BSCS-F21-019

03-134182-105 SYED AMMAR HUSSAIN ZAIDI

# Online Exam Proctoring

In partial fulfillment of the requirements for the degree of

**Bachelor of Science in Computer Science**

Supervisor: **SHAFIQ AHMED**

Department of Computer Sciences

Bahria University, Lahore Campus

June 2022

# Certificate

We accept the work contained in the report
titled “Online Exam Proctoring”
written by
Syed Ammar Hussain Zaidi
as a confirmation to the required standard for the partial fulfilment of the degree of
Bachelor of Science in Computer Science.

## Approved by:

Supervisor: Shafiq Ahmed

(Signature)

June 14, 2022

# DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

| Enrollment | Name | Signature |
|---|---|---|
| 03-134182-105 | Syed Ammar Hussain Zaidi | |

Date : June 14, 2022

Specially dedicated to
my beloved mother and
father (Syed Ammar Hussain
Zaidi)

## ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Mr Shafiq Ahmed for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, I would also like to express my gratitude to my loving parent and friends who had helped and given me encouragement.

Syed Ammar Hussain Zaidi

**Project Title**

# Online Exam Proctoring

## ABSTRACT

In recent years, online examinations have been increasingly utilized to assess students' abilities to teach knowledge, particularly during the COVID-19 outbreak. However, due to the lack of face-to-face connection, proctoring online tests is difficult. Furthermore, past study has demonstrated that online examinations are vulnerable to nu- serous duping techniques, which can compromise their legitimacy. Specifically, this project detects suspected student by user verification, Head-Pose Analysis, Gaze Estimation, Person Counting, Active Window detection, Mobile Phone Detection, Multiple Monitor Detection.

Massive open online courses (M O O Cs) and other kinds of distance learning are growing in popularity and reach. The ability to successfully proctor remote online tests is a key limiting issue in this next stage of education's scalability. Human proctoring is currently the most frequent method of assessment, which involves either forcing test takers to visit an examination center or watching them visually and audibly throughout exams via a digital camera.

## TABLE OF CONTENTS

**DECLARATION** **ii**
**AC KNOWLEDGEMENTS** **iii**
**ABSTRACT** **iv**
**TABLE OF CONTENTS** **v**
**LIST OF TABLES** **vii**
**LIST OF FIGURES** **viii**

**CHAPTERS**

**1 INTRODUCTION** **11**
1.1 Background 11
1.2 Problem Statements 11
1.3 Aims and Objectives 12
1.4 Scope of Project 12

**2 LITERATURE REVIEW (and/or SRS)** **14**
2.1 Overall Description
2.1.1 User Classes and Characteristics
2.2 Final Deliverable of Project and Beneficiaries
2.2.1 Final Deliverable
2.2.2 Beneficiaries
2.3 Optional Scope
2.4 Design and Implementation
2.5 External Requirements
2.6 Use case Diagram

**3 DESIGN AND METHODOLOGY 25**

3.1 Wire frames

3.1.1 Use cases

3.2 Activity Diagram

3.3 State Diagram

3.3.1 Sequence Diagram

3.3.2 Collaboration Diagram

3.3.3 Component Diagram

3.3.4 Deployment Diagram

3.3.5 ER Diagram 35

**4 DATA AND EXPERIMENTS (and/or IMPLMENTATION) 40**

4.1 Unit Testing

4.2 Integration Testing

4.3 System Testing

4.4 Acceptance Testing

4.5 Development Tools

4.6 Techniques and Tools

4.7 Development Details

4.7.1 Code Snapshot 44

**5 RESULTS AND DISCUSSIONS (or USER MANUAL) 46**

5.1 Web Snapshots

5.2 Cheating Detection Snapshots

5.2.1 Number of Person and Eye Detection

**6 CONCLUSION AND RECOMMENDATIONS 51**

6.1 Conclusions

6.2 Future Directions

**REFERENCES 52**

# LIST OF TABLES

**TABLE TITLE PAGE**

**Table 2.1: Online Assessment Proctoring Priority Table**

**Table 3.1: Use Cases Tables**

**Table 4.1: Components Testing Tables**

## LIST OF FIGURES

**FIGURE** **TITLE** **PAGE**

Figure 2.1: Class Diagram

Figure 2.1.1: Flow chart

Figure 2.1.2: Use case Diagram

Figure 3.1: Activity Diagram

Figure 3.1.1: State Diagram

Figure 3.1.2:Sequence Diagram

Figure 3.1: Collaboration

Figure 3.1:Component Diagram

Figure 3.1: Deployment

Figure 3.1: ER Diagram

# CHAPTER 1

# INTRODUCTION

## 1.1 Background

With the growing number of people learning online, online tests are becoming more popular. On-line person proctoring, partially-automated proctoring, and to the full automatic proctoring are the three types of online proctoring procedures. Human proctoring online means that students will be observed by remote staff during all online assessments. Several online check solution providers (e.g., Criterion [9], Friend Exam Services [5]) and some universities (e.g., Educational institution of Amsterdam)[10] use this technique. Still, it works well, and the cost rises when a large number of students choose to take an internet test. With the rapid growth of online learning over the last decade, online tests and checks have become a common way for academics to gather information. For example, MOOC like Cour-sera [1] and Ed x[7] typically require students to pass a series of online tests before receiving their final certificate. Meanwhile, institutions have continued to extend their online courses and administer online tests to their students. CLOSING COVID-19 has accelerated this trend, and many schools and institutions have converted to accepting online teaching and online testing. Still, how to proctor online tests in a convenient, effective, and reliable manner is a huge challenge. According to a survey, over 74 percent of students in 2013 believed this to be the truth with online exams, while approximately 29 percent of students admitted to cheating on online tests. These unsporting activities will erode online test trust, which is critical for Massive Open Online Courses program and educational institution to enhance online test use and use[3].

## 1.2 Problem Statements

Unlike traditional tests with proctoring on site, online tests have no face-to-face interaction. It brings trouble to online test demonstrations and various forms of cheating behavior can appear in online tests. Students, for representation, can type questions into a browser and search the Internet for answers. They can also use cell phones or internet chat apps to send messages to a third party (for example, friends) requesting assistance. It is difficult to detect such dishonest practices in online trials. Students are frequently asked to utilize cameras to monitor and record their activities during online tests in order to improve proctoring. When a large number of students take an online exam, however, more staff is required, and expenses rise. Using machine learning techniques to assess recorded footage of students during online testing, fully automated proctoring intends to reduce the potential efforts of physicians. It detects suspicious conduct automatically.

## 1.3 Aims and Objectives

- Proctor the activity of the test-taker: Because receiving screen and video sharing is a need for doing tests, an online task manager employing Al watches all actions and activities of the distant test-taker. Situations like the applicant's face being obscured or not visible at all, the presence of another person in the room, persistent disturbed movement, or the presence of an unapproved object are all regarded suspicious and are frequently highlighted during the testing process. The proctor has the authority to decide whether or not to mark the work that is being marked, such as cheating.

- Proctors safeguard the sanctity of the online proctored exam by monitoring the candidate's on-screen activity once the candidate's activity has been monitored using excellent video footage. Indeed, some cunning students try to persuade proctors to exercise their privilege to travel or to cheat on the screen by using online resources. Proctors may stay on top of things by using Secure Exam Browser, an incredible next-generation anti-cheating tool. On the system, the Lock browser uses full screen. As a result, a pupil cannot open a window or use any illegitimate application without permission.

## 1.4 Scope of Project

- Head Pose Analysis
- Mobile Phone Detection
- Person Counting
- Active Window Detection
- Gaze Estimation
- User Verification
- Multiple Monitor Detection

From those who provide an online state of the art, state of heart online authorization service to those who provide detailed student performance reporting re- ports, here are the 7 top industry software providers that provide secure and unbiased testing occurrence. Top 7 Software Providers for Test Proctoring:

- **Mercer or Mettle:** It is an online classification platform that includes online proctoring and Al-powered tools for administering safe virtual tests. With just one click, you can conduct incredibly secure, ascendant, and cost-effective online proctored tests.

- **Proctor U:** Proctor U is the world's most popular online exam proctoring service, providing a powerful, convenient, secure, and cost-effective alternative to traditional centered exams.

- **Proctor Track:** It is the most advanced online proctoring solution in the world, ensuring exam integrity. Proctor track is the only remote proctoring solution that includes both a lightweight desktop client for continuous identity verification and security monitoring and a browser extension for browser lock down and live monitor participation.

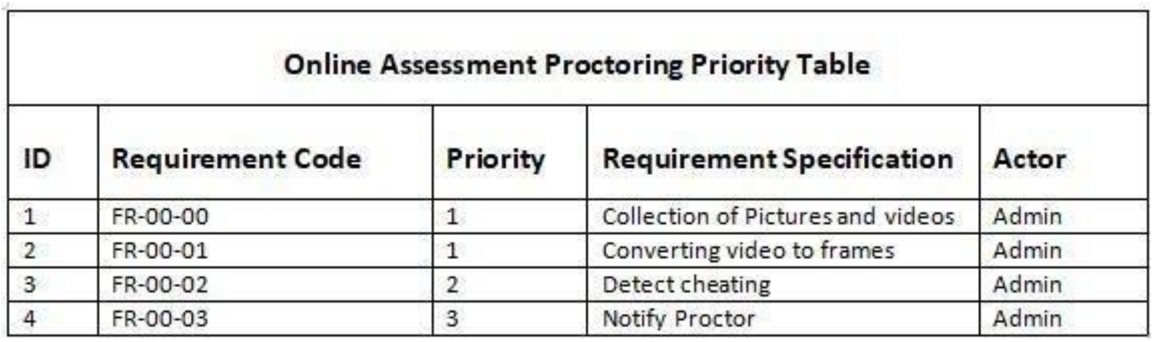
- **Proc-torio:** Proc-torio sees each proctored exam, whether online or in person, as a step toward students attaining their academic and career goals. That's why their technology allows anyone, at any time, to take proctored examinations.

- **Talviews:** Talviews assessment technology is designed for speed, and it offers formats like doubled superior exams that are safeguarded by sophisticated online monitor.

# CHAPTER 2

## Software Requirement Specification

A software system’s requirements specification provides a detailed explanation of the system’s functionality. Requirement specification contains functional and nonfunctional requirement which are given below of the project.



| Online Assessment Proctoring Priority Table | | | | |
|---|---|---|---|---|
| **ID** | **Requirement Code** | **Priority** | **Requirement Specification** | **Actor** |
| 1 | FR-00-00 | 1 | Collection of Pictures and videos | Admin |
| 2 | FR-00-01 | 1 | Converting video to frames | Admin |
| 3 | FR-00-02 | 2 | Detect cheating | Admin |
| 4 | FR-00-03 | 3 | Notify Proctor | Admin |

## Functional Requirement

- System shall allow Proctors to sign up
- System will automatically detect eye movement, head movement, number of person and mobile phone by webcam.
- System will also detect the active window of the student and number of monitors connected to the computer.
- Data will gathered in the form of pictures of various students in University. Data set will be created by these pictures.
- Proctors shall login
- Proctor shall see and monitor the students giving test
- Proctor shall approve or disapprove the students for test
- Proctors shall see warning if cheating is detected
- Proctors shall logout
- System shall allow the students to sign up
- Student shall login
- Students shall see the screen if proctor approved
- Students shall have the link of test
- Student shall see warning if there is any problem
- Student shall log out
- Admin shall login
- Admin shall manage information by user
- Admin shall logout

## Non-functional Requirement

- Design must be simple and user friendly
- System response must be high. So that, the student and proctor can effectively interact with each other without any trouble Database must be protected from unauthorized user.

## 2.1 Overall Description

Software is utilized in online exam proctoring to detect any instances of probable cheating or outside assistance. It can detect head movement, eye-gaze, and mobile phone recognition anytime different software is opened, or even if there is another person in the room. Any such instances are reported to the remote proctor, who is then able to review them.

You must ensure that the exam location is appropriate and that the technology you are using is dependable.

So, with that in mind, you'll require:

- A bright, prominent location where you can take the exam
- A computer with a dependable and clear webcam
- A stable internet connection that will not go down during your exam.

### 2.1.1 User Classes and Characteristics

Online Exam Proctoring consist of 6 classes.

#### 2.1.1.1 Login

This panel consist of main screen and features

Id

Password

Details

Verify

**Proctor**

This panel consist of main screen and features

Id

Name

Contact no

Address

Take exam

Proctoring

Verify students

Declare cheating

**Student**

This panel consist of main screen and features

Id

Name

Contact no

Address

Give exam

Check result

**Test**

This panel consist of main screen and features

Test id

Test name

Test duration

No of questions

**Verification**

This panel consist of main screen and features

Person counting

Head pose analysis

Gaze estimation

Mobile phone detection

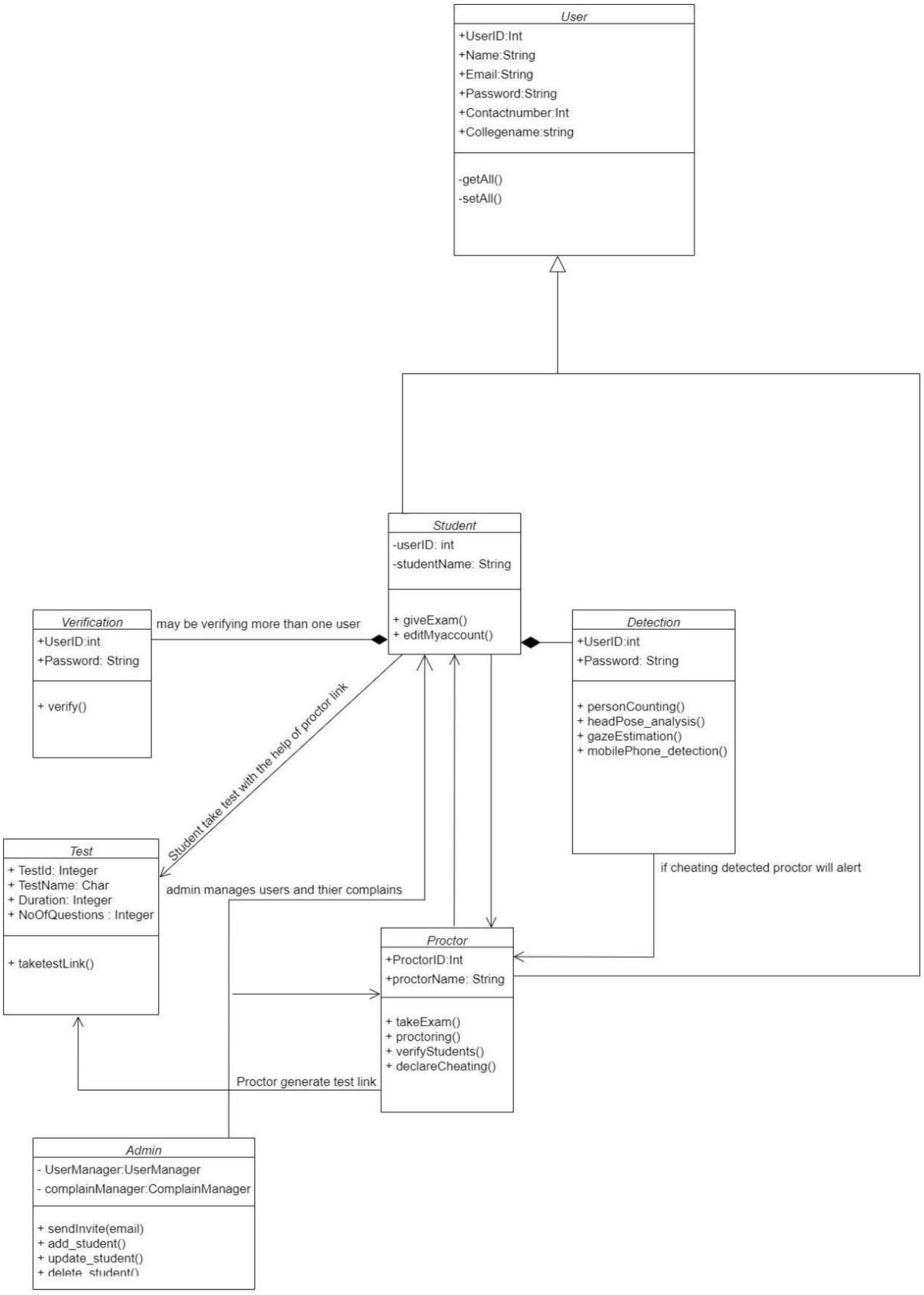
**Admin**

This panel consist of main screen and features

Sent invite (emails)

Manage users

# CLASS DIAGRAM



## 2.2 Final Deliverable of Project and Beneficiaries

The Final Deliverable of this Project is in the form of web. Main beneficiaries of this project will be teachers, professors, Universities, colleges, and school.

### 2.2.1 Final Deliverable

- Web based project

### 2.2.2 Beneficiaries

- User
- Teachers
- Professors
- University Students (BULC)
- Schools
- Colleges

## 2.3 Optional Scope

We are going to use a set of modern tools. Our platform is based on web, and we are going to use following tools:

- **Python:** Python is a high-level programming language that is free and open-source. It is mostly utilized in the construction of websites, games, and programming languages. There is a sizable developer community. This project will use the most recent version of Python (3.97).

- **Django:** Django is a Python language web framework that

encourages rapid and clean development. You can focus on writing your app in Django because it's open-source. Django(3.27) latest version will be used in this project.

- **React**: React is a front-end JavaScript toolkit for creating website UI components. The Facebook company looks after it. This project will use the most recent version of React(17).

