ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may2225

Web-Based Employee Timesheet Management System Using Face Recognition with Artificial Intelligence

Sanskruti Salunkhe[1]; Aditi Thombe[2]; Sushma Gunjal[3]; Nikhil Harde[4]; Chaitanya Kamble[5]

Prof.[3]

[1,2,3,4,5]Department of AI & DS, Ajeenkya D Y Patil, School of Engineering, Pune, India

Publication Date: 2026/06/10

Abstract: Managing employee attendance and work hours is a daily challenge for many organizations. Traditional methods like paper registers, punch cards, or swipe systems often lead to errors, wasted time, and even fraud such as proxy attendance. To solve these problems, this paper presents a smart, web-based system that uses artificial intelligence and facial recognition to automatically track when employees start and end their work. The system captures an employee's face through a standard webcam, verifies their identity using AI algorithms, and instantly records their check-in and check-out times. Built with HTML, CSS, JavaScript, Node.js, and MongoDB, the platform offers separate dashboards for employees and managers. Employees can see their own working hours, while administrators can generate detailed reports, monitor attendance trends, and manage employee data from anywhere. Testing shows that the system recognizes faces with over 95% accuracy, even under different lighting conditions. It eliminates manual paperwork, reduces administrative workload, prevents time theft, and works well for both office and remote teams.

Keywords: *Face Recognition, Artificial Intelligence, Timesheet Management, Attendance System, OpenCV, Web Application.*

How to Cite: Sanskruti Salunkhe; Aditi Thombe; Sushma Gunjal; Nikhil Harde; Chaitanya Kamble (2026) Web-Based Employee Timesheet Management System Using Face Recognition with Artificial Intelligence. *International Journal of Innovative Science and Research Technology*, 11(5), 3976-3980. https://doi.org/10.38124/ijisrt/26may2225

I. INTRODUCTION

In today's fast-paced work environment, keeping track of employee attendance and working hours is essential for payroll, productivity, and fairness. However, many companies still rely on old-fashioned methods such as signing paper sheets, using punch cards, or swiping ID cards. These approaches have serious drawbacks. Employees can mark attendance for absent colleagues (proxy attendance), manual entries often contain mistakes, and managers waste hours every week verifying records.

Fortunately, artificial intelligence (AI) and computer vision have matured enough to offer a better solution. Facial recognition technology, in particular, has become highly accurate and affordable. It allows systems to identify a person just by looking at their face through a simple webcam. This makes it ideal for attendance tracking because each person's face is unique and cannot be easily faked.

This paper describes a web-based employee timesheet management system powered by face recognition. The system automatically logs when an employee arrives and leaves, calculates total working hours, and stores everything in a centralized database. Employees can log in from any device with a browser and a camera. Managers can view real-time attendance, generate weekly or monthly reports, and export data for payroll processing. The main contributions of this work are:

- A real-time face recognition module that prevents proxy attendance.
- Role-based web dashboards for employees and administrators.
- Automated timesheet generation and report export (PDF/Excel).
- A scalable design that supports remote and hybrid work models.
- The rest of this paper is organized as follows. Section 2 reviews existing work in this area. Section 3 explains the system architecture. Section 4 describes the methodology and algorithms. Section 5 covers implementation details. Section 6 presents test results. Section 7 concludes and discusses future improvements.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may2225

II. LITERATURE SURVEY

Many researchers and developers have worked on face recognition for attendance systems. Viola and Jones [1] introduced a fast method called Haar cascade for detecting faces in real time. This became the foundation for many early systems. Later, Schroff and colleagues [2] developed FaceNet, a deep learning model that converts a face into a unique numerical code (embedding) and then matches it against stored codes. This approach achieved very high accuracy.

Zhang et al. [3] created MTCNN, a system that detects faces and identifies key points (eyes, nose, mouth) simultaneously. It works well even when lighting is poor or the person is tilting their head.

In 2025, Khandagale and his team [4] built a web-based attendance system using Haar cascades and LBP (Local Binary Patterns) with SQLite. Users found it easy to use, but it struggled with large numbers of employees. Another project reported in IRJET [5] used OpenCV and Dlib for recognizing multiple faces at once. It was fast, but performance dropped as the database grew. IJERT [6] achieved 99% accuracy on a standard dataset using Haar cascades and SVM, but poor lighting caused problems.

