University of Central Punjab

BSSE FINAL PROJECT

INVIGILEYE

Project Advisor

Ms. Saher Zia

Presented by:

Group ID: S25SE044

Registration #	Name
L1S22BSSE0011	Muhammad Faizan Ashraf
L1S22BSSE0151	Zieshan Ali
L1F21BSSE0238	Uzair Ali

Faculty of Information Technology & Computer Science

University of Central Punjab

Software Requirements Specification

Version 1.0

INVIGILEYE

Advisor: Ms. Saher Zia

Group: S25SE044

Member Name	Primary Responsibility
Muhammad Faizan Ashraf	- Use Case Diagram - Class Diagram - Documentation
Zieshan Ali	- Sequence Diagram - Data Flow Diagram - ERD
Uzair Ali	- Documentation

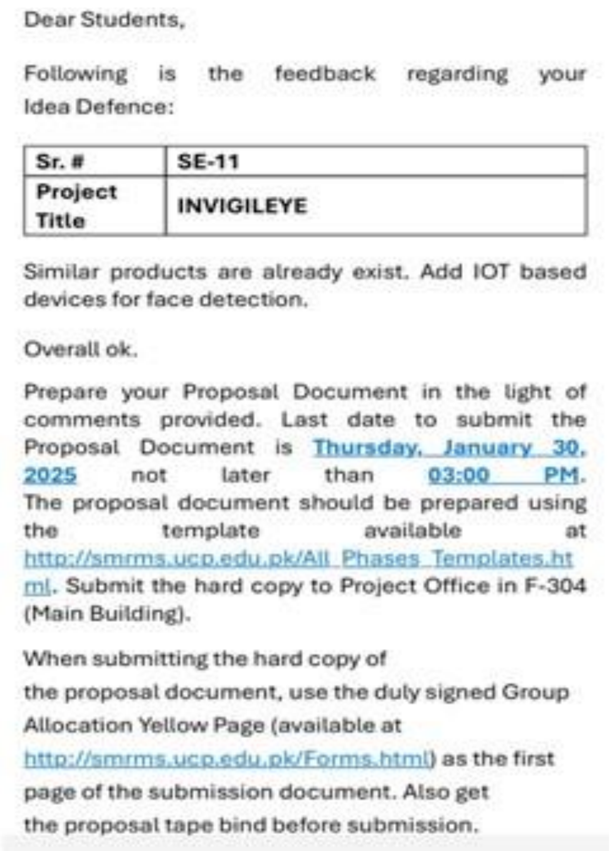
Table of Contents

Revision History

Name	Date	Reason For Changes	Version

Previous Phases Feedback

Idea Defense Feedback (Screenshot)



Dear Students,

Following is the feedback regarding your Idea Defence:

Sr. #	**SE-11**
Project Title	**INVIGILEYE**

Similar products are already exist. Add IOT based devices for face detection.

Overall ok.

Prepare your Proposal Document in the light of comments provided. Last date to submit the Proposal Document is Thursday, January 30, 2025 not later than 03:00 PM. The proposal document should be prepared using the template available at http://smrms.ucp.edu.pk/All_Phases_Templates.html. Submit the hard copy to Project Office in F-304 (Main Building).

When submitting the hard copy of the proposal document, use the duly signed Group Allocation Yellow Page (available at http://smrms.ucp.edu.pk/Forms.html) as the first page of the submission document. Also get the proposal tape bind before submission.

Fig.1: Idea Feedback

Abstract

InvigilEye is an AI-powered invigilation system for physical examinations. It enhances exam integrity and fairness through facial recognition for automated attendance and unauthorized face detection, posing estimation to flag suspicious behavior and instant alert to invigilators. The system will also be able to save snapshots (visual evidence) for post exam reports, while admins will be able to configure exam parameters (e.g., venues, students' data etc.) and be able to download reports. InvigilEye reduces 70-80% of manual workload by applying ML models and integrating cameras with desktop interface, ensures unbiased monitoring and provides insights-modernizing exam security from student authentication to detailed reporting.

Introduction and Background

1.1 Product (Problem Statement)

Examination integrity mainly remains such a highly critical concern for almost all educational institutions, especially during any sort of physical assessments, where manual invigilation is still the basic norm. Despite the highly active presence of human invigilators, the traditional monitoring process is also often prone to such problems like fatigue, human error, and subjectivity, leading to some major potential malpractice and unfair assessments.

InvigilEye majorly aims to simply address all these challenges by introducing a highly intelligent, AI-powered invigilation system. This system also provides automated attendance through a technology of facial recognition, which detects overall suspicious behavior using pose estimation, and generates important visual evidence and real-time alerts, offering a highly modern, tech-driven approach to overall physical exam supervision.

The primary motivation behind InvigilEye mainly stems from the overall lack of any reliable, integrated tools for physical exam monitoring. Most current solutions mainly focus solely on online exams, ignoring the part of behavior analysis, or just require continuous manual supervision. InvigilEye fills this pretty huge gap by just simply combining computer vision, machine learning, and real-time video processing to deliver a pretty robust, unbiased, and highly efficient exam monitoring experience.

By majorly minimizing the overall manual workload and simply standardizing most exam security practices, InvigilEye not only contributed to strengthening the complete academic integrity but also heavily supports in maintaining overall credibility and fairness in all sorts of assessments.

1.2 Background

Cheating in exams is still a big issue in most of the educational instructions. Physical invigilation might be prone to errors because invigilators might get tired in long exam sessions and cheating often goes unnoticed. We have many systems for online examinations, but there are no or very few systems for physical examinations.