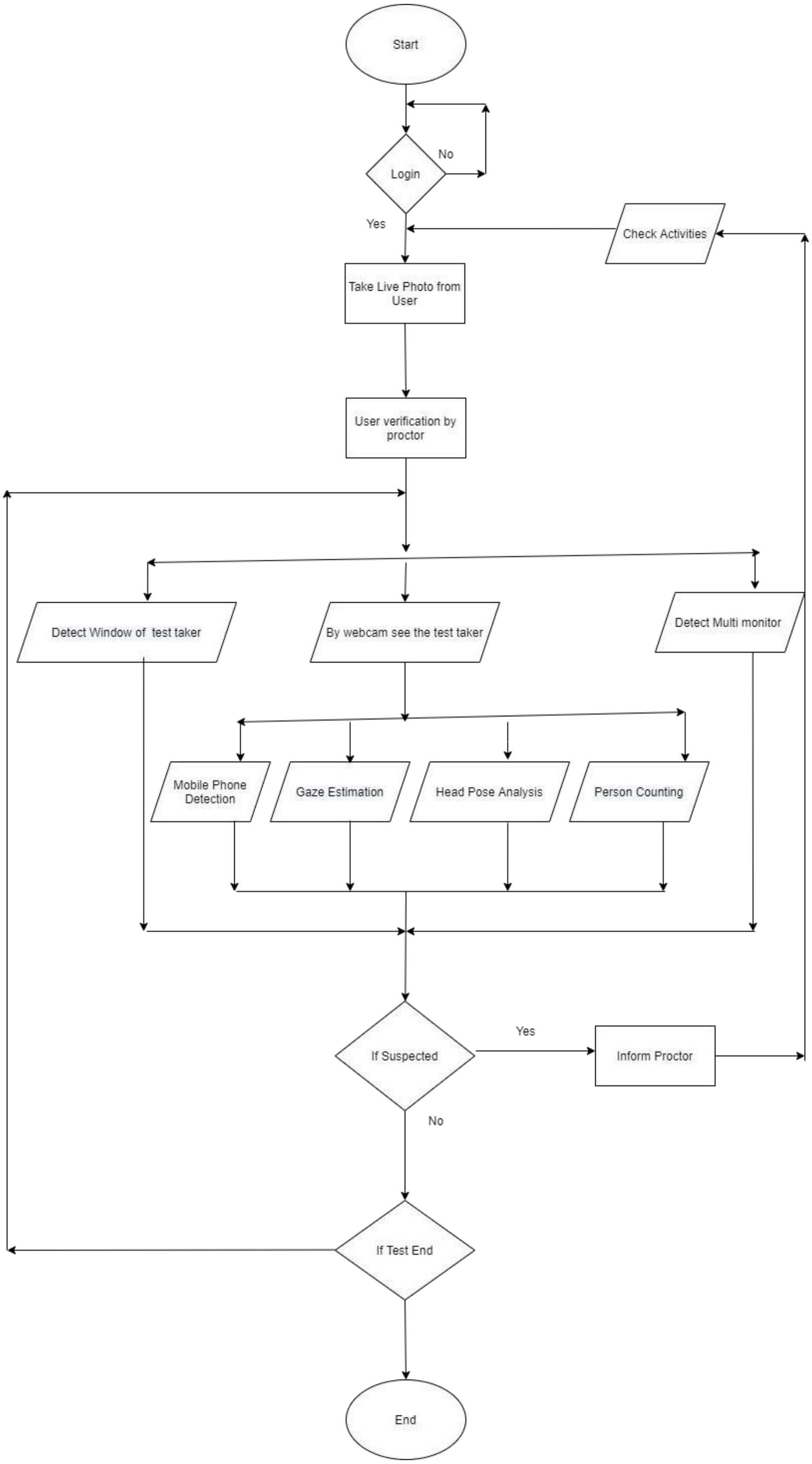
- **React Native**: Facebook, Inc. produced React Native, a free and open-source UI software framework. It allows developers to leverage the React framework alongside native platform capabilities to create applications. This project will use the most recent version of React Native (0.60).

  - Laptop (core i5, windows 10, Intel HD Graphics 4000)
  - Visual Studio Code V 1.62
  - Internet
  - PyCharm V 2021.2.3
  - Django(3.27)
  - React(17)

## 2.4 Design and implementation

For making this web based project Django is a Python language web framework that encourages rapid and clean development. You can focus on writing your app in Django because it's open-source. Django(3.27) latest version will be used in this project.

We are going to use a set of modern tools. Our platform is based on web.


Start
Login
No
Yes
Check Activities
Take Live Photo from User
User verification by proctor
Detect Window of test taker
By webcam see the test taker
Detect Multi monitor
Mobile Phone Detection
Gaze Estimation
Head Pose Analysis
Person Counting
If Suspected
Yes
Inform Proctor
No
If Test End
End

## 2.5 External Requirements

### 2.5.1 Hardware

We use laptop or computer for the implementation of this web base project

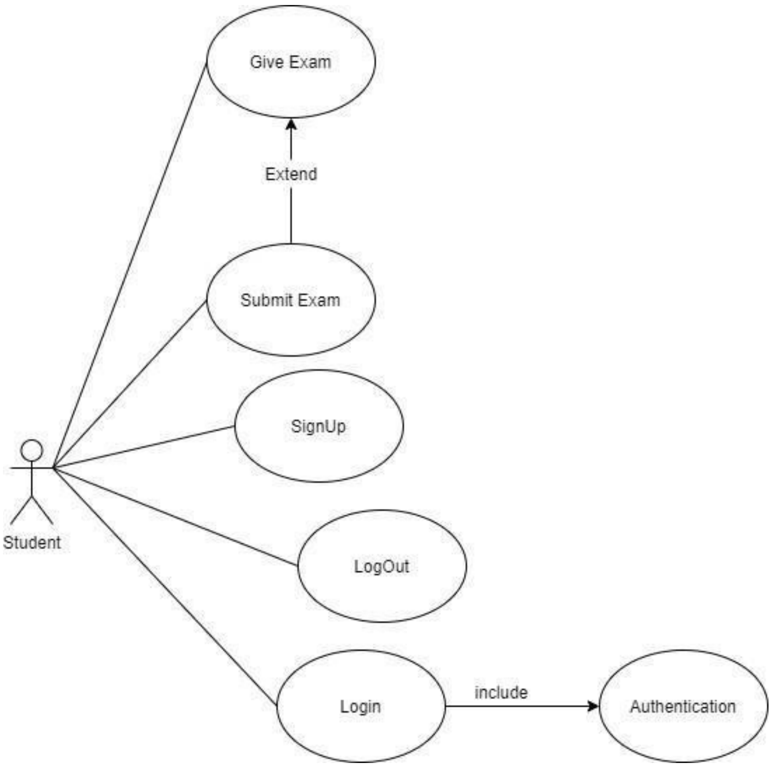
- Laptop (core i5, windows 10, Intel HD Graphics 4000)

### 2.5.2 Software

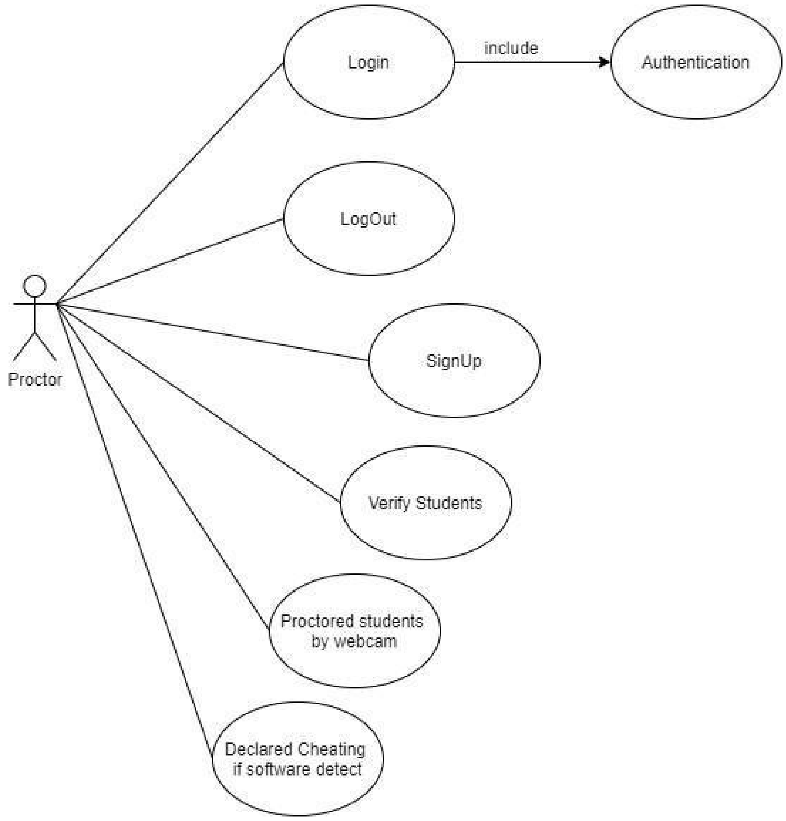
- Visual Studio Code V 1.62
- PyCharm V 2021.2.3
- Django(3.27)
- React(17)

## 2.6 Use case Diagram

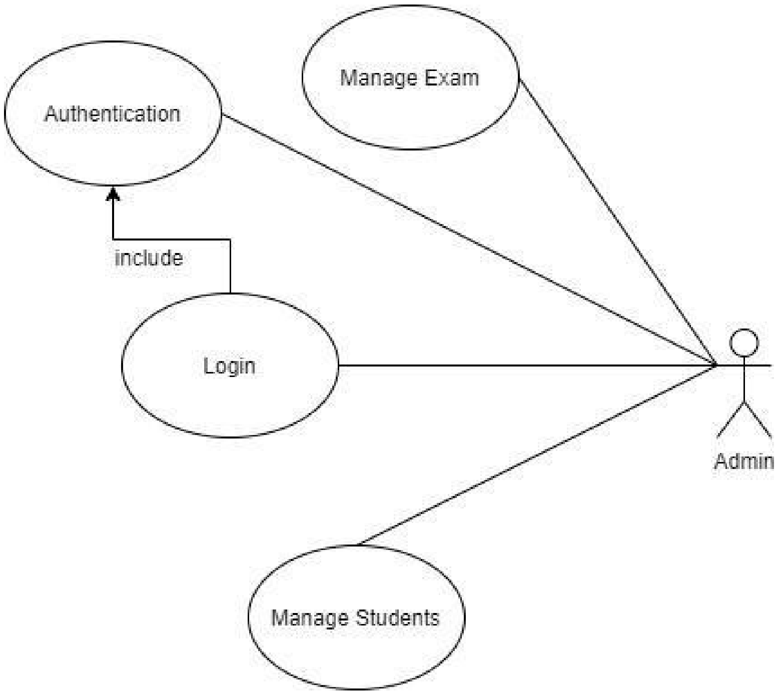
**For STUDENT**



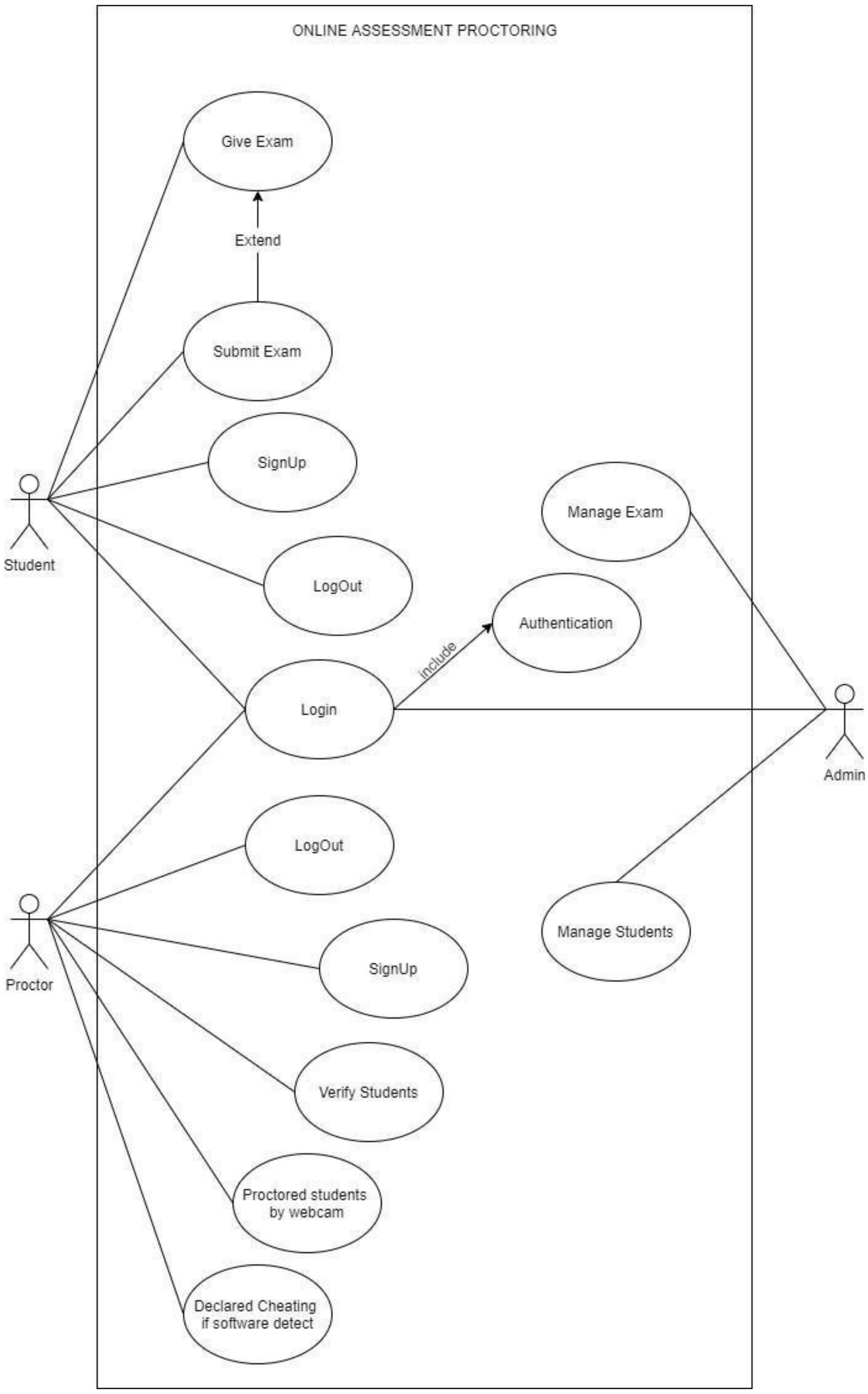
**For PROCTOR**


Login
include
Authentication
LogOut
Proctor
SignUp
Verify Students
Proctored students by webcam
Declared Cheating if software detect