More recent work by IJRASET [7] combined Flask, face_recognition, and DeepFace libraries to create a complete web portal with an admin dashboard. They emphasized scalability and user-friendliness.

A systematic review by Khairnar et al. [8] concluded that deep learning models (CNNs) deliver over 98% accuracy for face recognition attendance, outperforming traditional methods. Hybrid approaches that combine CNNs with PCA (Principal Component Analysis) work even better [9]. Practical systems now include anti-spoofing measures like blink detection to prevent photos or videos from fooling the camera [10].

Despite these advances, most existing systems focus only on attendance marking. They do not integrate deeply with timesheet management, project tracking, or payroll. Our system bridges this gap by providing a complete, integrated platform that handles both authentication and full timesheet analytics.

III. PROPOSED SYSTEM

The proposed system is a complete web application that automates employee timesheet management using AI-based face recognition. There are three main types of users: employees, administrators, and the system itself.

➢ *System Architecture*

The system follows a modular design with separate layers for the user interface, business logic, AI processing, and data storage.

- *Frontend (User Interface Layer):*

Built with HTML, CSS, JavaScript, and React. Provides login screen, face capture via webcam, employee dashboard (view personal timesheet and working hours), and admin dashboard (manage employees, view attendance, generate reports).

- *Backend (Application Layer):*

Developed using Node.js and Express.js. Handles user authentication, API requests, face verification logic, and timesheet calculations.

- *AI Layer:*

Implements face detection using MTCNN and face recognition using FaceNet embeddings. This layer can run as a separate Python microservice or be integrated using face-api.js in Node.js.

- *Data Layer:*

MongoDB stores employee profiles, facial embeddings (encrypted), attendance logs, and timesheet records.

All communication between layers uses RESTful APIs secured with HTTPS. Role-based access control ensures employees can only see their own data, while administrators have full access.

➢ *Functional Modules*

The system is divided into four main modules:

- *Authentication Module:*

Employees log in with a username and password, and then the system asks them to look at the camera. Their face is captured, converted into an embedding, and compared with the stored embedding. Only if both the password and face match does the system grant access.

- *Timesheet Module:*

After successful authentication at the start of work, the system records the clock-in time. When the employee finishes work, they click "Clock Out," and the system records the clock-out time after a second face verification. The system automatically calculates total working hours, subtracts break times, and stores the entry.

- *Reporting Module:*

Generates daily, weekly, and monthly summaries. Reports can be exported as PDF or Excel files. They include employee name, date, clock-in/out times, total hours, and any overtime.

- *Employee Management Module:*

Administrators can add new employees, update their information, deactivate accounts, and assign roles or departments. During registration, the system captures multiple face samples to create a reliable embedding.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may2225

IV. METHODOLOGY

This section explains the step-by-step process from capturing a face to recording a timesheet entry.

➢ *Face Detection and Preprocessing*

When an employee starts the login process, the system accesses the device camera using JavaScript's get UserMedia() function. Each video frame is analyzed to find faces. We chose MTCNN [3] because it reliably detects faces even when lighting is not ideal or the person is looking slightly away from the camera. Once a face is detected, the system extracts the face region, adjusts brightness and contrast (histogram equalization), and resizes it to 160×160 pixels. These steps ensure that the face recognition model receives consistent input.

➢ *Face Recognition*

For recognizing faces, we use FaceNet [2]. This deep learning model converts a face image into a 128-number code called an embedding. Think of it like a unique fingerprint made of numbers. During employee registration, the system generates this embedding and stores it in the database.

During login, the system generates a new embedding from the live camera feed. It then compares this live embedding with the stored one using **cosine similarity**. Cosine similarity measures how close two number patterns are, from -1 (completely different) to +1 (identical). The formula is:

$$\textbf{similarity} = (\mathbf{A} \cdot \mathbf{B}) / (\|\mathbf{A}\| \times \|\mathbf{B}\|)$$

If the similarity score is above 0.6 (a threshold we found works well through testing), the system considers it a match and logs the employee in.

➢ *Anti-Spoofing and Liveness Detection*

To prevent someone from holding a photo or playing a video in front of the camera, we added liveness detection. The system asks the user to blink naturally. It counts blinks by tracking eye landmarks. If no blink is detected within a few seconds, the system rejects the attempt. This simple but effective method stops most spoofing attacks.

