Student Attention Detection System

Submitted in partial fulfillment of the requirements
of the degree of

B. E. Computer Engineering

By

Hetik Chandaria	**10**	**182015**
Atharva Chiplunkar	**12**	**182017**
Tanmay Desai	**17**	**182025**
Vatsal Mehta	**53**	**182063**

Guide:

Mrs Varsha Nagpurkar
Assistant Professor

Department of Computer Engineering
St. Francis Institute of Technology
(Engineering College)

University of Mumbai
2021-2022

CERTIFICATE

This is to certify that the project entitled **"Student Attention Detection System"** is a bonafide work of **"Hetik Chandaria (10), Atharva Chiplunkar (12), Tanmay Desai (17), Vatsal Mehta (53)"**submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of B.E. in Computer Engineering

_ _ _ _ _ _ _ _ _ _ _ _ _ _

(Varsha Nagpurkar)

Guide

_ _ _ _ _ _ _ _ _ _ _ _ _ _

(Dr. Kavita Sonawane)

Head of Department

_ _ _ _ _ _ _ _ _ _ _ _ _ _

(Dr. Sincy George)

Principal

Project Report Approval for B.E.

This project report entitled ***Student Attention Detection System*** by ***Hetik Chandaria, Atharva Chiplunkar, Tanmay Desai and Vatsal Mehta*** is approved for the degree of ***B.E. in Computer Engineering.***

Examiners

1.) _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

2.) _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Date:

Place:

Declaration

We declare that this written submission represents my ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We

understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_ _ _ _ _ _ _ _ _ _ _ _ _

(Hetik Chandaria 10)

_ _ _ _ _ _ _ _ _ _ _ _ _ _

(Atharva Chiplunkar 12)

_ _ _ _ _ _ _ _ _ _ _ _ _ _

(Tanmay Desai 17)

_ _ _ _ _ _ _ _ _ _ _ _ _ _

(Vatsal Mehta 53)

Date:

Abstract

The COVID-19 pandemic, also known as the Coronavirus pandemic, is an ongoing global crisis that caused significant alterations to academia, demanding new regulations and creating unprecedented challenges for both learners and tutors . In order to minimize the transmission of the contagious virus, students have to study from home.

COVID-19 pandemic caused an increasing demand for online academic classes, which led to the demand for effective online exams with regards to limitations on time and resources.

Consequently, holding online exams with sufficient reliability and effectiveness became one of the most critical and challenging subjects in higher education. Online lectures have had their fair share of problems.

With online lectures comes the problem where students tend to lose concentration and on many occasions it is noted that students end up feeling drowsy. Also as exams are being conducted online, students are taking advantage of this situation by not giving exams in a fair manner. Students tend to cheat in online exams as the fear of getting caught does not exist. Hence we propose a system where the drowsiness of a student during lectures is tracked and during exams the eye movement of students is tracked to make sure students are giving exams fairly.

The system uses facial recognition and then applies various algorithms and image processing techniques to track the above mentioned parameters. For facial recognition, HOG based feature extraction technique is used.

The student is then alerted and at the same time the teacher is also given a summary of the behavior of the student during the lecture or exam. Also we have provided the question paper and the provision for uploading the answer sheet on the same system so that the student does not have to change tabs during exams. The system also detects if the student has changed tabs.

Contents

List of Figures

List of Abbreviations

Sr No	Abbreviation	Expanded Form
1	DFD	Data Flow Diagram
2	HOG	Histogram Oriented Gradient
3	CV	Computer Vision
4	EAR	Eye Aspect Ratio
5	RGB	Red Green Blue
6	CPU	Central Processing Unit
7	GPU	Graphics Processing Unit
8	GUI	Graphical User Interface

Chapter 1

Introduction

The World Health Organization has declared Covid-19 as a pandemic that has posed a contemporary threat to humanity. This pandemic has successfully forced the global shutdown of several activities, including educational activities, and this has resulted in tremendous crisis-response migration of universities with online learning serving as the educational platform. With online learning, the problem of students not paying attention increases due to less interaction between the teacher and students and also online learning can get a bit monotonous as a result students can get drowsy. The objective of this project is to build a system that detects whether the student is fully awake while attending the online lectures and if not, it will alert the student.

1.1 Description

In this project, we are making a system that will check whether the student is paying attention in the lecture or not. In this, we will be using two ways to check whether the student is paying attention or not. Firstly we will check whether the student is feeling drowsy or not. Secondly we will see whether the student is looking at the screen or doing something else. In this we will calculate the eye aspect ratio and check the blinking rate. Also, we will check the eye movement of the student during exams to ensure that the student is not looking anywhere but the screen.

1.2 Problem Formulation

Drowsiness is one of the prime factors contributing to the lack of attention in online classes. This condition may result in terrible causes since it affects the Student's concentration. Therefore, detecting drowsiness is important as one of the steps to preventing a lack of attention in online classes. In this work, we will be developing a Machine Learning, Pattern Recognition and Computer Vision-based algorithm that will use all the relevant features for fast and accurate classification of drowsiness in students. Also, the chances of students cheating in exams are higher in exams. Therefore we will develop a system that also tracks their eye movements for any suspicious activities.

1.3 Motivation

Because of the unexpected circumstances that the globe is presently confronting with Covid19, education is also profoundly impacted by this epidemic. As a result, all universities and colleges are required to deliver lectures online. Because students are unfamiliar with this technique, this unanticipated change in the learning process has numerous downsides. As a

result, many students do not pay attention in class. A tracking system is therefore required to assist teachers in keeping track of students, whether they are paying attention or not.

1.4 Proposed Solution

For this approach, we implement drowsiness detection using OpenCV and Python. The Dlib library is used to detect and localize facial landmarks using Dlib's pre-trained facial landmark detector. It consists of two shape predictor models trained on the i-Bug 300-W dataset, that each localize 68 and 5 landmark points respectively within a face image. In this approach, 68 facial landmarks have been used. To estimate where the student is looking we find the gaze direction. You can determine the gaze direction indirectly by determining the location of the center of the pupil relatively to the location of the eye.

1.5 Scope of the Project

Our goal here is to build a student attention detection system that will detect the drowsiness of students and also check whether a student is giving the exam fairly or doing any malpractices. In this project, we are planning to build a system:-

- that will detect the drowsiness of students in online lectures.
- that will detect the eye movement of students in the online examinations.
- that will send a mail to the teacher of whether the student gave the exam properly or not along with the answer sheet of the student attached to the mail.
- that will also alert the student if he/she attempts to change tabs.

Chapter 2

Review of Literature

Paper 1

High eye blinking rate indicates the drowsiness level of the student. Generally, for adults, there is an interval of 2-10 seconds between each eye blink. After the eye area is detected the camera will capture the eye of the driver it will count the eye blink. The eye-detection algorithm only detects the eyes if they were opened, this will help in detecting the driver's drowsiness. Using this information, it is possible to assess if the driver has his/her eyes closed and count the number of times the user blinks. One among two eyes is enough to detect the eye blink rate. The blink rate which is lasting 3 or 4 seconds was assumed to be a fatigue condition. In general, the blink rate for a normal person is 2-10 seconds and around 10 blinks per minute are considered a normal eye condition. When the blink rate reduces below 10 blinks per minute, is considered an abnormal eye condition. Based on these factors, when the eye is closed for too long the drowsiness is detected.

$$EAR = ||(p2 - p6) - ||(p3 - p5)/(2||p1 - p4||)$$

The Eye Aspect Ratio (EAR) between height and width of the eye is computed where p1 to p6 are the 2D landmark locations. The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye.

