



Princess Sumaya University for Technology

King Hussein School for Computing Sciences

# **Automated Behavior Monitor System**

## **ABMS**

### **Prepared By:**

Marwan Al-Khasawneh 20160088

Saif Al-Katout 20170394

Anas Mahmoud 20170421

### **Supervised By:**

Dr. Abdallah Aref

Project Submitted in partial fulfillment for the degree of Bachelor of Science in  
Computer Science

Spring - 2021

# Declaration of Originality

This document has been written entirely by the undersigned team members of the project. The source of every quoted text is clearly cited and there is no ambiguity in where the quoted text begins and ends. The source of any illustration, image or table that is not the work of the team members is also clearly cited. We are aware that using non-original text or material or paraphrasing or modifying it without proper citation is a violation of the university's regulations and is subject to legal actions.

Names and Signatures of team members:

Marwan Al-Khasawneh  
Saif Al-Katout  
Anas Mahmoud

## Acknowledgments

We would like to thank our families and friends who stood by us throughout our studies at PSUT. We would also like to especially thank our supervisor Dr. Abdallah Aref for guiding us throughout this project.

A special thank you to Dr. Ammar Odeh for helping us in some parts of this project, also, a big thanks to our colleagues who graduated before us and were a huge help in our journey in documenting this project.

# Table Of Contents

King Hussein School for Computing Sciences	0
Spring - 2021	0
Table Of Contents	2
List of Figures	4
List of Tables	5
Chapter 1: Introduction	6
<b>1.1 Overview</b>	6
<b>1.2 Problem Statement</b>	7
<b>1.3 Related Work</b>	7
Chapter 2: Project Plan	8
<b>2.1 Project Deliverables</b>	8
<b>2.2 Project Tasks</b>	9
<b>2.3 Gantt Chart</b>	12
<b>2.4 Risk Assessment</b>	15
<b>2.5 Cost Estimation</b>	16
<b>2.6 Project Management Tools</b>	16
Chapter 3: Requirement Specification	17
<b>3.1 Stakeholders</b>	17
<b>3.2 Functional Requirements</b>	18
<b>3.2.1 Functional Requirements Description:</b>	19
<b>3.3 Non-Functional Requirements</b>	24
Chapter 4: System Design	25
<b>4.1 Logical Model Design</b>	25
<b>4.1.1 Use Case Diagrams:</b>	25
<b>4.1.1.0 Face Identification Use Case:</b>	26
<b>4.1.1.1: Taking Screenshots Use Case:</b>	27
<b>4.1.1.2: Eye Movement Tracking:</b>	28
<b>4.1.1.3: Object Detection</b>	29
4.1.1.4: Gesture Recognition	30
<b>4.1.1.5: Cheating Possibility Calculation</b>	31
<b>4.1.1.6: Voice Recognition System</b>	33

<b>4.1.1.7: Storing Logs</b>	34
<b>4.1.1.8: Sign In Requirements</b>	35
<b>4.1.1.9: Sign Up Requirements</b>	36
<b>4.1.1.10: Forget Password</b>	37
<b>4.1.2 Sequence Diagrams</b>	38
<b>4.1.2.1 Facial Recognition</b>	38
<b>4.1.2.2 Screenshot taking</b>	39
<b>4.1.2.3 Eye Movement Analysis</b>	40
<b>4.1.2.4 Object Detection</b>	41
<b>4.1.2.5 Gesture Detection</b>	42
<b>4.1.2.6 Calculate Cheating Possibility</b>	44
<b>4.1.2.7 Audio Analysis</b>	45
<b>4.1.2.8 Storing Logs</b>	46
<b>4.1.2.9 Signing In</b>	47
4.1.2.10 Signing Up	48
<b>4.1.2.11 Forget Password</b>	49
4.1.3 Class Diagram:	50
4.1.4 Object Diagram:	51
4.1.5 Activity Diagram	53
4.1.6 Package Diagram	56
4.1.7 Component Diagram	57
4.1.8 Deployment Diagram	58
4.1.9 State-Change Diagram	59
<b>4.2 Physical Model Design</b>	62
<b>4.2.1 ERD Diagrams</b>	62
<b>4.3 Test Plan</b>	65
4.3.1 Testing Approach	65
1. Unit Testing	65
2. Stress Testing	65
3. Performance Testing	65
4.3.2 Testing Tools	65
4.3.3 Tested Features	65
4.3.4 Test Risks	66

4.3.5 Pass/Fail Criteria.	67
<b>4.4 User Interface design</b>	74
<b>5 Implementation</b>	83
<b>General Overview</b>	86
<b>Implemented Features</b>	87
<b>Eye Movement Detection Overview:</b>	89
<b>Cheating Prediction Percentage Overview:</b>	91
<b>Face Detection Algorithm Overview:</b>	94
<b>Object Detection Algorithm Overview:</b>	96
<b>Web Application Overview:</b>	97
<b>6 Testing</b>	96
<b>7 Conclusions and Future Work</b>	105
References	109

## List of Figures

Figure 1:Gantt chart Of Project Tasks (Meyringer, 2006)	8
Figure 2:PERT Chart With Critical Path Shown In Red	11
Figure 3:Face Identification Use Case	22
Figure 4:Screenshot Capture Use Case	23
Figure 5:Eye Movement Tracking Use Case	24
Figure 6:Object Detection Use Case Diagram	25
Figure 7:Gesture Recognition Use Case	26
Figure 8:Cheating Possibility Use Case	27
Figure 9:Voice Recognition Use Case	29
Figure 10:Store Logs Use Case	30
Figure 11:Sign In Use Case	31
Figure 12:Sign Up Use Case	32
Figure 13:Forget password use case	33
Figure 14:Facial Recognition Sequence Diagram	34
Figure 15:Screenshot Taking Sequence Diagram	35
Figure 16:Eye-Movement Detection Sequence Diagram	36

Figure 17: Object Detection Sequence Diagram	37
Figure 18: Gesture Detection Sequence Diagram	38
Figure 19: Calculate Cheating Possibility Sequence Diagram	39
Figure 20: Voice Analysis Sequence Diagram	40
Figure 21: Storing Logs Sequence Diagram	41
Figure 22: Signing In Sequence Diagram	42
Figure 23: Signing Up Sequence Diagram	43
Figure 24: Forget password sequence diagram	44
Figure 25: Class Diagram	45
Figure 26: Object Diagram	47
Figure 27: Activity Diagram	50
Figure 28: Package Diagram	51
Figure 29: Component Diagram	52
Figure 30: Deployment Diagram	53
Figure 31: State Change Diagram- Behavior Detection	55
Figure 32: State Change Diagram- Sign-Up and Login	56
Figure 33: ER Model	58
Figure 34: ER diagram	59
Figure 35: Login Figure	70
Figure 36: Sign Up Interface	71
Figure 37: Session interface	72
Figure 38: Object Detection interface	73
Figure 39: Impersonation Incident Interface	74
Figure 40: Eye movement detection interface	75
Figure 41: Logs Interface	76
Figure 42: Forget password interface	77
Figure 43: Passcode interface	77
Figure 44: New password interface	78
Figure 45: Students homepage interface	79
Figure 46: Monitors homepage interface	79
Figure 47: Face validation for session	80

## List of Tables

Table 1: Related work	6
Table 2: Detailed Gantt chart	9
Table 3: Roles and responsibilities	11
Table 4: Functional Requirements	14
Table 5: FR1	16
Table 6: FR2	16
Table 7: FR3	17

Table 8:FR4	17
Table 9:FR5	18
Table 10:FR6	18
Table 11:FR7	19
Table 12:FR8	19
Table 13:FR9 + FR10 + FR11 + FR15 + FR16	20
Table 14:FR12 + FR13 + FR14 + FR17	20
Table 15:Non-Functional Requirements	21
Table 16:Face identification	23
Table 17:Taking screenshot	24
Table 18:Eye movement	25
Table 19:Object detection	26
Table 20:Gesture Recognition	27
Table 21:Cheating Possibility	28
Table 22:Voice Recognition	29
Table 23:Storing Logs	30
Table 24:Login	31
Table 25:Signup	32
Table 26:Forget Password	33
Table 27:Tests Risk	61
Table 28:Test cases	62

## Chapter 1: Introduction

### 1.1 Overview

In the past year, lots of institutions around the globe were forced to use distance learning. This has created many problems that affected the teaching process negatively. One of the biggest problems that occurred was cheating during online exams; This was a result of the lack of monitoring and ease of access to different unauthorized tools during the time of the exams.

In our project, we tried to tackle this issue by creating an automated behavior detection system that utilizes image processing and AI to detect cheating breaches automatically without the need for a human monitor.

Many similar solutions were introduced in recent years, our project will introduce more features with fewer resources consumed compared to the other solutions.

## 1.2 Problem Statement

The main issues we aim to address are:

- 1- Confirming the identity of the person in an online exam preventing impersonation.
- 2- Time wasted during the process of taking attendance before an exam starts.
- 3- Students signing in but not attending their lectures.
- 4- Having multiple people within the same room trying to help a student during an online exam.
- 5- Students using their materials for cheating during online exams.

## 1.3 Related Work

*Table 1: Related work*

	<b>Room Scan</b>	<b>Gesture Recognition</b>	<b>Face Recognition</b>	<b>Eye Movement Tracking</b>	<b>Audio Processing</b>	<b>Attendance Matching</b>
<b>Mettl</b> [3]	Yes	No	Yes	Yes	No	No
Proctorio [6]	Yes	Yes	Yes	No	No	No
<b>Proctorfree</b> [1]	No	Yes	Yes	No	No	No
<b>ProctorU</b> [2]	Yes	No	Yes	Yes	Yes	Yes
<b>ProctorExam</b> [5]	Yes	No	Yes	No	No	Yes
<b>ABMS</b>	Yes	Yes	Yes	Yes	Yes	Yes

## 1.4 Document Outline



## **Chapter(2)**

In Chapter 2 we will discuss in detail our timeline, deliverables and tasks. We will also list the risks we are going to encounter and the responsibilities of each team member in addition to the cost estimation and tools used.

## **Chapter(3)**

In this chapter we will highlight the stakeholders we expect to benefit from this system, we will list what our system will offer and the requirements it will meet, both functional and nonfunctional.

## **Chapter(4)**

Chapter 4 focuses on how our functional requirements are visually represented (Both logical and physical model designs) by diagrams such as:

- Use Case Diagrams
- Sequence Diagrams
- Class Diagrams
- ER diagrams

# Chapter 2: Project Plan

## **2.1 Project Deliverables**

- A. Project Plan including Gantt chart to provide a timeline for the project's analysis and design process.
- B. An analysis of the functional and non-functional project requirements.
- C. Use-case diagrams for each process in the system.
- D. Logical and Physical Model Designs which our project will be based on when implemented.
- E. A functional system software that has the following capabilities:
  - Recognizes students and confirms their identity.
  - Takes a screenshot of the webcam video for image processing purposes.
  - Detects student behavior in real-time.
  - Calculates the probability of cheating for each student during online exams.

- Detects high audio levels in case of human help.
- F. A web application that will be used to test the developed software.
  - G. The source code for the developed system.
  - H. A design presentation made to present the project to the graduation project committee.
  - I. **Testing results:** the results of all the tests done and what errors did the risk item trigger. Also, showing which tests failed and what samples caused them to fail.

## 2.2 Project Tasks

Note: **Graduation Project 1 Tasks Starting From October 12:**

*Table 2: Detailed Gantt chart*

Task ID	Task name	<u>Start Date</u>	<u>End Date</u>	<u>Duration (Day)</u>
T1	Form team	12/10	13/10	2
T2	Researching Ideas	14/10	19/10	6
T3	Decide Idea	20/10	20/10	1
T4	Requirements Specification	21/10	27/10	7
T5	Modeling requirements	28/10	4/11	7
T6	Use Case Diagrams	5/11	9/11	5
T7	Sequence Diagram	10/11	17/11	7
T8	Class Diagrams	18/11	20/11	3
T9	Object Diagrams	21/11	23/11	3
T10	Activity Diagram	24/11	28/11	4
T11	Package Diagram	9/12	14/12	5
T12	Component Diagram	9/12	14/12	5

T13	Deployment Diagram	9/12	14/12	5
T14	State Transition Diagram	15/12	17/12	3
T15	ERD	16/12	19/12	3
T16	Testing	20/12	26/12	7
T17	Graphical User Interface (GUI)	27/12	1/1	6
T18	Review All Prior Work and Update	2/1	8/1	7
T19	Defining the software tools and programming languages	9/1	13/1	5
T20	Discussing with QRFED about the possibility of commercializing	14/1	16/1	3

**Note: Graduation Project 2 Tasks Starting From Feb 21:**

*Table 29: Tasks during graduation project 2*

Task ID	Task name	<u>Start</u>	<u>End</u>	<u>Duration (Weeks)</u>	<u>Role</u>
H1	Facial Recognition Algorithm	Week 1	Week 3	3	Anas
H2	Object Detection	Week 1	Week 3	3	Saif
H3	Eye movement	Week 1	Week 3	3	Marwan
H4	Code Enhancements on detection algorithms	Week 4	Week 5	2	All: Each on his work

H5	Cheating percentage calculation	Week 4	Week 5	2	Marwan
H6	Web Application Initialization: SQL, Login Page and sign up page	Week 6	Week 6	1	Anas + Saif
H7	Link all detection functions and cheating percentage calculator	Week 6	Week 6	1	Marwan
H8	Create HTML interface	Week 6	Week 6	1	Anas
H9	Collecting High quality Datasets for improved accuracy	Week 7	Week 7	1	Marwan + Saif
H10	Implementing the algorithms on the web environment	Week 8	Week 10	2	Saif
H11	Final modifications on the code	Week 11	Week 11	1	All
H12	Black Box Testing	Week 12	Week 13	2	All

## 2.3 Gantt Chart

**teamgantt**  
Created with Free Edition

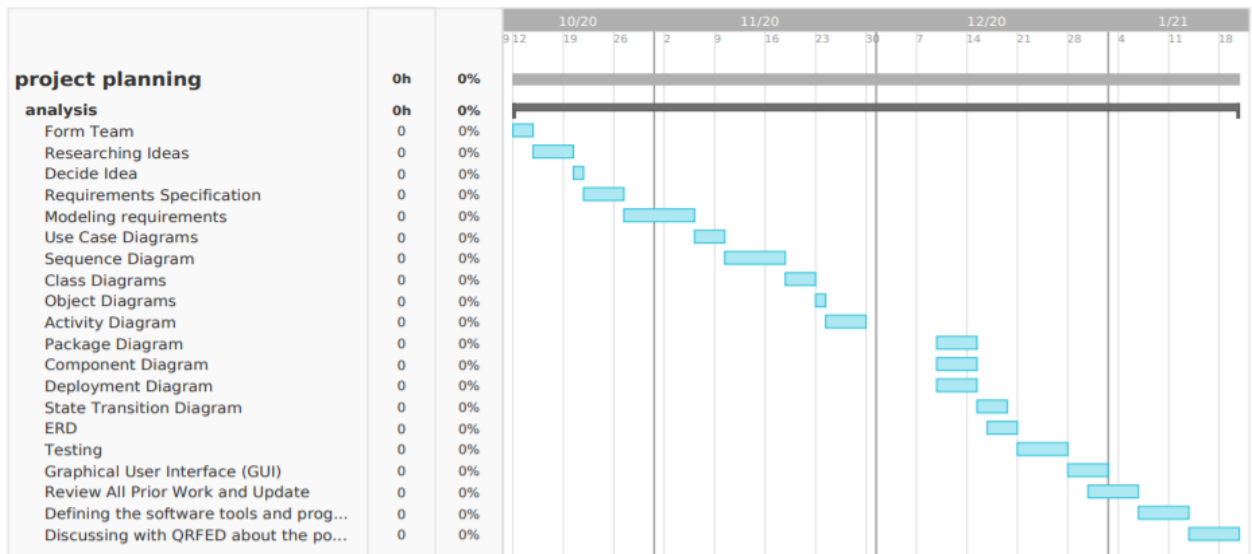


Figure 1: Gantt chart Of Project Tasks (Meyringer, 2006)

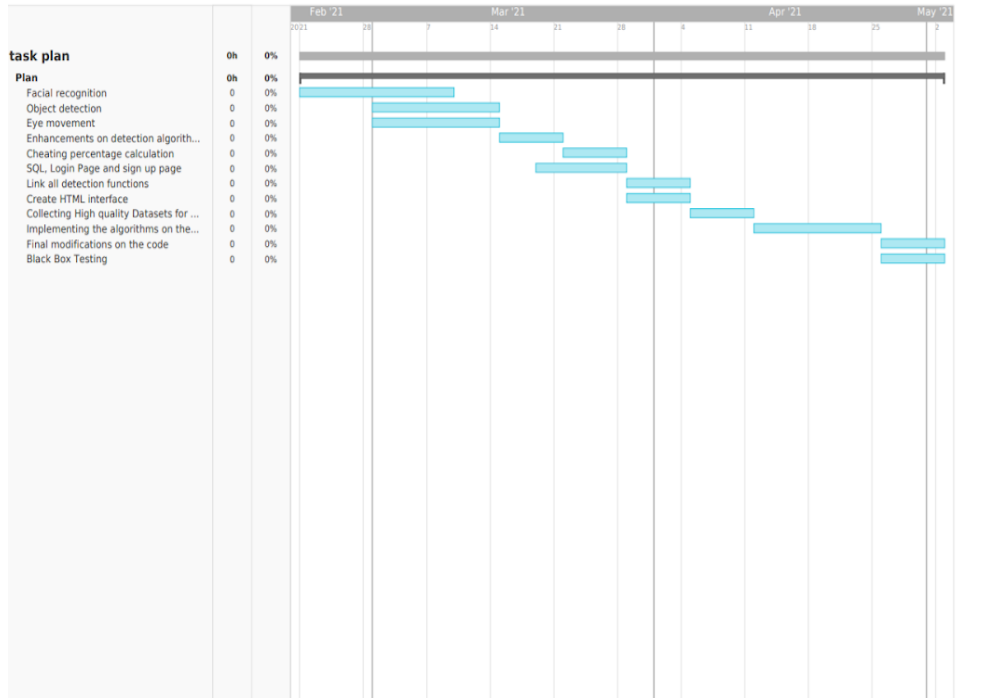


Figure 60:Gantt chart Of Project Tasks For Graduation Project 2 (Table 111) (Meyringer, 2006)

The Gantt chart shows the sequence of tasks and how they are distributed by weeks or days.

### 2.3.1 List of Roles and responsibilities

Table 3: Roles and responsibilities

Saif	Anas	Marwan
<ul style="list-style-type: none"> <li>● Research about prior work</li> <li>● Non-Functional Requirements</li> <li>● FR2, FR3, FR5 Use cases</li> <li>● Pert Chart</li> <li>● FR2, FR3, FR5 Sd Diagrams</li> <li>● Add extra details FR2,FR3,FR5</li> <li>● Review all work</li> <li>● Object Detection Implementation</li> <li>● Web Portal Implementation</li> </ul>	<ul style="list-style-type: none"> <li>● Write Overview</li> <li>● Problem Statement</li> <li>● Functional Requirements</li> <li>● FR6, FR7, FR8 Use cases</li> <li>● FR6, FR7, FR8 Sd Diagrams</li> <li>● Add extra details FR6,FR7,FR8</li> <li>● Test Plan Review</li> <li>● Attendance Implementation</li> <li>● Facial Recognition Implementation</li> </ul>	<ul style="list-style-type: none"> <li>● Research about prior work</li> <li>● Functional Requirements</li> <li>● FR1, FR4, Use cases</li> <li>● Progress Reports</li> <li>● Cost Estimation</li> <li>● Risk Assessment</li> <li>● Test Plan</li> <li>● Eye Movement Detection Implementation</li> <li>● Cheating Calculation Implementation</li> </ul>
<ul style="list-style-type: none"> <li>● <b>Class Diagram</b></li> <li>● <b>Object Diagram</b></li> <li>● <b>Activity Diagram</b></li> <li>● <b>Deployment Diagram</b></li> <li>● <b>Package Diagram</b></li> <li>● <b>Component Diagram</b> <ul style="list-style-type: none"> <li>● GUI</li> </ul> </li> <li>● <b>ERD Normal Form</b></li> <li>● <b>Database Implementation</b></li> </ul>		

### 2.3.2 Pert chart

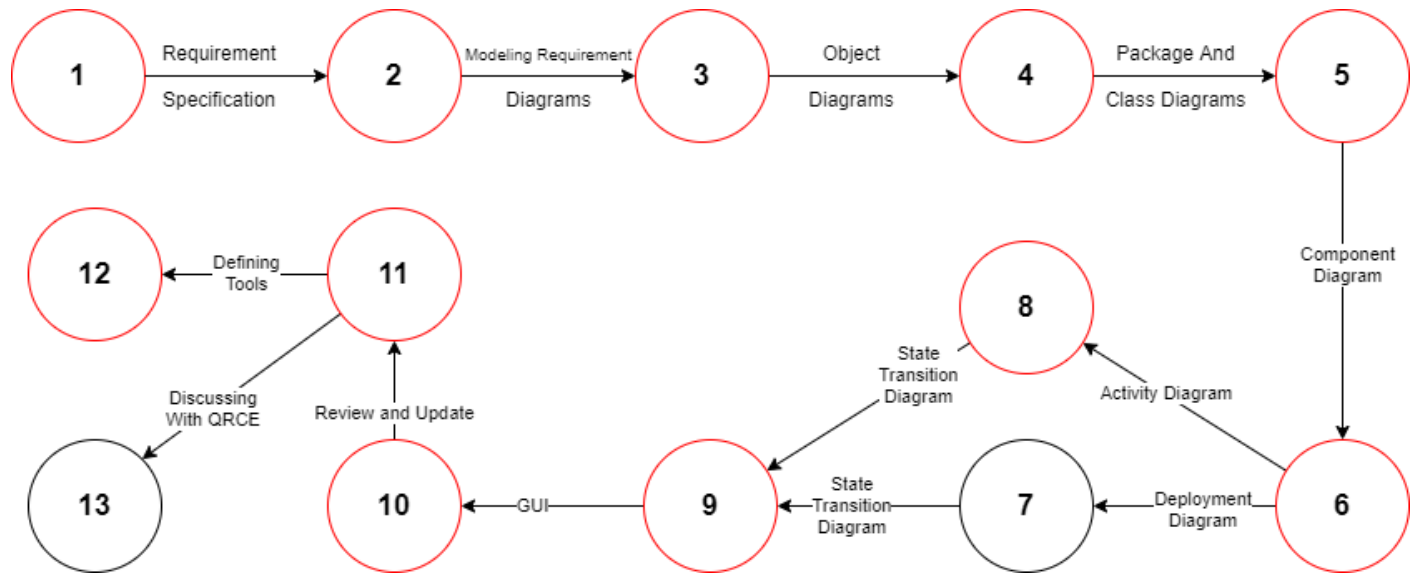


Figure 2: Pert Chart With Critical Path Shown In Red



Figure 61: Pert Chart Of Work Of The Second Semester (Table 111)

## 2.4 Risk Assessment

In this section we will list the risks that we are encountering while working in this project, some of which are critical and some of which are not.

1. Not being able to balance between the academic tasks such as assignments and exams with the graduation project tasks.
2. This project will be done in a time where a global pandemic is taking place, this imposes many challenges:
  - a. The risk of one or more members of the development team getting infected with covid-19 and not being able to work for a certain period of time.
  - b. After switching to online learning due to the pandemic there's a chance of the first academic semester to be canceled by the ministry of higher education which would lead to the cancelation of the whole project.
  - c. The chance that our supervisor will get infected with the virus which would slow our progress down until we find a substitute.
  - d. The need for social distancing has caused us to rely on Video Conferencing Applications as our main communication tool between the team members and



between the team and the supervisor. Calls over the web are not as clear as face to face meetings and confusion could take place causing our progress to slow down.

3. This project will include advanced Computer Science topics such as image processing and machine learning, these topics were not obligatory in the Computer Science bachelor's program at PSUT and therefore learning these topics in around 6 months will be challenging. There's a chance that one or more members of the development team will find it hard and not be able to become versed in these topics.
4. Not being able to find the required software libraries and components for our solutions which would cost more time in order to try to implement them from scratch.
5. Inaccurate estimation of the time needed to analyze and design the project due to underestimating the size of it.
6. Inaccurate results yielded from the machine learning algorithm used to implement the project.
7. In development teams there's always a chance of misunderstanding between two or more members which could cause someone or more to withdraw from the team would slow time as re-planning and redistributing the tasks will be needed.

## 2.5 Cost Estimation

The estimated cost of this project can be summed up in:

1. Hardware used by the developers such as Laptops, PCs, Printers and Smartphones which would be estimated at 3000\$.
2. Internet bundles used to communicate virtually between team members for around 8 months would be estimated at 2000\$.
3. Extra Computer Science courses that weren't included in the undergraduate Computer Science program in PSUT and are needed to be able to implement this project are estimated at 1000\$, such as (Complete Python Based Image Processing
4. Face masks and protection gear used to protect from Covid-19 when the development team meets in public places. 500\$
5. Time spent by development team members on discussing, designing, analyzing and implementing the project. 2500\$

## 2.6 Project Management Tools

- **GitHub:** It is the tool of choice to share the files between the team members and our supervisor. We choose GitHub since it is reliable, offers lots of free features and we have experienced using it during our study at PSUT.

- **Zoom:** It is the tool used to communicate between the members and our supervisor, Zoom is familiar to us since it is heavily used in Online Learning and it provides good features for conferencing such as screen sharing and polls, it is also the only tool offered by PSUT for such purposes.
- **TeamGantt.com:** Which is an online, free and easy-to-use tool that we used to plan our project tasks and organize roles and responsibilities that would fit our time schedules.

## Chapter 3: Requirement Specification

### 3.1 Stakeholders

1. **Educational Institutions:** This system would be useful for these institutions to automate the supervision of exams and lectures which would lead to more accurate monitoring.
2. **Students:** Distributing grades fairly and preventing cheating and impersonation which would lead to a healthier educational environment. This will also provide students the privacy they want by letting a machine monitor their behavior not a human.
3. **Proctors:** Instead of manually tracking several students at once this system will monitor each student independently and automatically which leads to more accurate results consuming less time and effort.
4. **Lecturers:** An automated attendance system and gesture detection would save their time and effort. Detecting students signing in but not attending the lecture would also be automated, in addition to detecting when someone not enrolled in the class is attending it.
5. **Project Team Members:** The team members will face new challenges in their field of study, they will learn new topics such as image processing and machine learning which would benefit them in their career.
6. **Project Supervisor:** Supervising the process of creating and following through with the project plan.
7. **Employers:** Some companies require their applicants to pass a certain exam to get the interview. Our project would provide them the ability to offer this exam online.

### 3.2 Functional Requirements

Table 4:Functional Requirements

<b>ID</b>	<b>Description</b>
FR1	User's faces should be identified and matched with their picture in the database to take attendance or detect impersonation incidents.
FR2	System should take a screenshot every random amount of seconds.
FR3	System should analyze the user's eye movement.
FR4	System should scan the user's background looking for objects such as smartphones, laptops and other people.
FR5	The system should provide a gesture recognition to perform a certain action
FR6	System should be able to calculate the possibility of a user cheating. This possibility is derived from the screenshot and audio processing in FR2,FR3,FR4 and FR7
FR7	System should be able to identify if the user is talking to someone.
FR8	System should store all the logs that resulted from the monitoring process for future use.
FR9	Users should be able to sign in using their University ID.
FR10	System should be able to recognize users stored in the database. Based on their picture identification.
FR11	System should be able to differentiate between students and monitors stored in the database in order to give each the right permissions.

