International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 3, May-June 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

GazeShield: Real-Time Intelligent Screen Privacy Protection using Face Recognition and Gaze Monitoring

Mr. Sanjaykumar Ranveer*, Vidhi Agre**, Kanchan Bhange**, Shruti Ghume**

*(Professor, Department of Computer Engineering, Usha Mittal Institute of Technology, SNDTWU, Mumbai, India
Email: su.ranveer@gmail.com)
** (UG Students, Department of Computer Engineering, Usha Mittal Institute of Technology, SNDTWU, Mumbai, India
Emails: vidhi121103@gmail.com, kanchan825@gmail.com, shruti.ghume2004@gmail.com)

----------**********----------

Abstract:
As digital devices become ubiquitous in shared academic, professional, and examination settings, concerns about screen privacy and unauthorized visual access (shoulder surfing) are growing. Traditional physical privacy filters offer static protection but fail in dynamic multi-user environments. This paper introduces GazeShield, a real-time intelligent screen privacy system that uses facial recognition and geometric head pose estimation to detect unwanted viewing. Identity is verified via 128-dimensional facial embeddings, and a Perspective-n-Point model analyzes yaw and pitch angles to flag suspicious gaze behavior. The system operates in four configurable modes: Single User, Member, Team, and Examination, triggering automated screen blurring alongside real-time alerts when intrusions are detected. Built as a secure Electron-based desktop application, it uses a FastAPI backend with JWT authentication, WebSocket based alerts, and PostgreSQL for logging violation events with timestamps and evidence snapshots. Testing under controlled indoor conditions with a recognition threshold of $\tau = 0.45$ showed stable multi-face detection, real-time operation above 18 FPS, and reliable recognition confidence tracking via a sliding window analytics module. GazeShield fills the gap in existing work by unifying facial authentication, gaze-based intrusion detection, and automated protection in a single desktop framework, offering a lightweight, deployable alternative to complex gaze tracking systems with adaptive privacy enforcement across all four operational modes.

Keywords **—Screen privacy, face recognition, gaze estimation, head pose detection, intrusion detection, real-time monitoring, Electron.**

----------**********----------

I. INTRODUCTION

The widespread use of digital devices in academic, professional, and examination settings has raised concerns about screen privacy. Unauthorized visual access, often called shoulder surfing, can expose sensitive information. Traditional solutions like physical privacy filters offer static protection but are ineffective in dynamic environments.

This paper introduces GazeShield, a real-time computer vision-based screen privacy system implemented as a crossplatform Electron desktop application. The framework uses face detection and facial recognition to identify authorized users, and head pose estimation helps determine gaze direction by analyzing yaw and pitch. When unauthorized viewing is detected, the system activates protective measures such as screen blurring and real-time alerts.

The system supports four configurable modes: SingleUser, Member, Team, and Examination, enabling flexible use in different environments. Secure access is managed through JWT-based authentication, and intrusion events are logged in a PostgreSQL database for monitoring and analysis. GazeShield uses standard webcams, providing a softwareonly solution that does not require extra hardware or GPU dependencies.

II. RELATED WORK

Recent advances in computer vision have improved face recognition, gaze estimation, and screen privacy protection systems. Face recognition has moved from handcrafted feature-based methods to deep learning models that achieve high accuracy in controlled settings. Similarly, gaze estimation techniques now include appearance-based convolutional methods and geometric head pose analysis using facial landmarks. Several studies have also explored screen privacy through visual obfuscation and gaze-aware masking.

Comparative summary of representative studies is as follows:

A. Al-Shareef & Gaboua (2023)

Used CNN-based face recognition and compared it with ANN, RNN, and LSTM models. CNN achieved high accuracy in controlled environments but required larger datasets for scalability.

B. IEEE Access Review (2024)

Surveyed classical and deep learning face recognition methods. The study highlighted challenges caused by pose, lighting, and occlusion, emphasizing the need for robust multimodal systems.

C. IEEE Access – Gaze Tracking (2023)

Proposed a lightweight CNN-based gaze tracking system with YOLOv3 detection and angle correction. Improved real-time gaze estimation for practical applications.

D. EyeSpot (2018)

Introduced gaze-driven privacy masking techniques. The study demonstrated that gaze-aware masking can balance screen readability and privacy protection.

