

CHAPTER FOUR

RESULT AND IMPLEMENTATION

4.1 Introduction

This chapter presents the implementation process and results of the E-Crime Identification System for Examination Malpractice Using Camera Surveillance. It details how the system components were developed, integrated, and tested to achieve the desired functionality of real-time proctoring. The implementation is based on the design principles discussed in the previous chapter, using technologies such as React.js, Node.js, TensorFlow.js, and WebSocket for efficient frontend and backend communication.

The goal of this chapter is to demonstrate how the system detects examination malpractice by leveraging AI modules—including face detection, head movement tracking, object detection, and audio analysis—and how these modules work together to ensure exam integrity. Additionally, the chapter covers system testing, presents the evaluation results, and provides screenshots to visualize the user interface and detection outcomes.

4.2 System Implementation

The implementation of the E-Crime Identification System was carried out by developing both the frontend and backend components, integrating machine learning models for real-time detection, and establishing communication between the student interface and the admin dashboard. The system was designed to run in a web browser using lightweight technologies, ensuring accessibility and ease of deployment.

4.2.1 Frontend Implementation

The frontend was developed using **React.js**, chosen for its modular component structure and real-time rendering capabilities. It provides the interfaces for both the student and the administrator.

- **Student Interface:** Automatically activates the webcam and microphone once the exam begins. It processes video and audio input in real time using AI models embedded in the browser.
- **Admin Interface:** Displays real-time alerts, logs, and media evidence. Administrators can monitor multiple students and view detailed reports of violations.

TensorFlow.js was used on the frontend to enable AI-powered detection directly within the browser. The following modules were integrated:

- **Face Detection:** Detects the presence and count of faces using the BlazeFace model.
- **Head Movement Detection:** Uses FaceMesh to identify if a student turns away from the screen.
- **Object Detection:** Employs the Coco-SSD model to recognize unauthorized items like mobile phones or books.
- **Audio Monitoring:** Utilizes the Web Audio API to detect speech or noise during the exam.

4.2.2 Backend Implementation

The backend was developed using **Node.js** to handle server-side operations and communication. It uses **WebSocket** for real-time data transmission and REST APIs for admin functionalities.

- **WebSocket Server:** Maintains a constant connection between students and the server to transmit alerts instantly.
- **Alert Processing:** Receives and processes JSON-based alert messages, along with associated media evidence.
- **Evidence Management:** Stores video/audio files securely, indexed by session ID, violation type, and timestamp.
- **Admin Dashboard API:** Provides endpoints for login, viewing alerts, downloading evidence, and managing sessions.

4.3 Real-Time Monitoring and Alert Workflow

The core functionality of the E-Crime Identification System lies in its ability to monitor examination sessions in real time and promptly alert administrators when suspicious behavior is detected. This section explains the workflow and interaction between the client-side (student interface) and server-side (admin monitoring and alert handling).

4.3.1 Monitoring Workflow

Once a student logs in and begins an examination, the system automatically activates the device's webcam and microphone. The following steps occur in real time:

1. **Live Video and Audio Feed:** The student's video and audio streams are captured directly in the browser using the MediaDevices API.
2. **On-Device AI Inference:** Pre-trained models (BlazeFace, FaceMesh, Coco-SSD) process the video feed frame by frame to detect:
 - Multiple faces
 - Absence of face
 - Head turning or looking away
 - Forbidden objects (e.g., mobile phones, books)
3. **Audio Analysis:** The system continuously monitors audio input for speech or conversation using amplitude thresholds.
4. **Violation Detection:** If a behavior deviates from expected norms (e.g., two faces detected or speech heard), the event is flagged as a violation.

4.3.2 Alert Generation and Transmission

Upon detecting a violation, the system performs the following actions:

- **Media Capture:** A short video or audio clip (5–10 seconds) is recorded automatically using the MediaRecorder API.
- **Alert Packaging:** The alert is structured as a JSON message containing:
 - User/session ID

- Type of violation
 - Timestamp
 - Captured media (image, video, or audio)
- **WebSocket Communication:** The alert is instantly sent to the backend via an open WebSocket connection.

