NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A

MINI PROJECT REPORT

ON

# "INTELLIGENT PUPIL MOVEMENT DETECTOR FOR ONLINE EXAMINATIONS"

Submitted in the partial fulfilment of the requirements in the 5th semester of

**BACHELOR OF ENGINEERING**

IN

**INFORMATION SCIENCE AND ENGINEERING**

BY

**ANARGHYA RAO – 1NH19IS012**

**SOWMYA .S – 1NH19IS161**

FOR

**COURSE NAME: MINI PROJECT**

**COURSE CODE: 20ISE59A**

***Under the guidance of,***

**Dr. Gautam K S**
Senior Assistant Professor,
Dept. of ISE, NHCE

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

**NEW HORIZON COLLEGE OF ENGINEERING**

(Autonomous College Permanently Affiliated to VTU, Approved by AICTE, Accredited by NAAC with 'A' Grade & NBA)

Ring Road, Bellandur Post, Near Marathahalli,

Bengaluru-560103, INDIA

www.newhorizonindia.edu

**NEW HORIZON**

**COLLEGE OF ENGINEERING**

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC

Accredited by NAAC with 'A' Grade, Accredited by NBA

# CERTIFICATE

Certified that the project work entitled "INTELLIGENT PUPIL MOVEMENT DETECTOR FOR ONLINE EXAMINATIONS" carried out by Ms. ANARGHYA RAO and Ms. SOWMYA .S, bearing USN, 1NH19IS012 and 1NH19IS161, respectively, bonafide students of 5th semester in partial fulfilment for the award of Bachelor of Engineering in Information Science & Engineering of the Visveswaraiah Technological University, Belagavi during the year 2021-22. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated. The project report has been approved as it satisfies the academic requirements in respect of Mini Project work prescribed for the said Degree.

| | | |
|---|---|---|
| Name & Signature of Guide | Name & Signature of HOD | Name & Signature of Principal |
| Dr. Gautam K S | Dr.Anandhi R.J. | Dr.Manjunatha |

**Examiners:**

**Name** **Signature**

1. ……………………………………… ……………………………….

2. ……………………………………… …………………………………

NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade

## DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

## CERTIFICATE ON PLAGIARISM CHECK

| | | |
|---|---|---|
| 1 | Name of the Student | ANARGHYA RAO<br>SOWMYA .S |
| 2 | USN | 1NH19IS012<br>1NH19IS161 |
| 3 | Course | UG |
| 4 | Department | ISE |
| 5 | Mini Project/Project Report / Internship Report/Seminar Report/ Paper Publication/Ph.D Thesis | Mini Project |
| 6 | Title of the Document | INTELLIGENT PUPIL MOVEMENT DETECTOR FOR ONLINE EXAMINATIONS |
| 7 | Similar Content(%) identified | 9% |
| 8 | Acceptable maximum limit (%) of similarity | 30% |
| 9 | Date of Verification | 09.02.2022 |
| 10 | Checked by (Name with signature) | Dr. Mangayarkarasi |
| 11 | Specific remarks, if any : | 1st Attempt |

We have verified the contents as summarized above and certified that the statement made above are true to the best of our knowledge and belief.

**Research coordinator**

# Abstract

Salient features of the project are: -

1) The project has been implemented using python, HTML, CSS and JavaScript.
2) The code for the detector is written using the popular python library, OpenCV, which also includes the famous “Haar Cascade Classifier”, used to detect the face and eye. The python micro-framework, Flask, is being used to create the web application.
3) The backend of the project consists of the code for the detector. Frontend consists of the website through which users interact with the detector and see the results.
4) The detector accurately detects the pupil and tracks it.
5) Users can choose to either upload a video or use the webcam for detection.

By implementing the pupil detector, this project takes a step towards building a better online learning environment.

## Acknowledgement

Any project is a task of great enormity and it cannot be accomplished by an individual without support and guidance. We are grateful to a number of individuals whose professional guidance and encouragement has made this project completion a reality.

We have great pleasure in expressing our deep sense of gratitude to the beloved Chairman Dr. Mohan Manghnani for having provided us with great infrastructure and well-furnished labs.

We take this opportunity to express our profound gratitude to the Principal Dr. Manjunatha for his constant support and management.

We are grateful to Dr. R J Anandhi, Professor and Head of Department of ISE, New Horizon College of Engineering, Bengaluru for her strong enforcement on perfection and quality during the course of our project work.

We would like to express our thanks to the guide Mr. Gautam, Professor, Department of ISE, New Horizon College of Engineering, Bengaluru who has always guided us in detailed technical aspects throughout our project.

We would like to mention special thanks to all the Teaching and Non-Teaching staff members of Information Science and Engineering Department, New Horizon College of Engineering, Bengaluru for their invaluable support and guidance.

ANARGHYA RAO (1NH19IS012)

SOWMYA .S (1NH19IS161)