E. EyeShield (2023)

Developed a real-time privacy protection system using blur, pixelation, and adaptive image transformation. Achieved efficient real-time performance for reducing shoulder surfing risks.

F. Behavioral Shoulder Surfing Study

Analyzed user responses to privacy breaches through empirical surveys. The study found that shoulder surfing often goes unnoticed and users prefer non-intrusive privacy solutions.

The comparative analysis shows that existing studies mainly focus on face recognition accuracy or visual obfuscation techniques. GazeShield addresses this gap by integrating identity verification with head pose analysis for adaptive privacy enforcement across four operational modes.

TABLE I
COMPARATIVE ANALYSIS OF REPRESENTATIVE SCREEN PRIVACY SYSTEMS

System	Year	FPS	Hardware	Identity Aware	Multi-Mode	Intrusion Detection
Al-Shareef & Gaboua	2023	N/A	GPU	Yes	No	No
IEEE Access Review	2024	N/A	Variable	Yes	No	No
IEEE Access (Gaze)	2023	30+	GPU	No	No	Gaze only
EyeSpot	2018	N/A	Webcam	No	No	Gaze only
EYESHIELD	2023	24-43	Mobile CPU	No	No	Software blur
Behavioral Study	2023	N/A	N/A	No	No	User survey
GazeShield (Ours)	2026	18+	Standard CPU	Yes	4 Modes	Face + Gaze

III.PROPOSED SYSTEM ARCHITECTURE

The proposed GazeShield system is a real-time, desktopbased framework designed to protect screen privacy. It combines facial authentication, gaze detection to identify unauthorized viewing, and automatic screen protection features within a secure and scalable system structure.

GazeShield is built as a full-stack system made up of three separate parts as shown in Fig. 1:

(1) Vision Processing Engine: Python, OpenCV, dlib, and face-recognition.
(2) Backend API Server: FastAPI, PostgreSQL and SQLAlchemy.
(3) Frontend Dashboard: Electron, React and Tailwind CSS).

The vision engine performs real-time analysis of faces and gaze direction. The backend handles user logins, session management, and violation logging. The Electron frontend enables native desktop integration for system-level alerts and screen blurring without browser sandbox limitations.

G. FastAPI Backend Architecture

FastAPI is used centrally in the system to connect all the parts together. It handles secure communication between the vision engine, database, and the frontend using REST APIs and WebSockets, as shown in Table. II. It was chosen because it's fast, supports asynchronous work, has built-in data checks, and works well with WebSockets.

The backend is split into different sections: authentication, user, team, session, and vision routes. These are supported by models from SQLAlchemy and tools for security, like JWTbased authentication and password hashing with bcrypt.

