A Web-Based AI System for Real-Time Suspicious Behaviour Detection in Online Assessments

Pravik Bunsaeng Som Bun[1], Nur Ziadah Harun[1]*, Nur Shahirah Azahari[2], A. Z. Rahman[3], Nargis Fatima[4]

[1]Faculty of Computer Science and Information Security, Universiti Tun Hussein Onn Malaysia (UTHM), Parit Raja, Johor, 86400, Malaysia

[2]Faculty of AI & Robotics, Raffles University, Medini Iskandar Malaysia Iskandar Puteri, 79250 Johor Bahru, Johor, Malaysia

[3]IoT Systems Lab, MIMOS Berhad, MRANTI Technology Park 57000 Kuala Lumpur, Malaysia

[4]Department of Software Engineering, National University of Modern Languages, Islamabad, Pakistan

***Corresponding Author**

DOI: https://doi.org/10.47772/IJRISS.2026.101000098

Received: 12 October 2026; Accepted: 18 October 2026; Published: 30 October 2026

ABSTRACT

Online assessments present challenges in maintaining academic integrity, particularly when direct supervision is limited. This study presents a web-based system designed to support real-time detection of potentially suspicious behaviour during online assessments. The system integrates TensorFlow-based object detection, MediaPipe-based facial analysis, WebRTC webcam streaming and browser-event monitoring to identify events such as face absence, multiple faces, mobile phone presence, abnormal head movement, tab switching and copy-paste activity. The system also provides assessment management, monitoring alerts and records to support lecturer review. Functional testing was conducted with 16 participants, with testing involving the Academic Administration. The testing examined the system's operational functions and monitoring workflow. The system is therefore presented as a decision-support tool for online assessment supervision, with AI-generated alerts requiring human verification rather than being treated as conclusive evidence of academic misconduct.

Keywords: Online Assessment System, Integrity, Suspicious Behaviour Detection, CNN Algorithm, Real-Time AI Monitoring

INTRODUCTION

Higher education assessment procedures have been strongly impacted by the growth of digital learning technology. Online assessments have gained popularity because of their flexibility, cost-effectiveness, and capacity to involve students from different places. This method gained particular importance in instances when face-to-face assessments were not possible, for example, during the pandemic of COVID-19. Online assessment is a useful tool that supports modern learning environments and allows ongoing assessment for universities. Therefore, secure online examinations are essential to assure academic integrity through similar assessment taking conditions in similar courses when using online assessments [1].

In e-learning platforms, students are required to complete the assessments within the given time frame, usually without direct monitoring [2]. In some cases, lecturers may ask students to attempt the assessment during class sessions, but it remains difficult to observe every student simultaneously. Since there is limited monitoring, there is a possibility that students may engage in academic misconduct by using AI tools or external devices to search for answers during the assessment. Existing system works well for distributing assessments, but it does not have any features for identity verification or real-time monitoring [3].

Additionally, there are a few issues with fairness and verification with the current approach. Large classes make manual observation impractical and password-only access is easily shared, raising the possibility of unwanted participation or impersonation. According to [4], biometric and intelligent monitoring technologies play a vital role in confirming user identity and maintaining integrity in digital systems. Without such mechanisms, it becomes difficult to ensure that each student completes the assessment independently and honestly.

This project integrates Artificial Intelligence (AI) technology employing Convolutional Neural Network (CNN) programming algorithm in the automatic detection and classification of suspicious actions without having to rely solely on human observations. The AI will monitor the data from the webcam and detect irregularities like a face being absent for too long, more than one person detected on camera, excessive movements of the head and eyes. According to [5], the models based on deep learning can be successfully used for identifying malpractices in online exams making it possible for the automated online proctoring system to be developed.

LITERATURE REVIEW

This section provides the literature review regarding the online assessment monitoring system, Convolutional Neural Networks (CNN) and the current comparative systems. The purpose of this review is to understand and discuss the context of the project related to online assessment monitoring system based on AI.

Human Monitoring

The first and most widely-used technique of conducting monitoring during online tests is through the employment of human supervision in which instructors assess students in real-time through video conferencing mechanisms like Google Meet, Microsoft Teams, or Zoom. This way, human supervision is somewhat similar to traditional testing, where human monitors ensure that the identity of a student is verified, their behavior is observed, and they can respond to dishonest actions immediately. The authors in [6] say human supervision can help monitor students' private information such as images, answers, and screens. But, at the same time, human supervision makes some significant limitations when applied to a large-scale examination. For example, this method does not allow for monitoring many students simultaneously and might lead to numerous issues such as tiredness, distractions, or even unconscious bias. Moreover, for applying this approach it is necessary to have a high-speed Internet connection making this approach impossible for various students due to technical limitations. While human supervision is a familiar and trusted approach, due to its limitations the application of this method becomes rather complicated in the case of either large or repeated examinations.

Recorded Monitoring

Recorded monitoring also called record-and-review monitoring, captures students' audio, screen activity and webcam feeds during assessments. After the assessment, these recordings are temporarily stored and reviewed by lecturers or other authorized staff [8]. Unlike live monitoring, this method lets instructors examine suspicious behaviour at their own pace, replaying footage and zooming in to detect potential misconduct more accurately, thereby providing stronger evidence for academic integrity. However, this approach has several challenges, as reviewing hours of recordings is time-consuming, especially for lecturers with multiple classes and storing videos securely is essential to protect students' privacy and comply with data regulations. Mishandling this data can lead to breaches and ethical issues. While recorded monitoring improves fairness and reviewability, it is still less efficient and more resource-intensive than modern AI-based monitoring systems.

Deep Learning-Based Online Assessment Monitoring

The introduction of deep learning-based online assessment monitoring systems has provided a new way of enhancing the accuracy of online assessments. Compared to the conventional methods of online assessment monitoring, deep learning-based monitoring has been able to provide instantaneous feedback on students' actions in the course of conducting online assessment. Deep learning-based monitoring is based on machine

learning algorithms that have the ability to observe students' online behaviour and notify about any possible misconduct. According to [9], intelligent monitoring improves the effectiveness of the online assessment process by means of continuous monitoring during the assessment process and creation of detailed analytics about the students actions in the course of the assessment. This capability is of particular significance for higher education, where a large number of students makes the use of human monitoring much less practical.

The main goal of monitoring systems based on deep learning is to detect student behaviours such as the appearance of different faces, excessive screen time or abnormal body movements that may imply inappropriate behaviour in the course of academic evaluation. Rather than solely relying on observers, lecturers can make use of the activity logs produced by the system to get alerts that can identify problems occurring in the assessment process. Deep learning systems rely on students' web cameras and computer vision technologies to track students' faces and behavioural patterns. One of the major techniques for achieving this involves the usage of Convolutional Neural Networks (CNN) which process visual data in layers and help identify such facial behaviours and actions as the position of the face in relation to the screen and whether or not students are in view of the camera.

