# AI-Powered Online Exam Proctoring System for Secure and Scalable Assessment

**Sakshi Manoorkar[1], Pallavi Dhuppe[2], Pratiksha Kadam[3] Rakhi Dumne[4]**

*[1, 2, 3] B. Tech Student Dept. of Computer Science and Engg., MGM's College of Engineering, Nanded*
*[4] Guide, Asst. Prof., Dept. of Computer Science and Engg., MGM's College of Engineering, Nanded*

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

**Abstract -** The rise of remote education has introduced new challenges in maintaining academic integrity during online examinations. Traditional methods of in-person or manual remote proctoring are either impractical or resource-intensive, leading to increased incidents of academic dishonesty. This project proposes an AI-powered online proctoring system capable of monitoring student behaviour in real time to ensure fair and secure assessment environments.

The system integrates facial recognition, eye gaze tracking, voice activity detection, and browser tab monitoring to detect behaviours such as looking away, multiple face presence, or window switching. Built using MediaPipe FaceMesh, OpenCV, and TensorFlow, it provides real-time anomaly detection and alerting. A Python-based backend coordinates proctoring logic, while Streamlit and React.js power the examiner and student interfaces. Detected events are logged to Cloud Firestore and accompanied by automated screenshots, enabling scalable, post-exam review. This framework offers academic institutions a robust, scalable, and automated approach to uphold integrity in digital assessments.

***Key Words*:** Academic Integrity, AI-based Monitoring, Eye Gaze Tracking Online Proctoring.

## 1. INTRODUCTION

The COVID-19 pandemic has accelerated the global shift toward remote education, fundamentally transforming how learning and assessments are conducted. While platforms like Microsoft Teams and Google Classroom have enabled the continuation of academic instruction, examinations—a critical component of education—have faced significant challenges. Traditional in-person exam proctoring has proven impractical in this digital age, leading institutions to explore online exam formats that are often vulnerable to academic dishonesty.

To uphold academic integrity in remote settings, many institutions have experimented with alternatives such as open-book formats or assignment-based evaluations. However, these solutions are not always effective or scalable. Manual remote proctoring, where human invigilators monitor students via video feeds, is resource-intensive and difficult to implement on a large scale. This highlights the urgent need for an automated, intelligent, and scalable exam supervision solution.

In response to this challenge, we propose an AI-driven online proctoring system that leverages computer vision, facial recognition, gaze tracking, voice detection, and browser activity monitoring to ensure secure and fair assessments. The system uses standard webcams and microphones—commonly available on laptops and smartphones—eliminating the need for specialized hardware. By continuously monitoring the test-taker's behavior, detecting anomalies such as looking away, multiple faces, or suspicious audio, the system provides real-time alerts and captures evidence, greatly enhancing exam integrity.

Our approach is grounded in the integration of advanced tools and frameworks such as **MediaPipe for facial landmark detection, OpenCV for real-time video processing, and TensorFlow for model development**. A user-friendly interface built with **Node.js and React.js** ensures seamless interaction for both students and invigilators, while **Cloud Firestore** serves as a reliable backend for storing exam logs and suspicious activity reports.

This AI-powered system is designed to be adaptable, scalable, and effective in mitigating cheating behaviors during online assessments. It represents a transformative step towards modernizing exam supervision, offering educational institutions a robust and trustworthy alternative to traditional proctoring methods in the evolving digital learning landscape.

## 2. METHODOLOGY

The methodology adopted for the AI-based proctoring system is structured around real-time video analysis, behavioral event logging, and automated alerting. The system ensures academic integrity by monitoring students' presence, facial orientation, and gaze behavior using advanced computer vision and machine learning tools. This section details the core modules—face detection, gaze estimation, anomaly tracking, alert generation, and examiner interface—and their integration in a modular and scalable architecture.

### 2.1. Facial Recognition and Presence Detection

To authenticate and continuously monitor the examinee, the system uses **MediaPipe FaceMesh**, which detects 468 facial landmarks per frame. The video stream is processed in real-time, ensuring that the examinee's face remains present and singular throughout the test duration.

Processing Pipeline:

*a.* Real-time webcam stream captured using **OpenCV**.

*b.* Conversion of BGR frames to RGB for MediaPipe compatibility.

*c.* Landmark inference using mp.solutions.face_mesh.FaceMesh.