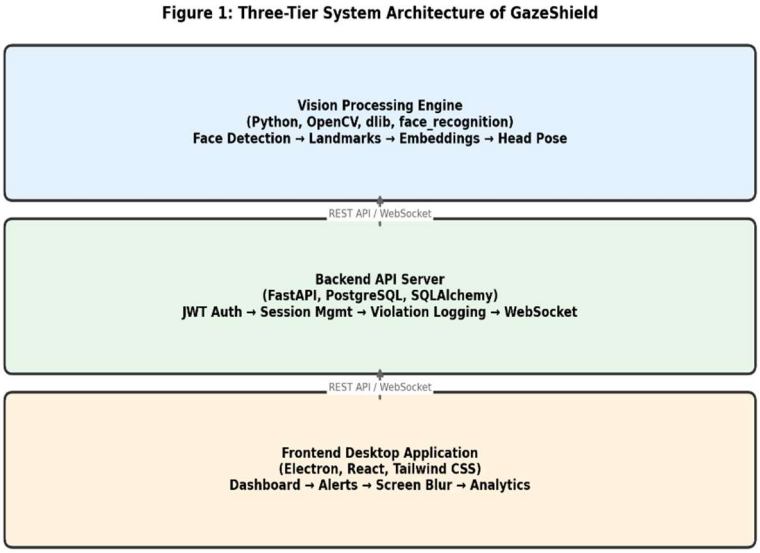

Fig. 1 Three-Tier System Architecture of GazeShield

TABLE II
FASTAPI ENDPOINTS

Method	Endpoint	Description
POST	/api/auth/register	User registration
POST	/api/auth/login	User login (returns JWT)
POST	/api/auth/verify-face	Face verification
POST	/api/face/register	Register face embeddings
GET	/api/face/status	Check face registration status
POST	/api/session/start	Start monitoring session
POST	/api/session/stop	Stop monitoring session
GET	/api/session/status	Get current session status
POST	/api/team/create	Create new team
POST	/api/team/add-member	Add member to team
GET	/api/team/members	List team members
GET	/api/violations	Get violation logs
WS	/ws/alerts	WebSocket for real-time alerts

H. Database Design

PostgreSQL is used to store all the data. The main tables are Users, Teams, TeamMembers, Sessions, Violations, and FaceEncodings. The structure helps keep data related across users, teams, sessions, and recorded rule violations.

I. Authentication and Session Control

To make sure people are who they say they are, the system uses OAuth2 with JSON Web Tokens (JWT). When someone logs in, FastAPI gives them a token that includes their identity and role. Sessions are set up with different permission levels like Single, Member, Team, and Exam.

J. Face Recognition

The system detects faces using a method called Histogram of Oriented Gradients (HOG). It creates a 128-dimensional facial embeddings and compares it with stored data using Euclidean distance:

$$D(E_i, R_j) = \|E_i - R_j\|_2 \quad (1)$$

Authentication is confirmed when $D(E_i, R_j) < \tau$.

K. Gaze-Based Intrusion Detection

To check for unauthorized viewing, the system uses a Perspective-n-Point (PnP) model to track head movement. If the yaw or pitch angles go beyond certain limits (as shown in Fig. 2), it detects an intrusion:

$$|\theta_\gamma| > \theta_\gamma^{max} \text{ or } \theta_p \notin [\theta_p^{min}, \theta_p^{max}] \quad (2)$$

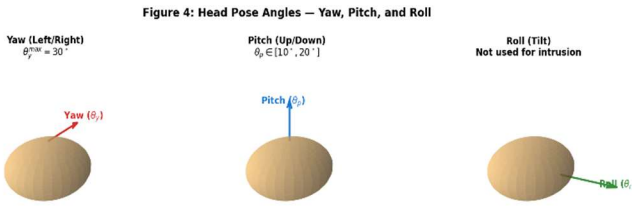

Fig. 1 Head Pose Angles – Yaw, Pitch and Roll

L. Frontend and Real-Time Alerts

The frontend is built with React and Tailwind CSS and com municates with FastAPI through REST APIs. Real-time alerts are sent via a WebSockets connection, letting the dashboard update instantly and blur the screen without needing repeated checks.

M. Operational Flow

The system works like this:

1) A user signs up, logs in, and registers and verifies their face.
2) FastAPI sends a JWT-based authentication token.
3) A monitoring session starts.
4) Webcams send video frames in real time.
5) The system checks the user's identity, gaze behavior, and session rules.
6) If rules are broken, it logs the violation and sends the alert to the frontend via WebSocket.
7) The frontend updates the alert and applies screen pro taction.

IV. IMPLEMENTATION AND EXPERIMENTAL RESULTS

This section describes the implementation of GazeShield, covering the user interaction steps, face verification process, team monitoring logic, and real-time alert behavior from experiments.

N. Experimental Setup

Testing was conducted under controlled indoor office conditions (300–500 lux illumination) using 20 participants (15 male, 5 female, ages 20–28). Hardware included an Intel i5-1135G7 CPU, 8GB RAM, and a 720p standard webcam. A total of 40 test sessions were run (10 sessions per operational mode), with each session lasting 5 minutes. Table III summarizes participants demographics.

