IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

Retina Face-Based Smart Unauthorized Access Detection with Anti-Spoofing and ESP32 Alert

Dr. Bhagya P.[1], Adiveppa Dundappa Mamadapur[2], Akshay Ramesh Acharya[3]

Basagoud B. Patil[4], Avanakumar[5]

Department of Electronics and Communication Engineering, Don Bosco Institute of Technology, Bengaluru, India[1,2,3,4,5]

Abstract: This paper presents a prototype smart access-control system integrating face identification, iris biometric verification, application-level access decisions, event logging, and ESP32-based visual alerts. A Zebron-ics USB webcam supplies image input to a computer running a React/Vite frontend and a Python FastAPI backend. The reported face-processing pipeline uses YuNet for face detection and SFace with Open Neu-ral Network Exchange (ONNX) model support for face representation and comparison. A separate iris-verification stage compares a presented iris sample with registered biometric templates; the reported train-ing source is the CASIA-Iris-Thousand dataset. The backend uses SQLAlchemy for relational data per-sistence. The exact database engine must be confirmed from the final deployment configuration before submission. Following the configured verification decision, the application can send a status command to an ESP32 over USB serial. The ESP32 firmware is reported to control a green authorized LED and a red unauthorized LED, with a buzzer for unauthorized outcomes. Values carried over from the existing project report are detection performance of 99.76% at intersection over union (IoU) ≥ 0.50, verification accuracy of 97.62% at threshold 0.3258 and 92.85% at threshold 0.8000, false acceptance rates (FAR) of 2.38% and 0.00%, false rejection rates (FRR) of 2.39% and 46.86%, anti-spoofing accuracy of 99.59%, and average decision latency of 16.92 ms. The dataset counts, evaluation protocol, exact anti-spoofing model and its position in the pipeline, and physical ESP32 trial counts were not available for independent reproduction. Therefore, these values are identified as previously reported results and are not presented as independently validated measurements.

Keywords: Biometric Access Control, Face Identification, Iris Verification, YuNet, SFace, ONNX, FastAPI, React, SQLAlchemy, ESP32, Anti-Spoofing.

1 INTRODUCTION

Access-control systems are used to protect laboratories, offices, equipment rooms, and other restricted areas. Passwords and access cards can be shared, misplaced, or copied. Biometric methods provide an additional way to assess whether a person corresponds to an enrolled identity. A practical biometric access system must combine image acquisition, identity processing, access-policy logic, persistent event records, and clear feedback to the user.

The project titled "Retina Face-Based Smart Unauthorized Access Detection with Anti-Spoofing and ESP32 Alert" combines face identification and iris biometric verification. The reported face-processing stage uses YuNet for face detection and SFace with ONNX model support for face representation and comparison. A separate iris-verification stage compares captured iris information with registered tem-plates. The reported system components include a Zebronics USB webcam, a React/Vite user interface, a FastAPI backend, a SQLAlchemy-backed relational database, and an ESP32 connected to the computer through USB serial communication.

The supplied project title is retained as requested. However, the implementation details available for this paper identify YuNet as the face detector; they do not establish that the RetinaFace detector is implemented. The second biometric stage described in the available project material is iris verification, not retinal-image recognition. Accordingly, the implementation described and evaluated in this paper is the reported YuNet, SFace, and iris workflow. The title should not be interpreted as evidence that RetinaFace inference or retinal-image acquisition has been implemented.

1.1 Problem Statement

Manual or card-based access systems can be vulnerable to credential sharing and may provide limited iden-tity assurance. A further challenge is coordinating biometric decisions with access logging and physical indicators. The project addresses the need for a camera-based prototype that checks a possible identity, ver-ifies iris information, records access events, and communicates the resulting decision to a microcontroller. The system should also distinguish a biometric rejection from technical failures such as an unavailable camera, an unusable image, or a communication error.

1.2 Objectives

- Capture a live image or image sequence using a Zebronics USB webcam.
- Detect a face and identify a possible registered user using the reported YuNet and SFace pipeline.



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511230

- Perform iris biometric verification against registered templates.
- Apply the configured access decision only after the required verification and security checks have been evaluated.
- Support registration, verification, check-in/check-out, and access-event monitoring through a web interface.
- Store application and access records through the SQLAlchemy database layer.
- Communicate authorized and unauthorized outcomes to an ESP32 through USB serial and control the corresponding LEDs and buzzer, subject to the deployed firmware and circuit.