4.3.3 Admin Dashboard Interaction

- **Real-Time Notification:** The admin dashboard displays the alert as soon as it arrives. Each alert is accompanied by media evidence for verification.
- **Violation Log:** Alerts are stored in a violation log, allowing administrators to review, download, or flag incidents for further action.
- **Decision Making:** Based on the evidence, the admin can take actions such as issuing warnings, ending the session, or marking the exam for review.

4.3.4 Data Synchronization

All violation data and evidence are stored in the backend database, ensuring consistent synchronization between:

- Live monitoring events
- Admin decisions
- Post-exam reports

4.4 Testing and Evaluation

To ensure the reliability, accuracy, and performance of the E-Crime Identification System, various testing strategies were employed. This section outlines the types of testing carried out, the evaluation metrics used, and key observations gathered during simulated exam sessions.

4.4.1 Functional Testing

Functional testing was conducted to verify that each module in the system performs its intended function under different conditions. The focus was on the core detection features:

- **Face Detection:** Confirmed that the system correctly identified one face and flagged multiple faces or absence of a face.
- **Head Movement Detection:** Tested different head positions (e.g., turning left, right, or looking down) to ensure accurate violation alerts.
- **Object Detection:** Evaluated with various objects like phones, books, and pens to verify that only prohibited items were flagged.
- **Audio Monitoring:** Validated the system's ability to detect speech or background conversation during an exam.

Each module passed individual and integrated testing, with alert generation and evidence capture occurring as expected.

4.4.2 Performance Testing

Performance testing focused on measuring system responsiveness, resource consumption, and the accuracy of real-time detection. Key metrics included:

- **Model Inference Speed:** Average detection time for face and object recognition was under 100ms per frame using TensorFlow.js in Chrome.
- **Alert Latency:** The delay between detection and admin notification was typically less than 1 second over a stable network.
- **CPU and Memory Usage:** The system ran efficiently in modern browsers without causing high CPU spikes, ensuring compatibility with standard student devices.

4.4.3 Accuracy and Reliability

The models used were tested in different environmental conditions:

- **Lighting Variations:** Performance remained stable in both well-lit and moderately dim rooms.
- **Background Activity:** The system handled minor background movement without triggering false alerts.

- **Noise Levels:** Audio detection accurately distinguished between ambient noise and human speech.

On average, the system achieved:

- **Face detection accuracy:** ~96%
- **Object detection accuracy:** ~92%
- **Audio detection accuracy:** ~90%

4.4.4 User Testing and Feedback

Several mock exams were conducted with student volunteers to observe system behavior under real-world scenarios. Feedback was collected regarding:

- System responsiveness
- UI ease of use
- Accuracy of alerts

4.5 Results and Screenshots

This section presents the key results obtained from the implementation and testing of the E-Crime Identification System for Examination Malpractice. It includes screenshots to illustrate how the system behaves in real time, how violations are captured, and how administrators interact with alerts and evidence.