FR12	New students should be able to sign up and upload their pictures.
FR13	System should be able to check whether a new picture is eligible for use in facial recognition.
FR14	System should be able to store new students' pictures in the database.
FR15	System should be able to give monitors access to the logs stored in the database.
FR16	System should provide a forget password option by sending an email that allows the user to enter a new password.
FR17	System should be able to give monitors access to the logs stored in the database.

### 3.2.1 Functional Requirements Description:

Table 5:FR1

<b>Functional Requirement:</b>	FR1
<b>Pre-Conditions:</b>	A student enters a session.
<b>Main Scenario:</b>	The student will enter the session and then the web application will pop up the webcam, the webcam will take a screenshot and send it to the image processing algorithm which will compare the face of the student to the student database, once recognized the student will enter the session and start their session.
<b>Alternate Scenario:</b>	The student face does not get recognized, in this case the student will be taken back to the main page with an error message.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>● Preventing impersonation incidents</li> <li>● Recording the students attendance</li> </ul>
<b>Input:</b>	A screenshot by the students webcam
<b>Output:</b>	A result of the recognition algorithm

Table 6:FR2

<b>Functional Requirement:</b>	FR2
<b>Pre-Conditions:</b>	A student is in a session.
<b>Main Scenario:</b>	Once the student is in a session the webcam will start and it will take a screenshot every random period of time and send it to the image processing algorithm.
<b>Alternate Scenario:</b>	The webcam does not work at which the system will warn the student after 3 failed attempts of taking a screenshot and if the issue is not fixed the student will be blocked from taking the exam
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>● Preventing cheating</li> <li>● Preventing impersonation during the exam</li> <li>● Prevent the use of outside help</li> <li>● Prevent turning off the camera during the exam</li> </ul>
<b>Input:</b>	A screenshot by the student webcam
<b>Output:</b>	Image analysis results

Table 7:FR3

<b>Functional Requirement:</b>	FR3
<b>Pre-Conditions:</b>	student is in an exam session.
<b>Main Scenario:</b>	The system will process the student's eye movement to check if the student's eyesight is focused on the monitor for a reasonable ratio, which will keep the cheating percentage in normal levels. .
<b>Alternate Scenario:</b>	If the student's eyesight is off monitor for a big amount of time this will trigger a warning and raise a flag, the student will be warned and it will be calculated in the focusing on the monitor percentage.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>● Tracking of the student's eye movement</li> </ul>
<b>Input:</b>	A screenshot by the student's webcam
<b>Output:</b>	<ul style="list-style-type: none"> <li>- A percentage of the ratio that the student was looking off-screen</li> <li>- The screenshots where the student wasn't looking at the screen for future use by proctors.</li> </ul>

Table 8:FR4

<b>Functional Requirement:</b>	FR4
<b>Pre-Conditions:</b>	student is in an exam session.
<b>Main Scenario:</b>	While the student is in an exam session the webcam will keep on sending screenshots to the image processing algorithm, if a certain prohibited object was detected the system will record it in the log and will pop up a warning message to the student.
<b>Alternate Scenario:</b>	If the system does not detect any object nothing will appear to them and they will be taking the exam normally.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>• Detecting any prohibited objects</li> </ul>
<b>Input:</b>	A screenshot by the student webcam
<b>Output:</b>	A result of the object detection algorithm

Table 9:FR5

<b>Functional Requirement:</b>	FR5
<b>Pre-Conditions:</b>	<ul style="list-style-type: none"> <li>- Student is in an lecture session.</li> <li>- Lecturer started a poll</li> </ul>
<b>Main Scenario:</b>	If the lecturer starts a poll, students gesture a thumbs up or down, votes are counted in the system then produced in the log.
<b>Alternate Scenario:</b>	In case of a poll If the system does not detect any gesture it will ask the student to try again.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>• Detection of votes in the poll</li> </ul>
<b>Input:</b>	A video stream of the student's webcams for a short period of time.
<b>Output:</b>	The number of positive or negative votes

Table 10:FR6

<b>Functional Requirement:</b>	FR6
<b>Pre-Conditions:</b>	The student is identified and in an exam session.
<b>Main Scenario:</b>	While the student is taking the exam, the behavioral system will detect the eye movement, objects and background image to make sure only the student is taking the exam.
<b>Alternate Scenario:</b>	The system is not able to detect the student's behavior due to a problem in the webcam.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>Calculating the cheating possibilities. Based on the input.</li> </ul>
<b>Input:</b>	Eye movement percentage, object detection result and background detection result.
<b>Output:</b>	The result of cheating percentage.

Table 11:FR7

<b>Functional Requirement:</b>	FR7
<b>Pre-Conditions:</b>	Voice levels are higher than normal.
<b>Main Scenario:</b>	The system will detect the audio locally and if a higher audio level was detected it would record it and process it to see if another human voice was present or not.
<b>Alternate Scenario:</b>	No high level voices are detected.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>Detection of other human voice sources.</li> </ul>
<b>Input:</b>	Room audio.
<b>Output:</b>	The result of audio analysis.

Table 12:FR8

<b>Functional Requirement:</b>	FR8
<b>Pre-Conditions:</b>	<ul style="list-style-type: none"> <li>- student is identified and in a session</li> <li>- The automated behavioral detection algorithms are working.</li> </ul>
<b>Main Scenario:</b>	While the student is taking the exam, the behavioral system will record eye movement, objects and background image and store them in a log for future use
<b>Alternate Scenario:</b>	None.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>● Storing logs.</li> </ul>
<b>Input:</b>	student's detected behavior.
<b>Output:</b>	Detailed saved logs for future use.

Table 13:FR9 + FR10 + FR11 + FR15 + FR16

<b>Functional Requirement:</b>	FR9 + FR10 + FR11 + FR15 + FR16
<b>Pre-Conditions:</b>	<ul style="list-style-type: none"> <li>- User is signed up</li> <li>- User enter correct login credentials</li> </ul>
<b>Main Scenario:</b>	User enters the webpage, fills their username and password, submits, logs in successfully and is given the right permissions.
<b>Alternate Scenario:</b>	User enters wrong login credentials or forget password.
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>● User is signed in.</li> <li>● User is given the right permissions to access logs in case of a monitor.</li> </ul>
<b>Input:</b>	Users credentials.
<b>Output:</b>	FR9 + FR10 + FR11 + FR15 □ Homepage. FR16 □ An email is sent to change password.



Table 14:FR12 + FR13 + FR14 + FR17

<b>Functional Requirement:</b>	FR12 + FR13 + FR14 + FR17
<b>Pre-Conditions:</b>	<ul style="list-style-type: none"> <li>- Username filled is unique.</li> <li>- Password is 3 - 12 characters long.</li> <li>- Picture used is eligible for facial recognition.</li> </ul>
<b>Main Scenario:</b>	The user enters the sign up page, fills their information and submits them which signs them up successfully, the system then gives the users the right privileges.
<b>Alternate Scenario:</b>	<ul style="list-style-type: none"> <li>- Username is a duplicate.</li> <li>- Password is weak</li> <li>- Picture is not eligible for facial recognition</li> </ul> <p>This will show an error message and ask the user to submit again.</p>
<b>Post Conditions:</b>	<ul style="list-style-type: none"> <li>● Users signed up successfully and their picture is stored in the database along with their other information.</li> </ul>
<b>Input:</b>	<ul style="list-style-type: none"> <li>- Users information.</li> <li>- Users picture.</li> </ul>
<b>Output:</b>	<ul style="list-style-type: none"> <li>- A user with the right privileges.</li> </ul>

### 3.3 Non-Functional Requirements

Table 15:Non-Functional Requirements

ID	Requirement	Description
NFR1	Security	Privacy of information taken. The use of images should be forbidden by non-authorized users.
NFR2	Performance	The system should get the results in less than 5 seconds when an incident happens.

NFR3	Accuracy	The cheating percentage that the system produces should be at least 75% representative of the actual behavior of the student.
NFR4	Usability	All system features should be functional and easy to use by the proctors after a maximum of two orientation sessions. People who have access to this system can learn how to use it from the first time.
NFR5	Reliability	This system should work perfectly with no errors even in the long term and can handle 90 students at the same time with no problems.
NFR6	Scalability	The system should be able to handle adding more features in the future.

## Chapter 4: System Design

### 4.1 Logical Model Design

Designing our system we decided to choose the object oriented approach because we're familiar with it. All the diagrams needed in this approach are listed below.

#### 4.1.1 Use Case Diagrams:

A use case diagram is a diagram that visualizes the behavior of the system according to the behavior of the actor. In other words, it visualizes the interaction between the system and the user.

### 4.1.1.0 Face Identification Use Case:

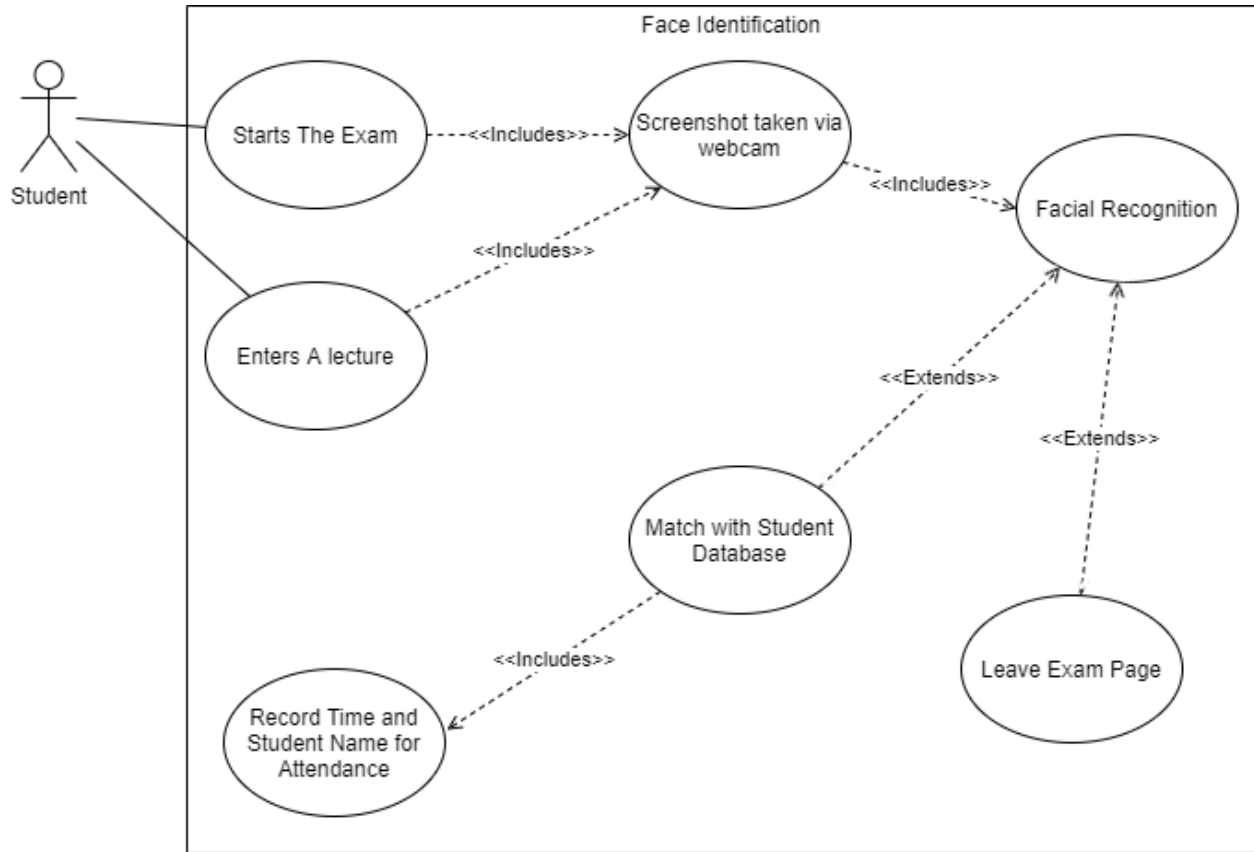


Figure 3:Face Identification Use Case

Table 16:Face identification

Use Case Element	Description
<b>Use Case Number:</b>	1
<b>Application Use Case Name:</b>	Attendance Tracking and Impersonation Prevention.
<b>Use Case Description:</b>	Once the student starts an exam or a lecture the webcam will open and it will take a screenshot which will be sent to the image processing algorithm for facial recognition, if the student was identified the system will record their attendance, if not it will leave the exam or the lecture page.
<b>Primary Actor:</b>	Student.
<b>Basic Flow:</b>	Student Enters, webcam opens, student is notified when the identification process is done and their attendance is recorded.

<b>Alternate Flow:</b>	The student's face is not identified and is taken out of the exam or lecture page.
------------------------	--

#### 4.1.1.1: Taking Screenshots Use Case:

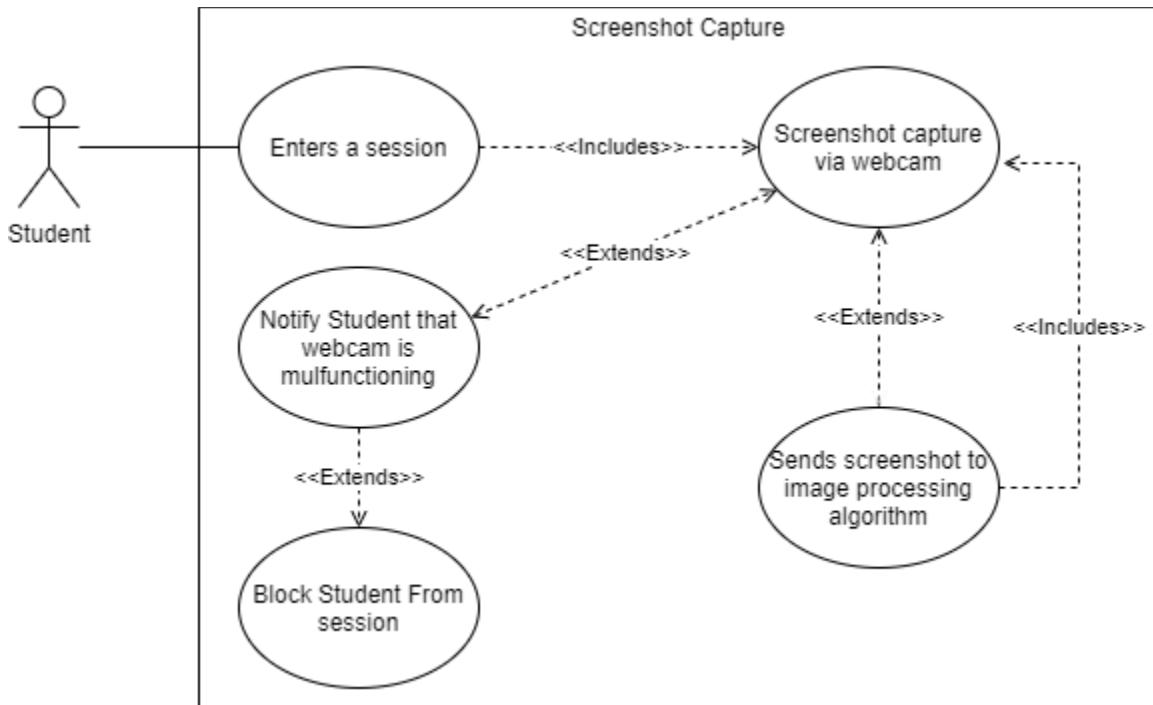


Figure 4: Screenshot Capture Use Case

Table 17: Taking screenshot

Use Case Element	Description
<b>Use Case Number:</b>	2
<b>Application Use Case Name:</b>	Screenshot capturing.
<b>Use Case Description:</b>	The system takes a screenshot every random interval of time, and then sends this image to the server in order to be processed.
<b>Primary Actor:</b>	Student.
<b>Basic Flow:</b>	Image is sent to server, passed on to algorithm for processing

<b>Alternate Flow:</b>	Webcam is not functioning, the system will try to capture 3 times while notifying the student about the malfunction, if it wasn't fixed the student will be banned from the session.
------------------------	--

#### 4.1.1.2: Eye Movement Tracking:

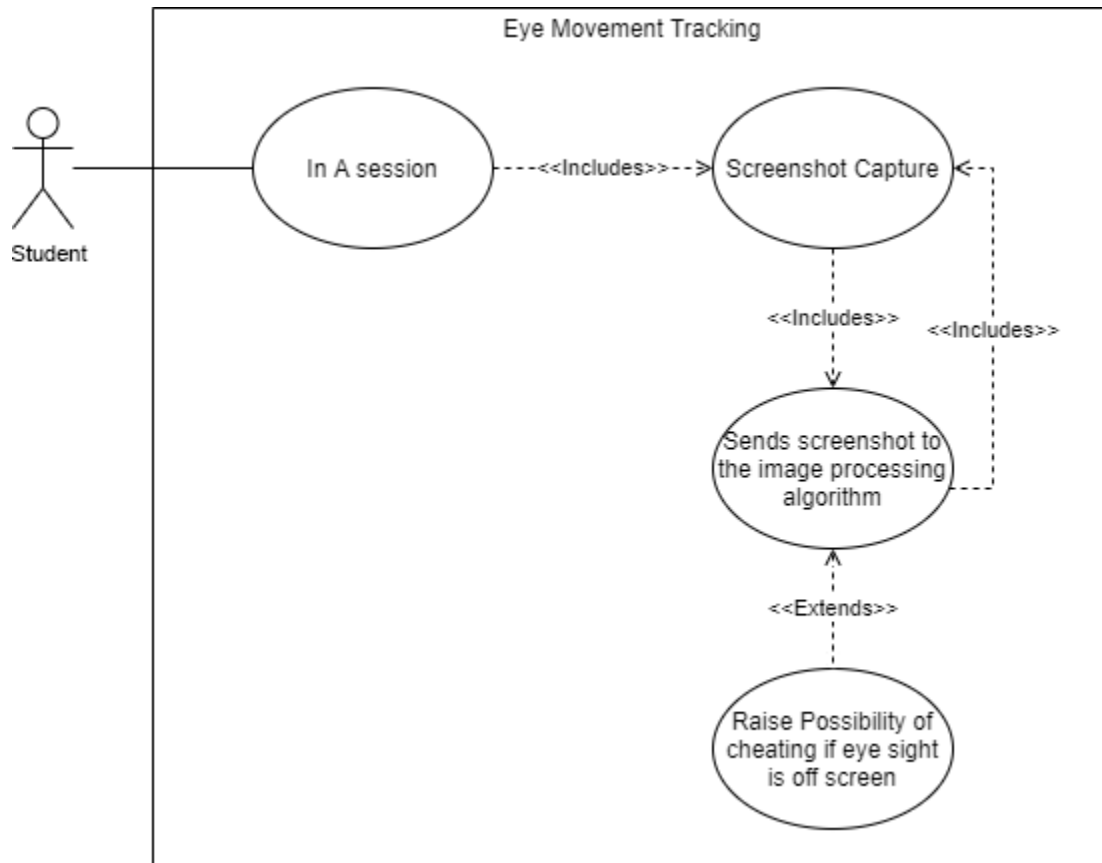


Figure 5:Eye Movement Tracking Use Case

Table 18:Eye movement

Use Case Element	Description
<b>Use Case Number:</b>	3
<b>Application Use Case Name:</b>	Eye Movement Tracking.
<b>Use Case Description:</b>	The system tracks the students' eye movement, to analyze what direction they are looking at, to count the possibility of them cheating using books or smartphones around them.
<b>Primary Actor:</b>	Student
<b>Basic Flow:</b>	While the student is in a session, the system captures a screenshot every

	period of time, and sends it to the server for analysis, if the student was looking off screen, it would raise the possibility of them looking off-screen, thus raising the possibility of them cheating.
<b>Alternate Flow:</b>	The system detects that the student is looking on screen and captures another screenshot after a period of time.

**4.1.1.3: Object Detection**

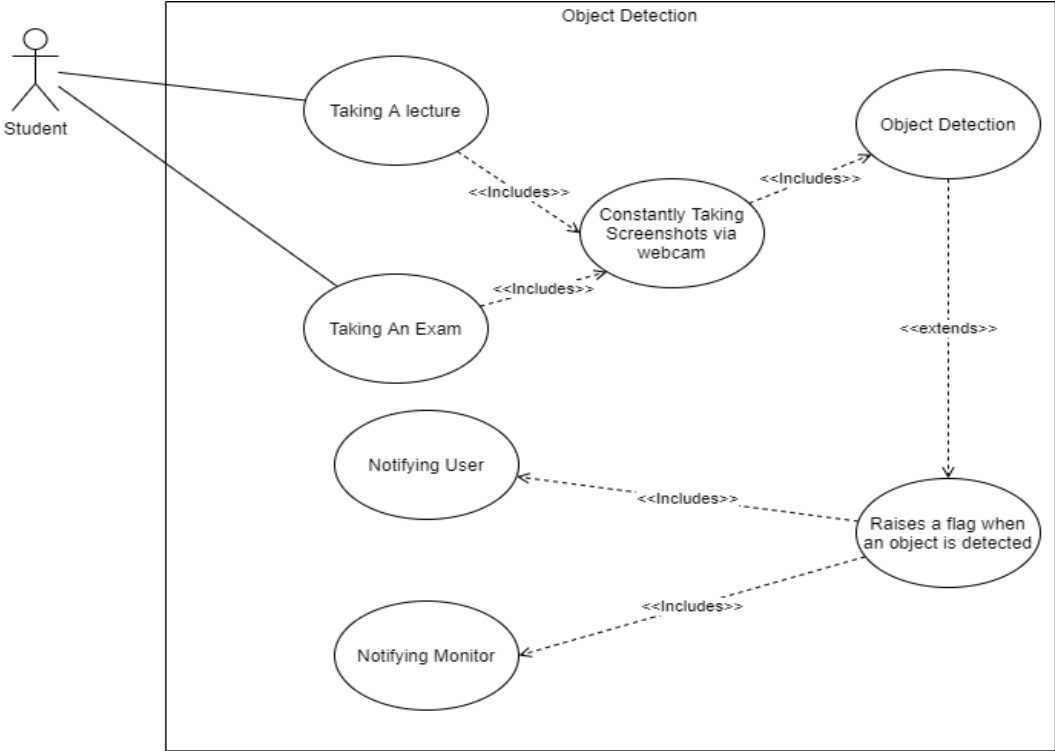


Figure 6: Object Detection Use Case Diagram

Table 19: Object detection

Use Case Element	Description
<b>Use Case Number:</b>	<b>4</b>
<b>Application Use Case Name:</b>	Preventing the use of smartphones during exams or lectures.
<b>Use Case Description:</b>	While the student is in an exam or a lecture screenshots will be constantly sent to the image processing algorithm to detect objects, if an object is detected the student will be notified as well as the monitor by recording it in the log.
<b>Primary Actor:</b>	Student

<b>Basic Flow:</b>	Student Taking A lecture or An Exam, An Object is Detected, Notifications Sent.
<b>Alternate Flow:</b>	No objects are detected.

4.1.1.4: Gesture Recognition

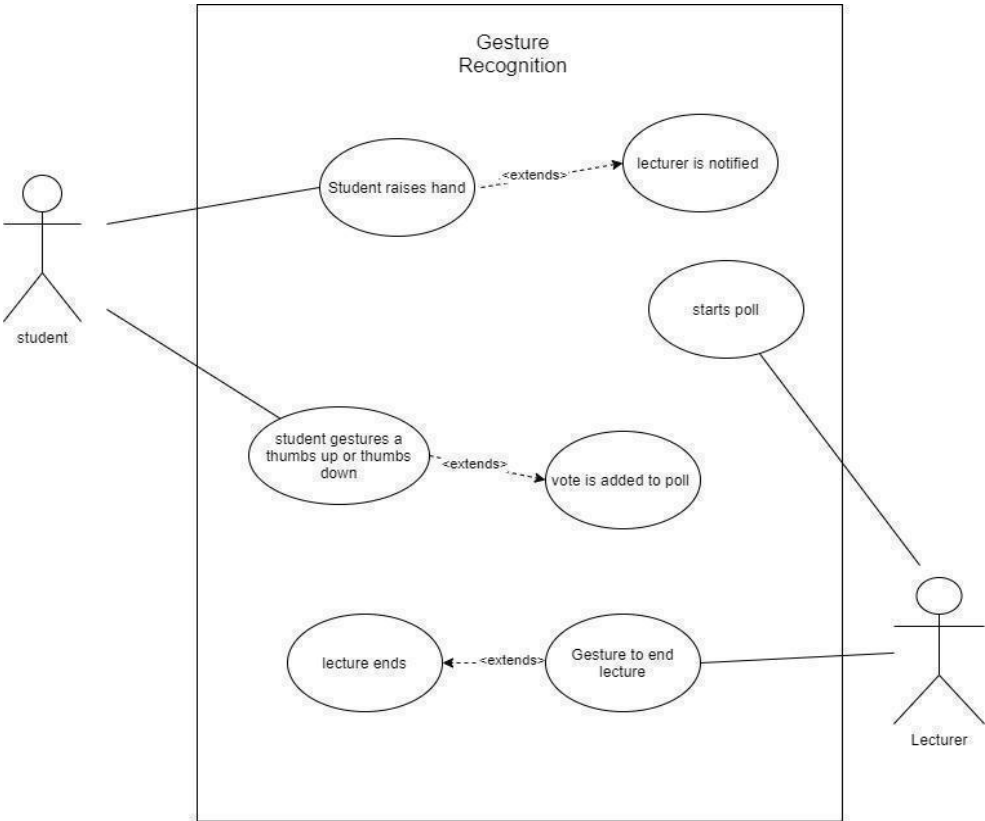


Figure 7:Gesture Recognition Use Case

Table 20:Gesture Recognition

Use Case Element	Description
<b>Use Case Number:</b>	5
<b>Application Use Case Name:</b>	Gesture Detection
<b>Use Case Description:</b>	The system should provide gesture recognition to perform a certain action
<b>Primary Actor:</b>	People in class(lecturer, student)

<b>Basic Flow:</b>	The gesture is recognized, the action that coincides with the gesture starts.
<b>Alternate Flow:</b>	The gesture is not recognized, the gesturer must try again.

