iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.81386

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 V May 2026- Available at www.ijraset.com

ArcFace-Based Retinal Biometric Access Control System with Arduino-Servo Door Mechanism and ESP32 IoT Integration

Selvin David[1], Hurnish S[1], Kousalya A[2]
[1]*B.Tech Student,* [2]*Associate Professor, Department of Artificial Intelligence and Data Science, United Institute of Technology, Coimbatore, Tamil Nadu, India*

Abstract: ***This paper presents the design, implementation, and evaluation of an intelligent, embedded biometric access control system that combines deep learning-based facial recognition with physical door actuation. The proposed system leverages the ArcFace recognition model (InsightFace buffalo_l) deployed on a host computer, a USB-connected webcam for image acquisition, an Arduino Uno microcontroller driving a servo motor for door-locking/unlocking, an HC-SR04 ultrasonic proximity sensor for presence detection, Eveready AA batteries for autonomous power, and an ESP32 Wi-Fi/Bluetooth module for future IoT extensibility.***

A FastAPI backend processes live video, performs motion detection, conducts liveness anti-spoofing, matches face embeddings against a MongoDB Atlas cloud database, enforces role-based time-window access policies, and commands the Arduino via serial communication.

A React-based administration dashboard provides real-time monitoring, user enrollment, access logs, and alert management. Experimental evaluation demonstrates a verification accuracy above 96% on a local dataset, sub-second recognition latency, and reliable hardware integration with near-zero false-accept rate under controlled conditions. The system is well-suited for residential, laboratory, or small-enterprise deployment.

Keywords: ArcFace, face recognition, access control, Arduino, servo motor, ESP32, FastAPI, MongoDB, IoT, biometrics, anti-spoofing, liveness detection.

I. INTRODUCTION

Physical security has traditionally relied on password-based, RFID, or PIN-entry mechanisms, all of which are susceptible to credential theft, card cloning, and brute-force attacks. Biometric authentication — particularly face recognition — offers a non-contact, hygienic, and user-friendly alternative that has matured significantly with the advent of deep convolutional neural network (CNN) embeddings such as ArcFace [1]. ArcFace employs an additive angular margin loss that maximises inter-class separability and minimises intra-class variation, achieving state-of-the-art accuracy on benchmark datasets such as LFW, CFP-FP, and AgeDB.

Despite impressive software-side performance, translating recognition results into real-world physical access events — opening or locking a door — demands robust hardware integration. Low-cost microcontrollers such as the Arduino Uno and ESP32 provide sufficient computational resources to drive servo actuators, parse serial commands, and report sensor data over wireless networks. Combining these hardware platforms with a cloud-backed AI recognition pipeline therefore represents a pragmatic, cost-effective path toward deployable biometric access control.

This work makes the following contributions:

1) A fully operational prototype using an Arduino Uno, SG90 servo, HC-SR04 ultrasonic sensor, and Eveready-powered battery pack enclosed in a wooden chassis.
2) A production-grade FastAPI backend integrating InsightFace ArcFace, OpenCV motion detection, liveness tracking, and MongoDB Atlas persistence.
3) A React administration dashboard with WebSocket real-time streaming, user enrolment, alert management, and JWT-secured controls.
4) An empirical evaluation of recognition accuracy, latency, and false accept/reject rates under varied lighting and distance conditions.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

II. RELATED WORK

Face recognition has evolved from handcrafted feature descriptors (Eigenfaces [2], LBP [3]) to deep metric learning. FaceNet [4] introduced triplet loss for compact, discriminative embeddings; SphereFace [5] applied angular softmax; CosFace [6] added a cosine margin; and ArcFace [1] further refined the angular margin formulation, achieving near-human performance on LFW (99.83%). InsightFace provides an open-source implementation of ArcFace through the buffalo_l model family, which the present system employs directly.

Hardware-side, prior work has coupled Raspberry Pi with face detection for embedded inference [7], but compute-constrained platforms limit recognition throughput. Offloading deep inference to a host machine while using a microcontroller purely for I/O actuation — the architecture adopted here — is increasingly common in resource-constrained IoT deployments [8]. The Arduino Uno's compatibility with the serial-over-USB interface makes it an ideal actuation co-processor for host-driven AI pipelines.