Paper 2

A. Overview Design

The overall design idea of Student Attention detection is to capture an image from the camera and approximately estimate the state of drivers with data processing to realize these requirements; required hardware and software materials have to be collected. For this project Python machine learning and Arduino are used along with camera and load cell sensors. For facial and eye detection OpenCV and HOG algorithms are used and OPENCV libraries are used.

B. Face Detection

Student Attention Detection requires hardware and software components which include a camera to detect face and eyes and process eye blinking rate. In this project, several methods are applied which are explained in this paper.

C. Histogram Oriented Gradient

HOG algorithm is used to preprocess the image which includes image resize and color normalization in this project HOG is used to detect efficient features for eye detection and extract HOG features from the image patterns and gives the exact region of eyes from the captured image of the student.

D. Eyes Detection

After capturing the student's image and preprocessing the next process to estimate the drowsiness of a driver based on the blinking eye blinking rate. Values are calculated for every frame and changes in the blink rate are verified with the threshold value. For effectively detecting eye blinking rate HOG is used which is useful for face detection and provides an accurate eye detection rate.

Paper3:

In this paper, they implement drowsiness detection using OpenCV and Python. The Dlib

library is used to detect and localize facial landmarks using Dlib's pre-trained facial landmark detector. It consists of two shape predictor models trained on the i-Bug 300-W dataset, that each localizes 68 and 5 landmark points respectively within a face image. In this approach, 68 facial landmarks have been used.

Histogram of Oriented Gradients (HOG) based face detector is used in Dlib. In this method, frequencies of the gradient direction of an image in localized regions are used to form histograms. In many cases, it is more accurate than Haar cascades as the false positive ratio is small. Also, tuning at test time requires fewer parameters. It is especially suitable for face detection as firstly, it can describe contour and edge features exceptionally in various objects. Secondly, it performs operations on regional cells which allows the motion of the subject to be overlooked. Moreover, Dalal and Triggs discovered that the HOG descriptor works well for human detection in images, which makes it appropriate for drowsiness detection. In our model, a HOG-based detector is first instantiated to find the location of the face in each individual frame of the input video stream.

The outline of the facial features made by the oriented gradients makes it easy to discern the location and even the state of facial features. Upon finding the location of the face, the facial landmarks predictor is called to map the points of interest (eyes and mouth) and extract their coordinates.

Paper 4:

The main objective is to determine the location on the computer screen where the user is looking at, in the following referred to as the target location. Basically, two kinds of information are needed, the first one is the location of the eyes relative to the computer screen and the second one is the gaze direction. The first task is to gain the relevant facial structures

and their locations from the image received by the camera. One structure needed is the location of the eye within the image.

The gaze direction cannot be gained directly. But it can be derived indirectly from the location of the pupil. As the pupil is the part of the eye, which absorbs the light, it is directed to the location the person is looking at. Therefore, you can determine the gaze direction indirectly by determining the location of the center of the pupil relative to the location of the eye. In conclusion, the information that has to be derived from the image is the locations of the eyes and pupils.

As the pupil is built to absorb light, its brightness is usually the lowest compared to the surrounding area of the face, which makes the brightness a possible property to filter by. Through experiments, it was discovered that the red channel of an RGB image led to the best selectivity for the pupil against the surrounding area.

Therefore, in the next step, a grayscale image from the extracted section is created by only using the red channel. As the pupil should be the darkest area in the image, a binary filter is used, which turns every pixel in the image above the given threshold to white, and every pixel below the threshold to black. Theoretically, only the pixels belonging to the pupil are the ones that turn black, if the threshold is set correctly.

Chapter 3
System Analysis

3.1 Functional Requirements

- Users should be able to enable their videos using webcams.

3.2 Non-Functional Requirements:

- The processing of each request should be done within 10 seconds.
- The computer running the software must have a powerful CPU and GPU so that the data can be processed faster and there won't be a lag in the entire process.
- The site should load in 5 seconds when the number of simultaneous users are > 10000
- The system should provide better accuracy and optimized results.
- Easy to use and user-friendly.

3.3 Specific Requirements:

Hardware requirements:

- System: Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz
- Hard Disk: 100GB
- RAM: 4 GB RAM or more
- Monitor: LCD
- Microphone
- Webcam

Software Requirements:

- Operating System: Windows
- Coding Language: Python

3.4 Use Case Diagram-

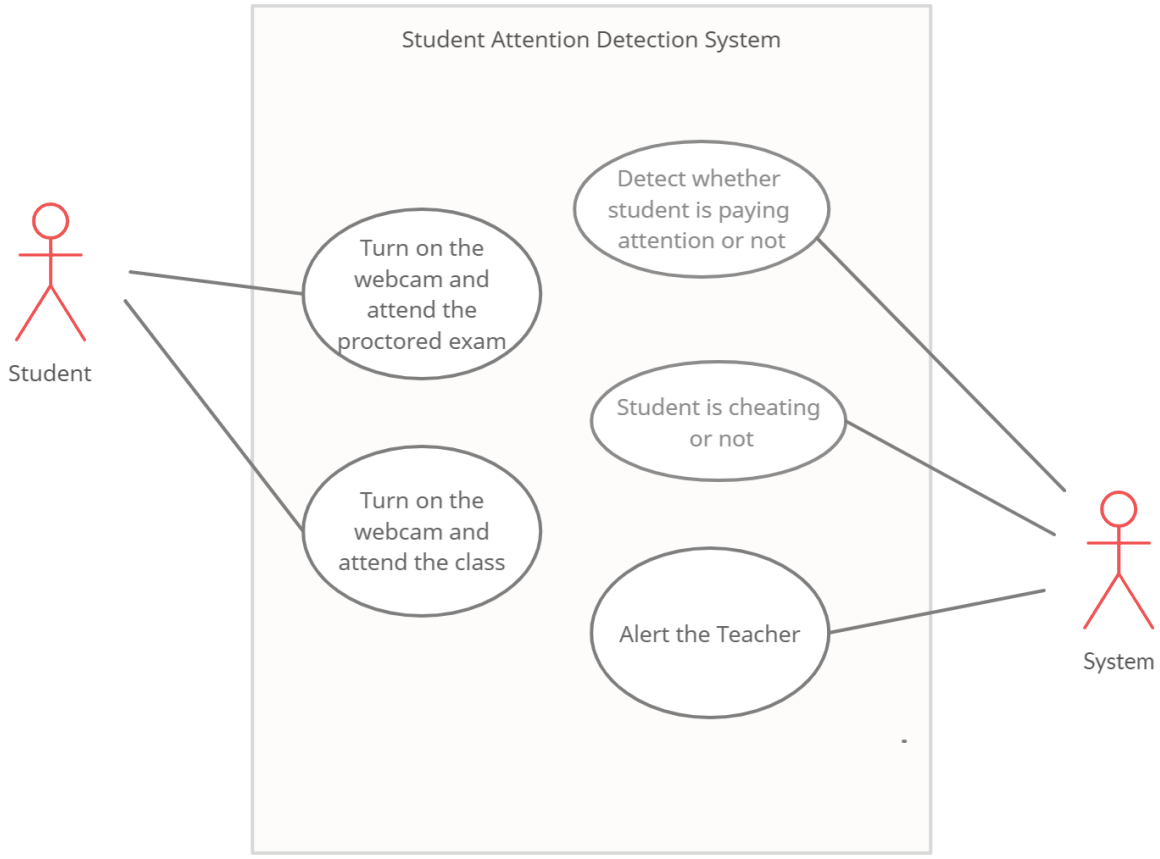

Fig 3.4.1. Use Case Diagram

The use case diagram consists of two actors. They are the Student and the System. The Students job is to turn on the webcam and attend the lecture. The job of the system is to detect whether the student is paying attention or not. If the student is not paying attention then the system will alert the teacher about the same.