After finding this gap we are trying to build a system that will focus on physical examinations through its AI integration to watch students, mark attendance with face recognition and detect suspicious activities by pose estimation.

1.3 Scope

The following features which are in-scope and out-scope of InvigilEye are:

1.3.1 In-Scope Features:

- **Automate Attendance:** It automatically verifies the student's attendance during exam using facial recognition system.

- **Cheating Detection:** It determines suspicious activity through pose estimation they may include cheating.
- **Alert System:** It alerts the invigilators about suspicious activities and about unknown students.
- **Snapshot Capturing:** It automatically captures the snapshots of suspicious actions for review and evidence.
- **Admin Dashboard:** It provides features for admins to login, set venues, manage exam sessions, and students' records.
- **Invigilator Dashboard:** It also provides features of invigilators to login, ask for UMC and demand material.
- **Exam Report:** It automatically generates downloadable exam and attendance report.

1.3.2 Out-Scope Features:

- Support for web/mobile platform.
- Integration with LMS.
- Grading System.

1.4 Objective(s)/Aim(s)/Target(s)

The aim of this project is:

- Building a system cut down manual invigilation work by 70–80%.
- Building a system that automatically detects and highlights suspicious activities like passing notes or unusual movements during exams.
- Building a system that automatically verifies attendance and identifying unauthorized students during exams.
- Building a system that sends instant real-time alerts to invigilators during any sort of suspicious actions.
- Building a system that helps the examination department by providing post-exam reports highlighting attendees and exam details.
- Building a system that provides an interactive desktop-based dashboard for both admins and invigilators where invigilators can easily monitor live alerts, call UMC, ask for material (e.g., extra sheets), and review exam reports.

1.5 Challenges

During the development of this system, we might face the following challenges:

- Train ML models for facial recognition and pose estimation.
- Integrating different ML models to work at the same time.
- Achieving high accuracy of models.
- Ensuring that facial recognition works correctly in different light conditions, angles, and face positions.
- Ensuring the security of data.

1.6 Learning Outcomes

We will gain a practical grasp of artificial intelligence (AI), computer vision, and how these technologies are used for solving issues in the real world via our part in this project.

To enable real-time examination supervision, we will create a desktop program. We will learn how to build systems and create user-friendly interfaces thanks to this.

Learning the proper safeguarding and privacy protection protocol concerning student information will be one of our main objectives.

This project primarily aims at addressing the current challenges within educational institutions. It will enhance our skills in dealing with technological problem-solving while creating a system that encourages academic integrity and simplifies administrative processes. In particular, we will learn to integrate new technological tools, reflect on ethical issues, and implement them in educational settings.

1.7 Nature of End Product

Our project will consider the end product when it contains the following functions:

- Automated attendance via facial recognition.
- Cheating detection through pose estimation.
- Generates Real-time alerts and post-exam report.
- Dashboard for admin and invigilators.

1.8 Completeness Criteria

The evaluation criteria are as follows:

Sr. No.	Criteria	Weightage %
1	Facial Recognition System (Attendance & Identity Verification)	25
2	Suspicious Activity Detection (Pose Estimation)	25
3	Integration	20
4	Interface	20
5	Documentation	10

Table 3: Completeness Criteria

1.9 Related Work/ Literature Survey/ Literature Review

1. **Online Proctoring Platforms**
   - [1]**Proctorio** mainly offers an overall AI-powered, browser-based monitoring system for conducting remote exams.
   - [2]**ProctorU** simply combines live and automated proctoring with two major features, like ID checks and session recording.

Limitations: Both are online-only systems with no physical contribution of any kind, lack in-hall support, overall behavioral analysis (pose estimation), and no real-time alerts.

2. **Traditional and CCTV-Based Invigilation**
   - **Manual Supervision:** Majorly relies on assigned staff simply walking the exam hall. It's highly prone to a lot of human error and oversight.
   - **CCTV Monitoring:** Some universities use some basic static cameras, but without any usage or implication of AI, they cannot detect any sort of cheating or generate automated reports about those incidents.

1.10 Document Conventions

The abbreviations used in this document are given below:

AI	Artificial Intelligence
GPU	Graphics Processing Unit
GUI	Graphical User Interface
ML	Machine Learning
SRS	Software Requirements Specification
RTSP	Real Time Streaming Protocol
OpenCV	Python library of face scanning
YOLOv8	You Only Look Once Version 8 (Python library for pose estimation)
UMC	Unfair Means Case
PDF	Portable Document Format
CSV	Comma Separated Values
ERD	Entity Relationship Diagram

Table 4: Document Conventions

2. Overall Description

2.1 Product Features

InvigilEye offers the following key features:

- **Automate Attendance:** When students sit on their seats, the system will automatically scan their face and mark attendance through the facial recognition system.
- **Cheating Detection:** During the exam if a student tries to look behind or sideways, the system considers it as a violation exam rule through pose estimation.
- **Alert System:** If an unknown person sits in the exam hall or the system detects suspicious behaviors, it instantly generates alerts to invigilators. The invigilator has a choice to consider it as a false alarm and ignores it.
- **Snapshot Capturing:** For future evidence the system stores the snapshots of suspicious actions.
- **Admin Dashboard:** It provides features for admins to login, sets venues for exams, handle requests from invigilators, and adds students' records.
- **Invigilator Dashboard:** It also provides features for invigilators to login to the system, manage exam session, ask for UMC and demand material like extra sheets from the exam department.
- **Exam Report:** It generates downloadable exam and attendance report after exam for record, and it is available for both admin and invigilators.