2 RELATED WORK

Face detection and face recognition are distinct tasks. Viola and Jones introduced a cascade-based object-detection approach that influenced early real-time face detection systems [1]. YuNet is a compact face detec-tor designed for efficient inference [2]. SFace is a face-recognition method based on sigmoid-constrained hypersphere loss and supports face representation and similarity-based comparison [3]. Open Neural Network Exchange (ONNX) provides a model representation format intended to support interoperability among machine-learning frameworks and runtimes [4].

Iris recognition is a separate biometric modality based on patterns in the iris region. Daugman described an influential iris-recognition approach based on phase information and iris codes [5]. The CASIA iris databases have supported research in iris recognition, including CASIA-Iris-Thousand [6]. The CASIA-IrisV4 collection includes several subsets, one of which is CASIA-Iris-Thousand [6]. This dataset-level description does not establish how many images or subjects were used in this project; the experimental record must identify the actual subset files, subject count, preprocessing, and split.

Presentation-attack detection seeks to identify attempts to deceive a biometric sensor, for example by pre-senting a photograph or a display. ISO/IEC 30107-3:2023 specifies testing and reporting principles for bio-metric presentation-attack detection [7]. The exact anti-spoofing model, attack categories, threshold, and its position in the current access pipeline were not identified in the supplied project report. ISO/IEC 30107-

3:2023 provides a framework for testing and reporting presentation-attack detection, but citing the standard does not establish that this prototype has been evaluated under it [7]. This paper therefore describes anti-spoofing only as a reported system function and does not assign an unverified model architecture or claim a specific integration sequence.

FastAPI supports Python API development [8], React provides a component-based interface [9], SQLAlchemy provides Python database tools [10], and Espressif documents the ESP32 microcon-troller platform and its interfaces [11]. These technologies support application and hardware integration but do not independently establish biometric accuracy.

3 SYSTEM OVERVIEW AND ARCHITECTURE

The prototype is described in terms of image acquisition, frontend interaction, backend processing, database persistence, and hardware-alert output. The frontend sends image data and user actions to the backend through API requests. The backend coordinates face detection, face identification, iris verification, config-ured security checks, access-policy evaluation, and database operations. It can then send a status command to the ESP32 through a USB serial connection. Figure 1 presents a conceptual view of the reported com-ponents. The anti-spoofing stage is shown as a configured security function because its exact model and execution point were not established in the available implementation description.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

Figure 1: Conceptual architecture of the reported face–iris access-control prototype. The dashed anti-spoofing note records a reported system function whose model and execution point require confirmation from the implementation.