## TABLE OF CONTENTS

# List of tables and figures

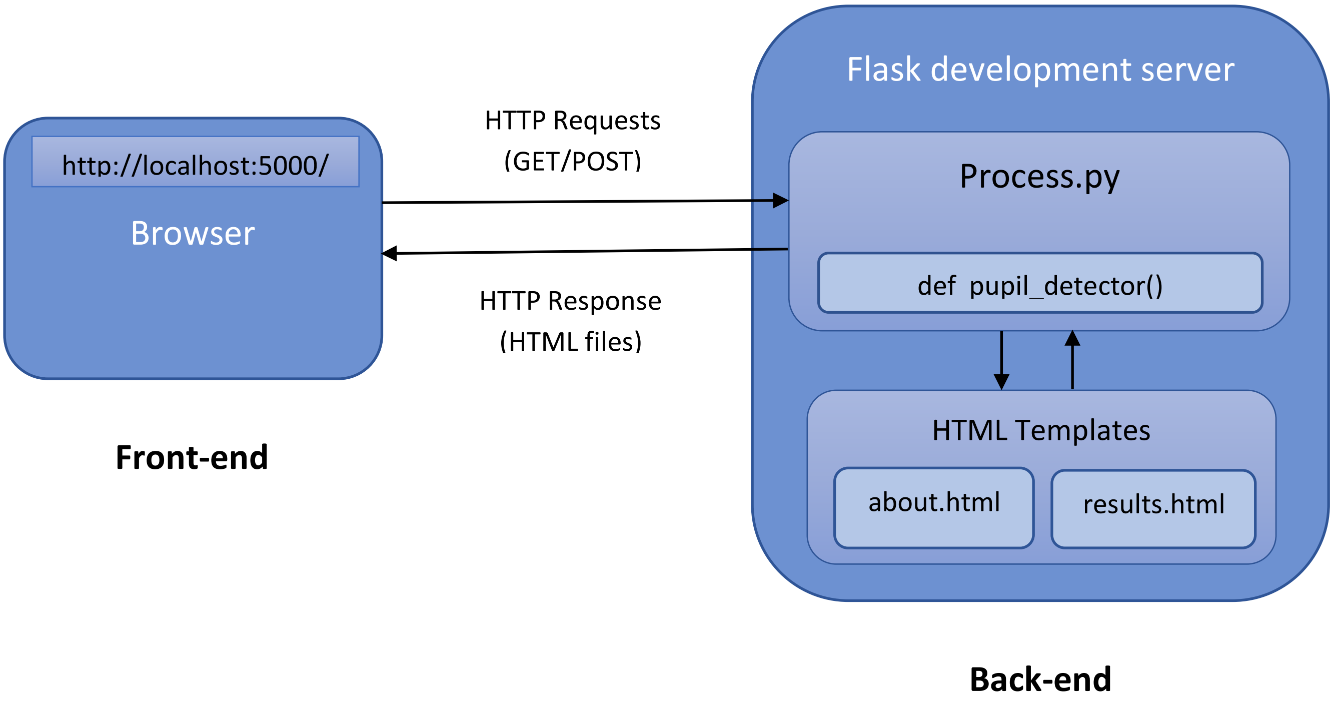


**Fig3.1**: Block diagram

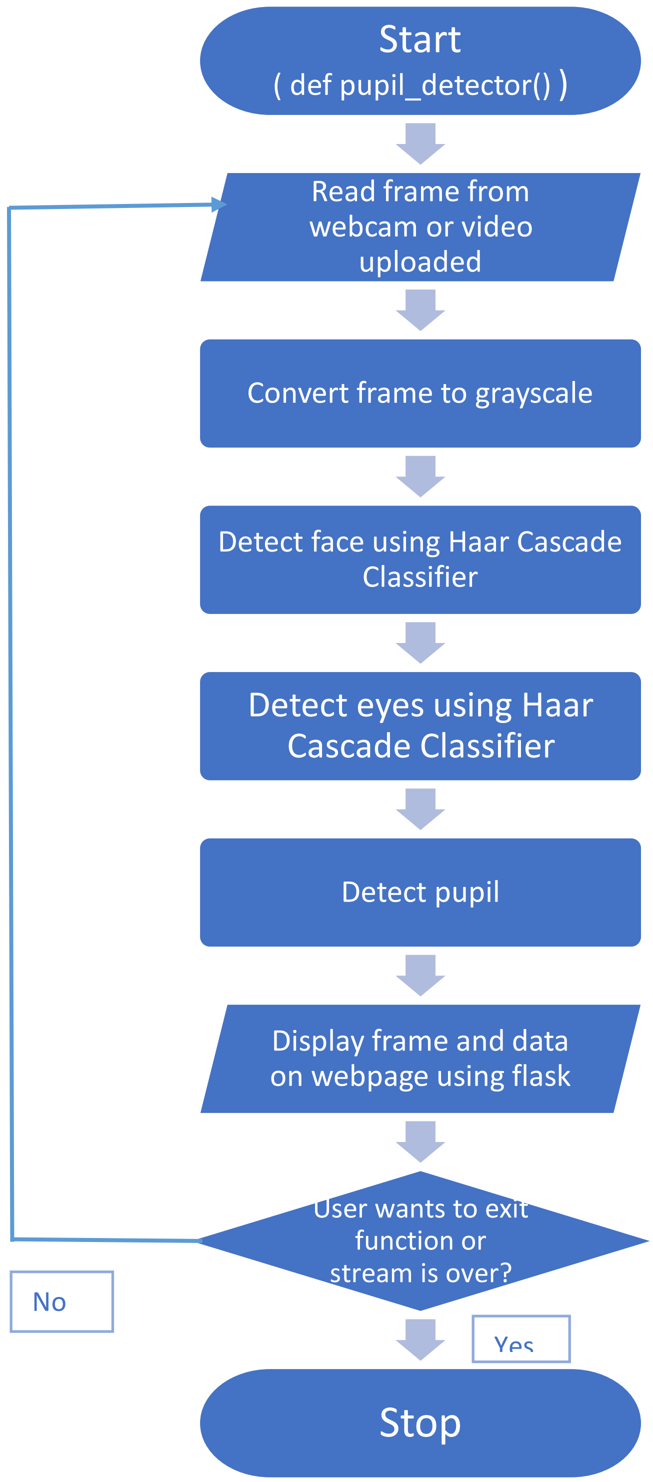

Start
( def pupil_detector() )
Read frame from webcam or video uploaded
Convert frame to grayscale
Detect face using Haar Cascade Classifier
Detect eyes using Haar Cascade Classifier
Detect pupil
Display frame and data on webpage using flask
User wants to exit function or stream is over?
No
Yes
Stop

Fig3.2: Pupil Detector flowchart

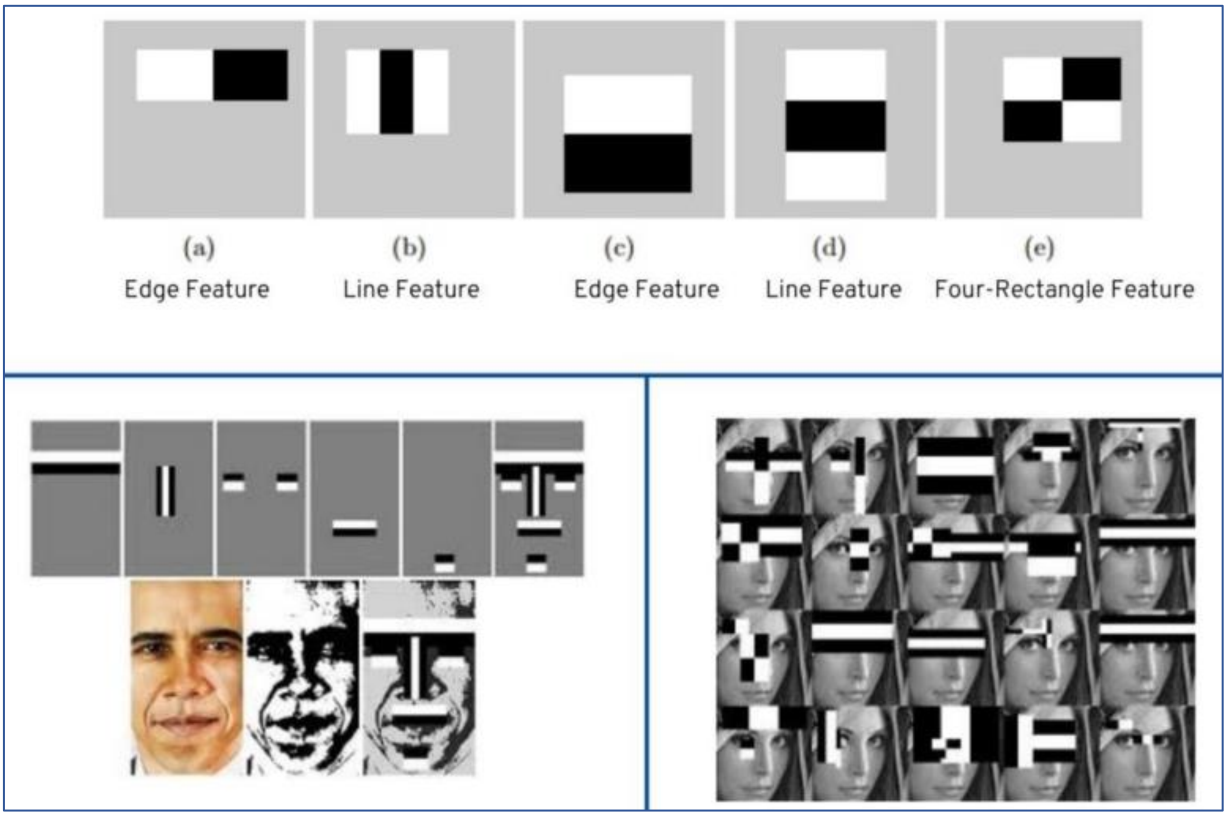


Fig3.3: Haar features examples

# CHAPTER 1: INTRODUCTION

## 1.1 Motivation of Project

The Covid-19 pandemic has been a truly unprecedented event in modern history, having affected nearly all countries, with lockdowns forcing millions to rapidly adapt to working, learning and playing while being stuck indoors. Within the span of a year or less, the world has had to shift almost entirely online. The education sector has been no stranger to this shift and in fact, has seen some of the most dramatic changes of all, with e-learning seeing a major rise in participation and enrolments.

