Software Design Document

for

Ai Invigilation System

Version 1.0

Prepared by Zicheng Guo, XXXX XX, XXXXXXX XXXX

Department of Computing and Software, McMaster University

Supervisor: Dr. Rong Zheng

Table of Contents

1. Introduction

1.1 Purpose

This software design document describes the architecture and system design of the AI Invigilation system that helps the supervisors of in-person exams by setting up exam timers, detecting and recording suspicious behaviors of examinees during the exams. It also includes the decomposition of the system and design details of decomposed subsystems.

The intended audience for this document is the software developers and testers. They will use this document to know the designs and structures of the system. This document also matches the system's functionalities with design details and provides developers with a guide on how to implement the system's features. The testers will use it to design corresponding test cases for the system.

1.2 Scope

The AI Invigilation system helps the supervisors of in-person exams by setting up exam timers, detecting and recording suspicious behaviors of examinees during the exams. For some large-scale examinations, it's hard for the supervisors to pay attention to each and every examinee. At the same time, detecting cheating behaviors usually requires a period of time to observe that specific suspicious examinee, which makes it harder to fairly supervise all examinees. The AI Invigilation system can monitor several examinees at the same time and once their behaviors meet some pre-set criterion, the corresponding examinees will be highlighted with explanations for the supervisors to make further decisions. Besides, the criteria and factors are customizable for different examinations by the supervisors. The system will store the videos on local machines to reduce disputes in the future.

1.3 Overview

This document describes how the system is decomposed into several subsystems, the design of each subsystem and how they will work together toward the features of the system. For each subsystem, the functionality and design are also included in this document with corresponding diagrams and justifications. Besides, the document also suggests how the data is stored and manipulated within the system. The user interface section describes how a user will interact with the system, especially the WI component with graphs simulating the system UI.

●

1.4 Reference Material

This section is optional.

List any documents, if any, which were used as sources of information for the test plan.

1.5 Definitions and Acronyms

MM: The main machine is a PC with multiple monitors running windows, and it will be used as the primary monitor for supervisors.

BE: The back-end server running the AI models analyzing captured camera videos.

WI: The web interface that is used to access the system.

Supervisor: Leading invigilator of the examination.

2. System Overview

The system's major functionalities are

- **Supervisors authentication**: To ensure only the technical staff have the access to the system, the system shall provide two different auto-generated hash codes to the leading supervisors and other invigilators through external communication app whenever an exam starts. The code only supports limited-time window access (30 minutes prior and after).
- **Suspicious behavior detection**: Once the supervisor starts the exam, the system will detect suspicious behavior of the examinee and flag an examinee if the pre-set criteria are met.
- **Exam management**: The system shall allow the leading supervisors to input exam information, set up timers, change the rules and add any specifications about the exam. The system shall also allow supervisors to customize pre-set triggers that the system will detect suspicious behaviors depending on.
- **Generating report**: The system shall generate a report including exam information and timestamps of operations (detected suspicious behaviors, cancelled highlighted behaviors, etc.) when the exam is ended. The system will also store the recording on the local machine for future use.
- **Streaming camera feeds**: The system shall display a GUI that allows supervisors to control (start and end) the stream of camera feeds to the BE and MM.

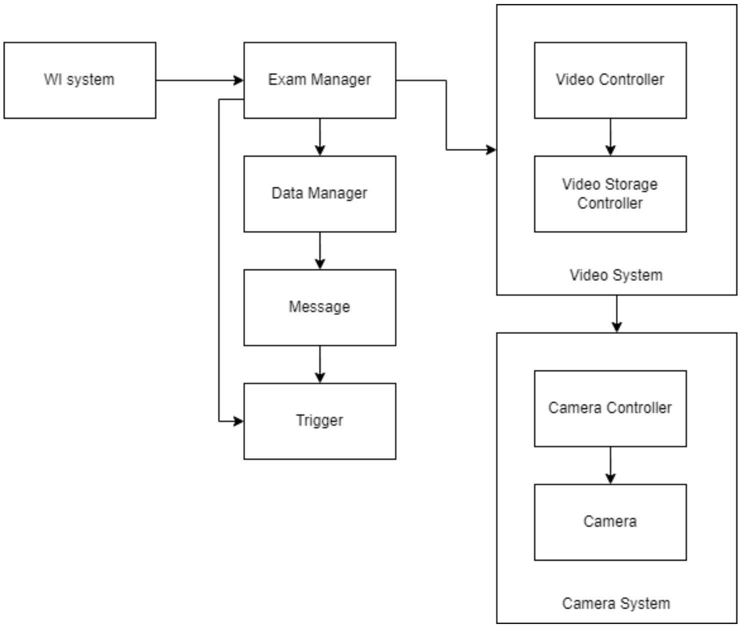
The system's features will work under the following environment:

- **BE**: A back-end server running Linux with enough computability for the AI model. The back-end server can be a bare-metal server or a cloud service. It shall have enough GPU power to execute model inference. It can be in the same room as the examinees or at any other location with stable internet bandwidth.
- **MM**: A PC with multiple monitors running windows, with enough I/O ports for connecting cameras. The main machine is required to have at least one monitor, but recommended one monitor for invigilator use while another screen(projector, large screen) is to display exam-relevant information for examinees. It should have enough I/O ports to connect cameras.
- **WI**: A modern PC with a browser to access the web interface. The system will operate in the web browser, and it mainly supports services on Google chrome. There is no particular requirement for a WI interface, a monitor or a screen of any kind, some put method (mouse, keyboard or touch screen) is enough. A screen size greater than 13 inches is recommended.

3. System Architecture

3.1 Architectural Design

The major subsystems of the MM and their relationships can be represented in the following diagram:



WI System

This subsystem is responsible for user authentication and exam initialization. It will be the first component that users come into contact with, mainly supporting the supervisors' authentication function. The wi system shall generate a hashcode for different types of users (with different levels of permissions) and authenticate users. Once authenticated as a supervisor, a user will be able to create an exam, which creates an exam manager object and collaborate with the exam manager subsystem.

Exam Manager

This system is the core of the entire MM component, and it controls the other models except for the WI system by sending and receiving data. The exam manager is responsible for setting up an exam (including exam info and triggers), initializing an exam given user input and ending the exam. At the same time, this module is connected to the video controller model (receiving camera feeds and managing local storage of the exam) and the data manager model (managing events and students in the exam). As for the functionality, it mainly supports the exam management feature of the product.

Video System

This system includes a video controller and a video storage controller, it works with the camera system to support the suspicious detection functionality during the exam.

The video controller contains a video storage controller and deals with the start/end recording instructions. Once it receives a start/end recording instruction, it will set the video storage controller to follow the instruction. At the same time, it is responsible for receiving camera feeds and frames from the camera system, then saving corresponding frames in the local storage given a path. Therefore, it collaborates with the camera system by receiving frames and sending commands (for example: get the next frame).

The video storage controller will follow the instructions sent by the video controller and store the series of frames of the current exam on a given path.

Camera System

This system includes a camera controller and several camera objects (the number of camera objects in the camera controller depends on the number of hardware cameras), it works with the video system to support the suspicious detection functionality during the exam.

The camera object is responsible for obtaining feeds and frames from the hardware, and the camera controller contains an array of cameras. Once it receives instructions from the video system to get a frame, the camera controller will get frames from corresponding cameras and send the frames to the video controller and BE, then the BE model will analyze the frames and the video controller will store the frames that are relevant.

Data Manager

This subsystem is responsible for storing all information in an exam (each exam is represented by an exam manager object, and it contains a data manager object) including camera feeds, present students and potential suspicious behaviors (represented by message objects). At the same time, the data in the subsystem will be displayed on the WI for the supervisors as a reference and supports the streaming camera feeds feature.

Message

This subsystem represents potential suspicious behaviors, it acts as the most important data structure of the suspicious behavior detection functionality. For each detected suspicious behavior, the involved student, timestamp and active triggers (represented as trigger objects) will be wrapped and stored as a message. Meanwhile, a supervisor is able to dismiss a message if he thinks the student is not cheating.

Trigger

This subsystem represents the criteria of suspicious behaviors, it can be set by the exam manager and also works as a parameter for a suspicious behavior represented by a message object. Thus, the system collaborates with the exam manager and the message subsystems. It’s worth mentioning that it is a hierarchy data type including different types of triggers (for example, head position trigger and gaze trigger). For further development, it allows users to add custom triggers under the high-level trigger system.

The major subsystems of the BE are:

Communication Manager

This subsystem is responsible for communication between MM and BE, which will decode messages and send them to the necessary module for processing. And responsible for returning the result of each request MM has made.

Trigger Manager

This subsystem will have all same triggers as ExamManger has but in boolean value(i.e. if a trigger is activate, it will be true), which those triggers can be accessed by the algorithm control for detection.

Algorithm Controller

This subsystem is responsible for detection, when inputting a frame, it will extract trigger information from the trigger manager and send the frame to a designate model to make a prediction, and based on the result and trigger setting, returning the final result.