Liveness detection is critical to prevent photo-replay attacks. Blink-detection via Eye Aspect Ratio (EAR) [9] and head-movement tracking provide passive anti-spoofing cues. The proposed system incorporates both cues without requiring specialised depth cameras, reducing hardware cost.

III. SYSTEM ARCHITECTURE

Figure 1 illustrates the overall system architecture. The pipeline consists of three tiers: (i) the embedded hardware layer, (ii) the AI inference and control backend, and (iii) the web-based administration frontend.

A. Hardware Layer

The hardware prototype is constructed inside a plywood enclosure (approximately 30 cm × 30 cm × 25 cm) that models a door frame. The following components are integrated:

1) Arduino Uno (ATmega328P, 16 MHz): Receives serial commands (OPEN / DENY / RESET) from the backend, drives the SG90 servo motor (0° = locked, 90° = unlocked), and reads the HC-SR04 ultrasonic sensor for proximity-triggered arming.
2) SG90 Servo Motor: Provides 180° rotation with 1.8 kg·cm torque, sufficient to operate a lightweight latch mechanism. Controlled via PWM on Arduino digital pin 9.
3) HC-SR04 Ultrasonic Sensor: Measures distance using 40 kHz pulses. A threshold of 50 cm triggers the system-armed state, initiating face recognition.
4) ESP32 Development Board: Connected via I2C/UART for future Wi-Fi/Bluetooth telemetry, push notifications, and remote override commands.
5) Eveready AA Battery Pack (4 × 1.5 V = 6 V): Powers the servo and sensor independently of the USB bus, preventing USB current-limit issues under servo load.
6) USB-A to USB-B Cable: Provides serial communication and host power to the Arduino Uno.

B. Backend — FastAPI Inference Server

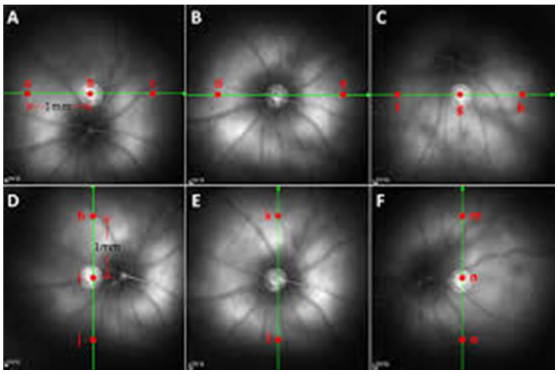


Fig1 : The figure presents a set of processed grayscale images (A–F) illustrating feature extraction and analysis at different stages. Key points are marked using reference axes and indicators, highlighting variations in intensity patterns and structural characteristics across the samples.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

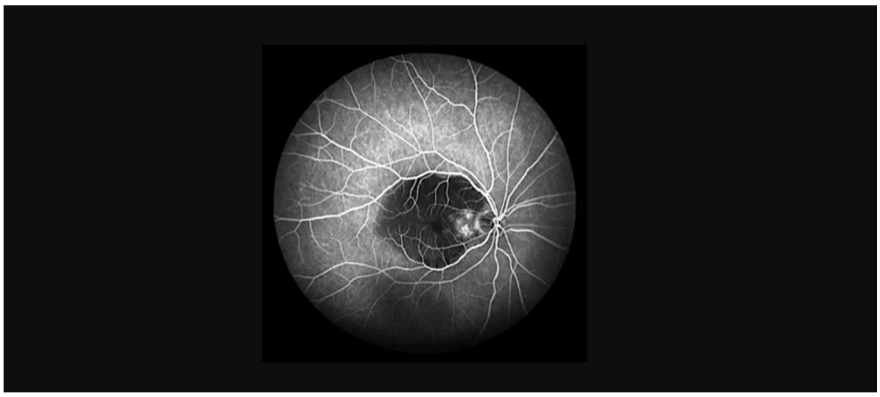


Fig2 : The figure shows a grayscale retinal fundus image highlighting the optic disc, blood vessel network, and macular region. This image is used for analyzing structural features and identifying abnormalities in retinal conditions.