Detecting unusual behaviour is a key feature of deep learning-based monitoring. CNNs based model analyze live webcam feeds to identify actions that are not normal, such as not showing a face, having multiple people in one frame, frequently looking away or using unauthorized devices like laptops or tablets. In addition, study in [11] state that this technology allows instructors to monitor many students at once while maintaining assessment security. The system can send real-time alerts or store suspicious activity for later review by lecturers. This reduces instructor’s workload and ensures equal monitoring for all students. According to [12], AI monitoring reduces human bias and errors, creating a fairer and more objective assessment environment, especially in large classes.

Comparison with The Existing System

Table 1 shows the comparison between the existing systems including Honorlock, ProctorU, Verificient-ProctorTrack and the proposed system. Table 1 compares the characteristics of the system.

Table 1: System’s Comparison

Feature	**Honorlock [13]**	**ProctorU [14]**	**Proctortrack [15]**	**Proposed System**
Monitoring Type	Automated	– AI + Human	– Automated + Multimodal	Automated AI
Identity Verification	Facial Scan	– Human Check + ID Verification	– Continuous Biometric	None (non-intrusive and designed to reduce privacy risks)
Behavior Detection	Yes	– Yes	– Very Advanced	Yes
Browser Lockdown	Yes	– No	– No	No (less restrictive, more student-friendly)
Cost	High	– Very High	– High	No licensing cost
Internet Requirement	High	– Very High	– High	Moderate – designed for Malaysian network conditions

Privacy Risks	Medium	– High	– High	Very low (No Biometric Storage)
Strength	Good AI automation, browser lockdown	– Human-Supervised Monitoring	– Continuous Identity Verification and Multimodal Detection	Cost-free, privacy-conscious, AI-assisted, open-source, and locally controlled

Honorlock is an Online proctoring service that uses artificial intelligence combined with web cameras to track and restrict the assessment-taker's activities during remote testing. One of its innovative features named "Search and Destroy" checks the web for any secrets related to a test. It uses a facial recognition system which allows to tell the student being verified. The problem is that the verification process only starts at the very beginning of an assessment and there is no further monitoring that follows the first scan. In reality, this means that impersonation can happen any time even when the initial identity verification has been already done. ProctorU is another online proctoring service which uses a mix of AI technology and human help. Students must verify their identity and complete an environmental scan before the exam and human proctors can respond to suspicious behaviour in real time, which reduces AI errors and improves exam security. This live supervision increases student compliance and discourages cheating making ProctorU suitable for high-stakes exams. However, the system is costly and time-consuming as institutions must pay per student and session making it less practical for frequent assessments. ProctorU also raises privacy concerns due to live webcam monitoring and requires strong internet access which can disadvantage low-income or rural students. ProctorTrack by Verificient is a highly advanced AI system designed to provide reliable assurance of student identity verification while taking exams to prevent impersonating behaviors. The system utilizes several monitoring techniques such as facial recognition, eye movement tracking, surrounding area checking, and fingernail pattern recognition to determine and identify cases of academic dishonesty. In addition, ProctorTrack has an innovative data analytics platform which allows an instructor to analyze student performance behaviors during tests. However, users have pointed out ethical concerns with the system due to the fact it collects sensitive personal information related to biometric data usage. The matter becomes serious as in 2020 the system was hacked revealing the private information of many students.

System Architecture

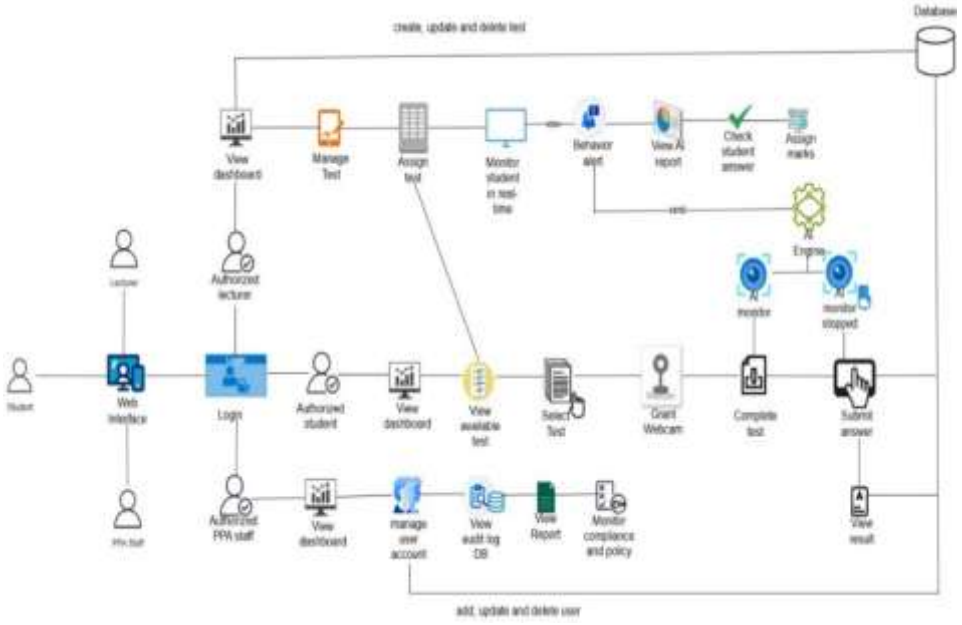

Figure 1: System Architecture

In Figure 1, the online testing centralized web system displays three types of users: students, lecturers, and ADMIN staff. Each user has his or her own login, consisting of a username and password, depending on his or

her authority. Students are only permitted to view and attempt the assessments, while their behaviour is also being monitored by the system through the use of AI. Once the assessment is over, students' answers are saved, and the result can be viewed on their dashboard. Teachers are responsible for inputting the assessments in the system, monitoring students during the assessments, and reading alerts created by the AI to ensure the assessment process is proceeding as expected. Admin staff manage the online testing system by looking after accounts, audit logs, making reports, and ensuring the operation policies are followed. All data and logs are stored securely in a centralized database for integrity and traceability.

System Implementation

The system was developed using PHP, MySQL, JavaScript, Node.js, WebRTC, TensorFlow and MediaPipe technologies to support online assessment management and real-time monitoring as listed in Table 2. The implementation includes user management, assessment management, real-time monitoring, report generation and security features such as password hashing, password reset, account lockout, input validation and secure session management. This section also discusses the operation of the system prototype and the implementation of the deep learning-based monitoring module used to detect suspicious student behaviour during online assessments.

Table 2:Technologies and Tools Used

Software	**Description**
Visual Studio Code	A code editor used for writing PHP, HTML, CSS, JavaScript, and Python scripts for the AI module.
TensorFlow [16]	Used to support deep learning-based behavioural detection and monitoring functionalities.
MediaPipe [17]	Used for real-time face detection and webcam-based monitoring features.
WebRTC [18]	Used to support real-time webcam streaming and live communication between the student monitoring interface and lecturer monitoring dashboard.
Node.js [19]	Supports real-time monitoring, alert generation, socket communication, and backend real-time functionalities.
XAMPP / Apache	Functions as the local server to host and run the web-based monitoring system.
MySQL / phpMyAdmin	Database system for storing user accounts, monitoring logs, test data and AI detection records.
Google Chrome / Firefox	Web browsers used for accessing, testing, and validating system features during development.