Model

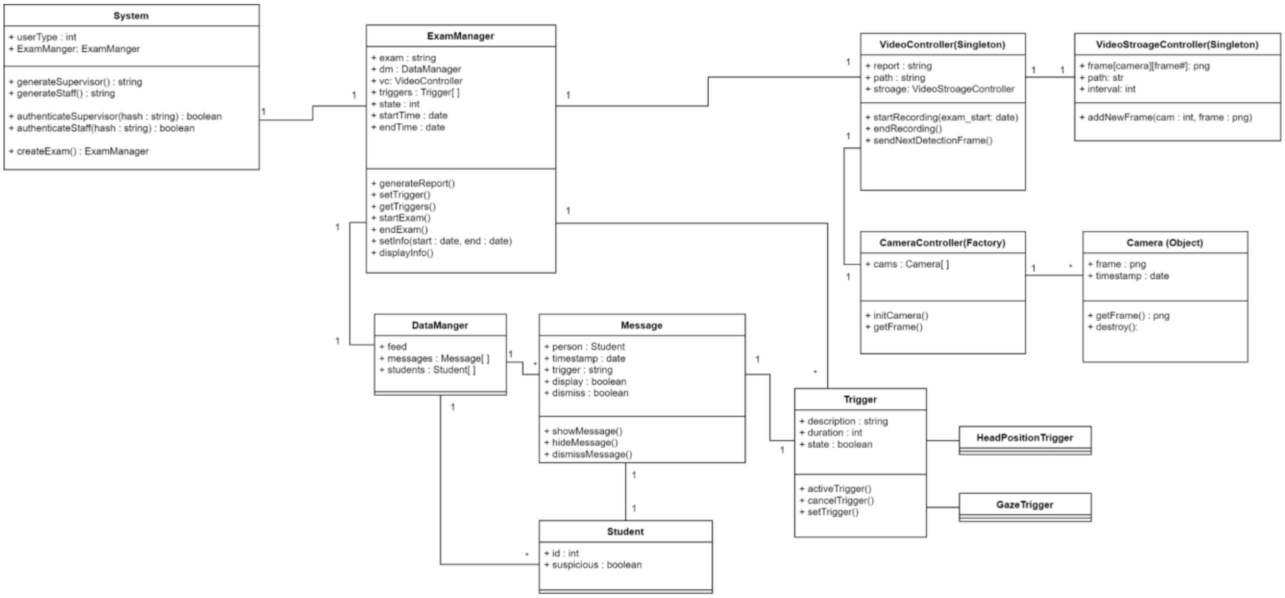
This subsystem represents each AI model we use, it can be called when the specific trigger is set and it will then process the frame and return the result for the algorithm controller to analyze.

Storage Manager

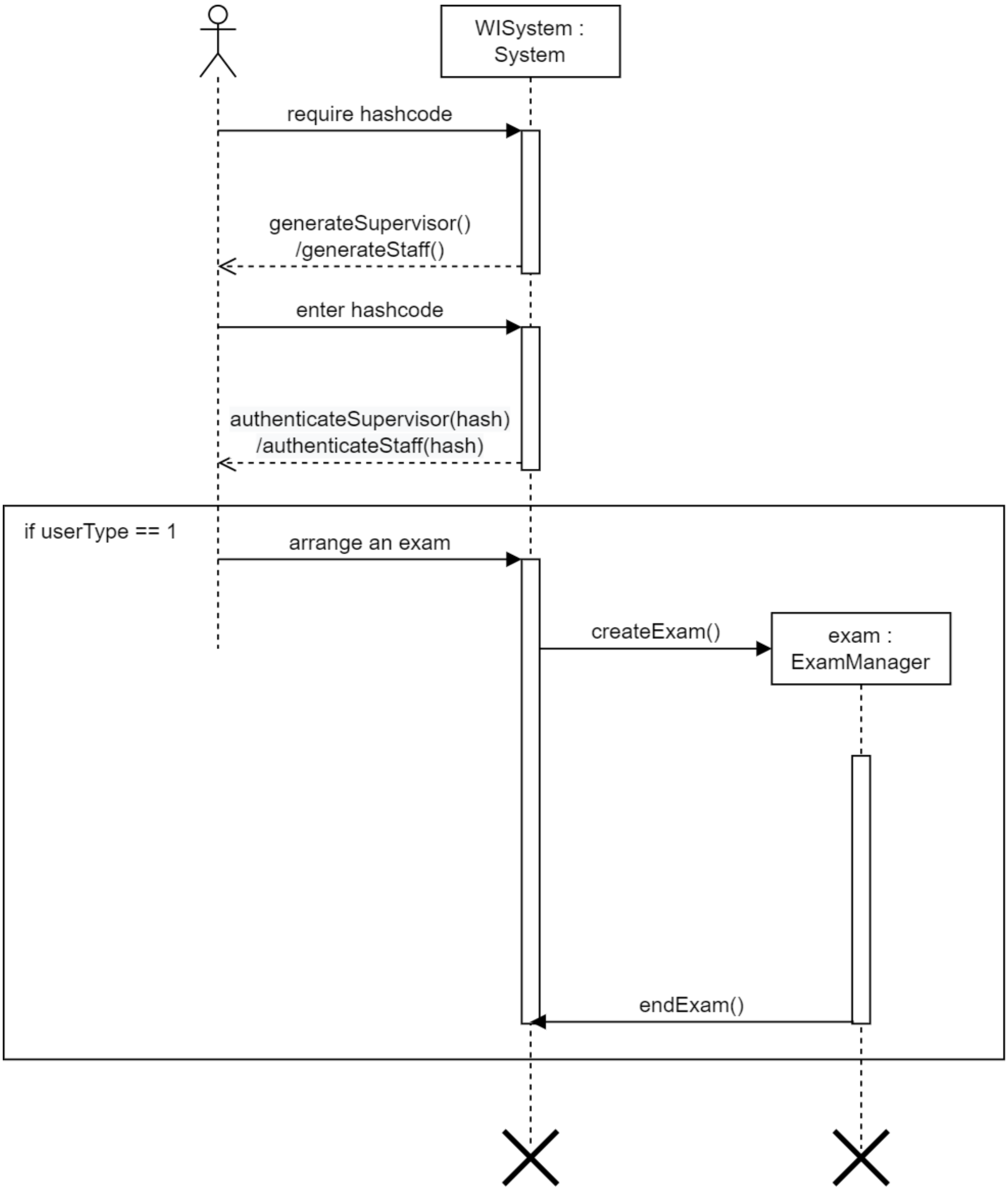
This subsystem is responsible for making a backup for every frame that a suspicious behavior is detected. This will ensure that if uncontrollable situation happened on MM (lose of power, hardware damage), we still have one minimal copy of the frame.

3.2 Decomposition Description

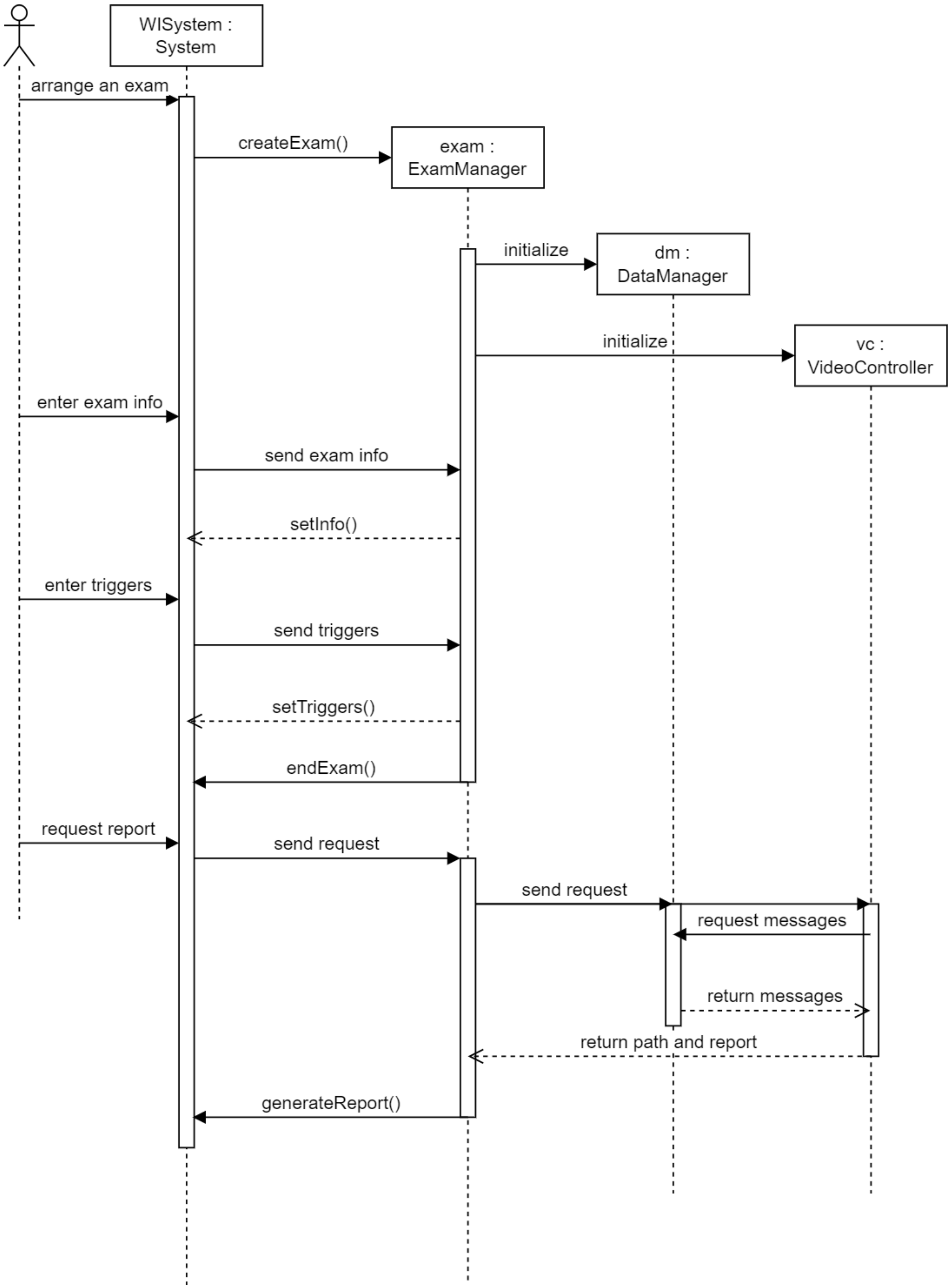
The class diagram of the MM and their relationships can be represented in the following diagram:


System
+ userType : int
+ ExamManger: ExamManger
+ generateSupervisor() : string
+ generateStaff() : string
+ authenticateSupervisor(hash : string) : boolean
+ authenticateStaff(hash : string) : boolean
+ createExam() : ExamManager
1
1
ExamManager
+ exam : string
+ dm : DataManager
+ vc: VideoController
+ triggers : Trigger[ ]
+ state : int
+ startTime : date
+ endTime : date
+ generateReport()
+ setTrigger()
+ getTriggers()
+ startExam()
+ endExam()
+ setInfo(start : date, end : date)
+ displayInfo()
1
1
1
1
1
VideoController(Singleton)
+ report : string
+ path : string
+ stroage: VideoStroageController
+ startRecording(exam_start: date)
+ endRecording()
+ sendNextDetectionFrame()
1
1
1
VideoStroageController(Singleton)
+ frame[camera][frame#]: png
+ path: str
+ interval: int
+ addNewFrame(cam : int, frame : png)
1
CameraController(Factory)
+ cams : Camera[ ]
+ initCamera()
+ getFrame()
1
*
Camera (Object)
+ frame : png
+ timestamp : date
+ getFrame() : png
+ destroy():
1
DataManger
+ feed
+ messages : Message[ ]
+ students : Student[ ]
1
*
1
Message
+ person : Student
+ timestamp : date
+ trigger : string
+ display : boolean
+ dismiss : boolean
+ showMessage()
+ hideMessage()
+ dismissMessage()
1
*
1
Trigger
+ description : string
+ duration : int
+ state : boolean
+ activeTrigger()
+ cancelTrigger()
+ setTrigger()
1
HeadPositionTrigger
GazeTrigger
1
1
*
Student
+ id : int
+ suspicious : boolean

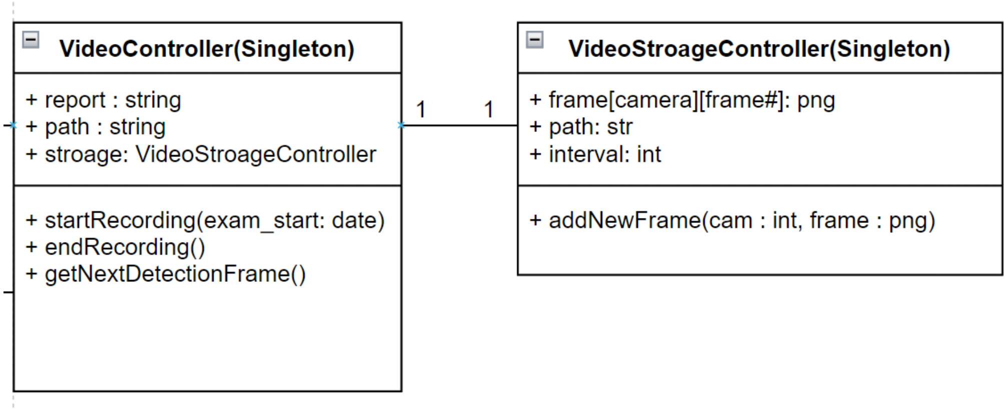
WI System

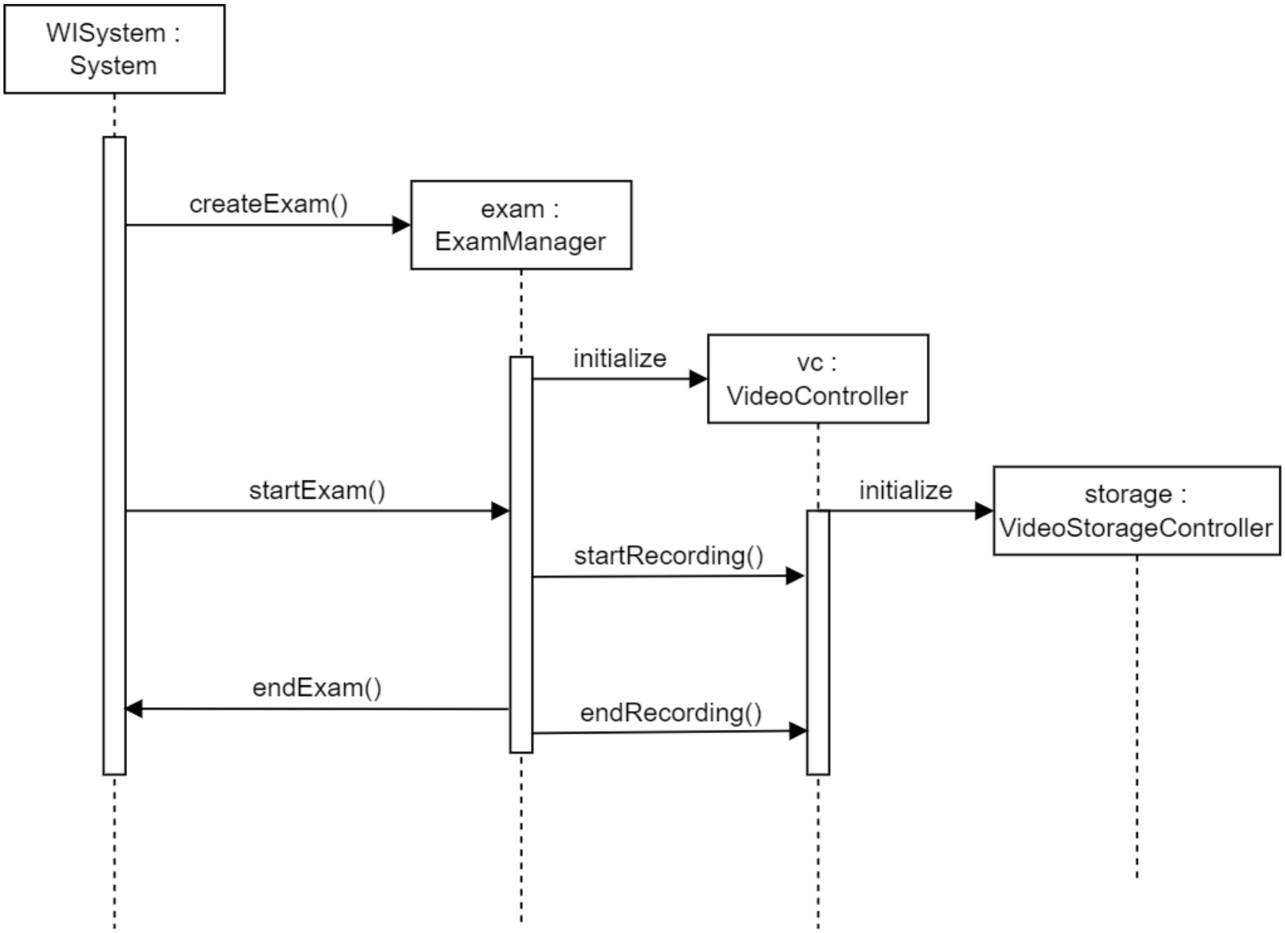


Exam Manager

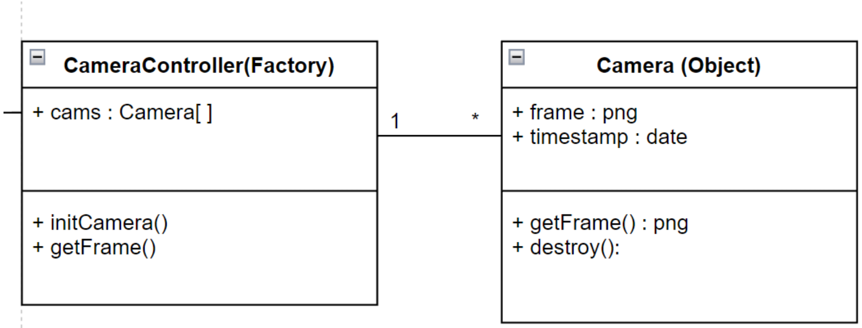

WISystem : System
exam : ExamManager
dm : DataManager
vc : VideoController
arrange an exam
createExam()
initialize
initialize
enter exam info
send exam info
setInfo()
enter triggers
send triggers
setTriggers()
endExam()
request report
send request
send request
request messages
return messages
return path and report
generateReport()

