International Journal of Science, Strategic Management and Technology
Volume **02** Issue **11** November-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Real-Time AI Smart Surveillance and Intrusion Detection System

Pradeep Gowda
Head of the Department
Dept. of CSE
Sri Revana Siddeshwara Institute of Technology, Chikkajala, Banglore 562157

Ushashree B N
Assistant Professor
Dept. of CSE
Sri Revana Siddeshwara Institute of Technology, Chikkajala, Banglore 562157
Ushashreebn@gmail.com

Bhavana N P
Dept. of CSE
Sri Revana Siddeshwara Institute of Technology, Chikkajala, Banglore 562157
bhavananp807@gmail.com

Dhanushree R Ganachari
Dept. of CSE
Sri Revana Siddeshwara Institute of Technology, Chikkajala, Banglore 562157
drganachari444@gmail.com

H Nagarathna
Dept. of CSE
Sri Revana Siddeshwara Institute of Technology, Chikkajala, Banglore 562157 nagarathnabh@gmail.com

Lolakshi S
Dept.ofCSE
SriRevana Siddeshwara Instituteof Technology, Chikkajala, Banglore Chikkajala, Banglore 562157
slolakshi580@gmail.com

https://doi.org/10.55041/ijsmt.v2i11.043

Cite this Article: Gowda, P., N, U. B., P, B. N., Ganachari, D. R., Nagarathna, H. & S, L. (2026). Real-Time AI Smart Surveillance and Intrusion Detection System. International Journal of Science, Strategic Management and Technology, *02*(11), 1-9. https://doi.org/10.55041/ijsmt.v2i11.043

License: This article is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting use, distribution, and reproduction in any medium, provided the original author(s) and source are properly credited.

***Abstract*—**Traditional surveillance systems often depend on continuous human monitoring to identify unauthorized entry into restricted areas. This paper presents a Real-Time AI Smart Surveillance and Intrusion Detection System that combines deep learning, computer vision, face recognition, embedded hardware, database management, email notification, and a web dashboard. A Logitech C270 HD webcam captures live video, and OpenCV processes the frames. A MobileNetV2-based model classifies images into PERSON and NON-PERSON categories, while YuNet and SFace support face detection and recognition of a registered authorized person. The system monitors a predefined restricted zone and confirms an intrusion when an unrecognized person remains in the zone for approximately three seconds. A confirmed event activates an ESP32-WROOM-32-based buzzer and LED alert, captures an evidence snapshot, stores event details in SQLite, and sends an email through Gmail SMTP. A login-protected Flask dashboard provides access to recorded events and snapshots. On the reported validation dataset of 115 images, the classifier achieved 93.91% accuracy, 97.22% precision, 93.33% recall, and a 95.24% F1-score. Functional testing covered detection, hardware alerts, evidence storage, email notification, and dashboard operation.

***Index Terms*—**Artificial Intelligence, Smart Surveillance, Intrusion Detection, MobileNetV2, OpenCV, Face Recognition, ESP32, Flask, SQLite.

I. INTRODUCTION

Surveillance systems are used to monitor locations where access must be controlled. Traditional surveillance commonly requires a person to observe camera footage and identify suspicious activity. Continuous manual monitoring can increase workload and may delay the identification of unauthorized entry. The proposed system uses artificial intelligence and computer vision to automate person classification, face recognition, and restricted-zone monitoring.

The system combines a webcam, a trained MobileNetV2-based classifier, YuNet face detection, SFace face recognition, and an ESP32-based hardware alert mechanism. It also records intrusion evidence, sends email notifications, and provides a web dashboard for reviewing events. The purpose is to support automated monitoring of a predefined restricted area while distinguishing a registered authorized person from an unrecognized person.

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

II. PROBLEM STATEMENT

Traditional surveillance systems often depend on continuous human observation to identify unauthorized entry. This creates a need for a system that can automatically monitor a restricted zone, check whether a detected person is authorized, generate an alert for a confirmed intrusion, and maintain event records for later review.

III. OBJECTIVES

The main objective is to develop an integrated AI-based surveillance system that distinguishes authorized individuals from unknown individuals, detects unauthorized entry into a restricted zone, generates timely alerts, stores supporting evidence, and provides an interface for reviewing recorded events.