#### 4.1.1.5: Cheating Possibility Calculation

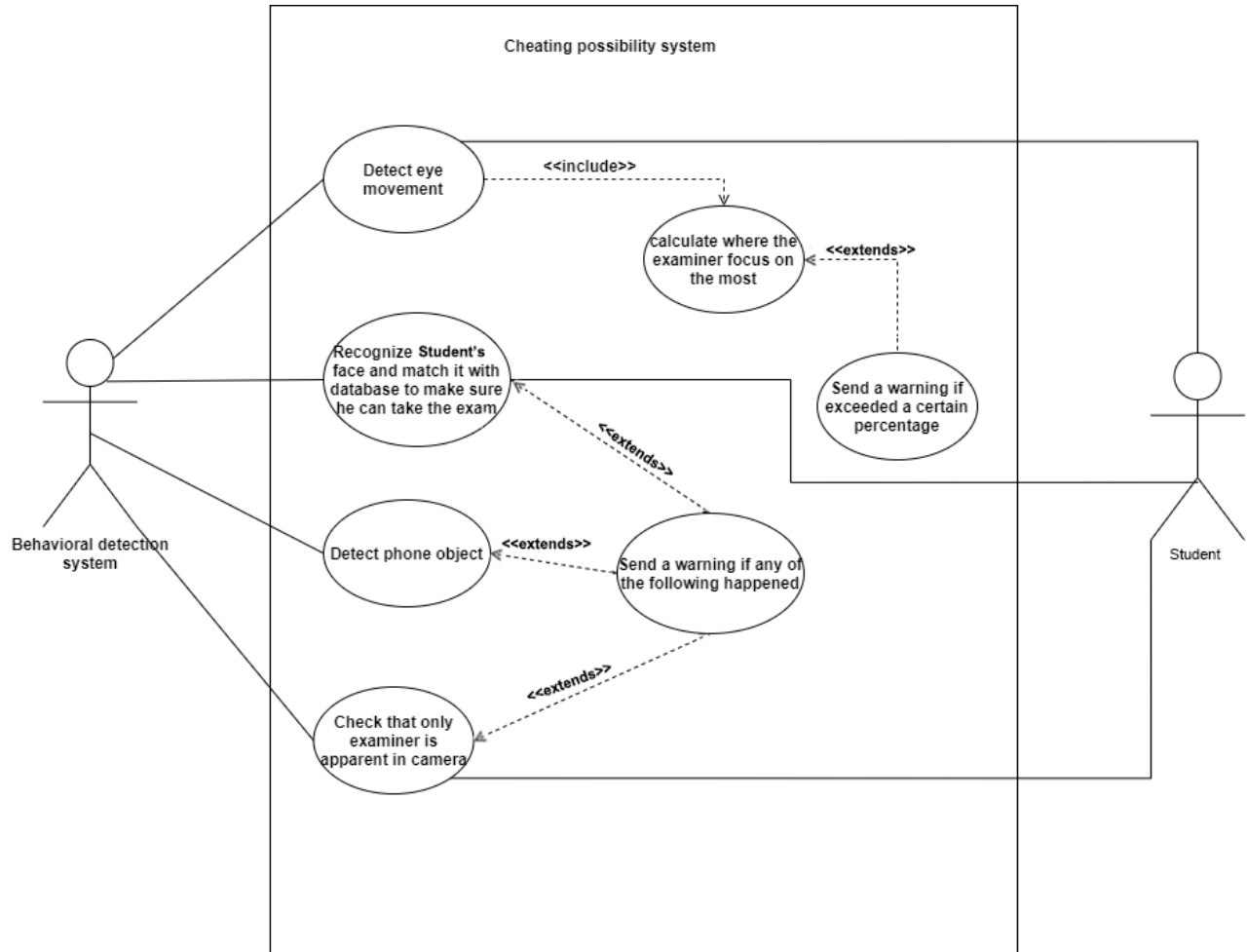


Figure 8: Cheating Possibility Use Case

Table 21: Cheating Possibility

Use Case Element	Description
Use Case Number	6
Use Case Name	Cheating possibility
Use Case Description	The system will make sure that the student is not cheating during the exam through the following: 1- Detect the eye movement and where the student is focusing on the most during the exam.



	<ul style="list-style-type: none"> <li>2- Check the identity of the student by comparing his/her picture with the database.</li> <li>3- Detect any object used to cheat like smartphones.</li> <li>4- Make sure only the student is apparent in the camera to guarantee no one is helping him/her.</li> </ul>
Primary Actor	student
Basic Flow	The examinee takes the exam normally while the cheating percentage is below 40%
Alternate Flow	If the examinee exceeds a certain percentage of the abnormal eye movement, another person is observed sitting in the place of the examinee, or there are two people present, a warning is sent to the examinee and the observer to alert as well as raise a flag to increase the possibility of cheating.

### 4.1.1.6: Voice Recognition System

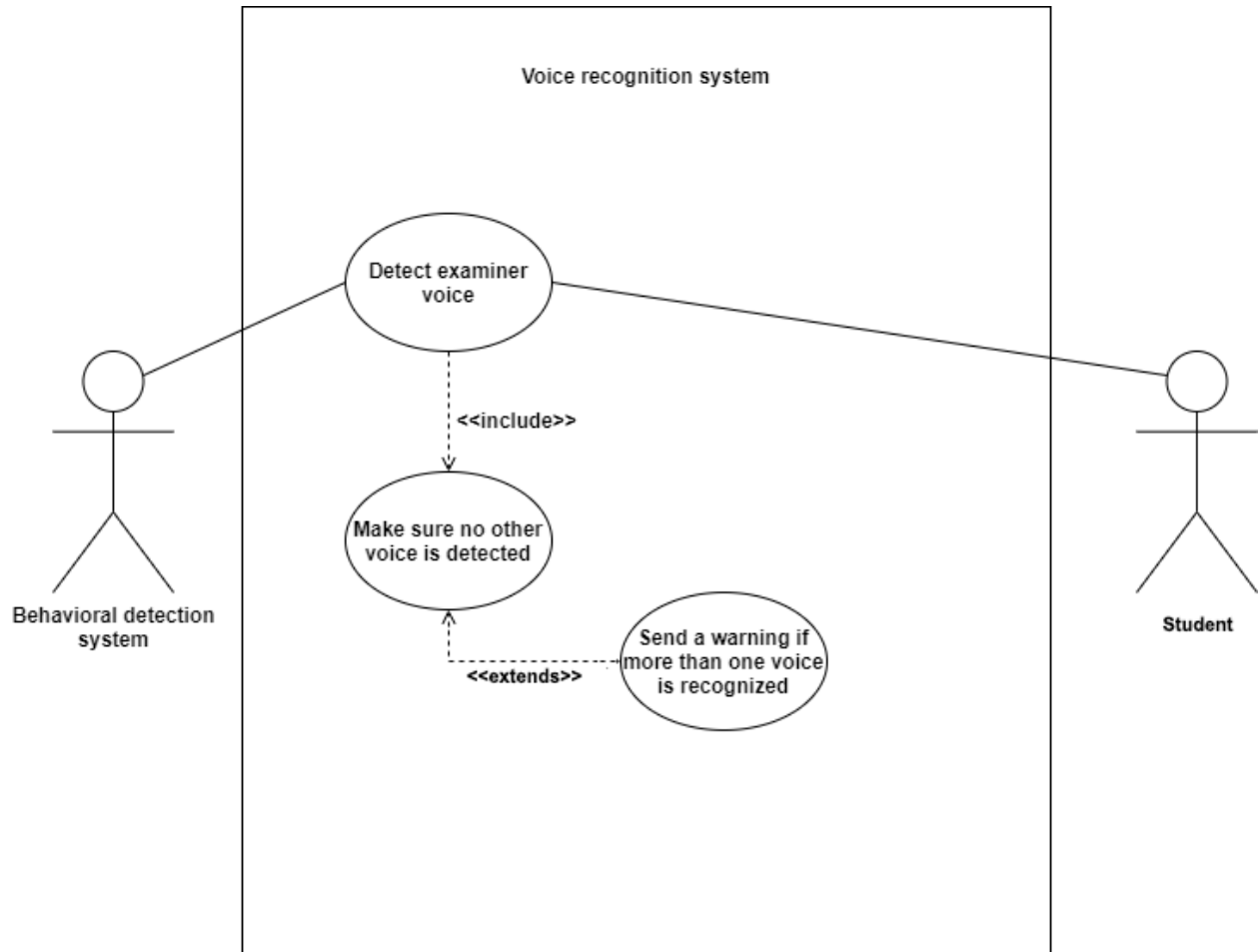


Figure 9: Voice Recognition Use Case

Table 22: Voice Recognition

Use Case Element	Description
Use Case Number	7
Use Case Name	Voice recognition
Use Case Description	The system should make sure that no one is talking to the student during the exam, as this is considered cheating.
Primary Actor	student
Basic Flow	No one is talking in the room.

Alternate Flow	If the system detected more than one voice it will send a warning to the proctor telling him there may be a fraud attempt.
----------------	--

**4.1.1.7: Storing Logs**

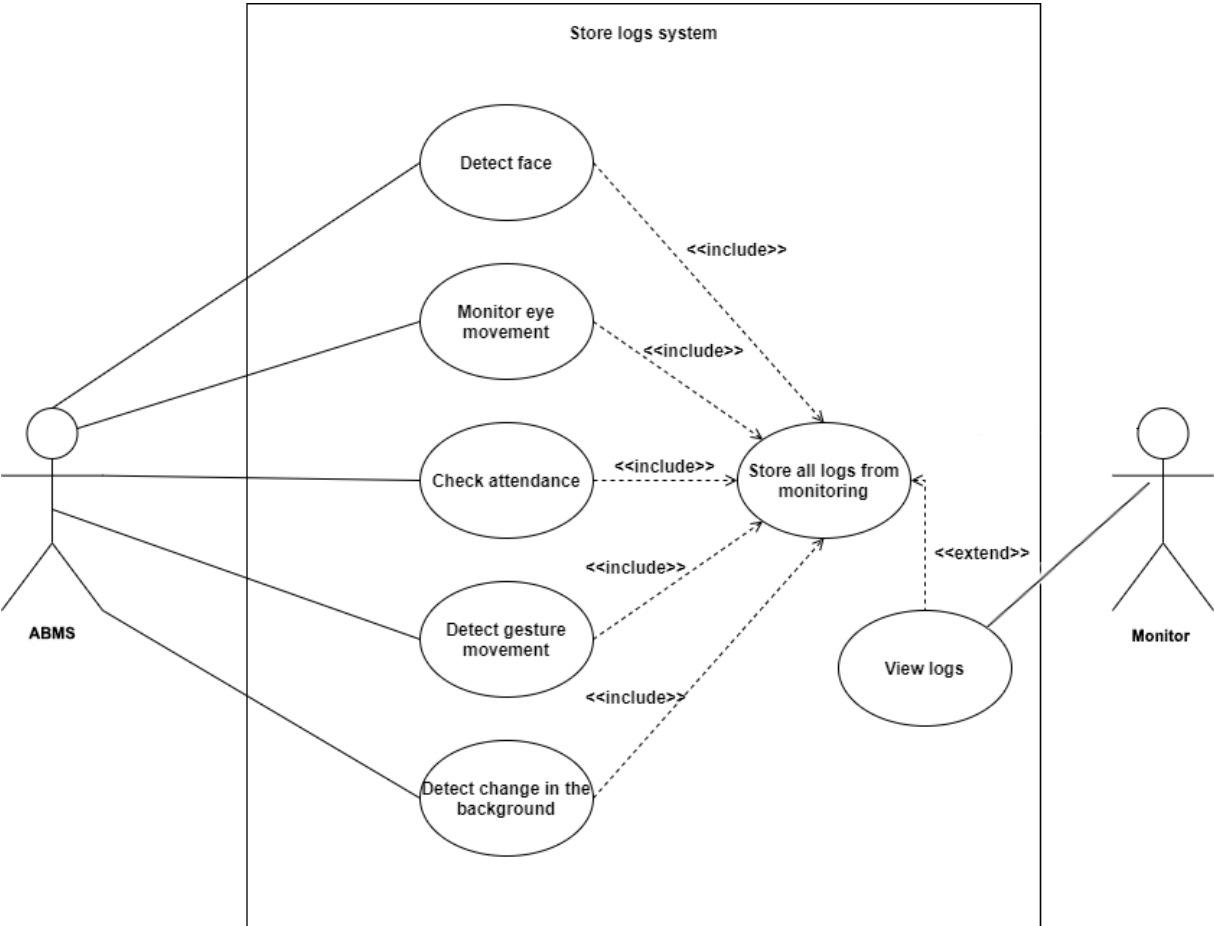


Figure 10:Store Logs Use Case

Table 23:Storing Logs

Use Case Element	Description
Use Case Number	8
Use Case Name	Store logs
Use Case Description	The system should monitor eye movement, face detection, any change in the background, detect gesture movement and check

	attendance. Then store all these actions in the database for any future use.
Primary Actor	Student
Basic Flow	Logs will be stored and can be viewed only by authorized people.
Alternate Flow	None.

**4.1.1.8: Sign In Requirements**

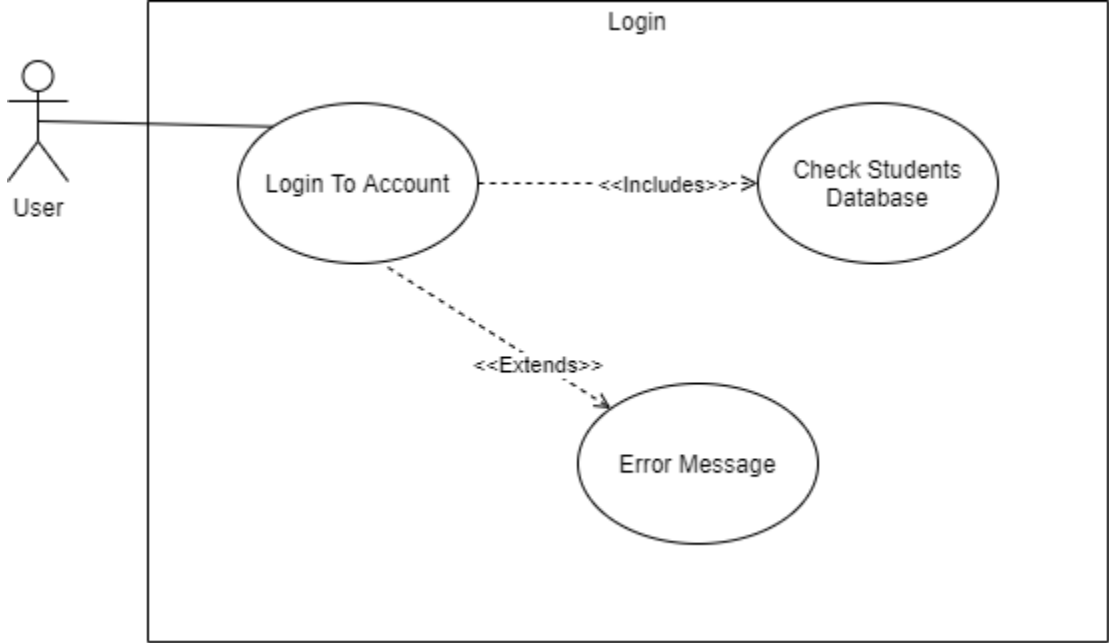


Figure 11: Sign In Use Case

Table 24: Login

Use Case Element	Description
Use Case Number:	9
Application Use Case Name:	Login
Use Case Description:	Each user will be able to login using their unique student ID and password.
Primary Actor:	User.

<b>Basic Flow:</b>	Users open the login page, enter their login information, once the verification process ends, the user returns to the main menu with the right permissions.
<b>Alternate Flow:</b>	The user enters false login credentials and an error message is shown.

**4.1.1.9: Sign Up Requirements**

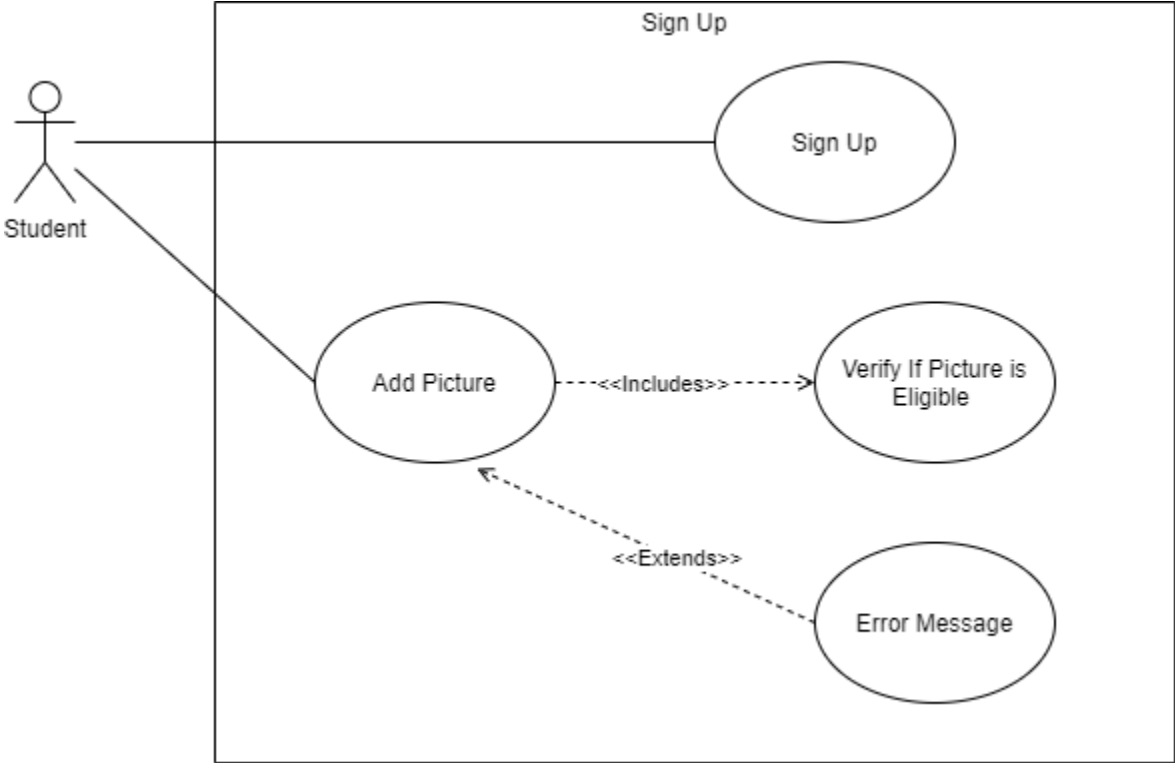


Figure 12: Sign Up Use Case

Table 25: Signup

Use Case Element	Description
<b>Use Case Number:</b>	10
<b>Application Use Case Name:</b>	Sign Up

<b>Use Case Description:</b>	Each student will be able to sign up and add their picture for future use in facial recognition.
<b>Primary Actor:</b>	Student
<b>Basic Flow:</b>	Student opens the sign up page, fills information and uploads their picture which will be stored in the database.
<b>Alternate Flow:</b>	The Student enters a false picture that fails the facial recognition algorithm test and is shown an error message.

**4.1.1.10: Forget Password**

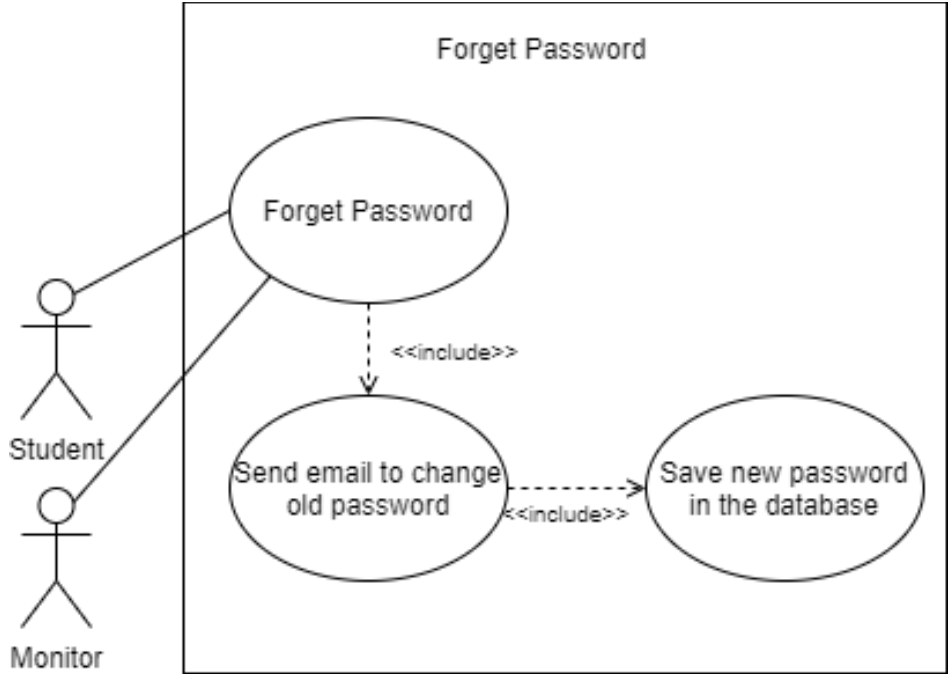


Figure 13:Forget password use case

Table 26:Forget Password

Use Case Element	Description
<b>Use Case Number:</b>	16
<b>Application Use Case Name:</b>	Forget Password

<b>Use Case Description:</b>	In case of a forgotten password, the user will be able to click on the forget password button, which will cause the system to send them an email to reset their password.
<b>Primary Actor:</b>	Student , Monitor.
<b>Basic Flow:</b>	User Can't remember the password, clicks forgot password and an email will be sent to enter the new password.
<b>Alternate Flow:</b>	User opens the login page and enters the correct login credentials.

### 4.1.2 Sequence Diagrams

#### 4.1.2.1 Facial Recognition

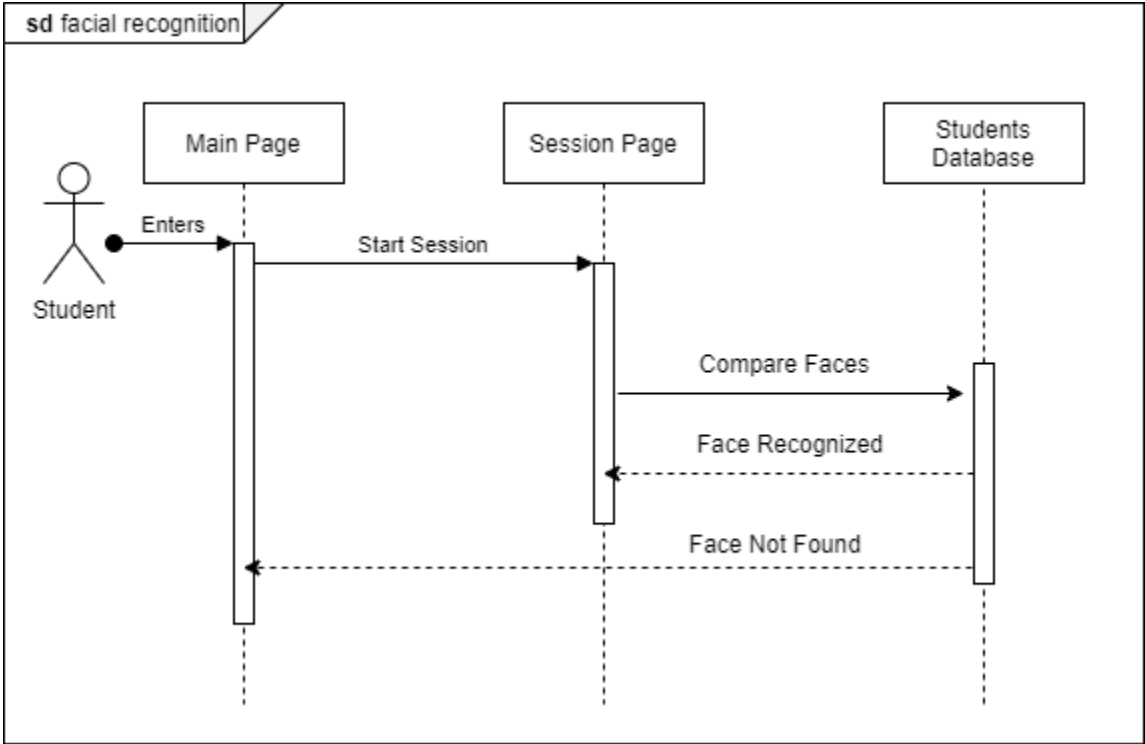


Figure 14: Facial Recognition Sequence Diagram

The student enters the lecture or exam through the main page, once in the session page a screenshot will be taken by the webcam and sent to an algorithm, which will compare the students face to the registered face in the database, if the same identity was confirmed the student will be able to enter the session, if not they will be taken to the homepage.

### 4.1.2.2 Screenshot taking

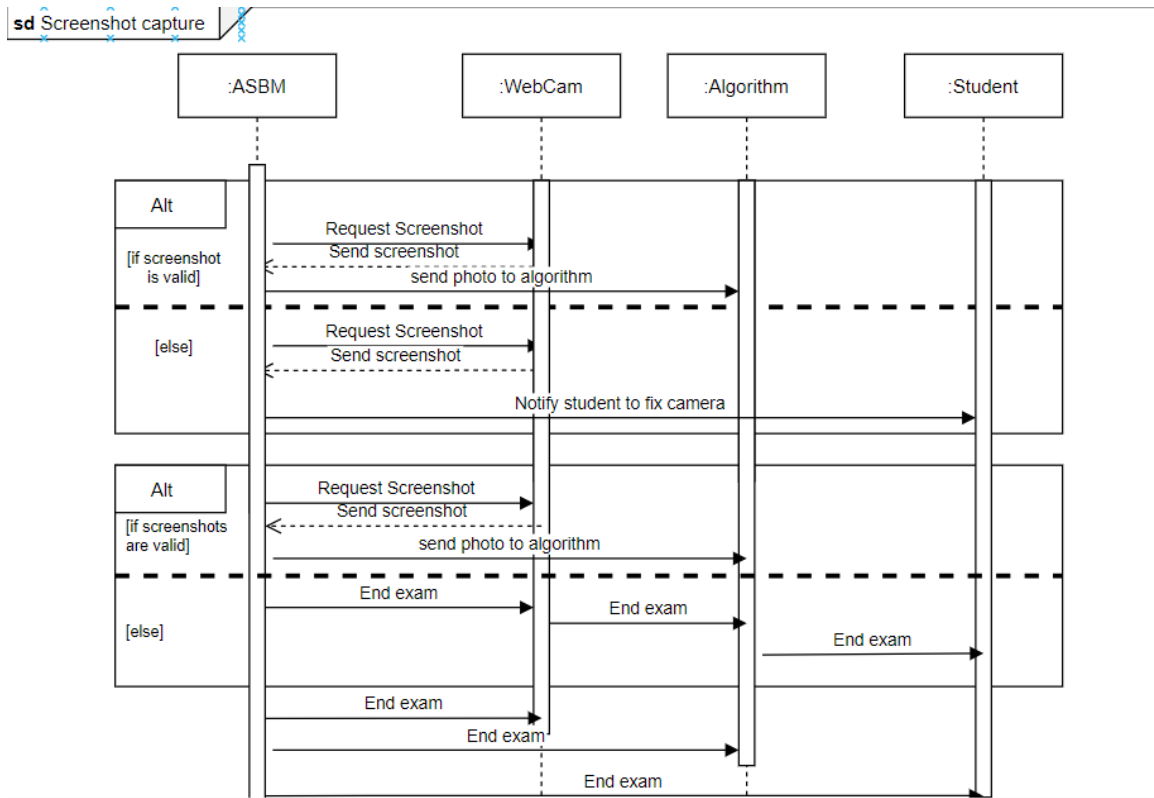


Figure 15: Screenshot Taking Sequence Diagram



When the session starts the webcam will start and run, it will take a screenshot every random interval of time to be processed by the cheating detection algorithms, if the webcam was not able to take the screenshot a notification will be sent to the student to fix it, if the problem is not fixed the exam will shut down.