2.2 User Classes and Characteristics

The system is designed for the following user types:

1. **Invigilators**
   - **Use**: Invigilators will be able to handle alerts about unverified or cheating students. They are also able to download exam reports, demand material and call for UMC.
   - **Expertise**: Moderate technical knowledge.

2. **Admin**
   - **Use**: Admin will be able to manage examinations. They upload students' data, set up exam details (such as venue and timing), and review post-exam reports and snapshots. Admins can also respond to material demand and UMC, raised by invigilators.
   - **Expertise**: Moderate technical skills.

3. **Students (Passive Users)**
   - **Use**: No direct interaction with the system - their faces and behavior are analyzed for verification and detection
   - **Expertise**: Not required.

The most critical users to satisfy are **invigilators** and **admins**, as they rely on the system for smooth and effective monitoring during exams.

2.3 Operating Environment

InvigilEye will operate in the following environment:

1. **Hardware**:
   - Camera.
2. **Software**:
   - **Operating System**: Windows (for the GUI).
   - **Development Tools**: Python with OpenCV, TensorFlow, PyTorch
   - **GUI Framework**: Java Swing
3. **Database:**
   - Local storage or lightweight SQL database records.

The software is designed to operate in controlled exam hall environments and must run smoothly with low-latency video feeds and reliable AI processing.

2.4 Design and Implementation Constraints

Here are the risk factors which are holding us back (and how we will oversee it):

- **Bad Lighting:**

  Challenge: Bad Lighting might affect the accuracy of the system – might face difficulty in scanning student's faces.

  Solution: Use WDR cameras + Train models on low light data as well.

- **Budget Issues:**

  Challenges: Schools cannot afford expensive cameras and GPUs.

  Solution: Support affordable (mid-range) cameras and consumer GPUs.

- **Privacy Concerns:**

  Challenges: Students' trust regarding their privacy.

  Solution: No raw images are saved, and snapshots can be deleted easily with a one-click option.

2.5 User Documentations

InvigilEye will include the following helpful documents to increase the usability and adaptability of invigilators:

- **User Manuals (PDF):**
  - Provide step-by-step instructions for cameras and software setup.
  - Teach them about different alert messages.
  - Give suggestions on how to improve the accuracy of the system.
- **Quick Start Guide:**
  - Visual walkthrough of dashboard layout and critical buttons.
- **Video Tutorials:**

- Provide tutorial on how to use the system.

2.6 Assumptions and Dependencies

2.6.1 Key Assumptions

It is highly assumed that all invigilators who will be using this system will be trained before using our InvigilEye system by simply reading the overall user manual and watching a few tutorial videos. The exam environment is mainly expected to have a highly stable power supply and a properly working Wi-Fi connection. Cameras should be correctly installed at the recommended heights and positions to simply ensure that our system works 100% correctly.

2.6.2 Dependencies

The system mainly depends on IP cameras that also support RTSP, and such computers with NVIDIA GPUs to run all the AI models smoothly. It will use all these software tools like Python, OpenCV, and YOLOv8 for its main functions. The system also majorly requires overall access to the school's database for all important student information and a pretty reliable internet connection to work properly.

3. Product Features / Functional Requirements

The proposed system should be able to perform the following tasks:

1. **Student Registration & Face Enrolment**
   - Admins can upload student details (name, roll no, email).
   - System captures and stores unique facial images.
   - Duplicate face entries are prevented.
2. **Automated Attendance**
   - Attendance is marked via facial recognition at entry.
   - Timestamped records are stored with AI verification status.
   - Unrecognized faces are logged for manual review.
3. **Real-Time Behaviour Monitoring**
   - System poses continuously.
   - Alerts are triggered for:
     - Unauthorized or multiple faces
     - Suspicious movements (leaning, looking away)
4. **Evidence & Reporting**
   - Flagged incidents are captured as snapshots.
   - System generates PDF reports with:
     - Attendance
     - Incident images
     - Timestamps

- Reports are downloadable by both admins and invigilators.

5. **System Administration**
   - Admin dashboard for scheduling, venue, and student management.
6. **Alert Handling**
   - Instant visual and audio alerts to invigilators.