- Capture live video using a webcam and process the frames using OpenCV.
- Classify images as PERSON or NON-PERSON using a MobileNetV2-based model.
- Detect faces using YuNet and recognize the registered authorized person using SFace.
- Monitor a predefined restricted zone and confirm an intrusion when an unrecognized person remains in the zone for the configured period.
- Activate local hardware alerts using an ESP32, buzzer, and LEDs.
- Capture snapshots and store event details in an SQLite database.
- Send email notifications through Gmail SMTP.
- Provide a login-protected Flask dashboard for reviewing intrusion records and snapshots.

IV. RELATED WORK

The project report identifies MobileNetV2 as the base architecture for efficient image classification. MobileNetV2 uses inverted residuals and linear bottlenecks and was introduced for efficient visual recognition

[1]. OpenCV provides computer-vision functions used for webcam access, image processing, and live video handling

[2]. MobileNets describe efficient convolutional neural network architectures for mobile and resource-constrained vision applications

[3]. Deep learning and computer-vision concepts are also discussed in the references listed in the project report.

The proposed implementation combines these software and AI components with face detection and recognition, restricted-zone logic, ESP32 hardware alerts, event storage, email notifications, and a web dashboard. The references below are reproduced from the project report; this paper does not claim a separate experimental comparison with other surveillance systems.

V. PROPOSED SYSTEM

The proposed system monitors a predefined restricted zone using a live webcam feed. OpenCV captures and prepares the video frames. The MobileNetV2-based model classifies person presence. If a person is detected, YuNet locates a face and SFace compares the facial representation with the stored representation of the registered authorized person. A recognized authorized person does not trigger an intrusion alert. An unrecognized or unverifiable person is not treated as authorized. When the configured restricted-zone condition is met for approximately three seconds, the system confirms an intrusion.

After confirmation, the application sends a command to the ESP32-WROOM-32, activates the buzzer and red LED, captures a snapshot, records the event in SQLite, and sends an email notification. The Flask dashboard retrieves the recorded information and provides access to the event history and snapshots.

VI. SYSTEM ARCHITECTURE

A. Data Layer: Stores the trained person-classification model, face-recognition model data, authorized-person representation, captured snapshots, configuration values, and SQLite database records.

B. Processing Layer: Performs webcam frame processing, MobileNetV2 person classification, YuNet face detection, SFace face recognition, restricted-zone monitoring, and intrusion confirmation.

C. Application Layer: Coordinates the live detection program, serial communication with the ESP32, snapshot management, SQLite database operations, Gmail SMTP notifications, and the Flask backend.

D. Presentation Layer: Provides the login interface and dashboard for reviewing event timestamps, confidence values, event status, email status, and captured snapshots. The buzzer and LEDs provide local status indications.

VII. METHODOLOGY

A. Data Collection: The classification dataset contains PERSON and NON-PERSON images. Live video is captured using a Logitech C270 HD webcam. The registered authorized person's facial representation is generated for comparison during monitoring.

B. Data Preprocessing: Images are resized to 160 × 160 pixels and organized into PERSON and NON-PERSON classes. Training and validation datasets are prepared separately, and the model's preprocessing configuration is applied consistently.

C. Model Selection and Training: A MobileNetV2-based convolutional neural network is developed using TensorFlow and Keras. The trained model is saved as person_classifier.keras and loaded by the live surveillance application.

D. Real-Time Classification and Face Recognition: OpenCV captures live webcam frames. MobileNetV2 classifies person presence. YuNet detects faces, and SFace

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

compares facial features with the stored authorized-person representation.

E. Restricted-Zone Monitoring: A virtual restricted zone is defined in the camera frame. The system checks person presence, zone entry, and recognition results. An unrecognized person remaining in the zone for approximately three seconds causes an intrusion event to be confirmed.

F. Hardware Integration: The ESP32-WROOM-32 communicates with the Python application through serial communication. The intrusion command activates the buzzer and red LED. The secured-state command is indicated by the green LED. A push button silences the buzzer while the red LED continues to indicate the intrusion condition.

