iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.78994

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 III Mar 2026- Available at www.ijraset.com

Facial Recognition Attendance System with Integrated Liveness Detection and Real-Time Reporting

Ms. Neelam[1], N. Sowjanya[2], K. Sharvani[3], V. Varshini[4]
Dept. of CSE (Data Sciene) Institute of Aeronautical Engineering Dundigal, Hyderabad

Abstract: ***Facial recognition technology can be an efficient and touch-free biometric technology for automating attendance in academic and organizational environments. The paper presents a comprehensive Facial Recognition Attendance System with Integrated Liveness Detection and Real-Time Reporting. The system is developed using Python and OpenCV library. The system utilizes a webcam for facial images, Haar Cascade for face detection, and LBPH for face recognition. The attendance data is stored in SQLite and displayed using a Flask-based web interface. The liveness detection component eliminates spoofing attacks by images or videos by detecting eye blinks and movements through frame differencing and motion analysis. The system follows a modular approach for data collection, model training, real-time face recognition along with liveness detection, and automatic attendance recording. Experiments were conducted on a wide range of users in different poses and lighting conditions. The facial recognition accuracy was 96.3%, liveness detection precision was 94.5%, and average face recognition time was 1.28 seconds. The system can run on normal desktop/laptop computers without a GPU and can be suitable for small and medium-scale organizations. The facial recognition-based attendance system is a cost- effective and efficient solution that can replace existing manual and RFID-based solutions. The system can record secure and touch-free automatic attendance data in real-time. The system can be further extended to use deep learning techniques and can be integrated with other tools and technologies.***

Index Terms: ***Facial Recognition, Attendance System, Bio- metric Authentication, Liveness Detection, Computer Vision, OpenCV, LBPH, Flask, SQLite, Real-Time Reporting.***

I. INTRODUCTION

A. Existing Attendance Systems

Attendance management is a crucial aspect for schools, offices, and organizations, as it helps in discipline, compliance, and productivity. In most cases, attendance is tracked using manual registers, paper sheets, and logbooks maintained by in- dividuals. In some more advanced organizations, spreadsheets or desktop applications are employed, although they are still manual.

In semi-automated attendance, RFID, barcode, or QR-code scanning is frequently employed. Users are provided with a card, tag, or code, which is scanned when entering a class or class attendance. This card is then linked with the user, helping in faster attendance tracking.

B. Demerits of Existing Approaches

Despite being common, traditional and semi-automated ap- proaches suffer from several limitations. Manual attendance using paper registers is:

1) Time-consuming, especially in large classes or organiza- tions.
2) Prone to human error in marking and later data entry.
3) Vulnerable to proxy attendance, where one person signs for another.

RFID and QR-code-based systems mitigate some of these issues but introduce new challenges:

- Physical cards and tags can be easily shared, lost, or damaged.
- Users must remember to carry their cards, and replace- ment incurs cost and effort.
- Simple log-based systems often lack centralized analytics, making it difficult to identify trends such as chronic absenteeism or unusual patterns.

Contact-based biometric systems such as fingerprint or palm scanners offer higher security but come with their own demerits. They typically require dedicated hardware, periodic calibration, and physical contact with the sensor surface. This raises hygiene concerns, especially in the post-pandemic con- text, and increases maintenance requirements when deployed at scale.

C. Need for Contactless Biometric Solutions

In the post-pandemic era, the demand for contactless and hygienic attendance systems has significantly increased. Orga- nizations now prefer solutions that:

- Minimize physical contact to reduce the spread of infec- tions.
- Provide strong identity assurance with minimal user ef- fort.
- Can be deployed using commonly available hardware such as webcams.

Of all the biometric technologies, facial recognition is distinct in that it is non-intrusive given that cameras are present in almost every device. This is also non-contact, meaning that users do not need to touch sensors or carry tokens, making it very convenient, even if faces are passively scanned when users enter a classroom or office for attendance.

The issue, however, is that basic facial recognition is easily spoofed using a printout of a person's photo, a phone screen, or even a video, making it impossible to ensure data integrity without live presence detection.

