

School of Sciences and Engineering

Department of Computer Science and Engineering

Spring 2022

Senior Project 1

**Software Requirements Specification Document [SRS]**

**G6 - “Classroom Monitoring Using AI”**

Supervised By:

Dr. Sherif Aly

Dr. [Hesham Eraqi](mailto:heraqi@aucegypt.edu)

* Table of Contents:

1. Introduction:

1.1 Purpose and Scope

1.2 Intended Audience

1.3 Contact information/SRS team members

1.4 Document Conventions

1.5 Definitions and Acronyms

2. Overall Description:

2.1 Product Perspective

2.2 Product Features

2.3 Operating Environment

2.4 User Classes and Characteristics

3. External Interface Requirements:

3.1 Hardware Interfaces

3.2 Software Interfaces

3.3 User Interfaces

3.4 Communication Protocols and Interfaces

4. System Features:

4.1 Overview of System Features

5. Non- Functional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

* -1 Introduction:
* 1.1: Purpose and Scope:

The purpose of the Software Requirements Specification document is to describe and highlight the software specifications of our senior project titled “Classroom Monitoring Using AI” and how it is expected to operate and function. The document will also describe the functional and non-functional requirements needed for the software to operate and achieve its goals by fulfilling the needs of the software’s users.

The scope of this project is to create a safe and fair examination environment for students that eliminates the high rates of cheating during examinations in Egypt, by implementing a system that detects possible cheating incidents in physical exams. Consequently, violating academic integrity creates a critical issue that negatively affects the reputation of the national education system and the quality, and competence of graduates.

* 1.2: Intended Audience:

This Software Requirements Specification document is to describe and highlight the software specifications of the project. The document is mainly intended to be written for the developers of the “Classroom Monitoring Using AI” project and for future references for the users of this project.

* 1.3: Contact Information of SRS Developers Team :

In case you have any inquiries or questions regarding this Software Requirements Specification Document, below are the contact details for the project/SRS team members:

* Noha Abdelkader

Undergraduate Computer Engineering Student at The American University in Cairo

[noha.alshabrawy@aucegypt.edu](mailto:noha.alshabrawy@aucegypt.edu)

* Marwan Amr Awad

Undergraduate Computer Engineering Student at The American University in Cairo

[marwanawad@aucegypt.edu](mailto:marwanawad@aucegypt.edu)

* [Mohamed Elsayed Gamil Elshabshiri](mailto:elshabshiri@aucegypt.edu)

Undergraduate Computer Engineering Student at The American University in Cairo

[elshabshiri@aucegypt.edu](mailto:elshabshiri@aucegypt.edu)

* [Mohamed Ashraf Taha](mailto:mohammedashraf@aucegypt.edu)

Undergraduate Computer Engineering Student at The American University in Cairo

[mohammedashraf@aucegypt.edu](mailto:mohammedashraf@aucegypt.edu)

* [Omar Sherif Elmahdy](mailto:omarmahdy122@aucegypt.edu)

Undergraduate Computer Engineering Student at The American University in Cairo

[omarmahdy122@aucegypt.edu](mailto:omarmahdy122@aucegypt.edu)

* [Youssef Khaled Beshir](mailto:youssefbeshir@aucegypt.edu)

Undergraduate Computer Engineering Student at The American University in Cairo

[youssefbeshir@aucegypt.edu](mailto:youssefbeshir@aucegypt.edu)

* 1.4 Document Conventions:

This document was created based on the IEEE (IEEE Standard 830-1998) template for System Requirement Specification Documents.

* 1.5 Definitions and Acronyms:

*YOLO*: You Only Look Once which is a real-time object detection algorithm that identifies specific objects in videos, live feeds, or images

*ElectronJS*: JavaScript framework used to build cross-platform desktop applications.