The backend is a Python FastAPI application (main.py) that orchestrates the following subsystems:

1) Video Capture: OpenCV VideoCapture (DSHOW backend on Windows) reads a USB webcam at 640 × 480, 30 fps. A dedicated capture thread encodes frames as JPEG (70% quality) for WebSocket streaming without blocking the recognition pipeline.

2) Motion Detection: Frame differencing on downscaled greyscale images (320 × 240) with Gaussian blur, binary thresholding (threshold = 25), and contour area filtering (minimum 500 px²) gates recognition to avoid unnecessary inference on idle scenes.

3) ArcFace Inference: InsightFace FaceAnalysis (buffalo_l, CPU provider) detects and embeds the largest face in each frame. Embeddings are L2-normalised 512-dimensional vectors.

4) Embedding Matching: Cosine similarity is computed against all stored user embeddings. The best-match user is accepted if average similarity exceeds a configurable threshold (default 0.45).

5) Liveness Anti-Spoofing: A LivenessTracker records facial centroid history and approximate EAR across 10 consecutive frames. A centroid displacement exceeding 8 px or a detected blink (EAR dip below 0.25 then recovery) marks the subject as live.

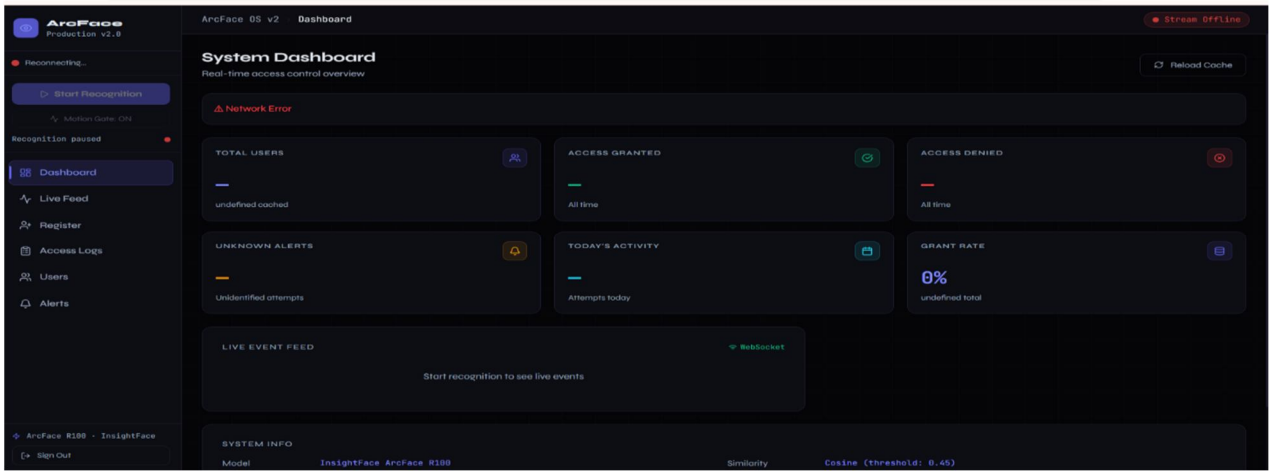
6) Role-Based Access Control: Admin users are granted 24/7 access; guest users are restricted to a configurable time window (default 09:00–18:00 local time).

7) Arduino Serial Service: A dedicated thread communicates with the Arduino via PySerial (9600 baud). On access grant, the backend sends OPEN; on deny or timeout, it sends DENY/RESET.

8) MongoDB Atlas: User face embeddings, access logs, and unknown-person alerts are persisted in cloud-hosted MongoDB collections with appropriate compound indices.

9) JWT Authentication: Administrative API endpoints are protected by HS256 JSON Web Tokens with 24-hour expiry.