**For ADMIN**



## For WHOLESYSTEM



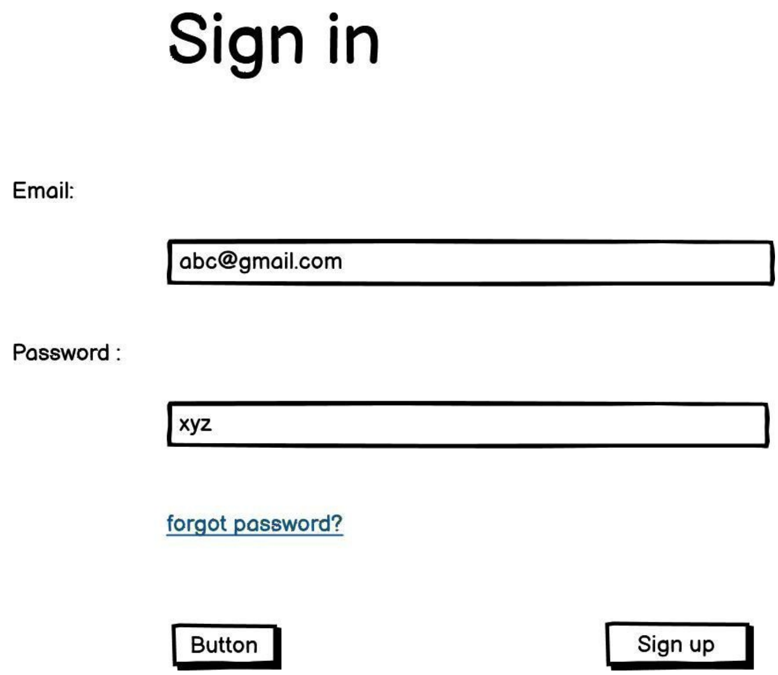
# CHAPTER 3

# DESIGN AND METHODOLOGY

## DESIGN SPECIFICATION

## 3.1 Wire frames

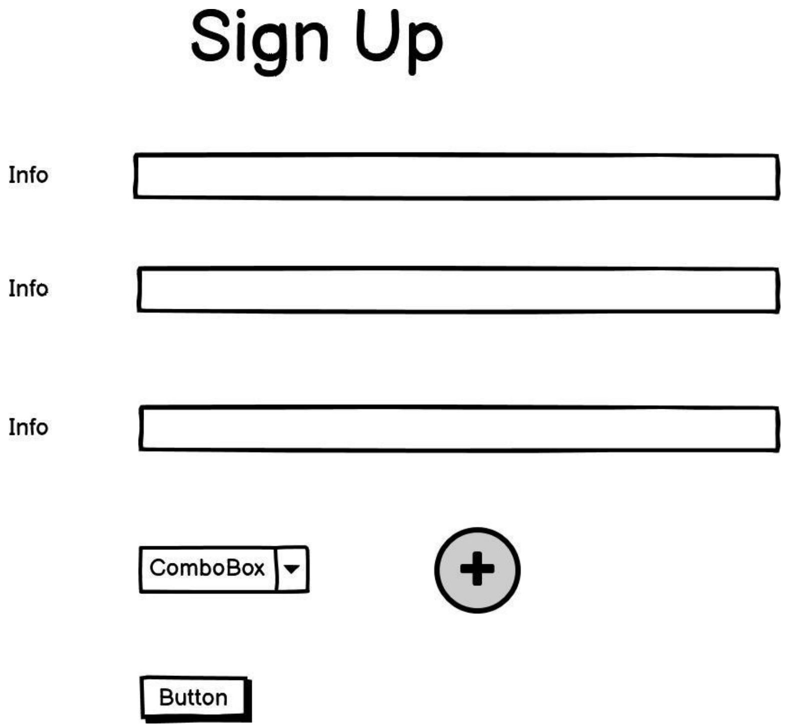
### FOR SIGN IN



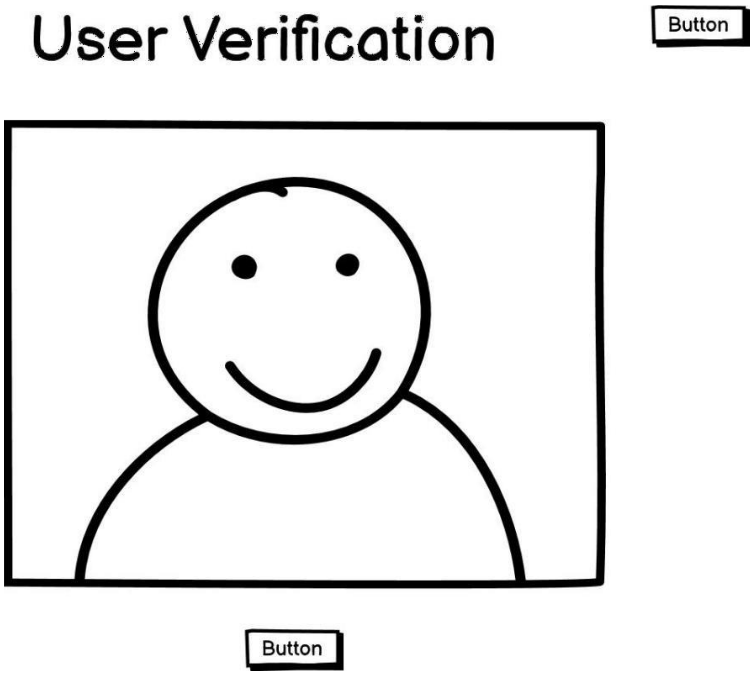
Email: abc@gmail.com

Password : xyz

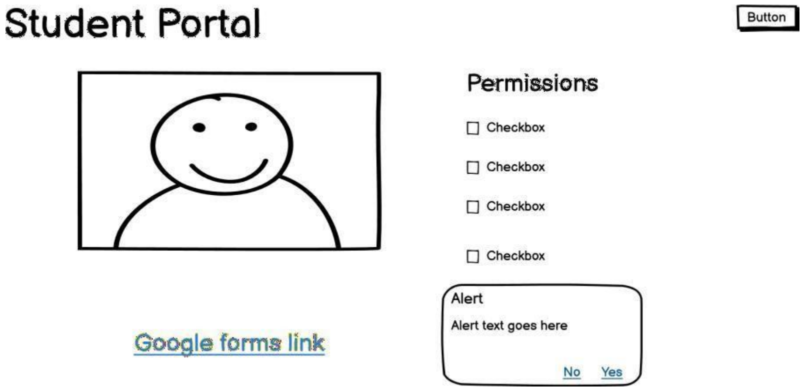
**FOR SIGN UP**



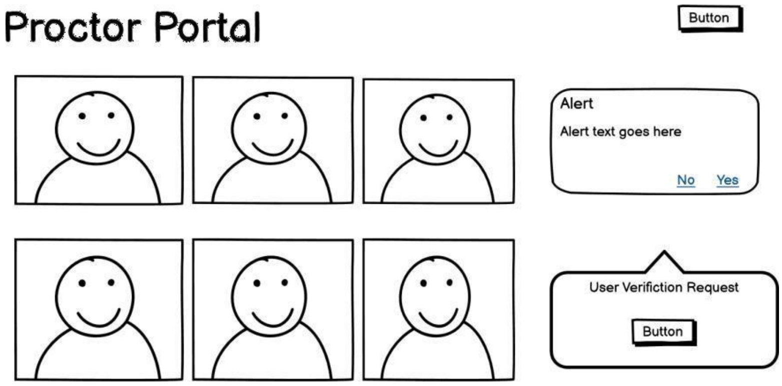
**FOR USER VERIFICATION:**



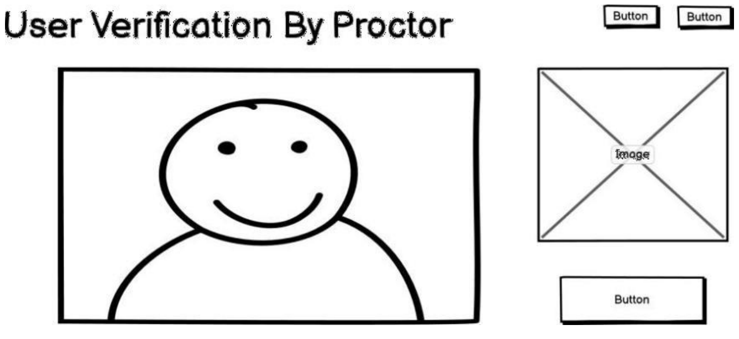
**FOR STUDENT**



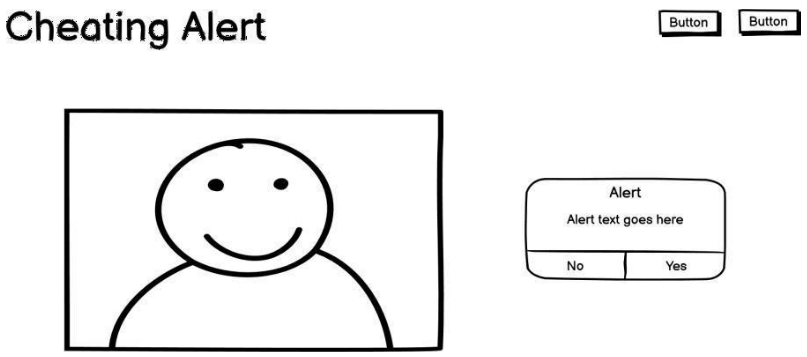
**FOR PROCTOR**



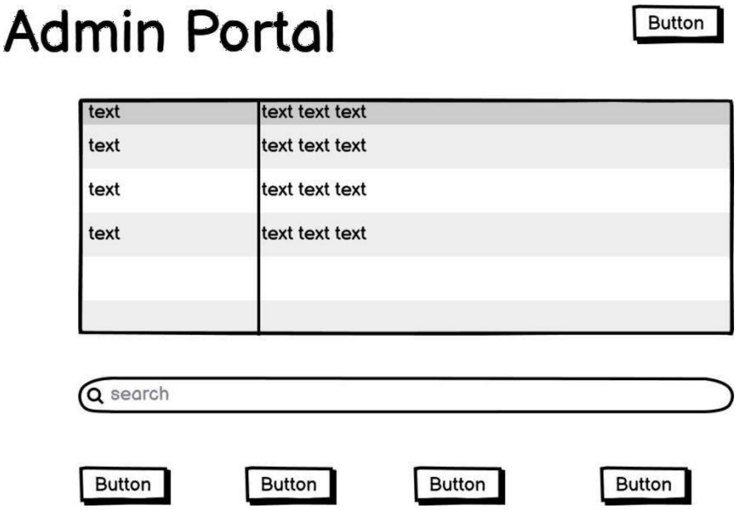
# FOR PROCTOR 2



**FOR CHEATING**



**FOR ADMIN**



| text | text text text |
|---|---|
| text | text text text |
| text | text text text |
| text | text text text |
| | |
| | |

### 3.1.1 USE CASE 1:

| Name | Sign Up |
|---|---|
| Actor | Proctor/Students |
| Description | Actor must be registered before giving or taking the exam |
| Pre Condition | This application must have a sign up form |
| Basic Flow of events | Data is fetched from this form and is saved into database using uery |
| Alternative Flow of events | Data is not fetched successfully om this form because of invalid ntry so, data is not saved in the atabase |
| Post Condition | Actor is successfully registered in the system |

**Use Case 2:**

| Name | Login |
|---|---|
| Actor | Admin/Students/Proctor |
| Description | Actor must login before giving,taking and managingexams |
| Pre Condition | Actor must be registered in database |
| Basic Flow of events | Data is fetched from this form and this data is matched with atabase data |
| Alternative Flow of events | Data is fetched from this form but this data is not match with atabase data. Or user enter in- lid data syntax. So an error has ccurred |
| Post Condition | Actor is successfully logged in |

**Use Case 3:**

| | |
|---|---|
| Name | Logout |
| Actor | Admin/Students/Proctor |
| Description | Actor sign out of the system |
| Pre Condition | Actor must be signed in system |
| Basic Flow of events | N/A |
| Alternative Flow of events | N/A |
| Post Condition | Actor is successfully logged ut |

**Use Case 4:**

| | |
|---|---|
| Name | Give Exam |
| Actor | Student |
| Description | Actor can give exam |
| Pre Condition | Actor must be registered in database and must be sign in in stem |
| Basic Flow of events | Data of registered users is fetched and displayed to actor |
| Alternative Flow of events | If no user is registered in the stem, nothing will be displayed to tor |
| Post Condition | Actor successfully sees the exam on screen and give the exam |

**Use Case 5:**

| | |
|---|---|
| Name | Submit Exam |
| Actor | Student |
| Description | Actor can submit exam after completing |
| Pre Condition | Actor must be logged in and must finish their exam |

| | |
|---|---|
| Basic Flow of events | Data from this form is fetched and saved in the database |
| Alternative Flow of events | Data is not successfully fetched because of exam timeout or be- use of bad internet connection |
| Post Condition | Exam is successfully saved |

**Use Case 6:**

| | |
|---|---|
| Name | Manage Exams |
| Actor | Admin |
| Description | Actor can see all the users |
| Pre Condition | Actor must be signed in system |
| Basic Flow of events | Data of added users is fetched and displayed to actor |
| Alternative Flow of events | If no user is added in the system, nothing will be displayed to actor |
| Post Condition | Actor successfully sees all details |

**Use Case 7:**

| | |
|---|---|
| Name | Manage Students |
| Actor | Admin |
| Description | Actor can add, delete and update students |
| Pre Condition | Actor must be logged in and there must be a form for add new udents and delete,update list. |
| Basic Flow of events | Data from this form is fetched and saved in the database |
| Alternative Flow of events | Data is not successfully fetched rom this form because of invalid ata entry |
| Post Condition | New student is successfully added, updated or deleted |

**Use Case 8:**

| | |
|---|---|
| Name | Authentication |
| Actor | Admin |
| Description | Actor can confirm the user that is student/proctor |
| Pre Condition | Actor must be logged in and user to be verify must exist |
| Basic Flow of events | User is to be verify and after rification user now logged in |
| Alternative Flow of events | Data is not successfully fetched rom this form because of invalid ata entry |
| Post Condition | User is verified successfully |

**Use Case 9:**

| | |
|---|---|
| Name | Verify Students |
| Actor | Proctor |
| Description | Actor can verify the students by seeing them by camera |
| Pre Condition | Actor must be logged in |
| Basic Flow of events | Proctor verify the students by seeing them before giving exam |
| Alternative Flow of events | Data is not successfully displayed to actor Because of low quality mera |
| Post Condition | User is verified successfully |

**Use Case 10:**

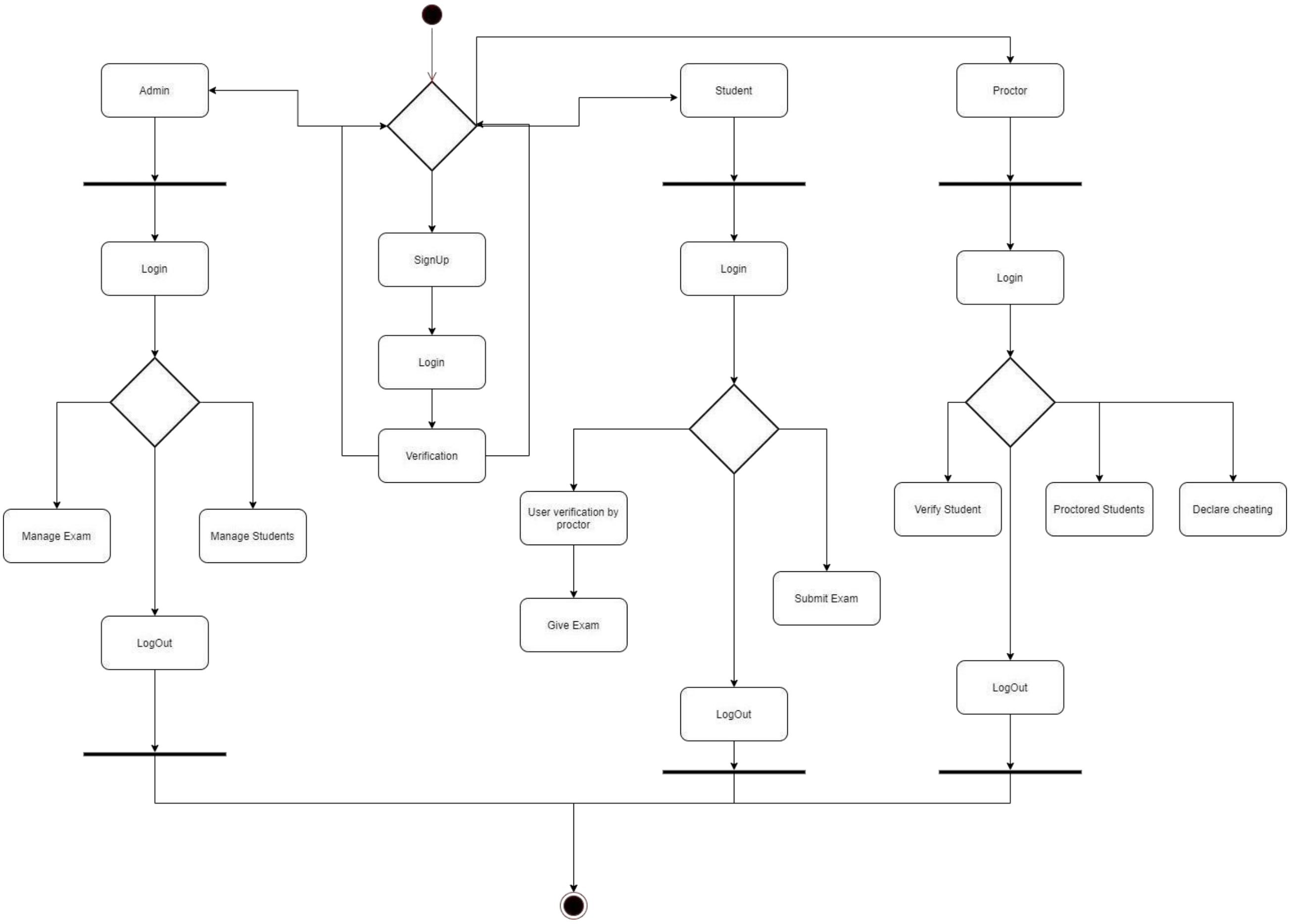
| | |
|---|---|
| Name | Proctored Students by webcam |
| Actor | Proctor |
| Description | Actor can proctor students during am |
| Pre Condition | Actor must be logged in |
| Basic Flow of events | Students are displayed to the actor through camera |

| | |
|---|---|
| Alternative Flow of events | Students are not properly displayed to the actor |
| Post Condition | Actor proctored students successfully |

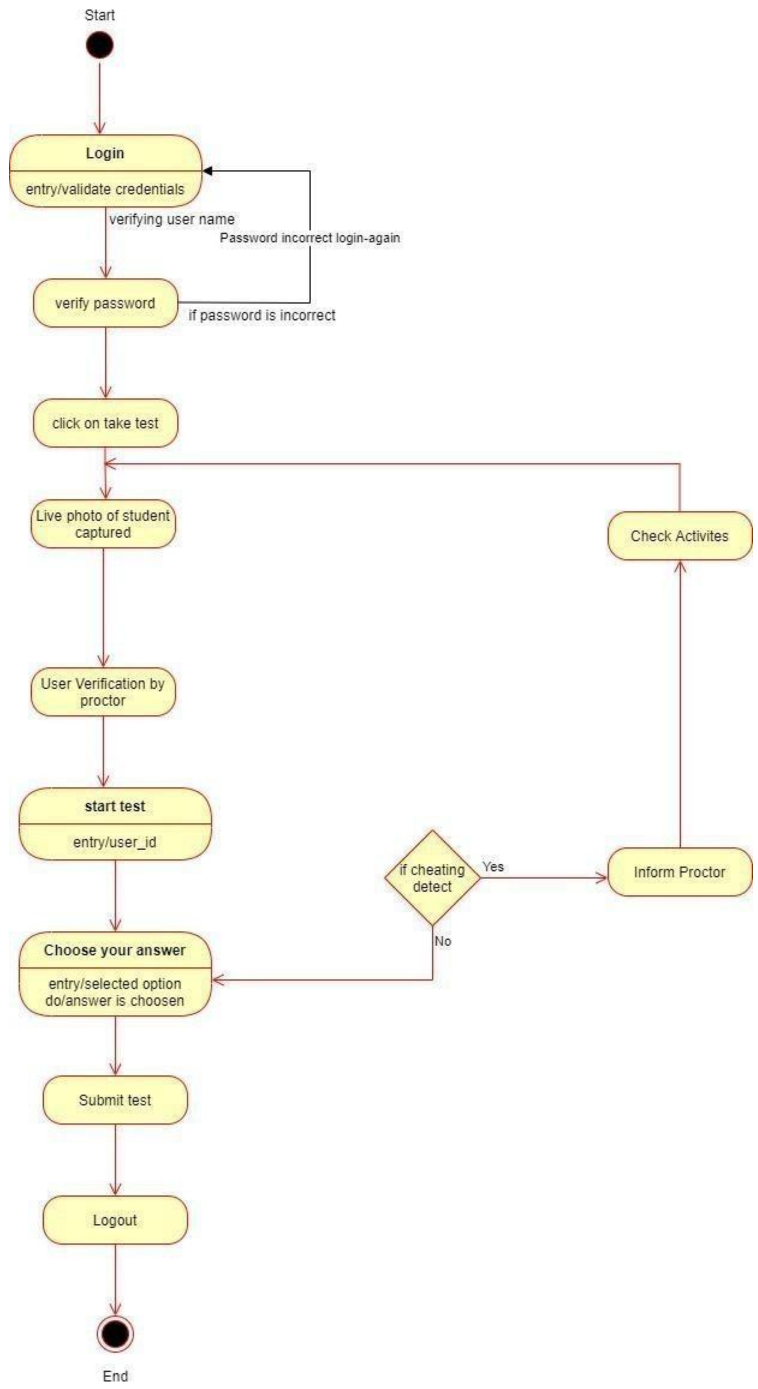
**Use Case 11:**

| | |
|---|---|
| Name | Declared cheating if detect |
| Actor | Proctor |
| Description | Actor can declare that the etected student is cheating |
| Pre Condition | Actor must be logged in |
| Basic Flow of events | Actor is informed by the software that student behaviour is spicious and actor can declare the udent is cheating |
| Alternative Flow of events | Actor is informed by the software that student behaviour is spicious, but actor not declare it eating |
| Post Condition | If suspicious activity is found, hen software detect and actor eclare t cheating |