### 4.1.2.3 Eye Movement Analysis

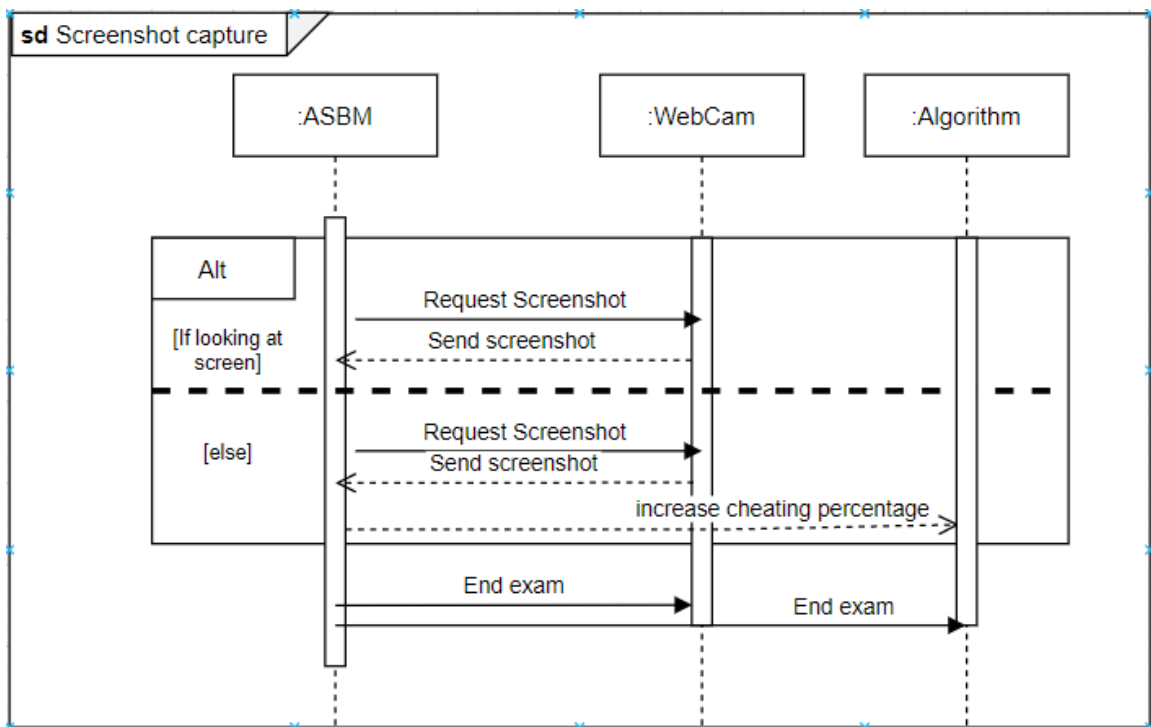


Figure 16: Eye-Movement Detection Sequence Diagram

While the student is taking the exam, the camera will take a screenshot every interval of time randomly to analyze the student's eye movement. If the screenshot is valid it will be sent to the image processing algorithm for processing, Otherwise the system will retake a clear screenshot and also send it to the image processing algorithm. If the student was not looking at the screen it will increase the possibility of cheating and if exceeded a specified limit it will send a warning to the proctor and the student and the exam will be ended.

#### 4.1.2.4 Object Detection

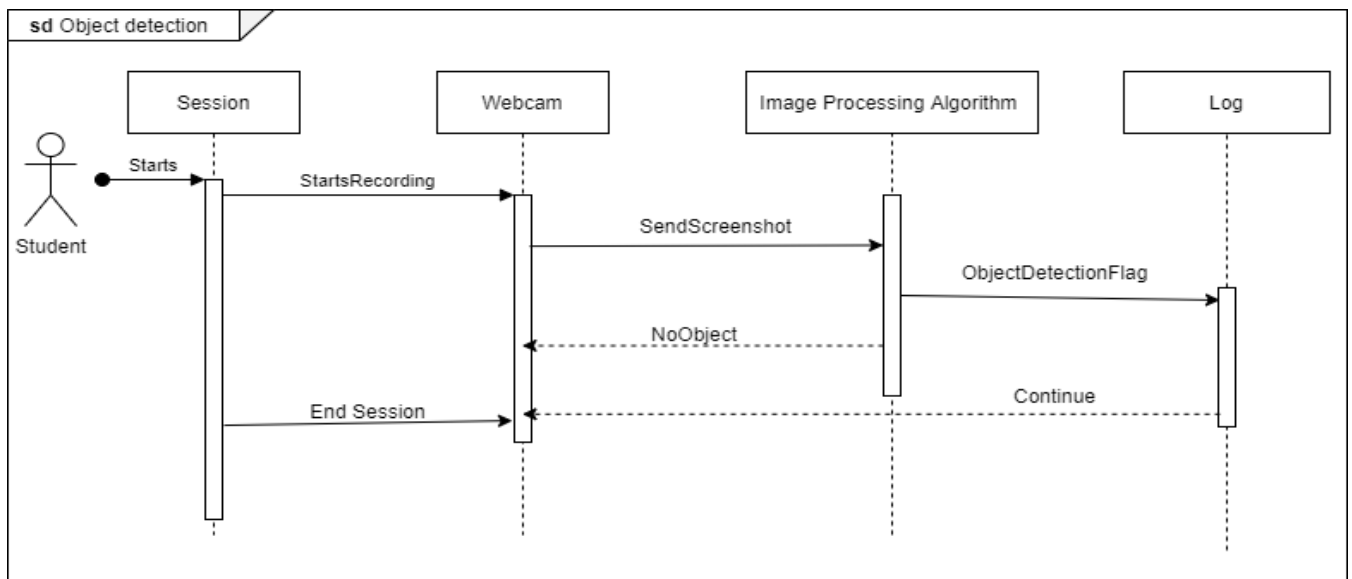


Figure 17: Object Detection Sequence Diagram

The student enters a session and starts the exam, the webcam will take a screenshot every interval of time randomly and send this screenshot to the image processing algorithm for analysis.

If some object is detected like a smartphone, a flag will be raised and both the monitor and student will be notified. Otherwise the student will continue the exam normally and nothing will change. Any detection of an object will be stored in a log for future use by the authorized people.

#### **4.1.2.5 Gesture Detection**

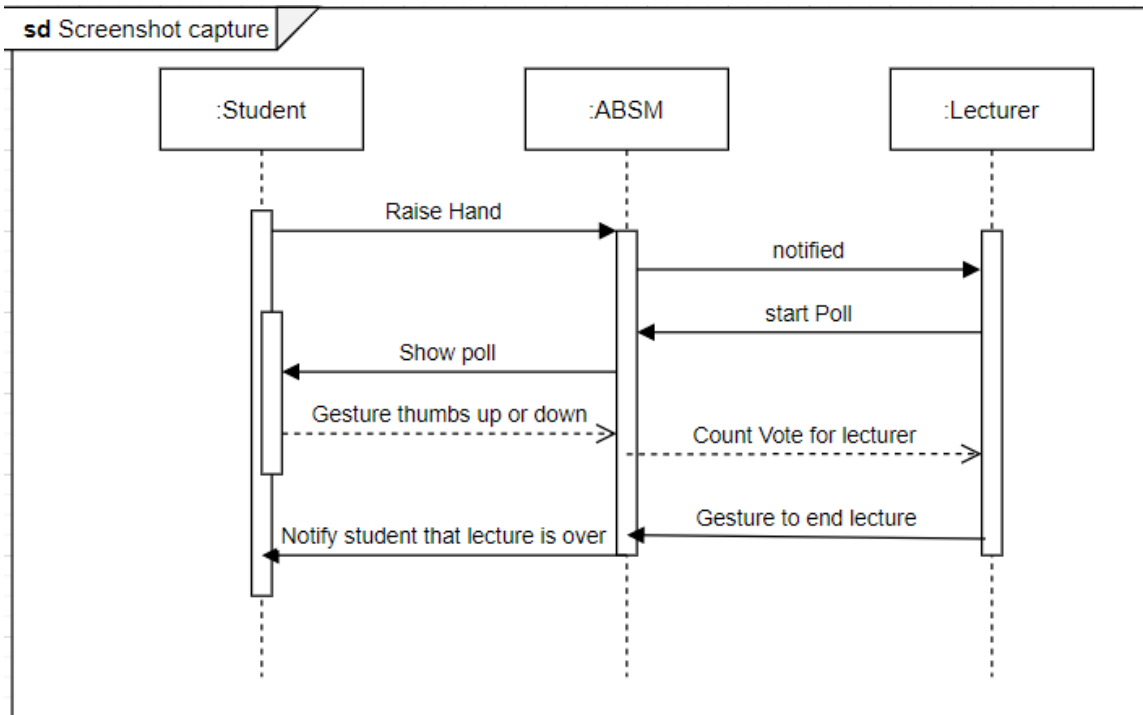


Figure 18:Gesture Detection Sequence Diagram

When the system detects a student raising their hand it will notify the lecturer, also the lecturer can start a poll and students interact with it by thumbs up or down gesture. Another thing the lecturer can do is to end the lecture by a certain gesture.

#### 4.1.2.6 Calculate Cheating Possibility

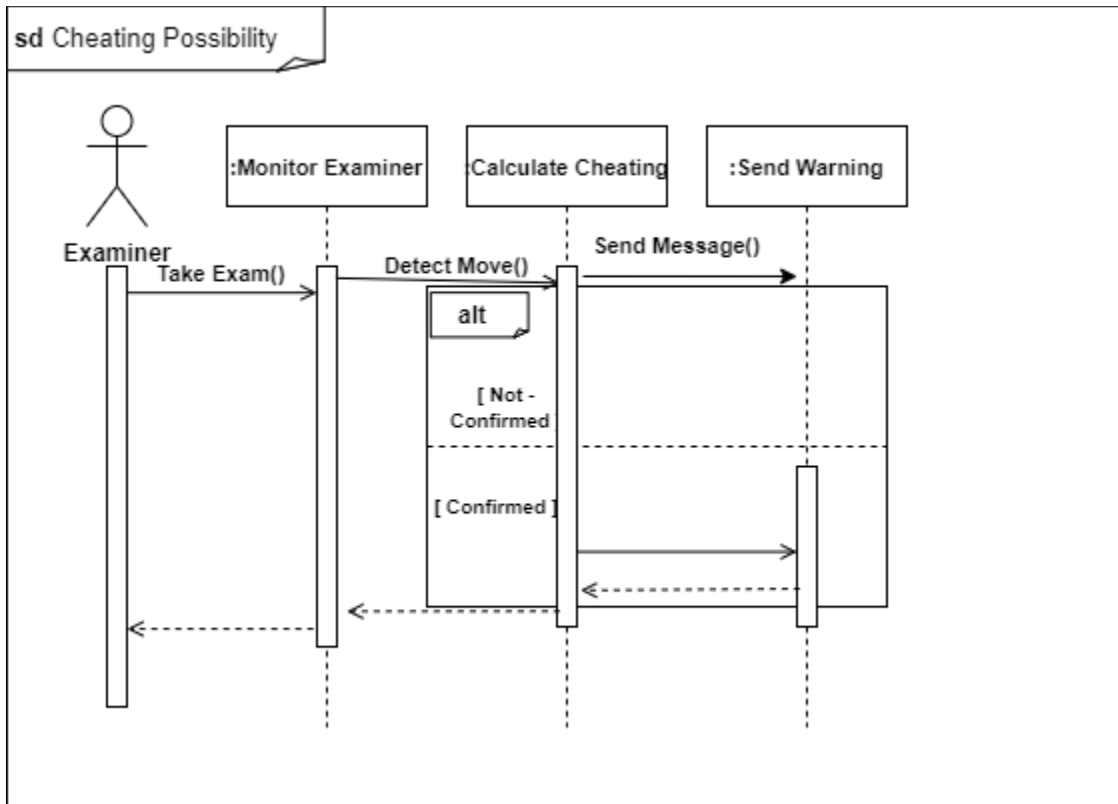


Figure 19: Calculate Cheating Possibility Sequence Diagram

When the student starts the exam, the system will start detecting several factors like eye movement, facial recognition and room scan, then the system will calculate the probability of cheating using complex math. If the probability exceeds a certain percentage it will be considered cheating and both the student and the proctor will be informed to take an action.

#### 4.1.2.7 Audio Analysis

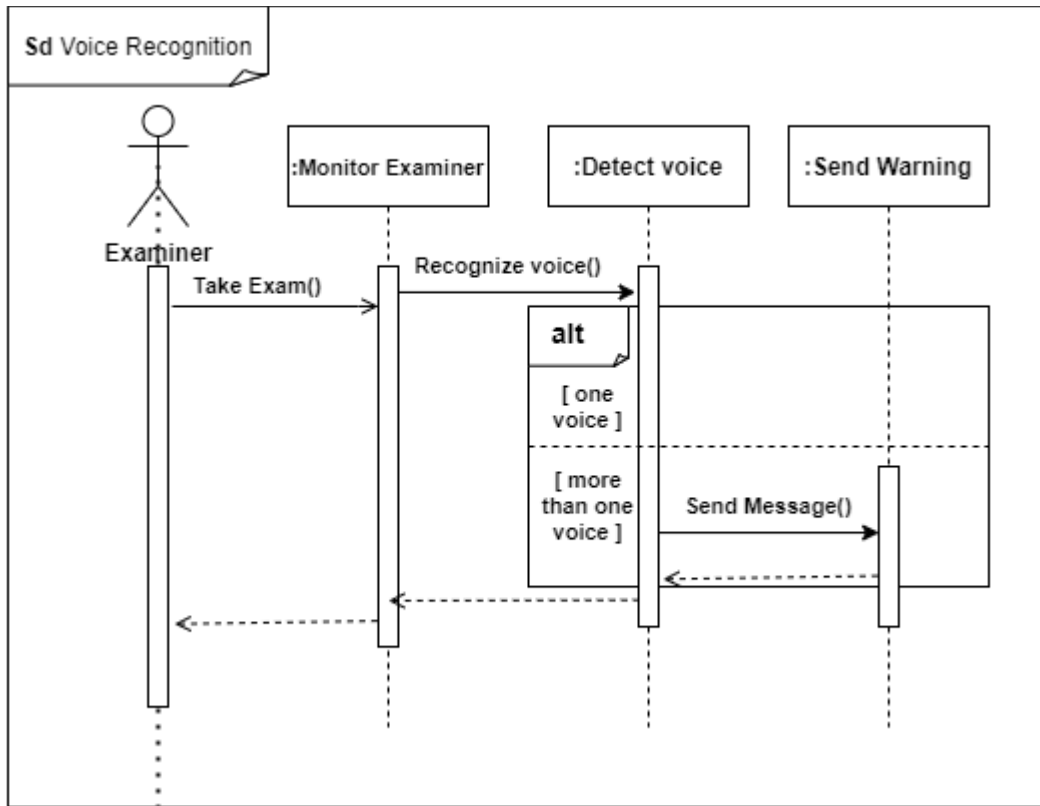


Figure 20:Voice Analysis Sequence Diagram

While the student is taking the exam, if the voice reaches a level where it can be recognized by the system through the microphone locally, it will make sure only one human voice is detected, otherwise it will be considered cheating and the system will send a warning to the student, proctor and store it in the log.

#### 4.1.2.8 Storing Logs

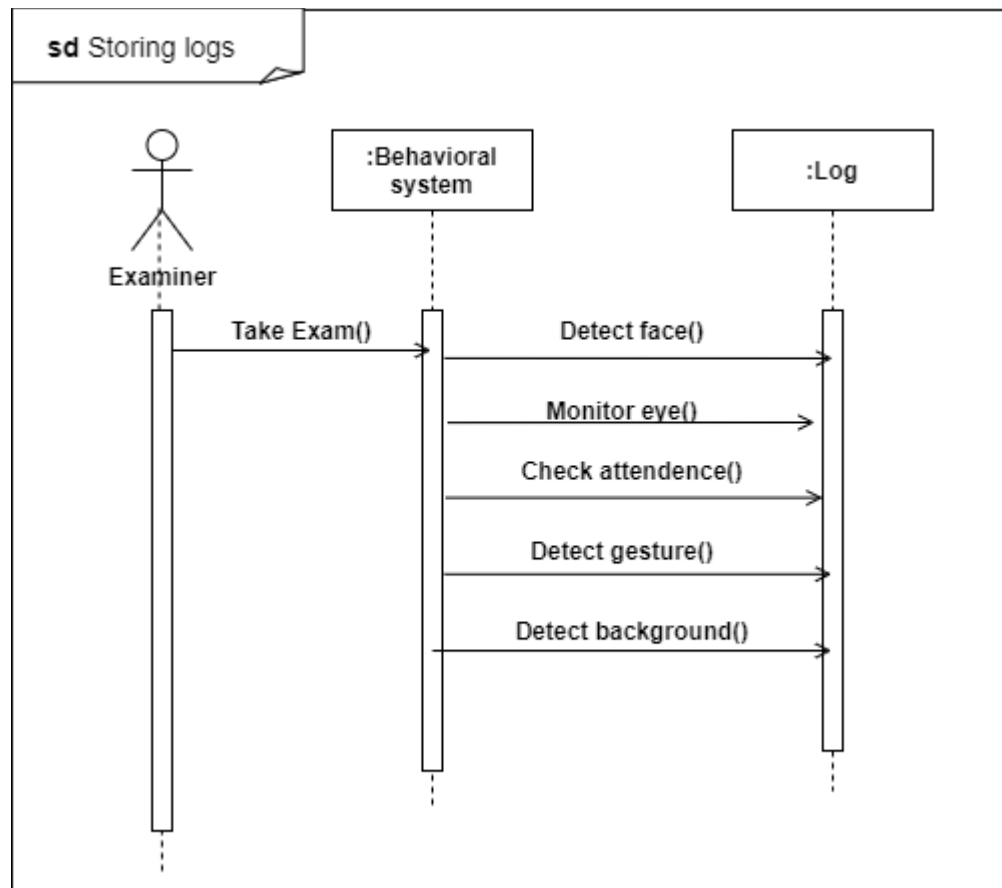


Figure 21: Storing Logs Sequence Diagram

When the student enters the exam, all detections like face recognition, eye movement, attendance check, gesture moves and changes in background will be stored in a log that can be used in future. This log can be accessed by authorized people who can take action or confirm the detections.

### 4.1.2.9 Signing In

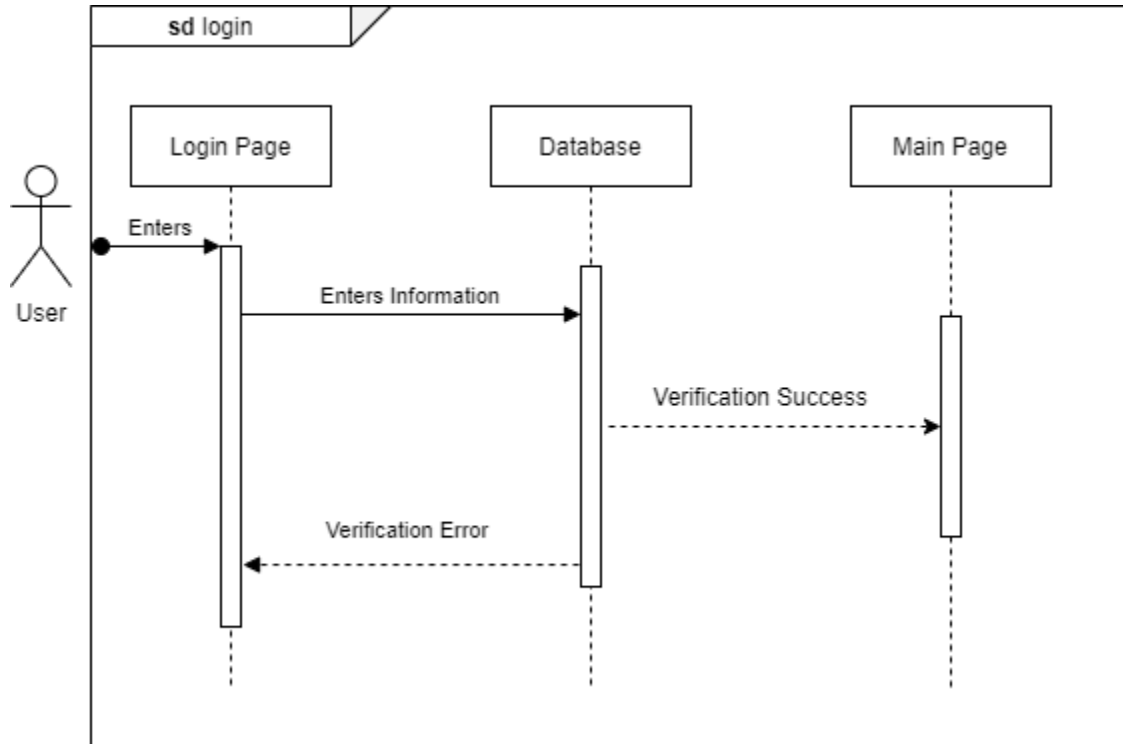


Figure 22: Signing In Sequence Diagram

When the student signs in they enter their information which then is verified by the database, if the verification was confirmed the user then is taken to the homepage, if not, an error message will be shown.



#### 4.1.2.10 Signing Up

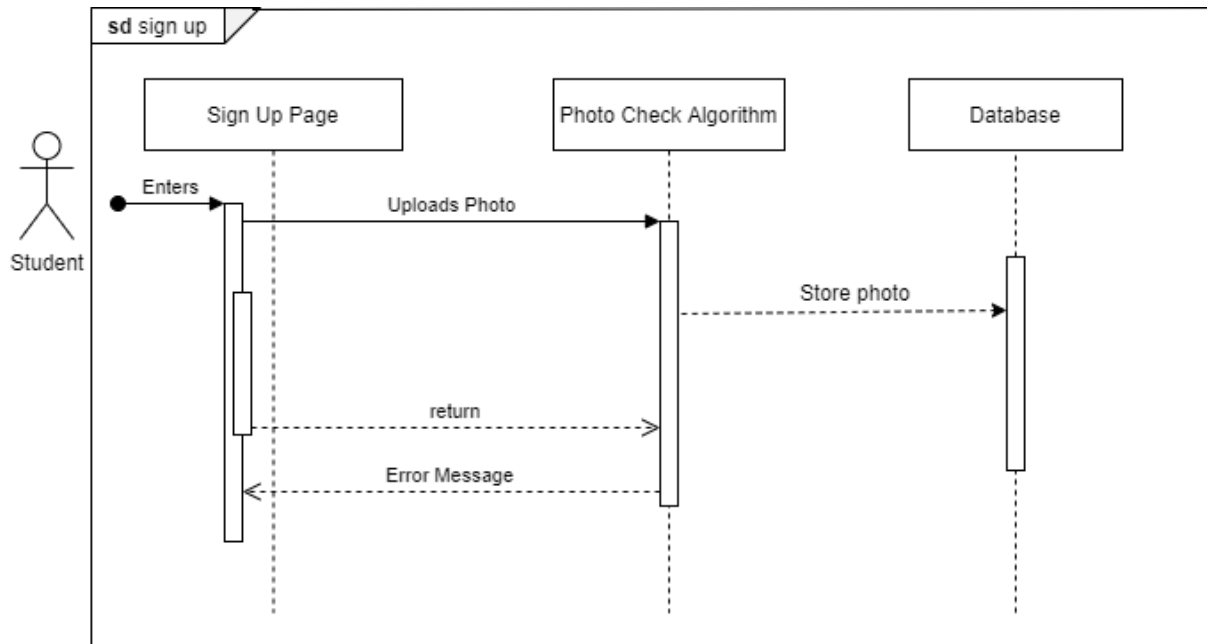


Figure 23: Signing Up Sequence Diagram

When the student signs up they fill their information and upload their picture which is supposed to be used in the facial recognition later on, once the photo is uploaded it will go through an algorithm that verifies whether this picture can or cannot be used in facial comparison later on, if it is eligible it will be stored in the database, if not, an error message will be shown and the student can upload another picture.

### 4.1.2.11 Forget Password

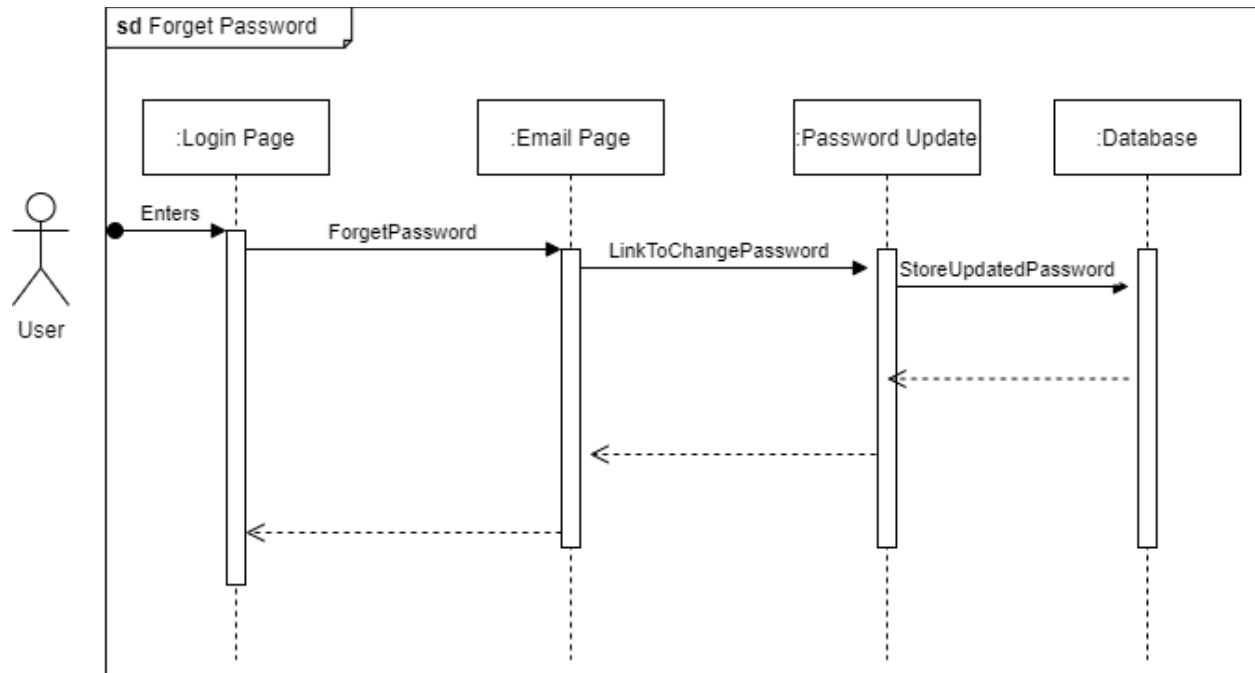


Figure 24:Forget password sequence diagram

User can be a student or a monitor who forgot their password. The above sequence diagram shows that the user enters email and a link will be sent to them to update their password and will be stored in the database instead of the old password. The new password is then used to login.

### 4.1.3 Class Diagram:

The class diagram visualizes the relationship between the different classes in our system, it is shown next.