Chapter 4
Analysis Modeling

4.1 Activity Diagram

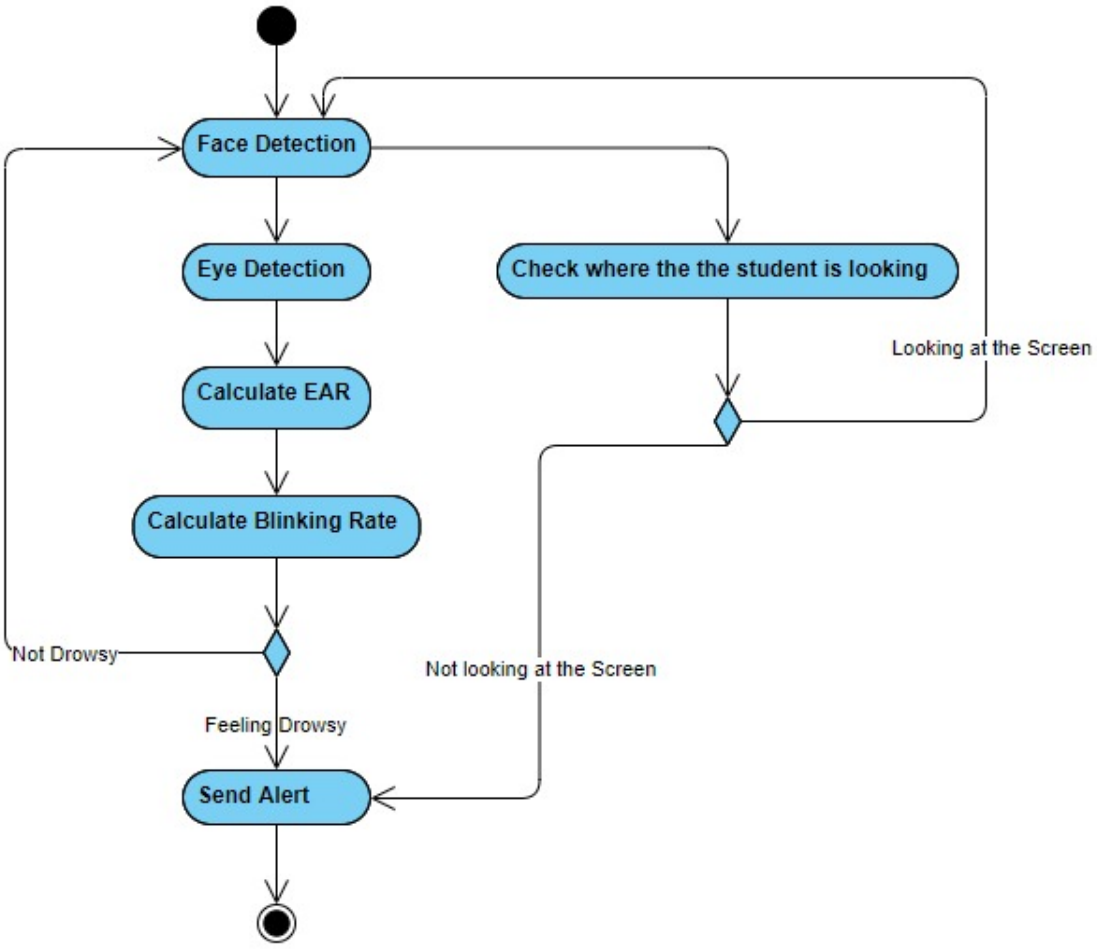

Fig 4.1.1. Activity Diagram

4.2 Functional Modeling

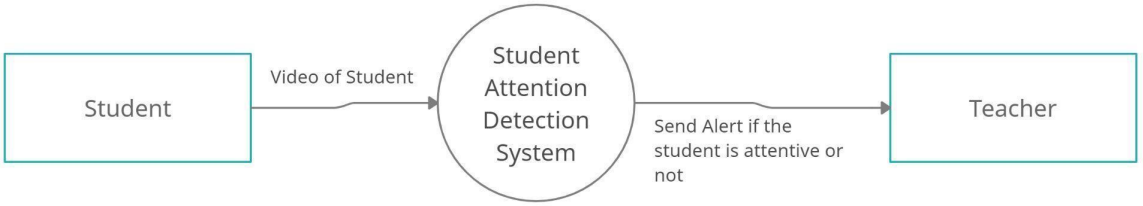

Fig 4.2.1. DFD Level 0

In this system the video of the student will be sent to the system. The system then will then check for drowsiness and where the student is looking and notify the teacher as well as the student.

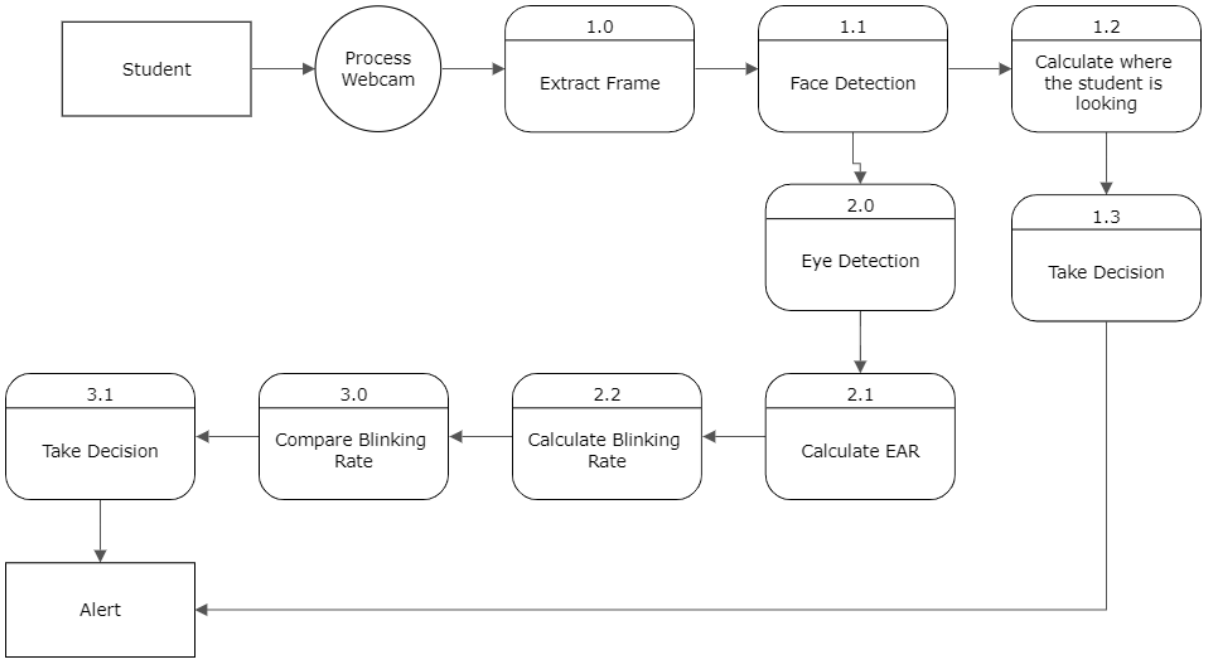

Fig 4.2.2. DFD Level 1

In This we first begin by identifying the face of the user from the video. We then extract the coordinates of the eye from the face. After getting the coordinates of the eye we then calculate the Eye Aspect Ratio. We then calculate the blinking rate of the eyes. If the blinking rate of the eyes is less than the threshold value it will alert the user. If the blinking rate is above the threshold then we continue with monitoring the students. We also check if the student is

looking at the screen or not. If the student is not looking at the screen then we will send an alert.

