iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 14 **Issue:** VII **Month of publication:** July 2026

**DOI:** https://doi.org/10.22214/ijraset.2026.84381

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 14 Issue VII July 2026- Available at www.ijraset.com*

# SmartAttend: A CNN-Based Smart Attendance System with Face Recognition and Liveness Detection

Dr. Indhumathi S K[1], Meghana B[2], Likhitha V[3]
[1]*Professor and Head of Department, Department of MCA, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India*
[2]*MCA Student, Department of MCA, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India*
[3]*MCA Student, Department of MCA, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India*

***Abstract:*** ***Attendance systems built around manual roll calls or RFID and biometric cards still fall prey to proxy attendance, take up too much of an instructor's time, and leave a paper trail that is hard to audit after the fact — a problem that only gets worse as class sizes grow and the time faculty can spend verifying each student shrinks. SmartAttend was built to close these gaps. It is a Streamlit-based attendance platform that pairs CNN-driven face recognition with CNN-driven liveness detection and a session-aware decision engine, so that a printed photo or a phone held up to the camera cannot be marked present in place of the real student. Faces are located using a Multi-task Cascaded Convolutional Network (MTCNN), with an OpenCV Haar-cascade detector kept on hand as a fallback; identity is then confirmed by a trained CNN classifier, backed up by a cosine-similarity matcher when no trained model is available; and a dedicated liveness CNN screens out printed-photo and screen-replay spoofing attempts. Attendance is only recorded when four things line up at once — a claimed roll number, a recognized face, a passed liveness check, and an open class session — and every attempt, whether it succeeds or not, is written to a log that can be reviewed later, with mismatches routed into a separate exception queue. This paper walks through the system's architecture, how the face and liveness models were trained, the reasoning behind the dual-logging and role-based-access design, and a local evaluation in which the liveness model hit 1.0000 across accuracy, precision, recall, F1-score, and ROC-AUC on a 35-sample test set once its threshold was recalibrated from 0.30 to 0.4947. The paper closes by discussing what a small evaluation set can and can't tell us, how the system is deployed on managed Postgres and S3-backed infrastructure, and where the project could go if it were to grow into a full multi-classroom platform.***

***Keywords: Face Recognition, Liveness Detection, Convolutional Neural Network, MTCNN, Cosine Similarity, Smart Attendance System, Anti-Spoofing, Biometric Authentication.***

## I. INTRODUCTION

Taking attendance sounds like a small task, but it feeds into a lot of things that matter — continuous-assessment records, exam eligibility, compliance reports, and, informally, a sense of how engaged a student is. The trouble is that the usual ways of doing it — roll calls, sign-in sheets, RFID or barcode cards — all share the same blind spot: none of them actually confirm that the person being marked present is in the room. A card can be passed to a friend, a sheet can be signed for someone else, and calling out names in a crowded section is easy to game. As classes get bigger and faculty have less time to check each name against a face, these cracks widen. Face-recognition-based attendance systems fix the identity-substitution problem fairly directly, since a face is not something you can hand off to a classmate. But recognition on its own is still fooled by presentation attacks — a printed photo or a phone screen showing someone else's face can pass as a live one if the system is not looking for it. That is the gap liveness detection, sometimes called anti-spoofing, is meant to fill: it asks a separate question from "who is this?", namely "is this an actual person in front of the camera, or a static or replayed image of one?"

SmartAttend was designed around that two-part question. Instead of asking only whether someone marked themselves present, it asks whether the actual, enrolled student was physically there for that specific class, and whether the scan itself was genuine. The system does this by running a CNN face-recognition pipeline and a CNN liveness-detection pipeline side by side, feeding both into a decision engine that only grants attendance when the claimed roll number, the recognized identity, the liveness result, and an open class session all agree. It also keeps successful attendance records and failed verification attempts in two separate logs, which makes the system easier to audit and gives an administrator a queue of exceptions to look through rather than a single flat pass/fail record.