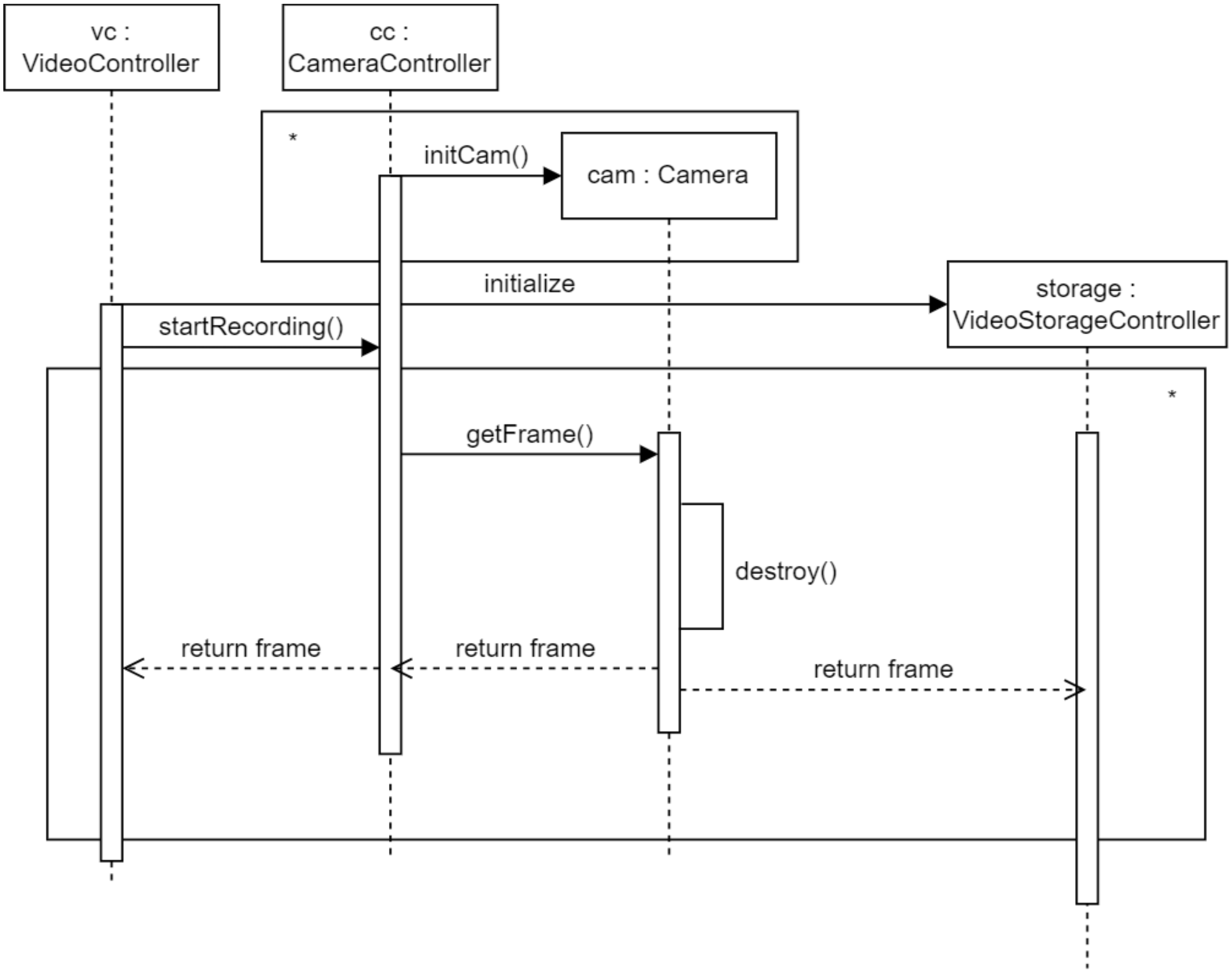
Video System


VideoController(Singleton)
+ report : string
+ path : string
+ stroage: VideoStroageController
+ startRecording(exam_start: date)
+ endRecording()
+ getNextDetectionFrame()
1
1
VideoStroageController(Singleton)
+ frame[camera][frame#]: png
+ path: str
+ interval: int
+ addNewFrame(cam : int, frame : png)