**E-learning** or **online learning** refers to learning conducted via electronic media, typically on the internet. It encompasses various ways to learn online like MOOCs (Massive Open Online Courses) (Example Coursera, Udemy, etc.), with EdTech companies (Example Byju's), etc. Online learning can also be synchronous or asynchronous, which allow flexibility. Online learning has many more advantages such as more personalized learning environments, saving of time and money, cost-effectiveness and allowing easy access to quality education to anyone, etc.. But Online learning also has some disadvantages. With now having experienced nearly two years of online school/college, educators have been heard raising one common, but major issue, over and over on online education i.e., the veracity of online examinations. Examinations conducted through online modes are more susceptible to malpractice by students. With proctors unable to adequately monitor students, students are able to indulge in cheating. This is a big problem which damages the learning process of the student and brings down the quality of education received.

Thus, solutions to this problem are in urgent requirement, which can be implemented during online examinations using already in-use resources, E.g., Webcams. With all this in mind, we were inspired to build an "Intelligent Pupil Detector for Online Examinations", to better the scene of online examinations and eventually improve the learning process of students during these pandemic times.

## 1.2 Problem Statement

- Online examinations conducted by schools and colleges cannot be easily proctored, which allows students to easily cheat, damaging the learning process and outcome.
- One solution is to track the movement of the eyes (using pupils) of the students using webcams during online examinations.
- This pupil tracking can allow exam monitoring software to recognize signs of cheating E.g., rapidly shifting eyes.
- Thus, this will discourage malpractices and improve the quality of online education.

# CHAPTER 2: SYSTEM REQUIREMENT

## 2.1 Software/Hardware used

### 2.1.1 Software:

The program was built using the following software: -

1) Python 3.10.1
2) Python framework and libraries: Flask, OpenCV
3) Web internet programming languages: HTML5, CSS3, JavaScript
4) Operating System (OS): Microsoft Windows 10

### 2.1.2 Hardware:

The program was built using the following hardware: -

1) Processor: AMD Ryzen 7 5700U, Intel CORE i5
2) RAM: 16.00 GB, 8.00 GB
3) Laptop used: LENOVO, HP

# CHAPTER 3: SYSTEM DESIGN

## 3.1 Modules

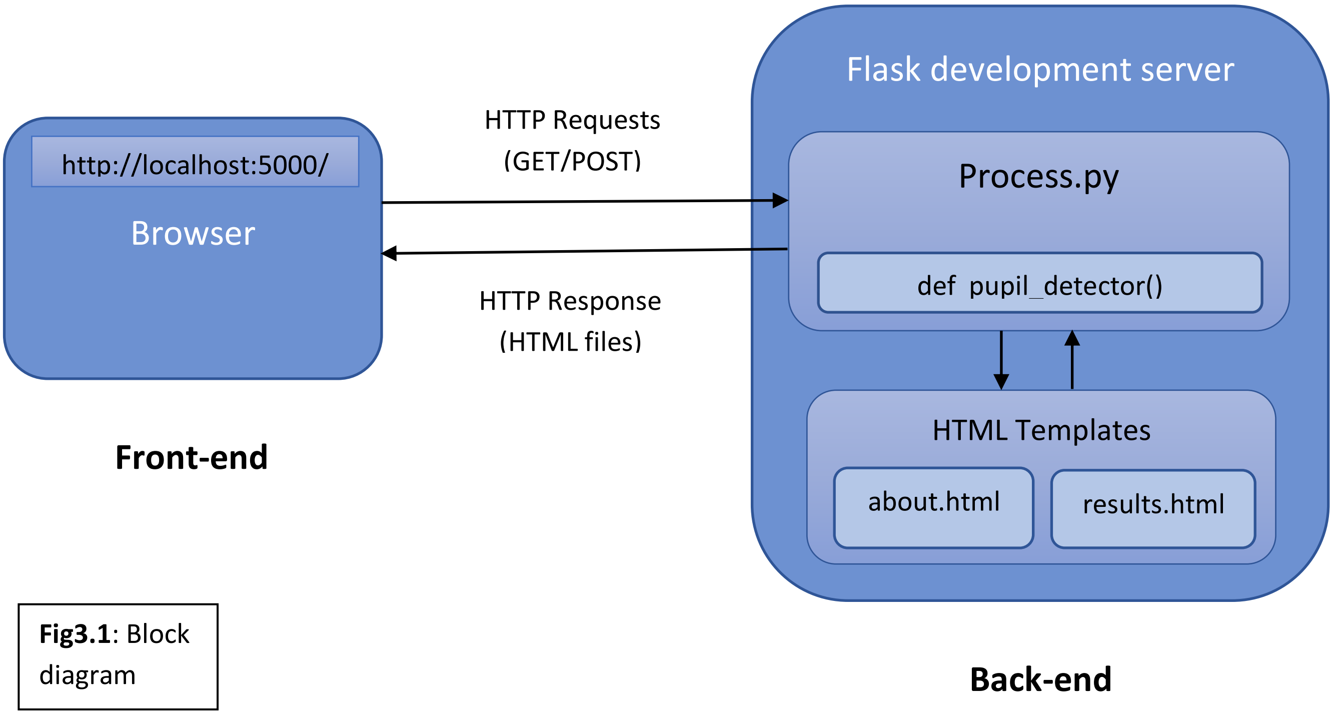


**Fig3.1**: Block diagram

The program is divided into two main modules based on functionality: -

1) Flask application module (process.py):

   This consists of a python program (process.py) that imports and uses the micro-framework for web development, “Flask” and the library for Computer Vision and Image processing, “OpenCV”.

   This module is to be executed to start running the project. It handles HTTP requests and responses, executing specific functions based on the URL specified. These functions are used to render HTML templates and also execute the pupil detector code. The pupil detector code consists of the famous “Haar Cascade Classifier” and other OpenCV functions used to successfully detect the pupil and track it.

2) HTML Templates module:

This module consists of two HTML files. These HTML templates are accessed by Flask application module functions, which render them to the browser. These functions can also pass them data. The HTML files are: -

(a) about.html: This is the first web page seen by the user when the project is run. It contains information regarding the pupil detector and also contains a form, through which user may select to either use their webcam or upload a video for pupil detection. Submitting the form routes the user to “results.html”.

(b) results.html: This web page is used to host the results of the pupil detector and displays this to the user. It also provides a “Go back” button to return to “about.html”.

## 3.2 Architecture [Flowchart]

Pupil Detector:

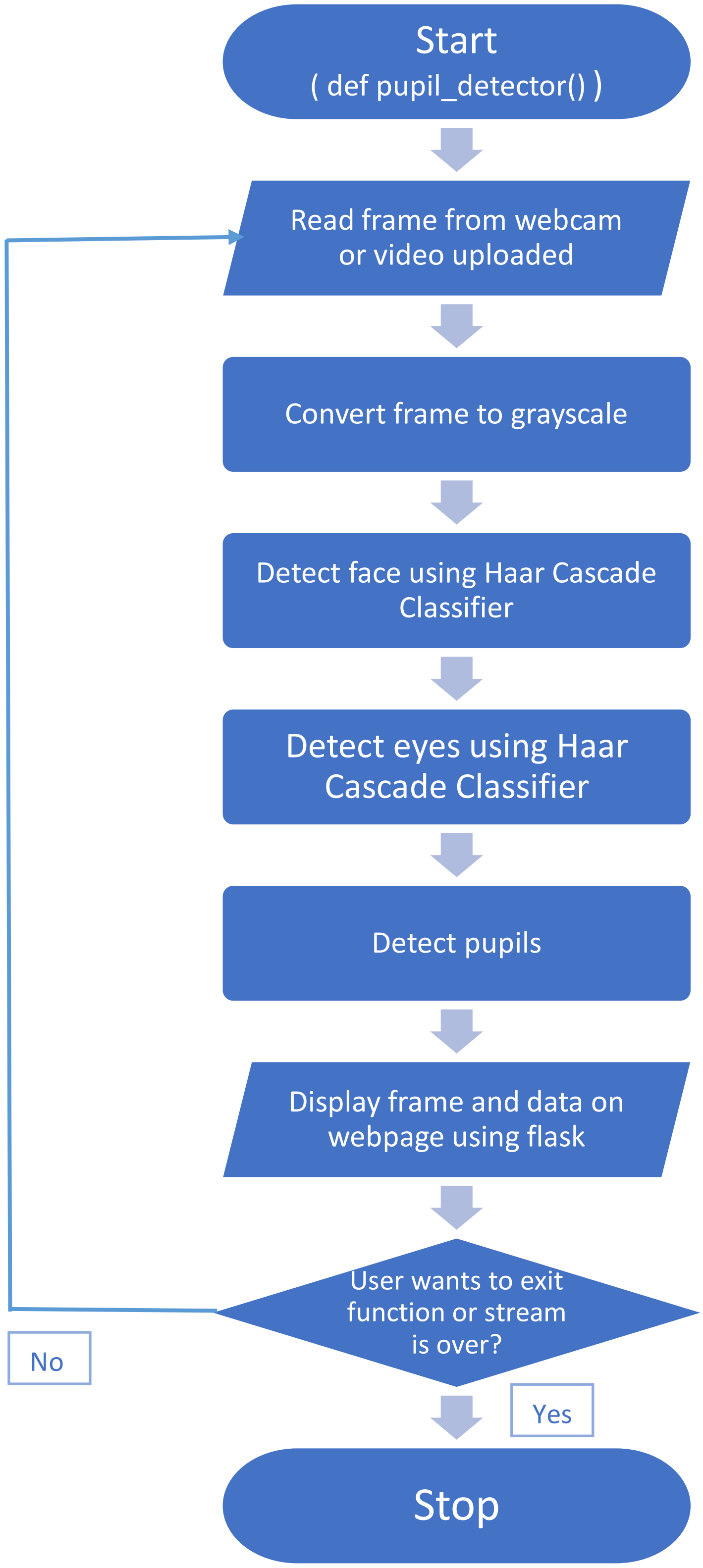

Start
( def pupil_detector() )
Read frame from webcam or video uploaded
Convert frame to grayscale
Detect face using Haar Cascade Classifier
Detect eyes using Haar Cascade Classifier
Detect pupils
Display frame and data on webpage using flask
User wants to exit function or stream is over?
No
Yes
Stop

Fig3.2: Pupil Detector flowchart

## 3.3 Algorithm used

**Step 1: Start** (process.py)

**Step 2:** "about.html" html template rendered on browser for URL http://localhost:5000/.

**Step 3:** Input user selection of webcam or video upload on "about.html".

**Step 4:** Form data returned using HTTP Post method and "results.html" requested.

**Step 5:** "results.html" is rendered.

**Step 6:** pupil_detector function called.

**Step 7:** While webcam or video stream is not over or till user does not exit function, repeat steps 8 to 13.

**Step 8:** Read frame from webcam or video uploaded.

**Step 9:** Convert frame to grayscale.

**Step 10:** Detect face using Haar Cascade Classifier.

**Step 11:** Detect eyes using Haar Cascade Classifier.

**Step 12:** Detect pupils.

**Step 13:** Display frame and data on "results.html".

**Step 14:** If user clicks on "Go back" button, return to Step 2. Else proceed to next step.

**Step 15: Stop**

### 3.3.1 Haar Cascade Classifier

A **Haar classifier**, also called a **Haar cascade classifier**, is a machine learning object detection program that identifies objects in image and video. It was proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier.

It uses "**Haar features**", extracted while training and stored in an XML file, to identify the object. A "Haar feature" consists of calculations that are performed on adjacent rectangular regions at a specific location in a detection window. The calculation involves summing the pixel intensities in each region and calculating the differences between the sums. Fig 3.3 shows some examples of haar features. Input images are compared with the haar features. If it passes through all the stages of haar feature comparison, then the object is said to be detected.

But among all the features calculated during training, most are irrelevant and including them leads to a large increase in computation required. Therefore, to pick only the most relevant features, an algorithm called **“AdaBoost”** is used.

AdaBoost stands for “Adaptive Boosting”. It is used to essentially choose the best features and then trains the classifier to use them. It uses a combination of “weak classifiers” to create a “strong classifier” that the algorithm can use to detect objects.

OpenCV provides pretrained models of Haar classifier that can be read using the **cv2.CascadeClassifier()** method. The pretrained models are located in the data folder in the OpenCV installation.

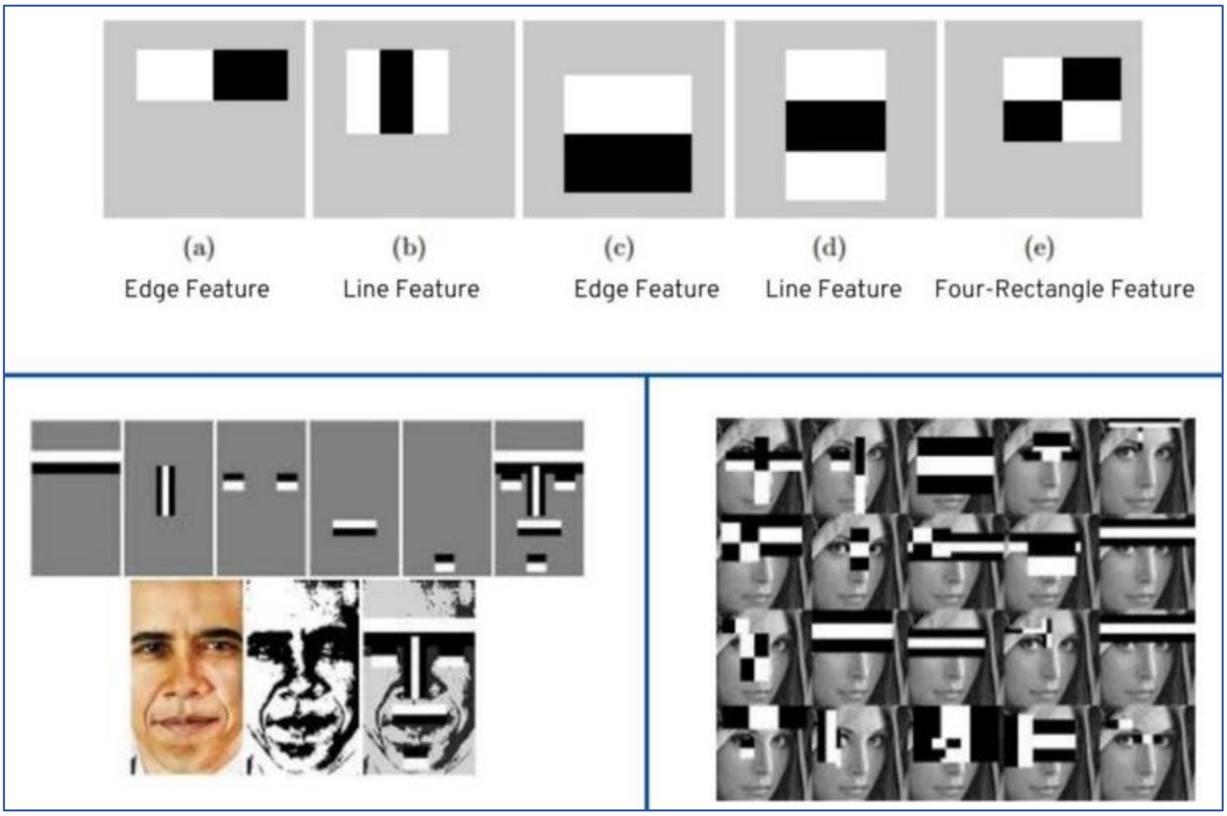


Fig3.3: Haar features examples

## 3.4 Code and implementation

### 3.4.1 Process.py

```
 from flask import Flask, render_template, request,Response
import cv2,imutils,time
import pyshine as ps
app = Flask(__name__)
@app.route('/')
def index():
   return render_template('about.html')

def pupil_detector(params):
    if(params['mode'] == "WEBCAM"):
        cap = cv2.VideoCapture(0)
    else:
        path=str(params['file'])
        cap = cv2.VideoCapture('videos\\'+path)
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
    eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')
    print('FUNCTION DONE')
    center=[0, 0]
    pupil_loc=[0, 0]
    while True:
        ret, frame = cap.read()
        if ret is False:   #If video ends
            cap.release()  #If webcam used, release the resource
            break
        rows, cols, _ = frame.shape
        gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        faces = face_cascade.detectMultiScale(gray_frame, 1.3, 5)
        for (fx, fy, fw, fh) in faces:
            cv2.rectangle(frame, (fx, fy), (fx + fw, fy + fh), (255, 0, 0), 5)
            face_gray = gray_frame[fy:fy+fw, fx:fx+fw]
            face_color = frame[fy:fy+fh, fx:fx+fw]
            eyes = eye_cascade.detectMultiScale(face_gray, 1.3, 5)
            count=0
            for (ex, ey, ew, eh) in eyes:
                #print("INSIDE EYE DETECTOR")
                cv2.rectangle(face_color, (ex, ey), (ex + ew, ey + eh), (0, 255, 0), 5)

                center[0] = ex + ew/2
                center[1] =  ey + eh/2

                eye_gray = face_gray[ey:ey+ew, ex:ex+ew]
                eye_color = face_color[ey:ey+ew, ex:ex+ew]

                _, threshold = cv2.threshold(eye_gray, 25, 255, cv2.THRESH_BINARY_INV)   # 255
is white , 0 is black

                contours, _ = cv2.findContours(threshold, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
                contours = sorted(contours, key=lambda x: cv2.contourArea(x), reverse=True)
                for cnt in contours:
                    #print("INSIDE PUPIL DETECTOR")
                    (x, y, w, h) = cv2.boundingRect(cnt)

                    pupil_loc[0] = ex + x + w/2
                    pupil_loc[1] = ey + y + h/2



                    cv2.rectangle(eye_color, (x, y), (x + w, y + h), (255, 0, 0), 2)
                    cv2.line(eye_color, (x + int(w/2), 0), (x + int(w/2), rows), (0, 255, 0), 2)
                    cv2.line(eye_color, (0, y + int(h/2)), (cols, y + int(h/2)), (0, 255, 0), 2)
                    break

                count+=1
                if(count == 2):
                    break

            direction = ["", ""]
            print(center[0], center[1])
            print(pupil_loc[0], pupil_loc[1])
            haxis =  center[0] - pupil_loc[0]
            vaxis = center[1] - pupil_loc[1]
            print(haxis, vaxis)
            if(haxis > 5):
                direction[0] = "RIGHT"
            elif(haxis < -5):
              direction[0] = "LEFT"
            else:
                direction[0] = "CENTRE"


            if(vaxis > 4):
                direction[1] = "TOP"
            elif(vaxis < -4):
                direction[1] = "DOWN"
            else:
                direction[1] = "CENTRE"
            print(direction[0], direction[1])
            print()
            cv2.putText(frame, direction[1], (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0,
    0), 2)
            cv2.putText(frame, direction[0], (125, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,
    0, 0), 2)


            frame = imutils.resize(frame, width=800)   #resize video width


            #Encoding frame to bytes to send
            stream = cv2.imencode('.JPEG',
    frame,[cv2.IMWRITE_JPEG_QUALITY,20])[1].tobytes()
            time.sleep(0.016)
            yield (b'--frame\r\n'b'Content-Type: image/jpeg\r\n\r\n' + stream + b'\r\n')
            if(params['mode']=="WEBCAM"):
                time.sleep(0.08)
            break
```

```
@app.route('/res',methods = ['POST','GET'])
def res():
    global result
    if request.method == 'POST':

        result = request.form.to_dict()

           #If mode is video, save video provided by user
        if(result['mode']=='VIDEO'):
            f = request.files['file']
            filename = f.filename
            if(filename==""):   #If no video provided by user, use default video
                filename = "default.mp4"
            else:
                f.save('videos/' + filename)
            result['file'] = filename
        else:      #If webcam option selected
            result['file']=""

        return render_template("results.html",result = result)


@app.route('/results')
def video_feed():
           global result
           params= result
           return Response(pupil_detector(params),mimetype='multipart/x-mixed-replace;
boundary=frame')
```

```
if __name__ == "__main__":
   app.run(debug=True,threaded=True)
```

3.4.2 About.html

```
<html>
   <head>
      <title>Pupil Movement Detector</title>

      <script type="text/javascript">
 function show(){
 document.getElementById("f").style.visibility = "visible";
 }

 function hide(){
 document.getElementById("f").style.visibility = "hidden";
 }
 </script>

 <link rel="stylesheet" type="text/css" href="static\style.css">

   </head>
   <body>
      <div id="background-image">
         <h1> Pupil Movement Detector </h1>

        <div class="color-overlay">
```

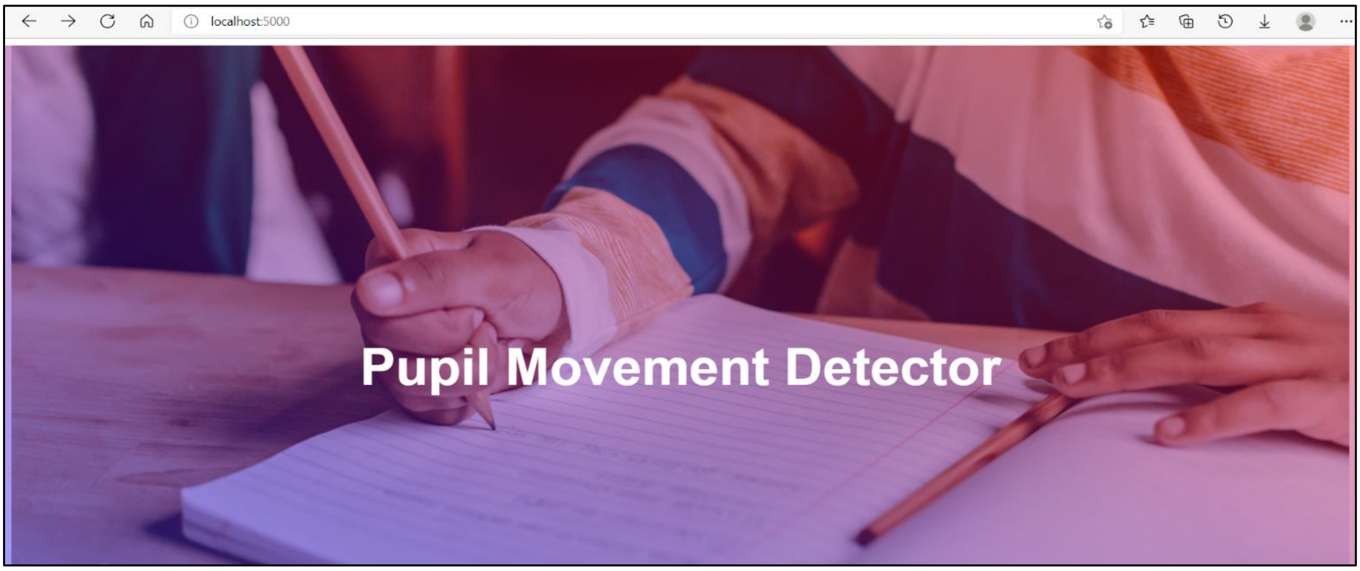
```
          </div>


         </div>


      <div class="fcontainer">

      <div class="content-section">



<div class="content">

<h2>Introduction</h2>

<p>The Covid-19 pandemic has been a truly unprecedented event in modern history, having affected nearly all countries, with lockdowns forcing millions to rapidly adapt to working, learning and playing while being stuck indoors. <br>

   Within the span of a year or less, the world has had to shift almost entirely online. The education sector has been no stranger to this shift and in fact, has seen some of the most dramatic changes of all, with e-learning seeing a major rise in participation and enrolments.</p>


<p><b>E-learning</b> or <b>online learning</b> refers to learning conducted via electronic media, typically on the internet. …………. This is a big problem which damages the learning process of the student and brings down the quality of education received.</p>


<p>Thus, solutions to this problem are in urgent requirement, which can be implemented during online examinations using already in-use resources, E.g., Webcams. With all this in mind, we were inspired to build an “Intelligent Pupil Detector for Online Examinations”, to better the scene of online examinations and eventually improve the learning process of students during these pandemic times.</p>


<h2><u><b>About the project</b></u></h2>
```

```
<p><ul>
<li>The project has been implemented using python, HTML, CSS and JavaScript.</li>
<li>The code for the detector is written using the popular python library, OpenCV. It also uses the famous “Haar Cascade Classifier” to detect the face and eye. The python micro-framework, Flask, is being used to create the web application.</li>
<li>The backend of the project consists of the code for the detector. Frontend consists of the website through which users interact with the detector and see the results.</li>
<li>The detector accurately detects the pupil and tracks it. It also provides useful information regarding the movement of the pupil.</li>
<li>Users can choose to either upload a video or use the webcam for detection.</li></ul>


      </div>


    </div>


    <div class="image-section">
      <img src="static/image1.jpg" alt="exam">
    </div>


    </div>


    <br>
    <br>


    <div class="sample-image">
      <h2>Sample Images Captured</h2>
     <img src="static/out1.png" width=48% border="10" style="float:left; margin-bottom:10px; border-color:#93FFE8;"/>
```

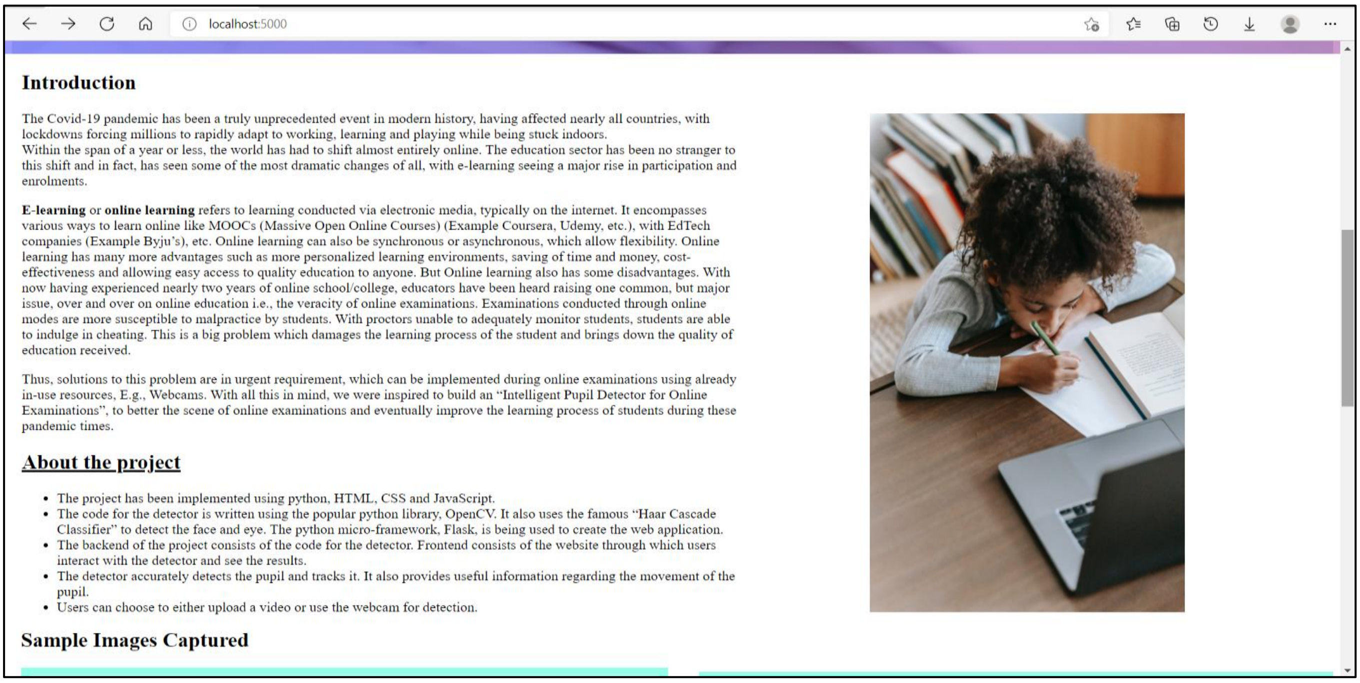
```
 <img src="static/out2.png" width=47% border="10" style="float:right; margin-
bottom:50px; border-color:#93FFE8;"/>


       </div>


<div class="scontainer">
    <div class="stitle">
       <h2><u>Instructions for use:</u></h2></div>
<ol>
<li>Input to the pupil detector can be given as a <b>video</b> or through
<b>webcam</b>. Select preferred option from the form given below.
<ul style="list-style-type: square;">
<li><b>Video upload:</b> Select the option and upload file. If video is not uploaded,
default video will play.</li>
<li><b>Webcam:</b> Ensure that permission to use webcam is enabled.</li>
</ul>
</li>
<li>Clicking "submit" will direct you to the page where pupil detection will take place.</li>
</ol>


<blockquote style="background-color:#93FFE8; padding:5px;"><b>Note: In both of these
inputs
<ul>
<li>The face must be well within the frame</li>
<li>Both eyes must be visible (No side profile)</li>
<li>Room should be well-lit</li>
<li>Objects blocking the face may cause accuracy to fall</li></ul>
</b></blockquote>
```

```
<h2>Please input the parameters:</h2><br />

   <blockquote style="background-color:#93FFE8; padding:5px;">

        <div class="form">

        <form action = "http://127.0.0.1:5000/res" method = "POST" enctype =
"multipart/form-data">

                <h3><u>Choose mode:</u></h3>

                <label><b>Webcam</b> <input type="radio" name="mode"
value="WEBCAM" onclick="hide()" checked/></label> <br />

                <label><b>Upload a video</b> <input type="radio" name="mode"
value="VIDEO" onclick="show()"/></label>

                <div id="f">

                <label>Upload video file: <input type="file" name="file"/></label>

                </div><br />

                <input type = "submit" value = "Submit" />

                <input type = "reset" value = "Reset" onclick = "hide()"/>

        </form>

      </div> </blockquote>

</div>

<br />

<br />

<br />

<br />

<br />



   </body>

</html>
```

## Output



Pupil Movement Detector



**Introduction**

The Covid-19 pandemic has been a truly unprecedented event in modern history, having affected nearly all countries, with lockdowns forcing millions to rapidly adapt to working, learning and playing while being stuck indoors.
Within the span of a year or less, the world has had to shift almost entirely online. The education sector has been no stranger to this shift and in fact, has seen some of the most dramatic changes of all, with e-learning seeing a major rise in participation and enrolments.

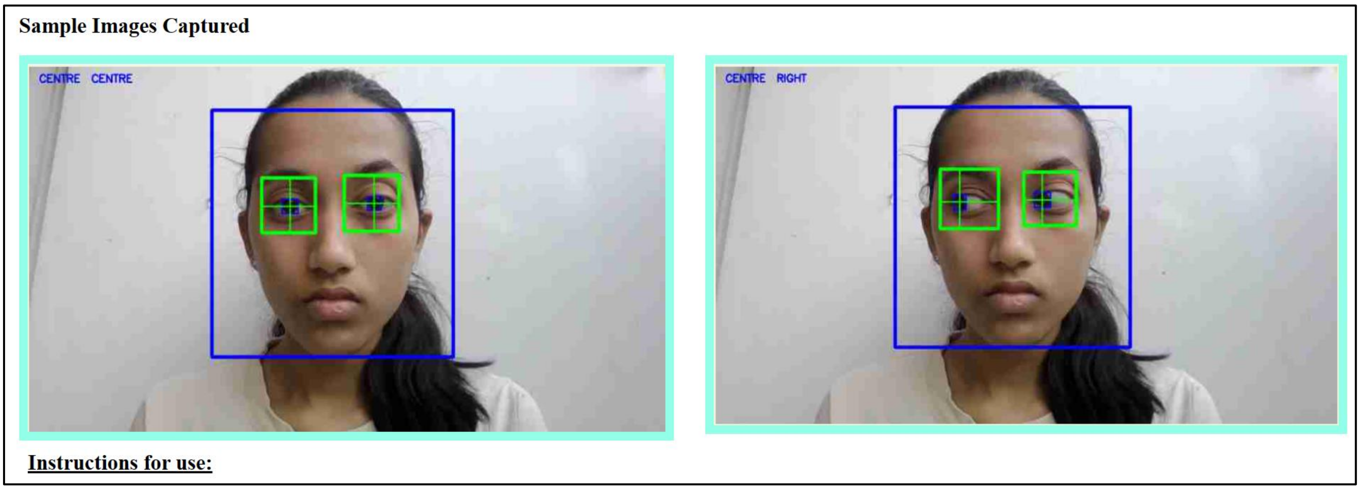
**E-learning** or **online learning** refers to learning conducted via electronic media, typically on the internet. It encompasses various ways to learn online like MOOCs (Massive Open Online Courses) (Example Coursera, Udemy, etc.), with EdTech companies (Example Byju's), etc. Online learning can also be synchronous or asynchronous, which allow flexibility. Online learning has many more advantages such as more personalized learning environments, saving of time and money, cost-effectiveness and allowing easy access to quality education to anyone. But Online learning also has some disadvantages. With now having experienced nearly two years of online school/college, educators have been heard raising one common, but major issue, over and over on online education i.e., the veracity of online examinations. Examinations conducted through online modes are more susceptible to malpractice by students. With proctors unable to adequately monitor students, students are able to indulge in cheating. This is a big problem which damages the learning process of the student and brings down the quality of education received.

Thus, solutions to this problem are in urgent requirement, which can be implemented during online examinations using already in-use resources, E.g., Webcams. With all this in mind, we were inspired to build an "Intelligent Pupil Detector for Online Examinations", to better the scene of online examinations and eventually improve the learning process of students during these pandemic times.

**About the project**

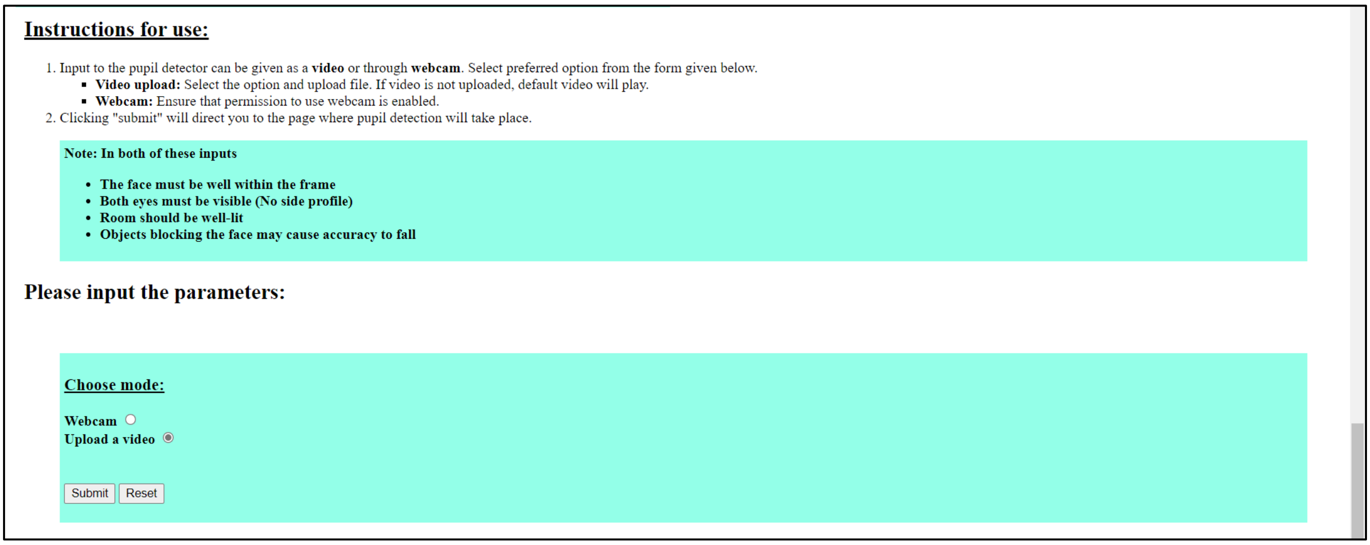
- The project has been implemented using python, HTML, CSS and JavaScript.
- The code for the detector is written using the popular python library, OpenCV. It also uses the famous "Haar Cascade Classifier" to detect the face and eye. The python micro-framework, Flask, is being used to create the web application.
- The backend of the project consists of the code for the detector. Frontend consists of the website through which users interact with the detector and see the results.
- The detector accurately detects the pupil and tracks it. It also provides useful information regarding the movement of the pupil.
- Users can choose to either upload a video or use the webcam for detection.

**Sample Images Captured**



**Sample Images Captured**

**Instructions for use:**



**Instructions for use:**

1. Input to the pupil detector can be given as a **video** or through **webcam**. Select preferred option from the form given below.
   - **Video upload:** Select the option and upload file. If video is not uploaded, default video will play.
   - **Webcam:** Ensure that permission to use webcam is enabled.
2. Clicking "submit" will direct you to the page where pupil detection will take place.

**Note: In both of these inputs**

- **The face must be well within the frame**
- **Both eyes must be visible (No side profile)**
- **Room should be well-lit**
- **Objects blocking the face may cause accuracy to fall**

**Please input the parameters:**

**Choose mode:**

**Webcam** ○
**Upload a video** ●

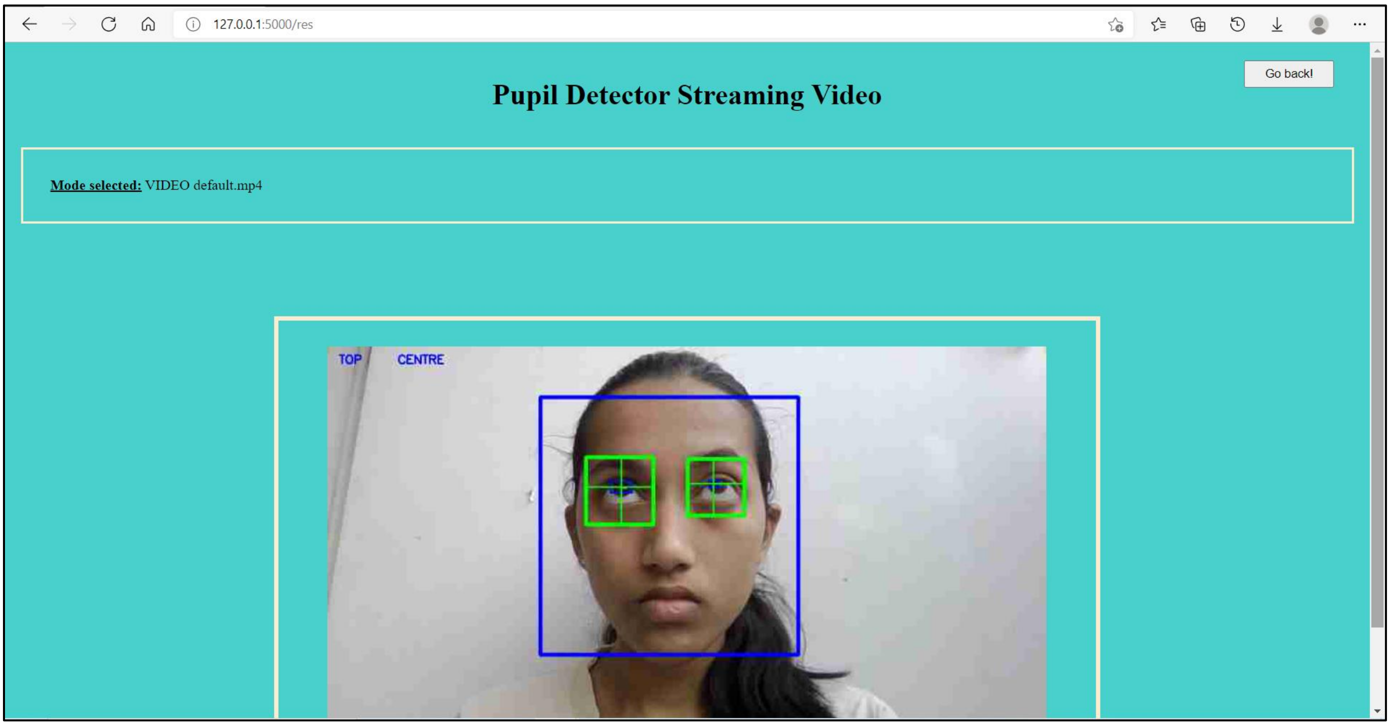
## 3.4.3 Results.html

```
<!DOCTYPE html>
<html>
<head>
<title>Pupil Movement Detector</title>
<style type="text/css">
body{
background-color:beige;
padding:10px;
}
section.a{
width:60%;
margin:30px;
padding:10px;
border: 5px blanchedalmond solid;
}
section.b{
```

```
border: 3px blanchedalmond solid;

padding: 30px;

}

</style>

</head>


<body style="background-color:MediumTurquoise;">

   <div class="resut">


 <h1><center>Pupil Detector Streaming Video</center></h1>

<input type="button" value="Go back!" onclick="history.back()" style="position:absolute;
top: 20px; right:40px; width:100px; height:30px;">

<br />


<section class="b">

<u><b>Mode selected:</b></u> {{ result['mode'] }}  {{ result['file'] }}<br />

</section>

<br>

<br>

<br>

<br>

<center>

<section class="a">

<br />

<img src="{{ url_for('video_feed') }}"/>

<br />

</section>

</center>
```

```
   </div>

</body>

</html>
```

Output



127.0.0.1:5000/res

Go back!

Pupil Detector Streaming Video

**Mode selected:** VIDEO default.mp4

### 3.4.4 style.css

```
#background-image{

   height: 100vh;

   background-image: url("background.jpg");

   background-size: cover;

   background-position: center;

   font-family: sans-serif;

   display: flex;

   align-items: center;

   justify-content: center;

}

#f{
   visibility:hidden;

  }

h1{ font-size: 60px;


z-index: 1;

   color: #fff;

  }

  .color-overlay{ background: linear-gradient(to top right, blue, red);

                width: 100%;

                height: 100%;

                opacity: .4;

                position: absolute;

  }

 .container{

        width: 80%;

        display: block;

        margin:auto;

        padding-top: 100px;


}

.content-section{

        float: left;

        width: 55%; }

.image-section{

        float: right;

        width: 40%;

}
```

```
.image-section img{
        width: 60%;
        height: 80%;


  padding: 70px;
}
.fcontainer{padding-left: 11px;}
.scontainer{padding-left: 20px;}
.sample-image{padding-left: 10px;}
```

# CHAPTER 4: RESULTS AND DISCUSSION

## 4.1 Results

- The pupil movement detector successfully detects and tracks the face, eyes and pupils.
- It provides useful information regarding the pupil movement like direction of sight.
- The user interface is simple and easy to understand.

## 4.2 Conclusion

- The “Intelligent Pupil Movement Detector” successfully tracks the pupil of the student and provides accurate information on the movement of the pupil.
- The project mainly requires the use of the OpenCV library and Haar Cascade classifier for detection and image processing. It also requires utilization of the python framework Flask to create a web application.
- Future improvements include creation of better and more enhanced classifiers for improved detection and better user-interface design.
- Future applications for this project include integration into a full-fledged exam monitoring software.
- Such exam monitoring software can lead to better learning environments and outcomes, therefore increasing the quality of online examinations. Thus, adding fuel to fire of the already growing E-learning sector.

# References

- “Data Science from Scratch”, Joel Grus
- “Machine Learning using Python”, Manaranjan Pradhan and U Dinesh Kumar
- Eye motion tracking - Opencv with Python – Pysource
- How to deploy Python video processing web application using Flask - YouTube
- Haar Cascades, Explained. A brief introduction into Haar… | by Aditya Mittal | Analytics Vidhya | Medium
- www.GeeksforGeeks.com
- www.javatpoint.com