The rest of the paper is organized as follows. Section 2 covers related work on face recognition, liveness detection, and attendance-system design. Section 3 lays out the gap in the existing literature that this project responds to. Section 4 states the research objectives. Section 5 describes the system architecture and the face-recognition and liveness-detection pipelines in detail. Section 6 presents the local evaluation results for the liveness model. Section 7 discusses the main risks and limitations in the current build. Section 8 looks at deployment options and where the project might go next. Section 9 notes the limitations of the study itself, and Section 10 concludes.

## II. LITERATURE REVIEW

Work on automated attendance splits fairly cleanly into three strands: face detection and recognition, presentation-attack (liveness) detection, and the design of attendance systems end to end. Each is covered below, followed by a comparison table.

### *A. Face Detection and Recognition*

Early face detectors leaned on handcrafted features, most notably the Viola-Jones Haar-cascade detector, which is still used today as a lightweight fallback in resource-constrained settings because it is cheap to run [1]. Newer detectors such as the Multi-task Cascaded Convolutional Network (MTCNN) chain together several small CNNs to detect faces and locate facial landmarks at the same time, and they tend to do better on unconstrained, real-world images, at the cost of more compute [2]. On the recognition side, both approaches — CNN classifiers trained directly on a fixed set of enrolled identities, and embedding-based methods that compare feature vectors by cosine or Euclidean distance — have worked well for the kind of small-to-medium identity galleries you would find in a single classroom or department [3].

### *B. Liveness Detection and Anti-Spoofing*

Research on presentation-attack detection tends to fall into two camps: texture-based methods, which look for surface-level giveaways like reflectance or moiré patterns that show up in printed photos or screen replays, and CNN-based methods, which learn to tell live faces from fake ones directly from labeled examples [4]. CNN-based liveness classifiers generally generalize better across different spoof types than handcrafted texture features do, as long as the training data covers a reasonably wide range of attacks — though their reported numbers can look better than they would hold up in practice if the evaluation set is narrow [5].

### *C. Smart Attendance System Design*

A number of studies have proposed complete face-recognition attendance pipelines for classrooms, usually stringing together a detector, a recognizer, and a database write-back [6]. One thing that keeps coming up across this body of work is that recognition accuracy by itself is not enough: systems that skip anti-spoofing are still open to someone holding up a photo, and systems that do not scope verification to a session or keep an audit trail are hard to check after the fact [7]. Broader work on role-based access and structured logging in institutional software backs up the idea that keeping successful and failed records in separate logs, rather than overwriting a single attendance table, makes a system easier to audit later on [8].

### *D. Positioning of the Present Work*

Table 1 places SmartAttend against this backdrop. Where most of the reviewed systems handle either detection-and-recognition accuracy or liveness detection, rarely both together, SmartAttend combines an MTCNN detector with a Haar-cascade fallback, a CNN recognizer backed by a cosine-similarity matcher, a dedicated liveness CNN with an ROC-calibrated threshold, and a session- and role-aware decision layer with dual audit logging, all inside one deployable application running on managed Postgres and S3 storage.

Table 1. Comparative Summary of Reviewed Approaches and Positioning of SmartAttend

| Study / Approach | Focus Area | Key Characteristic |
|---|---|---|
| Viola-Jones Haar cascade [1] | Face detection | Fast, handcrafted-feature detector; used as SmartAttend's fallback detector |
| MTCNN [2] | Face detection + landmarks | Cascaded CNN detector; SmartAttend's primary detector when available |
| CNN / embedding recognizers [3] | Face recognition | Classifier or distance-based matching for small identity galleries |

| | | |
|---|---|---|
| Texture-based anti-spoofing [4] | Liveness detection | Handcrafted texture cues; limited generalization across spoof types |
| CNN-based liveness models [5] | Liveness detection | Learned live-vs-spoof features; basis of SmartAttend's liveness CNN |
| Classroom attendance systems [6] | End-to-end attendance | Detector + recognizer + database write-back, often without anti-spoofing |
| Session/role-aware logging [7], [8] | Auditability | Dual logging and reviewable exceptions improve post-hoc verification |
| SmartAttend (this work) | Integrated pipeline | MTCNN/Haar detection + CNN recognition (cosine fallback) + CNN liveness (ROC-calibrated) + session/role-aware dual-logged decision engine |