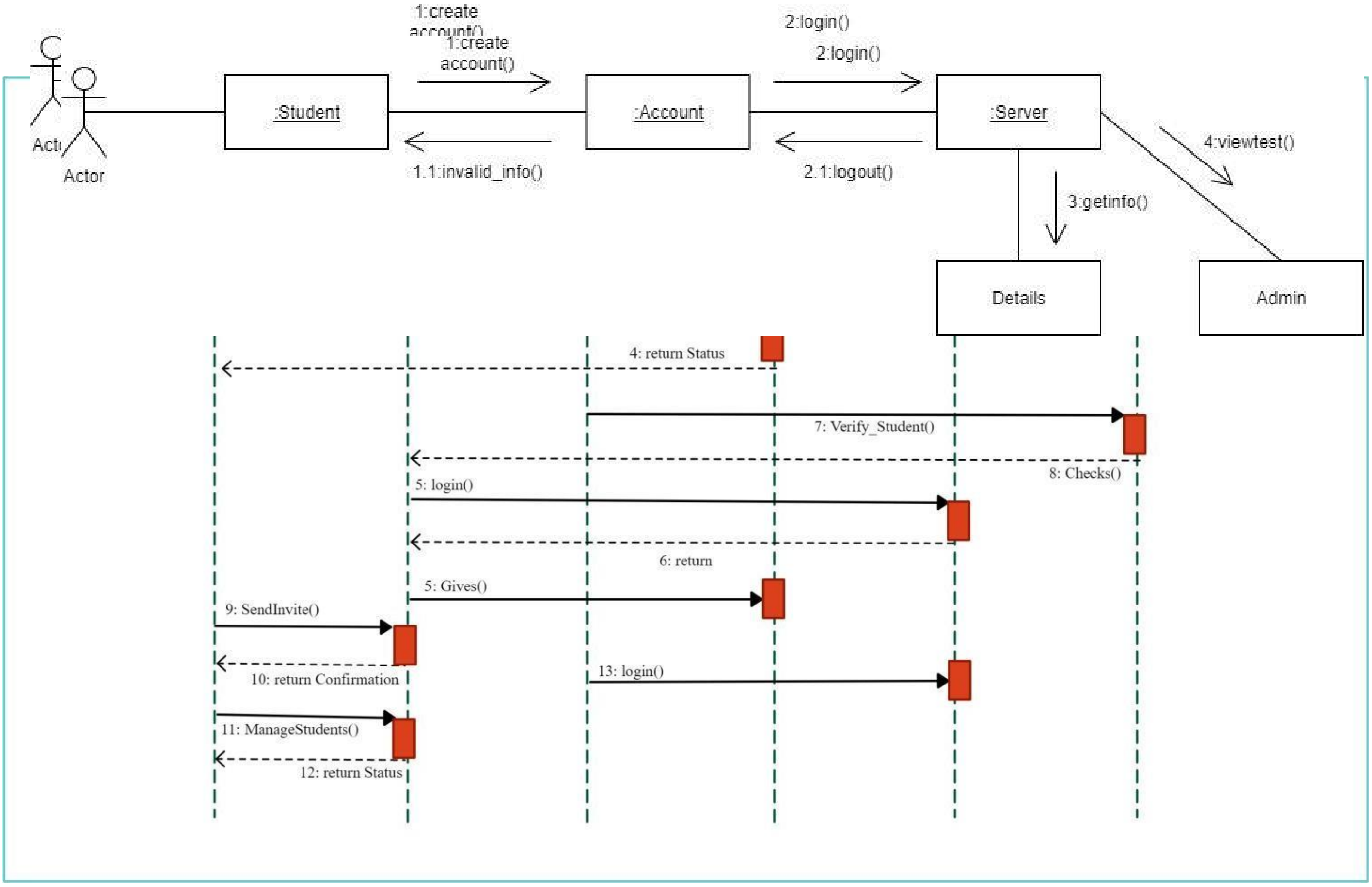
## 3.2 ACTIVITY DIAGRAM:



## 3.3 STATE DIAGRAM

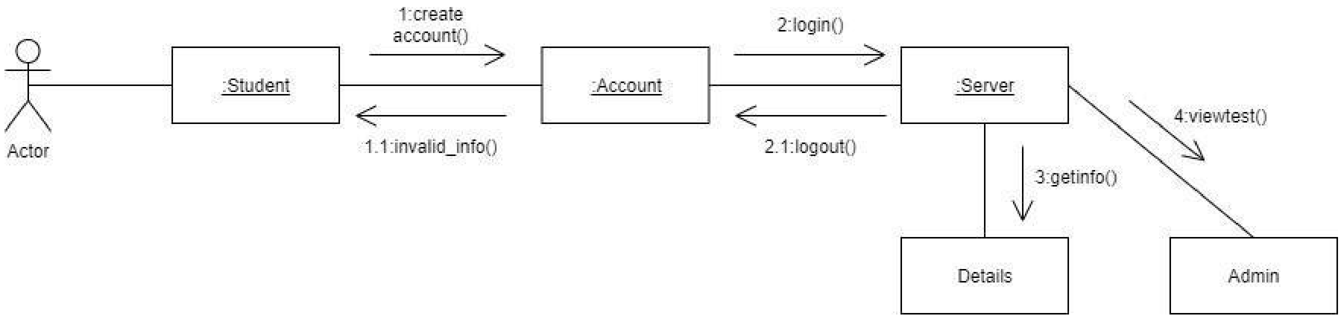


## 3.3.1 SEQUENCE DIAGRAM

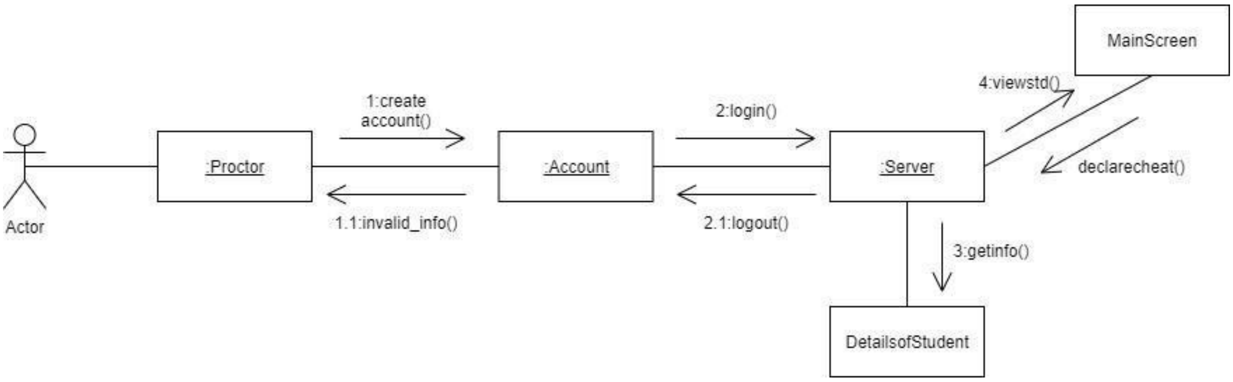


# COLLABORATION DIAGRAM:

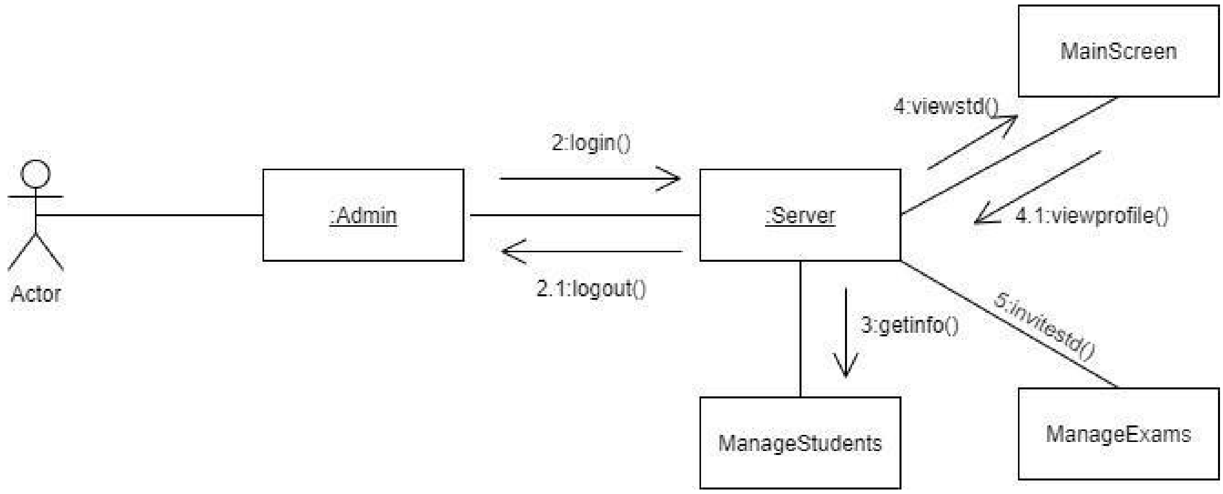
## FOR STUDENT



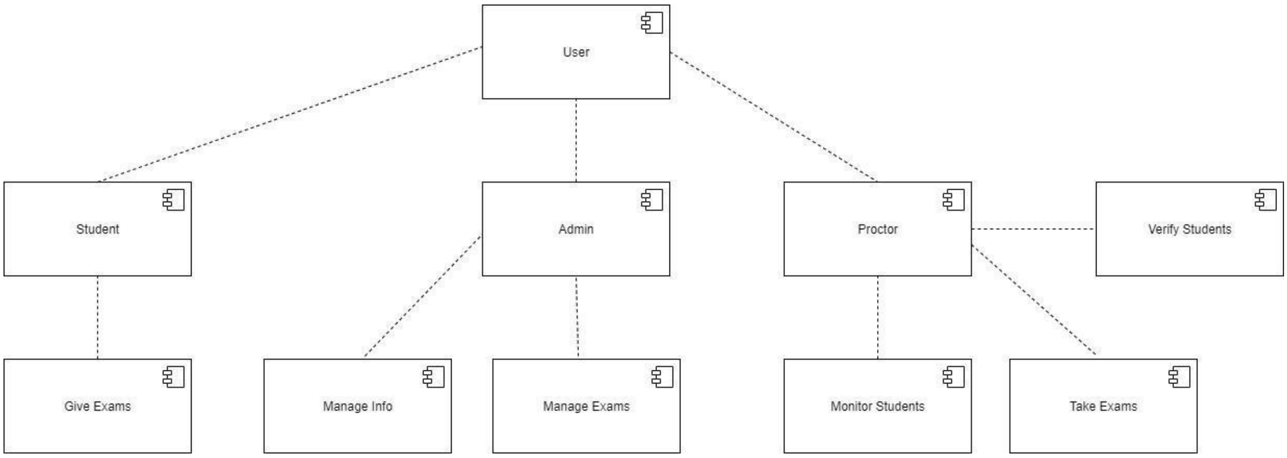
## FOR PROCTOR



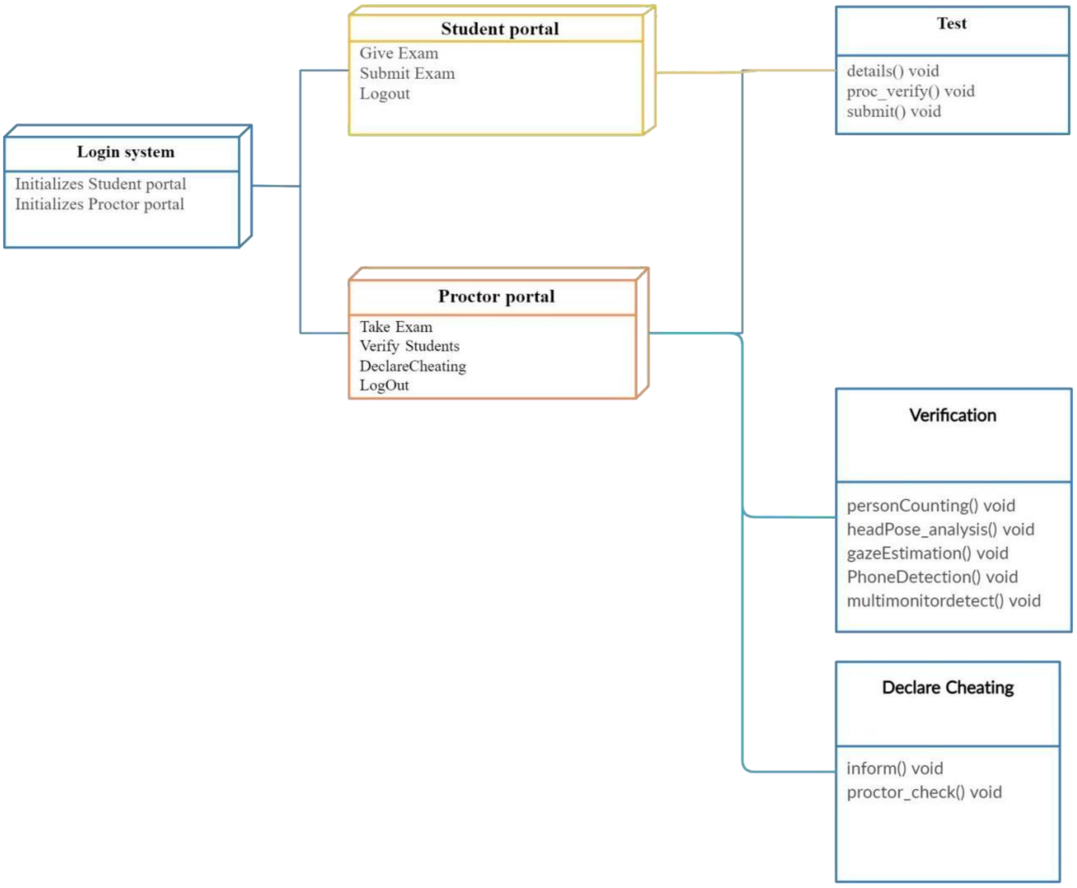
## FOR ADMIN



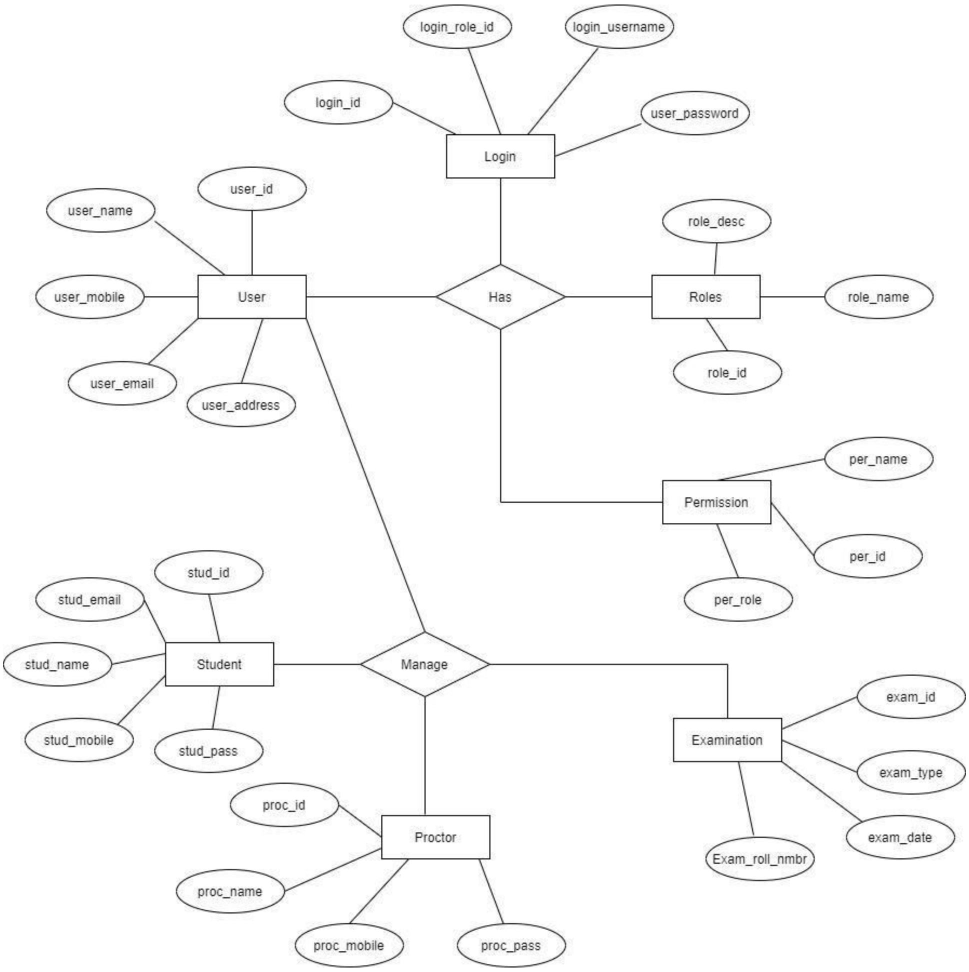
## COMPONENT DIAGRAM



## DEPLOYMENT DIAGRAM



# ER DIAGRAM


login_role_id
login_username
login_id
user_password
Login
user_id
user_name
role_desc
user_mobile
User
Has
Roles
role_name
user_email
user_address
role_id
per_name
Permission
per_id
stud_id
per_role
stud_email
stud_name
Student
Manage
exam_id
stud_mobile
stud_pass
Examination
exam_type
proc_id
Proctor
exam_date
Exam_roll_nmbr
proc_name
proc_mobile
proc_pass

# CHAPTER 4

# DATA AND EXPERIMENTS (and/or IMPLMENTATION)

The goal of introducing a variety of testing methodologies into your development process is to ensure that your software can be used in a number of situations and platforms. The most prevalent types of testing are functional and non-functional testing. Functional testing is the process of comparing an application to its business requirements. It includes all test techniques targeted at ensuring that each software component functions as expected, depending on use cases provided by the design team or business analyst. The following tests are usually performed in the following order:

- Individual testing
- Testing for integration
- System evaluation
- Testing for acceptance

All test types focusing on the operational characteristics of a piece of software are included in non-functional testing methods. Among them are:

- Performance evaluation
- Testing for security
- Testing for usability
- Testing for compatibility

## 4.1 Individual testing

We test each of our components separately during unit testing. It's a great way to ensure code quality by allowing engineers to identify errors early in the development process. As a result, whenever we created a new unit, we tested it individually to ensure that it was error-free. Unit testing was done by all of the group members, and various entities tested different units.