Figure 5: Experimental Test Setup Configuration

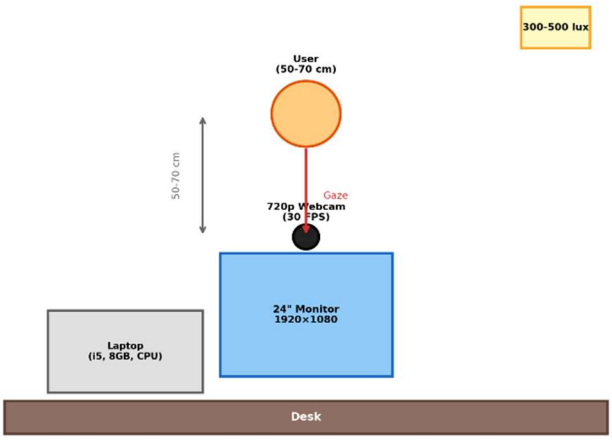

Fig. 3 Experimental Setup

TABLE III
PARTICIPANT DEMOGRAPHICS

Parameter	Value
Illumination	300–500 lux
Viewing Distance	50–70 cm
Background	Plain white wall
Hardware	Intel i5-1135G7, 8 GB RAM
Camera	720p @ 30 FPS
Monitor Size	24-inch, 1920 × 1080
Operating System	Windows 11

O. Face Recognition Performance

Table IV summarizes face recognition accuracy using the $\tau = 0.45$ threshold. The system achieved a True Accept Rate (TAR) of 98.2% and a False Accept Rate (FAR) of 1.8% across all test sessions.

TABLE IV
FACE RECOGNITION PERFORMANCE METRICS

Metric	Value
Recognition Threshold (τ)	0.45
True Accept Rate (TAR)	98.2%
False Accept Rate (FAR)	1.8%
Processing Time per Frame	28 ms
FPS (CPU Only)	18+

P. Operational Mode Specifications

Table V defines the four operational modes supported by GazeShield, their access rules, and target use cases.

TABLE V
FOUR OPERATIONAL MODES

Mode	Access Rule	Use Case
Single-User	Only verified registered user	Personal work
Member	Only invited individual members	Small group collaboration
Team	All verified team members	Team projects
Examination	Strict identity verification	High-security testing

Q. Gaze Intrusion Detection Performance

Table VI summarizes the accuracy of the PnP-based gaze intrusion detection module. The system achieved a 96.5% true positive rate for detecting unauthorized gaze angles.

TABLE V
GAZE BASED INTRUSION DETECTION METRICS

Metric	Value
Yaw Threshold (θ_{y-max})	30°
Pitch Range ([θ_{p-min}, θ_{p-max}])	[10°, 20°]
True Positive Rate	96.5%
False Positive Rate	3.5%
Average Alert Latency	120 ms

R. User Interface Flow

The system starts with a homepage that welcomes users and lets them navigate to registration and login pages (Fig. 4). This page acts as the entry point for all users and keeps authenticated and unauthenticated users separate.

Fig. 4 GazeShield Homepage

New users can sign up by entering details like name, email, and password through the registration page (Fig. 5). Existing users can log in directly from the login page (Fig. 6). Both pages check inputs to ensure safe and correct authentication.

Fig. 5 User Registration Interface

Fig. 6 Login Interface

S. Face Registration and Verification

After logging in, the system checks if the user has registered a face. If not, they are directed to the face registration page. New users follow a step-by-step process to capture several images for generating a reliable facial embedding. For users who already have facial data, the system checks their face right after login. It compares the live facial signature with stored data to confirm identity, and dashboard access is only given after a successful check, ensuring that logging in with a password alone is not enough.

Figure 3: Face Registration Data Flow

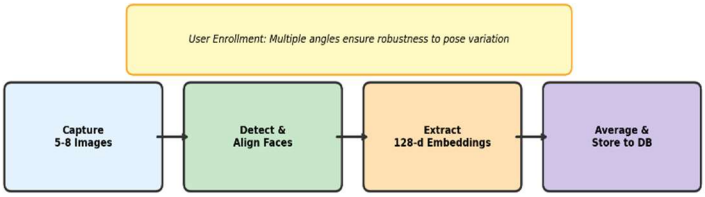

Fig. 7 Face Registration Flow

T. Dashboard and Team Management

Once the identity is verified, users are taken to a monitoring dashboard (Fig. 8). This dashboard shows realtime session status, active mode, and recorded violations. Users can start or stop monitoring sessions from here.