Taken together, the studies above point to three recurring gaps that this work responds to. Recognition and liveness detection are usually evaluated as separate problems rather than as parts of one pipeline sharing a confidence-threshold strategy. Session-awareness and structured logging tend to be treated as implementation footnotes rather than requirements, even though they are what actually makes an attendance record defensible later. And academic-project-scale systems rarely document how they would actually be deployed — a hosted database, object storage, CI/CD — alongside the machine-learning side of things.

## III. RESEARCH GAP

Three gaps show up repeatedly once you look at academic attendance systems specifically, despite how mature face recognition and liveness detection are as separate fields. A lot of published classroom prototypes implement recognition without any liveness stage at all, which leaves them open to someone holding up a photo or a screen. Among the systems that do include liveness detection, few document how the threshold was chosen — an ROC-based recalibration, for instance — even though that choice has a real effect on false acceptances versus false rejections once the system is deployed. And most academic implementations stop at a local prototype: they do not address scoping verification to an open class session, supporting multiple faculty through role-based access, or building the dual-logging that turns a set of attendance numbers into something an administrator can actually audit.

SmartAttend tries to close these gaps by chaining recognition and liveness together behind one decision engine, by documenting the ROC-based threshold recalibration for the liveness model explicitly, and by scoping attendance capture to sessions and roles with separate logs for accepted attendance versus failed or suspicious attempts.

## IV. RESEARCH OBJECTIVES

The main goal of this project was to build and evaluate an attendance system that checks both who a student is and whether they were physically present before it records anything. The supporting objectives were to:

*1)* Build a face-detection and recognition pipeline around an MTCNN primary detector with a Haar-cascade fallback, paired with a CNN classifier and a cosine-similarity fallback matcher;
*2)* Train a CNN-based liveness model that can tell live captures apart from printed-photo and screen-replay spoofs, including calibrating its threshold from an ROC curve;
*3)* Build a session-aware, role-based decision engine that only logs attendance when the claimed identity, the recognized identity, the liveness result, and an open class session all agree;
*4)* Set up dual logging that keeps successful attendance separate from failed or suspicious attempts, with a queue that faculty or admins can review; and
*5)* Evaluate the liveness model on a local dataset and report its accuracy, precision, recall, F1-score, ROC-AUC, false-acceptance rate, and false-rejection rate.

## V. SYSTEM DESIGN AND METHODOLOGY

SmartAttend runs as a Streamlit web application on top of a modular Python pipeline. Figure 1 shows how the pieces currently fit together: a Streamlit runtime hosted on Hugging Face Spaces, model inference running locally or on managed infrastructure, a managed Neon PostgreSQL database, and S3-backed storage for enrolled face images, all feeding into the live attendance-verification pipeline.