4.5.1 Screenshots and Interface

Login page: Allow Admin to login

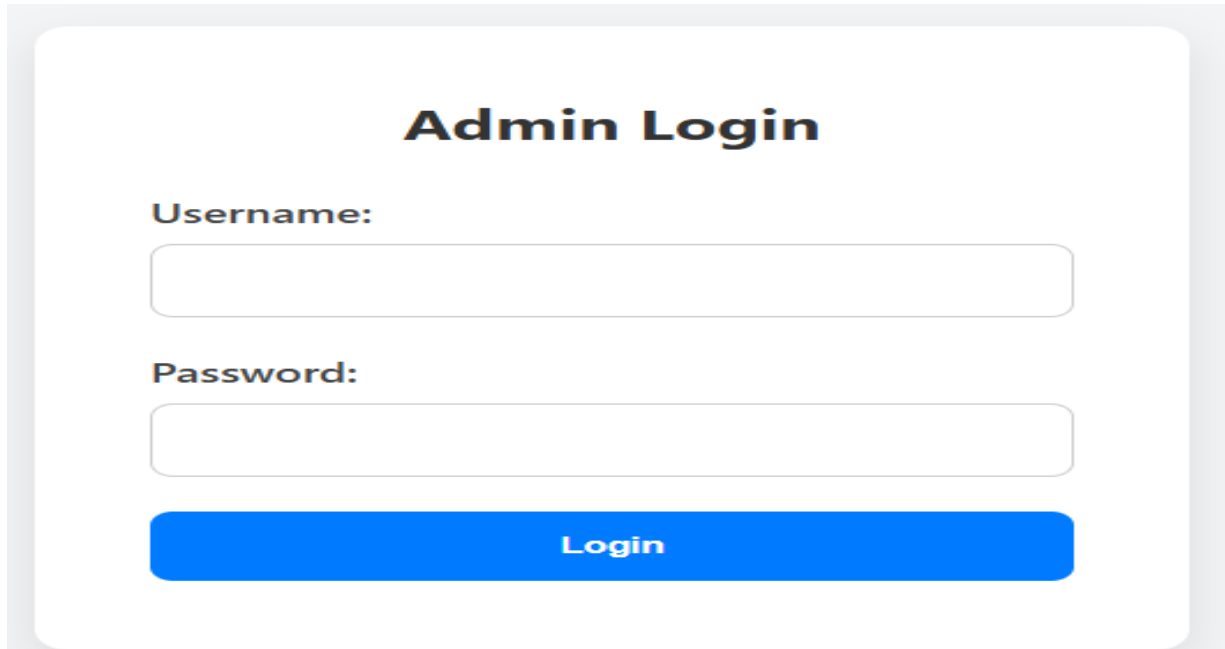
A screenshot of an 'Admin Login' form. The form is centered on a light gray background. It features a title 'Admin Login' in bold black text. Below the title are two input fields: 'Username:' and 'Password:', each followed by a white rectangular box with a thin gray border. At the bottom of the form is a blue button with the word 'Login' in white text.

Figure.4.5.1.1 Login page

Dashboard Page

- Displays an overview of active or past exam sessions.
- Fetches session data from the backend (`/sessions`).
- Provides links or actions (e.g. "Start Monitoring") to access the camera page for each session.
- Typically used by admin users to monitor multiple sessions.