*d.* Validation:

- If no landmarks are detected for over 2 seconds → “No Face” alert.
- If more than one face is detected → “Multiple Faces” alert.

This module guarantees identity continuity and flags unauthorized interruptions.

### 2.2. Gaze Estimation

Gaze behavior is analyzed to detect prolonged deviations away from the screen. The system computes gaze direction by comparing the spatial relationship between eye landmarks and facial reference points like the nose.

- Gaze Angle Calculation:

```
left_eye = face_landmarks.landmark[33]
 right_eye = face_landmarks.landmark[263]
nose_tip = face_landmarks.landmark[1]
angle = calculate_angle((left_eye.x,
left_eye.y), (right_eye.x, right_eye.y))
```

A calibrated threshold (e.g., >15° deviation for >3 seconds) is used to infer intentional gaze aversion. Alerts are raised if the condition persists, suggesting possible use of external resources.

### 2.3. Behavioral Anomaly Detection

A **rule-based detection system** identifies suspicious activities, which include:

- Gaze aversion
- Head turning
- Face absence
- Presence of multiple individuals

To reduce false positives, a **temporal filter** ensures that only sustained anomalies (lasting beyond a defined time window) trigger alerts. This component is designed to be extensible with supervised learning models in future iterations.

### 2.4. Real-Time Alert and Logging System

The system logs all flagged events with metadata including:

- Timestamp
- Event type (e.g., “Gaze Left”, “No Face”)
- Screenshot for examiner review
- Device/user identifier

Screenshots are captured at the moment of anomaly and stored securely using **Google Cloud Storage or Firebase.** Alerts are simultaneously streamed to the examiner dashboard.

### 2.5. Modular Examiner Dashboard

The examiner interface, built with Streamlit, provides real-time insights into student activity and anomaly detection. Features include:

- Live logs and alert stream
- Screenshot review interface
- Manual override of alerts (e.g., dismissing false positives)
- This dashboard empowers instructors to maintain oversight without constant video surveillance.

### 2.6. System Integration and Deployment

The architecture is modular, scalable, and cloud-compatible:

- **Frontend:** React.js enables interactive user experience and camera access.
- **Backend:** Python-based logic handles processing, detection, and database operations.
- **Database:** Event logs and metadata stored in Firestore.
- **Deployment:** Containerized using Docker for local or cloud-based deployment with Firebase integration for real-time data sync.

**Deployment Advantages:**

- Platform-independent
- Supports role-based access

Auto-scalable via cloud services (Google Cloud).

## 3. SYSTEM ARCHITECTURE AND DESIGN

The proposed system is a hybrid, AI-powered online exam proctoring solution designed to ensure academic integrity by continuously monitoring student behavior through webcam feeds. It integrates real-time video processing, AI-based facial and gaze analysis, and event-driven logging into a cohesive and modular architecture. Each component within the architecture is tailored to perform specific tasks such as user authentication, live anomaly detection, and examiner interaction. The architecture supports seamless deployment, scalability, and cross-platform compatibility.

### 3.1. System Architecture Diagram

The diagram above illustrates the major components and data flow of the system, including the sequence of user interactions and backend processing steps.

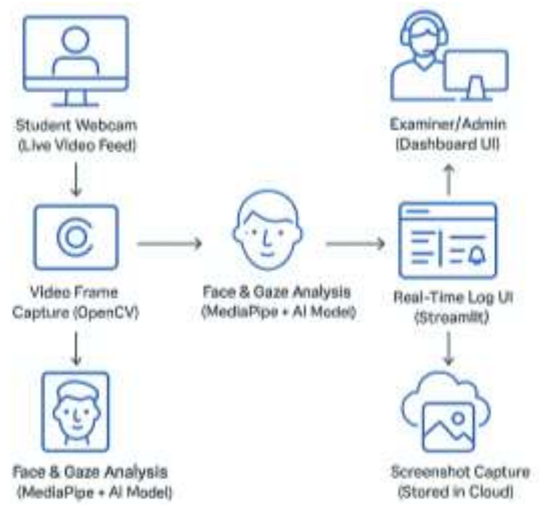

Student Webcam (Live Video Feed)
Examiner/Admin (Dashboard UI)
Video Frame Capture (OpenCV)
Face & Gaze Analysis (MediaPipe + AI Model)
Real-Time Log UI (Streamlit)
Face & Gaze Analysis (MediaPipe + AI Model)
Screenshot Capture (Stored in Cloud)