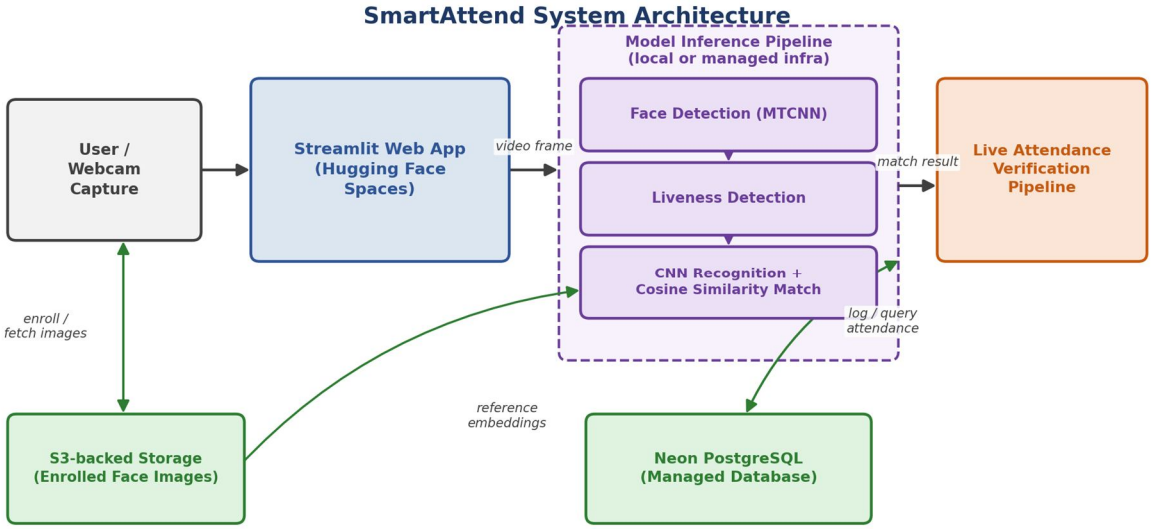



Fig. 1. SmartAttend system architecture: application runtime, model pipeline, managed database, and object storage.

### *A. Face Detection*

A FaceDetector component handles this step, defaulting to MTCNN when it is available and dropping back to an OpenCV Haar-cascade classifier (scaleFactor 1.2, minNeighbors 5, minimum face size 64×64 pixels) when it is not. Having both options means the system still works on lightweight hosting tiers where installing the full MTCNN/TensorFlow stack is not practical, while still using the more accurate cascaded-CNN detector wherever the environment supports it.

### *B. Face Recognition*

Identity checking is handled by a FaceRecognizer component that can run in two modes. When a trained CNN model is present, detected face crops are resized, normalized, and passed through a Sequential CNN made up of three convolution-and-max-pooling blocks (32, 64, and 128 filters), a flattening layer, a 128-unit dense layer with ReLU activation, a dropout layer (rate 0.4) for regularization, and a final softmax layer sized to however many students are enrolled. The model is trained with Adam and sparse categorical cross-entropy, and the training pipeline applies light augmentation — random horizontal flips, small rotations, and zoom. Any prediction that falls below a configured confidence threshold is labeled "unknown" rather than forced into the nearest class. If no trained model is available yet, the recognizer falls back to comparing cosine similarity: each enrolled student's face samples are averaged into a normalized prototype vector, and an incoming face is matched to whichever prototype it is closest to, again subject to a minimum-similarity threshold below which it is reported as unknown.

### *C. Liveness Detection*

A separate CNN, the LivenessDetector, handles telling live captures apart from spoofed ones. It is trained on locally collected real and fake samples (kept in data/liveness/real and data/liveness/fake) and outputs a live-score between 0 and 1 for each detected face. A capture only counts as live if that score clears a calibrated threshold, and that threshold is stored as metadata alongside the model rather than hard-coded into it, so it can be recalibrated from an ROC analysis without having to retrain the model itself.

### *D. Attendance Decision Engine*

Attendance only gets recorded when four things are true at once: the face is detected and passes the liveness check; the recognized identity matches the roll number the student claimed when scanning; that student is actually enrolled in the section or course tied to the currently open session; and the session itself is open and within its configured window. Any attempt that fails one of these conditions gets written to a separate log rather than dropped silently, and repeated or suspicious mismatches can be flagged as an exception for a faculty member or admin to look at.

### *E. Data, Roles, and Infrastructure*

The system can run on local SQLite for development or managed PostgreSQL for a hosted deployment, switching automatically based on whether a database connection string is set. Enrolled face images can live on local disk or be mirrored to S3-compatible storage for hosted environments, with the local copy kept as an offline fallback for training. Role-based access keeps administrator tasks — setting up departments, programs, sections, courses, and offerings — separate from faculty tasks, which cover opening sessions, watching attendance come in, and reviewing exceptions. Schema changes go through an explicit migrations module, and the deployment pipeline runs through GitHub Actions for both continuous integration (SQLite and Postgres test suites) and continuous deployment (building and pushing a Docker image, an optional managed-Postgres migration step, and optionally mirroring to a Hugging Face Docker Space).