Implementation of Security Modules

The system implements several security features including password hashing, password reset through email verification, account lockout, input validation, secure session management, and prepared SQL statements to protect user accounts and test data from unauthorized access and security threats. Figures 2 and 3 show the implementation of input sanitization, where user inputs are validated and cleaned before authentication to improve system security and data integrity.

```
BEGIN
  RECEIVE username and password from login form
  CALL sanitizeInputString(username, 64)
  FUNCTION sanitizeInputString(value, maxLength)
    CONVERT value into string
    REMOVE unnecessary white spaces from value
    IF length of value is greater than maxLength THEN
      CUT value until maxLength
    ENDIF
    RETURN cleaned value
  ENDFUNCTION
  CONVERT password into string
```

Figure 2: Pseudocode of Input Sanitization During User Login

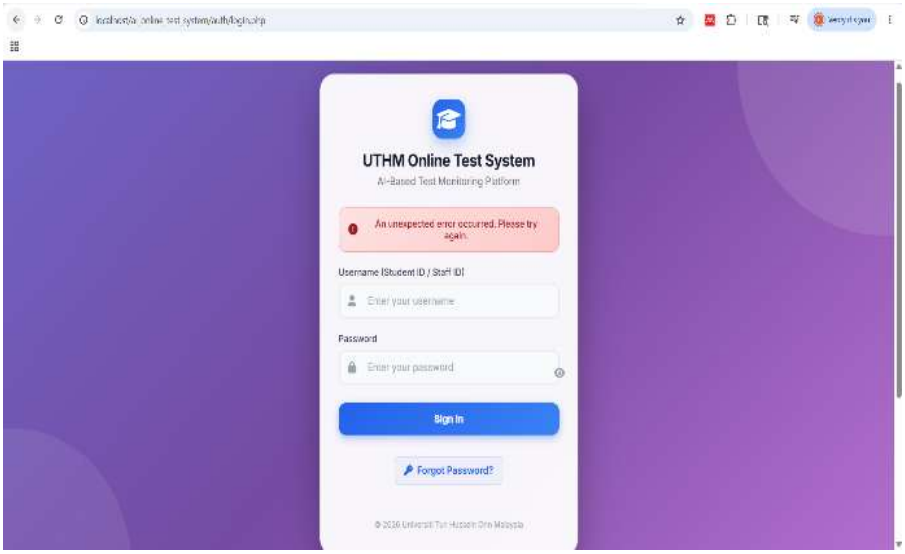

Figure 3: Interface of Input Sanitization during Login

```
BEGIN
  GET client IP address
  CREATE IP bucket using client IP address
  SET request limit = 20
  SET time window = 300 seconds
  CALL rateLimitAllow(IP bucket, request limit, time window)
  IF request is allowed THEN
    CONTINUE login process
  ELSE
    SET HTTP response code to 429 (Too Many Requests)
    DISPLAY message "Too many requests. Please try again later."
    STOP process
  ENDIF
END
```

Figure 4: Pseudocode of Rate Limiting for Login

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume X Issue X October 2026

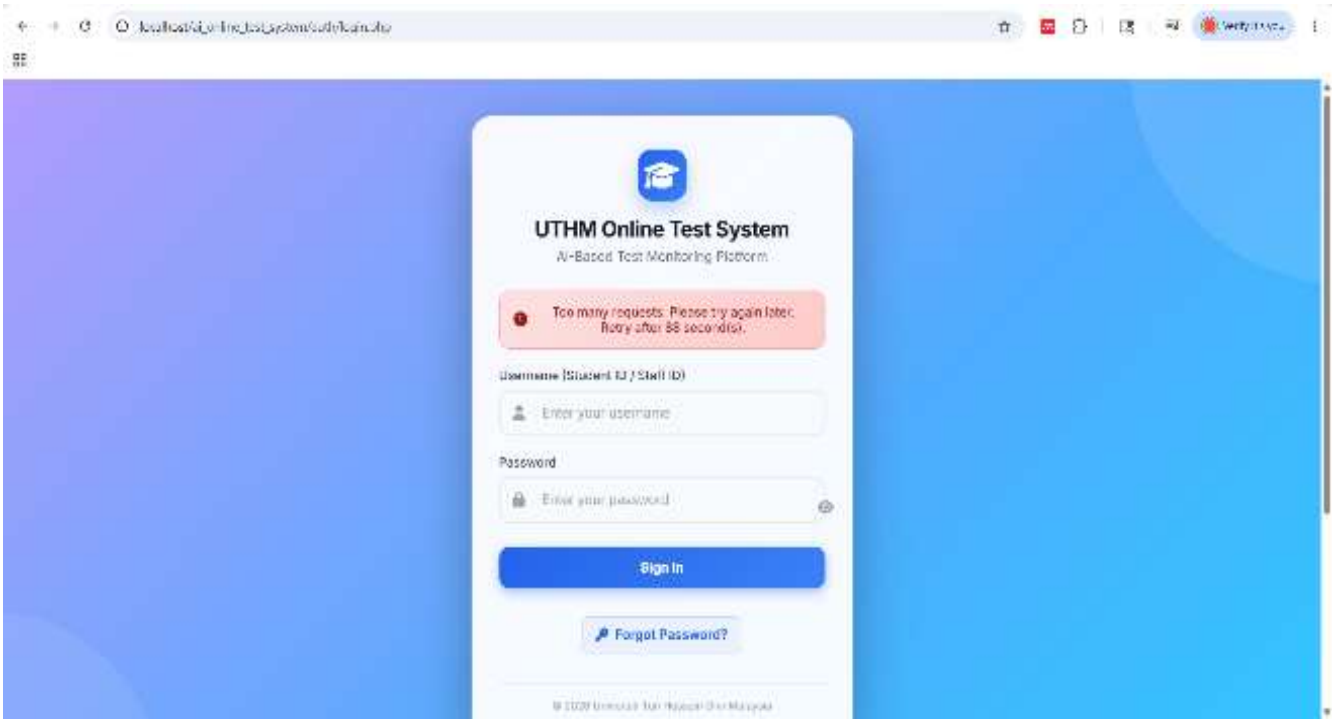

Figure 5: Interface of Rate Limiting

Figures 4 and 5 show the implementation of rate limiting, which restricts excessive login requests and temporarily blocks users when the request limit is exceeded.

```
START
IF login is unsuccessful THEN
   Increase failed login attempts by 1
   SET lockUntilValue = NULL
   IF failed login attempts >= maximum allowed attempts THEN
      SET failed login attempts = maximum allowed attempts
      SET account lock time = current time + lockout duration
   END IF
   UPDATE user account in database:
      - Save failed login attempts
      - Save lockout time
   IF failed login attempts >= maximum allowed attempts THEN
      RECORD audit log:
         - Username
         - User role
         - Security category
         - Action = "Account temporarily locked"
         - Reason = "Multiple failed login attempts"
         - User IP address
         - Warning status
      DISPLAY message:
         "Too many failed attempts. Account temporarily locked."
   END IF
```