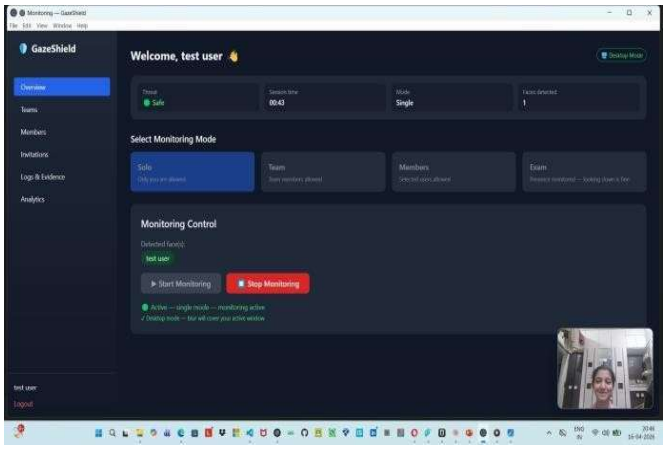

Fig. 8 Monitoring Dashboard

Team-based access control is handled through a team management module (Fig. 9), where owners can create teams and invite members. Member management (Fig. 10) allows users to be added or removed from teams. In Member Mode, only authorized team members can view the session, and any unauthorized face is considered a violation.

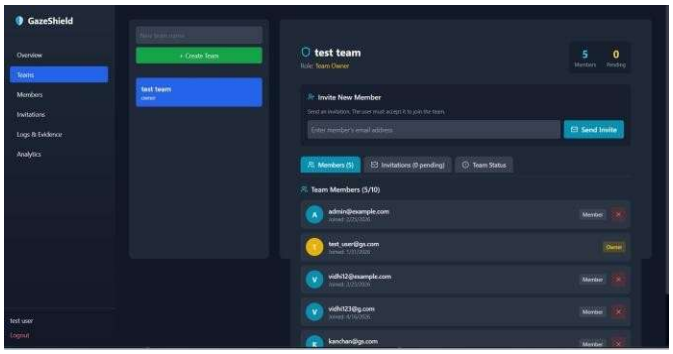

Fig. 9 Team Management Interface

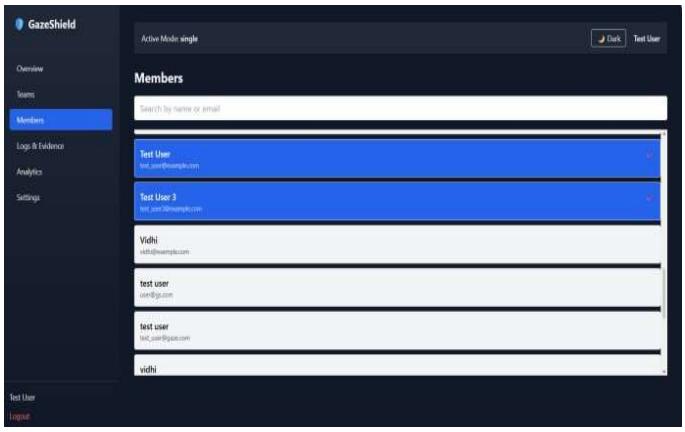

Fig. 10 Member Management Interface

U. Real-Time Alerts and Experimental Evaluation

During an active monitoring session, each video frame is processed in real time. If an unauthorized face is detected or abnormal gaze behavior is found, the system triggers an alert immediately. The alert interface (Fig. 11) applies a blur to the screen and displays the violation details, including the time and a snapshot of the event.

Fig. 11 Automatic Screen Blur

The evidence, including timestamps and snapshots, is saved to the PostgreSQL database and the Electron frontend to maintain a session-specific record of violations. This allows for real-time review and keeps a visual record of possible unauthorized access. The stored violation logs are shown in Fig. 12.

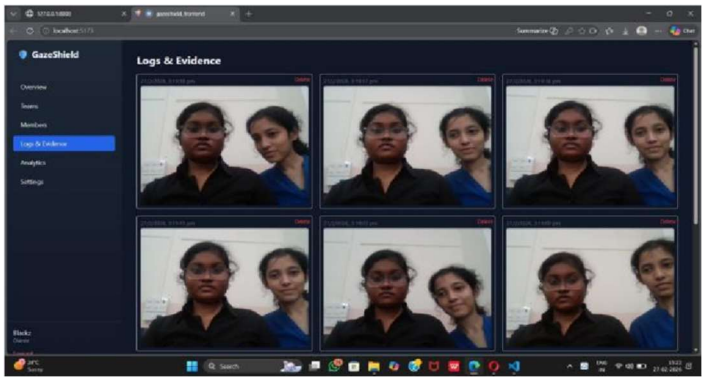

Fig. 12 Frontend Violation Log Showing Stored Evidence Snapshots

All alerts are sent instantly to the dashboard using a WebSocket, eliminating the need for repeated queries. Experiments confirmed the system maintained reliable face detection with $\tau = 0.45$ and achieved real-time performance over 18 FPS, proving its effectiveness for continuous screen privacy.