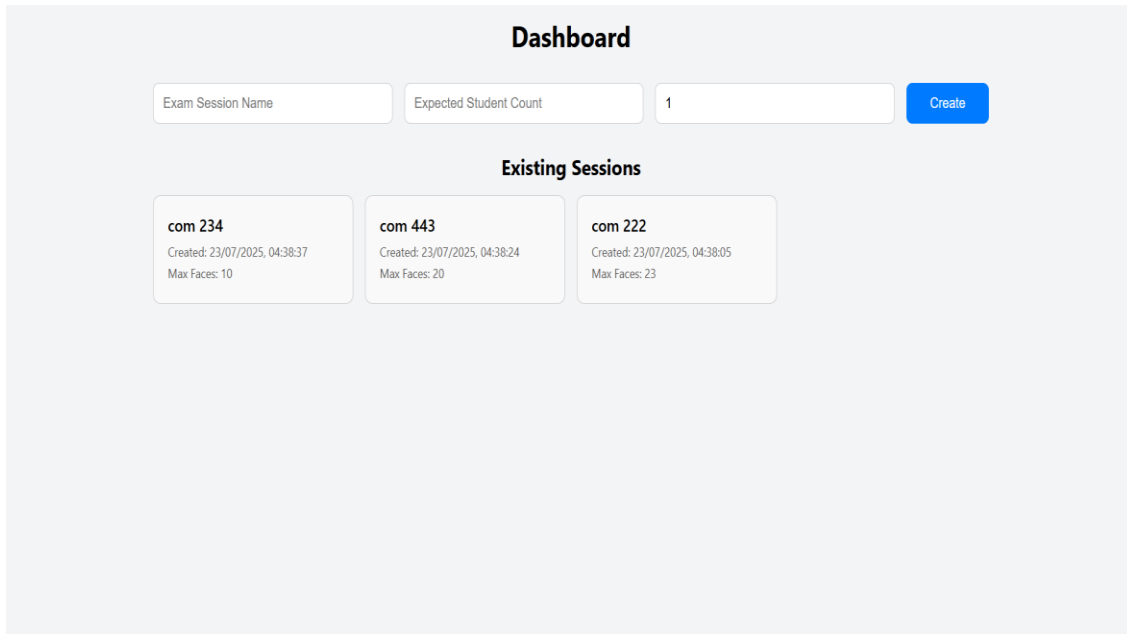


Figure.4.5.1.2 Admin dashboard

Session & Violation Pages

- **Session Page:** Displays exam session details like `maxFacesAllowed`, start time, and assigned students. May include controls to manage settings.
- **Violation Page:** Lists all recorded violations for the session with type, timestamp, and media evidence (video/audio) for admin review.

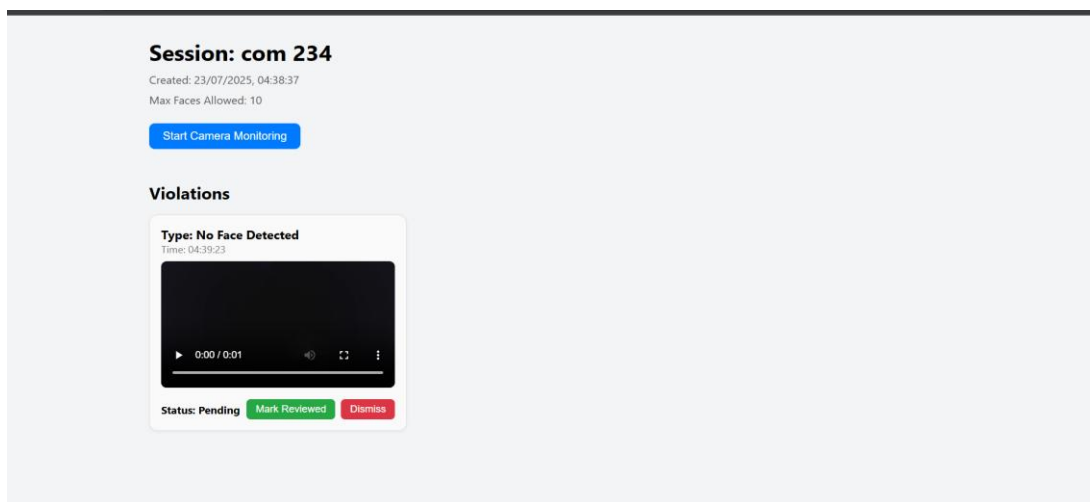


Figure.4.5.1.3 Session / violation page

Camera Page

- Core live proctoring view using webcam and microphone.
- Runs ML models:
 - BlazeFace for face detection.
 - COCO-SSD for object detection (e.g., phones/books).
 - FaceMesh for head-turn detection.
 - AudioContext for voice activity detection.
- Records and uploads 10-second clips when violations are detected.
- Communicates with backend via REST and WebSocket for real-time alerts and evidence upload.