Figure 6: Pseudocode of Account Lockout Mechanism After Failed Login Attempts

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume X Issue X October 2026

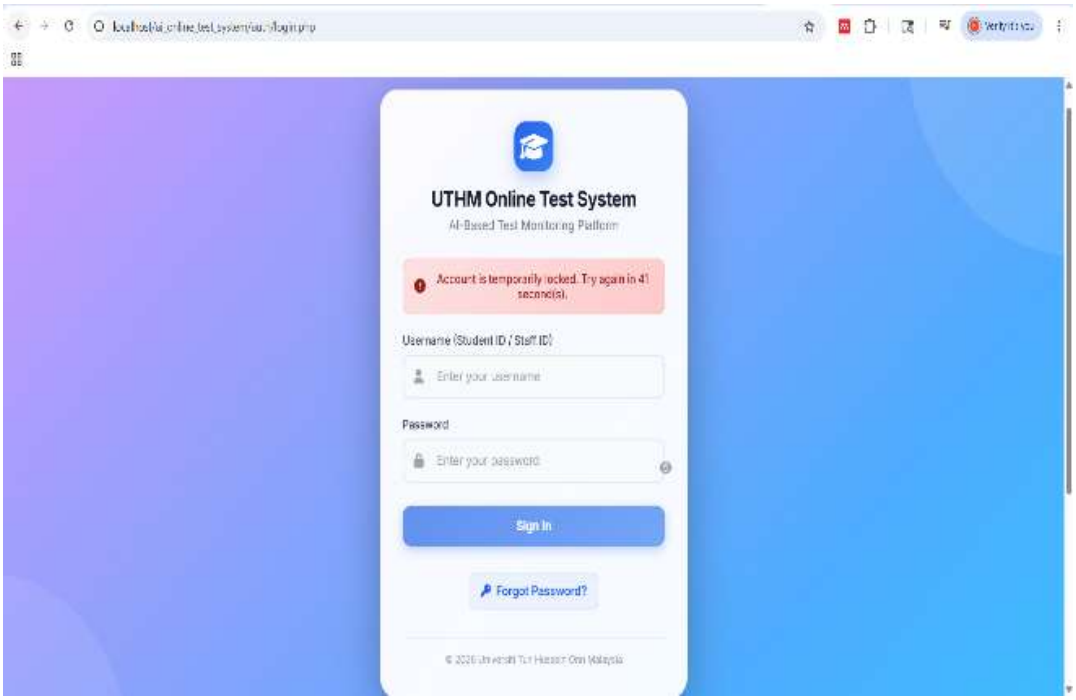

Figure 7: Account Lockout Interface

Figures 6 and 7 show the implementation of the account lockout mechanism, which temporarily locks user accounts after multiple failed login attempts to improve system security.

```
START
IF session is not active THEN
  START session
END IF
IF session IP address is not set THEN
  SAVE current user IP address into session
END IF
IF session browser information is not set THEN
  SAVE current browser user agent into session
END IF
IF session regeneration time is not set THEN
  SAVE current time into session
END IF
IF user is logged in THEN
  IF stored IP address is different from current IP address
    OR stored browser information is different from current browser information THEN
     CLEAR session data
     DESTROY session
     START new session
     RETURN
  END IF
END IF
IF current time - saved regeneration time > 300 seconds THEN
  REGENERATE session ID
  UPDATE session regeneration time
END IF
END
```

Figure 8: Pseudocode of Secure Session Initialization and Protection Mechanism

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume X Issue X October 2026

Figure 8 shows the implementation of secure session management in the system. The mechanism validates user session information and periodically regenerates session ID to improve security and prevent unauthorized access.

```
START
RECEIVE password from user
GENERATE hashed password using password hashing function
CREATE SQL query:
   UPDATE users table
   SET password = hashed password
   WHERE user number = selected user
PREPARE SQL statement
BIND:
    - Hashed password
    - User number
EXECUTE SQL statement
STORE hashed password into database
END
```

Figure 9: Password Stored in Database

Figure 9 shows the implementation of password hashing in the system. User passwords are converted into hashed values before being stored in the database to improve account security and protect sensitive information.

Implementation of Student Modules

The student module was developed to give students a safe and friendly environment to write their online assessments. The module allows students to take online assessments, provide answers to both subjective and objective questions, check their assessment results and maintain their personal information. Webcam and behavioral monitoring are enabled automatically when conducting assessments in the system to promote assessment integrity. Students' answers are also saved automatically to prevent any loss of data while taking assessments.





Figure 10: Available Assessment Page

Figure 11: Online Assessment Taking Page

Figure 10 displays the available assessment page where students can view, and access assigned online assessments while Figure 11 shows the online assessment page where students answer questions under real-time AI monitoring.

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume X Issue X October 2026

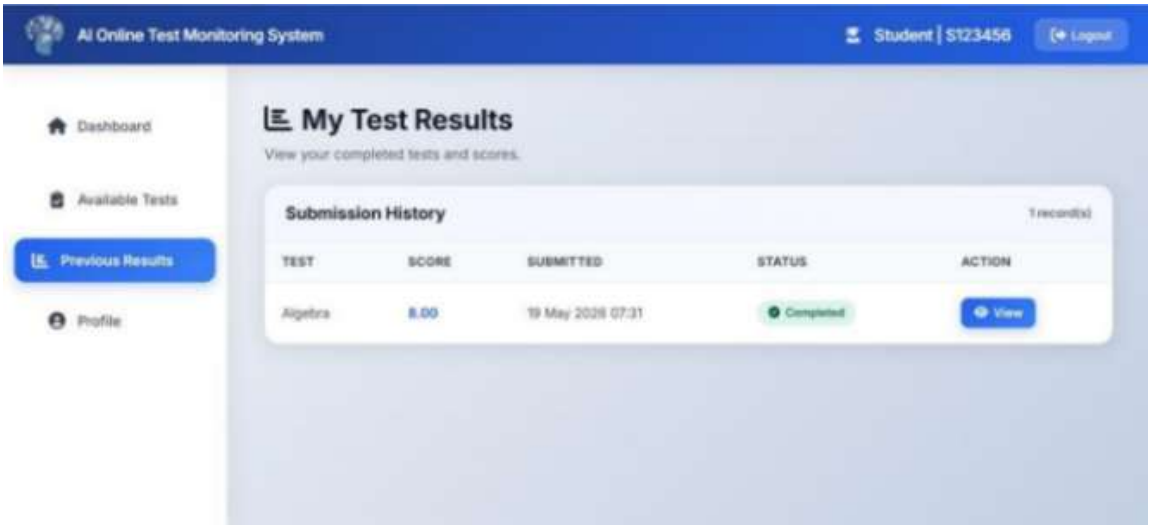

Figure 12: View Result Page

Figure 12 shows the result page where students can view their assessment scores, submission status and previous assessment results after completing an online assessment.