Table 4.1: Login form

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | Enter the name and password | Home page should display | Successful |
| 2 | Invalid detail | Home page will not display | Successful |

Table 4.2: Sign Up form

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | Enter name, email and password | Sign Up Successful | Successful |
| 2 | Invalid details | display error | Successful |

Table 4.3: Logout form

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | Click Logout | User should be logged out | Successful |

Table 4.4: Give Exam

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | Student can give exam through link | Successfully sees theexam on screen and give the exam | Successful |

Table 4.5: Manage Students

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | Can be add, delete and update students | student is successfully added, updated or deleted | Successful |

Table 4.6: Eye Movement Detection

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | If student eyes move left,right | declare that the detected student is cheating if limit reached | Successful |

Table 4.7: head Movement Detection

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | If student head moving Re-repeatedly | declare that the detected student is cheating | Successful |

Table 4.8: Person Counting Detection

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | If more than 1 student | declare that the detected student is cheating | Successful |

Table 4.9: Mobile Phone Detection

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | If student using mobile phone | declare that the detected student is cheating | Successful |

Table 4.10: Multi-Monitor Detection

| Sr. No | Test case | Expected Result | Test result |
|---|---|---|---|
| 1 | If student using multi-screens | declare that the detected student is cheating | Successful |

## 4.2 Testing for integration

After we've thoroughly examined the units and their individual capabilities, it's time to assess how well they'll function together. In the second level of testing, integration testing, we test each of our individual components in combination. The group built various units, which were then integrated and tested to ensure that the system performed as planned.

## 4.3 System Evaluation

System testing is a sort of testing that examines the completeness and integration of a software product. The end-to-end system specifications are evaluated using a system test. Software is usually simply one component of a larger computer system. This is a closed-loop evaluation.This black box method is used to assess the entire and integrated system to verify it meets the given standards. The software is thoroughly tested from beginning to end. This testing technique is used on our project to see if it fits all of the functional requirements. These are they:

- Sign Up
- Login
- Joining the Meeting
- Cheating Detecting
- Informing the Proctor

## 4.4 Testing for Acceptance

It is a type of formal testing that is based on user needs and function processing. Acceptance testing involves checking our system's compliance with functional and non-functional requirements as well as its overall performance. It is founded on the premise that your acceptance tests are responsible for determining experimentally whether your website functions as advertised to its users. Users are asked to evaluate websites based on a variety of characteristics. We've used tests in a variety of situations, such as when a student is delivering an exam and utilizing a cellphone, the proctor should be warned that the student is cheating.

## 4.5 Development Tools

The development details and techniques we used in our project are given below:

## 4.6 Techniques and Tools

The techniques and tools used in our system are given below:

- Language: Python[20]
- Operating System: Windows and Linux
- Library: D lib[13]
- For development environment: Visual studio code[4]
- Front-end: HTML5, CSS, Bootstrap3[9]
- Back-end: Python
- Database: SQLITE[12]
- Validation: PHP and JavaScript[16]
- Notepad++ : To implement the project we have used HTML (environment: Notepad++[19]) for its ease of use.

The web page's content is organized using HTML. CSS is used to give a web page's content some style. HTML offers display information for various tags to the browser. CSS adds styling to the same elements, enriching the data.

Tags in HTML Bootstrap is simple to set up and use, with a large number of components an excellent grid system, and styling for a variety of HTML components, including typography to the buttons Django is a high-level Python web framework that allows for quick development website construction that is secure and easy to maintain Built by seasoned programmers Django takes care of a lot of the headaches that come with web development.

## 4.7 Development Details:

We created a website theme that was simple to use and enjoyable to connect with. The user is greeted with a welcome screen that includes the sign-in and login choices If the user has previously registered, the user will be able to log in will immediately log in. Otherwise, the user will be prompted to login after signing in. A user after a successful login, the user's profile is displayed. If the user is a supervisor the proctor portal will open, and the student portal will open if the user is a student open.

Users can choose from a variety of alternatives. Permissions can be granted by the user. The through the link, the user can provide feedback. The user can logout after the assessment is completed. The proctor can verify pupils and receive activity pop-ups.

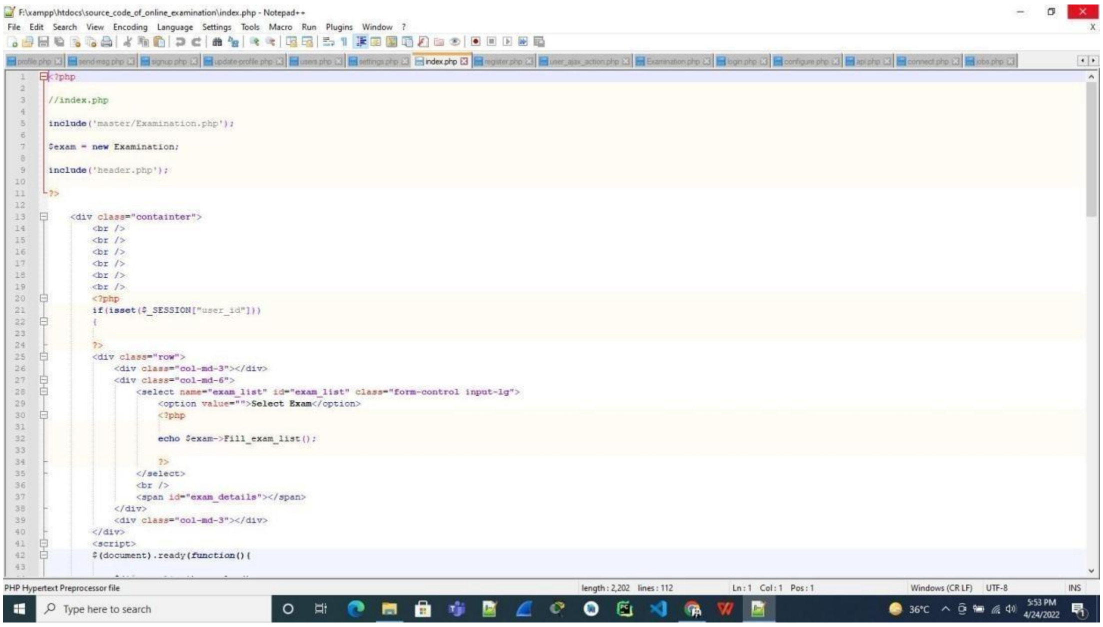
HTML 5, CSS, and Bootstrap are used to generate the web pages. We put the data in a SQLITE database because it can operate almost everywhere. SQLite is a database management system that runs on Windows, Mac OS, Linux, iOS, Android, and other platforms.

For detection, we employed real-time detection code and various models. The detection is fairly accurate. If there is any action, a pop-up will appear, and screenshots will be sent to the proctor. We used socket programming to connect the front-end (client) and back-end (server) (server). For detection, we have real-time detection and various re-trained models (yolo)[11] as well as built-in libraries like ’d lib’[2].

### 4.7.1 Code Snapshot

Screen Shots of our project(Online Assessment Proctoring) are following:

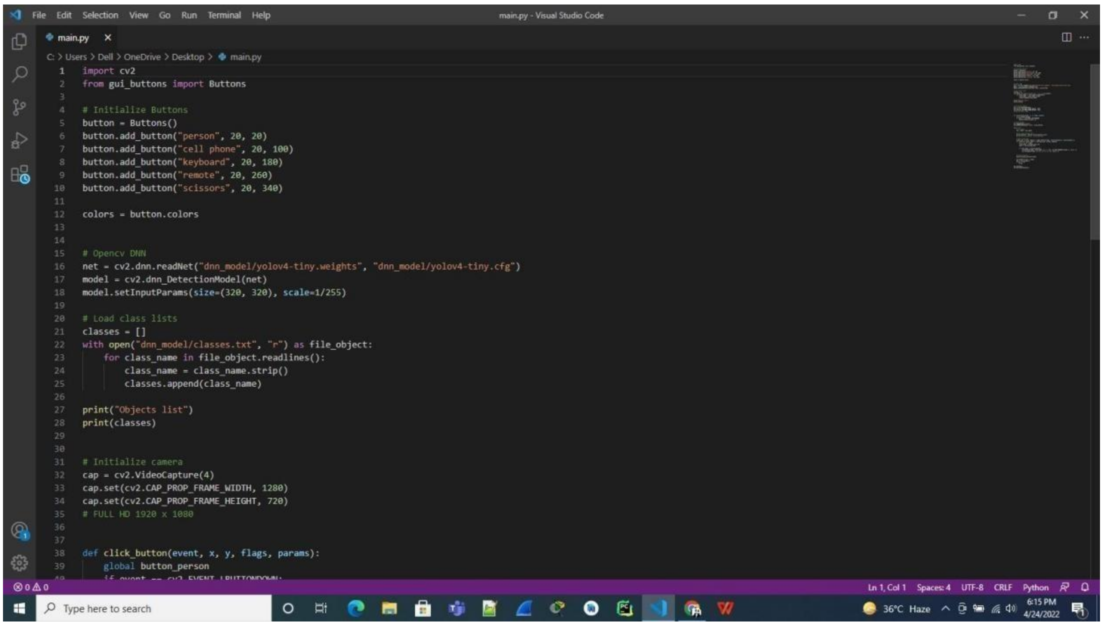
**Front end**



```
<?php

//index.php

include('master/Examination.php');

$exam = new Examination;

include('header.php');

?>

    <div class="containter">
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <?php
        if(isset($_SESSION["user_id"]))
        {

        ?>
        <div class="row">
            <div class="col-md-3"></div>
            <div class="col-md-6">
                <select name="exam_list" id="exam_list" class="form-control input-lg">
                    <option value="">Select Exam</option>
                    <?php

                    echo $exam->Fill_exam_list();

                    ?>
                </select>
                <br />
                <span id="exam_details"></span>
            </div>
            <div class="col-md-3"></div>
        </div>
        <script>
        $(document).ready(function(){
```

Figure 4.1: Front End

**Back end**



```python
import cv2
from gui_buttons import Buttons

# Initialize Buttons
button = Buttons()
button.add_button("person", 20, 20)
button.add_button("cell phone", 20, 100)
button.add_button("keyboard", 20, 180)
button.add_button("remote", 20, 260)
button.add_button("scissors", 20, 340)

colors = button.colors


# Opencv DNN
net = cv2.dnn.readNet("dnn_model/yolov4-tiny.weights", "dnn_model/yolov4-tiny.cfg")
model = cv2.dnn_DetectionModel(net)
model.setInputParams(size=(320, 320), scale=1/255)

# Load class lists
classes = []
with open("dnn_model/classes.txt", "r") as file_object:
    for class_name in file_object.readlines():
        class_name = class_name.strip()
        classes.append(class_name)

print("Objects list")
print(classes)


# Initialize camera
cap = cv2.VideoCapture(4)
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
# FULL HD 1920 x 1080


def click_button(event, x, y, flags, params):
    global button_person
```

File Edit Selection View Go Run Terminal Help
main.py - Visual Studio Code
main.py
C: > Users > Dell > OneDrive > Desktop > main.py
⊗ 0 ⚠ 0
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python
Type here to search
36°C Haze
6:15 PM
4/24/2022

Figure 4.2: Back End

# CHAPTER 5

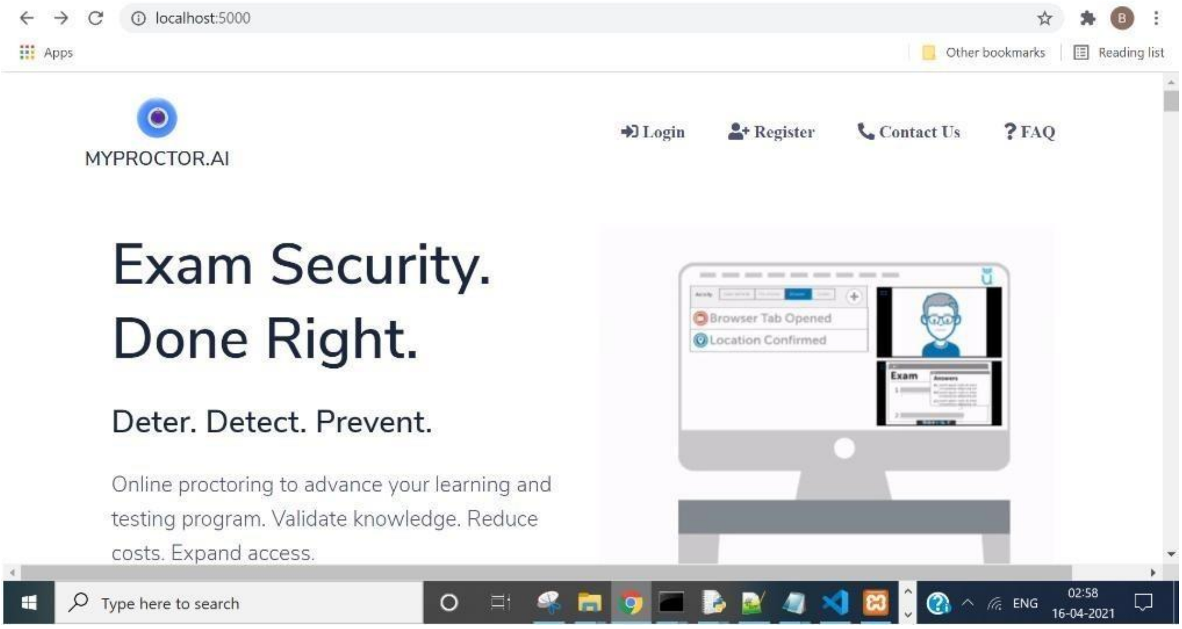
# RESULTS AND DISCUSSIONS (or USER MANUAL)

A successful trial run of the project has been completed. While the setup was done in accordance with the instructions. Dummy user registration and login has been completed successfully. After logging in or signing up, the user can change his profile and take a test. While taking the exam, our software will check for cheating in the background, and if cheating is found, our programme will display a pop-up notification on the proctor's screen. Search and Filter are both working properly. In the contact us page, users can also send their recommendations to the admin. All work was completed in a systematic manner, with seamless operations.

## 5.1 Web Snapshots

The following screen shots have been taken from the demo of Web.