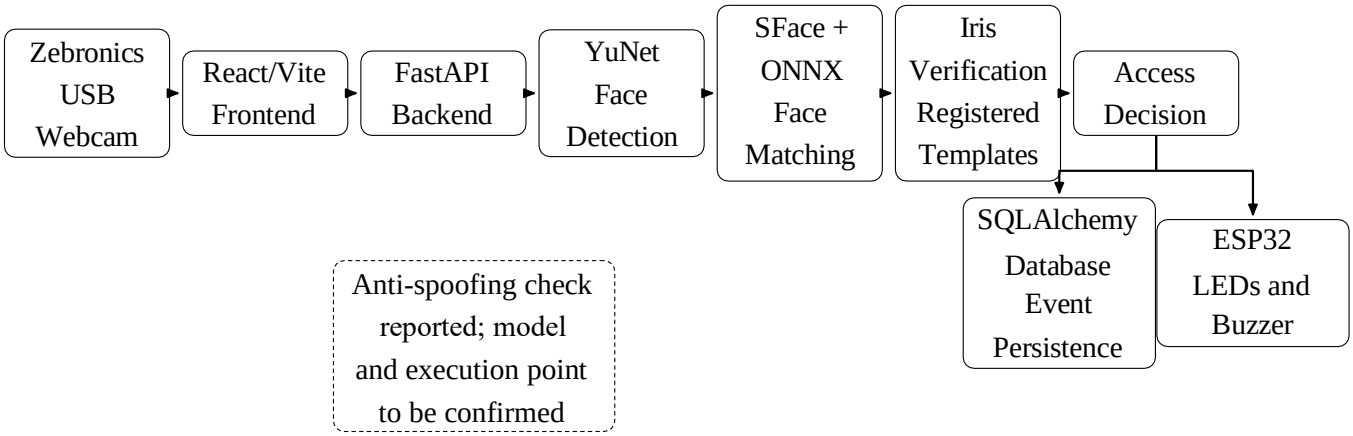
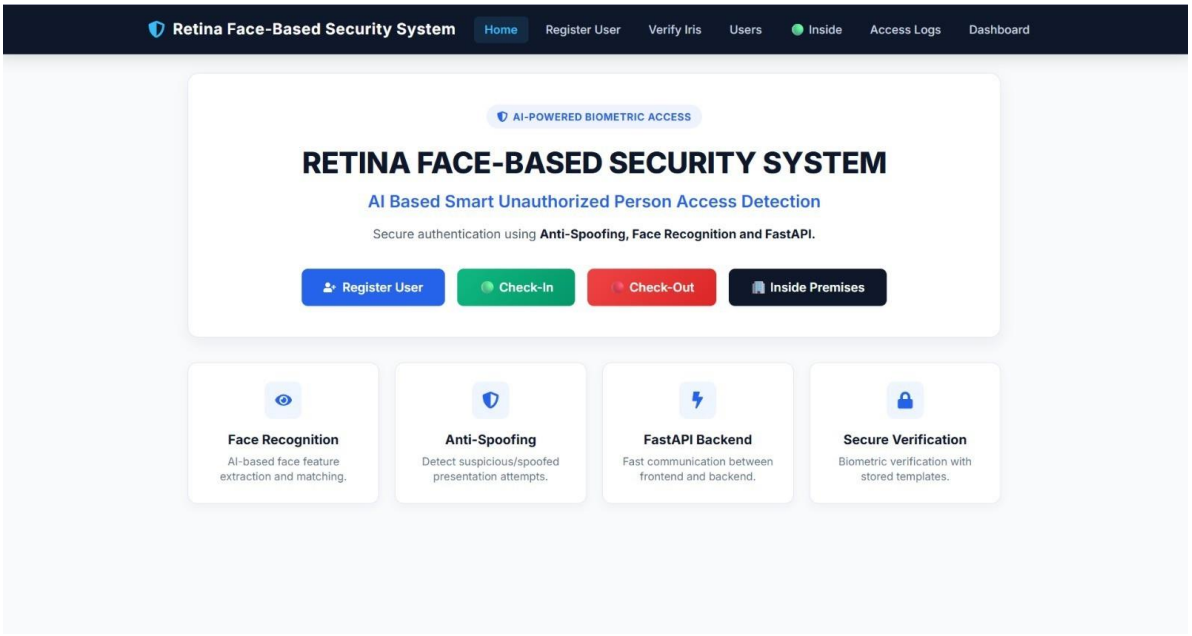


Figure 2: Application Dashboard for User Access Management.



3.1 Hardware Connections

Table 1 summarises the reported and indicative hardware connections. Exact GPIO numbers, resistor val-ues, buzzer-driver details, and serial command strings were not confirmed in the supplied project material and must be checked against the final firmware and physical circuit.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

Table 1: Indicative ESP32 Output Wiring

Signal	Connection	Note
USB webcam	PC USB port	Image input to the application.
ESP32 USB	PC USB port	Serial communication and board power, as supported by the board.
Green LED	Configured ESP32 GPIO through a resis-tor to LED anode; cathode to ground	Pin number must match firmware. Pin number must match firmware. Confirm voltage and current ratings.
Red LED	Configured ESP32 GPIO through a resis-tor to LED anode; cathode to ground	
Buzzer	Configured output, with a suitable driver if required	
Ground	Common circuit ground	Follow board and component datasheets.

4 METHODOLOGY

4.1 Image Capture and Frontend Request

The webcam supplies images to the computer. The React/Vite frontend captures or selects the image se-quence used by the verification workflow and sends the request to a FastAPI endpoint [8]. If the camera fails or the image is unusable, the request should return a technical-error or retry state rather than classify the person as an unauthorized identity.

4.2 Face Detection and Identification

OpenCV and YuNet are reported to locate the face region. The detected face is processed through the configured SFace model in ONNX format [3, 4] to generate a representation used for comparison with registered face data. The best candidate, comparison score, and configured threshold are evaluated ac-cording to the implementation. YuNet performs face detection; it is not an iris detector or a retinal-image recognition model.