Implementation of Admin Modules

The admin module was developed to perform administrative tasks within the system. Using this module, administrative staff will be able to administer users, courses, class sections, reports and audits among other activities. The admin module has capabilities for centralized administration of the online assessment environment.

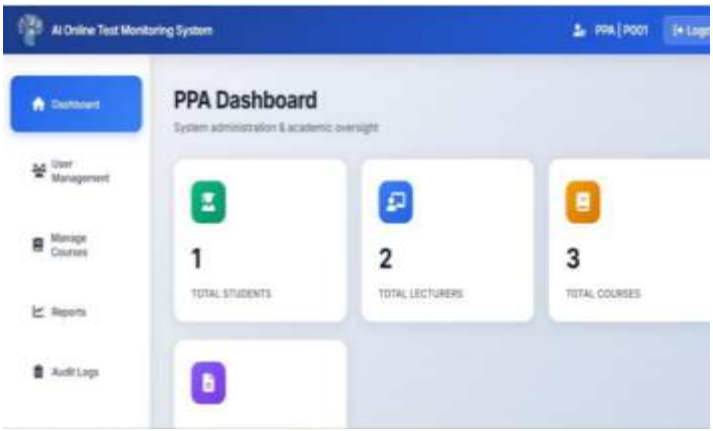

Figure 13: Admin Dashboard

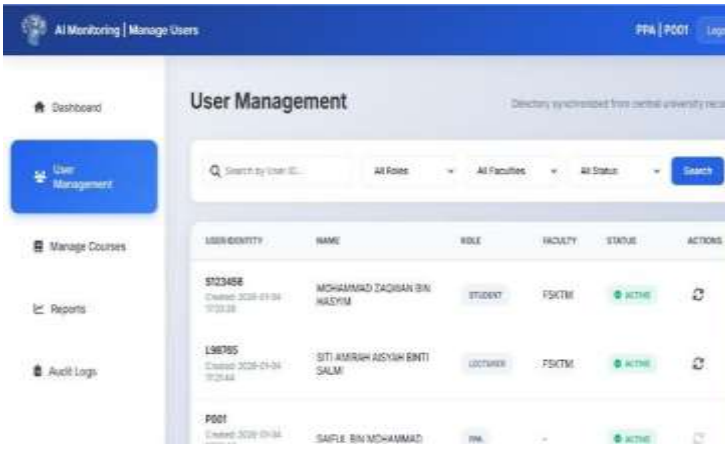

Figure 14: User Management Page

Figure 13 shows the admin dashboard which provides an overview of system information such as the total number of students, lecturers and courses. Figure 14 presents the user management page where admin staff can view and manage user accounts within the system.

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume X Issue X October 2026

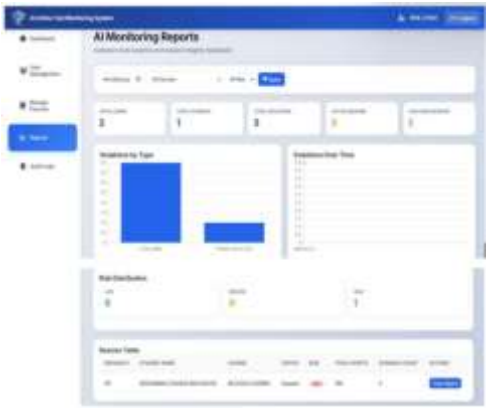

Figure 15: Report Page



Figure 16: Student Monitoring Report Page

Figure 15 shows the report page which provides an overview of monitoring statistics and student violations while Figure 16 presents the student monitoring report page where detailed monitoring records, event logs and evidence captured during online assessments can be reviewed.

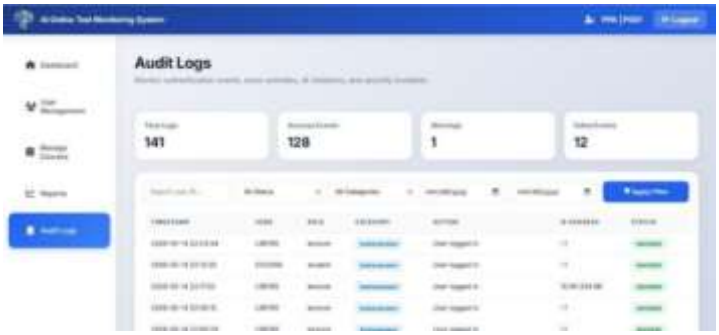

Figure 17: Audit Log Page

START
FUNCTION logAudit
RECEIVE:
- Database connection
- User number
- User role
- Action category
- Action performed
- Target user
- Test ID
- Additional details
- IP address
- Status
IF IP address is empty THEN
GET user IP address automatically
END IF
CREATE SQL query:
INSERT audit log information into audit_logs table
PREPARE SQL statement
BIND:
- User number
- Role
- Category
- Action

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS | Volume X Issue X October 2026

- Target user
- Test ID
- Details
- IP address
- Status

EXECUTE SQL statement
SAVE audit log record into database
SAVE current timestamp automatically
END FUNCTION
END

Figure 18: Pseudocode of Audit Logging Function

Figures 17 and 18 show the implementation of the audit logging module which records user activities and security-related events to support system monitoring and accountability.

Implementation of Lecturer Modules

The lecturer module was developed to assist with the management and monitoring of assessments taken online. The lecturers can utilize the module to generate assessments, alter and enter assessment questions, mark students' submissions and generate assessment reports. This module also has real-time monitoring functionality for enabling lecturers to monitor students taking assessments as well as examine alerts related to any suspicious activity detected by the monitoring system.



Figure 19**:** My Class Sections Page

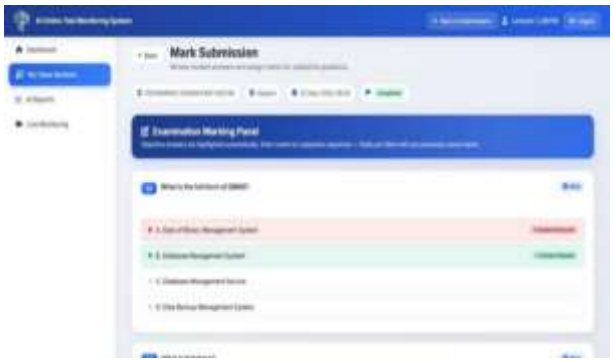

Figure 20: Mark Submission Page

Figures 19 and 20 show the main interfaces of the lecturer module, allowing lecturers to manage class sections, review student submissions and assign marks for completed online assessments.