*Salting*: The process of adding

*HTTPS*: Hypertext Transfer Protocol Secure

*AWS*: Amazon Web Services

* -2 Overall Description:
* 2.1: Product Perspective:

The project is a standalone project. The project is not built on any other projects. The project is developed using YOLOv3 object detection tool, Keras, TensorFlow Deep learning libraries based on Python programming language, in addition to Flask Python Framework and ElectronJS JavaScript framework.

* 2.2: Product Features:

The functionalities of the product enable the user to be notified of possible cheating incidents in examination processes in a classroom environment. This includes a notification alert to the proctor in the classroom in addition to a snapshot of the possible cheating incident of the student with his/her location in the classroom with an option to dismiss the received possible case in case the proctor does not consider the case as a cheating case. A report is available for the proctor, in case he/she suspects the student’s action. The software generates a report at the end of the examination with detailed information of the dismissed and reported cheating incidents.

* 2.3: Operating Environment:

The operating environment of the project is divided into 2 modules: the backend and the frontend. There are no required specified operating systems to use the system since the system is developed by a cross-platform framework that can operate on different operating systems such as macOS, Microsoft Windows, and Linux.

* 2.4: User Classes and Characteristics:

The project has different user classifications and characteristics, where each classification has its own privileges and authorizations, the classifications and characteristics are listed as follows:

* *System Admins*: System admins are responsible for assigning the users (proctors) to the exam rooms and the exam instances while maintaining the proctors’ registered data and credentials.
* *Proctors*: Proctors log in to the system using the provided credentials and are responsible for viewing and checking the real-time notifications received by the system and take decisions on the possible cheating incidents detected by the software.
* *System Support Team*: The system support team is responsible for maintaining the system and fixing possible issues and bugs that the users may experience while operating the system.
* 2.5: User Documentation:

A user manual will be provided along with the software to ensure that the user can use the software’s features properly and solve the most common problems. In addition to illustrating the user interfaces and their functionalities.

* 3: External Interface Requirements:
* 3.1: Hardware Interfaces:

The software can be connected to an IP security camera or Digital Camera that will be fixed in the middle-top of the classroom to capture the video footage of the examination that will be fed into the software for real-time processing. It is recommended to have a camera of 4MP resolution in order to capture the movement and object details in a classroom setting. The camera also should have a suitable range of a minimum of 8 meters in order to capture a classroom of average dimensions of 8\*6 meters.

* 3.2: Software Interfaces:

The software is packaged using ElectronJS where it can run on different operating systems with no need to download and install certain packages or libraries on the system. The frontend of the software that is developed using ElectronJs communicates with the backend of the system using HTTP requests that follow a client-server architecture, by invoking APIs that are developed using Flask that is hosted on Heroku hosting service, in addition to the database that is hosted on AWS EC2 instance and connected to AWS S3 Bucket service to store images of the possible cheating incidents.. The software is divided into 3 main services, which are listed as follows:

1. The Backend: Consisting of 18 APIs that serve the DL model service and the frontend.
2. The Deep Learning Model: Responsible for detecting the possible cheating incidents, which is trained by our generated dataset consisting of 10,000+ frames.
3. The Frontend: The user interface of the system that communicates with the backend APIs to send and retrieve data.

The abovementioned three components of the system communicate with each other simultaneously using HTTP requests to do the following:

* Allow the user to login into the system
* Initiate the DL Model.
* Retrieve data from the backend to view it on the frontend
* Send user data from the frontend to the backend of the system
* Communicate between the backend and the DL Model.
* 3.3: User Interfaces:

The front end of the system consists of 7 interfaces, in addition to the admin panel. The interfaces include the following:

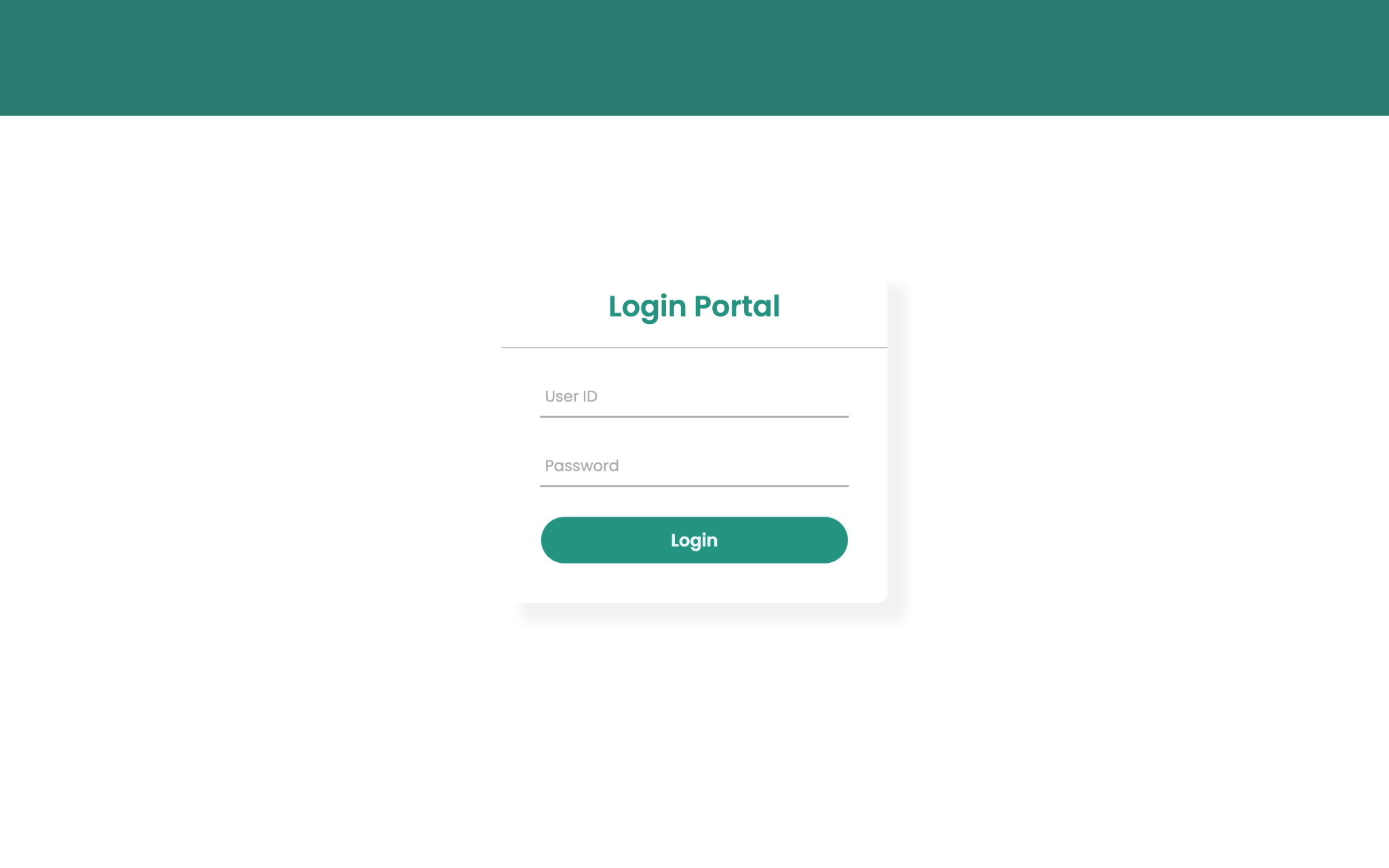
1. Software Landing Page
2. Proctor Login Page
3. Exam Session Details Confirmation Page
4. AI Model Sensitivty Page
5. Home Page
6. End of Exam Report Page
7. Log out Page

The following are screenshots illustrating the software interfaces:

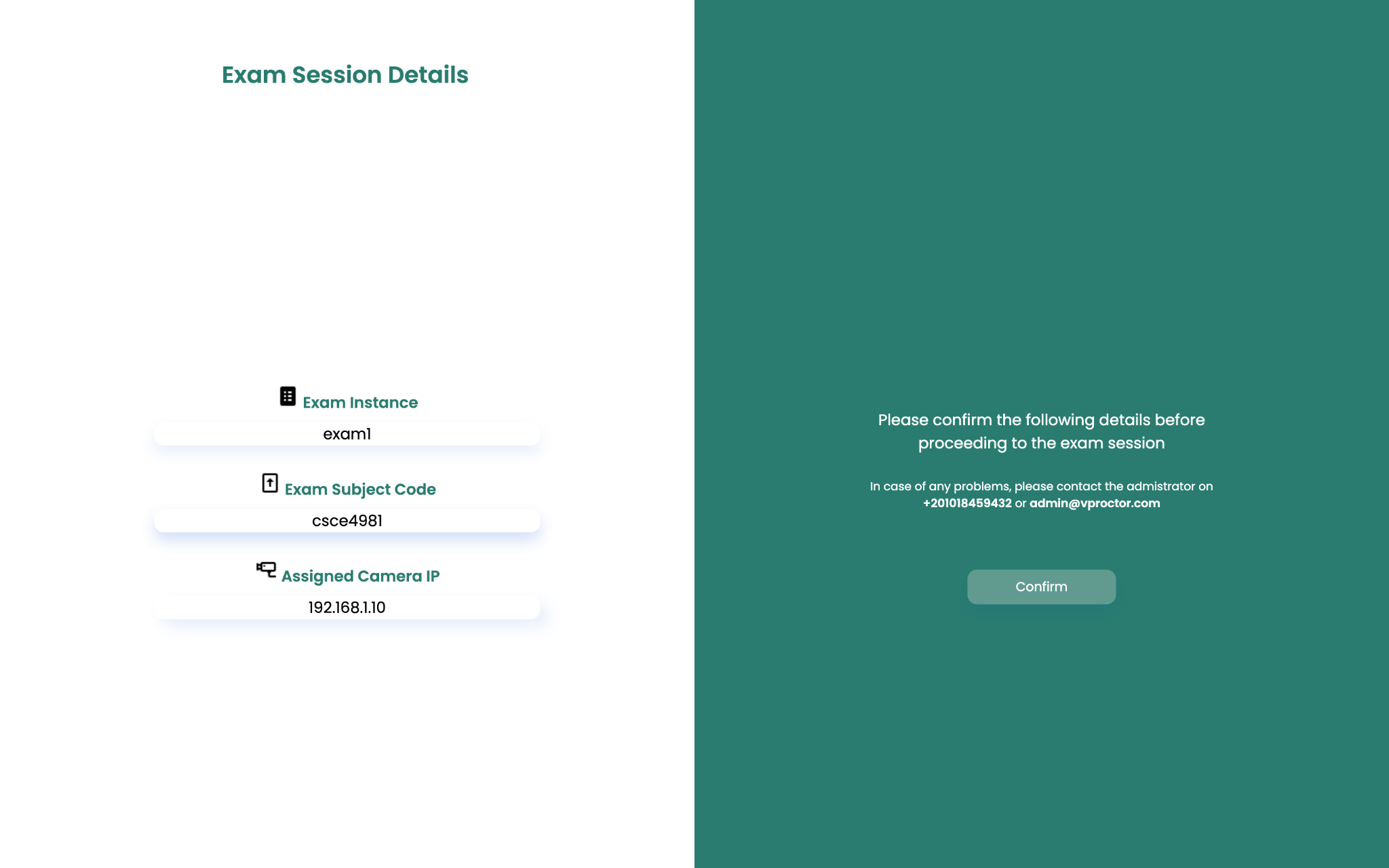
* Software Landing Page



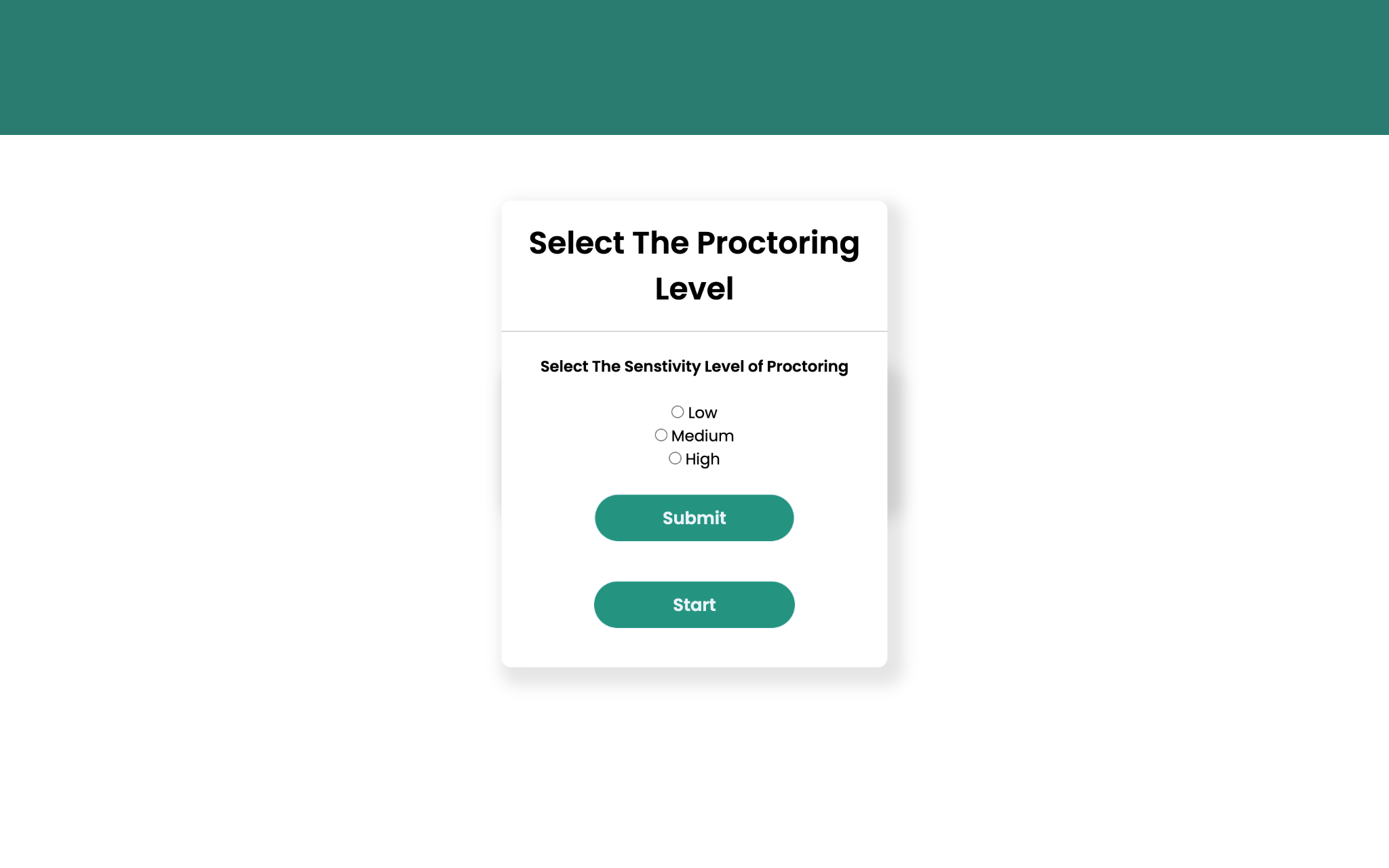
* Proctor Login Page



* Exam Session Details Confirmation Page