4.3 Iris Biometric Verification

After the face stage proposes a registered identity, the iris stage evaluates the presented iris sample against the enrolled iris template or templates associated with that user. The iris model was reported as trained using images from CASIA-Iris-Thousand [6]. The available project information does not confirm the number of identities, image count, exact training/testing split, or all preprocessing settings; these details are therefore not asserted here.

4.4 Anti-Spoofing and Access Decision

The project report includes an anti-spoofing stage and a reported anti-spoofing accuracy value. The avail-able evidence does not identify the attack presentation protocol or the number of bona fide and attack samples. However, the exact model, input representation, attack categories, decision threshold, and posi-tion of this stage in the running pipeline were not established in the material supplied for this paper. For this reason, the anti-spoofing component is treated as a configured security check whose detailed operation must be documented from the source code and model files before submission.

Let F denote successful face identification and I denote successful iris verification. Let S denote the result of any additional security check that is required and actually implemented. If no optional security check is configured, S is assigned the neutral value 1. A general access rule can be expressed as

$$A = F \wedge I \wedge S, \qquad (1)$$

where $A = 1$ means that access is authorized. Anti-spoofing contributes to S only if it is enabled in the deployed workflow. A technical failure, such as camera unavailability, malformed image, model-loading failure, or backend error, is a separate state and should not automatically be recorded as a confirmed identity mismatch.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

4.5 ESP32 Serial Alert

After the backend finalizes the access decision, it can send a configured command through USB serial. The ESP32 firmware [11] maps the received command to output states. An authorized command is intended to activate the green LED. An unauthorized command is intended to activate the red LED and buzzer according to the firmware. A successful serial write alone does not prove that a physical LED activated; hardware reliability must be measured through repeated end-to-end trials and recorded separately from software decisions.

4.6 Check-In, Check-Out, and Logging

The application supports access operations such as check-in and check-out. The backend validates the requested action, evaluates the verification result, and writes the resulting event through SQLAlchemy to the configured relational database. Event fields must reflect the actual database schema. Typical fields may include a user identifier, event type, timestamp, decision, and reason, but the availability of each field must be verified in the implementation.

5 ALGORITHMS

5.1 Biometric Access Workflow

The reported high-level workflow is summarised below. The order of any additional anti-spoofing opera-tion must be aligned with the actual implementation before the algorithm is treated as a precise description of the deployed system.

1. Receive an image or image sequence from the frontend API request.
2. Validate and decode the request. Return a technical-error state if the input is invalid.
3. Detect a face using YuNet. Return a retry or technical-error state if no usable face is detected.
4. Generate and compare the face representation using the configured SFace ONNX model and en-rolled face records.
5. If a possible registered identity is found, verify the corresponding iris sample against the enrolled iris template.
6. Evaluate the additional security checks that are actually enabled in the deployed configuration. In-clude anti-spoofing only if that check is present in the running pipeline.
7. Authorize access only when all required checks pass; otherwise apply the configured rejection pol-icy.
8. Record the supported access event and send the corresponding ESP32 status command, subject to the application and hardware communication rules.
9. Return the decision and relevant status to the frontend.

5.2 High-Level Pseudocode

The following pseudocode defines each intermediate result and separates technical failures from biometric rejection. It is a conceptual description and must be reconciled with the actual endpoint names, model calls, and serial commands in the source code.

```
FUNCTION process_access(request)

    image <- validate_and_decode(request)
    IF image is invalid THEN
        RETURN TECHNICAL_ERROR
    END IF

    face <- YuNet.detect(image)
```

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

```
    IF face is not found THEN
        RETURN RETRY_OR_TECHNICAL_ERROR
    END IF

    face_result, candidate <- SFace.identify(face, registered_faces)
    IF face_result is a technical failure THEN
        RETURN TECHNICAL_ERROR
    END IF
    IF face_result is rejected THEN
        record_rejection_if_supported()
        send_serial_command(UNAUTHORIZED)
        RETURN UNAUTHORIZED
    END IF

    iris_result <- verify_iris(request.iris_input, candidate.iris_template)
    IF iris_result is a technical failure THEN
        RETURN TECHNICAL_ERROR
    END IF

    security_result <- evaluate_enabled_security_checks(request)
    # Returns PASS when no optional security check is configured.
    IF security_result is a technical failure THEN
        RETURN TECHNICAL_ERROR
    END IF

    IF iris_result passes AND security_result passes THEN
        record_access_event_if_supported()
        send_serial_command(AUTHORIZED)
        RETURN AUTHORIZED
    ELSE
        record_rejection_if_supported()
        send_serial_command(UNAUTHORIZED)
        RETURN UNAUTHORIZED
    END IF

END FUNCTION
```

Here, evaluate_enabled_security_checks represents only the additional checks enabled in the de-ployed configuration. The pseudocode does not assert that a particular anti-spoofing model or execution order has been verified.