Figure 21: Real-time Monitoring Page

```
FOR each webcam frame DURING exam:
Detect faces using MediaPipe Face Detection
IF face count == 0 for more than 5 seconds:
trigger "No Face Detected"
IF face count > 1 for more than 2 seconds:
trigger "Multiple Faces Detected"
    Extract facial landmarks using MediaPipe Face Mesh
    Calculate:
        eyeCenter = (leftEye.x + rightEye.x) / 2
        headMovement = nose.x - eyeCenter
    IF absolute(headMovement) > 0.035 for more than 2.5 seconds:
        trigger "Prolonged head deviation from Screen"
    Run TensorFlow COCO-SSD object detection
    IF object == "cell phone" AND confidence > 0.75 for more than 2 seconds:
        trigger "Mobile Phone Detected"
    IF person count > 1 for more than 2 seconds:
        trigger "Multiple Persons Detected"
    IF violation triggered:
        send alert to lecturer
        save monitoring log
        update violation count
```

Figure 22: Pseudocode of Detection Process

Figures 21 and 22 show the real-time monitoring module, which detects suspicious behavior during online assessments and notifies lecturers through alerts and monitoring logs. This page allows lecturers to supervise students through live webcam streaming, activity monitoring, and automatic real-time notifications. The system constantly checks the webcam frames to identify any suspicious actions. The software uses MediaPipe Face Detection to check if the student is looking at the camera and identifies suspicious actions like no faces or multiple faces being captured in the frame. MediaPipe Face Mesh is then used to track facial features such as the eyes and nose to detect head and eye movements. An instance in which a student turns their head away from the computer screen for more than 2.5 seconds triggers an alert based on the predefined threshold. These parameters were established to provide consistent alert-triggering criteria. However, they have not been formally optimized or experimentally validated, and their effects on detection accuracy and false-positive rates require further evaluation.

The monitoring architecture comprises three distinct mechanisms. Firstly, TensorFlow COCO-SSD, which uses a CNN-based model for object detection, is employed to detect any suspicious objects such as cell phones based on confidence scores [20]. Secondly, MediaPipe provides face detection and facial-landmark estimation. Face detection is used to determine face presence and the number of visible faces, while Face Mesh provides landmark coordinates for the implemented head-deviation calculation. Thirdly, browser-event monitoring records selected user-interface events, such as tab switching and copy-and-paste actions. These browser events are detected through browser-side event handling rather than through CNN inference or facial-landmark estimation.

User Acceptance Testing

User Acceptance Testing (UAT) was conducted by students, lecturers and admin staff to evaluate the usability and functionality of the proposed system. The results indicated that the system was easy to use, met user requirements and was accepted by the respondents.

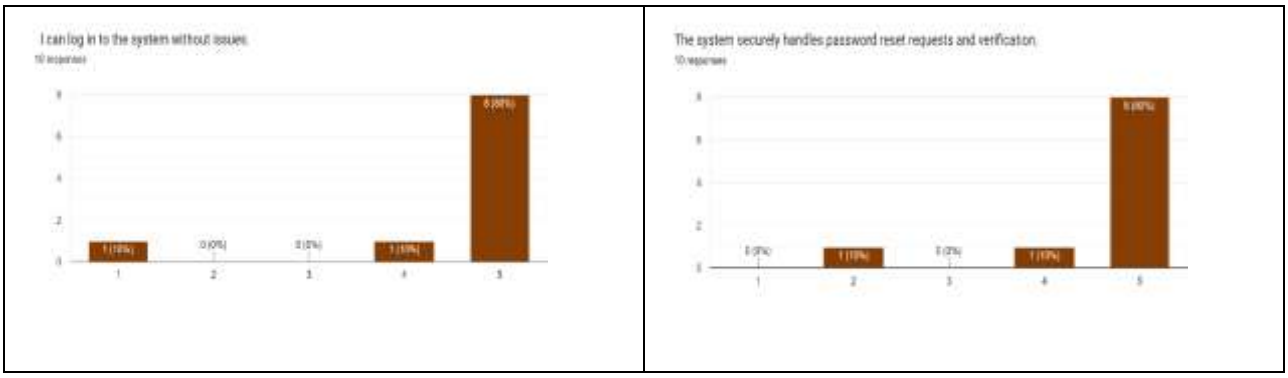

Figure 23: Result of Login and Password Reset Module

Figure 23 presents the evaluation results for the Login and Password Reset Module, indicating that most students were satisfied with the functionality and usability of the implemented features.

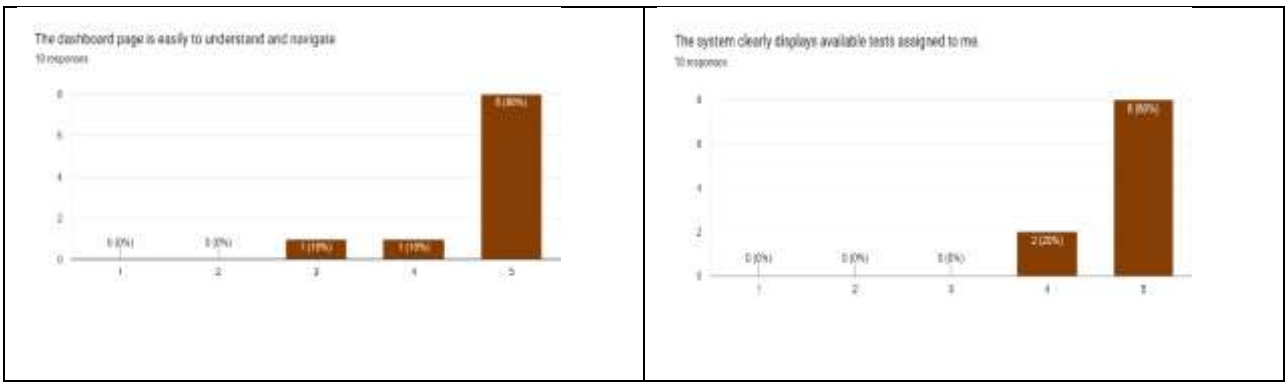

Figure 24: Result of Student Module

Figure 24 presents the evaluation results for the Student Module showing that the students were satisfied with the dashboard and online test features provided by the system.

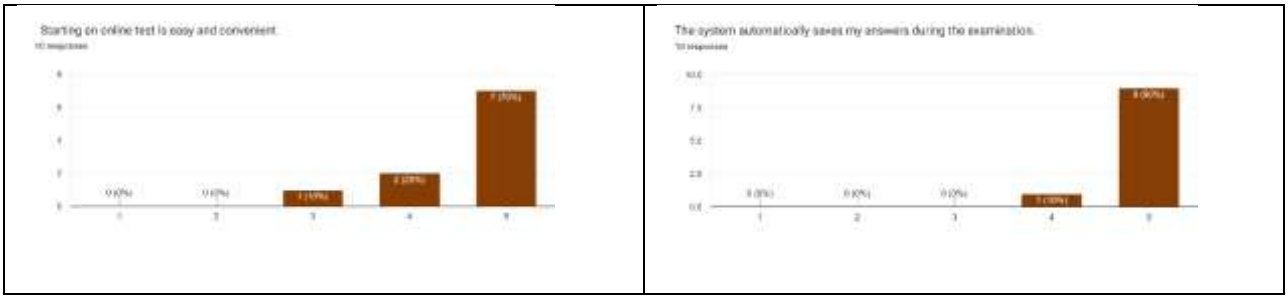

Figure 25: Result of Test Module

Figure 25 presents the evaluation results for the Test Module, indicating that students were satisfied with the online test functionality and automatic answer-saving feature provided by the system.