Chapter 5

Design

5.1 Architectural Design

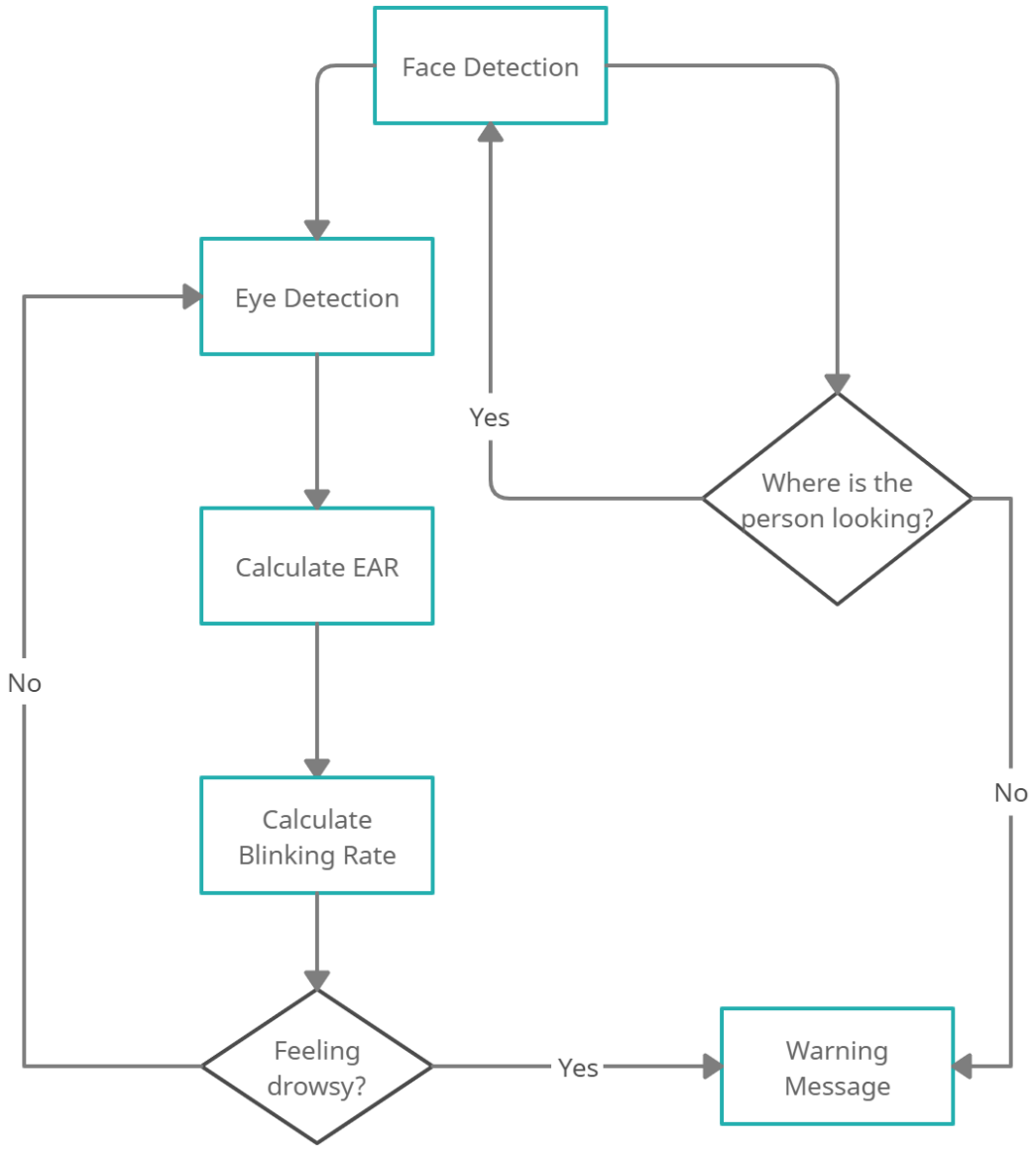

Fig 5.1.1. Flow Diagram

1. In this, we first begin by identifying the face of the user from the video.
2. We then extract the coordinates of the eye from the face.
3. After getting the coordinates of the eye we then calculate the Eye Aspect Ratio.
4. We then calculate the blinking rate of the eyes.
5. If the blinking rate of the eyes is less than the threshold value it will alert the user.
6. If the blinking rate is above the threshold then we continue with monitoring the students.
7. We also check if the student is looking at the screen or not.
8. If the student is not looking at the screen then we will send an alert.