6 DATABASE DESIGN

The application uses SQLAlchemy for relational data persistence. The database engine must be confirmed from the final configuration because the available project materials have described different database setups. The exact schema was not included in the reviewed project report; Table 2 is therefore a conceptual entity view rather than a claim about exact table or column names.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

Table 2: Conceptual Data Entities

Entity	Purpose
Registered user	Stores account/profile information needed to identify an enrolled user.
Face template or reference	Associates registered face information with a user, according to the implemented storage method.
Iris template	Stores or references enrolled iris biometric data used for comparison.
Access event	Records supported check-in/check-out or access-decision events.
System configuration	May hold thresholds or operational settings if such a table exists.

Biometric templates and personal records should be access-controlled. Production deployment should consider encryption, least-privilege database credentials, secure backups, retention limits, and deletion of biometric data when no longer required. These are security recommendations, not claims that all controls are currently implemented.

7 IMPLEMENTATION DETAILS

The React/Vite frontend [9] is responsible for user interaction and sending API requests. FastAPI coor-dinates request validation and backend operations. OpenCV provides image-processing support, YuNet performs face detection, and SFace ONNX models support face comparison. The iris-verification stage uses a model reported to be trained on CASIA-Iris-Thousand images and enrolled templates. The config-ured relational database persists application records through SQLAlchemy. A serial connection from the PC to the ESP32 carries the resulting access status to the firmware-controlled LEDs and buzzer.

The exact anti-spoofing model, its input, and its placement in the access-decision pipeline must be con-firmed in the current codebase before a specific algorithm is named. The paper also does not assume that every image-processing stage uses the same camera frame unless the source code confirms this.

8 TESTING AND RESULTS

The values in Tables 3 and 4 are transcribed from the existing project report. Test-set sizes, numbers of genuine and impostor comparisons, threshold-selection method, class proportions, confusion counts, and evaluation scripts were not supplied for independent reproduction. They must therefore be treated as reported values, not independently verified results. In particular, the accuracy values cannot be reconciled with FAR and FRR without knowing the class counts and whether the metrics were computed on the same comparison set.

Table 3: Reported Verification Results at the Two Supplied Thresholds

Threshold	Accuracy	FAR	FRR
$\tau = 0.3258$	97.62%	2.38%	2.39%
$\tau = 0.8000$	92.85%	0.00%	46.86%

Table 4: Other Performance Values Reported in the Existing Project Report

Metric	Reported Value
Detection score at IoU ≥ 0.50 (metric definition not supplied)	99.76%
Anti-spoofing accuracy	99.59%
Average decision latency	16.92 ms

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

The reported values show a threshold-dependent change in the stated false-acceptance and false-rejection rates: at threshold 0.8000, FAR is reported as 0.00% and FRR as 46.86%. A zero observed false-acceptance rate on a finite test set does not prove that false acceptance is impossible. Accuracy, FAR, and FRR use different denominators; without the genuine/impostor sample counts and confusion counts, the reported

92.85% accuracy at threshold 0.8000 cannot be independently reconciled with the reported FAR and FRR. The operational threshold should be selected using a documented validation protocol and the security re-quirements of the deployment. The meaning of the reported detection score at IoU $\geq$ 0.50, the latency measurement boundary, and the denominators used for the reported rates should be documented before these values are treated as reproducible experimental findings. For clarity, FAR is the proportion of impostor comparison attempts incorrectly accepted, while FRR is the proportion of genuine comparison attempts incorrectly rejected. The sample counts and comparison protocol required to independently verify the reported percentages were not supplied. The standard defini-tions are FAR = $FP/(FP + TN)$ and FRR = $FN/(FN + TP)$ for a binary genuine/impostor protocol, where FP, TN, FN, and TP are the corresponding confusion-matrix counts. Overall accuracy is $(TP + TN)/(TP + TN + FP + FN)$. These quantities should be reported together with the actual counts and the test-set composition.