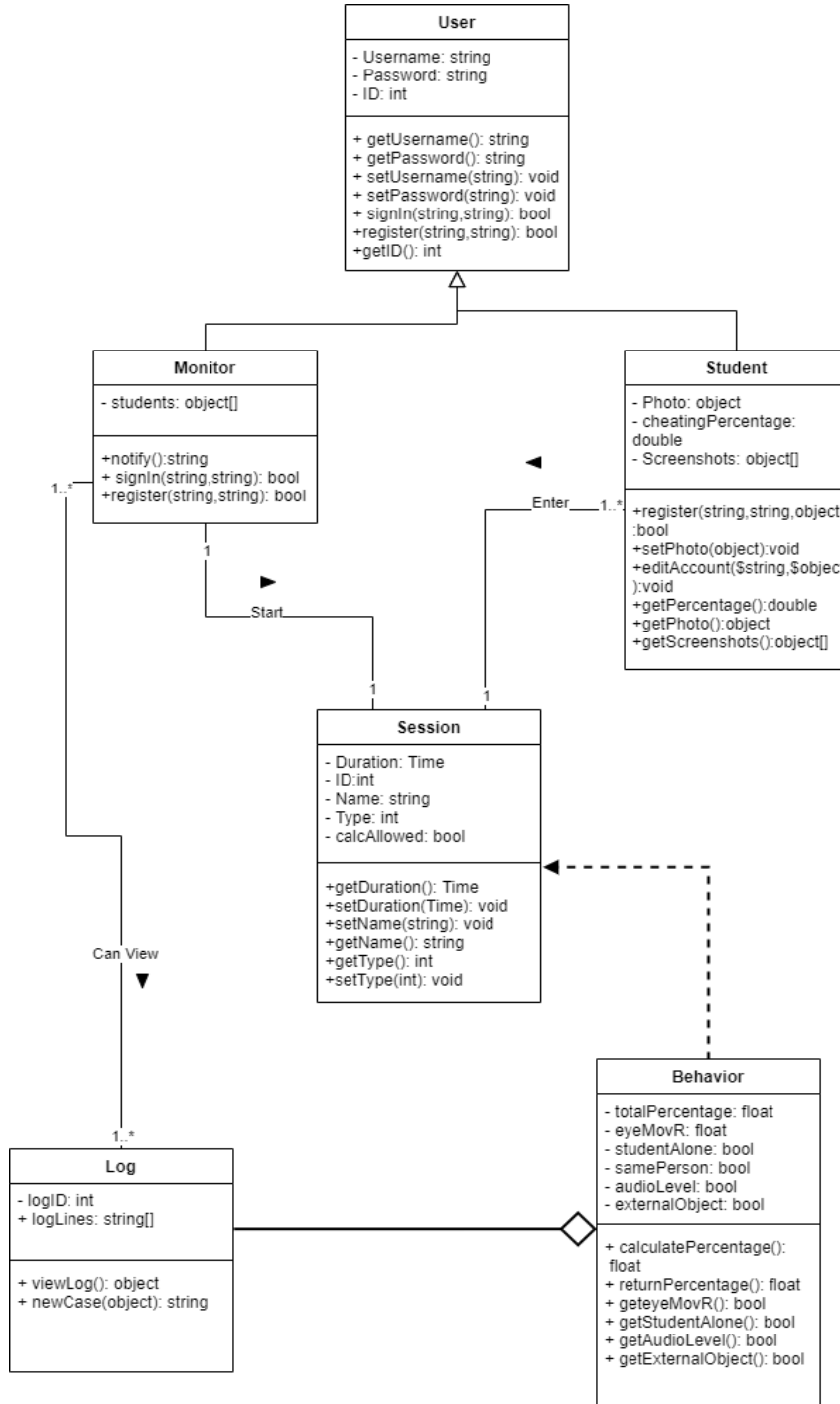


Figure 25: Class Diagram

Our system deals with an academic atmosphere, therefore two types of users are expected to use it, a student and a monitor, both of these users share a few attributes like a name, a username, an ID and a password while the monitor also has an array of the students he's able to monitor and the student has a picture that will be used later in facial recognition. To utilize our system in session each session must have a start and end time which will be used to calculate its total duration and it also has an ID, name, type and calcAllowed value which represents if it is okay to use a calculator or not. We also have a class called Behavior which will implement our detection algorithms such as eye movement, if the student is alone or not, if the student is recognized or not, if the audio levels are accepted and if it is suspected that an external tool is used or not such as a book or a smartphone one class depends on the behavior class which is the log, this class will register everything the behavior class detects in a string of lines that will be used later if needed.

#### 4.1.4 Object Diagram:

An Object Diagram shows an instance of the classes and is shown next.

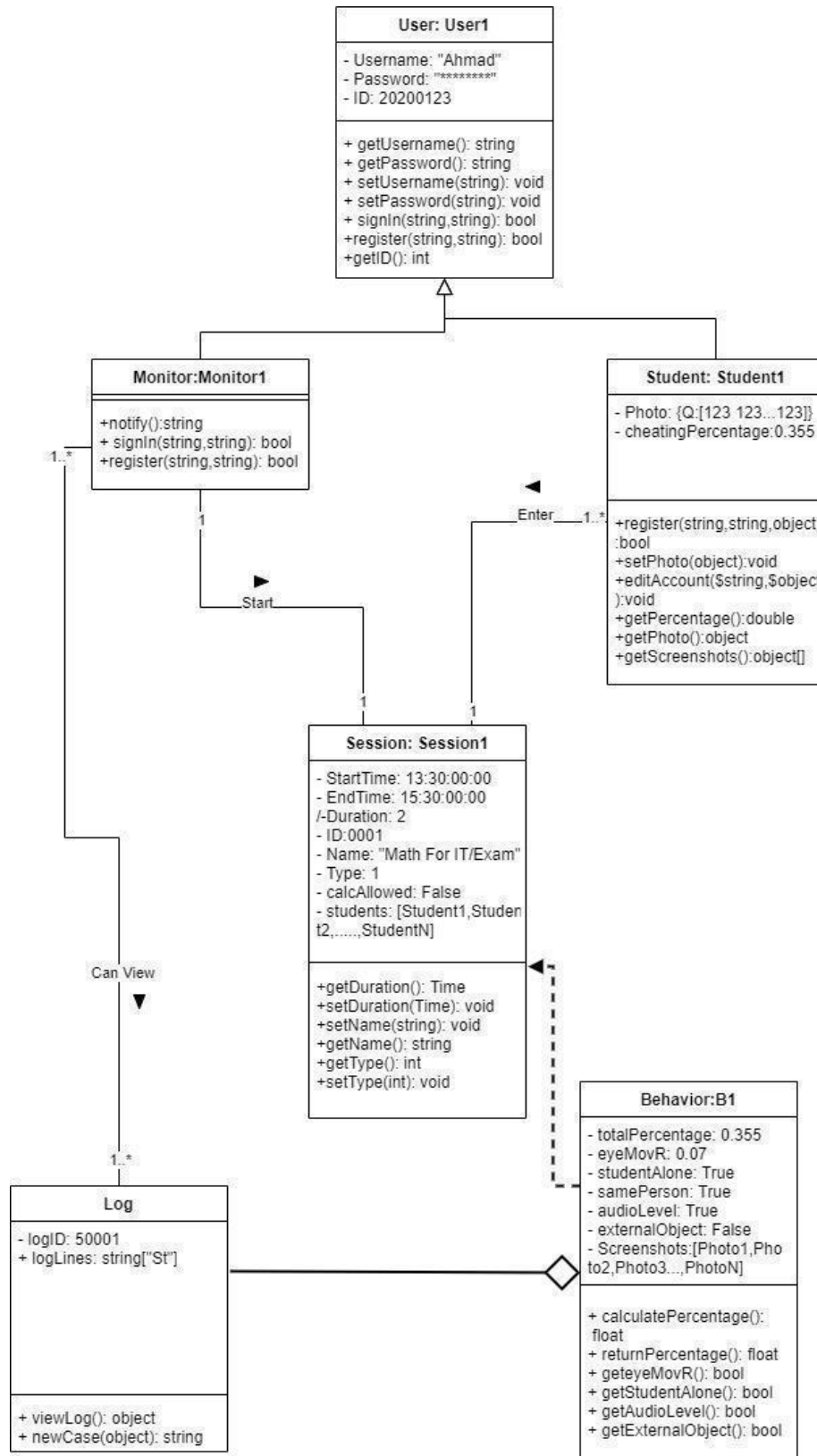
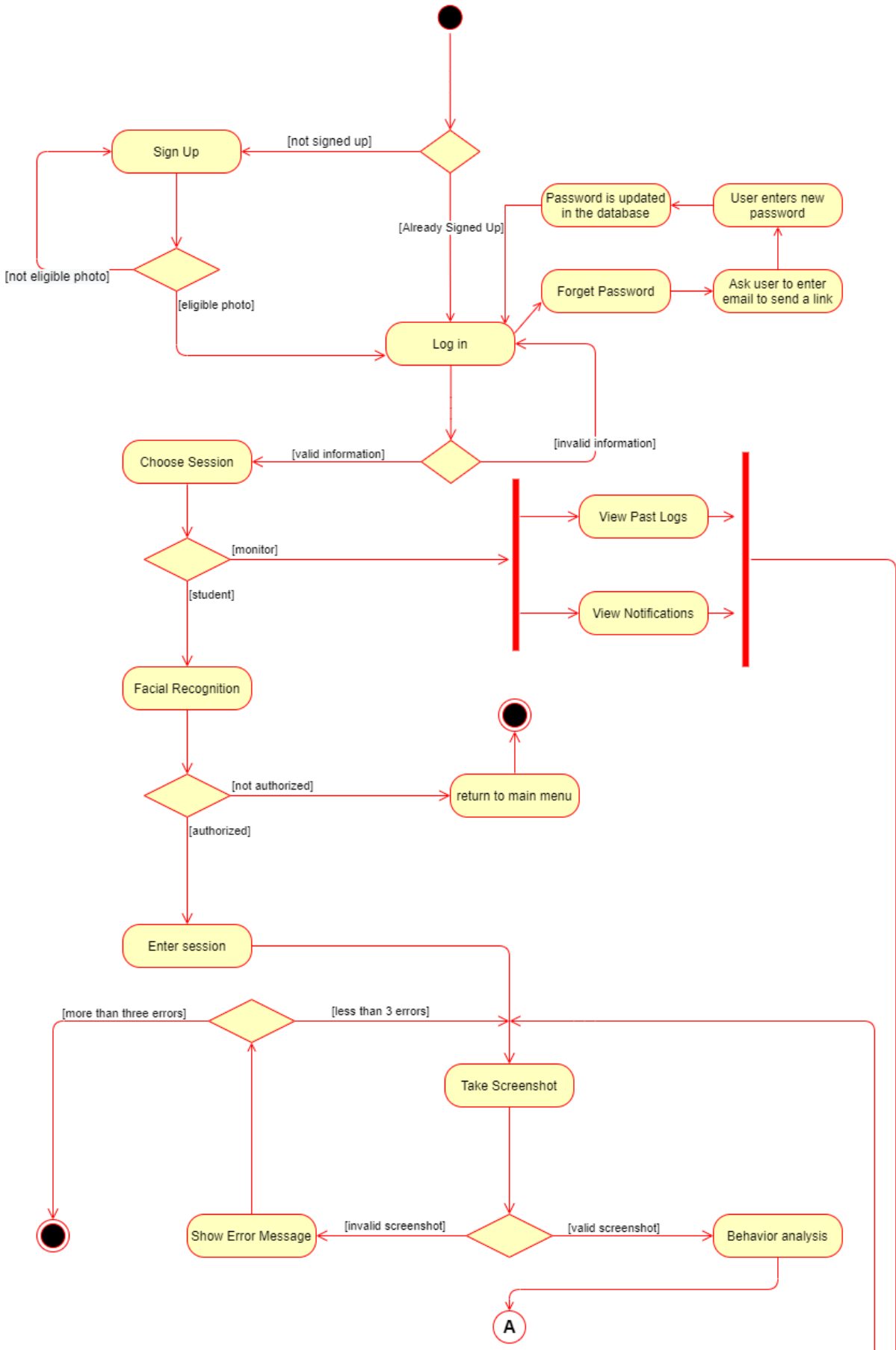


Figure 26: Object Diagram

In the above diagram which is an instance of the class diagram a User called “Ahmad” with an ID of 20200123 is imagined, Ahmad has his photo uploaded as an array of pixels and he is in a session that starts at 13:30 and ends at 15:30 it is of type 1 and called “Math for IT/Exam”. A calculator is not allowed in this session. Ahmad’s total cheating probability is 0.355 he is alone, eye movement ratio is 0.07 and the audio levels are accepted. His logID is 50001 and there’s one line in his log which is “St”.

#### 4.1.5 Activity Diagram

An Activity diagram shows the flow from one activity to the next activity.



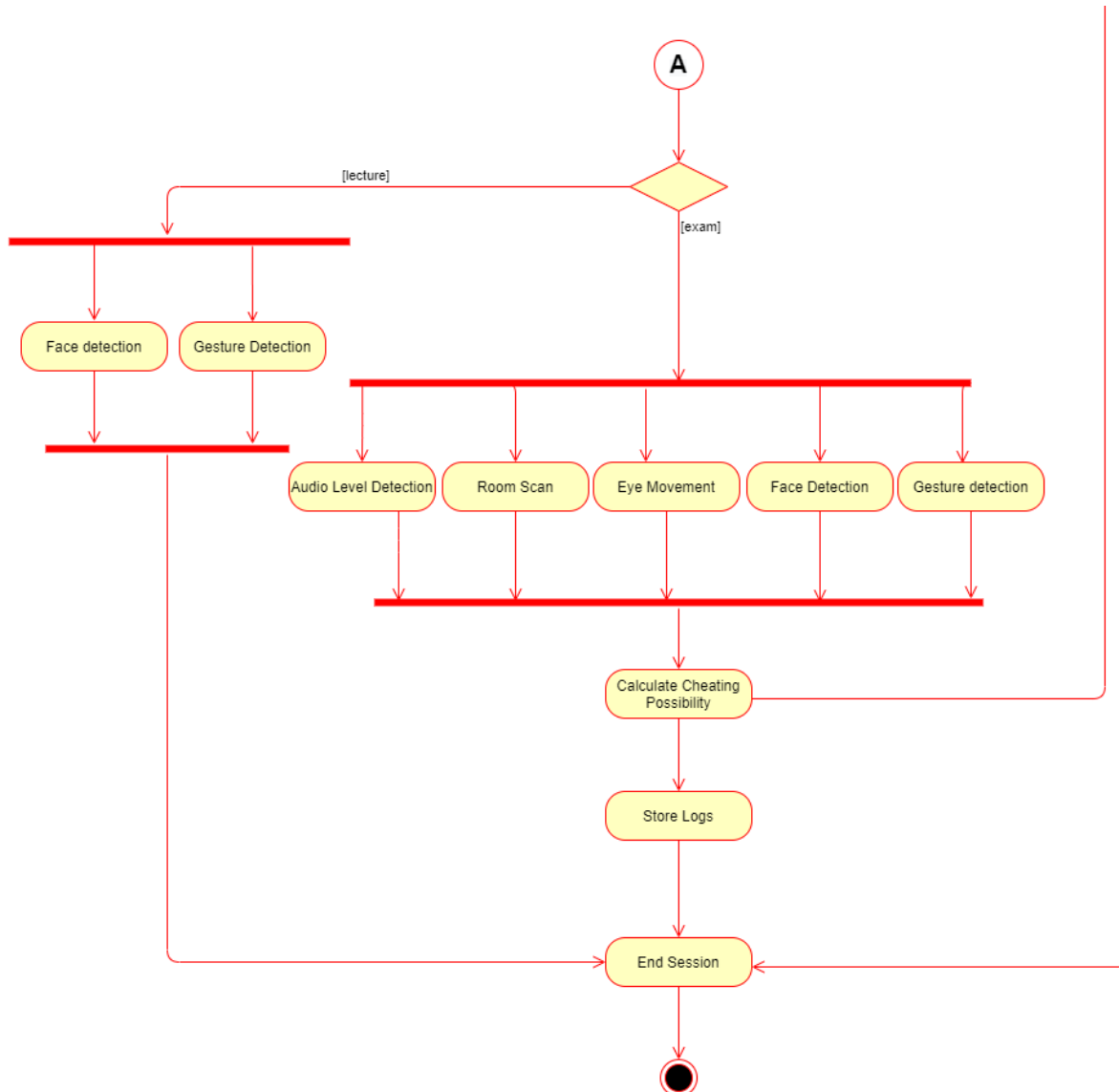


Figure 27:Activity Diagram

An Activity diagram shows the flow from one activity to the next activity. In the case of our project, we start with the not signed in/signed up stage, then we move on to choose a session, and depending on whether the user is a monitor or a student a different path is followed. If the user is a student he undergoes a facial recognition test and enters the session if the face matches the database, after which he is proctored automatically whether in an exam or in a lecture depending on the settings for this session. If it is a monitor he is allowed to view the logs of the session for each student.



#### 4.1.6 Package Diagram

A package diagram is a diagram that shows the packages which the classes were divided between.

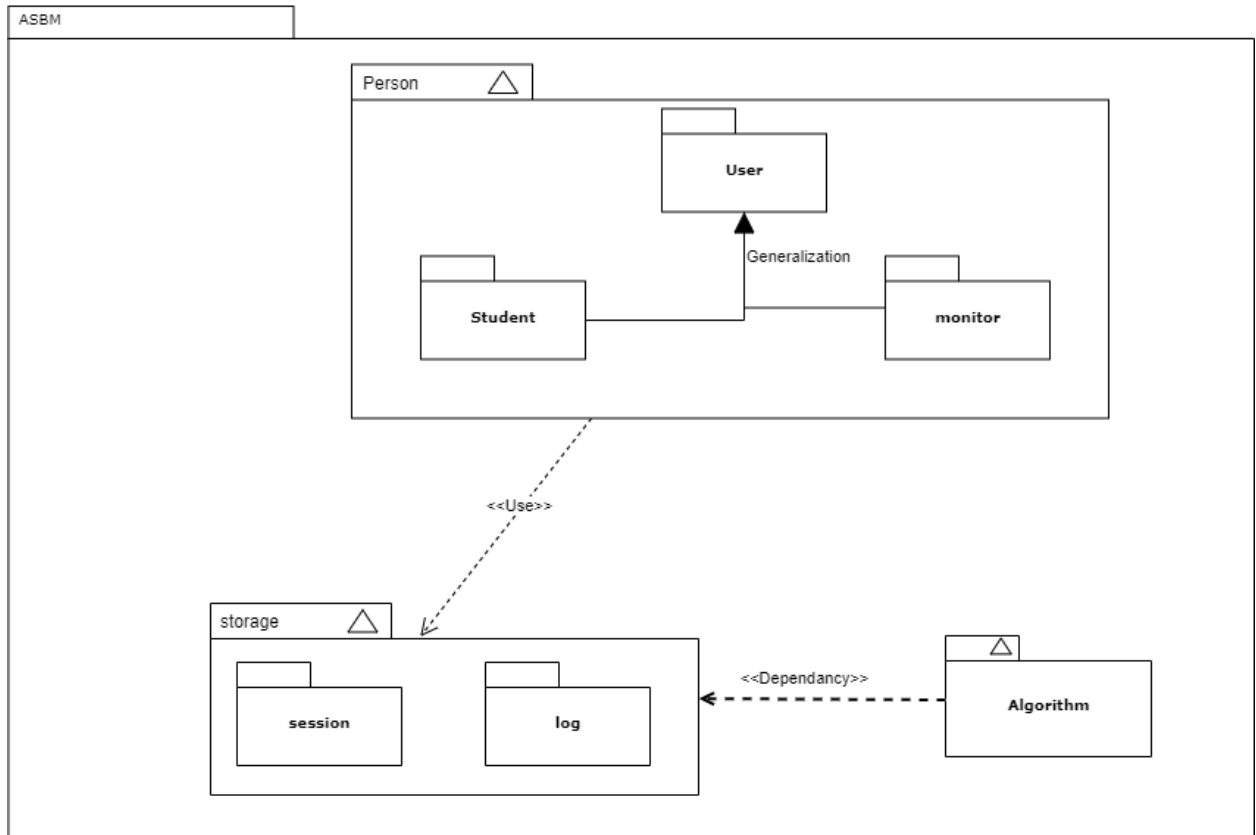


Figure 28:Package Diagram

The Package diagram shows the organization between elements and classes, The student and monitor classes are inherited from class user, These three classes are grouped in person package. Person package is connected to a storage package which contains the session class which can be controlled by the monitor and accessed by the student. The last package contains the algorithm class which will make all the calculations for the cheating possibility and it is dependent on the storage package because all calculations need a session and an exam to be performed.

#### 4.1.7 Component Diagram

A component diagram separates the project into groups of files that share the same theme called a component.

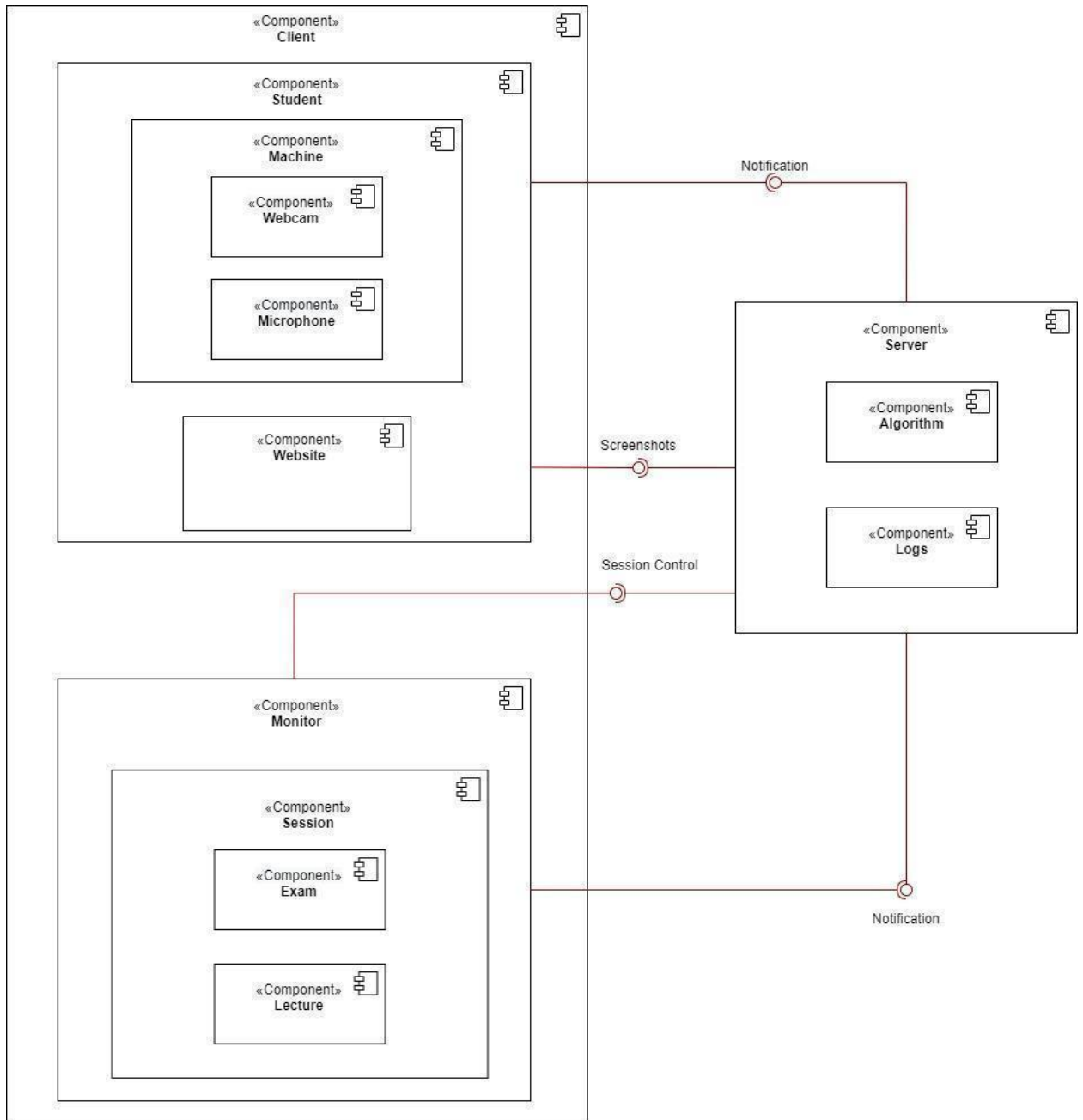


Figure 29:Component Diagram

The component diagram shows in details the components involved in the system, The client component contains both the student and monitor components. The student component is divided into hardware (machine component) that contains webcam and microphone.

The monitor component has the session component that has access on which contains the exam and lecture component.

The monitor has access to the server component to control the session and view logs. Also the server component sends notifications to the monitor and students based on the calculations done by the algorithm.

The screenshot is taken from the student and sent to the algorithm in the server component.

#### 4.1.8 Deployment Diagram

A deployment diagram shows the execution architecture of a system including nodes that represent hardware and software environment and what's connecting them.