Fig. 1: System Flowchart for AI-Powered Online Proctoring System

### 3.2. Module-wise Breakdown

The proposed system comprises four key modules working in synergy to ensure secure and intelligent online exam proctoring.

a. User Registration and Identity Verification Module:

Students register using their phone number and password. During initial setup, facial images are captured via webcam and stored as biometric templates. Each login session employs facial recognition to authenticate the user, thereby preventing impersonation and ensuring secure access.

b. Real-Time Behavior Monitoring Module:

Using OpenCV for video capture and MediaPipe for facial landmark detection, the system continuously tracks student behavior. Gaze estimation, face presence, multi-face detection, and head pose estimation are conducted to monitor attentiveness and detect possible cheating attempts.

c. Anomaly Detection and Alert Generation Module:

A rule-based engine identifies anomalies such as face absence, gaze deviation, or multiple faces based on defined thresholds. Events are buffered to avoid false positives, and validated alerts trigger screenshot capture and real-time notifications to the examiner dashboard.

d. Examiner Dashboard Interface:

Developed with Streamlit, the dashboard displays real-time alerts, behavioral logs, and captured screenshots. It supports role-based access, enabling examiners to review alerts, validate behavior, and ensure integrity throughout the examination session.

### 3.3. Scalability and Deployment Considerations

To support varying institutional loads, the system is built with modular services that can be independently scaled. Docker containers encapsulate each service, allowing for seamless deployment in local or cloud environments. Google Cloud Storage or AWS S3 is used for storing image data, and Firebase Cloud Firestore handles real-time event data. These services ensure low latency, high availability, and efficient resource utilization.

### 3.4. Scalability and Deployment Considerations

To support varying institutional loads, the system is built with modular services that can be independently scaled. Docker containers encapsulate each service, allowing for seamless deployment in local or cloud environments. Google Cloud Storage or AWS S3 is used for storing image data, and Firebase Cloud Firestore handles real-time event data. These services ensure low latency, high availability, and efficient resource utilization.

## 4. IMPLEMENTATION

The implementation of the intelligent proctoring system integrates multiple technologies and modules to ensure real-time monitoring, anomaly detection, and alert generation.

### 4.1. Gaze Estimation Algorithm

To detect potential distractions or malpractice during examinations, a gaze estimation algorithm is employed. For each video frame captured from the webcam, facial landmarks are extracted using MediaPipe. The system identifies the coordinates of the left eye, right eye, and nose tip. The relative orientation is used to compute the gaze angle. If this angle deviates beyond ±15 degrees for more than 3 seconds, an alert is generated, as outlined in the pseudocode below:

### 4.2. Alert Generation Logic

```
for frame in webcam_stream:
    landmarks = get_face_landmarks(frame)
    if landmarks:
        left_eye = landmarks[33]
        right_eye = landmarks[263]
        nose = landmarks[1]
        angle =
calculate_gaze_angle(left_eye, right_eye,
nose)
        if abs(angle) > 15:
            if time > 3s:
                trigger_alert("Gaze
Deviation")
```

```
if not face_detected:
    start_timer()
    if timer > 2s:
        log_event("No Face Detected",
timestamp, save_screenshot())
elif detect_multiple_faces():
    log_event("Multiple Faces",
timestamp, save_screenshot())
```