➢ *Timesheet Logic*

Once the system verifies the employee's identity at the start of the day:

- It records the current timestamp as clock_in_time.
- The employee can optionally select a project or task they are working on.
- At the end of the day, the employee clicks "Clock Out." The system again verifies their face.
- It calculates total_hours = clock_out_time - clock_in_time - break_duration.
- All data is stored in MongoDB with fields: employee ID, date, in-time, out-time, total hours, and project name.
- Administrators can review timesheet entries, approve them, or request changes before payroll is processed.

V. IMPLEMENTATION

➢ *Technology Stack*

Table 1 shows all the technologies used to build this system.

Table 1 Technology Stack

Layer	Technology / Tool
Frontend	React 18.3, TypeScript, Tailwind CSS, Axios
Backend	Node.js, Express.js, Passport.js
AI / Face Recognition	Python 3.9, OpenCV, face_recognition (dlib), MTCNN
Database	MongoDB Atlas (cloud)
Build Tools	Vite, TypeScript 5.6
Deployment	Docker, Nginx, AWS EC2

➢ *REST API Design*

The backend exposes several API endpoints. All require authentication and return JSON data. Table 2 lists the most important ones.

Table 2 REST API Endpoint Overview

Endpoint	Method	Module	Purpose
/api/auth/register	POST	Auth	Register a new employee and store face embedding
/api/auth/login	POST	Auth	Login with password and face verification
/api/timesheet/clockin	POST	Timesheet	Record clock-in time with face capture
/api/timesheet/clockout	POST	Timesheet	Record clock-out time with face verification
/api/timesheet/myhours	GET	Timesheet	Get logged-in employee's timesheet
/api/admin/employees	GET	Admin	List all employees (admin only)
/api/admin/report	GET	Admin	Generate attendance report in PDF/Excel
/api/face/verify	POST	AI	Verify face image against stored embedding

➢ *Database Schema*

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may2225

MongoDB stores data in flexible JSON-like documents. The main collections are:

- *Employees:*
  {_id, name, email, role, department, face_embedding (encrypted), created at}

- *Attendance:*
  {_id, employeeId, date, clock_in, clock_out, total_hours, status}

- *Timesheet:*
  {_id, employeeId, project, task_description, hours_worked, date}

Facial embeddings are encrypted using AES-256 before being stored. This ensures that even if the database is compromised, the biometric data remains safe.

➢ *Frontend Implementation*

The React application is organized into feature-based modules. The authentication screen includes a live webcam preview using the react-webcam library. When the user clicks "Verify Face," the app captures the current frame, converts it to base64 format, and sends it to the /api/face/verify endpoint. The employee dashboard shows a simple table of past attendance records and a button to clock in or out. The admin dashboard contains forms to add employees, a table of all attendance records, and a report generator with date range selection.

VI. RESULT ANALYSIS

We tested the system with 45 employees in a real office environment. Tests covered different lighting conditions (normal office light, low light, bright backlight), different accessories (with glasses, without glasses), and slight pose changes (looking slightly up or down).

➢ *Processing Speed and Reliability*

Table 3 shows how long each step takes and how often it succeeds.

Table 3 Processing Latency and Success Rate (45 trials)

Process Stage	Avg. Time (seconds)	Success Rate
Face Detection (MTCNN)	0.3	100%
Face Embedding Extraction	0.8	97.8% (44/45)
Similarity Matching	0.2	100%
Timesheet Update	0.1	100%
End-to-End Authentication	**1.4**	**97.8%**

Most failures happened when the camera could not capture a clear face due to very poor lighting. In such cases, the system simply asks the user to move to a better-lit area and try again.

➢ *Recognition Accuracy*

Table 4 breaks down accuracy by condition.

Table 4 Face Recognition Accuracy Under Different Conditions

Condition	Correct Identifications	Accuracy (%)
Normal Office Light	45/45	100.0
Low Light	42/45	93.3
Bright Backlight	43/45	95.6
Wearing Glasses	44/45	97.8
Slight Pose Change	42/45	93.3
Overall	**216/225**	**96.0**

The system achieved **96% overall accuracy**, which meets our target of over 95%. Accuracy was perfect under normal office lighting. Low light and pose changes caused most of the errors. In practice, these issues can be reduced by asking users to ensure good lighting and face the camera directly.