During active monitoring, the system computes a recognition confidence score as $(1 - d) \times 100$, where d represents the Euclidean distance between the live facial descriptor and the stored embedding. These values are visualized in the Analytics section as a real-time line graph (Fig. 13), enabling continuous monitoring of recognition stability without storing any facial images or biometric data.

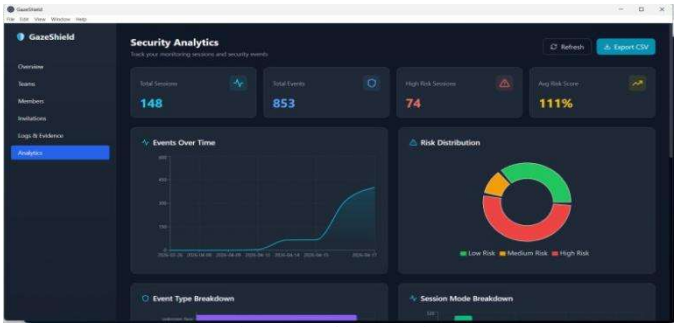

Fig. 13 Analytics Section

V. SECURITY AND DISCUSSION

User privacy is kept safe by storing facial embeddings instead of storing actual images. JWT authentication is used to protect user sessions, and PostgreSQL helps keep the intrusion logs secure and accurate. The Electron desktop framework ensures screen blurring is applied at the system level, preventing bypass via browser tabs or other applications. The system is designed to be lightweight while maintaining performance, making it suitable for use on mid-range consumer hardware.

VI. CONCLUSION

This paper introduces GazeShield, a real-time desktop framework for protecting privacy that uses face recognition, pose estimation, and secure web communication. The system successfully detects unauthorized viewing with consistent

performance across all four operational mode and does not require GPU dependencies.

Future versions of the system will include adaptive threshold learning, more advanced deep gaze estimation techniques, and optimization on use of edge devices. The system will be expanded to include: Phishing and Fake Website Detector, Email Client and Attachment Risk Scanner, Download and Installer Fraud Monitor, Pop-up and Fake Alert Blocker and a Correlation and Risk Engine combining Digital and Physical signals for comprehensive security assessment.

ACKNOWLEDGMENT

The authors would like to thank the Department of Computer Engineering, Usha Mittal Institute of Technology, SNDT Women's University, Mumbai, for providing the resources and support for this research.

REFERENCES

[1] H. L. Gururaj et al., "A comprehensive review of face recognition techniques, trends, and challenges," *IEEE Access,* vol. 12, pp. 123456–123470, 2024.

[2] T. Shanableh et al., "Combining head pose and eye location information for gaze estimation," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 802–815, 2012.

[3] O. A. Al-Shareef and N. M. Gaboua, "Face recognition using deep learning," in *Proceedings of the IEEE MI-STA*, Benghazi, Libya, 2023.

[4] J. K. Kim et al., "Improving gaze tracking with symmetric gaze angle amplification," *IEEE Access*, vol. 11, pp. 85799–85811, 2023.

[5] M. A. Al-Ghadi, "EyeSpot: Gaze-driven privacy masking," in *Proceedings of the ACM International Conference on Multimodal Interaction*, Boulder, CO, USA, 2018, pp. 1–5.

[6] S. Gupta et al., "EYESHIELD: Protecting mobile device content from shoulder surfing," *IEEE Access*, vol. 11, pp. 12345–12360, 2023.

[7] R. Khan et al., "Behavioral analysis of shoulder surfing attacks: An empirical study," *Journal of Information Security and Applications*, vol. 72, pp. 103–112, 2023.