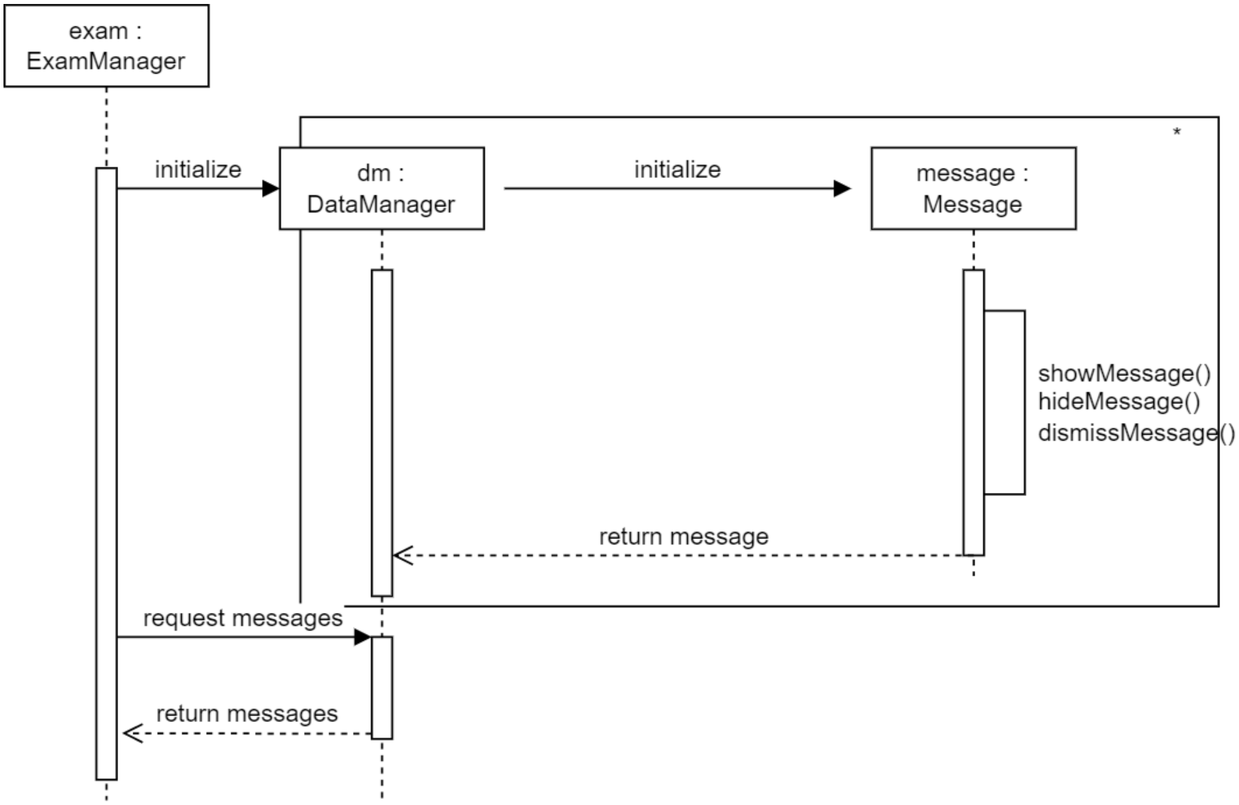


Camera System

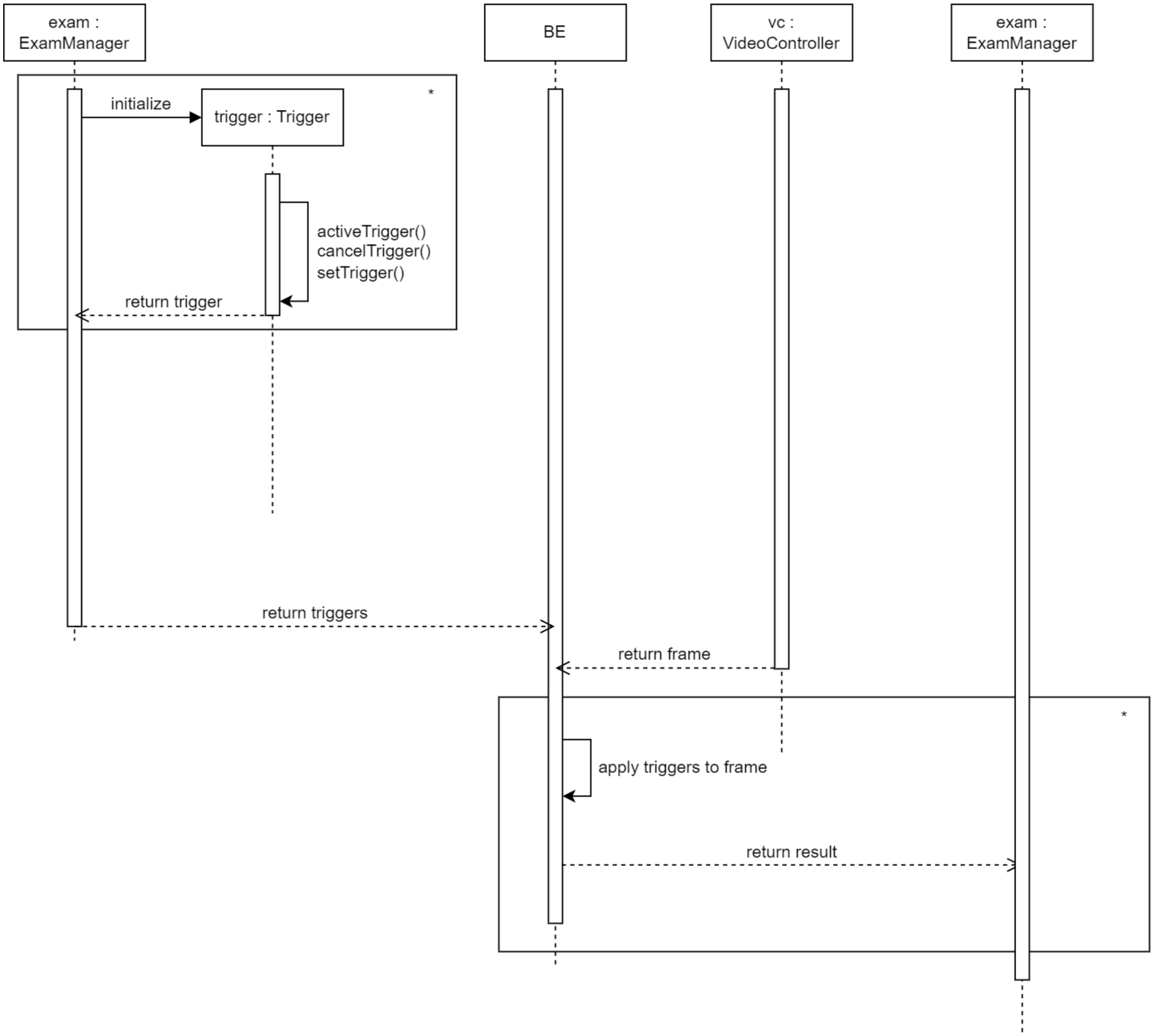




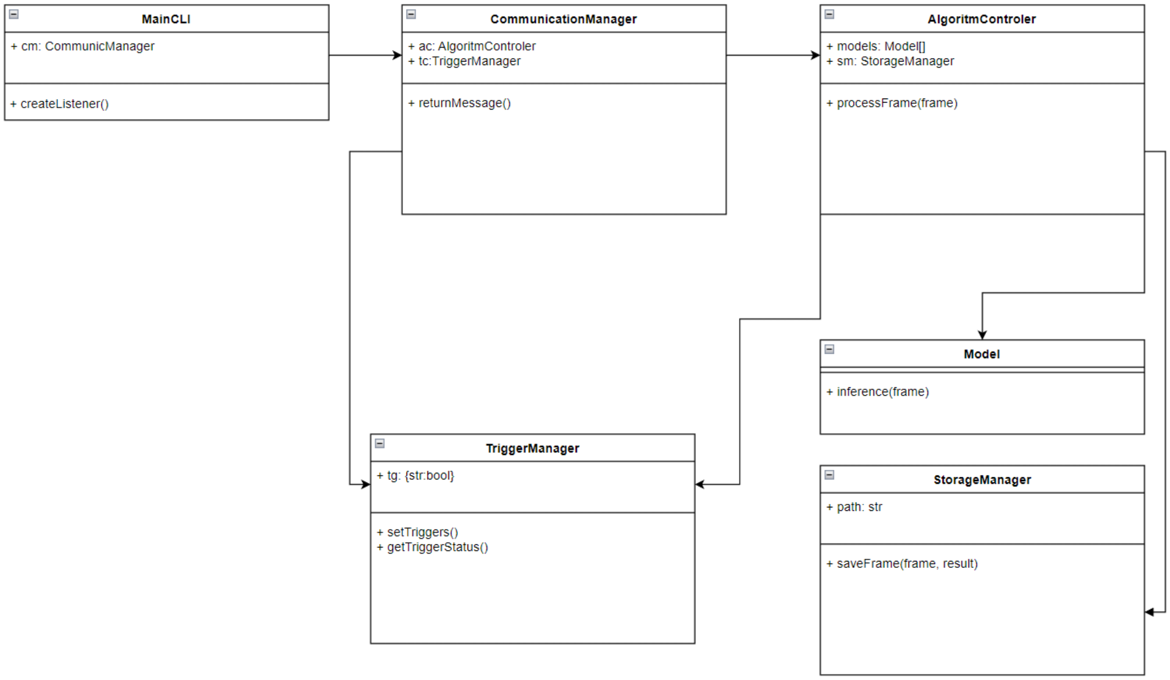
Data Manager, Message

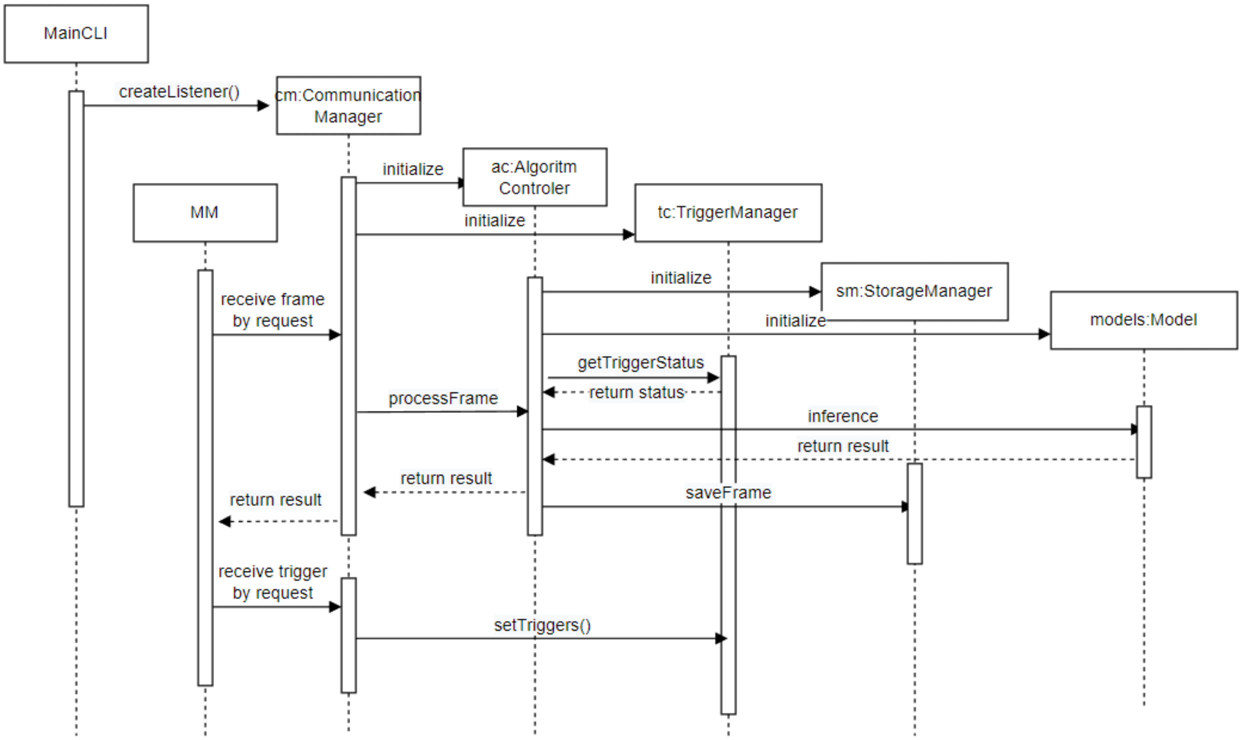

exam : ExamManager
initialize
dm : DataManager
initialize
message : Message
*
showMessage()
hideMessage()
dismissMessage()
return message
request messages
return messages

Trigger



The class diagram of the BE and their relationships can be represented in the following diagram:


MainCLI
+ cm: CommunicManager
+ createListener()
CommunicationManager
+ ac: AlgoritmControler
+ tc:TriggerManager
+ returnMessage()
AlgoritmControler
+ models: Model[]
+ sm: StorageManager
+ processFrame(frame)
Model
+ inference(frame)
TriggerManager
+ tg: {str:bool}
+ setTriggers()
+ getTriggerStatus()
StorageManager
+ path: str
+ saveFrame(frame, result)


MainCLI
createListener()
cm:Communication Manager
initialize
ac:Algoritm Controler
MM
initialize
tc:TriggerManager
initialize
sm:StorageManager
initialize
models:Model
receive frame by request
getTriggerStatus
return status
processFrame
inference
return result
return result
saveFrame
return result
receive trigger by request
setTriggers()

3.3 Design Rationale

The high-level system decomposition follows the SRS document, it divides the system into components including WI, MM and BE. Each component has different responsibilities, they operate independently for specific functionality and collaborate to constitute the entire system. For the MM and BE, they are decomposed into smaller subsystems since they are both complicated structures and we want the separation of concerns.

Considering the triggers subsystem in the MM component, we decided to use a hierarchy design for this data structure. This is because there is much research and evidence about the possible signs or triggers for suspicious behaviors, it's more flexible and sustainable to allow adding new types of triggers based on the Trigger class. In the current product, we involve two classes of triggers (HeadPositionTrigger and GazeTrigger), but we would like to design for change.

For the exam manager and data manager subsystems in the MM component, we designed it as a whole (an exam class) at first, but we realized that the combined data structure contains too many unrelated fields and methods. Therefore, we divided it into exam manager and data manager to control the size of a subsystem and make sure each module only does a limited amount of things instead of many things. This decision also follows the low coupling and high cohesion design principles.

Another potential design for the communication between BE and MM is making BE send results (the timestamp and relevant student) to MM directly once a suspicious behavior is detected. However, we realized that it would be more efficient and convenient if we can wrap these results and information into a data structure. Then, we design a data structure called message with fields person, timestamp and trigger. Besides, the supervisor can dismiss a message if the student is misjudged to have suspicious behavior. The design of the message class can wrap various data into a formalized data structure which can be easily stored in an array (the messages field in the data manager). At the same time, the massages array can be accessed directly by the exam manager to generate the report.

4. Data Design

4.1 Data Description

Our system has no database related since our system is for school exam use or any other formal educational exams, any leakage of recordings could be a huge problem for any examinees, in order to avoid this kind of situation happens, we simply don't have any database or functionalities to check on the recordings on our end.

4.2 Data Dictionary

Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

All functions in section 3.2

Group	Functions	Description	Input	Return Value	Input Type
Require HashCode	generateSupervisor()	generate hashcode for supervisors	none	output a integer type hashcode	
	generateStaff()	generate hashcode for staff	none	output a integer type hashcode	
Enter Hashcode Phase	authenticateSupervisor(hash)	authenticate hashcode for supervisors	hashcode	*open the main page window if successfully authenticated*	int()
	autheticateStaff(hash)	authenticate hashcode for staff	hashcode	*open the main page window if successfully authenticated*	int()
Supervisor Manage Exam	createExam()	create the exam	none	*pop out a small window asks you to type in the exam name*	
	endExam()	end the exam	none	will end record the exam	
Exam Manager	setInfo()	set exam information to examinee	none	*pop out a small window asks you to type in the information*	

				needed	
	displayInfo()	display exam information	none	display the exam info	
	setTriggers()	pre-set criteria which triggers	none	none	
After exam	generateReport()	generate report	none	output a txt file along with the recording to the local machine	

5. Human Interface Design

5.1 Overview of User Interface

We only have two kinds of users at this system, the technical staff and supervisors. In our WI, technical staff have different interfaces other than the supervisors. We have leading supervisors who are able to alter some of the core information of the exam, which is something the regular supervisors can't do, although they have the same interface on our WI.

- Login Page

Our login page does not require username and password like other products, which is more secure compared to them, we have the auto-generated hash code sent to the supervisors. In our login page, we separate the regular staff from the leading supervisors, you need to click on your role and then type in the credentials. After you click on your role, you get to type in your institution id and you will receive the code in about one minute, the code will get you to the main page. Feedback provided from the login page is either successful or failed, if you typed in correct credentials, you get logged into the main page, if you didn't, you get a message saying wrong credentials.

- Main Page

Our main page design is very intuitive, on the left hand side is our menu list, we consider put a menu button on the left top corner (not shown on the graph), after you click on it, you get a list of options shown, including set triggers, create exam, exam history, setup camera and logout.

Set Triggers

After clicking on the set triggers, leading supervisors get to see several triggers options, first column is id, second column is the trigger type, third column are the parameters, leading supervisors get to enable them at the fourth column, not only that, they also get to edit or delete any of the triggers.