➢ *Impact on Administrative Work*

We asked HR staff to track how much time they spent on attendance and timesheet tasks before and after using the system. Before automation, they spent about 2 hours every day manually entering data, verifying attendance, and resolving discrepancies. After deploying our system, this dropped to about 15 minutes per day—an **85% reduction**. Payroll accuracy also improved because timesheet data was automatically verified by face recognition.

➢ *User Satisfaction*

We surveyed 10 employees and administrators. Table 5 shows their average ratings.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26may2225

Table 5 User Satisfaction Survey Results (1–10 scale, N=10)

Dimension	Average Score
Ease of Use	8.7
Authentication Speed	8.5
Accuracy of Attendance Log	9.2
Report Generation Utility	8.9
Overall Satisfaction	8.8

Most users found the system easy to use and appreciated not having to remember badges or punch cards. A few mentioned that face recognition took a second or two longer than swiping a card, but they accepted this trade-off for better security.

➢ *Key Takeaways*

- Face recognition effectively eliminates proxy attendance because each person's face is unique and verified live.
- FaceNet embeddings are robust enough to handle moderate lighting changes and glasses.
- A web-based solution works well for remote employees who can use their own laptops and webcams.
- Integrating timesheet management with face recognition saves HR teams significant time and improves payroll accuracy.

VII. CONCLUSION AND FUTURE SCOPE

This paper presented a complete web-based employee timesheet management system that uses AI-powered face recognition. The system automatically records clock-in and clock-out times after verifying the employee's identity through a webcam. It prevents proxy attendance, reduces manual data entry, and provides real-time attendance reports.

Our experiments showed that the system achieves 96% recognition accuracy and completes the entire authentication process in about 1.4 seconds. Users found it easy to use, and HR teams reported an 85% reduction in time spent on attendance-related tasks. The system works well for both in-office and remote employees, making it suitable for modern hybrid workplaces.

While the current system is fully functional, there are several ways to improve it:

- *Mobile Application:*
  Develop native apps for Android and iOS that can work offline. Employees could mark attendance even without an internet connection, and the app would sync later.

- *Advanced Anti-Spoofing:*
  Add infrared camera support or depth sensing (available on newer smartphones) to detect sophisticated spoofing attempts like high-quality masks or deepfake videos.

- *Emotion and Wellness Detection:*
  Extend the AI to detect signs of fatigue, stress, or low engagement. This could help managers identify employees who might need support.

- *Full Payroll Integration:*
  Connect the system directly with payroll software so that approved timesheets automatically trigger salary calculations, deductions, and bank transfers.

- *Cloud-Scale Deployment:*
  Use Kubernetes and serverless functions to handle thousands of concurrent users. This would allow large enterprises to adopt the system without performance concerns.

Overall, this project demonstrates that combining web technologies with AI-based face recognition can transform a routine administrative task into a secure, efficient, and transparent process. The same approach could be extended to other areas such as visitor management, exam proctoring, or secure facility access.

REFERENCES

[1]. Viola, P., Jones, M.J.: Robust Real-Time Face Detection. *International Journal of Computer Vision* 57(2), 137-154 (2004)
[2]. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A Unified Embedding for Face Recognition and Clustering. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815-823 (2015)
[3]. Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters* 23(10), 1499-1503 (2016)
[4]. Khandagale, A.S., et al.: Real-Time Face Recognition for Web-Based Attendance. *IRJET* 12(3), 245-250 (2025)
[5]. IRJET: Automated Attendance System using OpenCV and Dlib. *IRJET* 11(4), 123-128 (2024)
[6]. IJERT: High Accuracy Face Recognition for Attendance. *IJERT* 11(6), 567-572 (2022)
[7]. IJRASET: Web Portal for Face Recognition Attendance with Flask. *IJRASET* 13(1), 88-94 (2025)
[8]. Khairnar, S., Gite, S., Kotecha, K., Thepade, S.D.: Face Liveness Detection Using AI: A Systematic Literature Review. *Big Data and Cognitive Computing* 7(1), 37 (2023)
[9]. Turk, M., Pentland, A.: Eigenfaces for Recognition. *Journal of Cognitive Neuroscience* 3(1), 71-86 (1991)
[10]. Li, H., et al.: AI Face Recognition and Processing Technology Based on GPU Computing. *Journal of Theory and Practice of Engineering Science* 4(5), 9-16 (2024)