3.1 Use Case Diagram

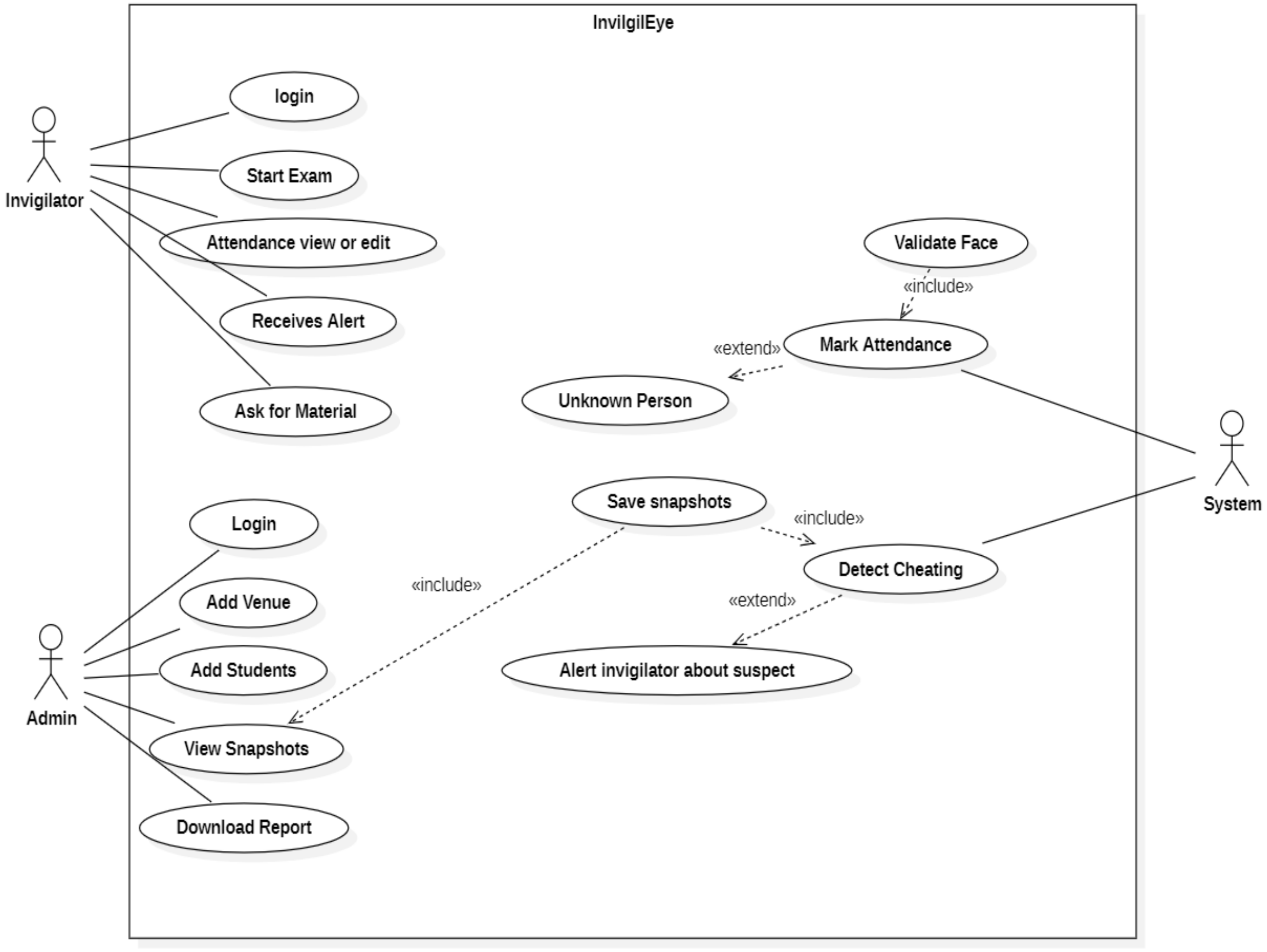


Fig.2: Use Case Diagram

3.2 Exam Setup Use Case

Identifier	UC-1
Purpose	Setup Exam Venues and add Students’ data
Priority	High
Pre-conditions	- Admin login the system

Post-conditions	- Student face data stored.	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Admin Set Exam Venue and Time.	System Flags Success.
2	Admin add students' data.	System Flags Success.
Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	There is a clash in venue.	System responds "Failed"

Table 5: UC-1

3.3 Student Face Registration Use Case

Identifier	UC-2	
Purpose	Automatically mark student's attendance via facial recognition model.	
Priority	High (Core Functionality)	
Pre-conditions	- Invigilator already login. - Exam session has started. - Student's profile image data is preloaded in the database. - Camera is active.	
Post-conditions	- Attendance is stored in the database. - Flags unrecognized faces.	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Students sit at their seats.	Camera starts capturing videos.
2	-	System process captures data using OpenCV.
3	-	If matched: - Mark attendance.
…	-	If not matched: - Flags unrecognized. - Send alert.

Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Camera capture’s data is blurry/low light.	- Adjust brightness and contrast.
2	Still unrecognized after 3 tries.	- Mark him/her “absent”. - Sent alert.

Table 6: UC-2

3.4 Detect Cheating Use Case

Identifier	UC-3
Purpose	Identify suspicious activities by pose estimation.
Priority	High (Core Functionality)
Pre-conditions	- Invigilator already login. - Exam session has started. - Camera is active.
Post-conditions	- Mark “Hot Suspect”, “Suspect”, or “Normal”. - Generate Alerts. - Stores Snapshots

Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Student writing his/her exam.	System continuously calculates his/her pose.
2	-	Triggers alert system detects anomalies(e.g., head turns> 30°).
3	Invigilator views alert	Highlight student on dashboard
Alternate Course of Action		
S#	**Actor Action**	**System Response**
1	Invigilator marks “False Alarm” (e.g., stretching).	System learns from feedback.

Table 7: UC-3

3.5 Generate Remote Use Case

Identifier	UC-4
Purpose	After finishing the exam session, the system will automatically generate exam report.
Priority	Medium
Pre-conditions	- Exam session successfully ended.
Post-conditions	- Generate exam report in the form of PDF/CSV.

Typical Course of Action

S#	**Actor Action**	**System Response**
1	Invigilator or Admin clicks on “Generate Report” button.	- Export PDF/Excel File.

Alternate Course of Action

S#	**Actor Action**	**System Response**
1	Data corruption.	Restores from Database.

Table 8: UC-4

3.6 Analysis and Modeling of Requirements

3.6.1 Class Diagram

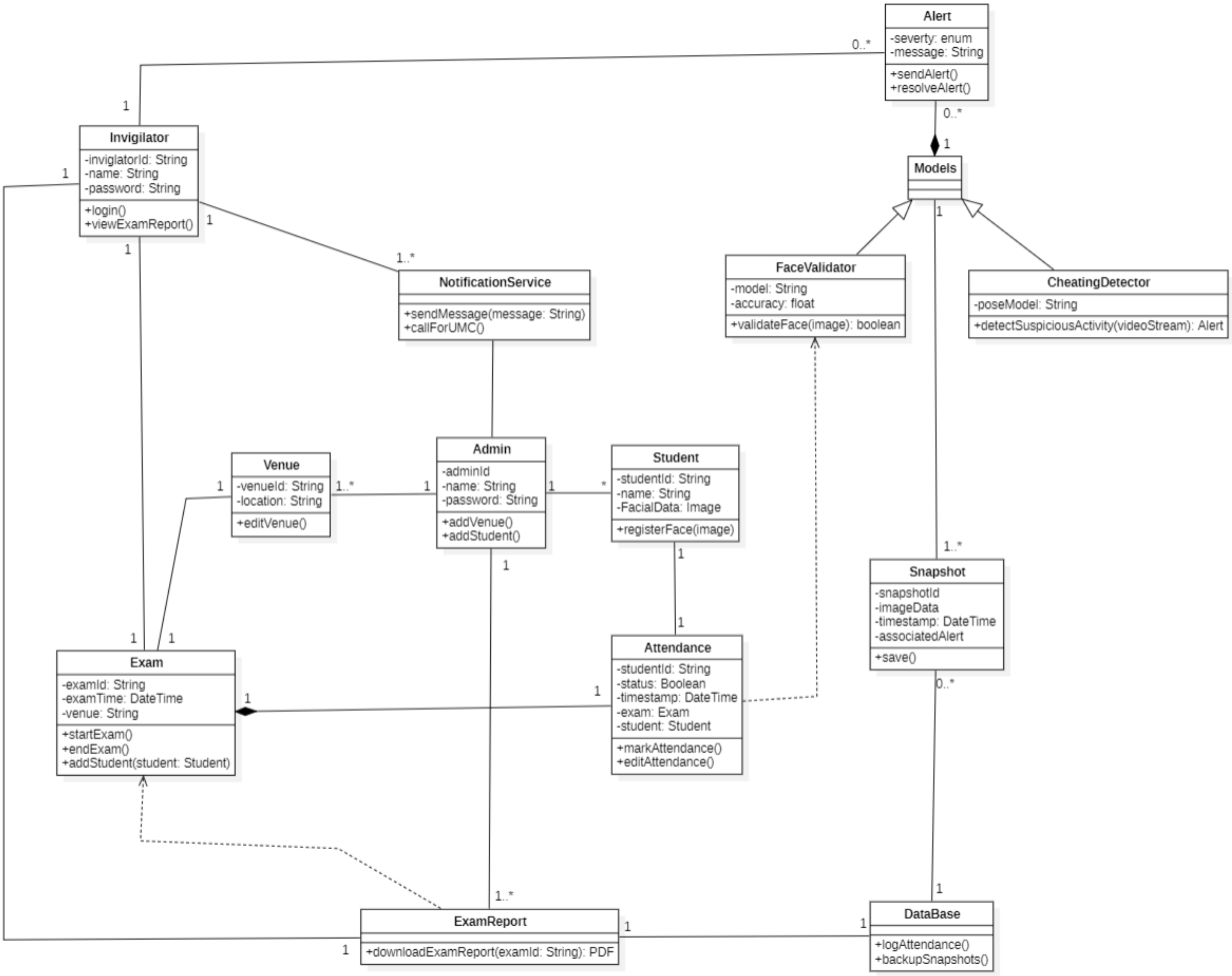


Fig.3: Class Diagram

3.6.2 Entity-Relationship Diagram

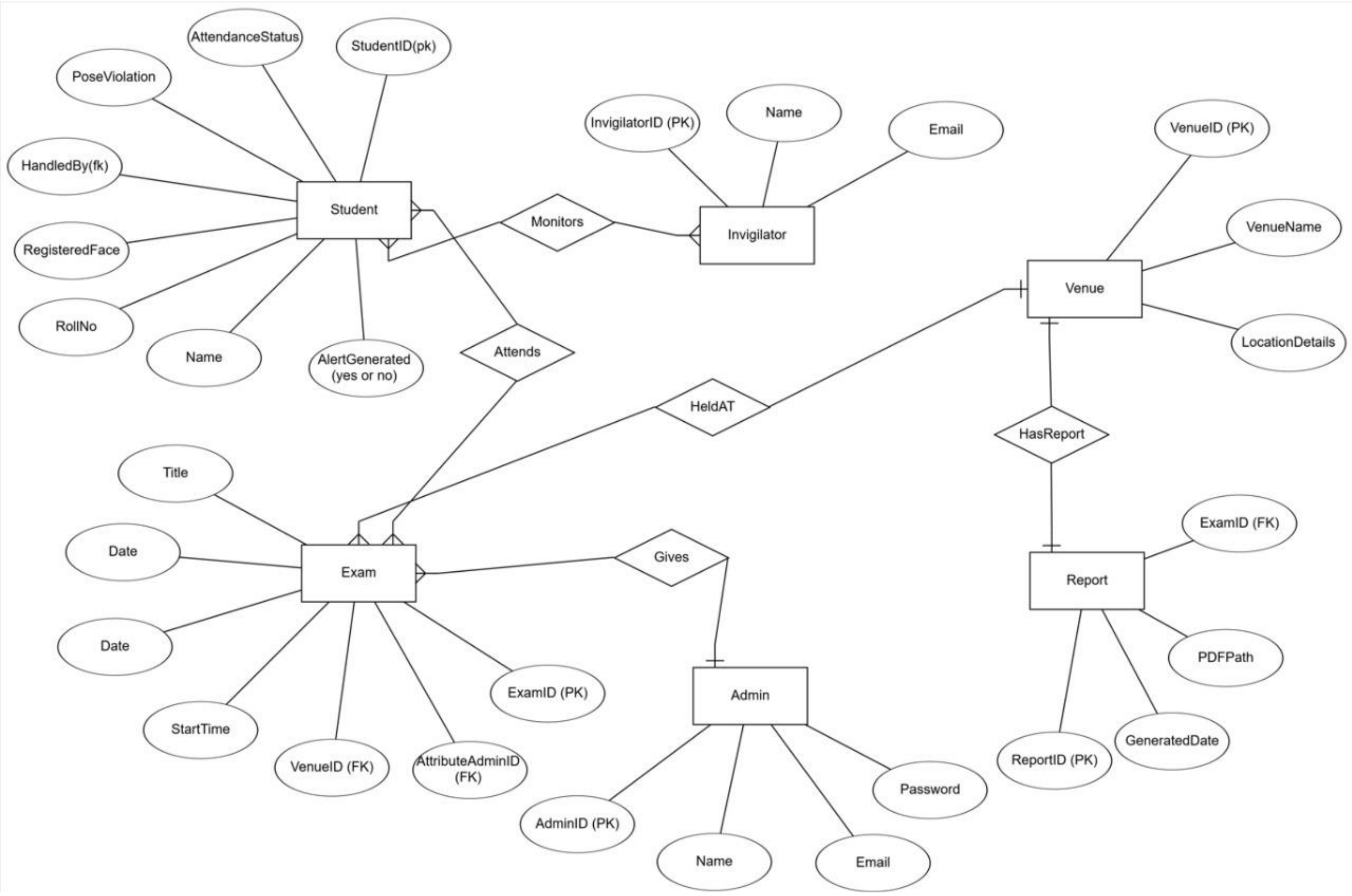


Fig.4: ERD

3.6.3 Data Flow Diagram

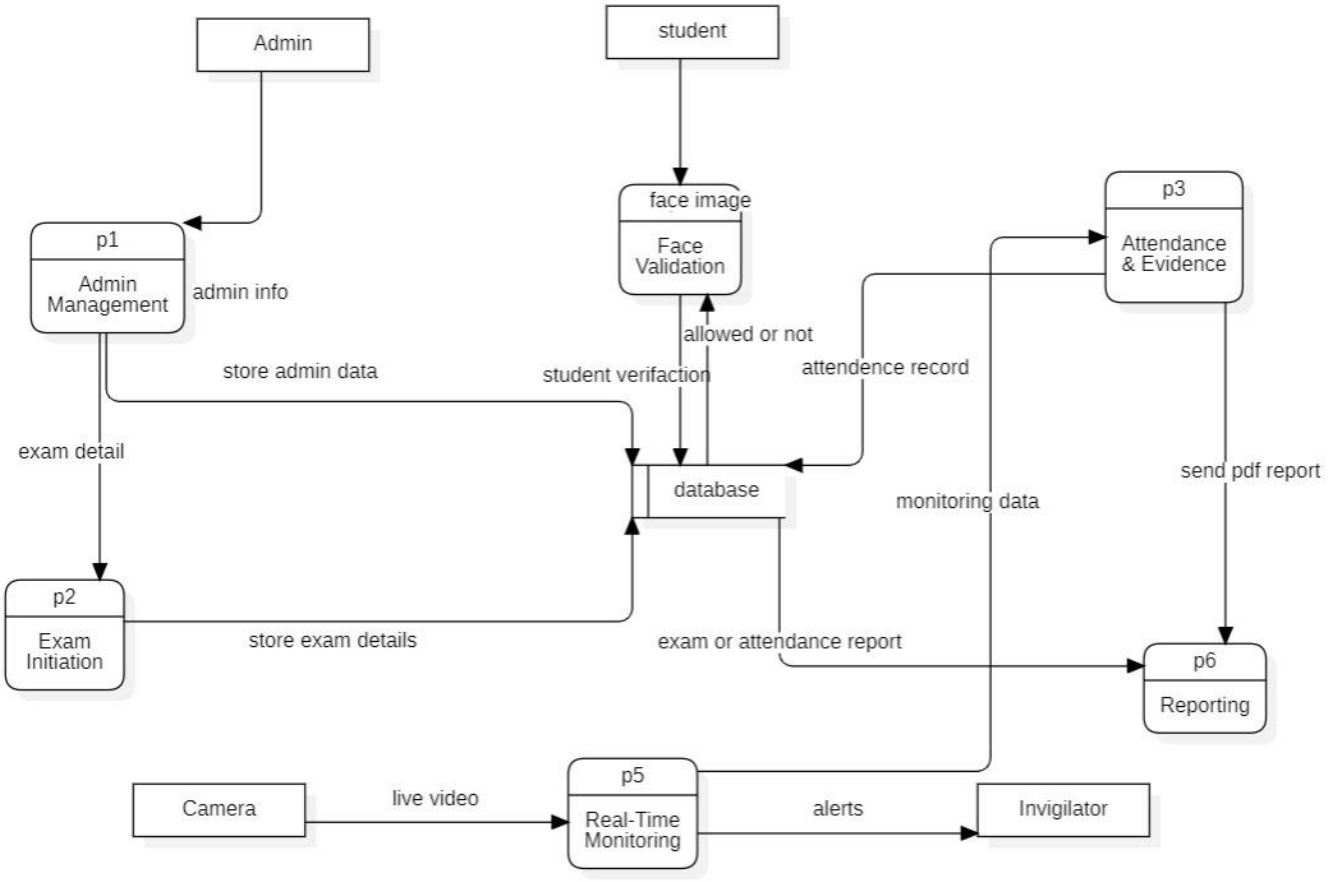


Fig.5: Data Flow Diagram

3.6.4 Sequence Diagram

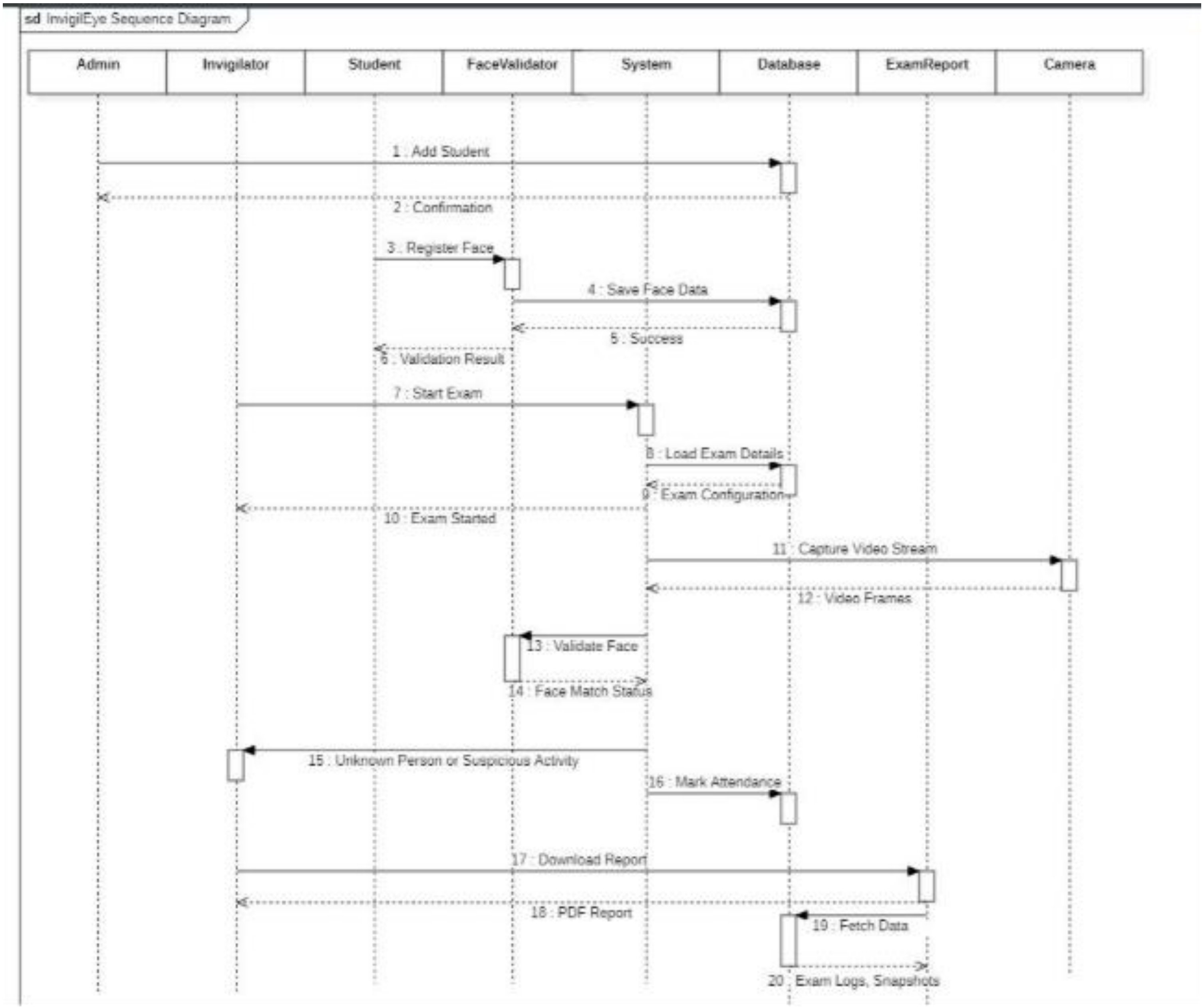


Fig.6: Sequence Diagram

4. Nonfunctional Requirements

4.1 Performance Requirements

The system should efficiently handle real-time exam monitoring while maintaining responsiveness and reliability. Key performance expectations include:

- **Face Matching Speed:** The system must quickly verify student identities through facial recognition, ensuring smooth attendance marking even with multiple concurrent verifications.
- **Report Generation:** Post-exam reports should be generated promptly, allowing invigilators to review exam logs without unnecessary delays.
- **System Availability:** High uptime must be maintained during exam sessions to prevent disruptions in monitoring.
- **Data Processing:** Live video feeds should be analysed in real time with minimal delay to support immediate detection of suspicious activities.

These requirements ensure the system operates effectively under typical exam conditions.

4.2 Safety Requirements

1. Data Protection

To ensure complete compliance with important privacy regulations, the system must:

- Anonymize all the exported reports by simply blurring identifiable faces in every generated PDFs.
- Automatically purge all evidence files containing data records (e.g., snapshots) after a certain defined retention period.

2. Fail-safes

The system must maintain overall functionality during any sort of disruption by:

- Reverting to simple manual attendance if camera feeds are unavailable at that moment.

4.3 Security Requirements

1. Authentication

The system should mainly implement role-based access control using only provided credentials verified against the Invigilator table, it’s done to simply ensure only authorized personnel can access such kind of sensitive functions.

2. Data Protection

Student facial data (Registered Face) should be quite securely stored to maintain overall data integrity and prevent all kinds of unauthorized access to provide high-level security of personal data.

3. Auditability

The system shall easily maintain immutable logs of all Alert table entries. It’s important to simply provide a completely tamper-proof record of huge security events for both compliance and forensic purposes.

4.4 Software Quality Attributes

1. **Usability**

Administrators shall be able to effectively operate the overall dashboard interface after not more than just a few minutes of quite simple and basic training.

2. **Reliability**

The system shall automatically recover from any sort of power interruption while also preserving and keeping all attendance records safe to mainly ensure complete data persistence.

3. **Maintainability**

The architecture shall most importantly follow a modular design pattern with quite clearly separated components for facial validation and reporting functions to mainly facilitate all future updates and troubleshooting.

5. Revised Project Plan.

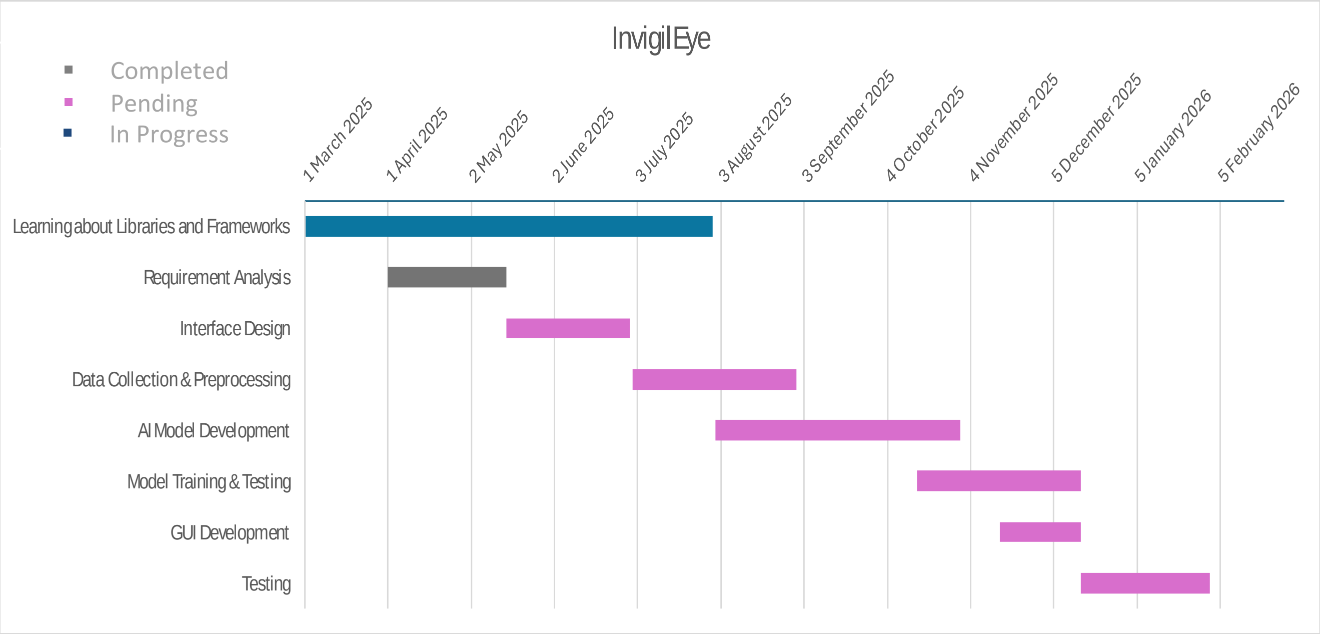


Fig.7: Gantt Chart

6. References

[1] *Proctorio. "Online Proctoring | Proctorio." [Online]. Available: https://proctorio.com. Accessed: May 19, 2025.*

[2] *ProctorU. "Remote Proctoring & Live Online Exam Proctoring." [Online]. Available: https://proctoru.com. Accessed: May 19, 2025.*

[3] *J. Smith, "Inefficiencies in Manual Exam Invigilation," Journal of EdTech, vol. 12, no. 3, pp. 45–60, 2023.*

[4] *H. Lee, "Pose Estimation for Anomaly Detection," IEEE Security, vol. 8, no. 2, pp. 112–125, 2022.*

Appendix A: Glossary

This glossary defines the key terms used throughout the Software Requirement Specification (SRS) for the InvigilEye.

Term	**Definition**
Admin	An admin is a primary system user with administrative privileges responsibilities.
GPU	A specialized processor designed to accelerate graphics rendering and is suitable for AI & ML tasks.
Invigilator	A person who is responsible for managing the system during exam session and has every right to take suitable action against suspicious activities.
Facial Recognition	AI technology that identifies and verifies identities through face and posture of body.
UMC	A formal action again illegal activities during exam session.
Snapshot	An image captured at a specific point of incident.
RTSP	Network protocol to stream video data from camera in real time.
Hot suspect	A label for students for highly suspicious.

Appendix B: IV & V Report

(Independent verification & validation)

IV & V Resource

Name Signature

S#	Defect Description	Origin Stage	Status	Fix Time	
				Hours	Minutes
1					
2					
3					
…					

Table 1: List of non-trivial defects