Create Exam

After click on the create exam, leading supervisors will see some blank spaces to be filled up, first come to mind is exam name, they get to type in the information to the blank space, for example, COMPSCI 4ZP6A, then is the start time and end time, you also get to review the triggers settings.

After typed in all the information, leading supervisors can either save the information for future use or display the information right away if they are all clear and correct. You can see two buttons at the bottom of the page.

About the displayinfo button, after you click on it, the monitor will show the image we put in 6.2 to the examinee, including the information that the leading supervisor just typed in and with the time remaining at the end.

Exam History

After clicking on the exam history, our web will list the past exams, including its dates, exam name, and also the clickable button to the certain exam, the button under the recording and report will take you to the path where our BE stored either the recording or report.(should be under the same path)

SetUp Camera

After clicking on the setup camera, you get to see the list of cameras, its hardware name and status, whether the camera is connected or not.

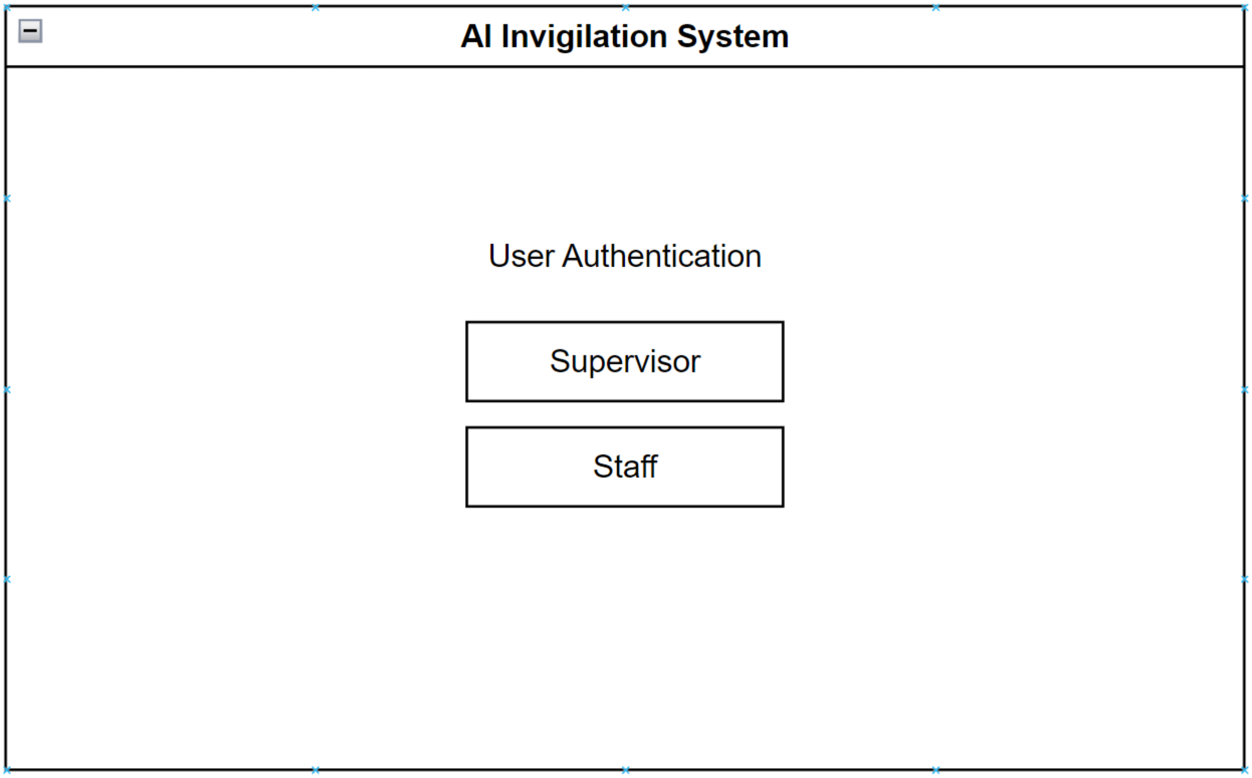
LogOut

After clicking on the logout, you will logout the account.

5.2 Screen Images

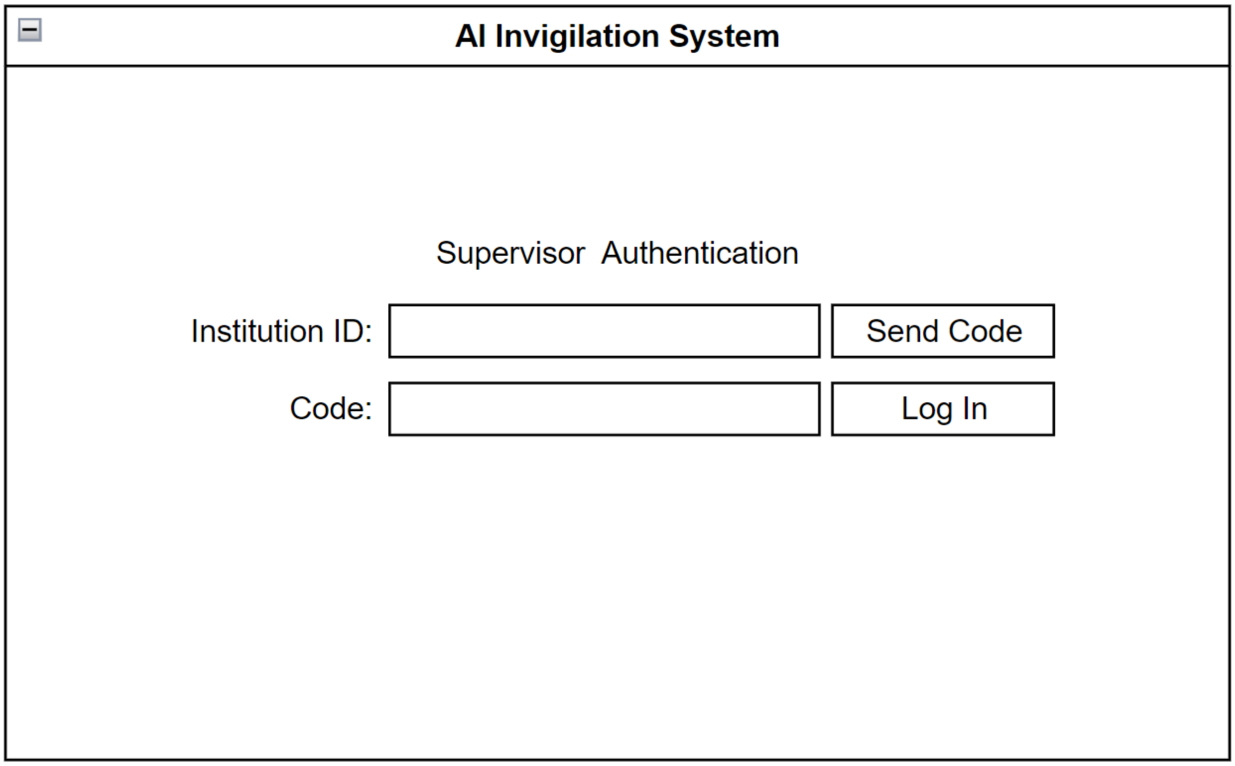
User Authentication Page

- Selecting user type



Our WI interface will ask the user to identify themselves as supervisor or staff, which will lead to different code generation and authentication methods.

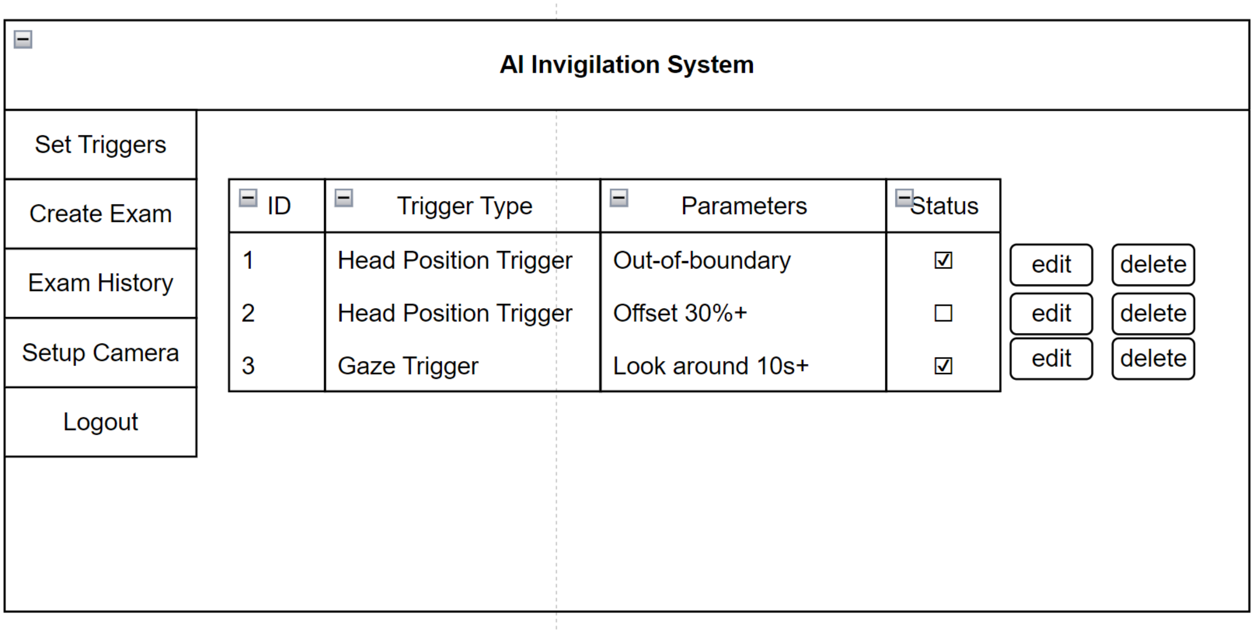
- Generating and authenticating hashcode



In order to get the correct code to log in, you need to type in the institution ID, which will send you the unique code to get you login, leading supervisor will get a different hashcode from the staff.