5.2 Graphical User Interface

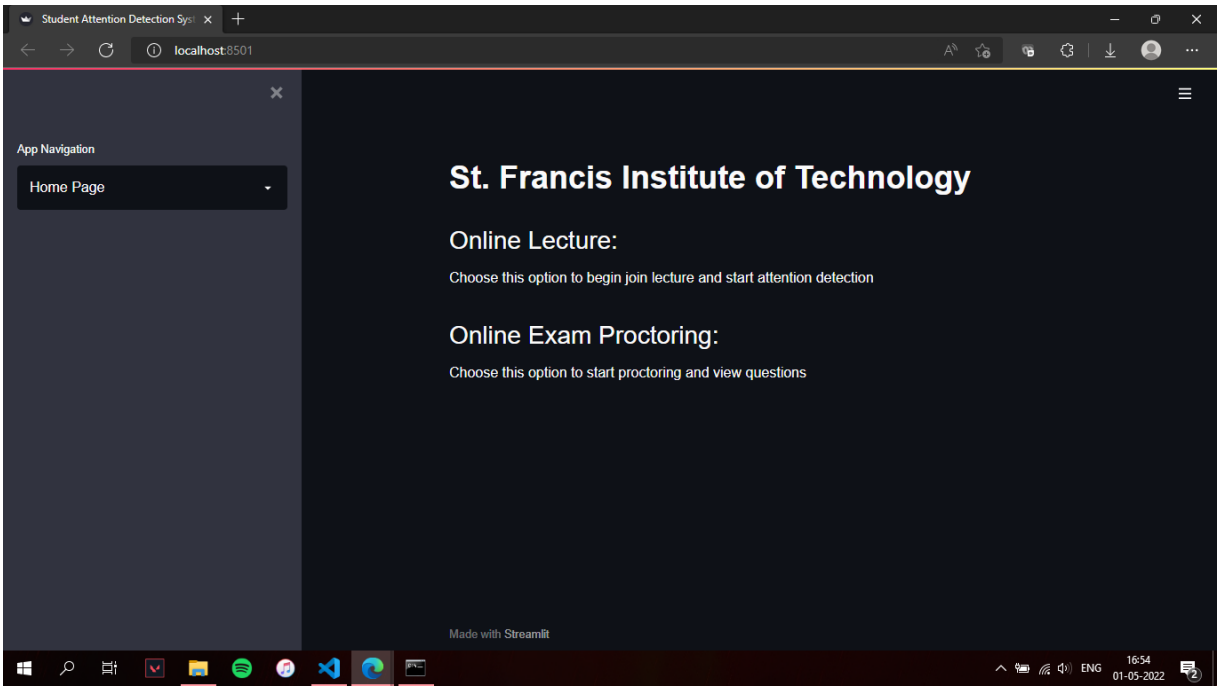

Figure.5.2.1. Home Page

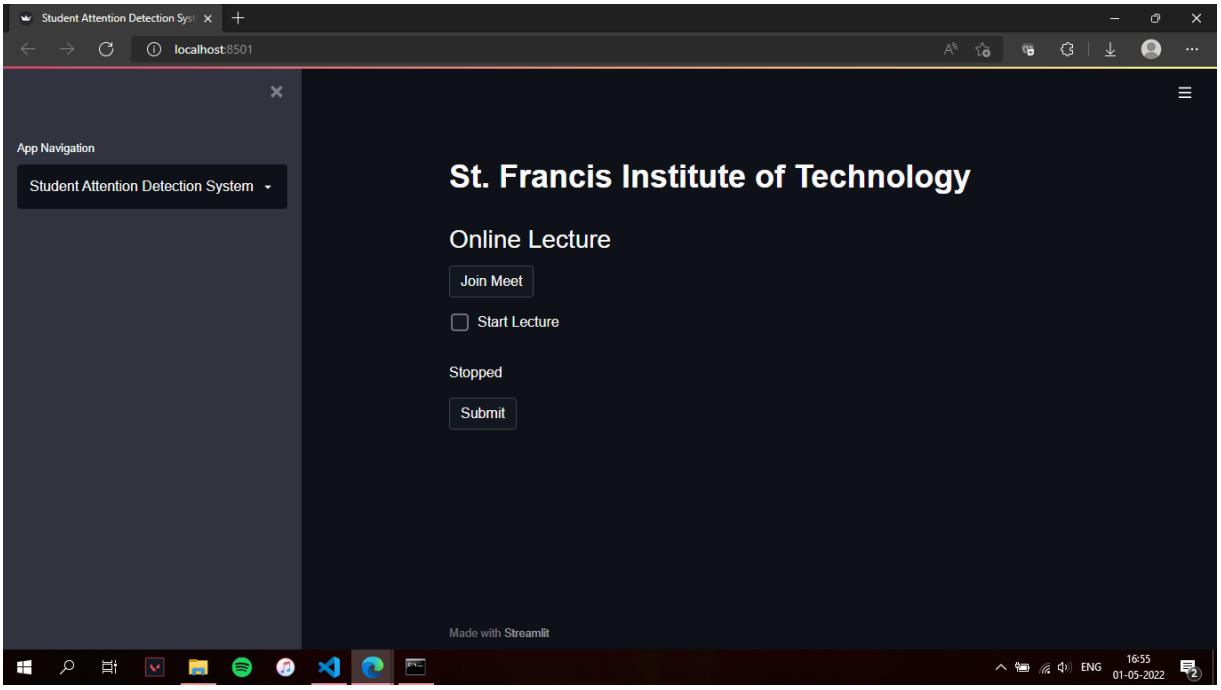

Figure.5.2.2. Online Lecture

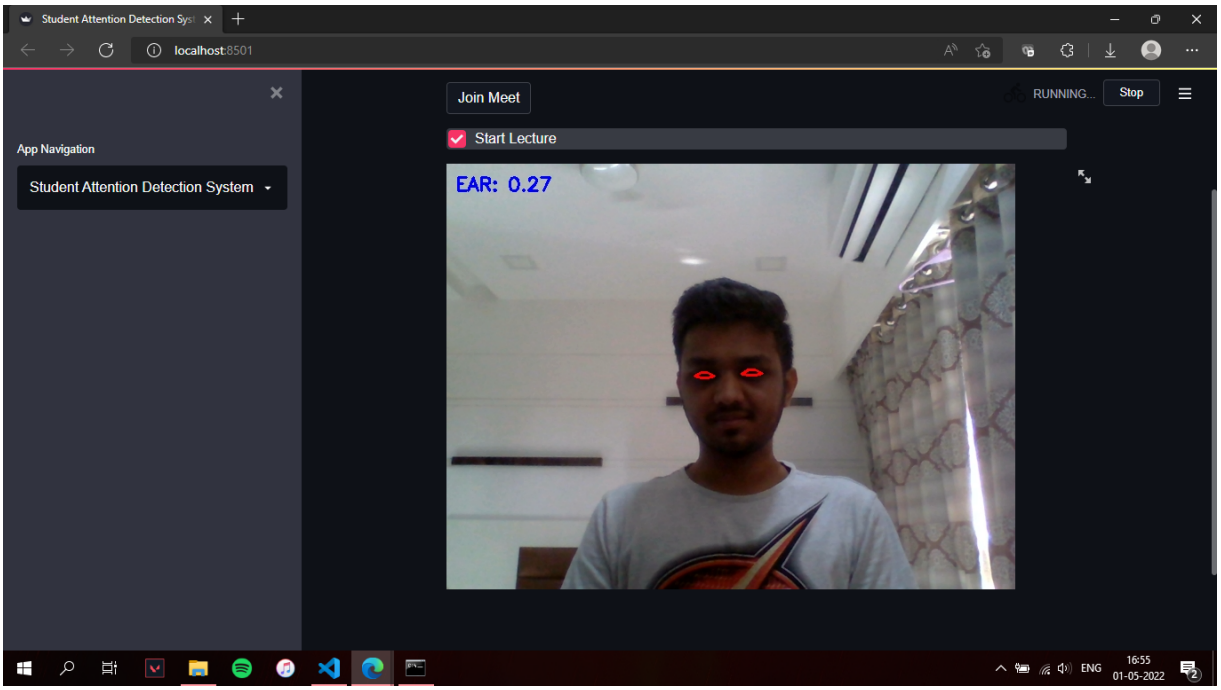

Figure.5.2.3. Drowsiness Detection in lecture

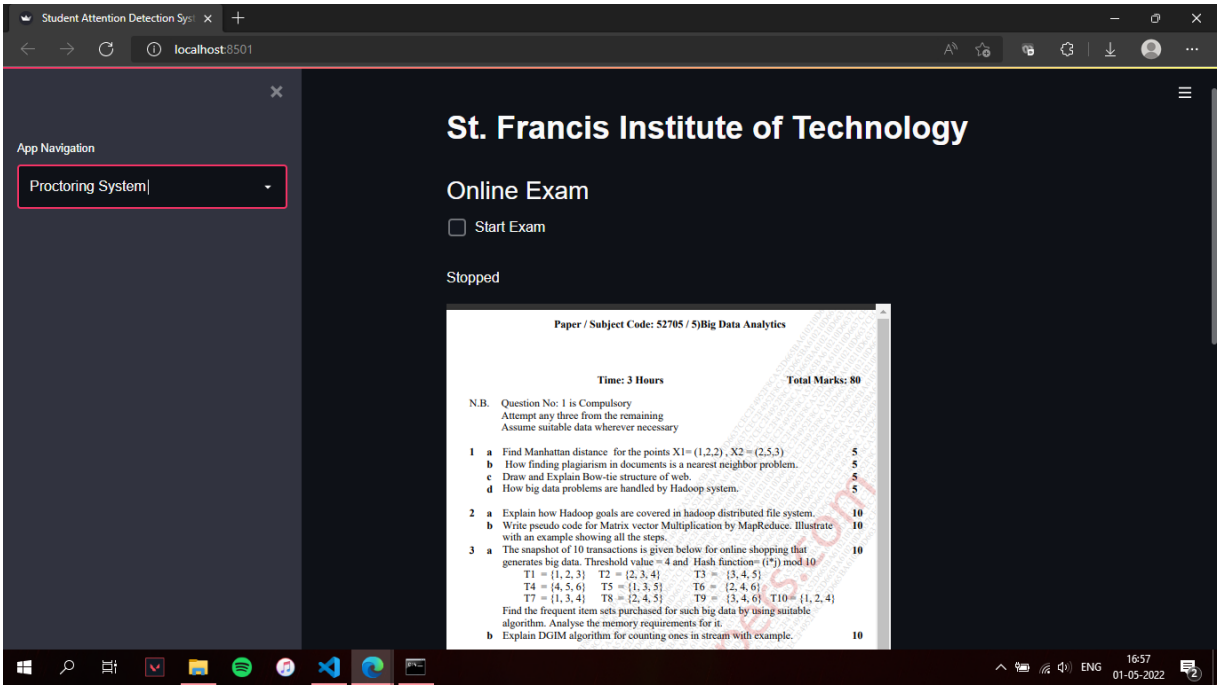

Figure.5.2.4. Online Exam

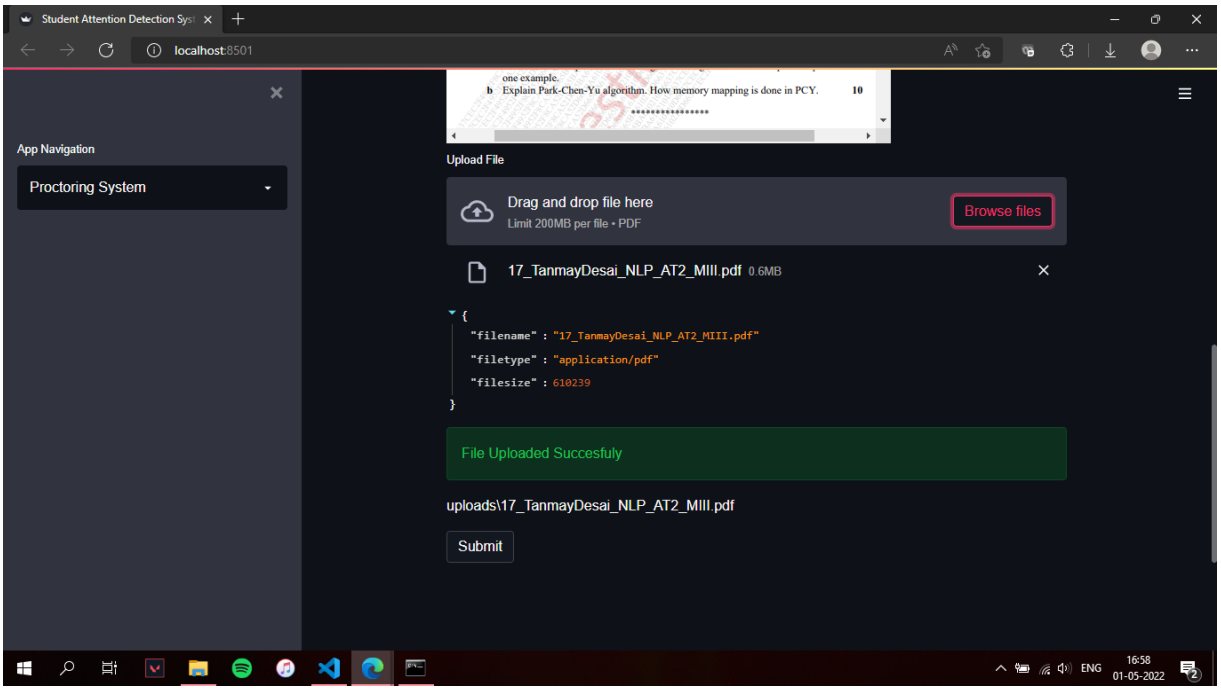

Figure.5.2.5. File Upload

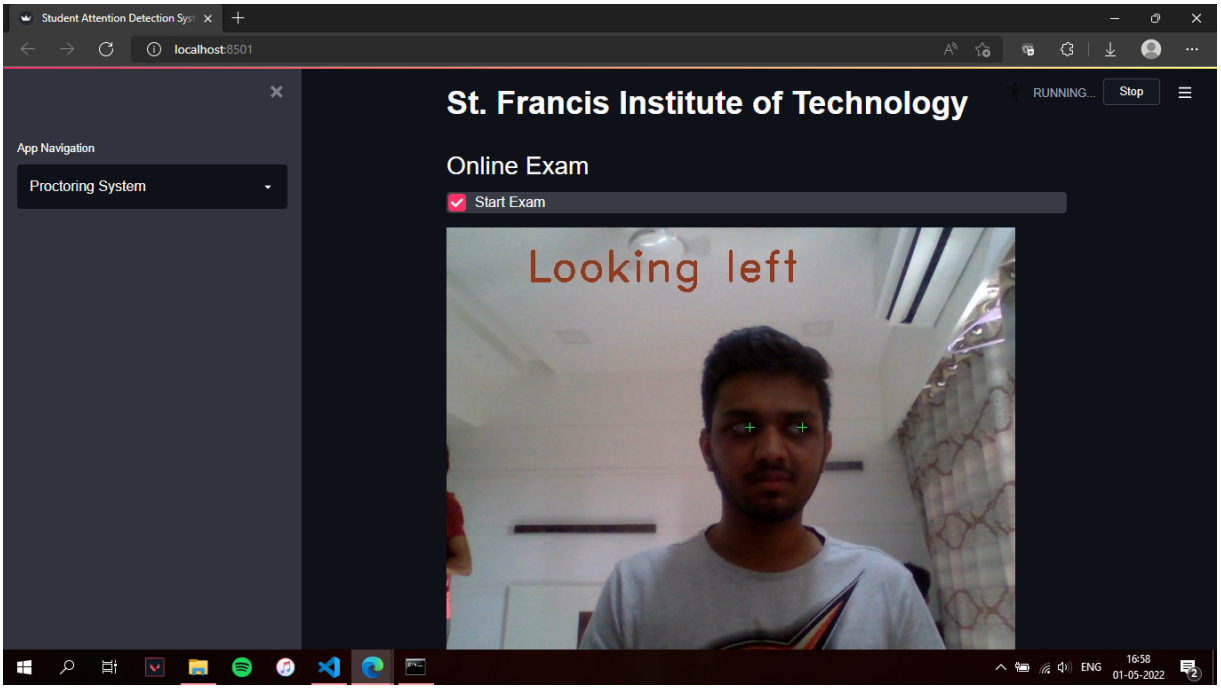

Figure.5.2.6. Proctoring System

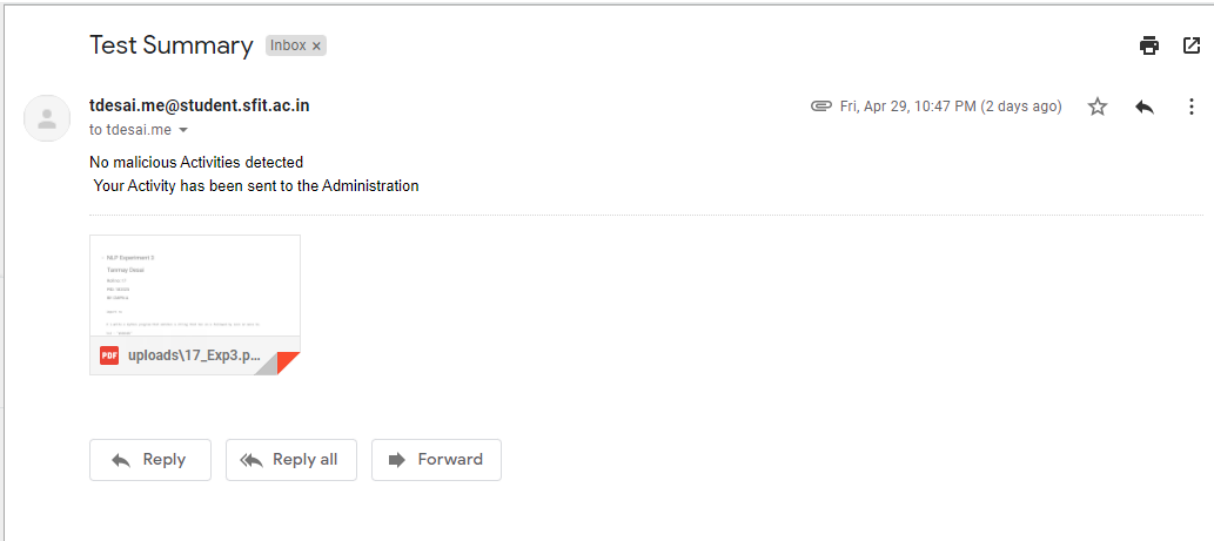

Figure.5.2.7. Email Notification if Malicious Activities are detected

Chapter 6

Implementation

6.1 Algorithms/ Methods Used:

1. **To check if the student is awake and paying attention or not:**

For this approach, we implement drowsiness detection using OpenCV and Python. The Dlib library is used to detect and localize facial landmarks using Dlib's pre-trained facial landmark detection. It consists of two shape predictor models trained on the i-Bug 300-W dataset, which each localizes 68 and 5 landmark points respectively within a face image. In this approach, 68 facial landmarks have been used.

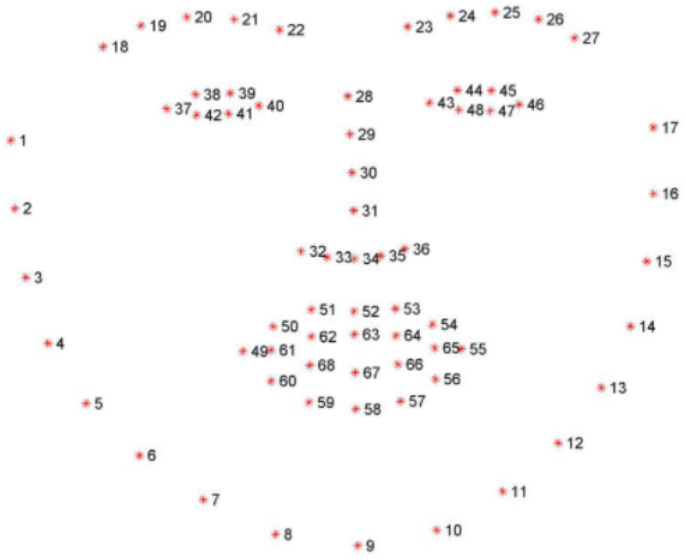

Fig 6.1.1: Manner in which 68 facial landmarks are mapped on a detected face.

Formula for EAR is :

$$EAR= ||(p2 - p6) -||(p3 - p5)/(2||p1 - p4||)$$

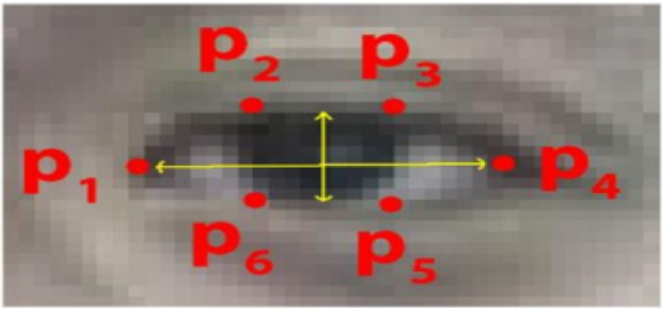

Fig 6.1.2: Eye Points

The Eye Aspect Ratio (EAR) is mostly constant when an eye is open and is getting close to zero while closing an eye. The blink rate which is lasting 3 or 4 seconds was assumed a