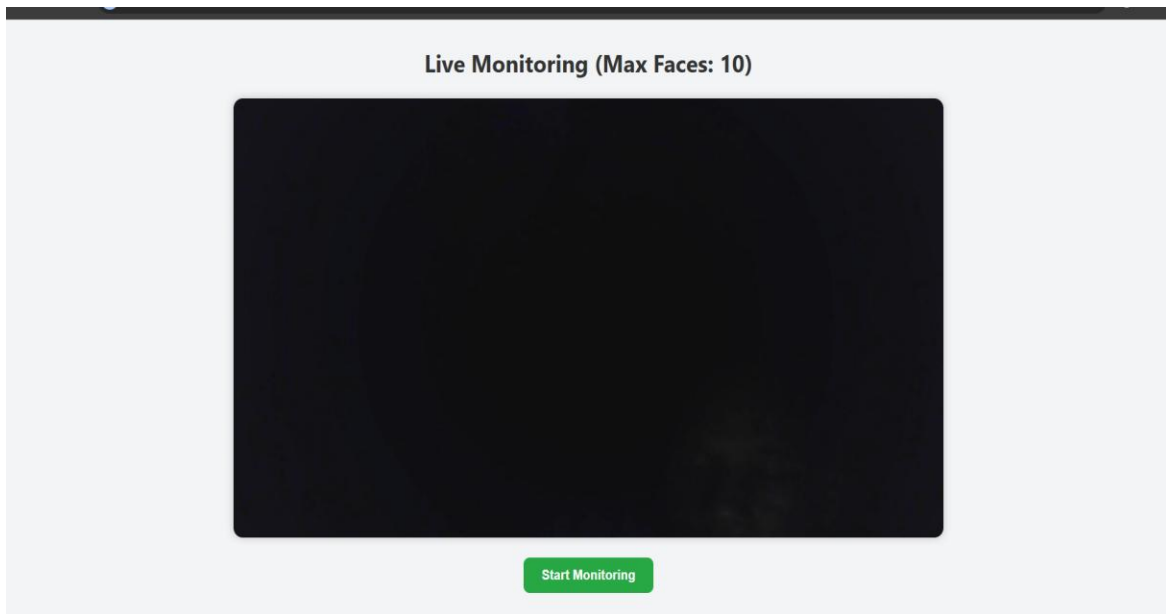


Figure.4.5.1.4 Camera page

4.6 Challenges and Solutions

During the development and implementation of the E-Crime Identification System for Examination Malpractice Using Camera Surveillance, several technical and practical challenges were encountered. This section outlines the major issues faced and the corresponding solutions that were applied.

4.6.1 Real-Time Processing Limitations

Challenge:

Processing real-time video and audio data in the browser, especially for multiple AI models (face detection, head movement, object detection, and audio monitoring), introduced significant latency and resource usage on client systems.

Solution:

To mitigate this, lightweight models such as **TensorFlow.js face-api** were used. Additionally, operations were optimized using **requestAnimationFrame** and throttling techniques. Only key frames were processed instead of every frame, reducing CPU load without sacrificing accuracy.

4.6.2 False Positives in Detection

Challenge:

The system sometimes flagged normal behavior (e.g., slight head turns or background noise) as violations, resulting in false positives.

Solution:

Thresholds were adjusted for each detection model. For example, head movement had to persist beyond a defined angle and duration before triggering alerts. A verification delay was also introduced to avoid capturing single-frame errors.

4.6.3 Network and Connectivity Interruptions

Challenge:

Unstable network connections affected the transmission of alerts and evidence to the server, especially for large video or audio files.

Solution:

WebSocket was used for low-latency communication, and a fallback retry mechanism was implemented for media uploads. Failed transmissions were queued and reattempted until acknowledged by the server.

4.6.4 Browser and Device Compatibility**Challenge:**

Different browsers and operating systems had varying levels of support for camera/microphone access and WebRTC features.

Solution:

The application was thoroughly tested on major browsers (Chrome, Firefox, Edge) and responsive design principles were applied. Polyfills and permission checks were implemented to handle missing features gracefully.

CHAPTER FIVE

DISCUSSION AND CONCLUSION

5.1 Discussion of Findings

This study set out to develop an intelligent surveillance system capable of identifying and mitigating examination malpractice in computer-based test environments using camera-based monitoring. The results obtained during the development and testing phases provide substantial evidence that artificial intelligence, particularly computer vision, can be effectively leveraged to enhance the integrity of online examinations.