8.1 Functional Test Plan

Table 5 lists functional cases for final validation. It is a test plan; it does not imply that all cases have already been executed.

Table 5: Functional Tests to Document During Final Validation

Test Case	Expected Behaviour	Evidence to Record
Registered user, valid face	Access decision follows the configured authorization policy.	API result, database event, physical LED and iris state.
Known user, iris rejected	Access denied when the rejection is definitive.	Score, threshold, decision reason, LED/buzzer state.
Unregistered person	Identity rejected according to the implemented matching policy.	Candidate/match result and event record.
Camera unavailable	Technical error or retry state; not automatically an identity mismatch.	Error response and application status.
No usable face detected	Retry or technical-error response.	Detection result and status.
ESP32 disconnected	Hardware communication error is recorded or shown; physical activation is not claimed.	Serial error and hardware state.
Check-in/check-out	Valid action is stored only when permitted by application rules.	Database record and timestamp.

8.2 Application and Hardware Evidence

The successful software-verification screenshot displays “0 ACTIVATIONS (SAFETY ACTIVE)”. It sup-ports the displayed software verification outcome, but it does not demonstrate a physical ESP32 activation during that event.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511230

Figure 3: Successful software-verification screen.

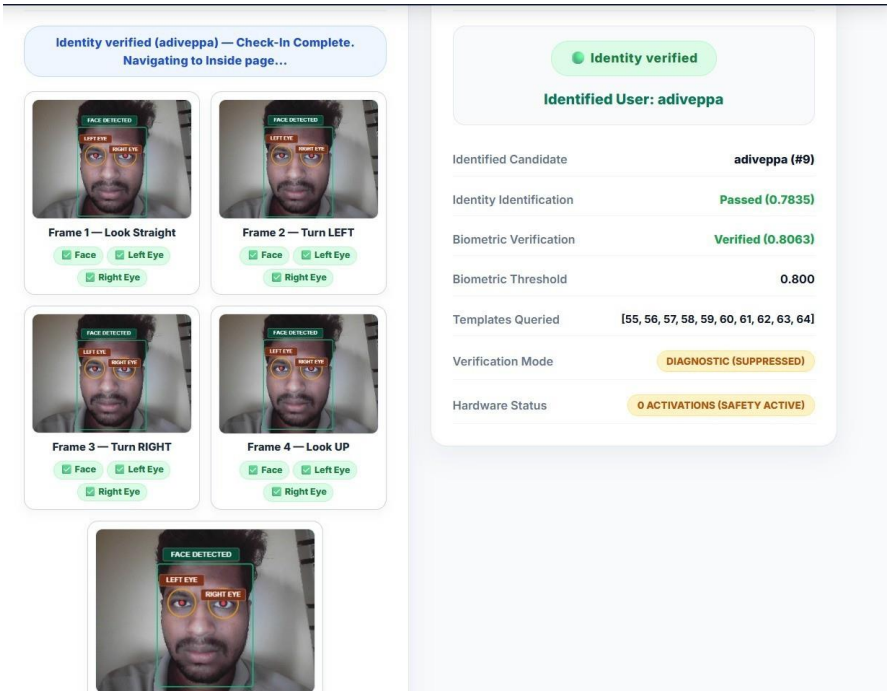
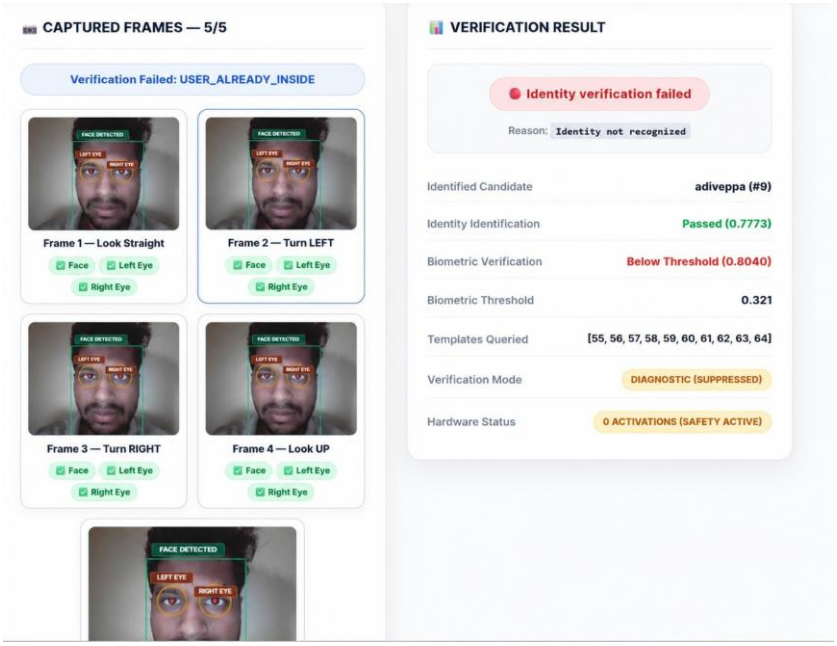