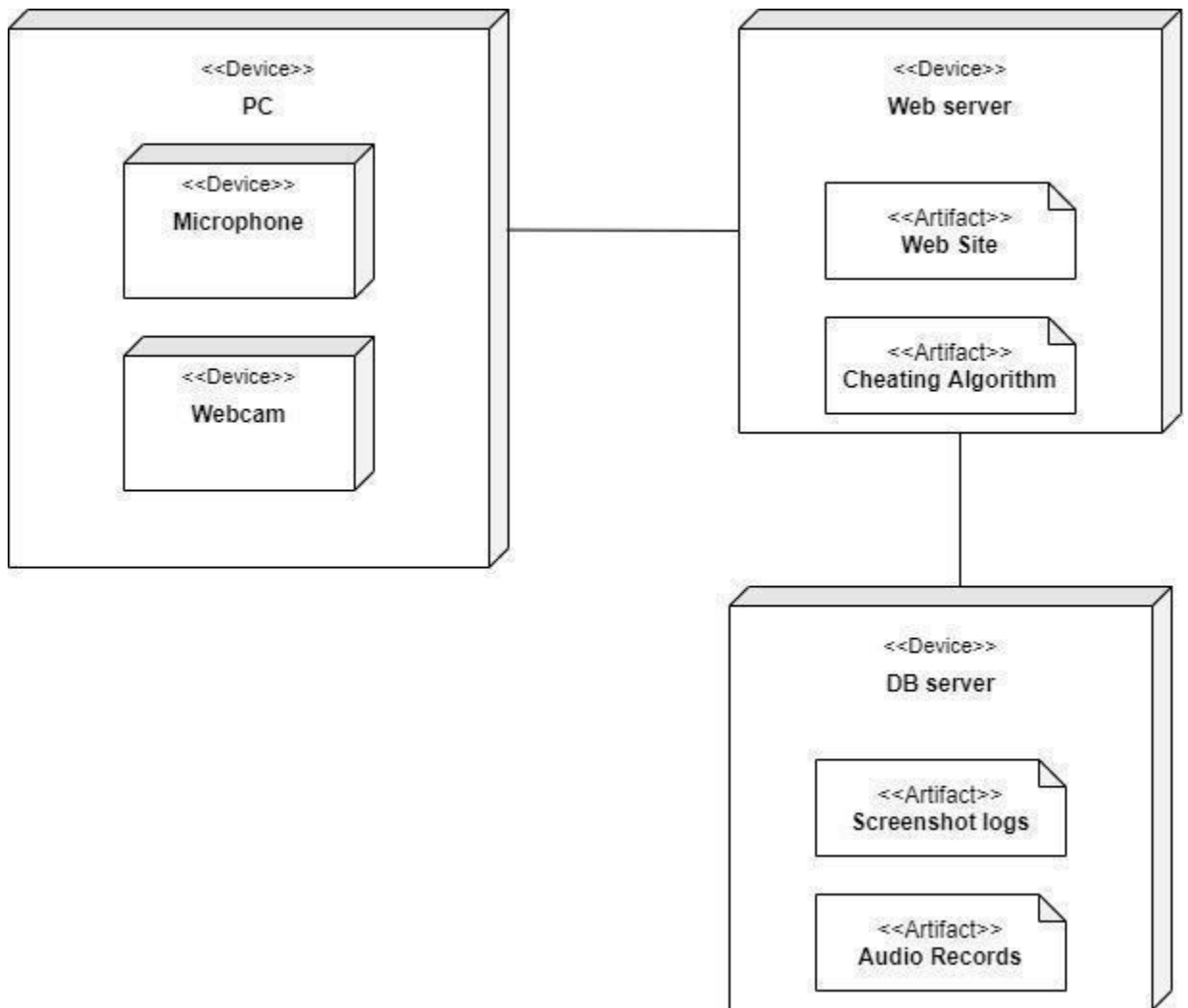


Figure 30:Deployment Diagram

The deployment diagram contains the pc device which has two hardware devices, the webcam and microphone. They are connected to the webserver that has the software part (Website and cheating algorithm). The webserver will use the hardware parts (Webcam for screenshots and microphone for audio records) for analysis through the algorithm and store the results in the database server that can be retrieved in the future.

#### 4.1.9 State-Change Diagram

A state change diagram shows the different steps that a user might take and how it affects their state.

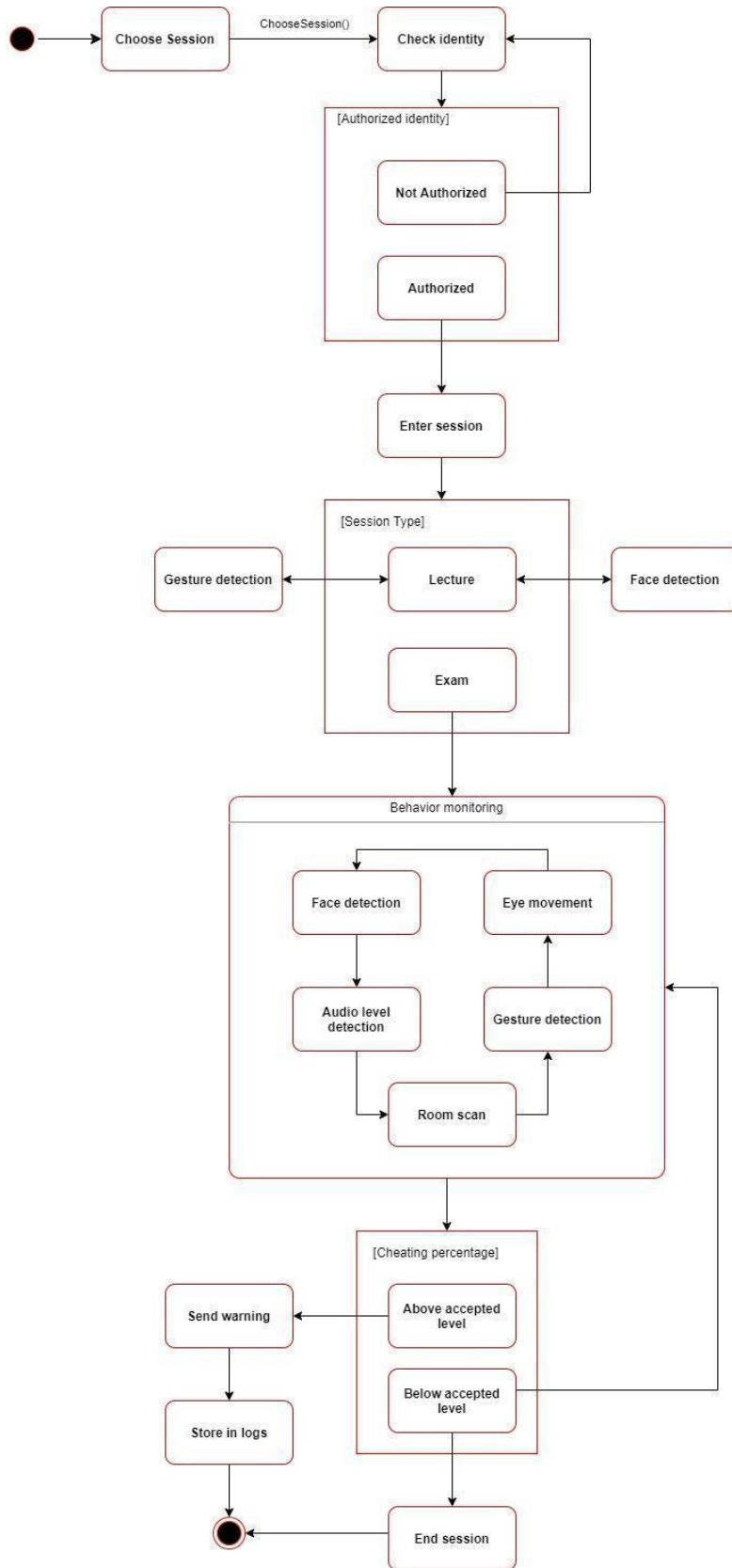


Figure 31: State Change Diagram- Behavior Detection

A state change diagram shows the different steps that a user might take and how it affects their state.

Behavior detection: the user chooses session and is in check identity state then if authorized moves on to the enter session state if not returned check identity state again.

After entering the session he moves on to the next state depending on session type if a lecture moves between gesture and face recognition states, if an exam he enters the behavior detection state. If the cheating percentage raises above normal levels the student enters “send warning” and “store logs”. And finally the session ends ending the state diagram.

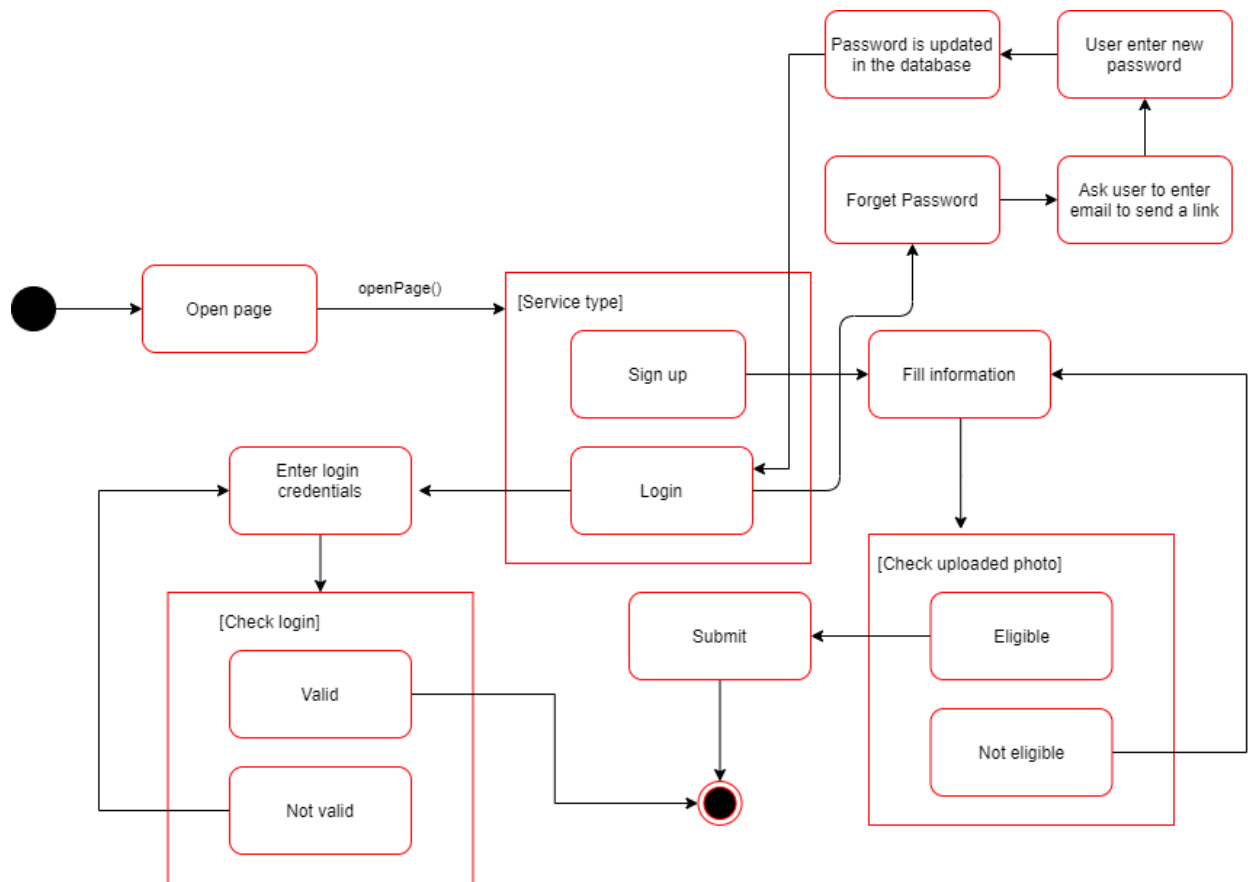


Figure 32: State Change Diagram- Sign-Up and Login

Signup and login: 2 paths in this state diagram; if the user is logged in he enters the enter credentials state if it is valid the diagram ends, if not valid goes back to enter credentials, the second path is if the user wants to sign up. Enters the fill information state if the information is eligible enter the signed in state, if not goes back to fill information state. If the user forgot the password for login he can create a new password from the forget password option and the new password will be used for login.

## 4.2 Physical Model Design

### 4.2.1 ERD Diagrams

Note: we did normalization in the following matter:

#### -0NF:

User (Username, Password, Photo, cheatingPercentage)

Session(Duration,StartTime,EndTime,Name,Type,calcAllowed,Students,LogLines,totalPercentage,eyeMovR,studentAlone,samePerson,audioLevel,externalObject,Screenshots)

#### -1NF: Add Primary Keys

User (ID, Username, Password, Photo, cheatingPercentage)

Session(SessionID,Duration,StartTime,EndTime,Name,Type,calcAllowed,Students,LogLines,totalPercentage,eyeMovR,studentAlone,samePerson,audioLevel,externalObject,Screenshots)

#### -2NF: Remove multivalued and columns that are frequently null (photo, cheatingpercentage)

User (ID, Username, Password, Photo, cheatingPercentage)

Student (SID, Photo, CheatingPercentage, UID)

Monitor (MID, SID)

Session(SessionID,Duration,StartTime,EndTime,Name,Type,calcAllowed,Students,LogLines,totalPercentage,eyeMovR,studentAlone,samePerson,audioLevel,externalObject,Screenshots)

#### -3NF: Remove Transitive Dependency

User (ID, Username, Password, Photo, cheatingPercentage)

Student (SID, Photo, CheatingPercentage, UID)

Monitor (MID, SID)

Session (SessionID, Duration, StartTime, EndTime, Name, Type, calcAllowed, Students, LogID)

Log (LogID,LogLines)

Behavior (SessionID, totalPercentage, eyeMovR, studentAlone, samePerson, audioLevel, externalObject, Screenshots)

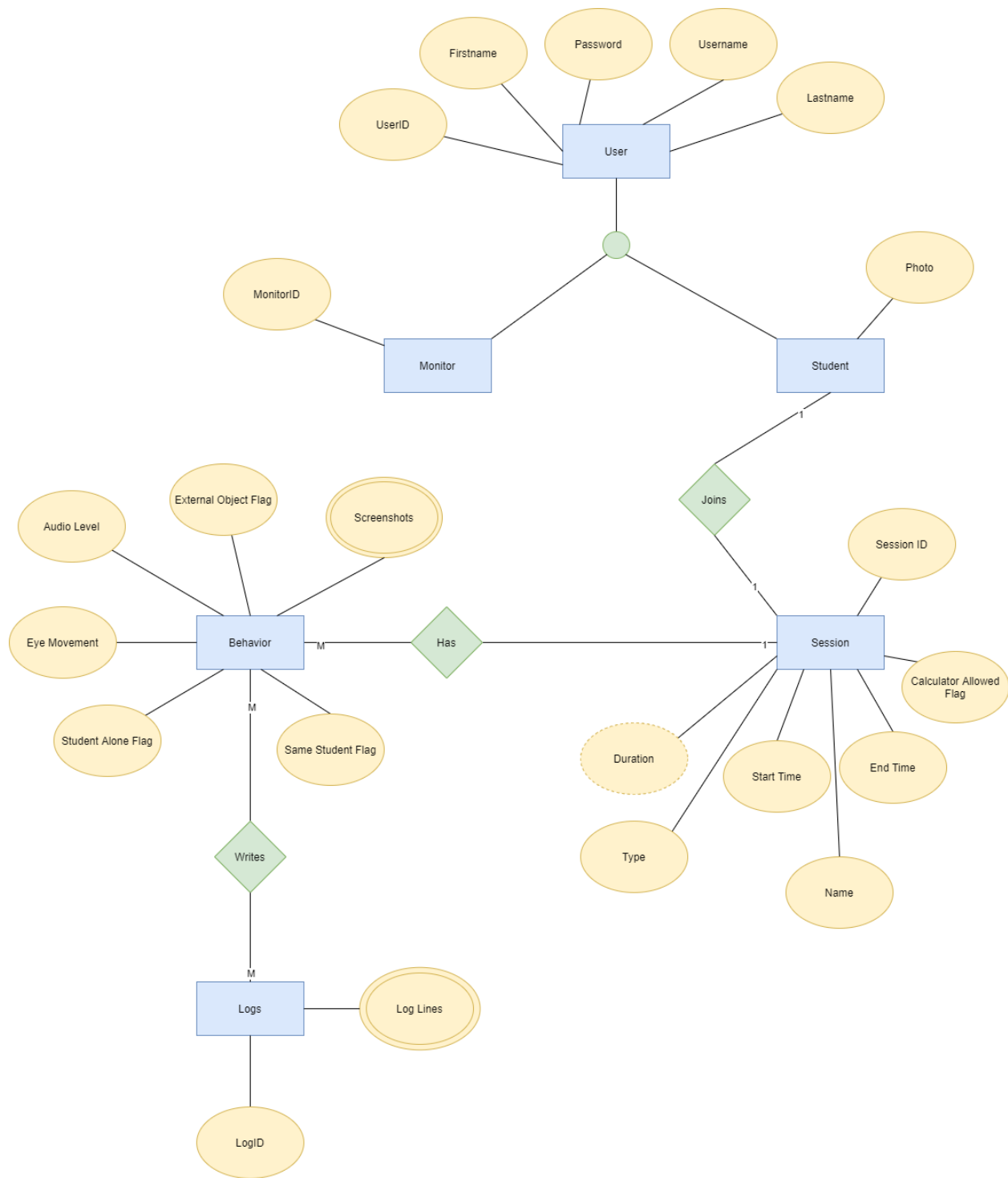


Figure 33:ER Model





## 4.3 Test Plan

### 4.3.1 Testing Approach

Our testing approach is going to follow the black box testing approach which tests the functionality of the features without focusing on the way they were implemented internally.

1. Unit Testing

In unit testing our main focus will be to see how accurate our detection algorithm works. Our goal is to have an acceptable percentage of accuracy which can be depended on without any human involvement. A percentage of 75% or above of accuracy will be considered acceptable.

2. Stress Testing

The Goal of this testing is to see how many concurrent users can be online on the system, this testing is important as this system is designed to be used on groups and not just individuals.

3. Performance Testing

It is important that our system has a small run-time in order to get the best results at the right time for maximum effectiveness.

### 4.3.2 Testing Tools

- **West Wind WebSurge Tool:** A tool that tests the load that the website can take at once without crashing or dysfunctioning.
- **Apache Jmeter:** This tool analyzes the performance of certain functions under different circumstances.

### 4.3.3 Tested Features

The features that were tested in our project are:

- Login.
- SignUp.
- Photo eligibility for facial recognition.
- Screenshots Clarity.
- Impersonation Incidents.
- Appearance of multiple people.
- Unauthorized object detection.
- Human voice detection.

#### 4.3.4 Test Risks

Table 27: Tests Risk

<b>ID</b>	<b>Risk</b>	<b>Test Objective</b>	<b>Strategy</b>
R1	Login doesn't recognize users in the database.	Make sure login recognizes users in the database and gives the right privileges	Avoid
R2	Signup lets users fill information with no constraints including their picture.	Make sure that the signup page forces the users to choose passwords that are 3-12 characters and no duplicate usernames	Avoid
R3	Users uploading pictures where their faces cannot be recognized and saved for further use in facial recognition algorithms.	Make sure that pictures uploaded by users are being processed and only those that can be used in facial recognition are approved and saved in the database.	Avoid
R4	When a webcam fails to take a clear screenshots for image processing the webpage won't warn them or kick them out of the session.	Make sure that the webpage detects when a screenshot is not clear enough for image processing and warn the user and take them out of the exam if they don't fix it.	Control
R5	The image processing algorithm does not detect when someone not authorized to take the exam is in the session.	Demonstrate that when an unauthorized person takes the exam the image processing algorithm will detect that, kick the user out of the exam and show an explanatory message.	Avoid
R6	The image processing algorithm does not detect when more than one person is in the session.	Demonstrate that the system will detect when more than one person is in session and act accordingly.	Avoid
R7	The image processing algorithm does not detect when an unauthorized object appears in use with the student.	Demonstrate that most unauthorized objects are detected by the system during a session.	Control
R8	The system doesn't recognize when another human voice appears in the session	Demonstrate that the system will detect a human voice in the session.	Control

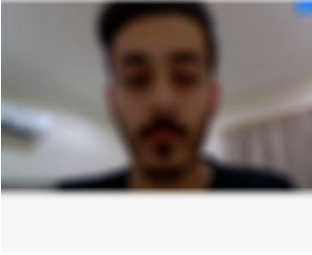
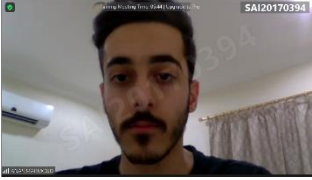

R9	System Results in an inaccurate cheating probability percentage that is not representative of what took place during a session.	Demonstrate that the accuracy of the calculated cheating percentage is above 75%	Control
----	---	--	---------

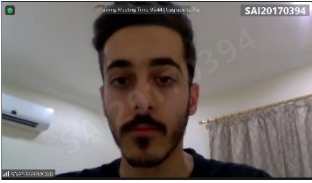
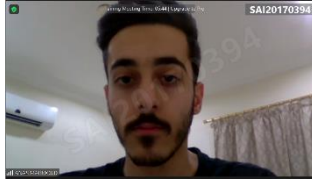

#### 4.3.5 Pass/Fail Criteria.

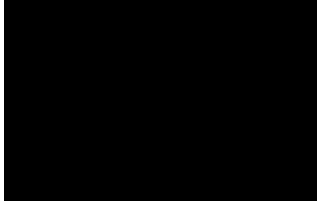

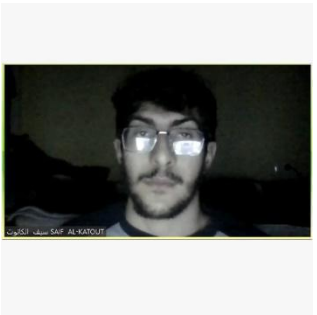

- 1. Unit Testing:** In unit testing the pass/fail criteria will be whether the objective of the risk item shown in table 1 was met or not.
- 2. Stress Testing:** Our main goal is for the system to be able to work with 30 concurrent users.
- 3. Performance Testing:** The system should find the detection results in 5 seconds at most after an incident takes place.



Table 28: Test cases

ID	Case Scenario	Test sample	Result	Output
	<b>Login:</b>			
P1	User enters wrong login credentials	<b>Username: "mmgsg"</b> <b>Password: "949204"</b>	<b>Fail</b>	"Wrong login Credentials"
P2	User enters correct Username	<b>Username: "Marwan"</b>	<b>Pass</b>	Main Page
P3	User enters correct Password	<b>Password: "12345678"</b>	<b>Pass</b>	Main Page
	<b>Sign Up:</b>			
P3	User enters a password not between 3-12 characters long	<b>Password: "12"</b>	<b>Fail</b>	"Please Enter a password between 3-12 characters long."
P4	User enters a duplicate Username	<b>Username: "Marwan"</b>	<b>Fail</b>	"Username is already registered in the database please try another username."
P5	<b>User enters a picture of themselves</b>		<b>Fail</b>	"Your face was

	where their face cannot be detected by the face recognition algorithm			not detected in the picture you've uploaded, please make sure it is not blurry or dark and upload again.”
	Blurry picture.		<b>Fail</b>	
	An image of an already registered user.	 Assuming this person has an account	<b>Fail</b>	
	A picture with more than 1 person in it		<b>Fail</b>	
P6	User enters correct information and eligible picture		<b>Pass</b>	Main Page



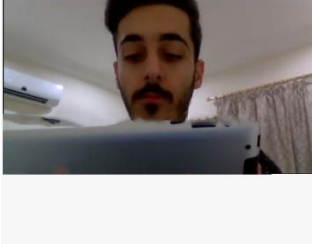
	User enters correct Username	Username: "Saif"	Pass	Main Page
	User enters correct Password	Password: "123456"	Pass	Main Page
	User enters correct Email	Email: " <a href="mailto:Saif@psut.io">Saif@psut.io</a> "	Pass	Main Page
	User enters correct Picture		Pass	Main Page
	<b>Photo eligibility for facial recognition:</b>			
P7	When signing up the user uploads a clear and eligible photo that can be recognized by the system as required.		Pass	Successful sign up.
	<b>Screenshots clarity:</b>			
P8	The system takes a screenshot every random time for analysis, The screenshot is clear and the person is recognized each time.		Pass	Successful photo analysis.
P9	The screenshot taken by the system is not clear due to many problems:		Fail	The system will retake the screenshot and discard the previous one.

	<b>Tape the camera.</b>		<b>Fail</b>	
	<b>Use scratched or broken camera.</b>		<b>Fail</b>	
	<b>Take screenshots in a dark room.</b>		<b>Fail</b>	
	<b>Focus the light on the camera.</b>		<b>Fail</b>	
	<b>Impersonation Incidents:</b>			
P10	The student recognized is always the same in every screenshot taken.	<b>Take a screenshot of the authorized user then take another screenshot that shows the same person.</b>	<b>Pass</b>	No impersonation incident detected.

P11	In one or more screenshots taken the face recognized is not the same for the student taking the exam.	<b>Take a screenshot of the authorized user then take another screenshot that shows a different person.</b>	<b>Pass</b>	A warning will be sent to the student (Impersonation incident detected)
	<b>Appearance of multiple people:</b>			
P12	More than 1 person detected in the room		<b>Pass</b>	Message: Please make sure that you are alone in the room. Or you will be blocked
P13	More than 1 person was in the room but it was not detected.		<b>Fail</b>	-
	<b>Unauthorized Object Detection:</b>			



P14	User is using an unauthorized object that appeared in the frame and it is detected.			Message: “An unauthorized object is detected”
	Use a mobile phone in the screenshot.		Pass	
	Use another laptop or tablet.		Pass	
	<b>The following must not be detected:</b>			
	Use a calculator.		Pass	
	Drink a beverage.		Pass	

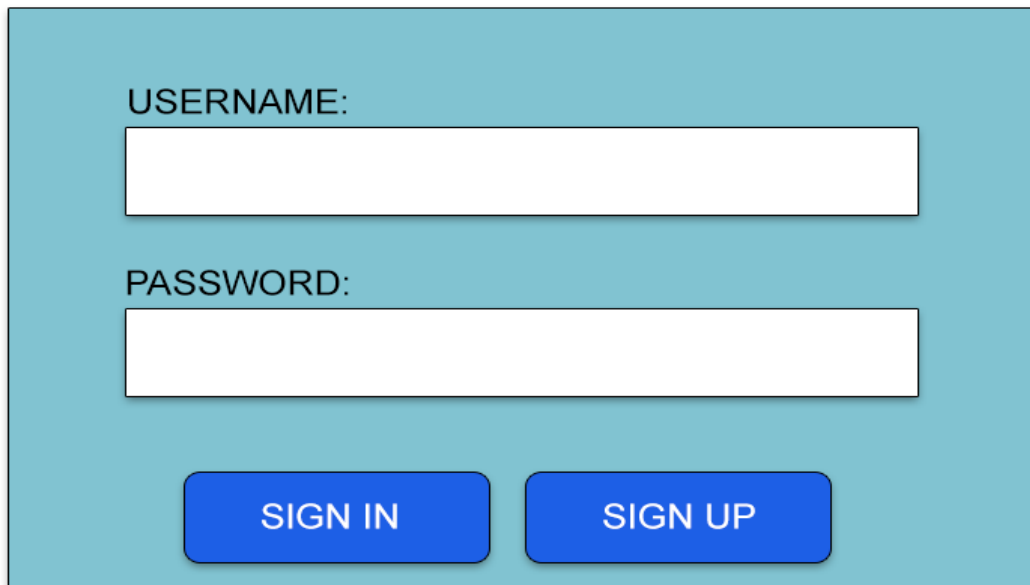
	A wall picture in the background.		Pass	
P15	User is using an unauthorized object that appeared in the frame and it was not detected.			-
	Using a mobile phone in the screenshot.		Fail	-
	Using another tablet or laptop.		Fail	-
	<b>Human voice detection:</b>			
P16	User is talking to someone else and it was not detected.	<ol style="list-style-type: none"> <li>1- Make one person talk only.</li> <li>2- Make two or more people talk together.</li> </ol>	Fail	-

		<ul style="list-style-type: none"> <li>3- High and close sound test.</li> <li>4- Low and far sound test.</li> </ul>		
P17	User is talking to someone else and it was detected.	<ul style="list-style-type: none"> <li>1- Make one person talk only.</li> <li>2- Make two or more people talk together.</li> <li>3- High and close sound test.</li> <li>4- Low and far sound test</li> </ul>	Pass	Message: "High audio levels detected."

#### 4.4 User Interface design

In this section we will include how our system is expected to look in different pages.