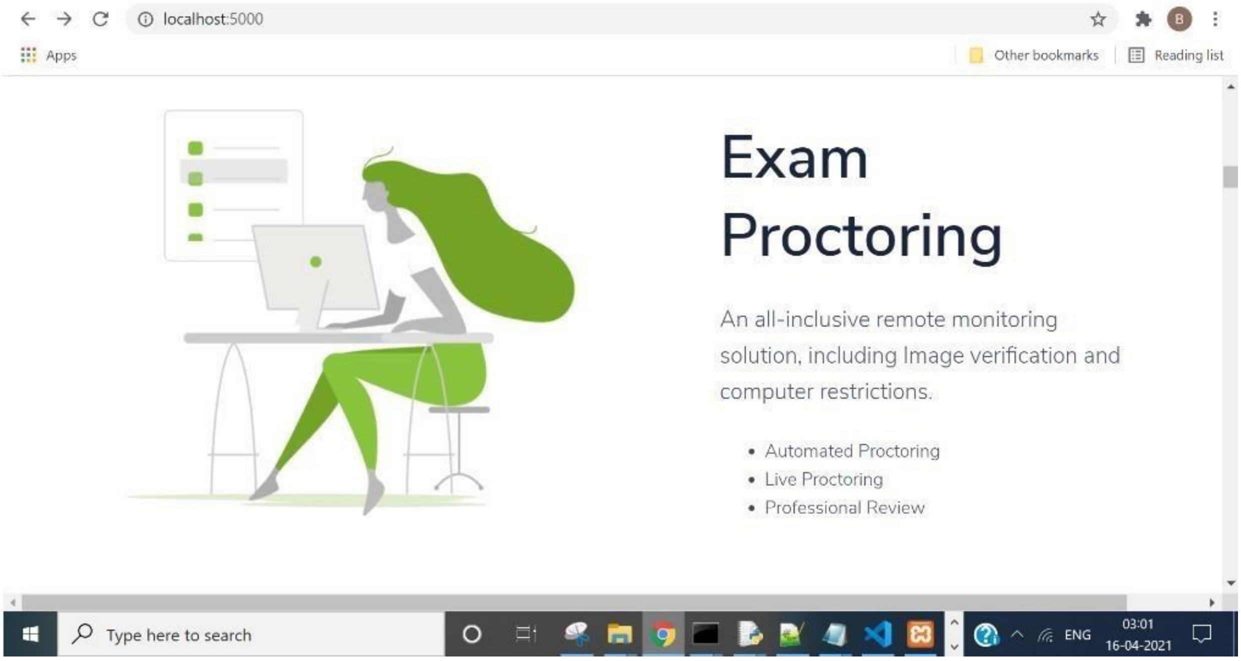
### Home Page



Exam Security.
Done Right.

Deter. Detect. Prevent.

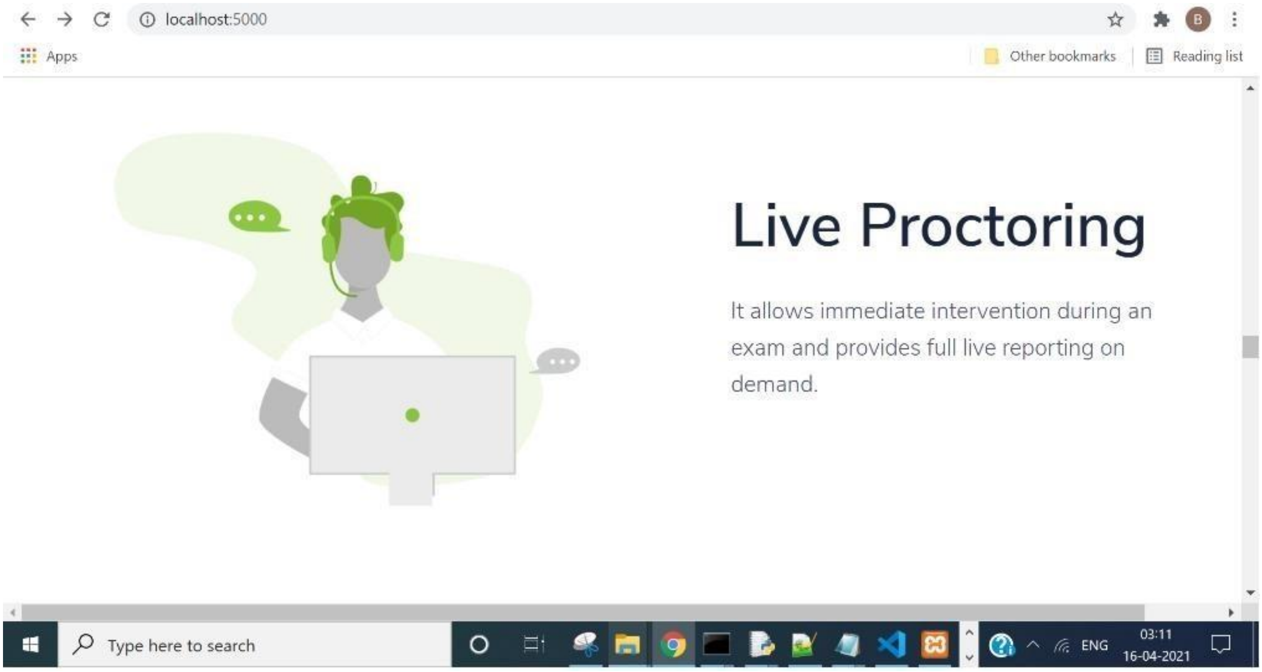
Online proctoring to advance your learning and testing program. Validate knowledge. Reduce costs. Expand access.



Exam Proctoring

An all-inclusive remote monitoring solution, including Image verification and computer restrictions.

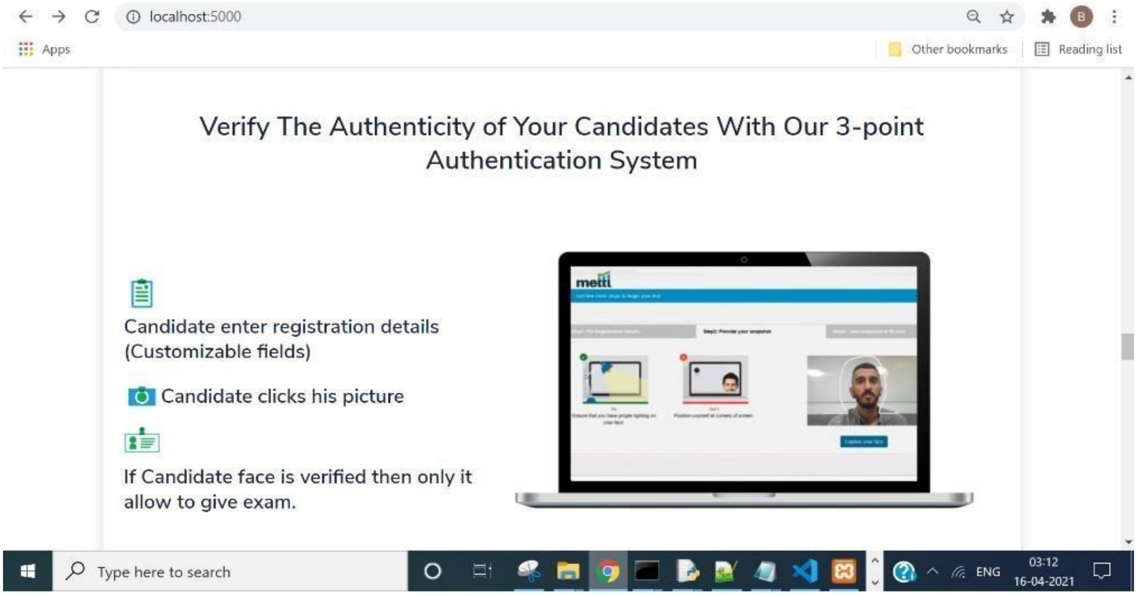
- Automated Proctoring
- Live Proctoring
- Professional Review



Live Proctoring

It allows immediate intervention during an exam and provides full live reporting on demand.

Figure 5.1: Home Page 1



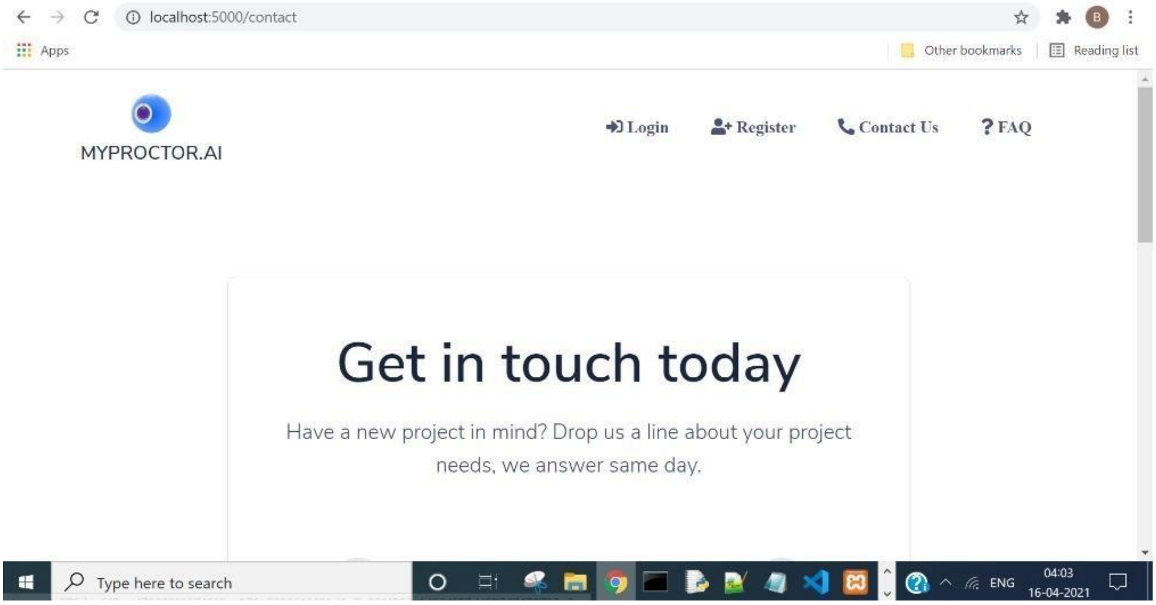
Verify The Authenticity of Your Candidates With Our 3-point Authentication System

Candidate enter registration details (Customizable fields)

Candidate clicks his picture

If Candidate face is verified then only it allow to give exam.

Figure 5.2: Home Page 2



MYPROCTOR.AI

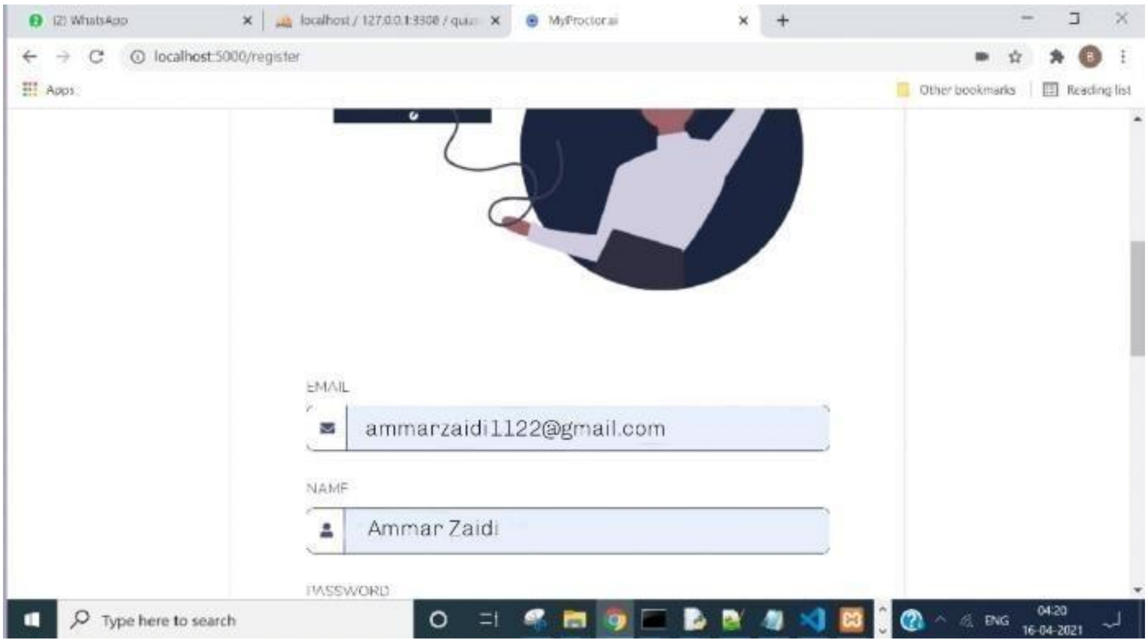
Login Register Contact Us ? FAQ

Get in touch today

Have a new project in mind? Drop us a line about your project needs, we answer same day.