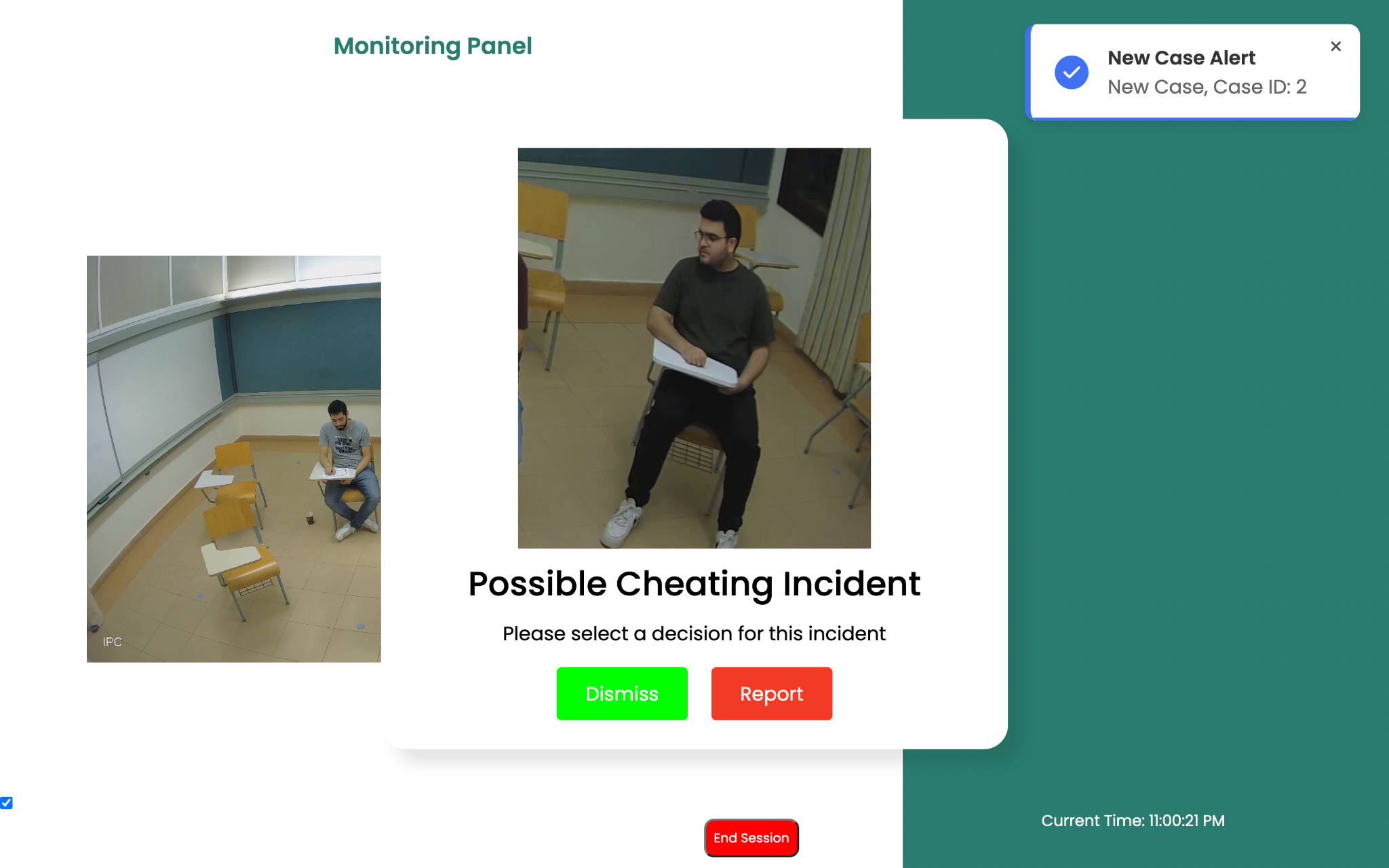


* AI Model Sensitivty Page

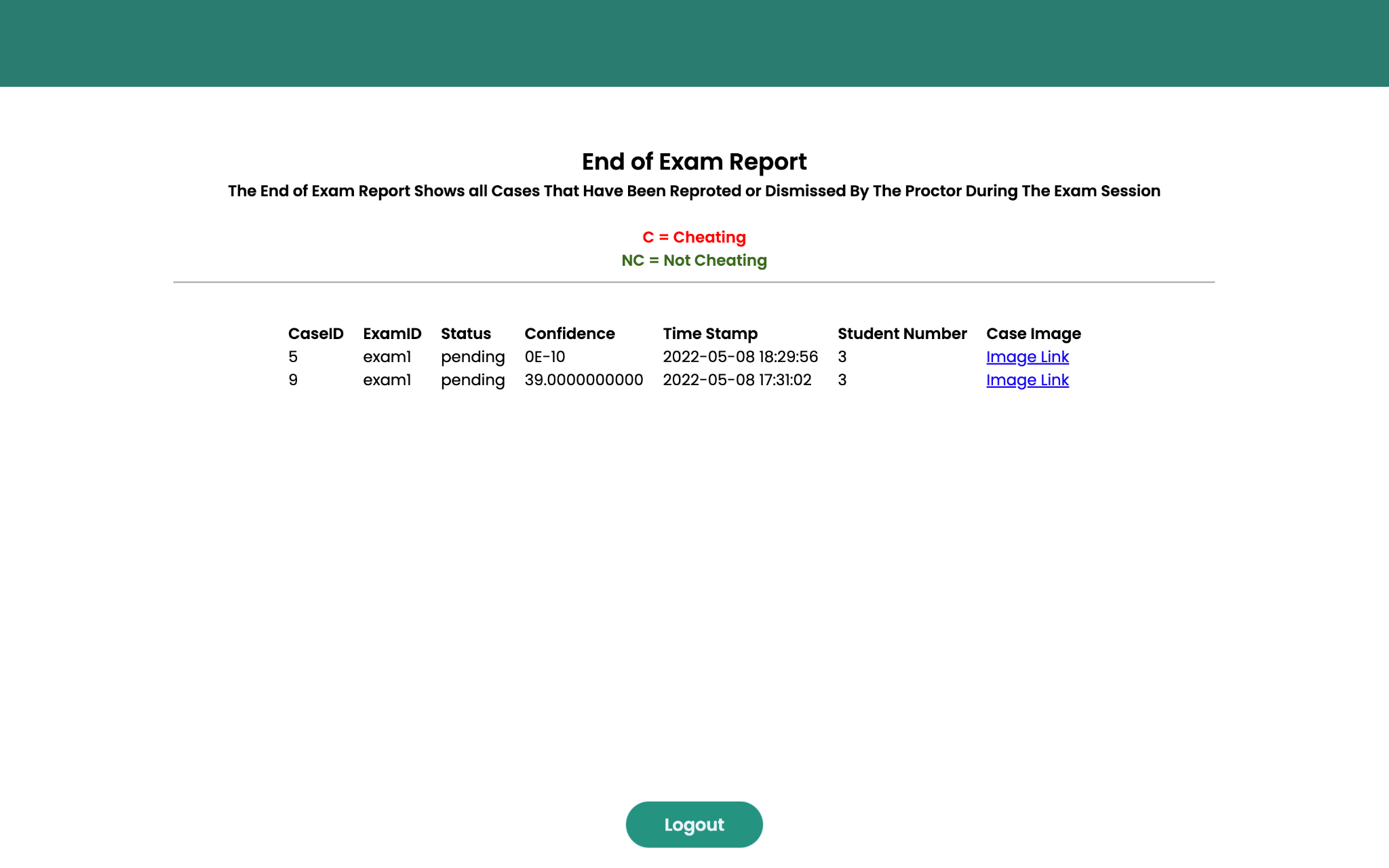


* Home Page





* End of Exam Report Page



* 3.4: Hardware Interfaces:

The recommended hardware specifications to operate the software are 8 Gigabytes of Random Access Memory (RAM) and an Intel Quad-core i7 processor, to be able to run the software with no complications. An IP/Digital camera can be used to send the video feed to the software to be compiled by the DL model.