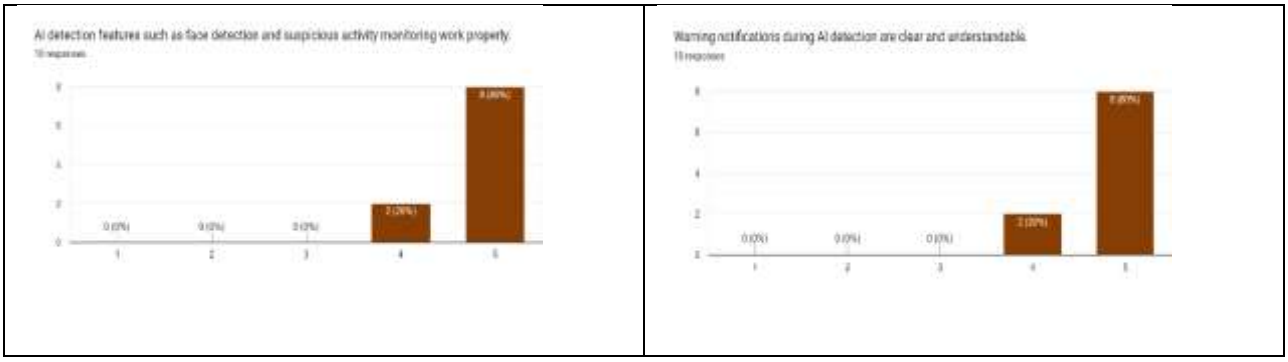

Figure 26: Result of Detection Module

Figure 26 presents the evaluation results for the Detection Module, showing that students were satisfied with the AI detection features and warning notifications provided by the system during online tests.

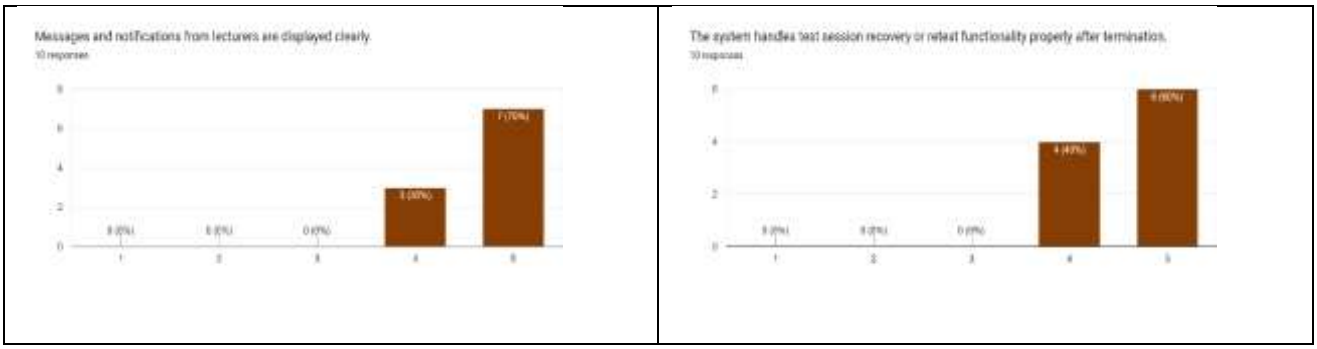

Figure 27: Result of Message and Retest Module

Figure 27 presents the evaluation results for the Message and Retest Module, indicating that students were satisfied with the communication features and retest functionality provided by the system.

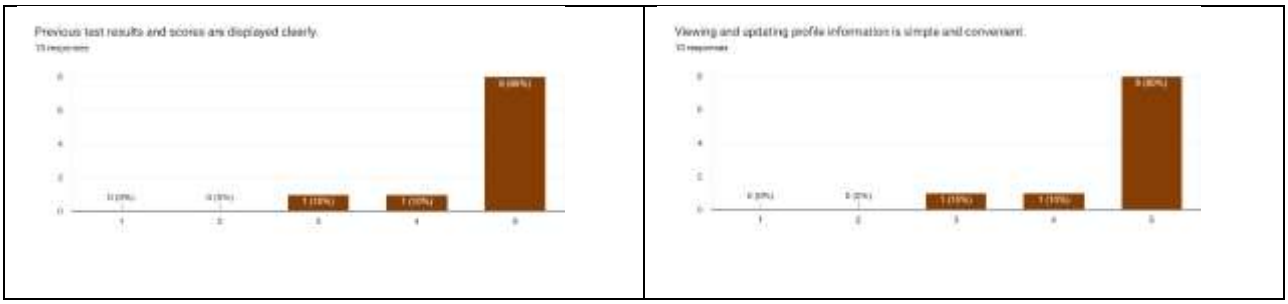

Figure 28: Result of Result and Profile Module

Figure 28 presents the evaluation results for the Result and Profile Module, showing that students were satisfied with the test result display and profile management features provided by the system.

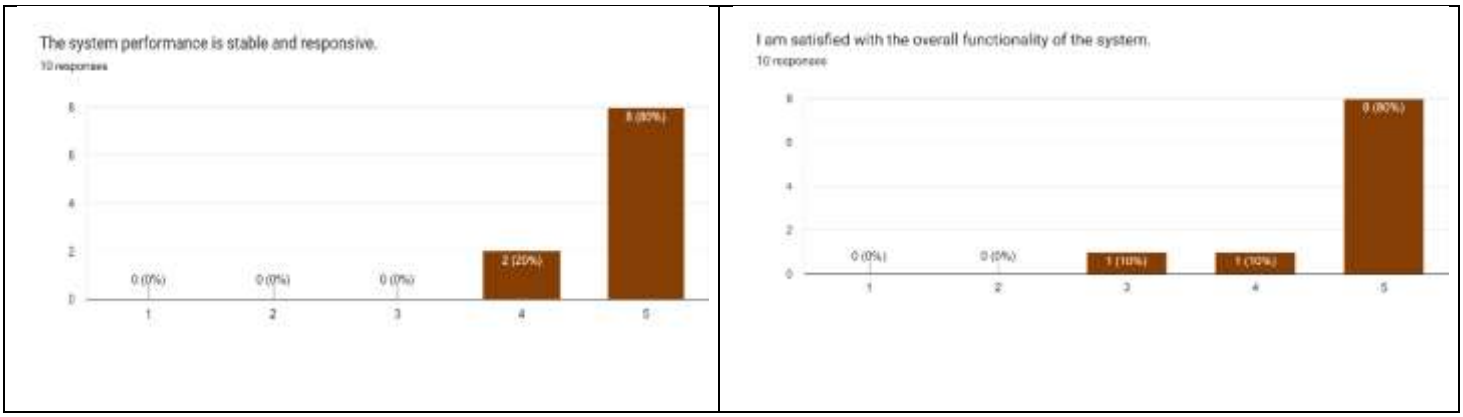

Figure 29: Result of Satisfactory of The System

Figure 29 presents the overall evaluation results of the system, indicating that students were satisfied with the system performance, functionality and user experience.