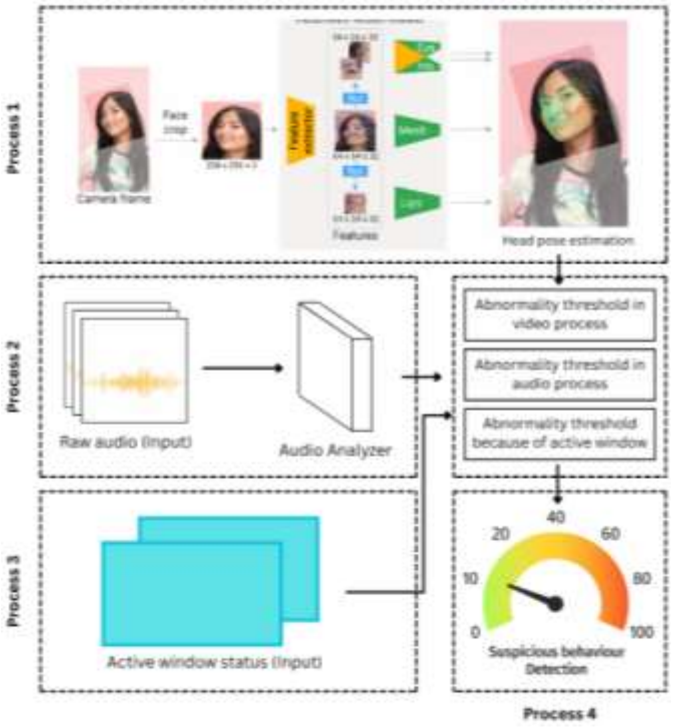
Alert generation is governed by a rule-based engine. For instance, if the examinee's face is not detected for over 2 seconds, or if multiple faces appear in the frame, the system logs an alert event and captures the associated screenshot, as follows:

### 4.3. Runtime Environment and Technology Stack

The system leverages a modern and modular tech stack to ensure efficient processing and scalability. Front-end interfaces are built using React.js and Streamlit for examiner dashboards. Real-time facial tracking is implemented using MediaPipe and OpenCV. Core logic and alert mechanisms run on Python, supported by TensorFlow for AI-based modeling. Data is stored securely on Firebase and AWS S3. The entire system is containerized using Docker and deployed on Google Cloud Platform for reliability and scalability.

## 5. RESULTS AND PERFORMANCE EVALUATION

The proposed AI-powered online proctoring system was evaluated for its ability to detect and flag suspicious behavior through multimodal analysis of video, audio, and screen activity. The complete system workflow is illustrated in Fig. 2, highlighting four key processes: (1) facial analysis and head pose estimation, (2) audio pattern analysis, (3) active window


Process 1
Face crop
Camera frame
Feature extractor
Features
Head pose estimation
Process 2
Raw audio (input)
Audio Analyzer
Abnormality threshold in video process
Abnormality threshold in audio process
Abnormality threshold because of active window
Process 3
Active window status (input)
0
10
20
40
60
80
100
Suspicious behaviour Detection
Process 4

monitoring, and (4) final suspicious behavior detection.

Fig. 2: System Process Diagram

This diagram demonstrates how input from the camera, microphone, and browser environment is processed using AI-based detection modules. Each module contributes to a unified anomaly detection framework, which calculates the suspiciousness level and triggers alerts based on predefined thresholds.

### 5.1. Test Setup and Evaluation Criteria

To validate system performance, a series of 30 mock examination sessions were conducted in a controlled environment. Each session was executed on a standard consumer-grade laptop equipped with a 720p webcam, running in the Google Chrome browser. During these tests, subjects simulated real-world behaviors, such as:

- Maintaining normal posture and eye contact,
- Looking away from the screen,
- Speaking or generating background noise,
- Leaving the frame,
- Switching to unauthorized windows.

Evaluation metrics included:

- Detection Accuracy: Correct identification of behavioral events,
- System Latency: Time delay in processing and responding to inputs,
- False Positive Rate: Incidences where normal behavior was incorrectly flagged.

### 5.2. Performance Metrics

Table 1 presents the results obtained from the performance evaluation:

Table 1: Detection Accuracy and System Performance Metrics

| Test Case | Accuracy (%) | Latency (ms) | False Positives |
|---|---|---|---|
| Face Detection | 98.7 | 110 | 1/30 |
| Gaze Estimation | 95.2 | 135 | 3/30 |
| Multiple Face Detection | 99.4 | 105 | 0/30 |
| No Face Alert Trigger | 97.6 | 120 | 2/30 |

The results demonstrate that the system performs with high accuracy and low latency across all critical functionalities, maintaining reliability in real-time proctoring scenarios.

### 5.3. Behavioral Detection and Logging

The system logs and visualizes anomalies in real time. Upon detecting behaviour outside accepted thresholds (e.g., gaze deviation, face absence, speech input, or unauthorized screen activity), the system captures:

- A screenshot of the incident,
- A timestamp,
- The type and source of the abnormality,
- A cumulative suspicious behaviour score.

These logs support post-exam review and help examiners make informed decisions.