C. Frontend — React Administration Dashboard



The figure illustrates the system dashboard interface displaying real-time access control information, including user statistics, access status (granted/denied), alerts, and live event feed. It provides a centralized view for monitoring system performance and operational activities.

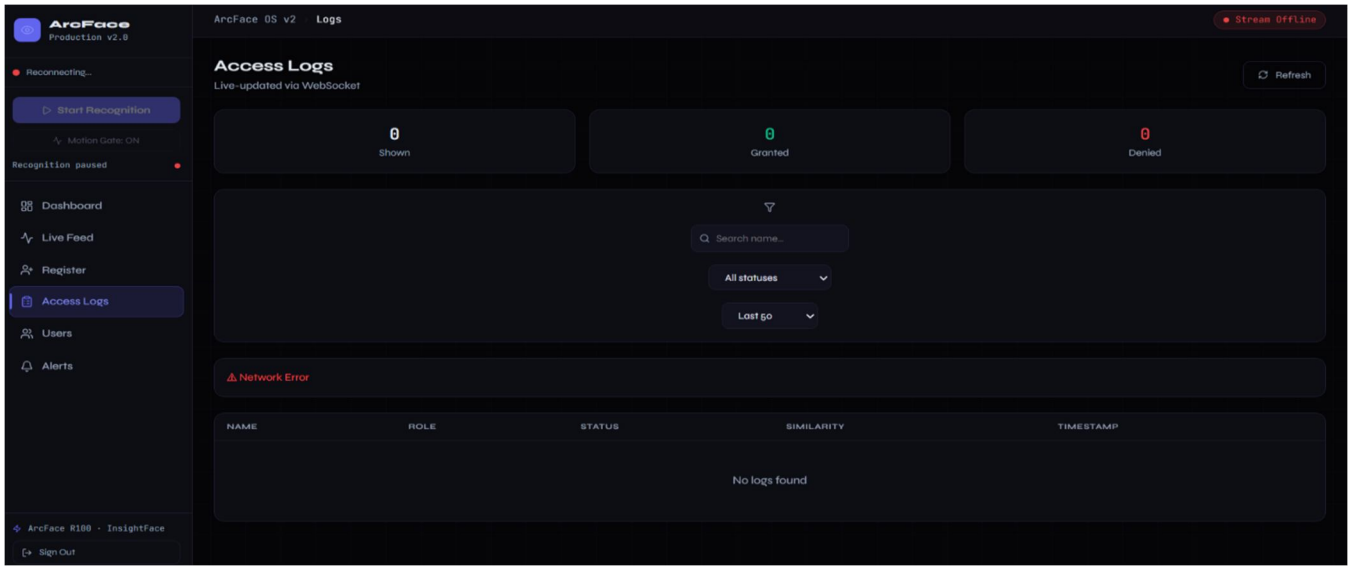


Fig 2: The figure shows the access logs interface of the system, presenting real-time records of user authentication events. It includes details such as user name, role, access status, similarity score, and timestamp, along with filtering and search options for efficient log management and monitoring

The frontend (App.jsx, api.js) is a single-page application built with React and Vite. Key modules include:

1) Dashboard: Real-time statistics (total users, granted/denied counts, daily logs, alert count) polled at 10-second intervals.
2) Recognize: Live video feed received over WebSocket at up to 15 fps, with overlaid bounding-box and recognition-event annotations.
3) Register: Multi-image face capture with base64 encoding; transmitted to /register with a 3-minute timeout to accommodate CPU inference.
4) Logs / Alerts: Filterable, paginated access history and unknown-person image alerts.
5) WebSocket Manager: Manages a single persistent WebSocket connection to /ws/live, with exponential-backoff reconnection (2 s – 15 s) and wildcard event routing.