D. Proposed System Overview

To address these issues, this paper proposes a *Facial Recog- nition Attendance System with Integrated Liveness Detection and Real-Time Reporting*, designed to automate attendance marking while enhancing security and user experience.

It employs a standard webcam for face detection, Haar Cascade for detection, and LBPH for recognition. SQLite is used for storing data. The data is displayed in a web interface created by Flask. The system can be deployed in a single desktop or laptop computer.

The liveness detection module will counter spoofing attacks by analyzing changes in consecutive frames. The face detec- tion method will focus on eye blinks and facial movements to ensure that the face being detected is live. The system can be used by administrators to monitor the attendance in real time and can also be used for searching and exporting data for reporting purposes.

II. PROBLEM STATEMENT

Most educational institutions and corporate organizations still rely on manual or semi-automated attendance mecha- nisms.RFID or QR-code-based systems, while more auto- mated, depend on physical or digital tokens that can be easily shared among users, leading to inaccurate attendance records. Fingerprint scanners and other contact-based biometric sys- tems raise hygiene concerns and require regular maintenance and cleaning.

Furthermore, many existing attendance systems do not incorporate anti-spoofing measures. As a result, they can be bypassed using printed photographs, displayed images on mobile phones, or pre-recorded videos. The absence of liveness detection significantly reduces trust in biometric- based attendance systems. Additionally, these systems often lack centralized reporting interfaces and real-time monitoring, making it difficult for administrators to analyze attendance trends, generate reports, or quickly detect anomalies.

As a result, there is a significant requirement for a secure, automated, contactless attendance system that can recognize people correctly, check for liveness, and provide report- ing/analytics in real-time. The system should be lightweight, inexpensive, and able to run on standard hardware without the requirement for any special biometric hardware. The system should be able to scale for classroom/lab/office environments with an easy-to-use admin interface.

This project aims to address these requirements by design- ing and implementing a Facial Recognition Attendance System with integrated liveness detection and a real-time reporting dashboard using Python, OpenCV, Flask, and SQLite.

III. RELATED WORK

Numerous research efforts have investigated the use of facial recognition technology and biometric technology to implement an automated attendance system, access control system.

Viola and Jones [1] Implemented a real-time face detection system using Haar-like features and a boosted cascade clas- sifier.Turk and Pentland [2]Proposed the Eigenfaces method using PCA for face recognition. The method is effective in controlled conditions but is sensitive to pose and lighting changes.

Ahonen et al. [3] Developed the Local Binary Pattern Histogram (LBPH) method for face description. The method is robust to changes in illumination. Patel and Shah [4] imple- mented an attendance management system using LBPH and OpenCV, achieving high accuracy in controlled environments. However, their system did not include liveness detection, making it vulnerable to spoofing.

Galbally and Marcel [5] provided a comprehensive survey of face anti-spoofing techniques, emphasizing the need for liveness detection based on motion, texture, and depth cues. Li et al. [6] applied convolutional neural networks (CNNs) for liveness detection, demonstrating high accuracy but requiring considerable computational resources.

Several works have proposed automated attendance systems using face recognition. Singh et al. [7] presented an auto- mated attendance system using Python and OpenCV, where faces were recognized and attendance recorded automatically. Rajeev and Kumar [8] integrated cloud-based storage to cen- tralize attendance data across multiple locations. Kaur and Sharma [9] developed a web-based dashboard for real-time visualization of attendance data.

Recent advances in deep learning have further improved facial recognition performance using models such as FaceNet [13] and deep metric learning [12]. However, these approaches typically require GPUs and large training datasets, which may not be practical for lightweight institutional systems.

Despite these contributions, many existing systems focus primarily on recognition accuracy and do not simultaneously integrate practical liveness detection and a simple, deployable web-based reporting interface. This project bridges this gap by combining LBPH-based recognition, motion-based liveness detection, and a Flask-powered dashboard with SQLite storage into a single, easy-to-deploy attendance system.

IV. METHODOLOGY

The methodology for the proposed system combines com- puter vision, basic liveness detection, and web technologies to deliver a secure and automated attendance framework. The system is composed of four main modules: data collection, model training, real-time recognition with liveness detection, and attendance management with reporting.