Figure 4: Example of a Failed Verification Screen.

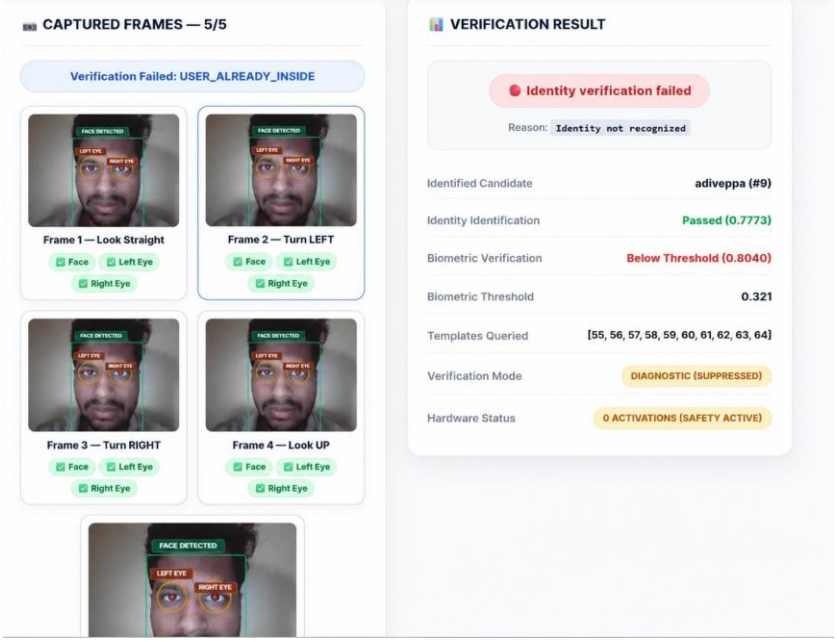


IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

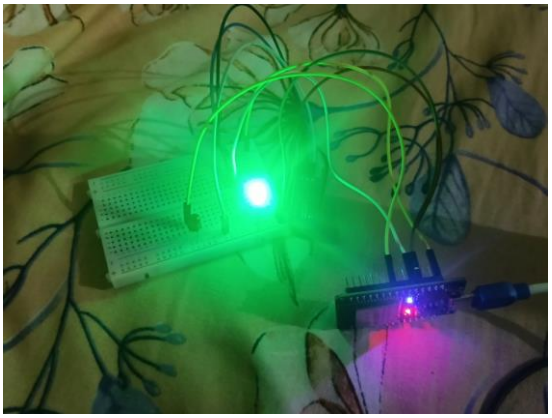
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

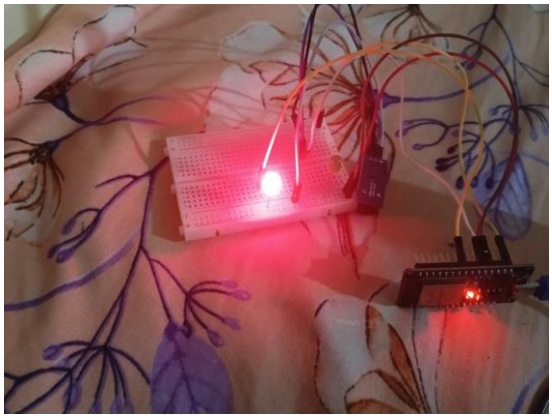
DOI: 10.17148/IJARCCE.2026.1511230



The photographs in Figures 5 and 6 show the green and red ESP32 LED states, respectively. They document visible indicator states, but the available material does not establish the number of repeated trials or link each photograph to a specific logged software event.