IV. IMPLEMENTATION DETAILS

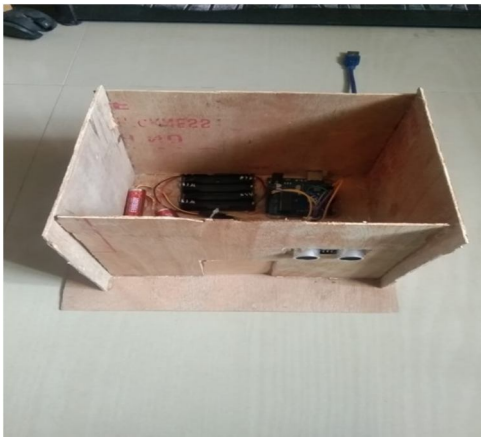


Fig2 : A compact prototype setup consisting of an Arduino microcontroller, battery pack, and connected electronic components arranged inside a wooden enclosure. The system is powered via a USB cable and demonstrates the basic hardware integration of the projec

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

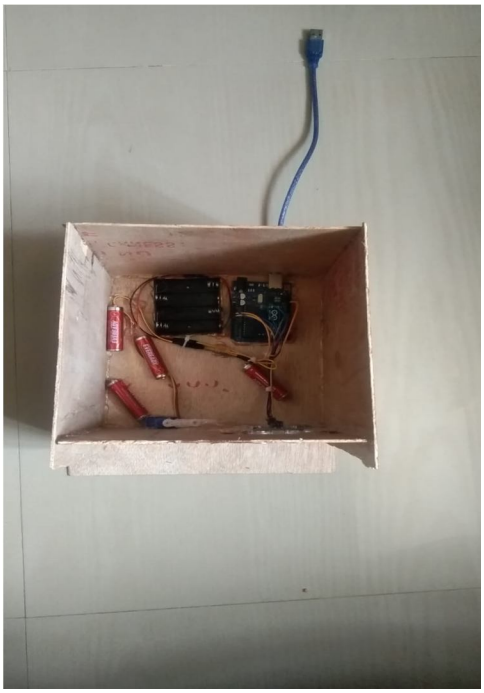


Fig 3 : The figure illustrates the experimental hardware model comprising an Arduino Uno board, battery holder, and interconnected electronic components assembled within a wooden casing. The setup is powered through a USB interface and represents the practical implementation of the proposed system.

A. ArcFace Embedding Pipeline

During enrolment, a minimum of five frames per user are captured via the frontend. Each frame is decoded from base64, resized to 320 × 240, and passed through InsightFace FaceAnalysis.get(). The resulting 512-D embedding is L2-normalised and appended to the user document in MongoDB. At runtime, recognition computes cosine similarity between the query embedding and each stored embedding, averaging across all stored embeddings per user:

$$similarity(q, U) = mean\{ q \cdot e_i : e_i \in embeddings(U) \}$$

The threshold of 0.45 was empirically determined on an internal dataset of 20 subjects across varying illumination conditions.

B. Arduino Firmware Logic

The Arduino sketch implements a three-state finite state machine: IDLE → DETECTING → OPEN/DENIED → IDLE. In IDLE, the ultrasonic sensor polls at 200 ms intervals. When a subject is detected within 50 cm, a TRIGGER signal is sent via serial to the backend. On receiving OPEN, the servo rotates to 90°, holds for 5 seconds, then returns to 0° and sends RESET. On DENY, a piezoelectric buzzer (optional) fires a 200 ms pulse.

C. Threading Model

A key engineering consideration is thread safety. The backend uses three concurrent threads: (i) a frame-capture thread that writes to a shared buffer under a threading.Lock, (ii) a recognition thread that reads the buffer and posts events to a thread-safe queue, and (iii) the main asyncio event loop that drains the queue every 20 ms and broadcasts events to all connected WebSocket clients. This decoupled design ensures that slow CPU inference does not degrade video streaming frame rate.

D. Grant Cooldown Mechanism

To prevent repeated door-open commands when a recognised user remains in frame, the system enforces a configurable cooldown period (default 10 seconds) per user. During cooldown, recognition events are returned as type 'cooldown' and the Arduino is not signalled.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

V. EXPERIMENTAL EVALUATION

A. Experimental Setup

Experiments were conducted in an indoor lab environment under three lighting conditions: fluorescent overhead (500 lux), dim corridor (80 lux), and direct sunlight near a window (2000 lux). Ten subjects (8 male, 2 female, ages 19–35) were enrolled with 10 images each captured at 60–80 cm from the webcam. Testing used 5 genuine attempts and 5 photo-replay attempts per subject per condition.