#### 4.4.1 Sign-In:



USERNAME:

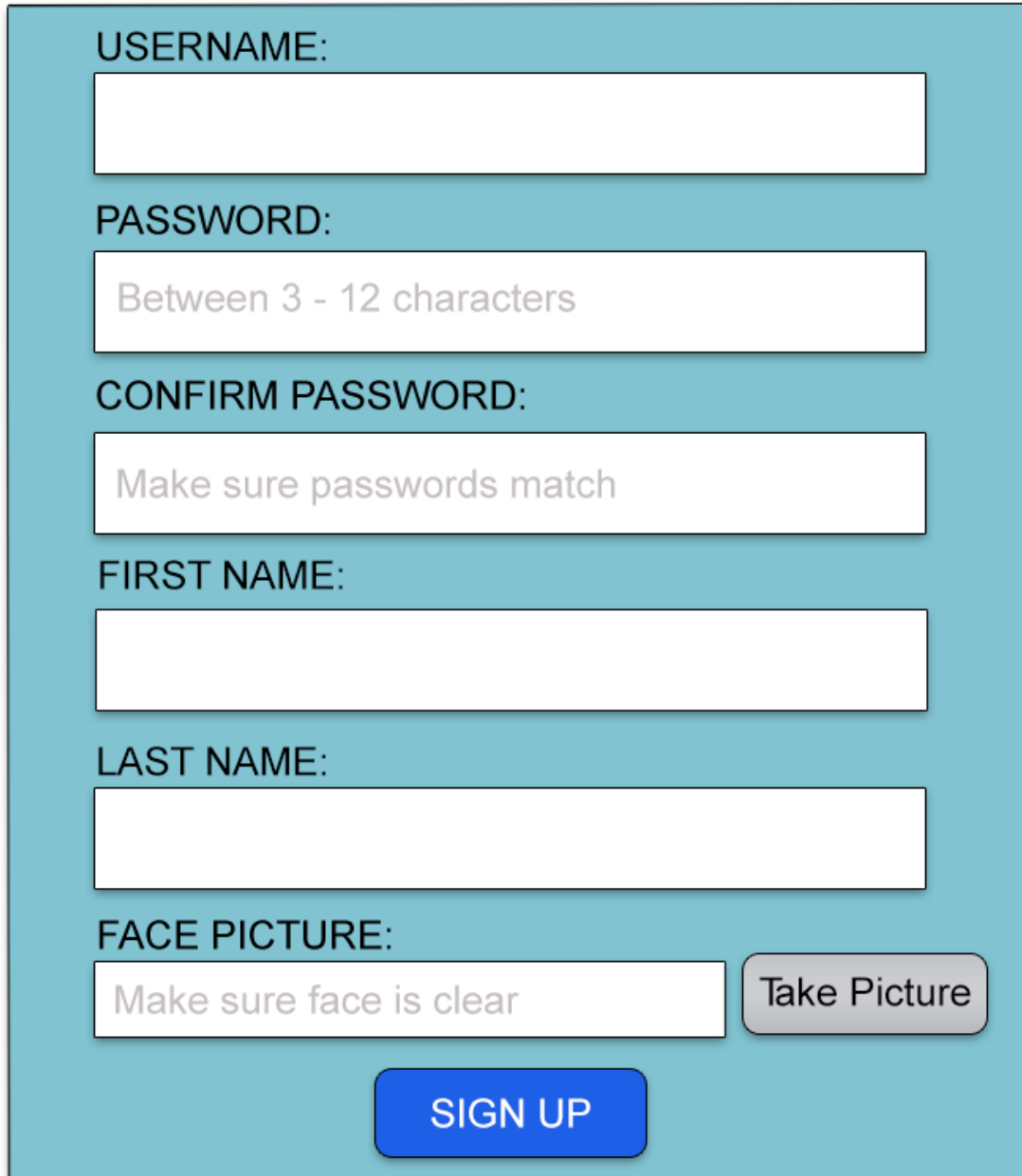
PASSWORD:

[SIGN IN](#) [SIGN UP](#)

[Forgot Password?](#)

Figure 35:Login Figure

#### 4.4.2 Sign-Up:



The image shows a sign-up form with a light blue background. It contains the following fields and buttons:

- USERNAME:** A white text input field.
- PASSWORD:** A white text input field with the placeholder text "Between 3 - 12 characters".
- CONFIRM PASSWORD:** A white text input field with the placeholder text "Make sure passwords match".
- FIRST NAME:** A white text input field.
- LAST NAME:** A white text input field.
- FACE PICTURE:** A white text input field with the placeholder text "Make sure face is clear". To its right is a grey button labeled "Take Picture".
- A large blue button labeled "SIGN UP" is centered at the bottom of the form.

Figure 36: Sign Up Interface

### 4.4.3 In-Session:

Course: Computer Science 101  
Session ID: 500143



● **Session 500143 is being recorded**

**WARNING:** this is a one way question

Question 17:

The best university in Jordan is \_\_\_\_\_ :

- HARVARD
- PSUT
- PSUG
- MIT

[NEXT](#)

Figure 37:Session interface

#### 4.4.4 In-Session unauthorized object detection:

Course: Computer Science 101

Session ID: 500143



- **Session 500143 is being recorded**

WARNING: this is a one way question

Question 17

The best un

HARVAR

PSUT

PSUG

MIT

WARNING: An Unauthorized object was detected!

This is your first warning

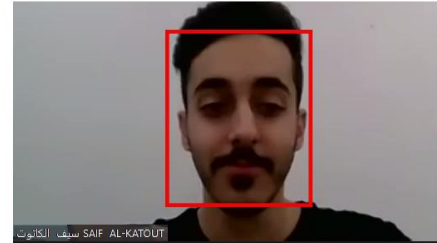
OK

NEXT

Figure 38: Object Detection interface

#### 4.4.5 In-Session impersonation incident detected:

Course: Computer Science 101  
Session ID: 500143



- **Session 500143 is being recorded**

WARNING: this is a one way question

Question 17

The best un

HARVAR

PSUT

PSUG

MIT

WARNING: UNAUTHORIZED PERSON DETECTED

You will be kicked out of the session

OK

NEXT

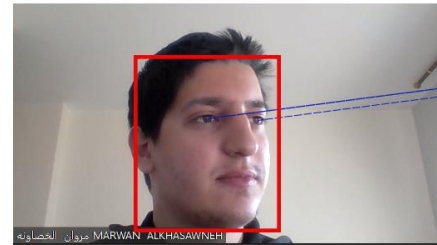
Figure 39: Impersonation Incident Interface



#### 4.4.6 In-Session eye movement detection:

Course: Computer Science 101

Session ID: 500143



● **Session 500143 is being recorded**

WARNING: this is a one way question

Question 17

The best un

HARVAR

PSUT

PSUG

MIT

WARNING: Off screen focus time is exceeding the limit.

This is your first warning. Focus on screen

OK

NEXT

Figure 40: Eye movement detection interface

#### 4.4.7 Logs:

Course: Computer Science 101  
Session ID: 500143

- ANAS MAHMOUD
- Student Recognized
  - Student entered session 500143
  - Student finished session
  - (14:55:52:29): Cheating probability LOW
  - Student left session.

- مروان الخصاصونه MARWAN ALKHASAWNEH
- Student Recognized
  - Student entered session 500143
  - (14:10:55:39): Eye movement ratio exceeded limit
  - (14:10:57:39): Student Warned.



- Student finished session
- Student left session.

- SAIF AL-KATOUT
- Student Recognized
  - Student entered session 500143
  - (13:40:56:40): UNAUTHORIZED OBEJCET DETECTED



- (13:55:34:12): UNAUTHORIZED PERSON DETECTED



- Student Kicked out of session for impersonation

Figure 41:Logs Interface

#### 4.4.8 Forgot Password Interface:

Enter the username used to sign in into the system:

Email Address

NEXT

Figure 42:Forgot password interface

#### 4.4.8.1 Pass Code Interface:

A passcode was sent to the email associated with the username you've entered, please enter the passcode:

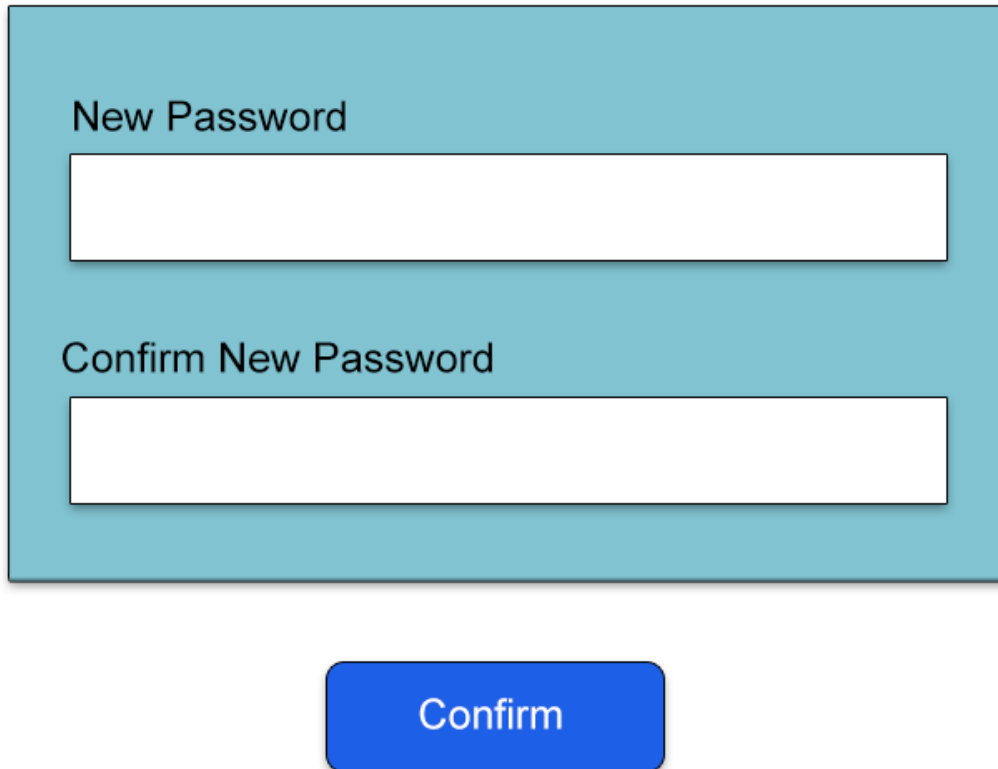
Pass Code:

NEXT

Figure 43:Passcode interface

#### 4.4.8.2 Enter New Password Interface:

Enter the new password you'd like to be used  
on your account:



The form is contained within a light blue rectangular box. It features two text input fields. The first field is labeled "New Password" and the second is labeled "Confirm New Password". Below these fields is a blue button with the text "Confirm".

New Password

Confirm New Password

Confirm

*Figure 44:New password interface*

#### 4.4.9.A Homepage Interface for students:

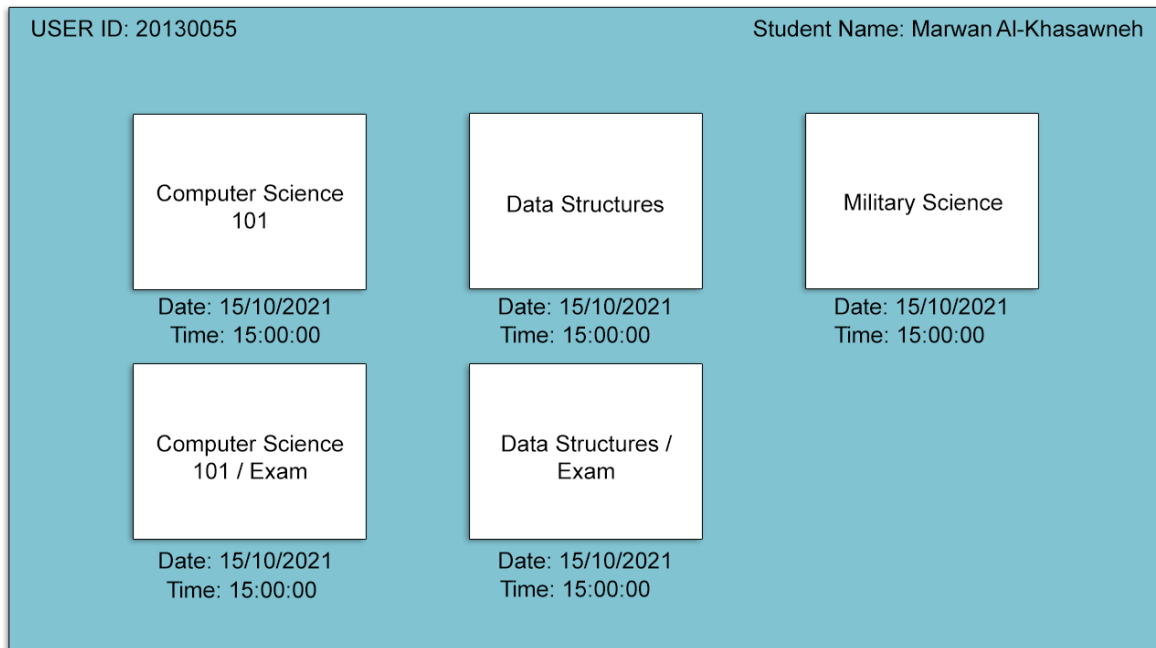


Figure 45: Students homepage interface

#### 4.4.9.B Homepage Interface for monitors:

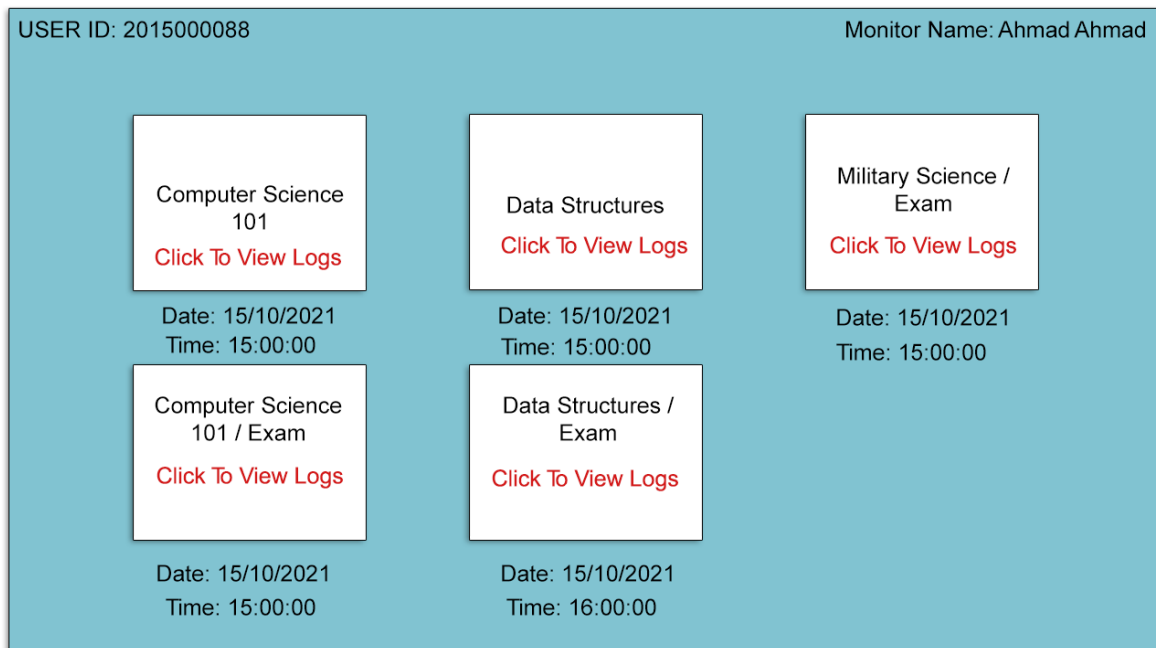


Figure 46: Monitors homepage interface

#### 4.4.10 Face Validation once entering a session:

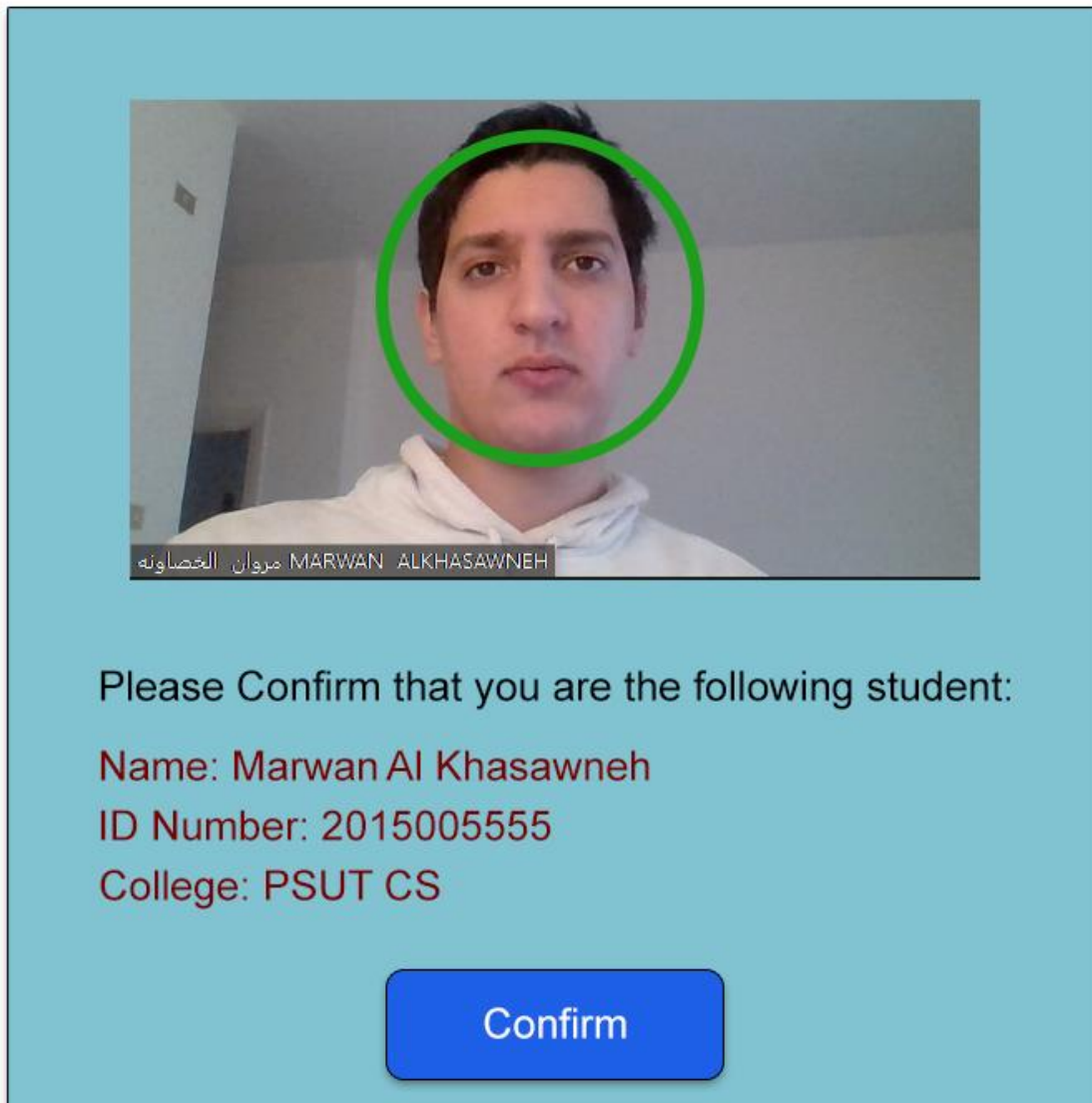


Figure 47:Face validation for session

## Chapter 5

# Implementation

To Implement ABMS we used a variety of tools, frameworks and programming languages.

- **Python:** It was the main language used in developing our project, the reason behind that is the powerful libraries that Python offers in computer vision, it was used to connect all the different libraries all together in our detection algorithms.
- **OpenCv Library:** Stands for Open Source Computer Vision, it was used in facial recognition, object detection and Eye-Movement detection.
- **DLIB Library:** It was used in particular in the Eye-Movement Detection algorithm, we made use of the 68 facial landmarks that this library provides in order to use them to detect where the pupil is at a given time.
- **YoloV3:** stands for You Only Look Once, it's a real-time object detection algorithm which we trained to detect pupils, calculators and phones.
- **Django:** we used Django in implementing our web system which is used to present ABMS.
- **SQLite:** Used to store the data in Django.
- **JavaScript:** We used it to run back-end functions on the web system, with ajax.
- **HTML:** for dividing our web pages into forms and buttons.
- **CSS:** for styling our web pages.

## General Overview

This overview covers the number of tables used in the databases, number of lines in each file and web page of the system.

Table 29 : Lines Of Code

File #	File Name	# Of Lines
1	AttendanceProject.py	67
2	Authentication.py	65
3	Cheating.py	96
4	EyeMovement.py	113
5	models.py	26
6	views.py	153

7	multiple_faces.py	24
8	same_person.py	41
9	urls.py	20
10	yolo_object_detection.py	58
11	apps.py	5
12	attendance.html	91
13	error.html	83
14	ExamPage.html	150
15	FinishExam.html	71
16	homepage.html	63
17	login.html	85
18	logs.html	13
19	monitor.html	67
20	signup.html	115
21	success.html	52
	<b>Total:</b>	<b>1458</b>

## Implemented Features

*Table 30: Implemented Features*

<b>ID</b>	<b>Description</b>	<b>Implementation</b>
FR1	User's faces should be identified and matched with their picture in the database to take attendance or detect impersonation incidents.	Implemented
FR2	System should take a screenshot every random amount of	Implemented



	seconds.	
FR3	System should analyze the user's eye movement.	Implemented
FR4	System should scan the user's background looking for objects such as smartphones, laptops and other people.	Implemented
FR5	The system should provide a gesture recognition to perform a certain action	Not Implemented
FR6	System should be able to calculate the possibility of a user cheating. This possibility is derived from the screenshots.	Implemented
FR7	System should be able to identify if the user is talking to someone.	Not Implemented
FR8	System should store all the logs that resulted from the monitoring process for future use.	Implemented
FR9	Users should be able to sign in using their University ID.	Implemented
FR10	System should be able to recognize users stored in the database. Based on their picture identification.	Implemented
FR11	System should be able to differentiate between students and monitors stored in the database in order to give each the right permissions.	Implemented
FR12	New students should be able to sign up and upload their pictures.	Implemented
FR13	System should be able to check whether a new picture is eligible for use in facial recognition.	Implemented
FR14	System should be able to store new students' pictures in the database.	Implemented

FR15	System should be able to give monitors access to the logs stored in the database.	Implemented
FR16	System should provide a forget password option by sending an email that allows the user to enter a new password.	Not Implemented
FR17	System should be able to give monitors access to the logs stored in the database.	Implemented

### **Eye Movement Detection Overview:**

The point of this algorithm is to find out whether the user is hiding the studying material beside the computer or is communicating with someone else in the room ,sitting beside them for example, or is using a prohibited device that is not appearing in the frame bypassing the object detection algorithm.

In implementing the eye movement detection algorithm we focused on detecting the eye as a whole at first, and then detecting the pupil independently, when doing that we'll end up with the coordinates of the eye itself in addition to the coordinates of the pupil, which would give us the ability to compare both coordinates predicting where the pupil is, thus, detecting where the user is looking at a given time.

This code was mainly developed using OpenCV and Dlib, integrating these libraries together was to use OpenCV's ability to implement machine learning algorithms (YOLOv3 in particular) in addition to Dlib's 68 facial landmarks.

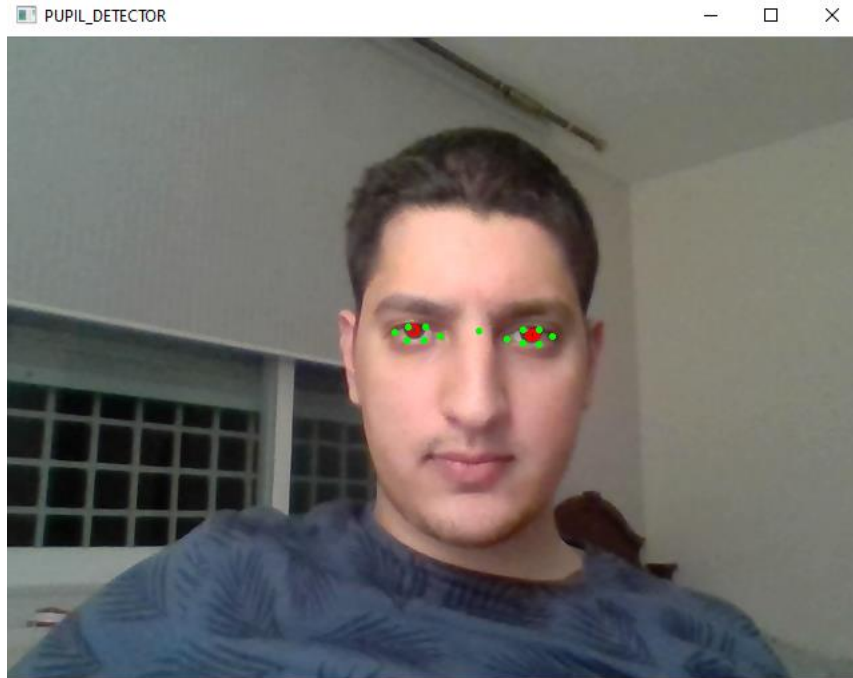


Figure 48: Eye & Pupil Detector Result

As you can see in Figure 100, the green cells are the ones detected through Dlib's facial landmarks detection, we only used the cells on the eye itself and one cell in the middle of the face to detect the head tilt, the red cells on the Pupils are the one's detected using OpenCV, after detection and drawing these cells (The drawing is also using OpenCV) we can extract their coordinates and the process them to predict where the user is looking, below is the coordinates processing function used.

```

def Coordinates_Process(PUPIL,X_DLIB,Y_DLIB):
    Sensitivity = 1.5 # BETWEEN 0 AND 2 ONLY
    Left = False
    Center = 0
    Right = False

    for x in range(0,len(PUPIL),2):
        if X_DLIB[6] > PUPIL[x]: #RIGHT EYE
            if X_DLIB[1] <= PUPIL[x] - Sensitivity <= X_DLIB[2]:
                Right = True
            if X_DLIB[2] <= PUPIL[x] <= X_DLIB[3]:
                Center+= 1
            if X_DLIB[3] <= PUPIL[x] + Sensitivity <= X_DLIB[4]:
                Left = True
        else: #LEFT EYE
            if X_DLIB[7] <= PUPIL[x] - Sensitivity <= X_DLIB[8]:
                Right = True
            if X_DLIB[8] <= PUPIL[x] <= X_DLIB[9]:
                Center +=1
            if X_DLIB[9] <= PUPIL[x] + Sensitivity <= X_DLIB[10]:
                Left = True
    if Head_tilt_detector(X_DLIB):
        print("User is tilted off screen")
        return True
    elif Right == True:

```

Figure 101:Part Of the Coordinates Processing Function

## Cheating Prediction Percentage Overview:

Our goal of this algorithm was to produce a percentage that represents the possibility of a specific student cheating by analyzing the factors we've detected, also, we wanted to use this possibility to increase or decrease the amount of screenshots taken depending on whether the user is cheating or not, in other words, we wanted to take more screenshots to analyze of the users who we think are cheating than the ones who are not. It's also worth mentioning that these screenshots are taken at random intervals to prevent users from predicting when the next screenshots are going to be taken.