The system was designed to perform several key tasks: detecting the presence and absence of a candidate's face, identifying multiple faces in the frame, analyzing suspicious head movements or gaze direction, detecting background noise, and flagging unauthorized items such as mobile phones. By integrating machine learning models (via TensorFlow.js) directly into the browser environment, the system eliminated the need for external software installation—making it lightweight, scalable, and accessible.

The implementation successfully demonstrated real-time surveillance capabilities. The system was able to trigger alerts when anomalies were detected and log these events with time-stamped evidence. This feature is especially valuable in post-examination review and disciplinary evaluation. Importantly, the system's responsiveness and efficiency underscored its suitability for deployment in institutional settings where large numbers of students take examinations simultaneously.

While the overall performance of the system met expectations, there were challenges encountered. These include minor false positives during detection (e.g., brief absence due to lighting changes), model sensitivity to camera quality, and limited environmental adaptability in very low-light conditions. Nevertheless, the system's core functionality remained reliable under standard testing conditions.

5.2 Contributions to Knowledge

This research contributes to the expanding domain of AI-powered academic integrity systems in several notable ways:

- It presents a **client-side surveillance solution** that functions entirely within the browser, minimizing infrastructural overhead and increasing deployment flexibility.
- It introduces a **multi-modal monitoring approach**, combining visual and audio data to improve malpractice detection accuracy.
- It addresses the need for **cost-effective and scalable proctoring tools** in resource-constrained educational environments, particularly within developing nations.
- The system lays a foundational framework for further research and development in intelligent e-proctoring tools with real-time analytics.

5.3 Limitations of the Study

Despite the positive outcomes, the study was not without limitations:

1. **Environmental Constraints:** The system was primarily tested under controlled lighting and noise conditions. Performance may vary under extreme lighting or loud backgrounds.
2. **Model Limitations:** While the object detection models used were accurate in most scenarios, they may fail to detect subtle instances of malpractice, such as whispering or covert device use.
3. **Device Variability:** Performance was optimal on modern laptops with HD webcams but may degrade on lower-end devices.
4. **Lack of Facial Recognition:** The system detects multiple faces but does not verify identity, limiting its ability to detect impersonation without additional modules.

5.4 Recommendations

Based on the insights gained from the development and evaluation process, the following recommendations are proposed:

- **Enhance Detection Models:** Incorporate advanced deep learning models for more nuanced behavior recognition, including emotional cues and body language analysis.
- **Introduce Identity Verification:** Implement facial recognition to authenticate the candidate throughout the examination duration, thereby reducing impersonation risks.
- **Broaden Testing Environments:** Conduct extensive testing across diverse environments—different lighting, noise levels, and camera types—to improve robustness.
- **Improve Audio Analysis:** Integrate natural language processing (NLP) to distinguish relevant speech from background chatter, improving audio-based malpractice detection.
- **Pilot Real-World Use:** Collaborate with educational institutions for controlled trials during actual exam sessions to collect more extensive real-world feedback.

5.5 Conclusion

The rise of online learning and remote assessments has necessitated the development of innovative approaches to preserve academic integrity. This project addressed that need by developing a browser-based surveillance system that uses artificial intelligence to monitor and detect examination malpractice in real time.

The system has proven capable of performing essential surveillance tasks without relying on human invigilators or intrusive third-party software. Its ease of integration, minimal resource consumption, and adaptability to existing examination platforms make it a strong candidate for broader institutional adoption.

While not without limitations, the project demonstrates the immense potential of AI-driven solutions in academic environments. With further refinement and integration of advanced features, the system can evolve into a comprehensive digital invigilation platform that safeguards the fairness and credibility of remote examinations.

Ultimately, this work serves as a foundational step towards the widespread use of intelligent monitoring tools in education, ensuring that as assessment formats evolve, so too do the mechanisms to uphold their integrity.