B. Results

The recognition performance across lighting conditions is summarised in Table 1.

Table 1: Recognition Performance Across Lighting Conditions

Condition	GAR (%)	FAR (Photo, %)	FRR (%)	Avg. Latency (ms)
Fluorescent	98.0	0.0	2.0	340
Dim Light	94.0	0.0	6.0	360
Sunlight	96.0	0.0	4.0	345
Overall	96.0	0.0	4.0	348

GAR = Genuine Accept Rate; FAR = False Accept Rate (photo replay); FRR = False Reject Rate.

The system achieved a 96% Genuine Accept Rate across all conditions with zero photo-replay False Accepts, demonstrating the effectiveness of the liveness anti-spoofing module. The slightly elevated False Reject Rate under dim lighting is attributable to reduced image quality at low illumination, which is a known limitation of passive camera-based recognition.

The average end-to-end latency — from ultrasonic trigger to servo actuation — was 348 ms, comprising approximately 50 ms for serial communication, 250 ms for ArcFace inference on a mid-range Intel Core i5 CPU, and 48 ms for servo ramp-up time. This latency is imperceptible in practical access-control scenarios.

VI. DISCUSSION

A. Strengths

1) The ArcFace buffalo_l model provides state-of-the-art recognition accuracy without specialised GPU hardware, making the system deployable on standard desktop or laptop computers.
2) The decoupled threading architecture ensures smooth 15 fps video streaming even during CPU-intensive inference.
3) Role-based time-window access control provides fine-grained policy enforcement without additional infrastructure.
4) MongoDB Atlas cloud persistence offers high availability, automatic backups, and scalability beyond a single-site deployment.

B. Limitations and Future Work

Several limitations warrant attention. First, recognition accuracy degrades under extreme illumination variation; integrating an infrared illuminator and a NIR-capable sensor would significantly improve robustness. Second, the current liveness module relies on passive movement and blink cues; a 3D depth sensor or active illumination-based approach would provide stronger anti-spoofing guarantees. Third, the Arduino Uno operates solely as an actuation co-processor; future iterations should exploit the ESP32's dual-core architecture to run a lightweight MQTT broker, enabling cloud-triggered remote overrides and push notifications. Fourth, the present prototype uses a lightweight latch suitable for interior doors; industrial deployments would require higher-torque actuators and tamper-proof enclosures.