Figure 5: ESP32 Green LED State. Figure 6: ESP32 Red LED State.





9 LIMITATIONS AND FUTURE ENHANCEMENTS

The evaluation cannot be independently reproduced without dataset counts, identity distribution, train-ing/testing split, preprocessing details, comparison protocol, confusion counts, and the scripts used to cal-culate each metric. The exact anti-spoofing model and its placement in the running pipeline have not been confirmed in the supplied implementation summary. The database engine also requires confirma-tion because the paper draft and repository-level documentation have not been fully reconciled. Camera performance may vary with lighting, distance, pose, image quality, and occlusion. The reported FRR at threshold 0.8000 is high and may cause legitimate users to be rejected. The successful verification screen-shot also reports zero hardware activations, so software authorization and physical output activation must be evaluated separately.

Future work should include documenting the exact model files and preprocessing pipeline; defining sep-arate face-identification and iris-verification evaluation protocols; specifying the anti-spoofing model, at-tack types, and integration point; testing on an independently held-out dataset; reporting subject and im-age counts; calibrating thresholds using validation data; measuring end-to-end latency separately from model-inference time; conducting repeated ESP32 command-and-physical-output trials; and strengthen-ing biometric-data protection and serial-communication error handling.



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511230

10 CONCLUSION

This paper presents a camera-based access-control prototype combining face identification, iris biomet-ric verification, a React/Vite frontend, a FastAPI backend, the configured relational database persistence through SQLAlchemy, and ESP32-controlled visual alerts. The reported face-processing components are YuNet for face detection and SFace with ONNX model support for face comparison. A separate iris-recognition model is reported to use CASIA-Iris-Thousand images and registered iris templates. The orig-inal project title has been retained, while the implementation distinction is made explicit: the available implementation description does not establish RetinaFace inference or retinal-image recognition. The per-formance values have been retained from the existing project report and are identified as reported rather than independently reproduced. Before submission, the authors should verify the anti-spoofing implemen-tation, dataset counts, threshold protocol, database schema, GPIO assignments, and repeated physical alert results against the final code and hardware.

REFERENCES

[1]. P. Viola, M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," in Pro-ceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, 2001, pp. 511–518, doi: 10.1109/CVPR.2001.990517.
[2]. W. Wu, H. Peng, S. Yu, "YuNet: A Tiny Millisecond-Level Face Detector," Machine Intelligence Research, vol. 20, no. 5, 2023, pp. 656–665, doi: 10.1007/s11633-023-1423-y.
[3]. Y. Zhong, W. Deng, J. Hu, D. Zhao, X. Li, D. Wen, "SFace: Sigmoid-Constrained Hypersphere Loss for Robust Face Recognition," IEEE Transactions on Image Processing, vol. 30, 2021, pp. 2587–2598, doi: 10.1109/TIP.2020.3048632.
[4]. ONNX, "Open Neural Network Exchange Documentation." Available: https://onnx.ai/ (accessed Nov. 25, 2026).
[5]. J. Daugman, "How Iris Recognition Works," IEEE Transactions on Circuits and Systems for Video Technology, vol. 14, no. 1, 2004, pp. 21–30, doi: 10.1109/TCSVT.2003.818350.
[6]. Chinese Academy of Sciences, Institute of Automation, "CASIA Iris Databases." Available: https: //english.ia.cas.cn/db/201610/t20161026_169399.html (accessed Nov. 25, 2026).
[7]. International Organization for Standardization, ISO/IEC 30107-3:2023, Information Technology—Biometric Presentation Attack Detection—Part 3: Testing and Reporting, Geneva, Switzerland, 2023.
[8]. FastAPI, "FastAPI Documentation." Available: https://fastapi.tiangolo.com/ (accessed Nov. 25, 2026).
[9]. Meta Open Source, "React Documentation." Available: https://react.dev/ (accessed Nov. 25, 2026).
[10]. SQLAlchemy, "SQLAlchemy Documentation." Available: https://docs.sqlalchemy.org/ (ac-cessed Nov. 25, 2026).
[11]. Espressif Systems, "ESP32 Series Documentation." Available: https://docs.espressif.com/ projects/esp-idf/en/stable/esp32/ (accessed Nov. 25, 2026).