Figure 5.3: Home Page 3

### Login and sign up



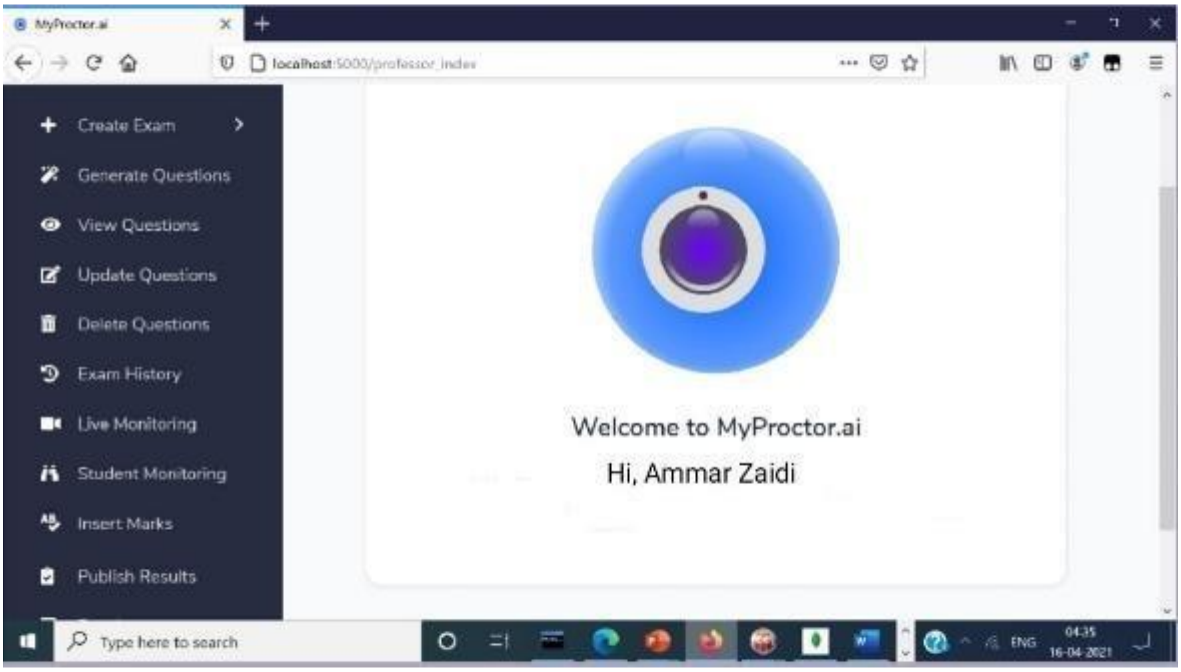
EMAIL

ammarzaidi1122@gmail.com

NAME

Ammar Zaidi

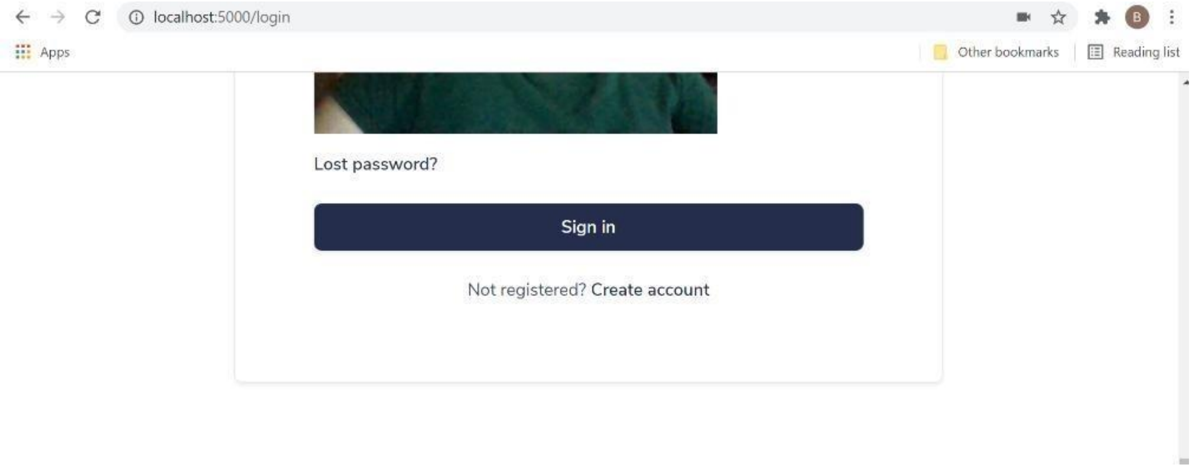
PASSWORD

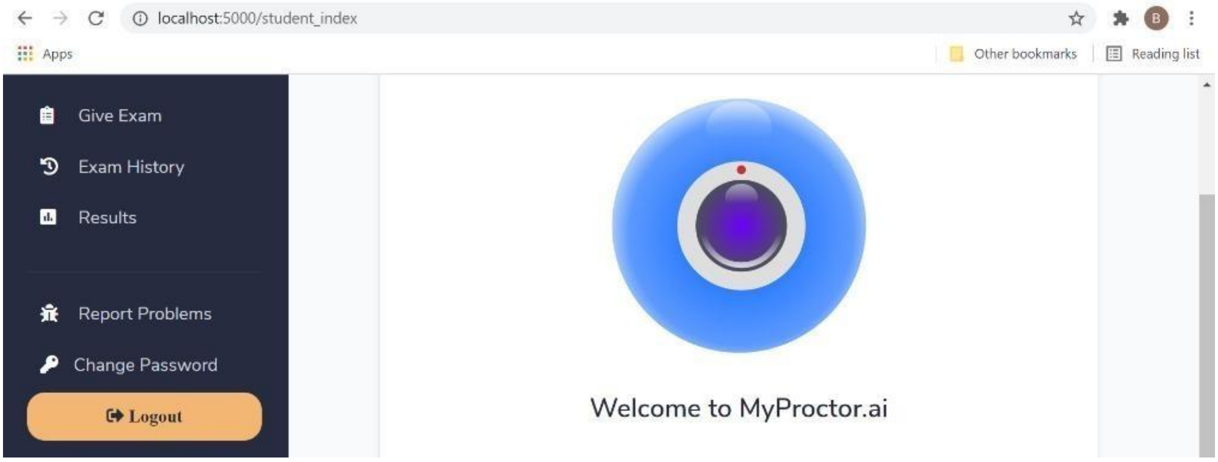


Welcome to MyProctor.ai

Hi, Ammar Zaidi

Figure 5.4: Login

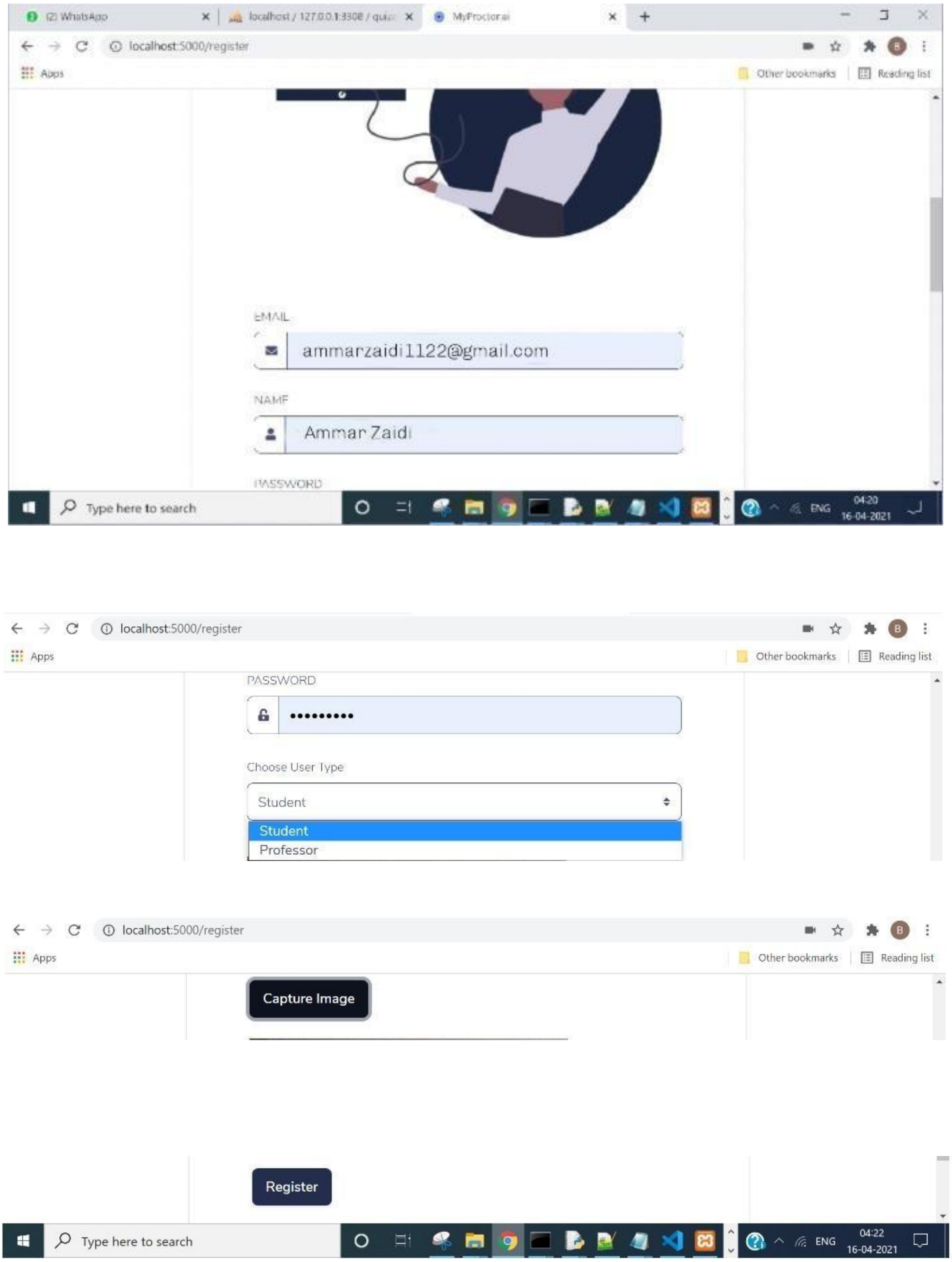




Welcome to MyProctor.ai

Figure 5.5: Sign Up

### Student Profile



EMAIL

ammarzaidi1122@gmail.com

NAME

Ammar Zaidi

PASSWORD

•••••••••

Choose User Type

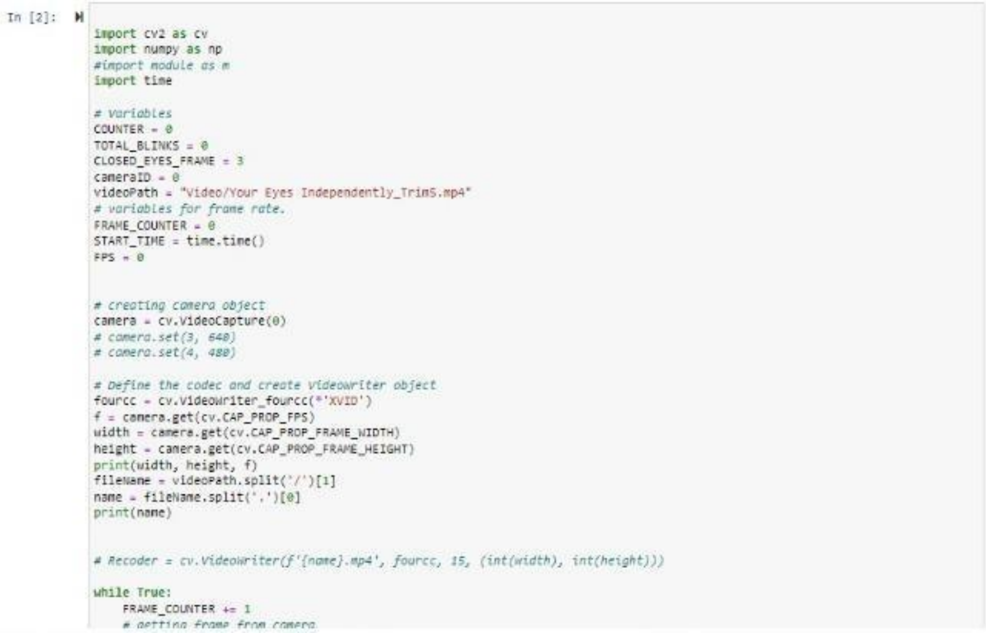
Student

Figure 5.6: Student Profile

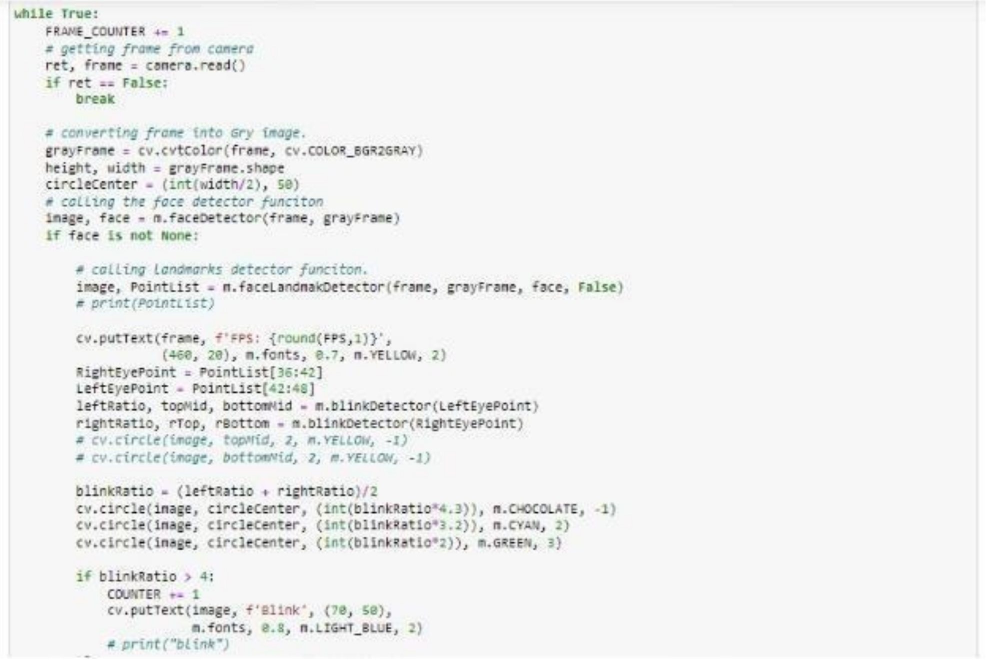
## 5.2 Cheating Detection Snapshot

The screenshots of detection that will run on background are as follows:

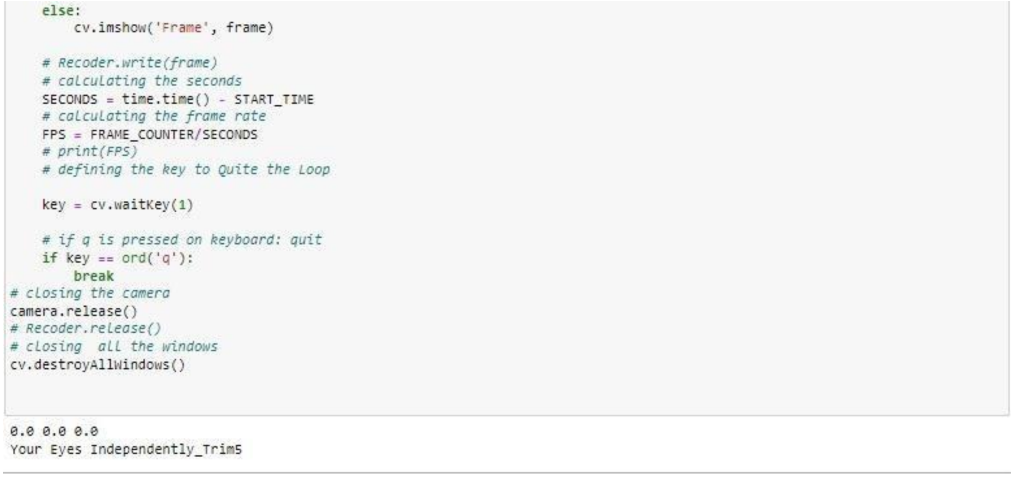
### 5.2.1 Number of Person and Eye detection



```
In [2]:
import cv2 as cv
import numpy as np
#import module as m
import time

# variables
COUNTER = 0
TOTAL_BLINKS = 0
CLOSED_EYES_FRAME = 3
cameraID = 0
videoPath = "Video/Your Eyes Independently_Trim5.mp4"
# variables for frame rate.
FRAME_COUNTER = 0
START_TIME = time.time()
FPS = 0


# creating camera object
camera = cv.VideoCapture(0)
# camera.set(3, 640)
# camera.set(4, 480)

# Define the codec and create videowriter object
fourcc = cv.VideoWriter_fourcc(*'XVID')
f = camera.get(cv.CAP_PROP_FPS)
width = camera.get(cv.CAP_PROP_FRAME_WIDTH)
height = camera.get(cv.CAP_PROP_FRAME_HEIGHT)
print(width, height, f)
fileName = videoPath.split('/')[1]
name = fileName.split('.')[0]
print(name)


# Recoder = cv.VideoWriter(f'{name}.mp4', fourcc, 15, (int(width), int(height)))

while True:
    FRAME_COUNTER += 1
    # getting frame from camera
```



```
while True:
    FRAME_COUNTER += 1
    # getting frame from camera
    ret, frame = camera.read()
    if ret == False:
        break

    # converting frame into Gry image.
    grayFrame = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
    height, width = grayFrame.shape
    circleCenter = (int(width/2), 50)
    # calling the face detector funciton
    image, face = m.faceDetector(frame, grayFrame)
    if face is not None:

        # calling landmarks detector funciton.
        image, PointList = m.faceLandmakDetector(frame, grayFrame, face, False)
        # print(PointList)

        cv.putText(frame, f'FPS: {round(FPS,1)}',
                   (460, 20), m.fonts, 0.7, m.YELLOW, 2)
        RightEyePoint = PointList[36:42]
        LeftEyePoint = PointList[42:48]
        leftRatio, topMid, bottomMid = m.blinkDetector(LeftEyePoint)
        rightRatio, rTop, rBottom = m.blinkDetector(RightEyePoint)
        # cv.circle(image, topMid, 2, m.YELLOW, -1)
        # cv.circle(image, bottomMid, 2, m.YELLOW, -1)

        blinkRatio = (leftRatio + rightRatio)/2
        cv.circle(image, circleCenter, (int(blinkRatio*4.3)), m.CHOCOLATE, -1)
        cv.circle(image, circleCenter, (int(blinkRatio*3.2)), m.CYAN, 2)
        cv.circle(image, circleCenter, (int(blinkRatio*2)), m.GREEN, 3)

        if blinkRatio > 4:
            COUNTER += 1
            cv.putText(image, f'Blink', (70, 50),
                       m.fonts, 0.8, m.LIGHT_BLUE, 2)
            # print("blink")
```

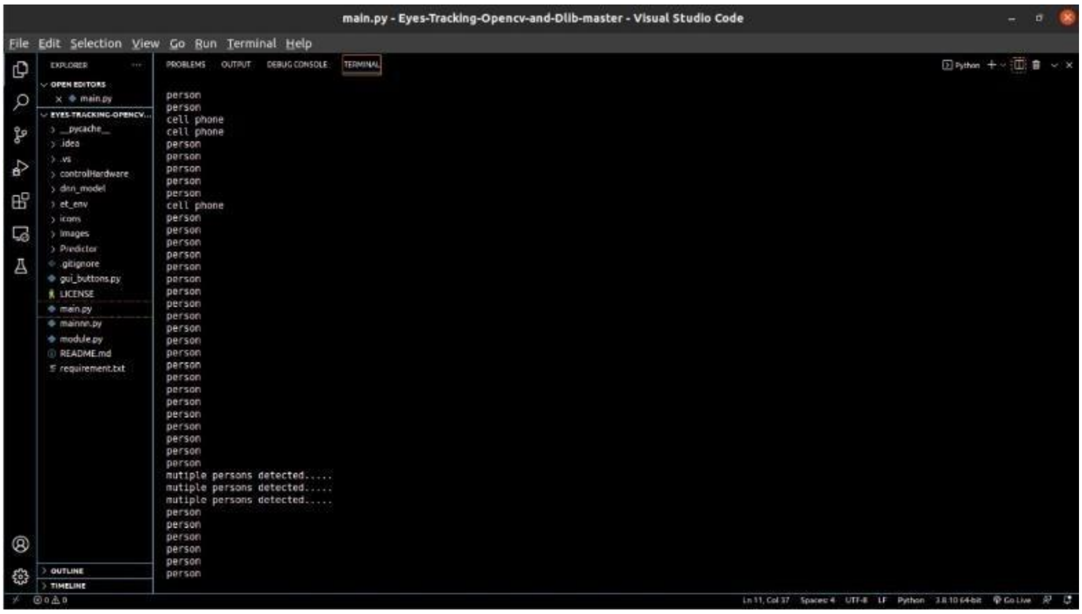


```
    else:
        cv.imshow('Frame', frame)

    # Recoder.write(frame)
    # calculating the seconds
    SECONDS = time.time() - START_TIME
    # calculating the frame rate
    FPS = FRAME_COUNTER/SECONDS
    # print(FPS)
    # defining the key to Quite the Loop

    key = cv.waitKey(1)

    # if q is pressed on keyboard: quit
    if key == ord('q'):
        break
# closing the camera
camera.release()
# Recoder.release()
# closing  all the windows
cv.destroyAllWindows()
```

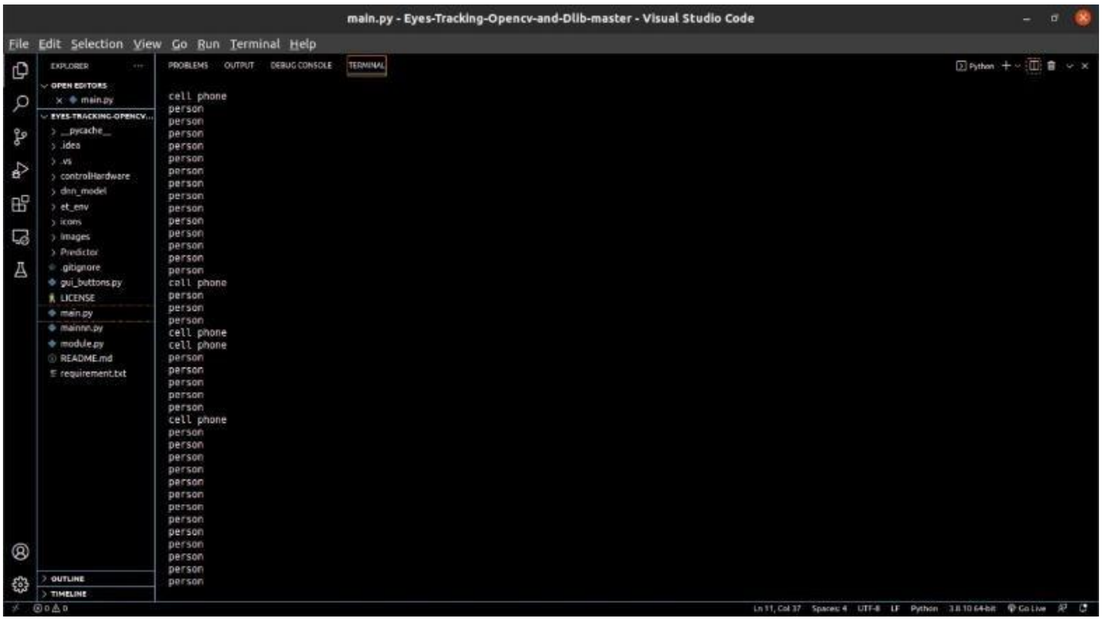
```
0.0 0.0 0.0
Your Eyes Independently_Trim5
```