WI Main Page

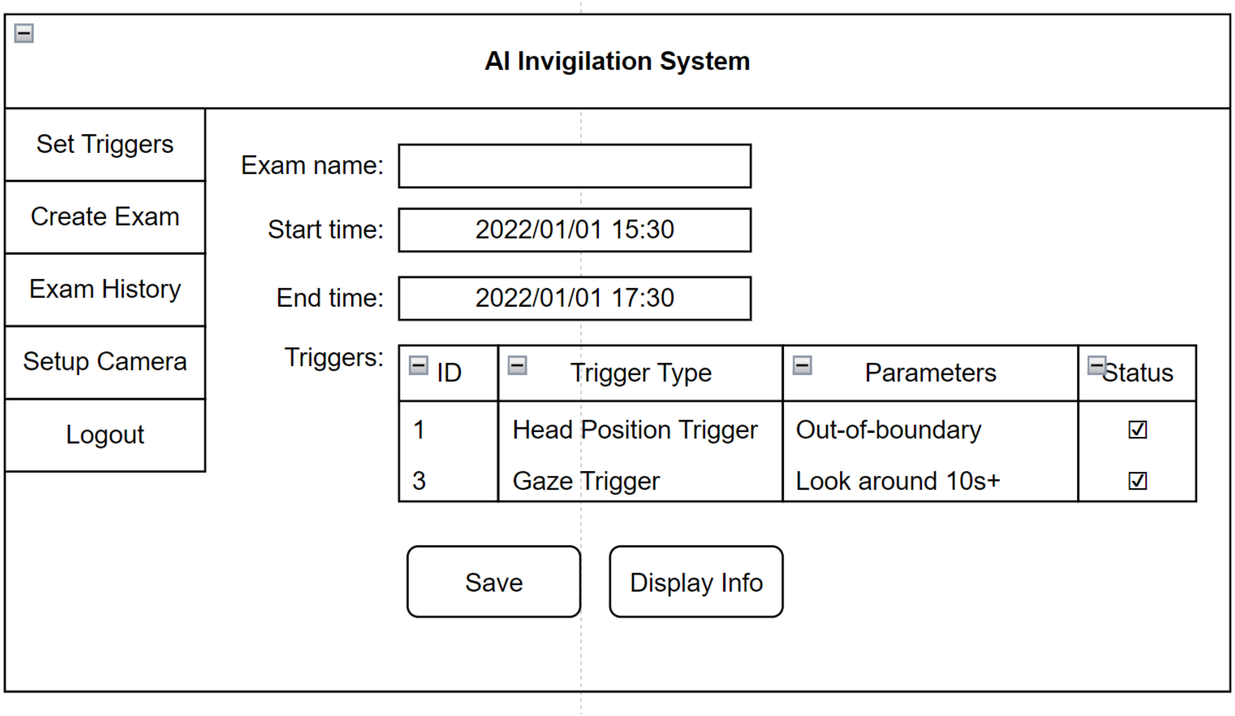
- Setting triggers



ID	Trigger Type	Parameters	Status
1	Head Position Trigger	Out-of-boundary	☑
2	Head Position Trigger	Offset 30%+	□
3	Gaze Trigger	Look around 10s+	☑

As for the feature of cutomizing triggers, as shown the image, supervisors can enable or disable any of the triggers but just simply click on the edit and then click on the status.

- Enter exam information and start the exam



Exam name:

Start time: 2022/01/01 15:30

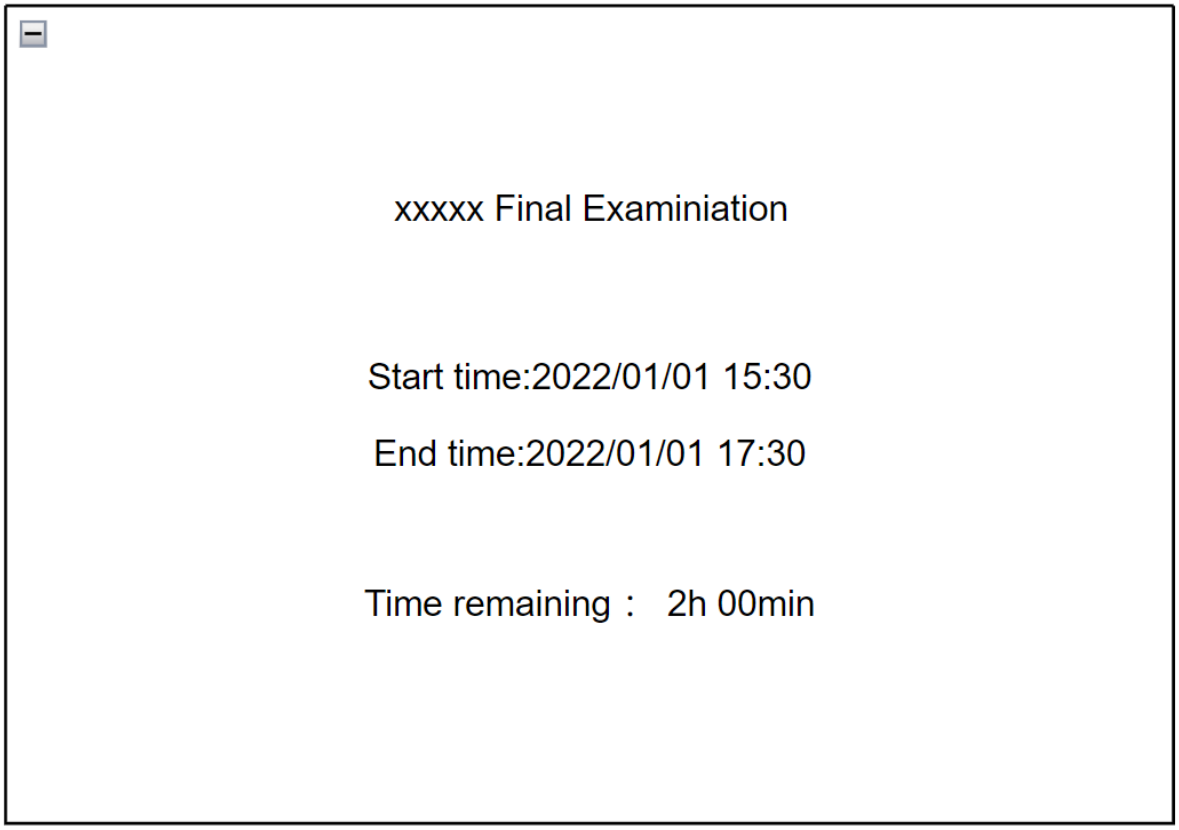
End time: 2022/01/01 17:30

Triggers:

ID	Trigger Type	Parameters	Status
1	Head Position Trigger	Out-of-boundary	☑
3	Gaze Trigger	Look around 10s+	☑

The above image is for create an exam, it will ask the supervisor to input all the information needed above, and it also allows the supervisors to save it or simply display it.

- Displayed exam information window



xxxxx Final Examiniation

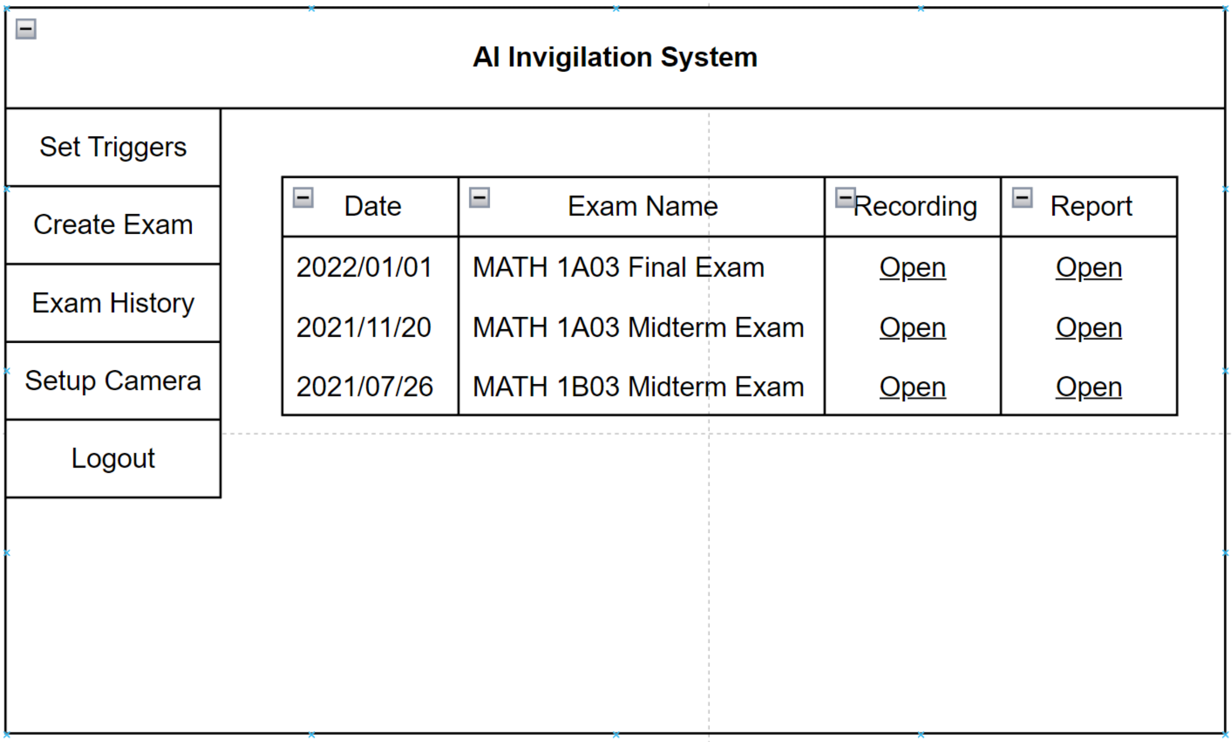
Start time:2022/01/01 15:30

End time:2022/01/01 17:30

Time remaining： 2h 00min

The above image is the demo for displaying the exam info as a separated pop-up window.