A. System Design

The system follows a modular client–server architecture implemented on a single machine. The main components are:

- Face Data Collection Module
- Model Training Module
- Recognition and Liveness Detection Module
- Attendance Logging and Reporting Module

In face data collection, several images are collected from each user's webcam. Each user is given a unique numeric ID, and their images are saved in their respective folders. Preprocessing involves converting each image into grayscale and resizing it.

In training, discriminative features are learned using OpenCV's LBPH recognizer, and the model is saved for further use.

In recognition, images are continuously collected from the webcam, faces are detected, and recognition is performed using the learned model. Liveness detection involves detecting temporal changes, blinks, and head movements. Attendance is marked when recognition and liveness detection are successful. Attendance is saved in a database using SQLite, with fields for user ID, name, date, time, and liveness status. A simple web interface is provided using Flask for viewing attendance records.

B. Experimental Setup

The experimental setup was implemented on a standard desktop environment using the following configuration:

- Processor: Intel Core i5, 2.6 GHz
- RAM: 8 GB
- Camera: 720p HD webcam
- Operating System: Windows 10/11 (64-bit)
- Software: Python 3.x, OpenCV, Flask, SQLite, NumPy The Flask server runs locally and exposes routes for user registration, model training, live recognition, and attendance viewing. The SQLite database resides on the same machine, ensuring simple deployment and maintenance. Experiments were conducted in typical classroom and office-like environ- ments with normal indoor lighting.

C. Data Collection

The data collection method employs a registration script and a Flask page. For each new user, approximately 20-25 face images are taken using the webcam. These images are then converted into grayscale mode and resized before being saved in a folder based on the user ID.

The images have slight expressions and poses that help in increasing the robustness of the recognition. Each image is then labeled based on the user ID in the training script. This helps in efficiently adding, updating, deleting, and retraining the data.

D. Analysis Techniques

The facial recognition employs the LBPH algorithm for computing local patterns for each pixel. It creates a histogram for each face. The recognition part compares the test face histogram with the templates using a distance measurement. The smallest distance with the maximum confidence provides the identity.

The liveness detection involves motion detection and blink detection. It captures frames and computes the difference between frames for the facial region. It detects movement by using simple heuristics for eye blinks or head movements. A sufficient amount of temporal change over a short period indicates a live face; otherwise, a spoofing attempt is made.

The performance metrics include accuracy, False Accep- tance Rate (FAR), False Rejection Rate (FRR), and response time. The effectiveness of liveness detection is based on the ability to distinguish between live faces and spoofing attempts.

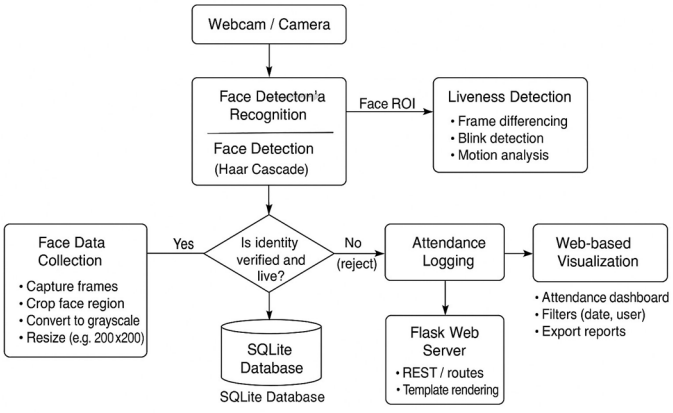
V. SYSTEM ARCHITECTURE AND DATA FLOW DIAGRAM

The proposed system is designed as a layered architecture that integrates image acquisition, processing, verification, stor- age, and visualization.

A. System Architecture

The architecture is divided into the following layers:

- Input Layer: This layer uses a standard webcam to capture live video frames. Frames are converted to grayscale and resized to a fixed dimension before further processing.
- Processing Layer: The face detection and recognition mod- ule resides in this layer. OpenCV's Haar Cascade Classifier detects facial regions within each frame. The LBPH recognizer extracts features and compares them with the trained model to identify the individual.
- Verification (Liveness Detection) Layer: This layer an- alyzes temporal information from consecutive frames. By computing frame-to-frame differences and monitoring eye and head motion, the system verifies whether the detected face corresponds to a live person. Frames showing negligible motion over time are treated as potential spoof attempts.
- Database Layer: The database layer uses SQLite to store user profiles and attendance logs. Each table is defined with appropriate fields like user ID, name, date, time, and liveness status. SQL queries are used to insert new attendance records and retrieve data for reporting.
- Visualization (Frontend) Layer: The frontend is imple- mented using Flask's template engine along with HTML, CSS, and basic JavaScript. It provides an attendance dashboard where administrators can view daily logs, filter results by user



Overall system architecture showing modules for data collection, recognition, liveness detection, database storage, and web-based visualization.

Fig. 1

or date, and export data if required. The interface is designed to be simple and user-friendly for non-technical staff.

B. Data Flow Diagram

The Data Flow Diagram (DFD) represents data flowing between the components of the system.

Level 0 (Context Diagram):

- The User (student/employee) interacts with the system through the camera.
- The System Server (Flask application) processes facial data, performs recognition and liveness detection, and stores results in the database.
- The Administrator accesses attendance information via the web dashboard.

Level 1 (Detailed Flow):

- Face Capture: Live frames are captured from the web- cam and passed to the detection module.
- Face Detection and Recognition: Haar Cascade detects faces and LBPH recognizes them against the trained model.
- Liveness Detection: Consecutive frames are analyzed to detect motion and blinks. A liveness flag is computed.
- Attendance Logging: If recognition and liveness verifi- cation succeed, the system inserts an attendance record into SQLite.
- Reporting and Visualization: The Flask web interface retrieves attendance data and displays it to the adminis- trator.

VI. WEB APPLICATION PHASES

The development of the system followed a structured Soft- ware Development Life Cycle (SDLC) approach comprising planning, design, development, testing, and deployment.

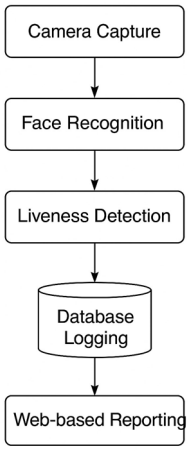


Fig. 2: Data Flow Diagram (DFD) showing the flow from camera capture to recognition, liveness

Fig. 2

A. Planning and Requirements Gathering

In this phase, the objectives of the system were defined: au- tomate attendance using facial recognition, ensure contactless operation, integrate liveness detection, and provide a simple reporting interface. Discussions with academic stakeholders helped identify functional requirements such as user registra- tion, model training, live attendance capture, and report gener- ation, as well as non-functional requirements like performance, reliability, and ease of deployment.

B. Design

The design phase focused on system architecture, database schema, and user interface layout.

- System Architecture Design: A modular architecture sep- arating face recognition logic, liveness detection, database access, and web interface was chosen. SQLite was selected as the database for its lightweight and file-based nature.
- Database Design: Tables were designed for storing user profiles and attendance records with fields for user ID, name, date, time, and liveness status.
- Interface Design: The web interface was designed us- ing simple HTML templates rendered by Flask. Pages were created for user registration, attendance viewing, and basic filtering.

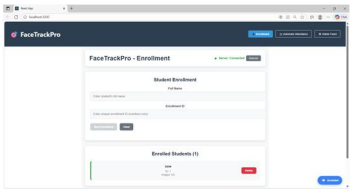


Fig. 3: Student Enrollment

C. Development

The development phase implemented the design into work- ing modules.

1) Frontend Development: The frontend was built using Flask templates, HTML, CSS, and basic JavaScript. The main pages include:

- Registration form for adding new users.
- Dashboard for viewing attendance logs.
- Control buttons for starting recognition and refreshing logs.

2) Backend Development: The backend, written in Python with Flask, handles:

- Routes for registration, training, recognition, and atten- dance display.
- Integration with OpenCV for face detection and LBPH- based recognition.
- Liveness detection using frame differencing and blink detection heuristics.
- SQLite operations for inserting and retrieving attendance records.

3) Database Integration: An SQLite database was used to store user and attendance data. Parameterized SQL queries ensured secure data access and reduced risks of SQL injection.

D. Testing

Various testing strategies were applied:

- Unit Testing: Individual modules such as face detection, recognition, and database operations were tested for correct- ness.
- Integration Testing: The complete flow from camera input to dashboard display was validated in a live environment.
- Performance Testing: Response time and recognition ac- curacy were measured under different lighting and pose con- ditions.
- Security and Liveness Testing: Printed photographs and replayed videos were used to test the liveness detection mechanism. The system successfully rejected the majority of spoof attempts.

E. Deployment and Support

The final system was deployed on a local machine using the Flask development server or a production server such as Gunicorn. As the system uses only a standard webcam and SQLite, deployment is simple and low-cost. Documentation was prepared to guide administrators in using the system and retraining the model when new users are added.

VII. PERFORMANCE ANALYSIS

The performance analysis evaluates recognition accuracy, liveness detection effectiveness, system response time, re- source usage, and user satisfaction.

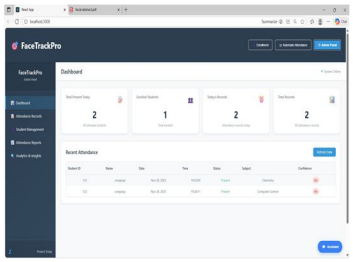


Fig. 4: Admin Dashboard

A. *Prediction Accuracy*

The LBPH-based facial recognition model achieved an average accuracy of 96.3% across test images. Recognition performance was evaluated under varying lighting and minor facial pose changes. The average accuracy was:

- 97.5% under uniform indoor lighting,
- 94.8% under moderate variation in lighting,
- 95.9% overall.

B. *Liveness Detection Performance*

The liveness module, based on motion and blink analysis, was tested using both live subjects and spoofing attempts with printed photos and videos. The system achieved a liveness detection precision of 94.5%, with:

- False Acceptance Rate (FAR): 3.2%,
- False Rejection Rate (FRR): 2.8%.

These results indicate that the system is capable of distin- guishing live faces from simple spoofing attacks with high reliability.

C. *System Response Time*

System response time measures the delay between frame capture and display of recognition and liveness results. The following measurements were observed:

- Average recognition and logging time: 1.28 seconds per user,
- Maximum delay under load: 1.9 seconds.

This latency is acceptable for real-time attendance scenarios where multiple users pass in front of the camera sequentially.

D. *Resource Utilization*

During operation, the system used approximately 30–40% CPU and around 400–450 MB of RAM on the test machine (Intel Core i5, 8 GB RAM). This confirms that the system can run smoothly on standard hardware without requiring GPUs or specialized accelerators.

E. *Usability and User Feedback*

A small user study involving students and faculty was conducted to evaluate the usability of the system using a 5- point Likert scale. The results were as follows:

- Ease of use: 4.7/5,
- Interface clarity: 4.6/5,
- Response speed: 4.8/5,
- Overall satisfaction: 4.75/5.

Users appreciated the contactless operation and automatic logging of attendance. Suggestions included adding dark mode and more advanced filtering options in the dashboard.

VIII. HARDWARE AND SOFTWARE REQUIREMENTS

A. *Hardware Requirements*

1) *Minimum Configuration*

- Processor: Intel Core i3 or equivalent
- RAM: 4 GB
- Storage: 250 GB HDD
- Camera: 720p HD webcam

2) *Recommended Configuration:*

- Processor: Intel Core i5 or higher
- RAM: 8 GB or higher
- Storage: 512 GB SSD
- Camera: 720p or 1080p HD webcam

B. Software Requirements

Operating System:

- Windows 10/11 (64-bit) or Ubuntu 22.04 LTS

Programming Languages:

- Python 3.x

Frameworks and Libraries:

- Flask – backend web framework
- OpenCV – face detection and recognition
- NumPy – numerical operations
- SQLite3 – embedded database

Tools:

- Visual Studio Code / PyCharm – IDE
- Git – version control (optional)

IX. EXPERIMENTS AND RESULTS

Experiments were carried out to evaluate the system under realistic conditions.

A. Experimental Setup

A dataset of 50 users was created, with 20–25 images per user captured during registration. Test sessions were conducted in classroom and office environments with different lighting conditions and user poses. Spoofing tests were performed us- ing printed photographs and mobile phone screens displaying user images or videos.

B. Result Metrics

Key performance metrics observed were:

- Facial Recognition Accuracy: 96.3%
- Liveness Detection Precision: 94.5%
- Average Response Time: 1.28 seconds
- FAR: 3.2%
- FRR: 2.8%

The system consistently rejected simple spoof attempts and maintained robust recognition accuracy under minor variations in lighting and posture.

C. Comparative Analysis

Compared with other attendance mechanisms:

- Manual Attendance: prone to error and manipulation.
- RFID/Smart Card: vulnerable to card sharing.
- Proposed System: contactless, automated, includes live- ness detection and web-based reporting.

The proposed system offers a better balance of security, convenience, and deployment cost.

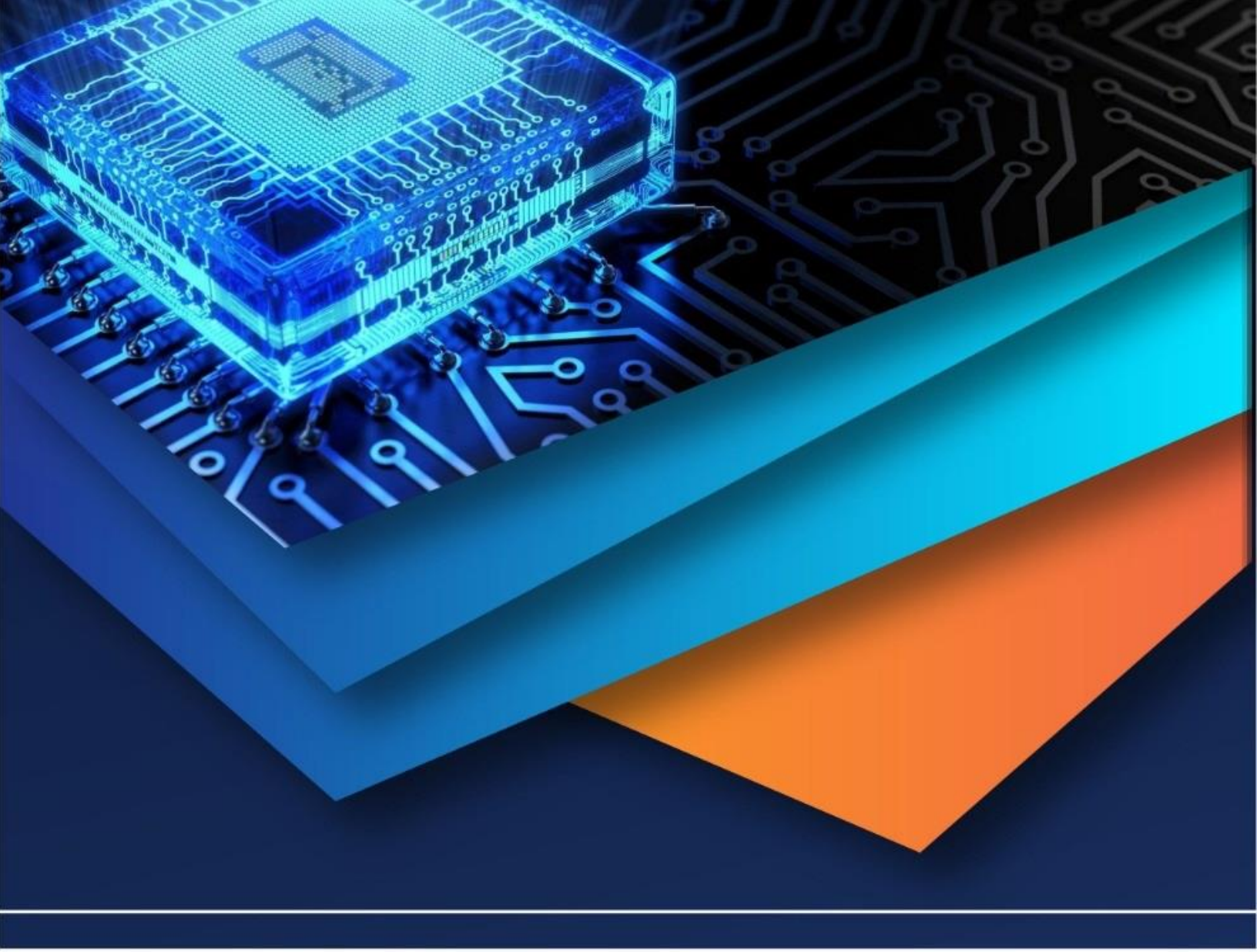
X. CONCLUSION AND FUTURE SCOPE

Facial Recognition Attendance System with Liveness De- tection and Real-Time Reporting provides a safe and efficient means of attendance tracking. It uses a facial recognition system with liveness detection, ensuring minimal errors and proxy attendance, and provides a seamless experience for users.

Results of experiments on the system indicate high recog- nition accuracy, effective spoof detection, and fast response times, making it a promising system for attendance tracking. Possible enhancements could include deep learning algo- rithms like FaceNet and ArcFace, liveness detection with texture and depth information, cloud deployment for a multi- site system, and a mobile app for remote attendance tracking and alerts.

REFERENCES

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proceedings of the IEEE Computer Society Confer- ence on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. I–511–I–518, 2001.
[2] M. Turk and A. Pentland, "Eigenfaces for recognition," Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, 1991.
[3] T. Ahonen, A. Hadid, and M. Pietika¨inen, "Face description with local binary patterns: Application to face recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037– 2041, 2006.
