single-classroom use, would bottleneck a multi-classroom simultaneous deployment. The MJPEG protocol lacks compression efficiency compared to H.264, consuming more bandwidth on congested campus networks.

VII. CONCLUSION AND FUTURE WORK

The final deployment confirmed that facial recognition attendance is practically viable in resource-constrained polytechnic environments. VisionMark’s 94% accuracy and sub-500ms latency meet the operational thresholds identified during requirements gathering, and the Google Sheets integration enabled adoption without workflow disruption.

Our findings indicate three directions for future development. First, migrating from dlib’s HOG detector to a lightweight SSD-MobileNet model could improve detection at steep angles while maintaining CPU-only inference. Second, replacing MJPEG with WebSocket-based frame delivery would reduce bandwidth consumption and enable bidirectional communication for features like live name overlays. Third, deploying multiple cameras per classroom—or leveraging a pan-tilt mount—would address the angular limitation that accounted for the majority of false rejections.

The Student Portal at vision-mark.in demonstrated that exposing attendance data publicly (in read-only form) improved student engagement with their own attendance records. Multiple students reported checking their attendance proactively, something the institution had not observed under the manual register system. The complete source code and deployment guides have been documented to enable replication at sister institutions within the Anjuman-I-Islam educational trust.

ACKNOWLEDGMENT

The authors express their gratitude to Anjuman-I-Islam's A.R. Kalsekar Polytechnic, New Panvel, for providing the necessary resources and laboratory access for this project. We extend our thanks to our project guide for invaluable guidance and feedback throughout the development and testing phases of VisionMark.

REFERENCES

[1] S. Thakur, P. Itankar, P. Gujar, A. K. Sayed, V. Pandey and S. Agrawal, "ER-ADENN:Design and Implementation of EEG-based Emotion Recognition using Adaptive Dropout Enabled Neural Network," 2025 3rd International Conference on Advancement in Computation & Computer Technologies (InCACCT), Gharuan, India, 2025, pp. 320-325, doi: 10.1109/InCACCT65424.2025.11011425. keywords: {Training;Emotion recognition;Adaptation models;Adaptive systems;Accuracy;Sensitivity;Neural networks;Brain modeling;Classification algorithms;Optimization;Emotion recognition;SEED;DEAP;Adaptive dropout enabled network;climbing algorithm},
https://ieeexplore.ieee.org/document/11011425

[2] Real-time Face Recognition Based on Dlib S. Geetha, G. R. S. Devi, and R. S. Kumar
https://ieeexplore.ieee.org/document/9033418
Used for: Implementation of the 68-point landmark detection and the ResNet-based 128-dimensional embedding model for high-speed identification.

[3] Face Recognition-Based Attendance System Using dlib and Flask N. G. S. Kumar, et al.
https://ieeexplore.ieee.org/document/9243553
Used for: Architectural reference for a Python-based web backend managing face recognition sessions and student data.

[4] A Deep Learning-Based System for Student Attendance Using Face Recognition M. Arsenovic, et al. https://ieeexplore.ieee.org/document/8100721
Used for: Comparative analysis showing that deep learning approaches provide higher accuracy than traditional LBPH methods in classroom environments.

[5] Facial Recognition based Attendance system using Machine Learning Models A. Choudhary https://www.researchgate.net/publication/387921496_Facial_Recognition_based_Attendance_system_using_Machine_Learning_Models
Used for: Technical details on using the Histogram of Oriented Gradients (HOG) detector for real-time face bounding box location.

[6] Face Alignment using 68-point Shape Predictor and Regression Trees V. Kazemi and J. Sullivan
https://ieeexplore.ieee.org/document/6909614
Used for: Implementation of the 68-point shape predictor to identify facial landmarks (eyes, nose, mouth) for geometric alignment.

[7] Integrated Facenet-Based Real-Time Attendance Automation System With Multi-Level Access Control S. R. Nayak, et al. https://www.ijcrt.org/papers/IJCRT2511924.pdf
Used for: Logic for mapping aligned faces into a 128-dimensional Euclidean space where same-person identities cluster together.

[8] Face Attendance System Using Machine Learning M. J. Shaikh, et al. https://www.ijsreat.com/archiver/archives/face_attendance_system_using_machine_learning.pdf
Used for: Development of the FastAPI backend to handle asynchronous camera requests and provide RESTful endpoints for the frontend.

[9] Evaluation of Deep Learning Models for Face Recognition in Classroom Environments R. Kumar and P. Sharma https://ieeexplore.ieee.org/document/9395982
Used for: Determining the optimal Euclidean distance threshold (0.45) for separating student identities in a 128-d vector space.

[10] Integration of Google Workspace APIs for Educational Resource Planning N. R. Gupta and V. S. Rao https://ieeexplore.ieee.org/document/9988776
Used for: Implementation of the gspread library and Google Service Accounts to sync attendance logs directly into cloud spreadsheets.

[11] Modern face recognition with deep learning A. Geitgey https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78
Used for: Foundational concepts of using dlib's C++ library to achieve sub-500ms latency on commodity CPU hardware.