* 3.5: Communication Protocols and Interfaces:

The software requires an internet connection to connect to the remote database of the software, where exam reports and user information are stored. The connection between the security camera in the classroom and the computer is based on Real-Time Streaming Protocol (RTSP). The protocol can be used to transmit the real-time live video to the software for processing and viewing. The HTTP protocol is used to communicate with the backend of the software to retrieve and send data from and to the database using APIs.

* 4: System Features:
* 4.1: Overview of System Features:

1. Viewing Possible Cheating Incidents:

System users (proctors) can view streaming of a classroom setting with bounding boxes around each student in the classroom (green box indicating normal activity). A notification is displayed beside the live stream where all possible cheating incidents are viewed by the proctor with the case number of the student, the user can dismiss or report the incident by viewning the frame of the possible cheating incident.

1. Confirming Exam Session Details:

The exam proctor can review his/her assigned exam details before proceeding to the next steps and pages in the software, to ensure that the proctor logged into and correctly assigned to the correct exam instance.

1. Reporting Cheating Incidents:

System users (proctors) can report a cheating case by viewing the details of a notification displayed on the notification panel on the software homepage. The user can view a snapshot of the suspected student while doing the suspicious cheating activity.

1. Dismissing Cheating Incidents:

System users (proctors) can dismiss a cheating case by viewing the details of a notification displayed on the notification panel on the software homepage. The user can view a snapshot of the suspected student while doing the suspicious cheating activity.

1. Viewing The Generated Exam Report:

System users (proctors) can view a full examination report after the end of the exam. The report contains a summary of all the dismissed and reported cheating incidents during the exam with a timestamp, model confidence, student number, and case ID for each incident.

* 5: Non-functional Requirements:
* 5.1 Performance Requirements:

The minimum hardware specifications to operate the software is 8 Gigabytes of Random Access Memory (RAM) and an Intel Quad-core i7 processor, to be able to run the software with no complications.

* 5.2 Safety Requirements:

To ensure the safety of all user information, user credentials are safely stored on the software’s database which offers network encryption to keep the communication between the client and the database as safest as possible, using TLS and SSL security certificates.

* 5.3 Security Requirements:

To ensure optimum security during the usage of the software. Users are required to enter their credentials [username and password] to allow the users to access the user privileges of the proctor.

* 5.4 Software Quality Attributes:

* Availability:

The system will be run frequently according to the exam time may take about 1-3 hours. In case of any system crash, the system should recover within minutes and all the data would be stored and saved frequently on the cloud.

* Reliability:

The system will ask the user to enter his/her login credentials to enter only allowed users to the system. In case of wrong login credentials, the system will prompt the user of the wrong login attempt. All generated data by the system will be stored in hard drives and cloud storage to ensure the security and accessibly of data.

* Security:

The system will use an online database that encrypts users’ data to ensure the security and confidentiality of the data. HTTPS will be also used to ensure the security of the connection between the client and server.

* Maintainability:

In case of a failure, any system module should be able to recover in a duration of 10-15 minutes to minimize the absence of the functionality of the system during the examination process

* Testability:

Testing will be done as a continuous process during the development of the system, including pre-testing stages, which ensures that system requirements are properly met. The following phase will be the system testing phase which will include testing the system after development. Several testing methodologies will be followed, such as, unit testing, functional testing, white box testing for the developed code, and the black box testing of the software in order to ensure the complete functionality of the system.