G. Evidence and Event Management: When an intrusion is confirmed, a snapshot is saved in the alerts directory. SQLite stores event date and time, confidence value, detection status, snapshot path, and email status.

H. Email and Dashboard: Gmail SMTP sends an intrusion notification to the configured recipient. Flask provides a login-protected dashboard that displays recorded events and provides access to snapshots.

VIII. IMPLEMENTATION REQUIREMENTS

The project was developed in a Windows 11 environment using Python. The hardware includes a computer system, Logitech C270 HD webcam, ESP32-WROOM-32 development board, buzzer, LEDs, push button, and USB cable. The software and libraries include Python, Visual Studio Code, TensorFlow/Keras, MobileNetV2, OpenCV, YuNet, SFace, Flask, SQLite, and Gmail SMTP. The computer configuration stated in the project report recommends an Intel Core i5 or equivalent processor and 8 GB RAM or higher.

IX. RESULTS AND DISCUSSIO

A. Model Evaluation:

The project report evaluates the classifier using 115 validation images: 75 PERSON images and 40 NON-PERSON images. The confusion matrix is shown in Table I. The model correctly classified 108 out of 115 validation images. The reported validation metrics are 93.91% accuracy, 97.22% precision, 93.33% recall, and 95.24% F1-score. These values describe performance on the stated validation dataset and do not guarantee identical performance in every real-world environment.

TABLE I
CONFUSION MATRIX

Actual Class	Predicted NON-PERSON	Predicted PERSON
NON-PERSON	38	2
PERSON	5	70

B. Functional Testing:

The system was tested under bright lighting, dim lighting, and empty-room conditions. The report states that no false intrusion alert was observed during an approximately two-to-three-minute empty-room test. Integration testing covered webcam capture, person classification, face detection and recognition, restricted-zone checking, ESP32 alerts, snapshot capture, SQLite event recording, email notification, and dashboard access.

C. Hardware and Software

Output: A confirmed intrusion activates the buzzer and red LED. The green LED indicates the secured state. The push button can silence the buzzer while the red LED continues to indicate the intrusion condition. Event details and snapshots are available through the login-protected Flask dashboard.

X. CONCLUSION

The Real-Time AI Smart Surveillance and Intrusion Detection System combines deep learning, computer vision, face recognition, embedded hardware, database management, email notification, and web technologies for automated restricted-zone monitoring. The MobileNetV2-based classifier achieved the reported validation accuracy of 93.91%, with 97.22% precision, 93.33% recall, and a 95.24% F1-score. The integrated system supports person classification, authorized-person recognition, intrusion confirmation, local hardware alerts, evidence capture, database logging, email notification, and dashboard-based event review. The reported tests demonstrate the operation of the integrated components under the tested conditions. Further evaluation over longer periods and in more varied environments is needed to assess broader real-world reliability.

XI. FUTURE WORK

The project report identifies several possible enhancements: using a larger and more diverse training dataset; incorporating object detection for more precise person localization and multiple-person handling; supporting multiple restricted zones and cameras; adding secure cloud-based monitoring; developing a mobile application with push notifications; extending hardware integration; improving event filtering and report generation; evaluating detection reliability over longer periods and more challenging conditions; detecting additional objects; strengthening dashboard authentication and access control; and supporting multiple registered authorized individuals.

REFERENCES

[1] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510–4520.

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

[2] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[3] A. G. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.

[4] F. Chollet, Deep Learning with Python, 2nd ed. Manning Publications, 2021.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Adv. Neural Information Processing Systems, vol. 25, 2012, pp. 1097–1105.

[6] R. Szeliski, Computer Vision: Algorithms and Applications, 2nd ed. Springer, 2022.

[7] Espressif Systems, "ESP32 Series Datasheet." Available: https://www.espressif.com/

[8] TensorFlow, "TensorFlow Documentation." Available: https://www.tensorflow.org/

[9] OpenCV, "OpenCV Documentation." Available: https://docs.opencv.org/

[10] Pallets Projects, "Flask Documentation." Available: https://flask.palletsprojects.com/

[11] SQLite Consortium, "SQLite Documentation. Available: https://www.sqlite.org/docs.html

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.