CONCLUSION

The developed system provides increased security, validity and effectiveness in carrying out online assessments through the incorporation of real-time monitoring, deep learning-based behaviour detection and assessment management components in one web-based platform. The system allows lecturers to create and manage their online assessments, monitor students in real time, analyze the monitoring reports generated from the monitoring process and analyze students' online assessment submissions and students to take part in monitored online assessments. The incorporation of TensorFlow, MediaPipe, WebRTC and CNN-based detection models enables the system to detect potential suspicious behaviours such as the absence of faces, multiple faces, use of mobile phones during the assessment, switching tabs, copying and pasting activities and abnormal head movement. Security features in place include password hashing, password reset, protection against account lockout, data validation and session management among others. Testing and evaluation outcomes have shown that the developed application is functioning well, reliable and easy to use. The development of the technology in the future will consist of enhancing its detection capabilities, decreasing the number of false alarms, adding more monitoring methods as eye-tracking and voice analysis, working on mobile devices. In the end, the technology reached its goals and proved the effectiveness of deep learning technology for maintaining academic integrity in e-learning.

REFERENCES

1. R. Yelkar, R. V. Pawar, D. A. Khopkar, P. P. Ratwadkar, and R. O. Singh, “Online Exam Proctoring System,” International Journal of Creative Research Thoughts, vol. 10, no. 1, pp. e954–e959, Jan. 2022. [Online]. Available: https://ijcrt.org/papers/IJCRT2201560.pdf.
2. Universiti Tun Hussein Onn Malaysia (UTHM), UTHM Distance Learning Assessments Guideline. UTHM, 2020. [Online]. Available: https://cad.uthm.edu.my/images/COVID-19/UTHM_Assessment_Guidelines_120420.pdf.
3. Universiti Tun Hussein Onn Malaysia (UTHM), Author UTHM Instructor Manual. [Online]. Available: https://cad.uthm.edu.my/images/manual_pengguna/author-document-instructor_part1.pdf. [Accessed: Nov. 27, 2025].
4. K. Jain, A. A. Ross, and K. Nandakumar, Introduction to Biometrics. New York, NY, USA: Springer, 2011, doi: 10.1007/978-0-387-77326-1.
5. S. Essahraui et al., “Deep Learning Models for Detecting Cheating in Online Exams,” Comput. Mater. Contin., vol. 85, no. 2, pp. 3151–3183, 2025. https://doi.org/10.32604/cmc.2025.067359
6. C. Topuz, E. Saka, Ö. F. Fatsa, and E. Kurşun, “Emerging trends of online assessment systems in the emergency remote teaching period,” Smart Learning Environments, vol. 9, no. 1, Art. no. 17, 2022, doi: 10.1186/s40561-022-00199-6.

7. Santhakumar, “Human proctoring vs. AI proctoring: A comprehensive comparison,” Talview, Oct. 10, 2024. [Online]. Available: https://blog.talview.com/en/human-proctoring-vs-ai-proctoring. [Accessed: Nov. 26, 2025].
8. O. L. Holden, M. E. Norris, and V. A. Kuhlmeier, “Academic Integrity in Online Assessment: A Research Review,” Jul. 14, 2021, Frontiers Media S.A. doi: 10.3389/feduc.2021.639814.
9. V. A. S., “AI-based proctoring system for cheating prevention in online exams,” International Research Journal of Education and Technology, 2025.
10. M. Ramzan, A. Abid, M. Bilal, K. M. Aamir, S. A. Memon, and T. S. Chung, “Effectiveness of Pre-Trained CNN Networks for Detecting Abnormal Activities in Online Exams,” IEEE Access, vol. 12, 2024, doi: 10.1109/ACCESS.2024.3359689.
11. P. K. Sahu and V. Kumar, “AI-Based Proctoring System for Online Tests,” International Journal of Research Publication and Reviews, vol. 6, no. 4, pp. 12939–12946, Apr. 2025. [Online]. Available: https://ijrpr.com/uploads/V6ISSUE4/IJRPR43415.pdf
12. F. Fitter and S. Hunt, “How AI can end bias,” SAP Insights, Jul. 24, 2024. [Online]. Available: https://www.sap.com/insights/viewpoints/how-ai-can-end-bias.html. [Accessed: Nov. 26, 2025].
13. The University of Texas at Dallas, “Honorlock FAQs and Instructions for Students.” [Online]. Available: https://ets.utdallas.edu/testing-center/honorlock/honorlock-faqs-and-instructions-for-students-3/. [Accessed: Oct. 23, 2026].
14. “Resource Detail.” Accessed: Nov. 27, 2025. [Online]. Available: https://inscribe.education/main/2u/6754110229500883/compositions/6749461749601517?backToListTab=resources
15. Verificient Technologies Inc., Student Guide: How to Use Proctortrack for Live Proctored Exams: Level 4—ProctorLive AI. [Online]. Available: https://cmva.com/wp-content/uploads/2020/08/ProctorLive-AI-Hosted-Flow-Student-Manual.pdf.
16. M. S. Irsyad, Z. Che Embi, and K. I. Ghauth, “Assessing the Efficiency of Deep Learning Methods for Automated Vehicle Registration Recognition for University Entrance”, JIWE, vol. 3, no. 2, pp. 57–69, Jun. 2024.
17. W. I. W. Wan Noor Asmawi and S. P. . Kumaresan, “Real-Time Posture Monitoring for Effective Exercise Using MediaPipe Python”, JIWE, vol. 5, no. 1, pp. 236–251, Feb. 2026.
18. WebRTC, “WebRTC: Real-time communication for the web,” WebRTC. [Online]. Available: https://webrtc.org/. [Accessed: Sep. 2, 2026].
19. Node.js, “Node.js: Run JavaScript Everywhere,” Node.js. [Online]. Available: https://nodejs.org/. [Accessed: Sep. 2, 2026].
20. G. L. Moepi, T. E. Mathonsi, T. Muchenje, and T. M. Tshilongamulenzhe, “Comparative evaluation of COCO-SSD prototype and YOLOv8 for agricultural surveillance systems,” in Proc. 2026 6th Int. Conf. Electr., Comput. Energy Technol. (ICECET), Rome, Italy, 2026, pp. 1–6, doi: 10.1109/ICECET65726.2026.11632459.