fatigue condition. When the blink rate reduces below 10 blinks per minute, is considered an abnormal eye condition. Based on these factors, when the eye is closed for too long the drowsiness is detected.

2. To Check where the student is looking:

In object detection, there's a simple rule: from big to small. Meaning you don't start with detecting eyes on a picture, you start with detecting faces. Then you proceed to eyes, pupils and so on. It saves a lot of computational power and makes the process much faster. So now we know how to read frames from a webcam, it's time to work on them to reach our goal. We create a new black mask using NumPy of the same dimensions as our webcam frame. Store the (x, y) coordinates of the points of the left and right eyes from the keypoint array shape and draw them on the mask using cv2.fillConvexPoly. It takes an image, points as a NumPy array with data type = np.int32 and color as arguments and returns an image with the area between those points filled with that color.

After doing this we have a black mask where the eye area is drawn in white. This white area is expanded a little using a morphological operation cv2.dilate. Using cv2.bitwise_and with our mask as the mask on our image, we can segment out the eyes. Convert all the (0, 0, 0) pixels to (255, 255, 255) so that only the eyeball is the only dark part left. Convert the result to grayscale to make the image ready for thresholding.

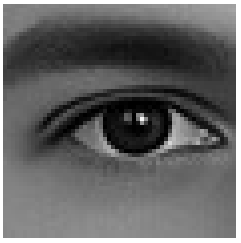

Fig 6.1.3. Grayscale Image of eyes

Thresholding is used to create a binary mask. So our task is to find an optimal threshold value against which we can segment out the eyeballs from the rest of the eye and then we need to find its center.

Fig 6.1.4. Thresholding

The eyeballs are segmented out and we can utilize cv2.findContours for finding them. Right now our background is white and our eyeballs are in black. However, in OpenCV's cv2.findContours() method, the object to find should be in white and the background is black. So we need to invert our threshold using cv2.bitwise_not. Now we can find contours. Theoretically, we can say that all we need to do is now find the two largest contours and those should be our eyeballs. However, this leaves out a little room for false positives that can be tackled by finding the midpoint between the eyes and dividing the image by that. Then we find the largest contours in those divisions and should be our eyeballs. The key points 40 and 43 (39 and 42 in Python because the index starts from zero) are used to find the midpoint. Find the largest contours on both sides of the midpoint by sorting it with cv2.contourArea. We can utilize cv2.moments to find the centers of the eyeballs.

Once the position of the eyeballs is located it is then compared with the center of the eyes and then a decision is made to tell if a person is looking to the left or right or above.

6.2 Working of the Project

```
status, frame = webcam.read()
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    grayImage = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = faceDetector(grayImage, 0)

    for face in faces:

      faceLandmarks = landmarkFinder(grayImage, face)
      faceLandmarks = face_utils.shape_to_np(faceLandmarks)

      leftEye = faceLandmarks[leftEyeStart:leftEyeEnd]
      rightEye = faceLandmarks[rightEyeStart:rightEyeEnd]
```

```
        leftEAR = eye_aspect_ratio(leftEye)
        rightEAR = eye_aspect_ratio(rightEye)

        ear = (leftEAR + rightEAR) / 2.0

        leftEyeHull = cv2.convexHull(leftEye)
        rightEyeHull = cv2.convexHull(rightEye)

        cv2.drawContours(frame, [leftEyeHull], -1, (255, 0, 0), 2)
        cv2.drawContours(frame, [rightEyeHull], -1, (255, 0, 0), 2)

        if ear < MINIMUM_EAR:
           EYE_CLOSED_COUNTER += 1
        else:
           EYE_CLOSED_COUNTER = 0
        cv2.putText(frame, "EAR: {:.2f}".format(ear), (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        if EYE_CLOSED_COUNTER >= MAXIMUM_FRAME_COUNT:
           cv2.putText(frame, "Drowsiness", (10, 50), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
(0, 0, 255), 2)

def eye_aspect_ratio(eye):
   p2_minus_p6 = dist.euclidean(eye[1], eye[5])
   p3_minus_p5 = dist.euclidean(eye[2], eye[4])
   p1_minus_p4 = dist.euclidean(eye[0], eye[3])
   ear = (p2_minus_p6 + p3_minus_p5) / (2.0 * p1_minus_p4)
   return ear

#EYE DIRECTION
eyes = cv2.bitwise_and(img, img, mask=mask)
      mask = (eyes == [0, 0, 0]).all(axis=2)
      eyes[mask] = [255, 255, 255]
      mid = int((faceLandmarks[42][0] + faceLandmarks[39][0]) // 2)
      eyes_gray = cv2.cvtColor(eyes, cv2.COLOR_BGR2GRAY)
      threshold = cv2.getTrackbarPos('threshold', 'image')
      _, thresh = cv2.threshold(eyes_gray, threshold, 255, cv2.THRESH_BINARY)
      thresh = process_thresh(thresh)

      eyeball_pos_left = contouring(thresh[:, 0:mid], mid, img, end_points_left)
      eyeball_pos_right = contouring(thresh[:, mid:], mid, img, end_points_right, True)
      result = get_eye_pos(img, eyeball_pos_left, eyeball_pos_right)
      if(EYE_COUNTER>= MAXIMUM_FRAME_COUNT):
                              cv2.putText(img,"Please Look at the Screen", (10, 50),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
```

```
            EYE_COUNTER = 0
        cv2.imshow("thresh",thresh)

def find_eyeball_position(end_points, cx, cy):
    """Find and return the eyeball positions, i.e. left or right or top or normal"""
    x_ratio = (end_points[0] - cx)/(cx - end_points[2])
    y_ratio = (cy - end_points[1])/(end_points[3] - cy)
    if x_ratio > 3:
        return 1
    elif x_ratio < 0.33:
        return 2
    elif y_ratio < 0.33:
        return 3
    else:
        return 0

def process_thresh(thresh):

    thresh = cv2.erode(thresh, None, iterations=2)
    thresh = cv2.dilate(thresh, None, iterations=4)
    thresh = cv2.medianBlur(thresh, 3)
    thresh = cv2.bitwise_not(thresh)
    return thresh
```

Chapter 7

Testing

Testing is the process of evaluating a software item to detect differences between given input and expected output. Testing assesses the quality of the product. Software testing is a verification and validation process. It is a process where complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. Black box testing is often used for validation and white box testing is often used for verification.

1. Black-box Testing

Black-box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

2. White-box Testing

White-box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing.

7.1 Test Cases

Testing was done under various conditions to find out how the system performed.

The different test cases done were as follows

1. Ideal Conditions (Proper lighting and no spectacles).

Fig 7.1.1. Ideal Condition (Drowsiness Detection)

Fig 7.1.2. Ideal Condition (Eye Movement)

2. A Person with specs.

Fig 7.1.3. A Person with Specs (Drowsiness Detection)

Fig 7.1.4. A Person with Specs (Eye Movement)

3. Poor lighting conditions.

Fig 7.1.5. Poor Light Condition

4. When sunlight is coming from behind the user.

Fig 7.1.6. Light is coming from Behind

7.2 Types of Testing

1. **Unit Testing**

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. We have performed this testing in order to check if the individual modules work properly or not.

2. **Integration Testing**

Integration testing is testing in which a group of components is combined to produce output. It may fall under both white box testing and black-box testing. We carried out this testing to check if our project works properly after integrating all the modules.

3. **Functional Testing**

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black-box testing.

4. **Performance Testing**

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black-box testing. To check how better our system can work in all environments we performed this testing.

5. **Acceptance Testing**

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. This testing was carried out in order to check if the model prepared by us was approved by our professors or not.

Chapter 8

Results and Discussions

The accuracy of the system was calculated by carrying out various tests under various lighting conditions and various facial conditions such as spectacles and without spectacles. The accuracy for the two systems is as follows:

Test Cases	**Drowsiness Detection**	**Proctoring System**
Ideal Condition	85.6%	88.9%
Person with Specs	70.3%	74%
Poor Lighting Conditions	45.6%	51.3%

Light is coming from Behind	46.2%	53.2%
Overall Accuracy	61.92	66.85

The system struggles in poor light conditions and when the light is coming from directly behind the student. This is because in these conditions the camera struggles to find the face in the frame and as a result, it cannot locate the eyes and therefore no further processing is done. Students attend lectures and give exams under ideal lighting conditions so the overall accuracy comes down to:-

Test Cases	**Drowsiness Detection**	**Proctoring System**
Ideal Condition	85.6%	88.9%
Person with Specs	70.3%	74%
Overall Accuracy	77.95%	81.45%

Chapter 9

Conclusion and Future Scope

9.1 Conclusion:

Student attention detection completely meets the objectives and requirements of the system. It will use real-time video of the student and will make a judgment whether or not the student is feeling sleepy or is attentive in the lecture. This will help the teacher to monitor whether students are paying attention in the online class or not. Also with the proctoring system, teachers can come to know if a student is giving the exams fairly or doing any malpractices.

9.2 Future Scope:

This project can further be combined with various other Video calling/ Meeting applications such as Google Meet, Zoom, MS Teams, etc. It can be used to detect these features for all the participants at the same time and give the summary to the Host of the meeting. Also, the accuracy of the system can be improved. It can further be implemented across the ERP of the

colleges to take attendance automatically and store the data, thus reducing the teacher's work.

References

[1] Fouzia, R. Roopalakshmi, J. A. Rathod, A. S. Shetty and K. Supriya, "Driver Drowsiness Detection System Based on Visual Features," *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, 2018, pp. 1344-1347, doi: 10.1109/ICICCT.2018.8473203.

[2] K. Satish, A. Lalitesh, K. Bhargavi, M. S. Prem and T. Anjali., "Driver Drowsiness Detection," 2020 International Conference on Communication and Signal Processing (ICCSP), 2020, pp. 0380-0384, doi: 10.1109/ICCSP48568.2020.9182237.

[3] C. Yashwanth and J. S. Kirar, "Driver's Drowsiness Detection," TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), 2019, pp. 1622-1625, doi: 10.1109/TEN

CON.2019.8929429.

[4]Tobias Brächter, Dietmar Gerhardt (2020); Camera Image Based Method of Real Time Gaze Detection and Interaction; International Journal of Scientific and Research Publications (IJSRP) 10(11) (ISSN: 2250-3153), DOI: http://dx.doi.org/10.29322/IJSRP.10.11.2020.p10777

Acknowledgements

It has been a sincere desire of every individual to get an opportunity to express his/her views and skills, attitude and talent in which he/she is proficient so as to give them satisfaction and confidence in their ability to do or produce something useful for mankind.

The satisfaction that accompanies the successful completion of any task would be incomplete without acknowledging those who have made it possible and those whose constant encouragement and guidelines has been a source of inspiration throughout the course of this project.

We take this opportunity to express our gratitude and deep regards towards our project guide **Ms.Varsha Nagpurkar**, St. Francis Institute of Technology, for her constant encouragement, support, constructive criticism and guidance in our endeavor, without which we would have found it difficult to maintain our tempo and enthusiasm. Working with her has been a wonderful experience.

We also thank our project coordinators who had, through their meticulous planning, created a comfortable co-existence of the project and college schedule. We also thank our **HOD Dr. Kavita Sonawane** who played a key role in scheduling all of this.

Lastly, but not the least we would like to thank the Almighty, our parents, siblings and

friends for their constant support and encouragement without which this project would not be possible.