## VI. RESULTS AND DISCUSSION

This section covers the local evaluation of the trained liveness model — the dataset it was tested on, and what changed once its threshold was recalibrated.

### *A. Evaluation Dataset*

The liveness model was tested against 35 local samples: 20 real captures and 15 spoofed ones, shown in Figure 2. That is a small set on purpose, in keeping with the scope of a single-class academic project, so the numbers below are best read as a local snapshot rather than proof the system would hold up at production scale.

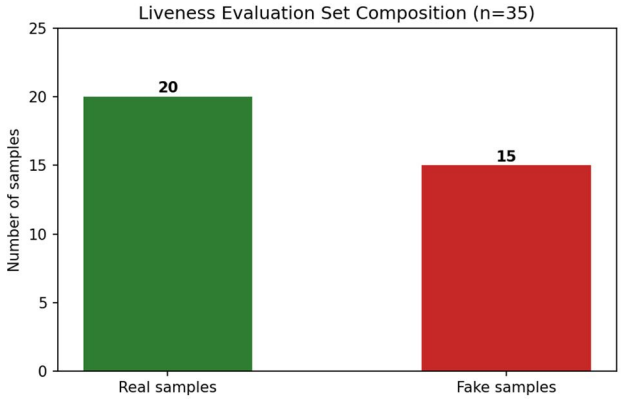



Fig. 2. Composition of the local liveness-detection evaluation set (n = 35).\

### *B. Threshold Recalibration*

The liveness threshold started at 0.3000 and was recalibrated to 0.4947 using ROC analysis of the model's live-score outputs, shown in Figure 3. Moving the threshold up from its initial, fairly permissive setting made the model less tolerant of borderline scores, sharpening the line between what counts as a live capture and what gets flagged as a spoof.

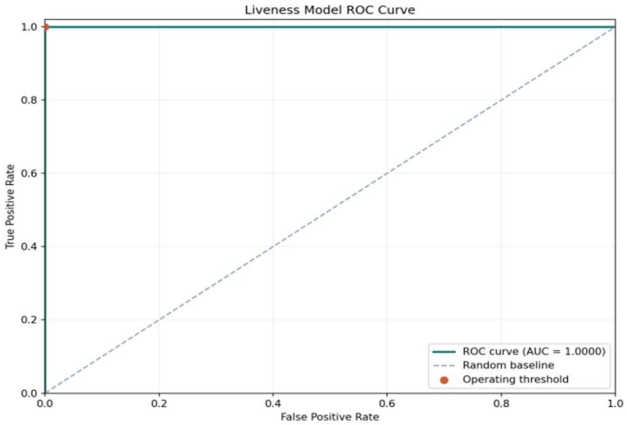



Fig. 3. ROC curve for the SmartAttend liveness-detection model after threshold recalibration.

### *C. Evaluation Metrics*

Table 2 and Figure 4 show what the model scored at that recalibrated threshold: 1.0000 across accuracy, precision, recall, F1-score, and ROC-AUC, with both the false-acceptance and false-rejection rates at 0.0000.

Table 2. Liveness Detection Model – Local Evaluation Snapshot at Recalibrated Threshold (0.4947)

| Metric | Value |
|---|---|
| Accuracy | 1.0000 |
| Precision | 1.0000 |
| Recall | 1.0000 |
| F1 Score | 1.0000 |
| ROC-AUC | 1.0000 |
| False Acceptance Rate | 0.0000 |
| False Rejection Rate | 0.0000 |