### 5.4. Observations and Practical Considerations.

- Video Analysis: The head pose and facial landmark models performed consistently across users and environments.
- Audio Analysis: Effective at detecting spoken words or irregular sounds, though ambient noise remains a challenge.
- Window Tracking: Provides robust detection of tab-switching or off-screen activities but assumes full-screen test enforcement for best results.
- System Responsiveness: Average detection latency remained below 140 milliseconds, meeting real-time monitoring standards.

The AI-powered proctoring system demonstrates strong potential for enhancing academic integrity in remote examinations. It combines multimodal input processing with intelligent behavior scoring to ensure fair assessments. The framework achieves a balance between automation, reliability, and user transparency, making it suitable for large-scale deployment in academic institutions.

## 6. CONCLUSIONS

This paper presented the design, implementation, and evaluation of an AI-based online examination proctoring system that leverages real-time facial behavior monitoring to uphold academic integrity in remote assessments. By integrating technologies such as OpenCV, MediaPipe's Face Mesh, and rule-based anomaly detection, the system effectively identifies suspicious activities including gaze deviation, face absence, and multiple face presence. The modular architecture facilitates identity verification, continuous monitoring, and real-time alerting, all within a user-friendly dashboard designed for examiners.

The system achieved high accuracy in face detection and behavior tracking, with minimal latency and low false positive rates, validating its practical feasibility. The alert logging, screenshot capture, and examiner interface collectively contributed to enhanced transparency and examiner control during evaluations. The results demonstrate the potential of this system to offer a scalable and efficient alternative to manual proctoring, particularly in the context of mass digital education.

## 7. FUTURE SCOPE

While the current implementation addresses key proctoring challenges, several enhancements are envisioned for future work. One promising extension is the integration of audio-based monitoring, such as detecting human voice activity or background noise to flag potential collaboration. Additionally, replacing the rule-based gaze tracking with a machine learning classifier trained on eye movement patterns could improve the system's ability to distinguish between genuine attention shifts and malicious behavior.

Further, the deployment of the system on edge devices or mobile platforms can increase accessibility in resource-constrained environments. Implementing adaptive alert thresholds based on user behavior and exam context may also reduce false positives and personalize the proctoring experience. Moreover, research can be extended to explore emotion recognition and posture analysis to understand the test-taker's stress or engagement levels.

Scalability for institutional-level integration and compliance with data protection laws such as GDPR should be considered for wide-scale deployment. Lastly, incorporating Natural Language Processing (NLP) techniques for automatic feedback generation and candidate support can further enhance usability.

## REFERENCES

1. Google Developers: MediaPipe Face Mesh. https://mediapipe.readthedocs.io/en/latest/solutions/face_mesh.html
2. Google Developers: Face Landmarker — MediaPipe Solutions. https://developers.google.com/mediapipe/solutions/vision/face_landmarker/
3. Deshmukh, V., Kunjir, A.B.: Online Proctoring System Using Artificial Intelligence. Int. J. Novel Res. Dev. (IJNRD) 9(4) (2024) 1–8. https://www.ijnrd.org/papers/IJNRD2404290.pdf
4. Bradski, G.: The OpenCV Library. Dr. Dobb's J. Software Tools (2000)
5. Redmon, J., Farhadi, A.: YOLOv3: An Incremental Improvement. arXiv:1804.02767 [cs.CV] (2018). https://arxiv.org/abs/1804.02767
6. King, D.E.: Dlib-ml: A Machine Learning Toolkit. J. Mach. Learn. Res. 10 (2009) 1755–1758
7. Shih, Y.-S., Zhao, Z., Niu, C., Iberg, B., Sharpnack, J., Baig, M.B.: AI-assisted Gaze Detection for Proctoring Online Exams. arXiv:2409.16923 [cs.AI] (2024). https://doi.org/10.48550/arXiv.2409.16923
8. Yang, X., Wu, D., Yi, X., Lee, J.H.M., Lee, T.: iExam: A Novel Online Exam Monitoring and Analysis System Based on Face Detection and Recognition. arXiv (2022). https://arxiv.org/abs/2206.13356
9. Aggarwal, A., Kumar, A.: AI Based Real-Time Online Proctoring System using Deep Learning. Int. J. Eng. Res. Technol. (IJERT) 10(5) (2021) 134–139. https://www.ijert.org/ai-based-real-time-online-proctoring-system-using-deep-learning