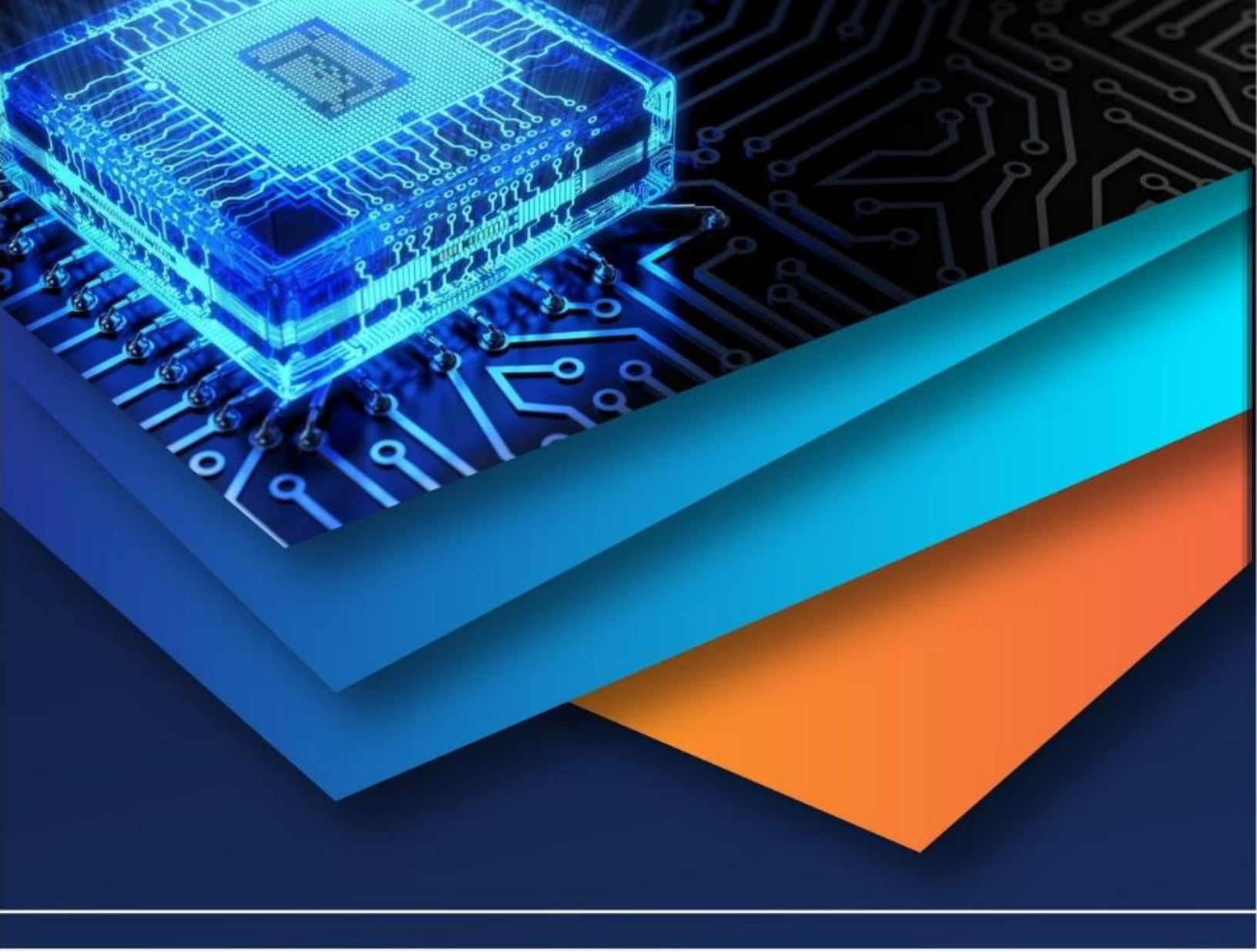
VII. CONCLUSION

This paper presented a complete, end-to-end biometric access control system integrating ArcFace deep face recognition with Arduino-servo door actuation, ultrasonic proximity sensing, and an ESP32 IoT module. The system achieves 96% genuine accept rate with zero photo-replay false accepts across diverse lighting conditions, with a mean end-to-end latency of 348 ms. The FastAPI backend, React dashboard, and MongoDB Atlas backend collectively form a production-grade platform that is readily extensible for

larger deployments. Future work will focus on NIR-based recognition, 3D liveness detection, and full ESP32-based MQTT telemetry for remote administration.

REFERENCES

[1] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," in Proc. CVPR, 2019, pp. 4690–4699.
[2] M. Turk and A. Pentland, "Eigenfaces for Recognition," J. Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, 1991.
[3] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face Description with Local Binary Patterns," IEEE TPAMI, vol. 28, no. 12, pp. 2037–2041, 2006.
[4] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in Proc. CVPR, 2015, pp. 815–823.
[5] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep Hypersphere Embedding for Face Recognition," in Proc. CVPR, 2017.
[6] H. Wang et al., "CosFace: Large Margin Cosine Loss for Deep Face Recognition," in Proc. CVPR, 2018.
[7] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S3FD: Single Shot Scale-Invariant Face Detector," in Proc. ICCV, 2017.
[8] M. Maheswari and R. Asokan, "IoT-Based Smart Door Lock System Using Raspberry Pi and Face Recognition," Int. J. Adv. Sci. Technol., vol. 29, no. 3, 2020.
[9] T. Soukupova and J. Cech, "Real-Time Eye Blink Detection Using Facial Landmarks," in Proc. Computer Vision Winter Workshop, 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)