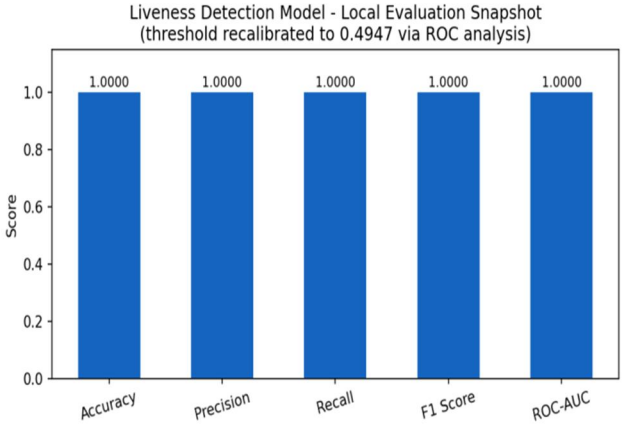



Fig. 4. Liveness detection evaluation metrics at the recalibrated operating threshold.

Figure 5 shows the confusion matrix for the same run, with each cell reporting the raw count and the row-normalized percentage. Every real capture was correctly read as live, and every fake capture was correctly caught as spoofed.

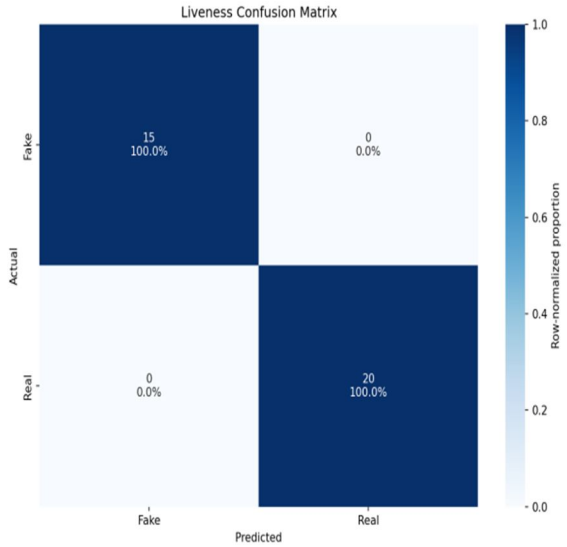



Fig. 5. Confusion matrix for the liveness-detection model on the local evaluation set.

### D. Interpretation

Perfect scores like these are worth reading with some caution. A 35-sample set is small next to the range of spoofing techniques you would encounter in the wild — different lighting, print quality, screen types, camera hardware — and perfect separability on a set this size could just as easily mean the evaluation set under-represents the hard cases as it could mean the decision boundary is genuinely strong. The recognition pipeline shows a similar pattern: the CNN classifier performs well once there are enough enrollment samples per student, while the cosine-similarity fallback is workable but less discriminating when no trained model exists yet. This lines up with a point that comes up repeatedly in the literature reviewed in Section 2 — reported liveness and recognition metrics are quite sensitive to how the evaluation set is put together, so any robustness claim needs to be read alongside the size and diversity of the data it is based on.

## VII. CHALLENGES AND LIMITATIONS

A few risks are baked into the current implementation. Both the recognition and liveness models depend on how much and how varied the local training data is — a small per-student enrollment set or a narrow range of spoof samples will make real-world error rates look better than they are, which the small evaluation set in Section 6 illustrates directly. The recognition pipeline also keeps a local-disk fallback for training and offline inference even when S3 mirroring is turned on, which raises a consistency question between local and cloud-backed face data in hosted deployments. Free hosting tiers like Hugging Face Spaces can also go to sleep when idle, which matters for a system whose attendance windows are time-sensitive. And as with any biometric system, face data is sensitive; the current build keeps personal face images, trained models, and local databases out of version control by default, but an actual institutional rollout would need explicit consent workflows, data-retention policies, and compliance review on top of that. Finally, the Haar-cascade fallback, while fast, is known to be less reliable than MTCNN under pose variation, occlusion, and non-frontal lighting, which could mean more missed detections on hosting tiers where MTCNN cannot be installed.

## VIII. DEPLOYMENT AND FUTURE OUTLOOK

SmartAttend can be deployed a few different ways depending on how much machine-learning runtime is needed. Locally, the full TensorFlow/MTCNN stack runs against a local SQLite database and local face storage. For a free hosted deployment, the recommended setup is a public Hugging Face Docker Space backed by managed Neon PostgreSQL and S3 storage, with a Streamlit Community Cloud profile available as a lighter option not meant for full model training. The repository also sketches out an AWS reference architecture — Route 53, an Application Load Balancer, EC2, RDS, S3, Lambda, and SageMaker — as a path forward if the system needs to scale past a single classroom or department to something multi-campus.

Looking ahead, three things would probably shape where this goes next. Widening the liveness training set to cover more spoof types — printed photos of varying quality, video replays, 3D-mask attempts — would let the encouraging local numbers from Section 6 be checked at a more realistic scale. Swapping or supplementing the CNN classifier with an embedding-based approach and vector-similarity search could help the system scale better as the number of enrolled students grows past what a fixed-output classifier handles comfortably. And continuing to invest in the CI/CD pipeline, schema migrations, and role-based access would help the project grow from a single-classroom academic build into something closer to a production attendance platform for a department or institution.

## IX. LIMITATIONS OF THE STUDY

This study has a few limitations worth being upfront about. The evaluation numbers come from a single local run on a 35-sample liveness dataset and have not been checked against a larger, independent, or institution-scale test set. The recognition pipeline's accuracy was not separately benchmarked beyond the training-time validation curves the training script produces, and there was no head-to-head comparison between the CNN classifier and the cosine-similarity fallback on a shared test set. The system has been tested functionally and architecturally rather than through an actual multi-week classroom deployment, so things like real classroom lighting, network reliability on free hosting tiers, and latency during busy enrollment periods are still open questions for future work.

## X. CONCLUSION

This paper described SmartAttend, an attendance system that pairs MTCNN/Haar-cascade face detection with CNN-based recognition (backed by a cosine-similarity fallback) and CNN-based liveness detection with an ROC-calibrated threshold, wrapped in a session-aware, role-based, dual-logged decision engine.