To implement this algorithm we had to research about probability distributions to choose the right one depending on our data collected, we chose the Poisson Distribution, using SciPy library, which calculates whether a trial succeeds or fails during a fixed amount of time, which, works in our case of trying to see whether a student is cheating or not during the time of the exam.

```

def driver(student_id, exam_id):
    global log_string
    window_between_screenshots = 10
    List_of_prob = []
    List_of_Mistakes = []
    time_end = time.time() + 60 * 5
    total_number_of_screenshots = 0
    l = Log(Exams_id=exam_id, Student_id=student_id)
    l.save()
    while time.time() < time_end:
        if end_exam_flag:
            break
        X,Y,Z,Q = run_trials(window_between_screenshots, exam_id, student_id)
        if Z == True:
            Prob = Cheating_Probability(X,Y)
            List_of_prob.append(Prob)
            List_of_Mistakes.append(X)
            total_number_of_screenshots+=Y
            if Prob > 0.70:
                window_between_screenshots-=5
            else:
                window_between_screenshots+=3
        else:
            print("Different Person Detected!")

```

Figure 102: The driver function

```

def run_trials(window, exam_id, student_id):
    logger = ""
    Mistakes = 0
    Counter = 0
    Same_Person_Flag = True
    for x in range(10):
        if end_exam_flag:
            break
        img = random_screenshot()
        print("Screenshot Time:", time.asctime(time.localtime(time.time())))
        logger += ("Screenshot Time:" + time.asctime(time.localtime(time.time())) + ",")
        X, Y, Z = Pupil_Eye_Detector(img)
        if multi_faces(img):
            logger += "More than one person detected!"
        else:
            logger += "No more than one person detected ,"
            #cv2.imwrite('.')
            Mistakes+=2 #Higher Weight
        if same_person(img) == '[False]' and not multi_faces(img):
            print("Impersonation detected, you will be kicked out of the exam!")
            Same_Person_Flag = False
            return Mistakes, Counter, Same_Person_Flag
        if Object_Detection(img):
            Mistakes+=1

```

Figure 103: Running Trials Code

```

end_exam_flag = False

def Cheating_Probability(x, counter):
    Screenshots_taken = counter
    Mistakes_Detected = x
    Number_Of_Mistakes_to_Happen = 4
    Poisson = scipy.stats.poisson(Mistakes_Detected)
    probability = (1-Poisson.cdf(Number_Of_Mistakes_to_Happen)) #X.cdf(5)=P(X<=5)
    print(probability)
    return probability

```

Figure 104: Calculating Probability

## Face Detection Algorithm Overview:

This algorithm is divided into four main parts but all have the same purpose, which is to make sure only the exam taker is appearing in the webcam.

We used OpenCV which has built in functions that will help to implement this type of algorithms.

### 1- User Authentication:

The first algorithm checks whether the user is allowed to enter the exam or not by comparing his face locations and face landmarks which are unique for every person from a webcam screenshot with an original image of the user in the database. If there was a match then user is allowed to take the exam.

```
def user_auth():
    taken_image = screenshot()
    global taken_image_encoding, profile_img, face_encoding
    print('Checking Identity...')
    path = glob.glob('C:\\Users\\Anas BeLa\\Downloads\\ABMS\\src\\media\\images\\*.jpg')
    for file in path:
        profile_img = face_recognition.load_image_file(file)
        profile_img = cv2.cvtColor(profile_img, cv2.COLOR_BGR2RGB)
        face_encoding = face_recognition.face_encodings(profile_img)[0]
        taken_image = cv2.cvtColor(taken_image, cv2.COLOR_BGR2RGB)
        taken_image_encoding = face_recognition.face_encodings(taken_image)[0]
        result = face_recognition.compare_faces([face_encoding], taken_image_encoding)
        result_string = str(result)
        if result_string == '[True]':
            break

    i = 0
    check = False
    while i < 3 and check == False:
        try:
            taken_image_encoding = face_recognition.face_encodings(taken_image)[0]
            check = True
            break
        except:
            cv2.rectangle(taken_image, (150, 200), (500, 300), (255, 255, 255), -1)
            cv2.putText(taken_image, 'Authentication failed, Retaking image after 5 seconds', (190, 250), cv2.FONT_HERSHEY_TRIPLEX, 0.5, (0, 0, 255), 1)
            #cv2.imshow('image', taken_image)
            cv2.waitKey(5)
            cv2.waitKey(5000)
            screenshot()
            #taken_image = face_recognition.load_image_file('image.jpg')
            taken_image = cv2.cvtColor(taken_image, cv2.COLOR_BGR2RGB)
            i += 1

    result = face_recognition.compare_faces([face_encoding], taken_image_encoding)
    result_string = str(result)

    if result_string == '[True]':
```

Figure 110: User Authentication Algorithm

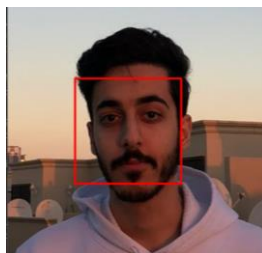


Figure 49: Face Detected

## 2- Same Person:

The second algorithm starts by the time the exam starts. It basically checks every random time that the user is always the same by checking also the face landmarks and compare it every time. Whenever there is a miss match in the comparison, this will cause a 100% cheating which we call an impersonation incident.

```
def same_person(taken_image):
    #taken_image = random_screenshot()
    global profile_img, taken_image_encoding
    path = glob.glob('C:\\Users\\Anas Belal\\Downloads\\ABMS\\src\\media\\images\\*.jpg')
    for file in path:
        profile_img = face_recognition.load_image_file(file)
        profile_img = cv2.cvtColor(profile_img, cv2.COLOR_BGR2RGB)
        face_encoding = face_recognition.face_encodings(profile_img)[0]
        taken_image = cv2.cvtColor(taken_image, cv2.COLOR_BGR2RGB)
        taken_image_encoding = face_recognition.face_encodings(taken_image)[0]
        result = face_recognition.compare_faces([face_encoding], taken_image_encoding)
        result_string = str(result)
        if result_string == '[True]':
            break

    face_encoding = face_recognition.face_encodings(profile_img)[0]
    #taken_image = face_recognition.load_image_file('same-person-screenshot.jpg')
    taken_image = cv2.cvtColor(taken_image, cv2.COLOR_BGR2RGB)

    try:
        taken_image_encoding = face_recognition.face_encodings(taken_image)[0]
    except:
        #taken_image = face_recognition.load_image_file('same-person-screenshot.jpg')
        taken_image = cv2.cvtColor(taken_image, cv2.COLOR_BGR2RGB)

    result = face_recognition.compare_faces([face_encoding], taken_image_encoding)
    result_string = str(result)

    if result_string == '[False]':
        print('Different person detected')
    else:
        print('No Impersonation Detected')

    return result_string
```

Figure 50: Same Person Detection Algorithm

## 3- Multiple Faces

The third algorithm also starts by the time the exam starts. To make sure there is only one detected face in the screenshot by using OpenCV, face locations and face landmarks. We used also a trained cascade sheet for detecting the frontal face (**haarcascade\_frontalface\_default.xml**)

The way it work is by detecting all the faces in the screenshot first then using a counter it counts the number of faces appeared, If the number is not equal to one it will increase the cheating percentage.



```

import cv2
import time
from .Eye_Movement import *
import os
from src.settings import BASE_DIR

def multi_faces(img):
    #img = random_screenshot()
    face_cascade = cv2.CascadeClassifier(os.path.join(BASE_DIR, 'monitoring\\haarcascade_frontalface_default.xml'))
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)
    for (a, b, c, d) in faces:
        cv2.rectangle(img, (a, b), (a+c, (b+d)), (255, 0, 0), 2)
    try:
        if faces.shape[0] != 1: #If more than one face appeared
            detection_time = time.asctime(time.localtime(time.time()))
            return True
    except:
        return False

```

Figure 51: Multiple Faces Detection

#### 4- Attendance:

Another feature was used is exam attendance using face detection. With the help of the OpenCV to compare faces and store the matching faces on an external csv file that can be viewed by the monitor.

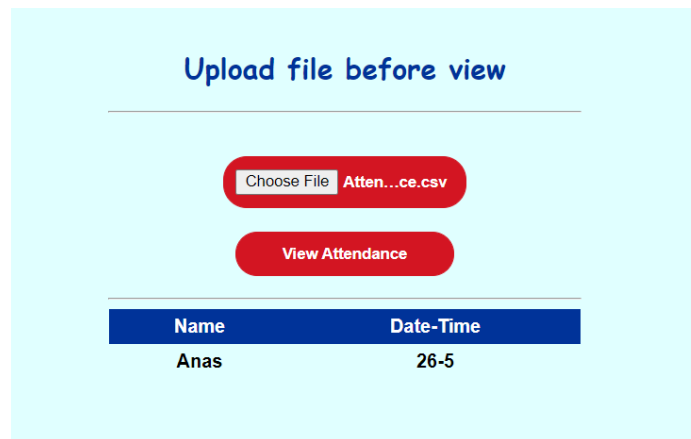


Figure 52: Attendance Recording

#### Object Detection Algorithm Overview:

This function detects unauthorized objects in the picture which is taken, most specifically it detects phones, laptops and calculators, this can be done automatically by the computer, which can result in higher accuracy and it can create privacy for the student, because they need not to open the camera for other students or in unwarranted situation; when they are not cheating.

We implemented this function by first using the yolo object detection algorithm which was trained on the cocoa data set for phones, laptops, however for calculators we needed to create a custom dataset, by having a large dataset of photos which we took of different calculators in different angles, since we didn't find any calculator photo datasets fit for our use on the internet (google datasets and others were looked into). We used a google notebook to access google GPUs and created the datasets that would work with the yolov3 algorithm with darknet.

We used yolo because it has an API to work with openCV which is perfect for keeping the same theme for the whole project and not create a cluster of different tools that were hard to communicate with each other.

```
def Object_Detection(img):
# Load Yolo
net = cv2.dnn.readNet(os.path.join(BASE_DIR, 'monitoring\\yolov3.weights'), os.path.join(BASE_DIR

with open(os.path.join(BASE_DIR, 'monitoring\\coco.names.names'), "r") as f:
    classes = [line.strip() for line in f.readlines()]
layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
colors = np.random.uniform(0, 255, size=(len(classes), 3))

# Loading image
img = cv2.resize(img, None, fx=0.4, fy=0.4)
height, width, channels = img.shape

# Detecting objects

blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
net.setInput(blob)
outs = net.forward(output_layers)
```

Figure 53: Object Detection Algorithm

## Web Application Overview:

For the web application we used the web framework called Django for the python language, this allowed us to use all the programs that we created without the need of an API between different languages, or implementing shared resources for example.

Django comes fit with a database API that works with sqllite files, so python function are able to created database tables with columns and restrictions etc..., it also comes with a small-scale webserver out of the box, that we find perfect for our use case, since we will focus on development rather than full scale production.

Furthermore, we resorted to using JQuery to get some information sent from the client to the server, because django doesn't have that functionality, but for our server to client needs django contains context objects and templates that allow for that type of information flow.

Continuing, we used threading which would allow us to run the backend main function that calculates the cheating possibility while the photos are being feed to it, while the client was able to take the exam, and when the user/client exits the exam, the backend function stops.

Finally, we used the database API in the backend files to store the logs of the attempt of each student and we used database design to assign each set of logs which they represent and the exam they were taken in, so the admin/proctor can view each exam's logs for all students.

```
def ExamPage(request):
    try:
        eid = request.GET.get('id')
        if request.method == 'GET':
            if request.is_ajax() == True:
                time.sleep(3)
                Cheating.driver(request.session['user_id'], eid)
            if user_auth() == '[True]':
                main()
                return render(request, 'monitoring/ExamPage.html', {'id': eid})
            else:
                return render(request, 'monitoring/notAllowed.html')
    except Exception as e:
        print(e)
        return render(request, 'monitoring/faceError.html')

def FinishExam(request):
    Cheating.end_exam_flag = True
    return render(request, 'monitoring/FinishExam.html')
```

Figure 54: Web Application Algorithm

# Chapter 6

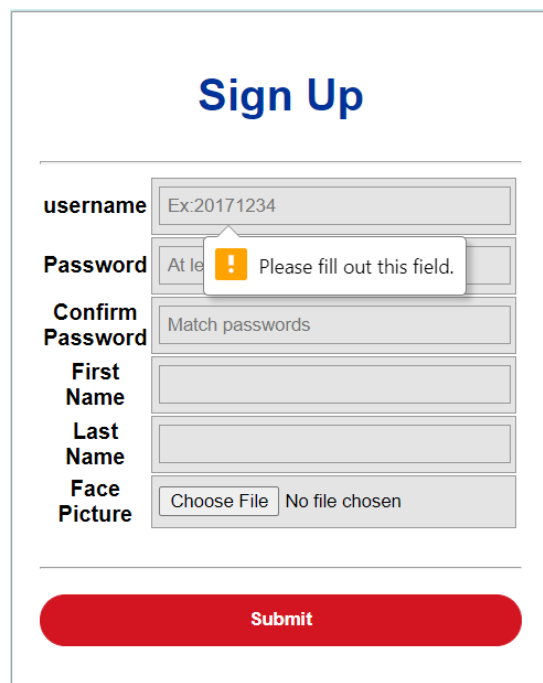
# Testing

## 6.1 Testing Approach

Our system ABMS was tested using Black Box Testing. This approach focuses on the functionality of the system rather than the internal structure.

Validations:

- 1) User should create an account and fill all the fields, which are required to successfully signup. The figure below shows the error message when the user don't fill the fields.



The image shows a 'Sign Up' form with the following fields and error messages:

- username**: Input field with placeholder text 'Ex:20171234'.
- Password**: Input field with an error message: 'At le [Warning Icon] Please fill out this field.'
- Confirm Password**: Input field with placeholder text 'Match passwords'.
- First Name**: Input field.
- Last Name**: Input field.
- Face Picture**: File upload field with a 'Choose File' button and the text 'No file chosen'.

At the bottom of the form is a red 'Submit' button.


Figure 55:: Sign Up Form

- 2) Passwords must be at least 8 characters and passwords should match. The figure below shows the error message when password is less than 8 characters.

## Sign Up

**username**

**Password**

 Please lengthen this text to 8 characters or more (you are currently using 7 characters).

**First Name**

**Last Name**

**Face Picture**  No file chosen

Figure 56: Sign Up Verification

3) When the user fill all the required fields in a correct way he will be informed.

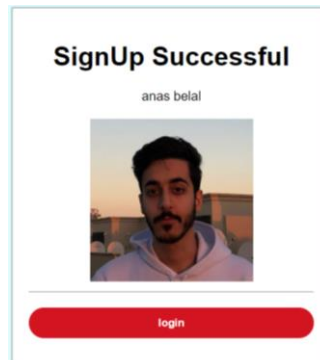


Figure 57: Sign Up Successful

4) The user must enter correct credentials. The figure below shows the error message in case of entering wrong credentials.

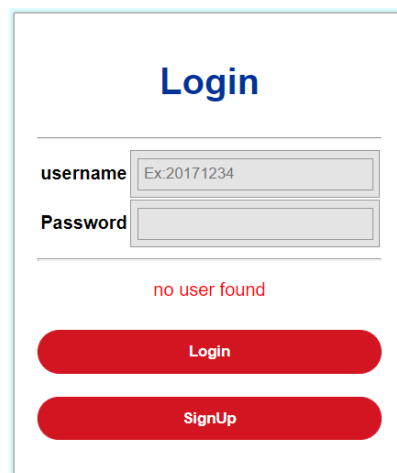


Figure 58: Login Form Verification

After login:

1) If the user has an exam, the available exams will appear to the user. Otherwise, no exam will be shown to the user.

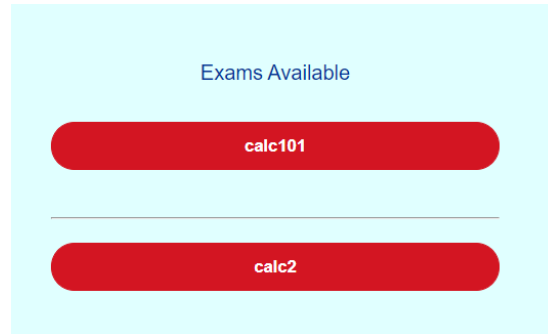


Figure 59: Exams Available Page

- 2) When user click on the exam, The system will check for authentication using webcam. The figure below shows the message when the user is not allowed to take the exam.

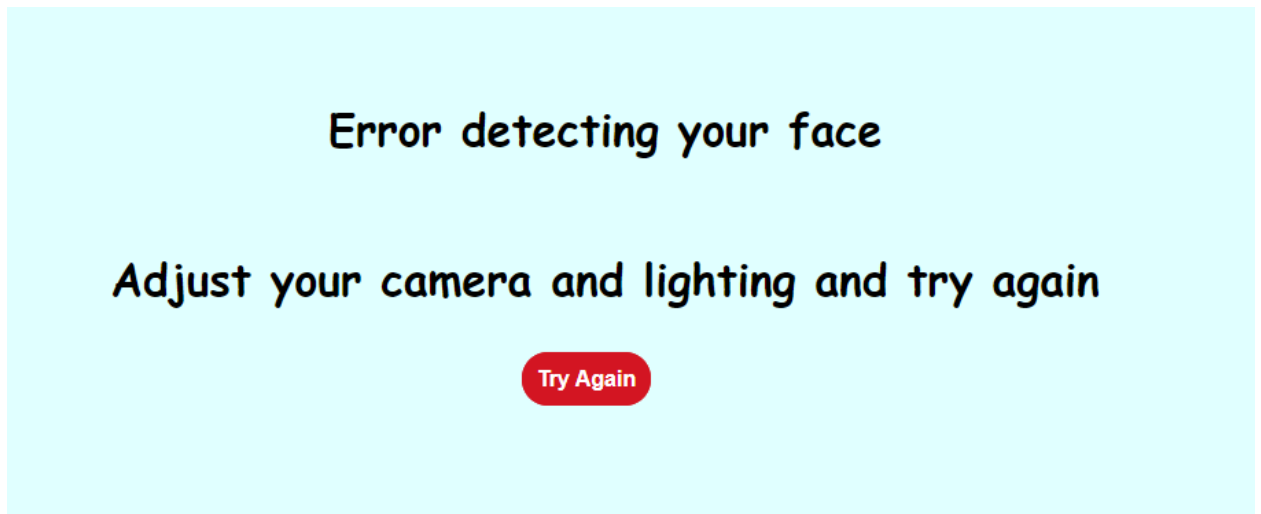
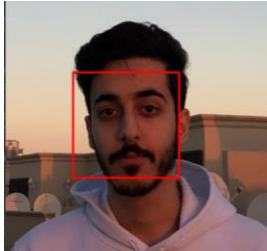



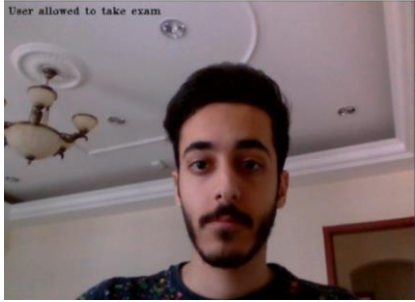


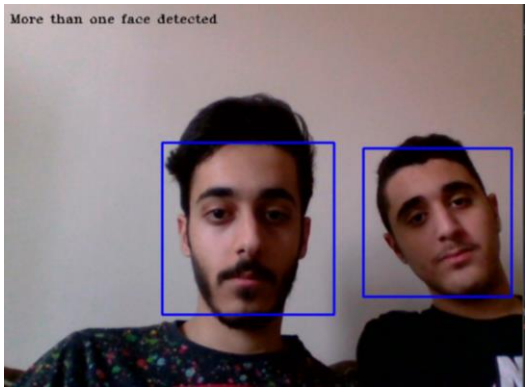
Figure 125: Exams Available Page

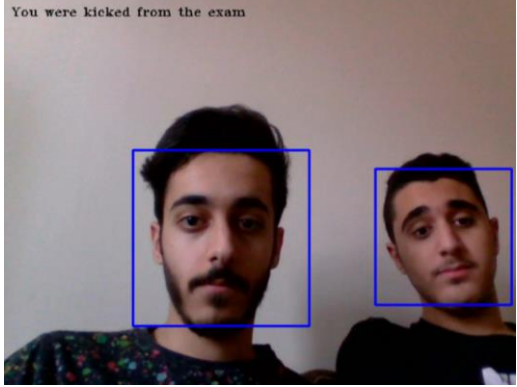

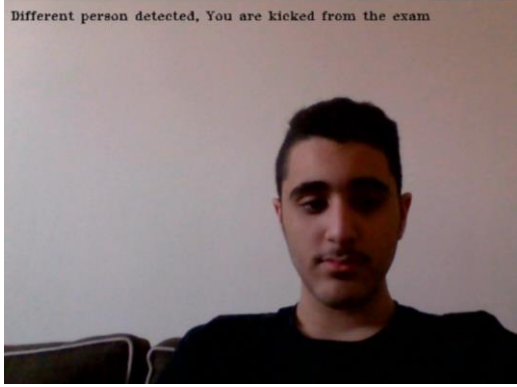
## 6.2 Tested features

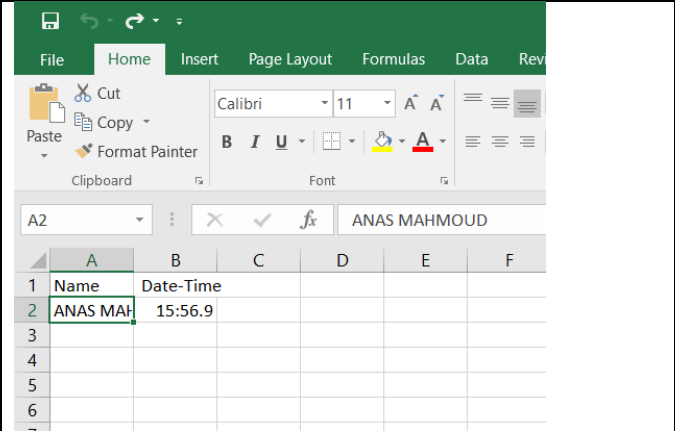
Table 31: Tested Features

Facial Recognition cases		
<p>User Authentication</p>	<ol style="list-style-type: none"> <li>1- Users taken picture matches the picture in the database. This is the normal scenario and user can take the exam because the system recognized the face.</li>   <li>2- Users taken picture does not match the picture in the database. In this scenario the system detected a stranger face so the system won't allow him to take the exam.</li>   <li>3- If the picture wasn't clear the system will retake a new picture after 5 seconds with a total of 3 times to</li> </ol>	<p>Profile picture saved in the database to be compared with</p>  <p>Fail when there is no match</p>  <p>Pass when there is a match</p>






	<p>give the user time to fix the lighting or remove any blocked objects from taking a valid picture.</p>	 <p>The user wasn't recognized because the taken picture is not clear</p> 
<p>Multiple faces detected</p>	<ol style="list-style-type: none"> <li>1- The system will take multiple and random screenshots from the webcam during the exam.</li> <li>2- The normal scenario is only the exam takers face is detected by the system.</li> <li>3- If the system detected more than one face it will give the exam taker a warning and if the user keep getting an external human help on the third warning he will be kicked from the exam.</li> </ol>	<p>Normal picture for the exam taker</p>  <p>More than one person detected with a warning is shown</p>  <p>The last warning and the user is kicked from the exam</p>

		
<p>Impersonation incident</p>	<ol style="list-style-type: none"> <li>1- In this case the user might bring someone in his place to solve the exam so we need to check that the user is always the same and he is the allowed user by comparing his taken picture with the database picture in every screenshot.</li>   <li>2- If the system detected a different person who is not allowed to take the exam he will be kicked immediately as this is an extreme case of cheating.</li> </ol>	<p>The exam taker</p>  <p>The other person detected</p> 
<p>Attendance</p>	<ol style="list-style-type: none"> <li>1- The system will take a screenshot from the webcam and compare it to the picture in the database.</li>   <li>2- Any match between two pictures means that the user has attended the exam, the name, time</li> </ol>	

	<p>and date of the user will be registered and applied to csv file then to the database.</p>	
--	--	--

**Object detection cases**

<p>Laptop detection</p>	<p>If an external laptop was detected which is considered as cheating a certain action will be taken</p>	
<p>Mobile detection</p>	<p>If any mobile was detected in the screenshot then a certain action will happen</p>	
<p>Calculator</p>	<p>We have two cases here  1- The exam needs a calculator so the user can use it without considering as cheating.  2- The calculator is not allowed in the exam so using it will be considered as cheating.</p>	<p>Case 1 where the calculator is shown in the screenshot but not detect because it is allowed</p>  <p>Case 2 where the calculator is not allowed</p>

		
--	--	---

**Eye Movement cases**

Eye detection

When taking a screenshot for analysis, the exam taker should be looking at screen otherwise it will increase the cheating percentage.

Looking at screen



Looking off screen



## Chapter 7

# Conclusions and Future Work

Working on this project, our main goal was to achieve a foundation of a simple, affordable and accurate system that detects wrong acts that take place in distance learning environments without overloading servers.

After countless days and nights of hard work, research and versions we believe that we did our best and reached the best possible results in our circumstances. Only 3 students who are not completely free, who worked only 2-4 hours a day - when lucky - were able to reach this completely functional system that provides actual results depending on real behaviors.

Implementing ABMS was challenging considering how new we were to the concepts and the tools we had to use, it was not the code that was challenging as much as the research behind learning how to implement these functions and libraries, integrating them together, to produce a result that represents reality.

In the future, we plan to increase the accuracy of our system by collecting datasets of higher quality, improve the usability and the customizability to make it fit everyone's needs, we plan to make it easier to install resulting in making it capable of running on existing systems and of course implementing the functional requirements which we unfortunately did not have enough time to research about and implement.

## References

- [1] : Gill, G. (2013). Proctorfree: deterring online cheating. *Journal of Information Technology Education: Discussion Cases*.
- [2] Kolowich, S. (2013). Behind the Webcam's Watchful Eye, Online Proctoring Takes Hold. *Chronicle of Higher Education*.
- [3] *mettl*. (2010, January 1). From mettl: [https://mettl.com/en-ae/?ads\\_cmpid=11403285036&ads\\_adid=110892226799&ads\\_targetid=kwd-372040230718&ads\\_loc\\_inrst=&ads\\_loc\\_physcl=1011785&ads\\_network=g&ads\\_gclid=CjwKCAiAgJWABhArEiwAmNVTByvm6GILMRqu1-VJzQXkKBnUnsu9fA6kP6yYiRIArfzi7iqpUbFUZR0Ch0kQAvD\\_BwE&a](https://mettl.com/en-ae/?ads_cmpid=11403285036&ads_adid=110892226799&ads_targetid=kwd-372040230718&ads_loc_inrst=&ads_loc_physcl=1011785&ads_network=g&ads_gclid=CjwKCAiAgJWABhArEiwAmNVTByvm6GILMRqu1-VJzQXkKBnUnsu9fA6kP6yYiRIArfzi7iqpUbFUZR0Ch0kQAvD_BwE&a)
- [4] Meyringer, M. (2006). Interactive and web-based Gantt Chart.
- [5] Proctoring to improve teaching practice. (2016). *MSOR Connections*.
- [6] *proctorio*. (2013, January 1). From proctorio: <https://proctorio.com/>