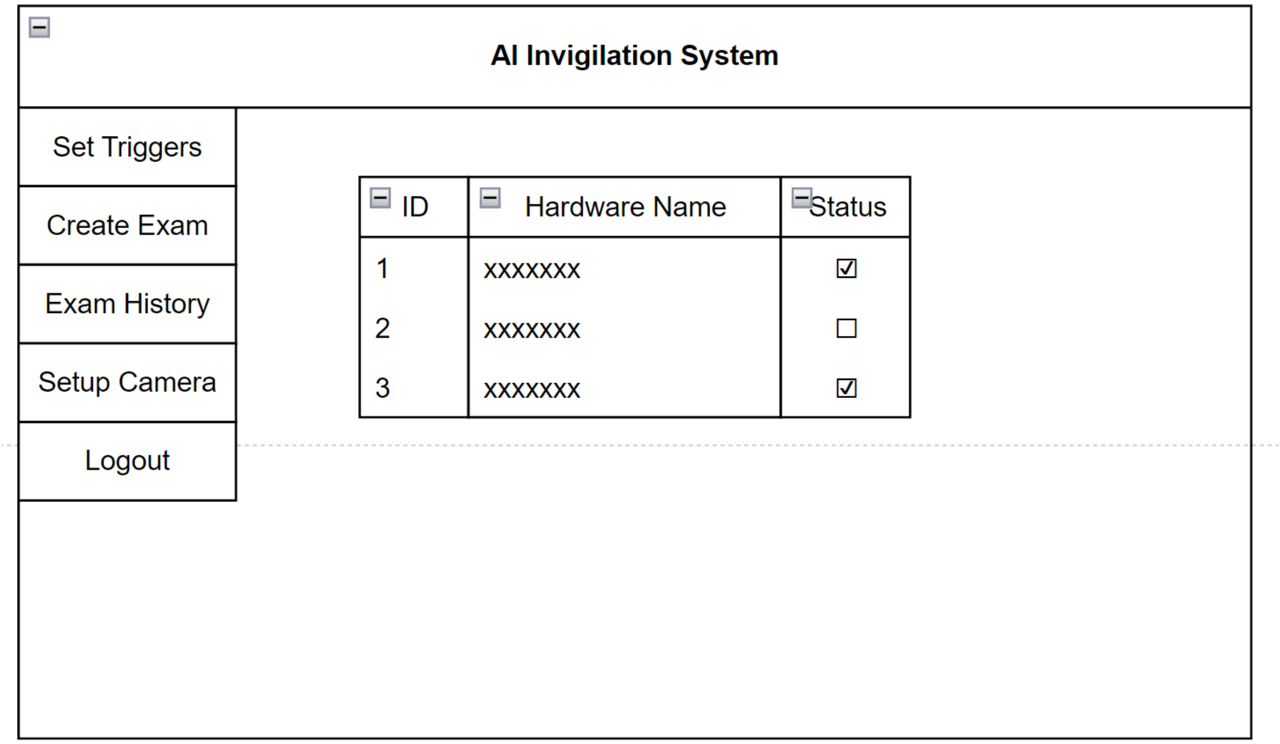
- Browsing history exams and reviewing video recording / report



Date	Exam Name	Recording	Report
2022/01/01	MATH 1A03 Final Exam	Open	Open
2021/11/20	MATH 1A03 Midterm Exam	Open	Open
2021/07/26	MATH 1B03 Midterm Exam	Open	Open

The above image is for exam history, users can click on any of the clickable button, which will redirect staff to the path where the corresponding files are.

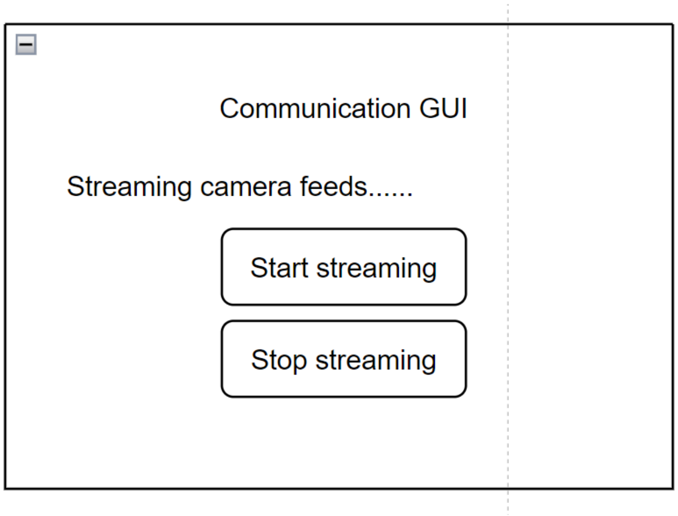
- Setting up cameras



ID	Hardware Name	Status
1	xxxxxxx	☑
2	xxxxxxx	□
3	xxxxxxx	☑

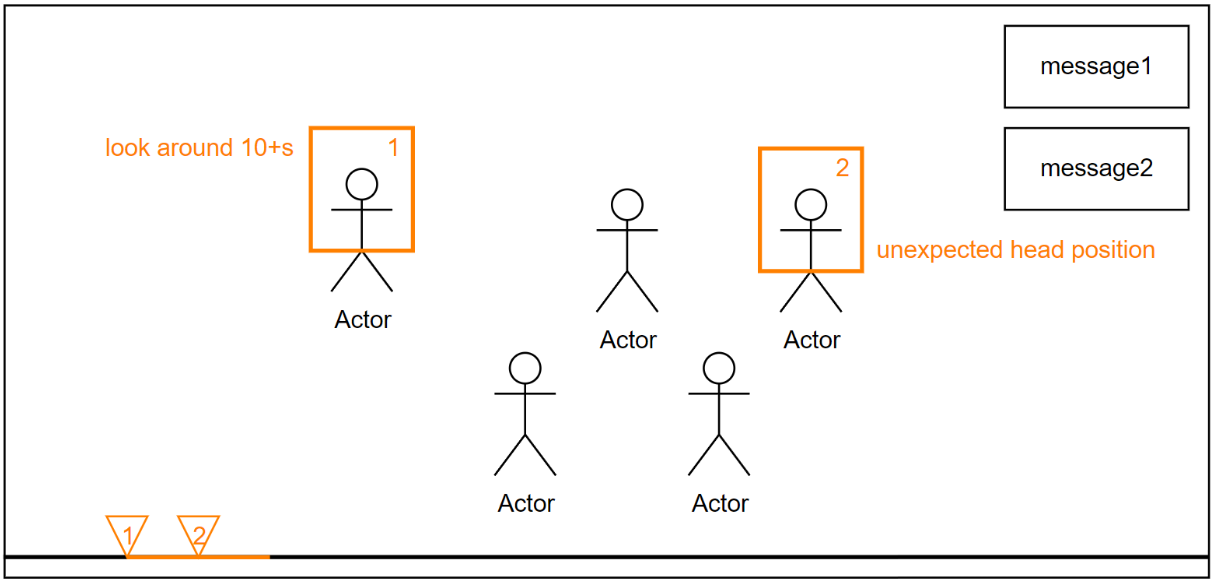
Above image shows what the setup camera features look like, it will display the status of all successfully connected cameras and users can active / disable any cameras.

Communication GUI



Streaming camera feeds......

Exam monitoring page



6. Requirements Matrix

Feature	Requirement ID	System Components	Objects	Methods

Supervisors Authentication	REQ1-1, REQ1-2, REQ1-3, REQ1-4	WI System	System	generateSupervisor()
				generateStaff()

	REQ1-5, REQ1-7	WI System	System	authenticateSupervisor(hash)
	REQ1-6, REQ1-8, REQ1-9	WI System	System	authenticateStaff(hash)
Suspicious Behavior Detection	REQ2-1, REQ2-2, REQ2-3	Data Manager, Exam Manager, Video System, Camera System	System, ExamManager, DataManager, VideoController, CameraController, Camera	startRecording()

	REQ2-4, REQ2-5	Data Manager, Exam Manager, Video System, Camera System, Trigger, Algorithm Controler	DataManager, VideoController, CameraController, Camera, Trigger, Model	getFrame()
	REQ2-6, REQ2-7	Data Manager, Exam Manager, Video System, Trigger, Message	DataManager, VideoController, Trigger, Message	dismissMessage()

	REQ2-8, REQ2-9, REQ2-10	Data Manager, Exam Manager, Video System, Trigger, Message	DataManager, VideoController, Trigger, Message	showMessage()
				hideMessage()

	REQ2-11, REQ2-12, REQ2-13	Data Manager, Exam Manager, Video System, Camera System	System, ExamManager, DataManager, VideoController, VideoStorageController, CameraController, Camera	endRecording()
	REQ2-14	Storage Manager	Storage Manager	
Exam Management	REQ3-1	WI System	System	

	REQ3-2	WI System, Exam Manager	System, ExamManager	createExam()
				setInfo()

	REQ3-3	WI System, Exam Manager	System, ExamManager	displayInfo()
	REQ3-4	WI System, Exam Manager, Trigger	System, ExamManager, Trigger	setTrigger()
	REQ3-5	WI System, Exam Manager, Trigge, Communication Manager, Trigger Manger, Algorithm Controler, TriggerManager	Trigger, Model	

Generating Report	REQ4-1, REQ4-2	Exam Manager, Video System,	ExamManager, VideoController, VideoStroageController	generateReport()

Streaming Camera Feeds	REQ5-1	WI System, Exam Manager	System, ExamManager	startExam()
				endExam()

	REQ5-2, REQ5-3	Communication Manager, ExamManager,VideoController		

7. Appendices

This section is optional.