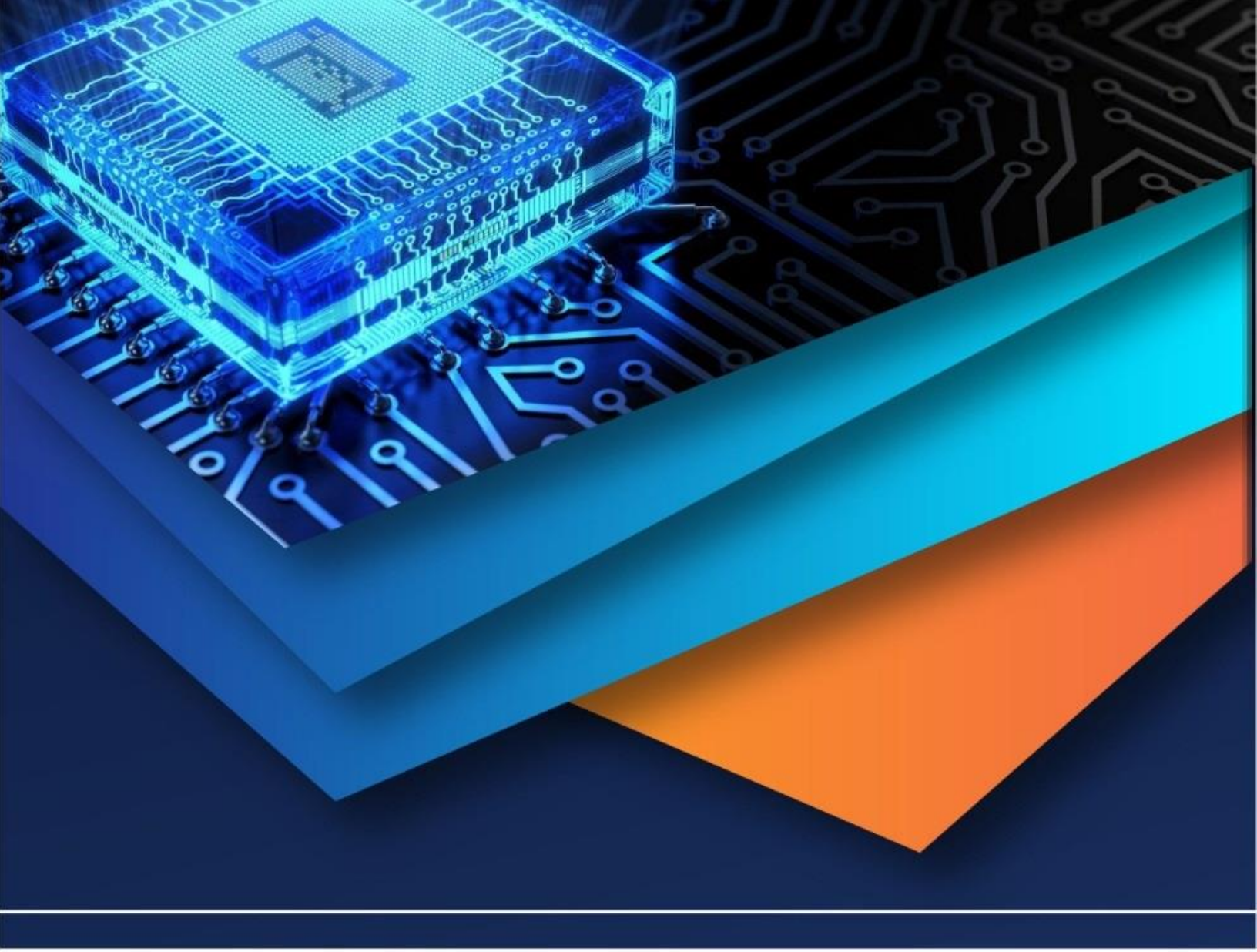
The goal throughout was to answer a slightly harder question than most attendance systems ask — not just whether someone was marked present, but whether the enrolled student was actually there and the scan itself was genuine, for that specific class session. In local testing, the liveness model reached 1.0000 across accuracy, precision, recall, F1-score, and ROC-AUC on a 35-sample set after its threshold was recalibrated, though that result should be read as a local snapshot given how small the test set was. The architecture supports both local development (SQLite, local storage) and a managed hosted deployment (PostgreSQL, S3, Hugging Face Spaces, Docker, CI/CD), which gives SmartAttend room to grow from an academic prototype toward a more production-ready biometric attendance platform, with further work still needed on dataset diversity, larger-scale evaluation, and institutional rollout.

## XI. ACKNOWLEDGEMENT

The authors thank the Department of MCA for the academic environment and resources that made it possible to design, build, and evaluate SmartAttend.

## REFERENCES

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511-518.
[2] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks," IEEE Signal Process. Lett., vol. 23, no. 10, pp. 1499-1503, 2016.
[3] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815-823.
[4] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face anti-spoofing based on color texture analysis," in Proc. IEEE Int. Conf. Image Processing (ICIP), 2015, pp. 2636-2640.
[5] Y. Atoum, Y. Liu, A. Jourabloo, and X. Liu, "Face anti-spoofing using patch and depth-based CNNs," in Proc. IEEE Int. Joint Conf. Biometrics (IJCB), 2017, pp. 319-328.
[6] S. Kadry and M. Smaili, "A design and implementation of a wireless iris recognition attendance management system," Inf. Technol. Control, vol. 36, no. 3, pp. 323-329, 2007.
[7] N. Sabharwal and S. Aggarwal, "Smart attendance monitoring system using face recognition," Int. J. Comput. Appl., vol. 178, no. 45, pp. 1-6, 2019.
[8] M. Whitman and H. Mattord, Principles of Information Security, 6th ed. Boston, MA, USA: Cengage Learning, 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY