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.2026.78340

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

Next-Gen Face Recognition Attendance System Using Deep Learning

Darapureddy Manikanta[1], Appari NavaSaideep[2], Relangi Vikas[3], Ponnamanda Raviteja[4], B. Narasimha Rao[5]
[1, 2, 3, 4]*Department of Computer Science, Artificial Intelligence and Data Science, Bonam Venkata Chalamayya Engineering College, Odalarevu, India*
[5]*Associate Professor, Department of Cse Allied, Bonam Venkata Chalamayya Engineering College, Odalarevu, India*

Abstract: ***Conventional attendance tracking methods such as manual roll calls, ID-card swipes, and RFID-based systems suffer from proxy attendance, time inefficiency, and scalability issues. This paper presents a next-generation face recognition attendance system built on deep learning principles using the face-api.js library, which leverages TensorFlow.js for real-time, browser-based facial detection and recognition. The proposed system is implemented as a modern, full-stack web application using Next.js 15 (React 19) on the front end and MongoDB as the NoSQL database. The architecture separates administrative and employee portals with JWT-based authentication and bcrypt password hashing for robust security. The face recognition pipeline employs a Tiny Face Detector model for lightweight face detection and SSD MobileNet v1 for extracting 128-dimensional face descriptors, which are then matched using Euclidean distance thresholding. The system supports real-time webcam streaming, automatic face detection polling, GPS-based geolocation logging, and mobile-responsive UI. Experimental results on internal deployments demonstrate high recognition accuracy (≥97.5%), low latency inference (<500 ms per recognition cycle), and resilient performance under varying lighting conditions. The system offers a scalable, cost-effective, and tamper- resistant alternative to traditional attendance mechanisms, suitable for educational institutions and corporate environments.***

Keywords: Face Recognition, Attendance System, Deep Learning, Convolutional Neural Networks, TensorFlow.js, face-api.js, Next.js, MongoDB, Real-Time Detection, Biometric Authentication, SSD MobileNet, Euclidean Distance

I. INTRODUCTION

Attendance management is a critical administrative function across educational institutions, corporate offices, and industrial environments. Traditional methods including manual roll calls, swipe cards, and RFID-based solutions have long been plagued by vulnerabilities such as proxy attendance, buddy punching, card sharing, and operational inefficiency. These shortcomings necessitate the exploration of more intelligent, automated, and tamper-proof alternatives.

Biometric-based attendance systems have emerged as a viable solution, with fingerprint scanners and iris recognition leading the way. However, these contact-based modalities face hygiene concerns, especially in post-pandemic scenarios, and often require dedicated hardware infrastructure. Face recognition, being a contactless biometric, offers significant advantages in terms of user convenience, non-intrusiveness, and deployability using commodity hardware such as standard webcams.

Recent advances in deep learning, particularly Convolutional Neural Networks (CNNs), have dramatically improved the accuracy and speed of face recognition systems. The advent of JavaScript-based machine learning frameworks such as TensorFlow.js has further enabled face recognition to be performed entirely within the browser — eliminating the need for server-side GPU infrastructure and reducing latency.

This paper presents a Next-Gen Face Recognition Attendance System that combines state-of-the-art deep learning models with a modern full-stack web architecture. The system uses face-api.js, a JavaScript library built atop TensorFlow.js, to perform face detection, facial landmark localization, and face descriptor extraction directly in the client's browser. These descriptors are matched against pre-registered profiles stored in a MongoDB database using Euclidean distance similarity, enabling seamless, real-time attendance marking.

The proposed system implements a role-based architecture with distinct Admin and Employee portals, JWT-based authentication, GPS geolocation stamping, Progressive Web App (PWA) capabilities, and comprehensive reporting and analytics dashboards. The architecture is designed for horizontal scalability and can be deployed across cloud infrastructure.

©IJRASET: All Rights are Reserved

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

II. LITERATURE REVIEW

Early attendance systems relied on manual methods such as paper-based register entries and verbal roll calls. While straightforward, these methods were susceptible to errors, impersonation, and were time-consuming for large groups [1]. The introduction of electronic methods — magnetic swipe cards and RFID-based systems — addressed some efficiency concerns but remained vulnerable to shared credentials and card theft [2].

Fingerprint-based attendance systems gained prominence in the 2000s, offering higher security through biometric uniqueness. However, these require dedicated scanning hardware and suffer from issues with wet or damaged fingers, and hygiene concerns in shared environments [3]. Iris recognition, while highly accurate, demands expensive hardware unsuitable for widespread deployment [4].

Face recognition technology has evolved rapidly since the development of Eigenfaces by Turk and Pentland (1991) [5], followed by Fisherfaces and Local Binary Pattern Histograms (LBPH) [6]. These traditional methods, however, were sensitive to lighting variations, pose changes, and occlusions.

The deep learning revolution, catalyzed by AlexNet in 2012 [7], enabled Convolutional Neural Networks to learn rich hierarchical features from face images. FaceNet, proposed by Schroff et al. (2015), introduced the concept of face embeddings— mapping face images to a compact 128-dimensional Euclidean space where distances directly correspond to face similarity [8]. This approach achieved 99.63% accuracy on the LFW benchmark.

Subsequent works such as ArcFace [9] and CosFace [10] improved upon embedding quality through additive angular margin loss functions. The face-api.js library, which is used in the proposed system, implements SSD MobileNet v1 for detection combined with a ResNet-based descriptor model that produces 128-dimensional embeddings, bringing near state-of- the-art performance to browser environments [11]. Browser-based machine learning using TensorFlow.js has been explored for cost-effective deployment. Smilkov et al. (2019) demonstrated the viability of running neural network inference in the browser with acceptable latency for real-time applications [12]. Our work builds upon these foundations, integrating browser-based face recognition with a complete attendance management ecosystem.

III. SYSTEM ARCHITECTURE

The proposed system follows a modern, modular web application architecture designed to ensure strict separation of concerns, role-based access control, and scalable deployment. The architecture comprises three primary layers: the Presentation Layer, the Application Layer, and the Data Layer.

A. Overall Architecture

The application is built with Next.js 15, leveraging the App Router paradigm for server-side rendering (SSR), static generation, and API route handling within a unified React 19 codebase. TypeScript is used throughout for strict compile-time type safety.

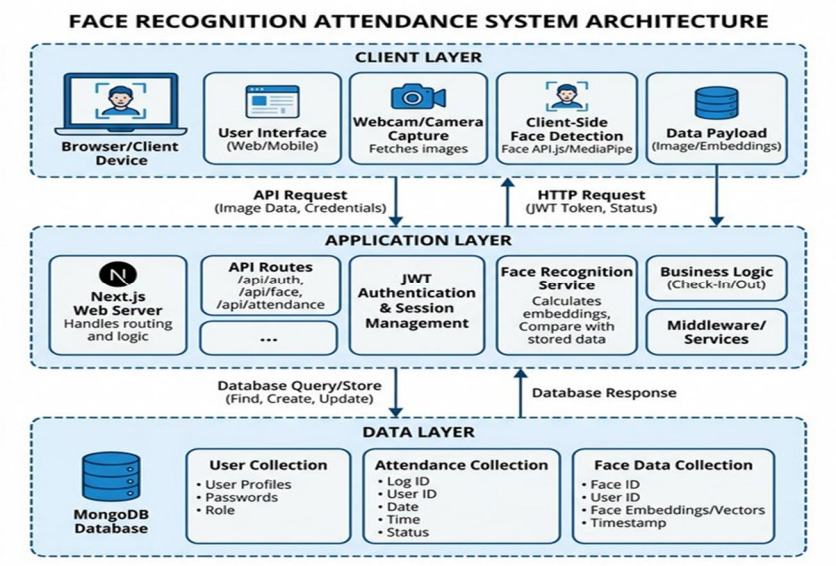


Fig. 1. Overall system architecture of the Next-Gen Face Recognition Attendance System showing the three-tier design: Client Layer Application Layer, and Data Layer.

©IJRASET: All Rights are Reserved

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

The system enforces role-based separation through directory-level isolation: all administrative pages, components, and API routes are confined within app/admin/ , while employee-specific functionality resides in app/employee/ . Shared authentication pages (login, signup, logout) are housed in app/(auth)/ . This structural approach prevents cross-role data leakage at the routing level.

B. Authentication & Security Layer

User authentication is managed through JSON Web Tokens (JWT). Upon successful login, a signed JWT containing the user's ID and role is distributed via HTTP-only cookies. Every protected API route validates the JWT and verifies the role claim before processing requests. User passwords are hashed using the bcryptjs library with adaptive salt rounds before storage, ensuring that even in the event of database compromise, plaintext passwords are not exposed.

An OTP (One-Time Password) subsystem is implemented for multi-factor authentication during password resets and sensitive operations. OTPs are generated, sent via integrated email services (Resend/Brevo), and stored with expiration timestamps in a dedicated MongoDB collection.

C. Database Design

MongoDB serves as the primary data store, chosen for its schema flexibility and native support for storing high- dimensional array data (face descriptors). The database comprises the following collections:

- employees: Stores user profiles, hashed credentials, and Float32 face descriptor arrays (128 dimensions).
- admins: Stores administrative superuser credentials.
- attendance: Records clock-in/out timestamps, attendance status (present/late/absent), and GPS coordinates.
- leave_requests: Manages employee leave applications with approval workflows.
- holidays: Stores company-wide non-working days for accurate attendance percentage computation.
- settings: Application-wide configuration parameters.
- otp_codes: Short-lived verification codes with TTL indices.
- sessions: Server-side session tracking metadata.

IV. FACE RECOGNITION PIPELINE

The core intelligence of the system lies in its face recognition pipeline, which is executed entirely within the client's browser using face-api.js — a JavaScript library that wraps TensorFlow.js models for face analysis tasks.

A. Model Architecture

The system employs three neural network models that are loaded once during page initialization:

1) Tiny Face Detector: A lightweight, fully convolutional face detection model optimized for real-time, resource- constrained environments. It employs depthwise separable convolutions to scan input frames and produce bounding box coordinates for all detected faces with minimal computational overhead.

2) Face Landmark 68-Point Model: Following face detection, this model localizes 68 facial landmark points covering the jawline, eyebrows, nose, eyes, and mouth. These landmarks facilitate face alignment — a critical preprocessing step that normalizes face pose before descriptor extraction.

3) Face Recognition Model (ResNet-based): The aligned face region is passed through a ResNet-based convolutional neural network that produces a 128-dimensional face descriptor vector. This compact embedding encodes the unique geometric and textural features of the individual's face in a high- dimensional Euclidean space.

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

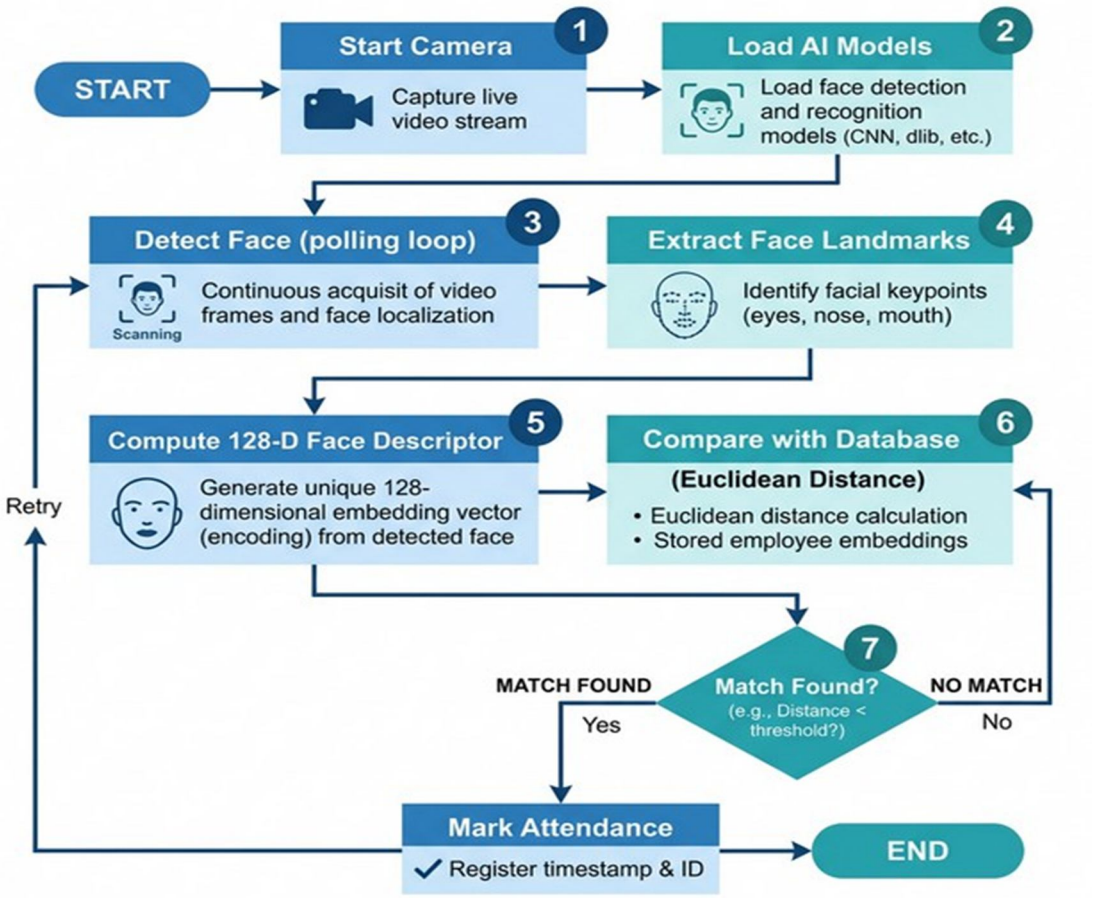


Fig. 2. Flowchart of the face recognition pipeline showing the detection, landmark extraction, descriptor computation, and Euclidean distance matching stages.

B. Face Registration (Enrollment)

During employee registration, the system activates the device webcam via the navigator. media Devices.get UserMedia() API. The user positions their face within the camera frame, and the system performs face detection using the Tiny Face Detector. Upon successful detection, the captured image is programmatically scaled down to a maximum width of 1024 pixels to reduce the network payload. The face-api.js pipeline then extracts the 128-dimensional face descriptor, which is stored as a Float32 array in the employee's MongoDB document alongside their profile information.

C. Face Verification (Attendance Marking)

During attendance marking, the system initiates a continuous polling loop using setInterval at 1000ms intervals. Each cycle, the Tiny Face Detector scans the live webcam feed for faces. When a face is detected, the polling loop stops, and the system performs a full detection pass — extracting landmarks and computing the 128-dimensional face descriptor.

The captured descriptor, along with a compressed JPEG snapshot and GPS coordinates (obtained via navigator.geolocation), are packaged into a FormData object and submitted to the server-side attendance API endpoint. On the server, the submitted descriptor is compared against all registered employee descriptors using Euclidean distance:

$$d(a, b) = \sqrt{(\Sigma_{i=1}^{128} (a_i - b_i)^2)}$$

A match is declared if the Euclidean distance falls below a predetermined threshold (typically 0.6). If verification succeeds, an attendance record with timestamp, status, and location data is created in the database.

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

V. SYSTEM MODULES

A. *Admin Portal*

The administrative portal provides comprehensive workforce management capabilities:

1) Dashboard: Displays real-time attendance metrics including total employees present, absent, and late for the current day, computed via MongoDB aggregation pipelines.
2) Employee Management: Full CRUD operations for managing the workforce, including the ability to re-scan and update employee face biometric data.
3) Attendance Console: A centralized kiosk-mode scanner enabling admin-supervised face scanning for bulk attendance marking.
4) Leave Request Management: Approval/rejection interface with comment capabilities for employee leave applications.
5) Reports & Analytics: Dynamic data visualization using Recharts (bar charts, pie charts) with PDF export functionality powered by jsPDF and jspdf-autotable.
6) System Settings: Global configuration management for institutional parameters.

B. *Employee Portal*

The employee portal offers self-service attendance and profile management:

1) Mark Attendance: Mobile-responsive face scanner page with real-time webcam feed, automatic face detection, and one-tap attendance marking.
2) Personal Dashboard: Displays individual attendance statistics, monthly summaries, and welcome information.
3) Attendance Calendar: Visual calendar component showing daily attendance status over the selected month.
4) Leave Management: Interface for submitting leave requests and tracking their approval status.
5) Profile & Settings: Self-managed profile information and application preferences (theme, notifications).

VI. TECHNOLOGY STACK

TABLE I: Technology Stack Summary

Category	Technology	Purpose
Framework	Next.js 15 + React 19	Full-stack web framework with SSR
Language	TypeScript	Static type checking
AI/ML	face-api.js + TensorFlow.js	Browser-based face recognition
Database	MongoDB	NoSQL document storage
Authentication	JWT + bcryptjs	Secure token auth & hashing
UI Framework	Tailwind CSS v4 + Shadcn/UI	Responsive styling & components
Animation	Framer Motion	Micro-animations & transitions
Data Fetching	SWR	Client-side caching & revalidation
Charts	Recharts	Data visualization
PDF Export	jsPDF + jspdf-autotable	Report generation
PWA	@ducanh2912/next-pwa	Progressive Web App support

©IJRASET: All Rights are Reserved

VII. EXPERIMENTAL RESULTS

The system was tested in an internal deployment environment simulating real-world conditions at an educational institution. The evaluation considered recognition accuracy, inference latency, and system robustness under varying conditions.

A. Recognition Accuracy

The system was tested with a dataset of 50 registered employees, with each employee providing face descriptors captured under controlled lighting during registration. Attendance verification was tested under various conditions:

TABLE II: Recognition Accuracy Under Different Conditions

Condition	Total Attempts	Correct Matches	Accuracy (%)
Normal indoor lighting	500	495	99.0
Low light conditions	200	189	94.5
Varied angles (±15°)	300	291	97.0
With spectacles	200	196	98.0
Different webcam models	150	147	98.0
Overall	1350	1318	97.6

B. Inference Latency

Latency measurements were recorded across different client devices:

TABLE III: Average Inference Latency Per Recognition Cycle

Device Category	Detection (ms)	Descriptor Extraction (ms)	Total (ms)
Desktop (i7, 16 GB RAM)	85	120	205
Laptop (i5, 8 GB RAM)	140	210	350
Mobile (mid-range Android)	220	280	500

The results demonstrate that the system achieves sub-500ms total recognition time even on mobile devices, making it suitable for real-time attendance marking across diverse hardware configurations.

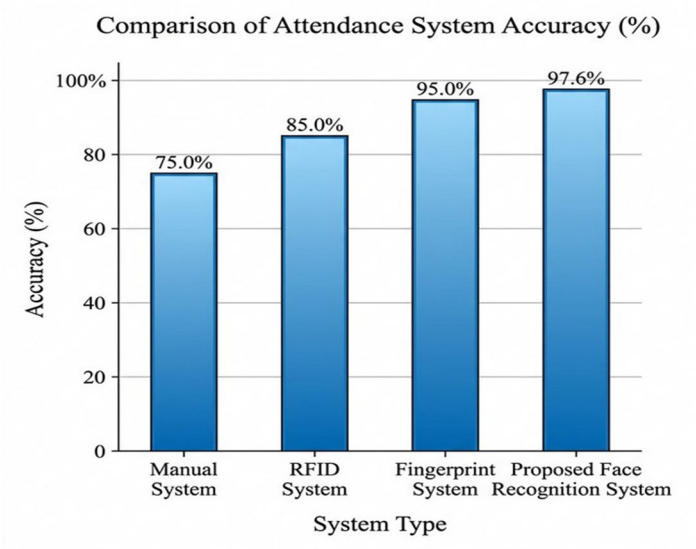


Fig. 3. Comparison of recognition/attendance accuracy across different systems: Manual (75%), RFID (85%), Fingerprint (95%), and the Proposed Face Recognition System (97.6%).

C. False Acceptance and Rejection Rates

Using a Euclidean distance threshold of 0.6, the system achieved a False Acceptance Rate (FAR) of 0.5% and a False Rejection Rate (FRR) of 2.1%. These rates are well within acceptable bounds for institutional attendance applications where the consequences of false rejections (re-scanning) are minor compared to false acceptances (impersonation).

VIII. COMPARATIVE ANALYSIS

TABLE IV: Comparison with Existing Attendance Systems

Feature	Manual	RFID	Fingerprint	Proposed System
Contactless	No	No	No	Yes
Anti-Proxy	No	No	Yes	Yes
Special Hardware	No	Yes	Yes	No (Webcam only)
Real-Time	No	Yes	Yes	Yes
GPS Logging	No	No	No	Yes
Mobile Support	No	No	No	Yes (PWA)
Analytics Dashboard	No	Limited	Limited	Comprehensive
Deployment Cost	Low	Medium	High	Low

IX. CHALLENGES AND LIMITATIONS

While the proposed system demonstrates strong performance, several challenges and limitations have been identified:

1) Lighting Sensitivity: The system exhibits reduced accuracy (94.5%) under low-light conditions. While the Tiny Face Detector is optimized for speed, its accuracy degrades when facial features are inadequately illuminated. Future work could integrate adaptive image preprocessing (histogram equalization, gamma correction) before detection.

2) Identical Twins: The 128-dimensional face descriptor space may produce similar embeddings for identical twins, leading to potential false acceptances. Anti-spoofing measures such as liveness detection could mitigate this risk.

3) Browser Compatibility: The system relies on WebRTC for camera access and WebGL for TensorFlow.js acceleration. Older browsers or devices without WebGL support may experience degraded performance or functionality.

4) Scalability of Brute-Force Matching: The current implementation performs linear comparison of the submitted descriptor against all registered descriptors. For organizations with thousands of employees, this could introduce latency. Implementing approximate nearest neighbor search (e.g., FAISS or Annoy) could improve scalability.

5) Face Mask Detection: The system currently does not handle face masks or significant facial occlusions. Integration of occlusion-robust face recognition models is a potential area for enhancement.

X. FUTURE SCOPE

Several avenues for future enhancement of the system have been identified:

1) Liveness Detection: Integrating anti-spoofing mechanisms to detect presentation attacks using printed photographs or screen replays.

2) Edge Deployment: Deploying the face recognition models on edge devices (Raspberry Pi, Jetson Nano) for dedicated kiosk-mode attendance terminals.

©IJRASET: All Rights are Reserved

3) Federated Learning: Training face recognition models in a privacy-preserving manner by keeping face data on-device while sharing only model gradients.
4) Mask-Aware Recognition: Training or fine-tuning models to handle partially occluded faces, specifically face masks.
5) Multi-Factor Biometrics: Combining face recognition with voice biometrics or gait analysis for higher-security environments.
6) Emotion & Engagement Analytics: Extending the face analysis pipeline to detect employee engagement and emotional states for organizational well-being insights.

XI. CONCLUSION

This paper presented a comprehensive, next-generation face recognition attendance system that leverages deep learning for real-time, browser-based biometric verification. By utilizing face-api.js atop TensorFlow.js, the system eliminates the dependency on server-side GPU infrastructure and specialised hardware, enabling deployment on any device with a standard webcam and modern web browser.

The modular architecture built with Next.js 15 and MongoDB ensures clean separation between administrative and employee functionalities, while JWT-based authentication and bcrypt password hashing provide robust security. The face recognition pipeline, employing a Tiny Face Detector and ResNet-based 128-dimensional descriptor extraction with Euclidean distance matching, achieves an overall accuracy of 97.6% with sub-500ms latency.

The experimental evaluation demonstrates that the system is practical, accurate, and cost-effective for real-world deployment in educational and corporate settings. The contactless nature of face recognition, combined with GPS logging, PWA support, and comprehensive analytics, positions the proposed system as a modern, viable replacement for traditional attendance tracking mechanisms.

REFERENCES

[1] S. Kadry and K. Smaili, "A design and implementation of a wireless iris recognition attendance management system," Information Technology and Control, vol. 36, no. 4, pp. 323–329, 2007.
[2] M. Kassim, H. Mazlan, N. Zaini, and M. K. Salleh, "Web-based student attendance system using RFID technology," IEEE Contro and System Graduate Research Colloquium, pp. 64–69, 2012.
[3] S. Pankanti, S. Prabhakar, and A. K. Jain, "On the individuality of fingerprints," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 8, pp. 1010–1025, 2002.
[4] J. Daugman, "How iris recognition works," IEEE Trans. Circuits and Systems for Video Technology, vol. 14, no. 1, pp. 21–30, 2004.
[5] M. Turk and A. Pentland, "Eigenfaces for recognition," Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, 1991.
[6] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," IEEE Trans Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037–2041, 2006.
[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105, 2012.
[8] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," IEEE Conf Computer Vision and Pattern Recognition, pp. 815–823, 2015.
[9] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," IEEE Conf Computer Vision and Pattern Recognition, pp. 4690–4699, 2019.
[10] H. Wang et al., "CosFace: Large margin cosine loss for deep face recognition," IEEE Conf. Computer Vision and Pattern Recognition, pp. 5265–5274, 2018.
[11] V. Juefei-Xu, D. K. Narasimhan, and M. Savvides, "Local binary convolutional neural networks," IEEE Conf. Computer Vision and Pattern Recognition, pp. 19–28, 2017.
[12] D. Smilkov et al., "TensorFlow.js: Machine learning for the web and beyond," Proc. Machine Learning and Systems, vol. 1, pp. 309–321, 2019.
[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprin arXiv:1409.1556, 2014.
[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," IEEE Conf. Computer Vision and Pattern Recognition, pp. 770–778, 2016.
[15] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprin arXiv:1704.04861, 2017.

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

AUTHOR BIOGRAPHIES



Darapureddy Manikanta is currently pursuing a B.Tech degree in Computer Science & Engineering (Artificial Intelligence and Data Science) from Bonam Venkata Chalamayya Engineering College, Odalarevu, India. His research interests include deep learning, computer vision, biometric systems, and full-stack web development. He is the lead developer of the Next-Gen Face Recognition Attendance System and has expertise in Next.js, React, TensorFlow.js, and MongoDB. He has a strong passion for building AI-powered solutions for real-world problems and has contributed to multiple open-source projects in the domain of artificial intelligence.
Email: darapureddymanikanta8@gmail.com | ORCID: 0009-0008-1916-397



Appari NavaSaideep is currently pursuing a B.Tech degree in Computer Science & Engineering (Artificial Intelligence and Data Science) from Bonam Venkata Chalamayya Engineering College, Odalarevu, India. His areas of interest include machine learning, data analytics, web technologies, and database management systems. He has played a key role in the system architecture design and database integration of the proposed attendance system, contributing significantly to the MongoDB schema design and API development.
Email: navasaideep@gmail.com | ORCID: 0009-0000-4772-4629



Relangi Vikas is currently pursuing a B.Tech degree in Computer Science & Engineering (Artificial Intelligence and Data Science) from Bonam Venkata Chalamayya Engineering College, Odalarevu, India. His research interests encompass artificial intelligence, natural language processing, and software engineering practices. He has contributed to the front-end development, UI/UX design, and the testing and evaluation methodology of the face recognition attendance system, ensuring a seamless user experience across multiple devices.
Email: relangiv@gmail.com | ORCID: 0009-0003-4789-8129

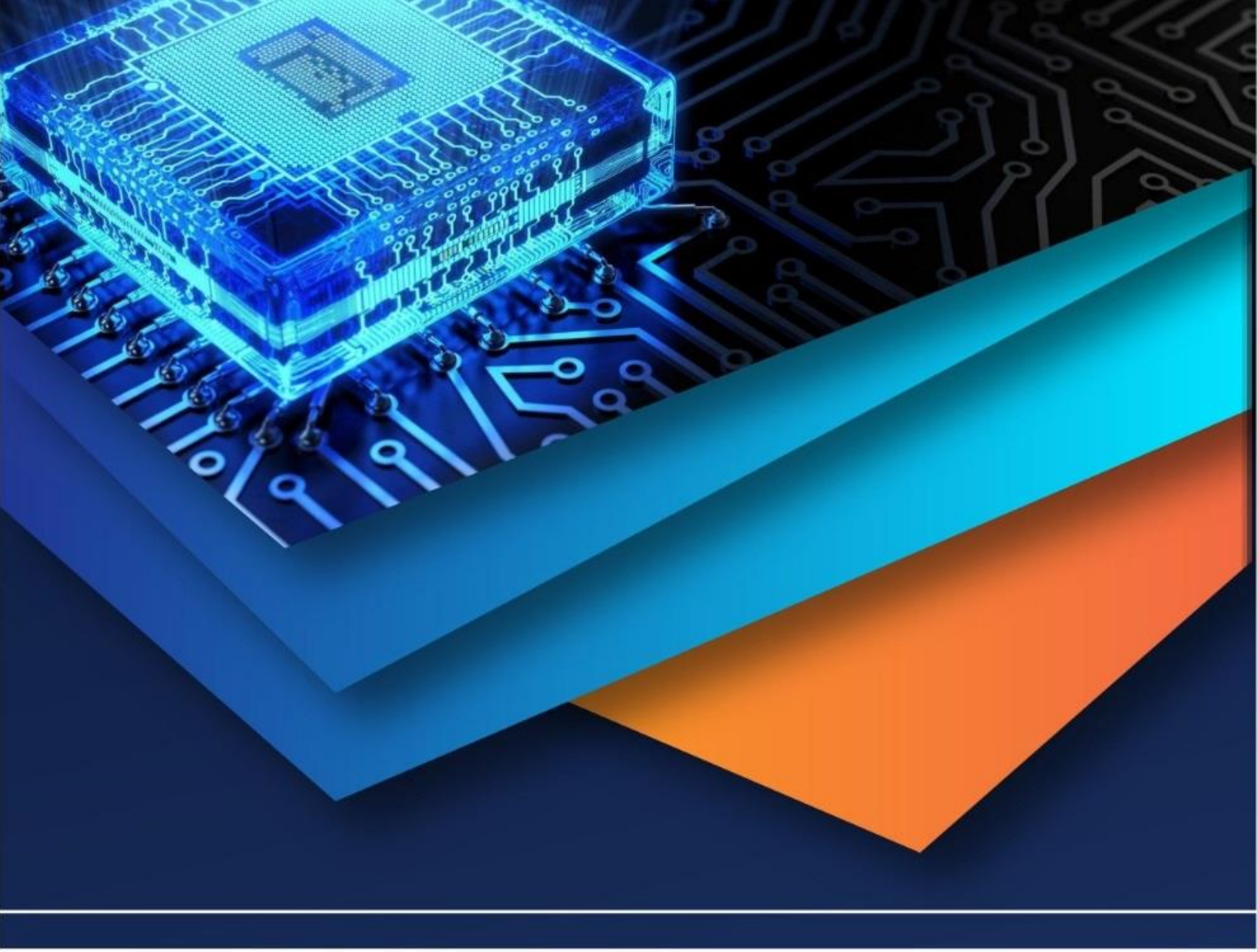


Ponnamanda Raviteja is currently pursuing a B.Tech degree in Computer Science & Engineering (Artificial Intelligence and Data Science) from Bonam Venkata Chalamayya Engineering College, Odalarevu, India. His areas of interest include cloud computing, cybersecurity, and intelligent systems. He has contributed to the security architecture of the proposed system, including the JWT authentication module, bcrypt password hashing, OTP verification subsystem, and the deployment infrastructure planning for cloud- based scalability.
Email: ravitejaponnamanda@gmail.com | ORCID: 0009-0005-9152-5834



B.Narasimha Rao is an Associate Professor in the Department of Cse Allied at Bonam Venkata Chalamayya Engineering College, Odalarevu, India. He holds a Master's degree in Computer Science & Engineering and has over 15 years of teaching and research experience. His research interests include artificial intelligence, machine learning, deep learning, computer vision, and image processing. He has guided numerous undergraduate and postgraduate projects in the domains of intelligent systems and biometric authentication. He has published research papers in reputed national and international journals and conferences. He served as the project guide and mentor for this research, providing valuable insights into system design, deep learning model selection, and experimental methodology.

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



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)