Figure 5.7: Number of Person and Eye detection

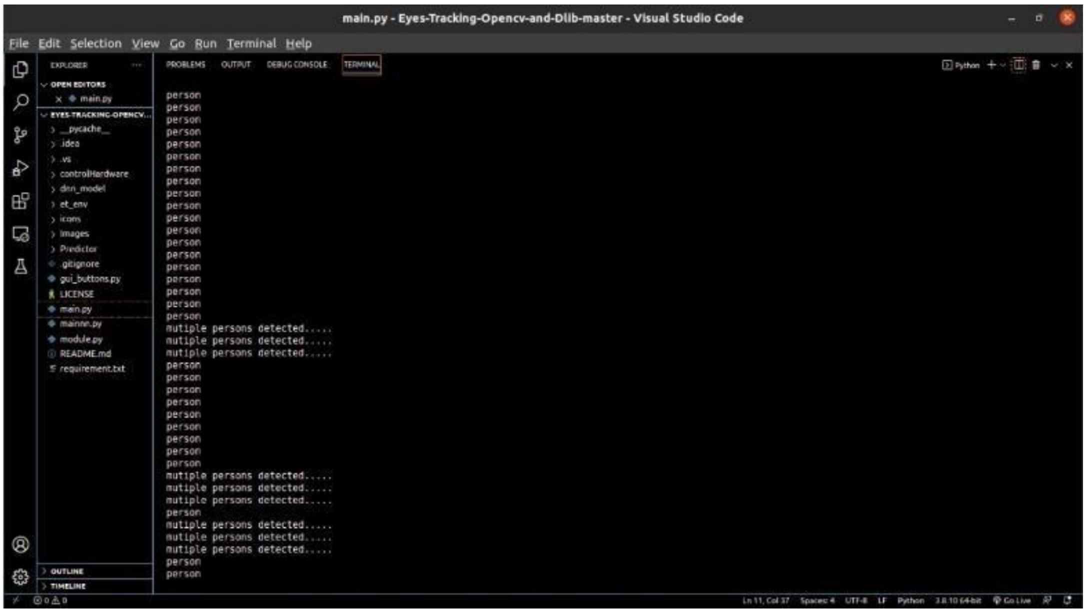
**OUTPUTS:**



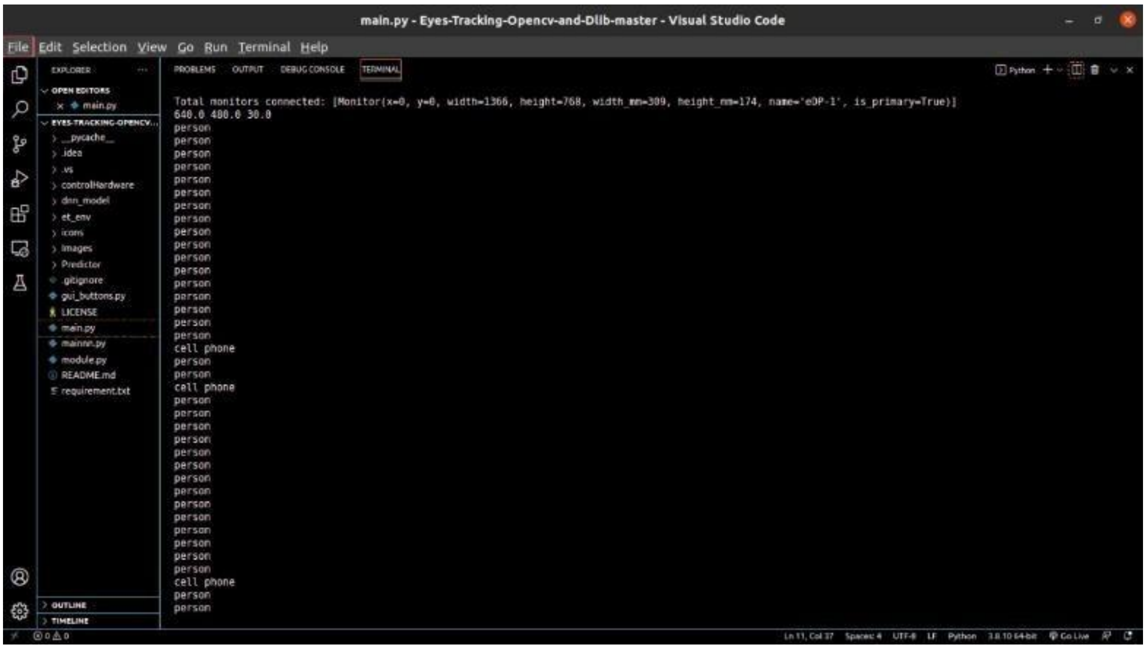
```
person
person
cell phone
cell phone
person
person
person
person
person
cell phone
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
person
person
person
person
```



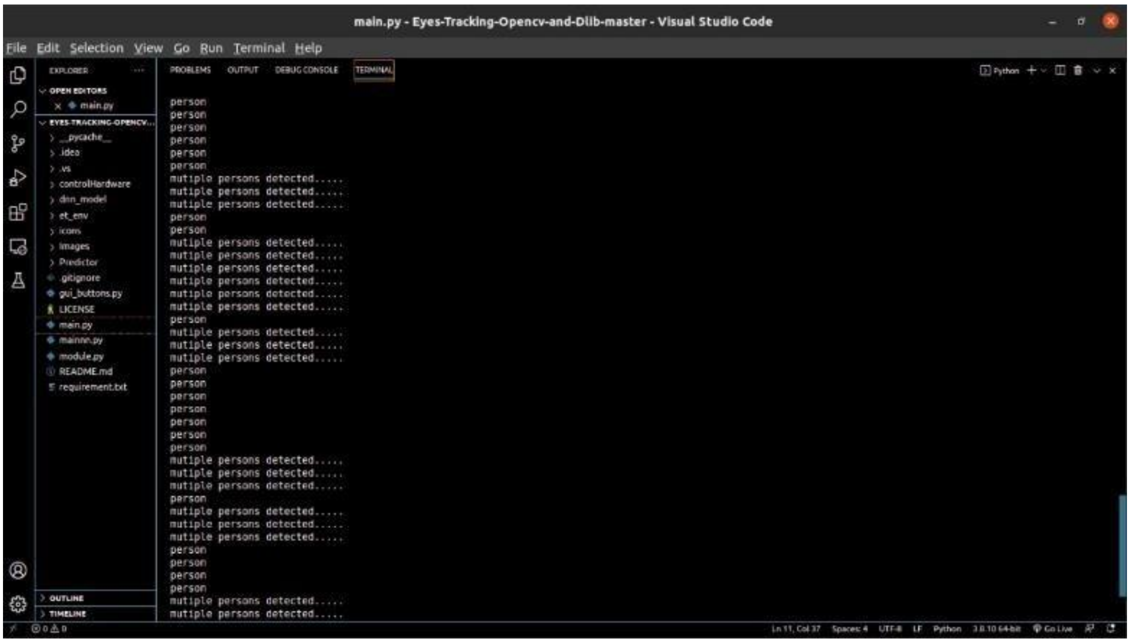
```
cell phone
person
person
person
person
person
person
person
person
person
person
person
person
person
person
cell phone
person
person
person
cell phone
cell phone
person
person
person
person
person
cell phone
person
person
person
person
person
person
person
person
person
person
person
person
person
```



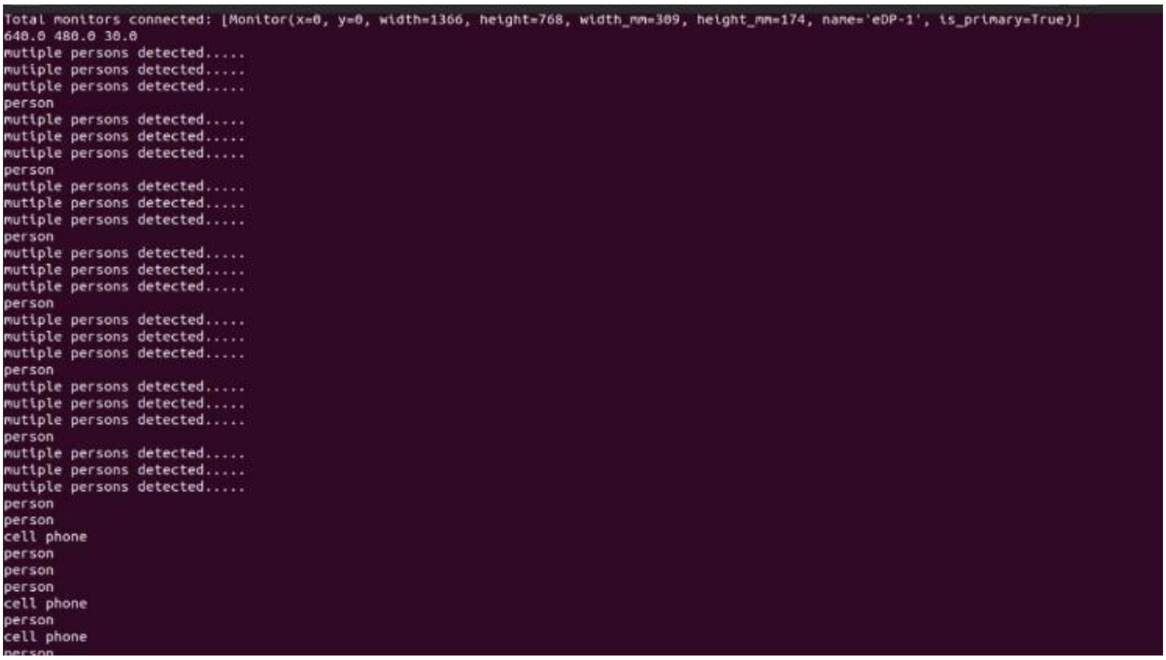
```
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
person
person
person
person
person
person
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
```



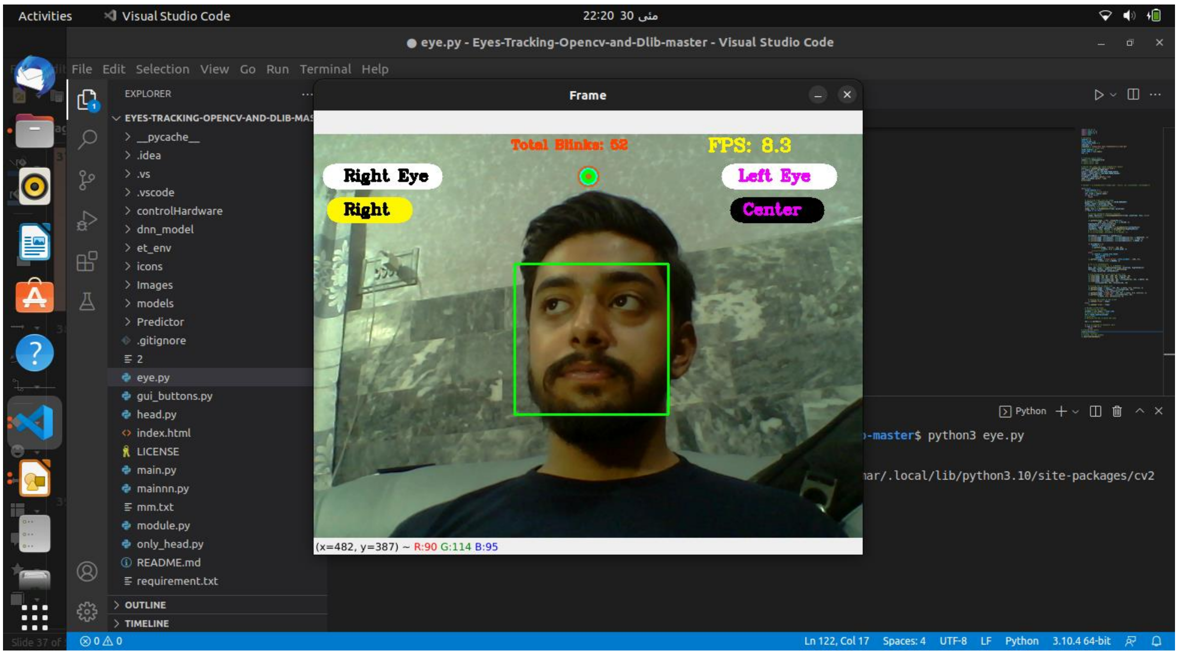
```
Total monitors connected: [Monitor(x=0, y=0, width=1366, height=768, width_mm=309, height_mm=174, name='eDP-1', is_primary=True)]
640.0 480.0 30.0
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
person
cell phone
person
person
cell phone
person
person
person
person
person
person
person
person
person
person
person
person
person
person
cell phone
person
person
```



```
person
person
person
person
person
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
person
person
person
person
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
person
person
mutiple persons detected.....
mutiple persons detected.....
```



```
Total monitors connected: [Monitor(x=0, y=0, width=1366, height=768, width_mm=309, height_mm=174, name='eDP-1', is_primary=True)]
640.0 480.0 30.0
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
mutiple persons detected.....
mutiple persons detected.....
mutiple persons detected.....
person
person
cell phone
person
person
person
cell phone
person
cell phone
person
```



```
-master$ python3 eye.py

nar/.local/lib/python3.10/site-packages/cv2
```

# CHAPTER 6

# CONCLUSION AND RECOMMENDATIONS

## 6.1 Conclusions

When compared to traditional proctoring methods, online proctoring solutions are significantly better, faster, cheaper, and more equitable. From scheduling exams, monitoring tests, grading assessments, and declaring results online, the process of administering an examination can be successfully automated with the correct proctoring solutions. AI-driven proctoring solutions also ensure accuracy by eliminating the possibility of human error, resulting in a seamless and cost-effective experience for both educators and test candidates.

## 6.2 Future Direction

We did not use cutting-edge technologies in our project. So, in the near future, we will try to apply cutting-edge technology to improve efficiency, accuracy, and add additional features to our project. We'll also create a mobile application for it, so it can be used on any platform.

## REFERENCES

[1] Tony Bates. What’s right and what’s wrong about coursera-style moocs. *EdTech in the Wild*, 2019.

[2] Nataliya Boyko, Oleg Basystiuk, and Nataliya Shakhovska. Performance eval uation and comparison of software for face recognition, based on dlib and opencv library. pages 478–482, 2018.

[3] Stefano Cesco, Vincenzo Zara, Alberto F De Toni, Paolo Lugli, Giovanni Betta, Alexander CO Evans, and Guido Orzes. Higher education in the first year of covid-19: Thoughts and perspectives for the future. *International Journal of Higher Education*, 10(3):285–294, 2021.

[4] Nuremberg Code. The nuremberg code. *Trials of war criminals before the Nuremberg military tribunals under control council law*, 10(2):181–182, 1949.

[5] Seife Dendir and R Stockton Maxwell. Cheating in online courses: Evidence from online proctoring. *Computers in Human Behavior Reports*, 2:100033, 2020.

[6] Anne Eisenberg. Keeping an eye on online test-takers. *New York Times*, 2, 2013.

[7] Jeff Forcier, Paul Bissex, and Wesley J Chun. Python web development with django. 2008.

[8] David Foster and Harry Layman. Online proctoring systems compared. *Online verf¨ugbar unter https://ivetriedthat. com/wp content/uploads/2014/07/Caveon-Test-Security. pdf*, 2013.

[9] Suraj Shahu Gaikwad and PRATIBHA Adkar. A review paper on bootstrap

framework. *IRE Journals*, 2(10):349–351, 2019.

[10] Fiseha M Guangul, Adeel H Suhail, Muhammad I Khalit, and Basim A Khid hir. Challenges of remote assessment in higher education in the context of covid-19: a case study of middle east college. *Educational assessment, evalu ation and accountability*, 32(4):519–535, 2020.

[11] Rachel Huang, Jonathan Pedoeem, and Cuixian Chen. Yolo-lite: a real-time object detection algorithm optimized for non-gpu computers. pages 2503–2510, 2018.

[12] Sangjun Jeon, Jewan Bang, Keunduck Byun, and Sangjin Lee. A recovery method of deleted record for sqlite database. *Personal and Ubiquitous Com puting*, 16(6):707–715, 2012.

[13] Davis E King. Dlib-ml: A machine learning toolkit. *The Journal of Machine Learning Research*, 10:1755–1758, 2009.

[14] Steve Kolowich. How edx plans to earn, and share, revenue from its free online courses. *The Chronicle of Higher Education*, 21, 2013.

[15] Rahul Kumar. Assessing higher education in the covid-19 era. *Brock Educa tion: A Journal of Educational Research and Practice*, 29(2):37–41, 2020.

[16] Robin Nixon. *Learning PHP, MySQL, JavaScript, and CSS: A step-by-step guide to creating dynamic websites*. ” O’Reilly Media, Inc.”, 2012.

[17] Chris Rose et al. Virtual proctoring in distance education: An open-source solution. *American Journal of Business Education (AJBE)*, 2(2):81–88, 2009.

[18] Klaus Schoenbach. Myths of media and audiences: Inaugural lecture as pro fessor of general communication science, university of amsterdam. *European Journal of Communication*, 16(3):361–376, 2001.

[19] Jerry Shum. Jerry blogger: Notepad++ 6.5 is released. 2013.

[20] Guido vanRossum. Python reference manual. *Department of Computer Sci ence [CS]*, (R 9525), 1995.