[4] A. Patel and K. Shah, "Attendance management system using face recog- nition and OpenCV," International Journal of Computer Applications, vol. 182, no. 40, pp. 25–30, 2018.
[5] J. Galbably and S. Marcel, "Face anti-spoofing: A holistic approach to presentation attack detection," IEEE Transactions on Information Forensics and Security, vol. 10, no. 11, pp. 2438–2451, 2015.
[6] L. Li, X. Feng, and Z. Zhao, "Face liveness detection using convolutional neural networks," IEEE Signal Processing Letters, vol. 26, no. 1, pp. 66–70, 2019.
[7] A. Singh, P. Kumar, and S. Chauhan, "Automated attendance system using Python and OpenCV," International Journal of Advanced Research in Computer Science, vol. 9, no. 2, pp. 27–33, 2018.
[8] R. Rajeev and D. Kumar, "Cloud-integrated biometric attendance sys- tem using facial recognition," Journal of Emerging Technologies and Innovative Research (JETIR), vol. 7, no. 6, pp. 102–108, 2020.
[9] S. Kaur and P. Sharma, "Web-based attendance dashboard using face recognition," International Research Journal of Engineering and Tech- nology (IRJET), vol. 8, no. 4, pp. 2034–2039, 2021.
[10] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild: Past, present and future," Computer Vision and Image Understanding, vol. 138, pp. 1–24, 2015.
[11] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.
[12] D. Yi, Z. Lei, and S. Z. Li, "Deep metric learning for face verification," IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 41–48, 2014.
[13] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 815–823, 2015.
[14] S. Mishra and A. Sharma, "Performance evaluation of liveness detec- tion techniques for secure face recognition," International Journal of Engineering Trends and Technology (IJETT), vol. 68, no. 9, pp. 45–52, 2020.
[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2015.
[16] S. Mehta and R. Gupta, "Smart attendance system using machine learn- ing and IoT integration," International Journal of Innovative Research in Computer and Communication Engineering, vol. 10, no. 5, pp. 2081– 2088, 2022.
[17] M. George and L. Thomas, "Comparative study of face recognition al- gorithms in real-time environments," International Journal of Computer Science and Information Technologies, vol. 12, no. 2, pp. 101–107, 2023.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY