Jaypee University of Engineering And Technology, Guna

BLINK

by: **(Project Number: 51)**

Anshika Soni (201B052)

Dishita Jain(201B099)

Ishita Jain(201B126)

Under the Supervision of

Dr. Amit Kumar Srivastava

Asst. Professor (SG)

Department of Computer & Engineering

Table of Contents

Points to discuss:

- Motivation
- Introduction
- Requirement Analysis
- Block Diagram
- Use Case Diagram
- Flow Chart
- General Structure of DesignedSystem
- Implementation
- Expected Outcome
- References

Motivation:

- **A mouse cursor** , also known as A mouse arrow, or mouse pointer, is a graphical image used to activate or control certain elements in a graphical user interface.

- The mouse pointer follows the path of the user's hand as they move their mouse. Using it for long hours puts the user's hand in a bad posture that increases inflammation in the wrist and hand.

- This current technology i.e. **A mouse cursor** cannot be used by some people like 'amputees' as they do not have their hands to operate.

- We therefore investigate the introduction of eye movements as a computer input medium.

- It is particularly important for disabled people who cannot interact with computers in a normal manner.

INTRODUCTION

- Moving the finger on the computer mouse has become a common way to move the mouse cursor along the computer screen in the present technology.
- The computers which have been used by us can be improved and can give us more efficient results by replacing the mouse with the eyes.
- The idea behind this system is to enhance the interaction of a person with the computer.
- Eye gaze tracking is a process of detecting and measuring eye movements.
- This system explores the potential of the human eye gaze which can be used as a pointing device on the computer.

Requirement Analysis:

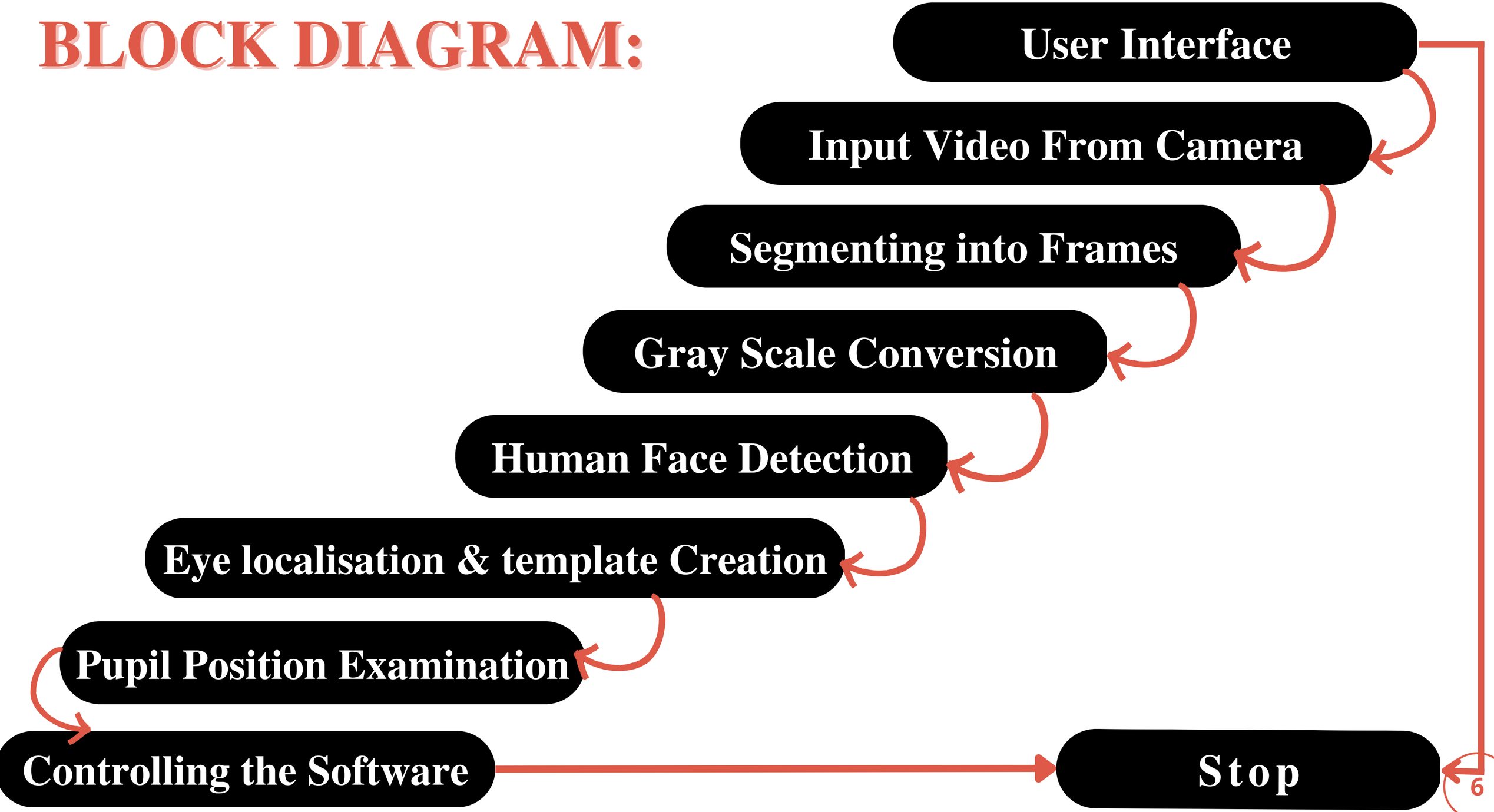
Software Requirement

- Python
- Visual Studio Code
- OpenCv
- Mediapipe
- Pyautogui

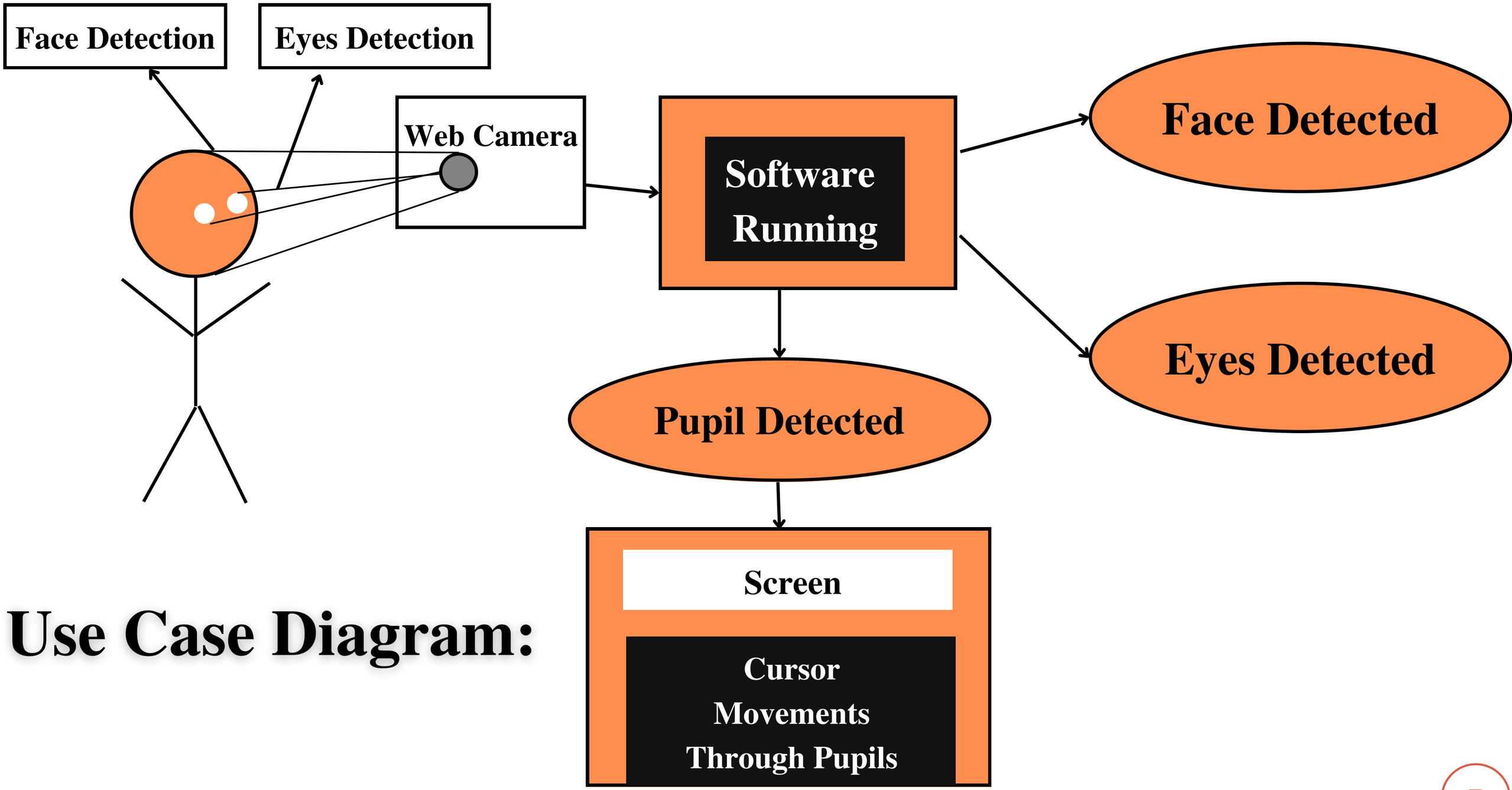
Hardware Requirement

- At least 4 GB RAM
- Web Cam

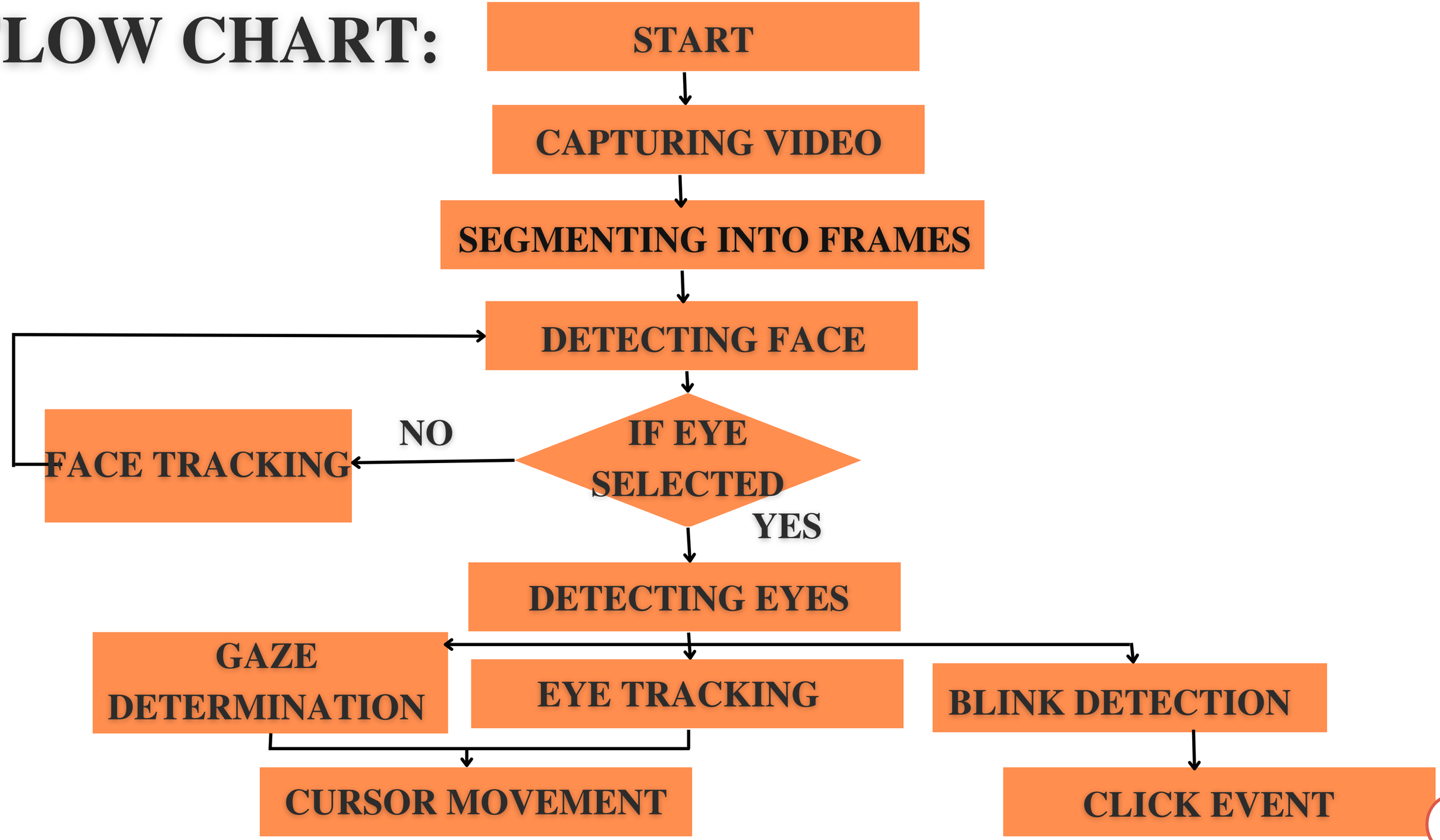
BLOCK DIAGRAM:



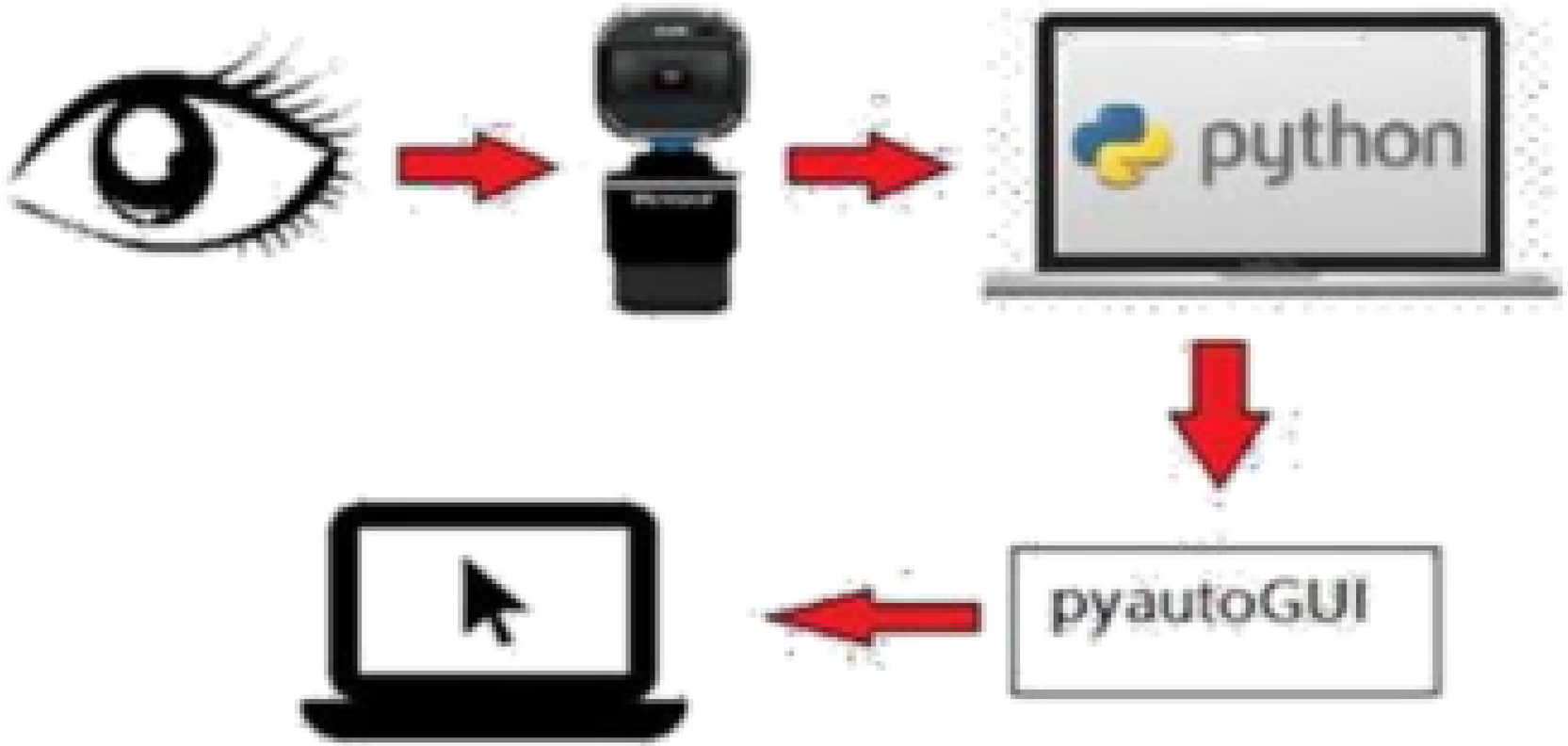
Use Case Diagram:



FLOW CHART:


START
CAPTURING VIDEO
SEGMENTING INTO FRAMES
DETECTING FACE
IF EYE SELECTED
NO
FACE TRACKING
YES
DETECTING EYES
GAZE DETERMINATION
EYE TRACKING
BLINK DETECTION
CURSOR MOVEMENT
CLICK EVENT

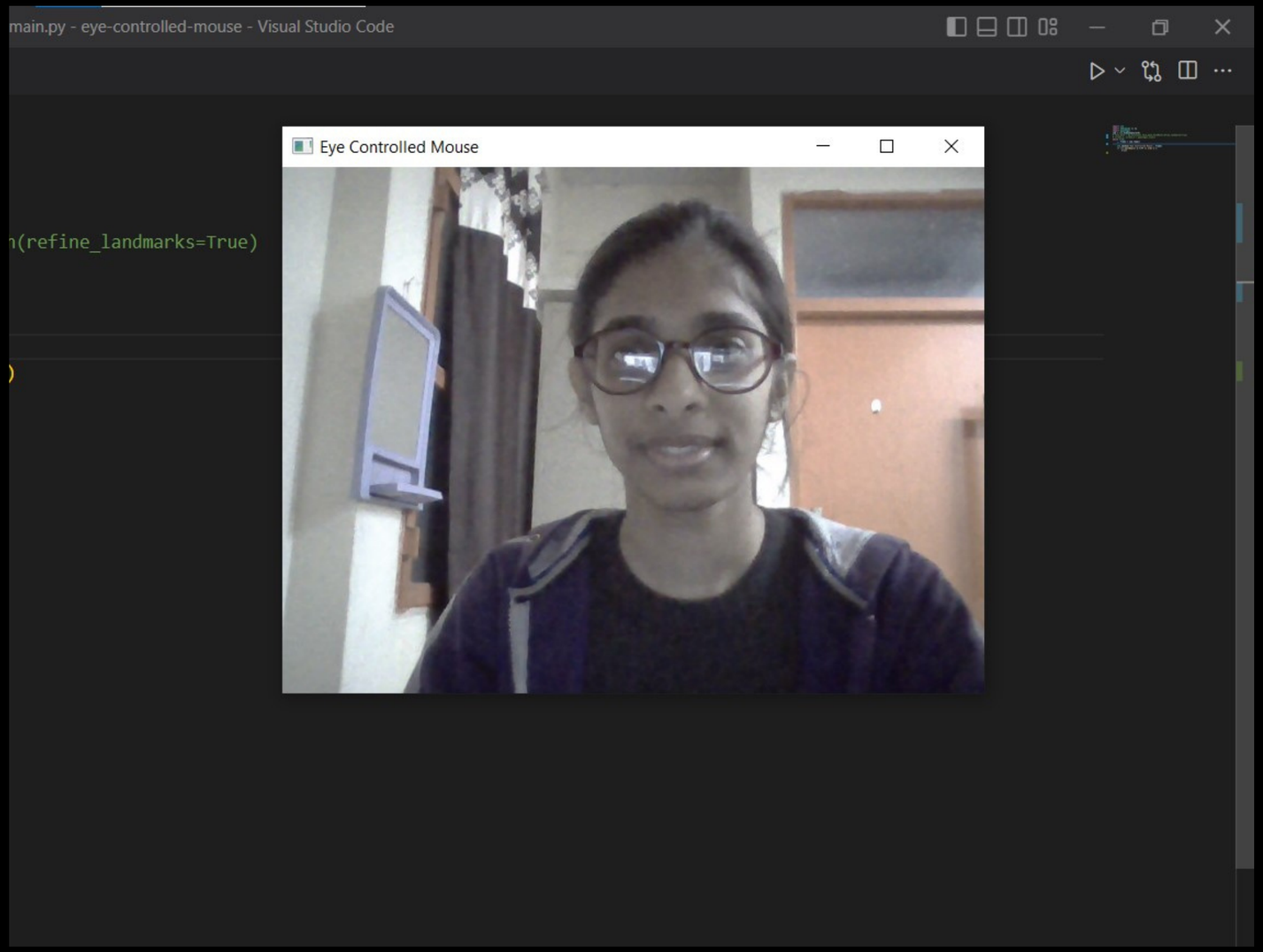
General Structure of the Designed system.



Implementation :

Step 1:
Capturing the Video with WebCam or External Camera

Step 2:
Segmenting the Video into Frames



Step 3:
Detecting the face
And
Noting its Coordinates
(Face Tracking)

```python
import pyautogui
cam = cv2.VideoCapture(0)
face_mesh = mp.solutions.face_mesh.FaceMesh(refine_landmarks=True)
# screen_w, screen_h = pyautogui.size()
while True:
    _, frame = cam.read()
    # frame = cv2.flip(frame, 1)
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    output = face_mesh.process(rgb_frame)
    landmark_points = output.multi_face_landmarks
    print(landmark_points)
    cv2.imshow('Eye Controlled Mouse', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE

```
  z: 0.02057567611336708
}
landmark {
  x: 0.616266131401062
  y: 0.4359799027442932
  z: 0.02057567611336708
}
landmark {
  x: 0.604926586151123
  y: 0.4232890009880066
  z: 0.02057567611336708
}
landmark {
  x: 0.5941579341888428
  y: 0.43802696466445923
  z: 0.02057567611336708
}
landmark {
  x: 0.605589747428894
  y: 0.4503684639930725
  z: 0.02057567611336708
}
]
```

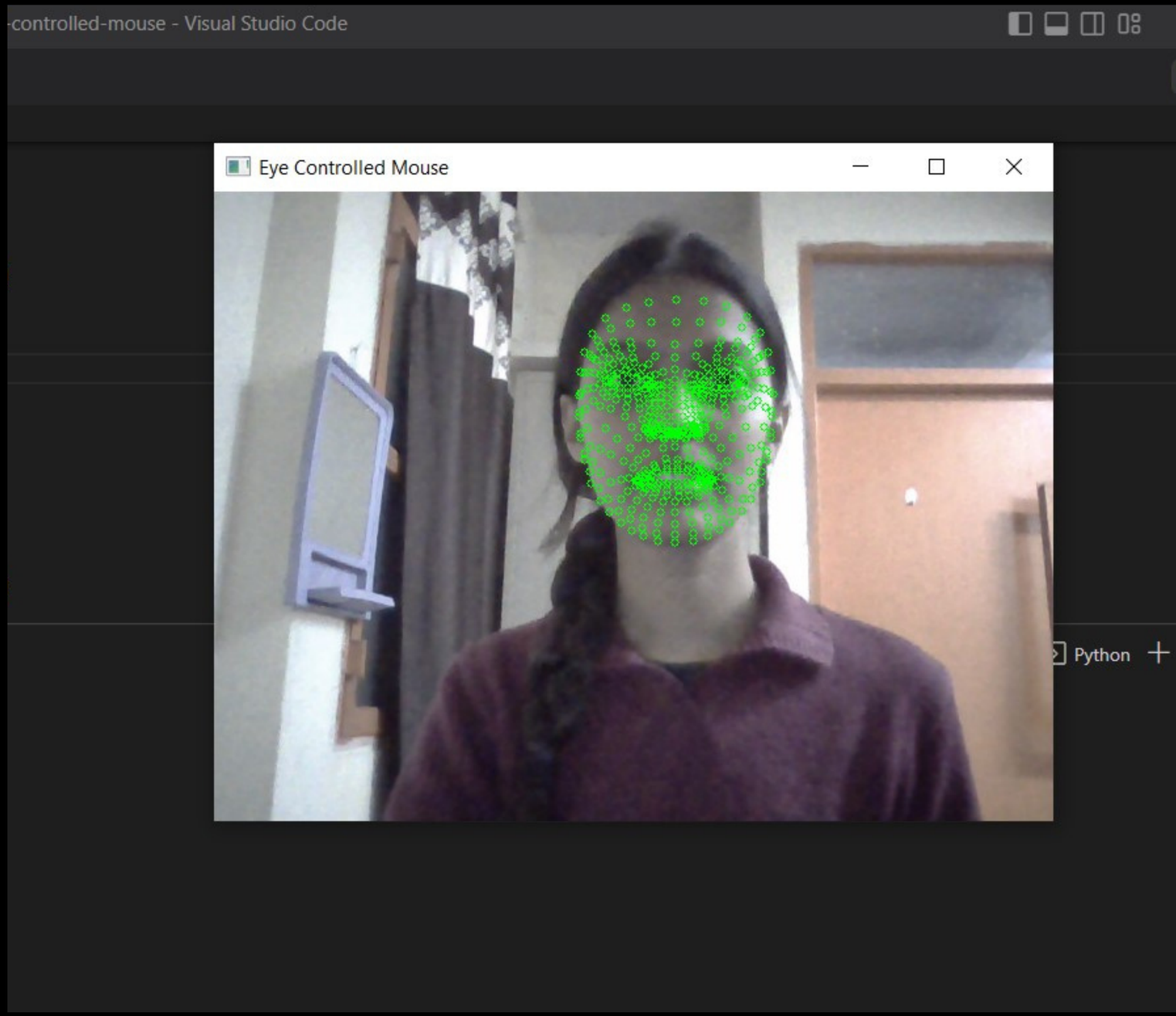
If the face is not detected then no coordinates will be noted i.e None (Unable to Track the face)

```python
import mediapipe as mp
import pyautogui
cam = cv2.VideoCapture(0)
face_mesh = mp.solutions.face_mesh.FaceMesh(refine_landmarks=True)
# screen_w, screen_h = pyautogui.size()
while True:
    _, frame = cam.read()
    # frame = cv2.flip(frame, 1)
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    output = face_mesh.process(rgb_frame)
    landmark_points = output.multi_face_landmarks
    print(landmark_points)
    cv2.imshow('Eye Controlled Mouse', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
```

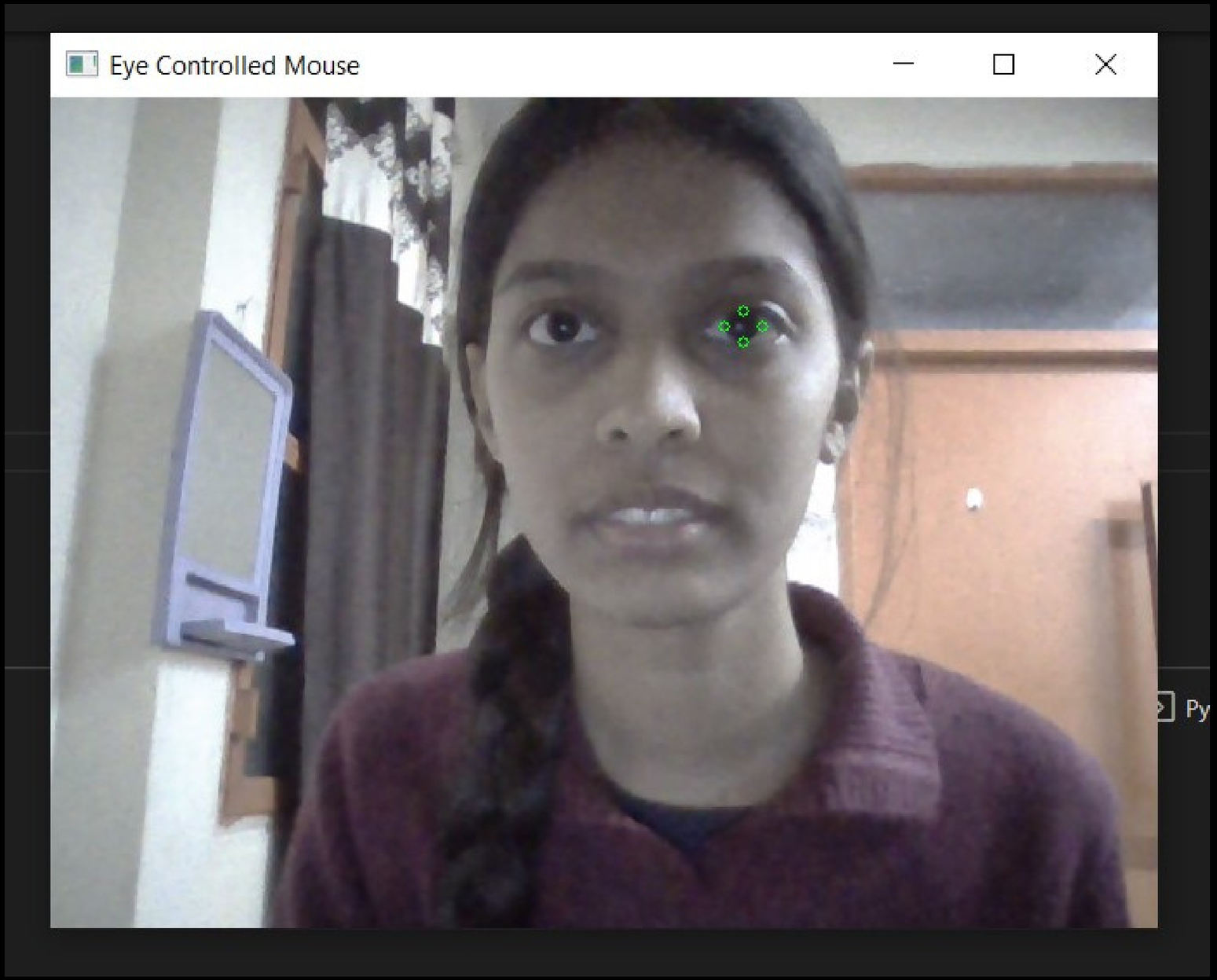
PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE

```
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
None
```

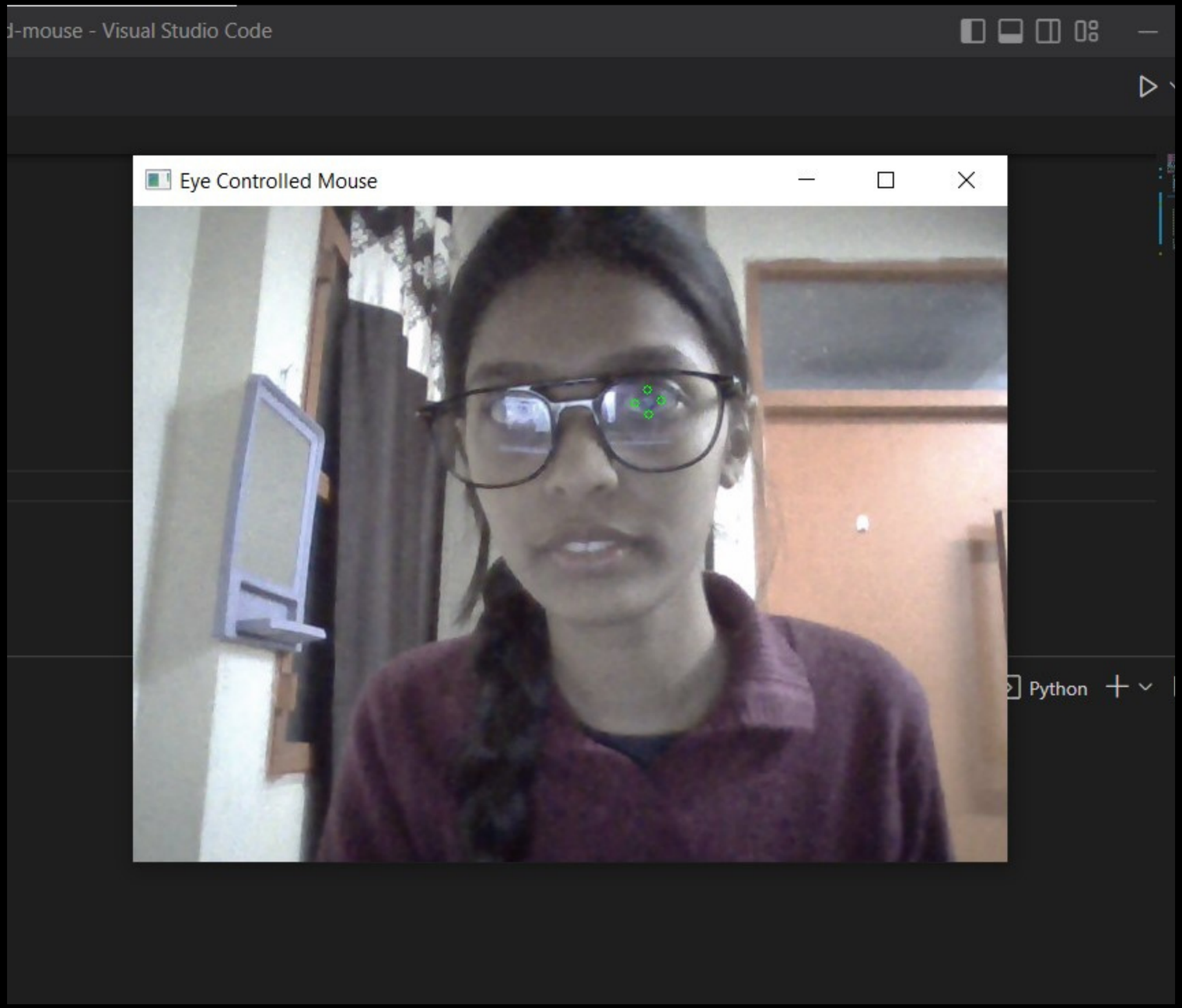
Step 4: Extracting the Facial Landmarks with the help of Mediapipe



Step 5:
Extraction of Eye Landmarks out of Detected Facial Landmarks
i.e.
Eye Detection



Step 5 (Cont...) : Making Eye Detection Feasible when the user is wearing Specs



Step 6: Moving the Cursor According to the pupil movement

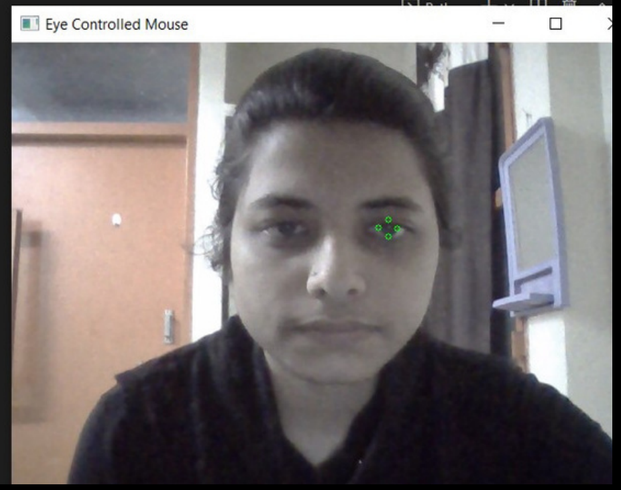
https://drive.google.com/drive/folders/1VWlOXUmh9jTvJlRjf71Kn7UFqLi8XgwT

main.py > ...

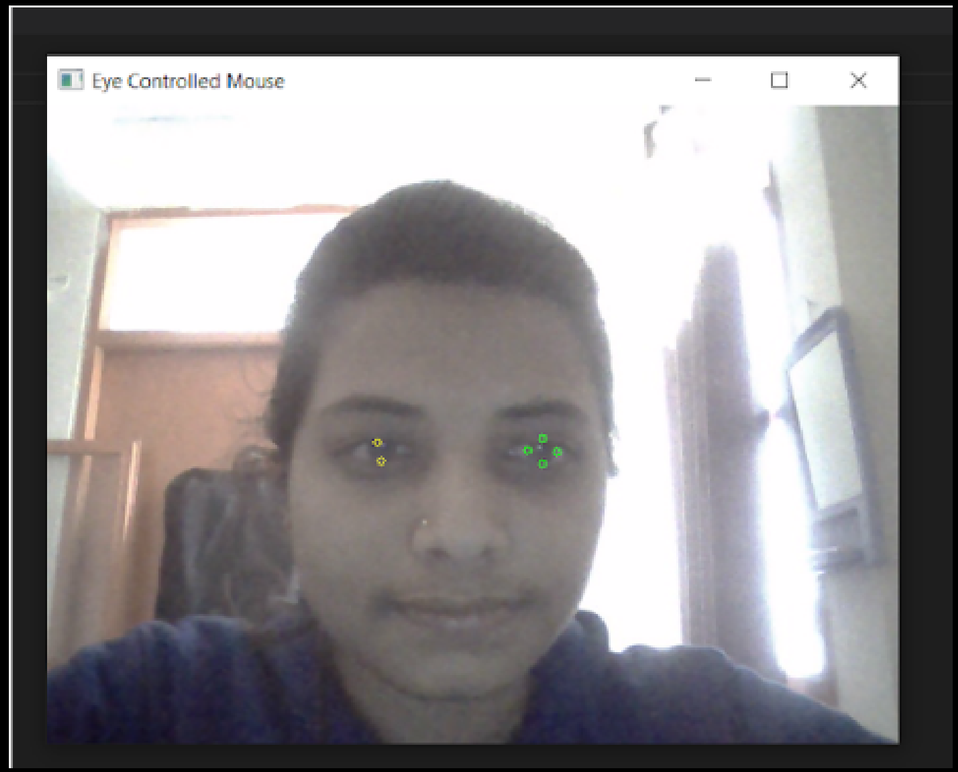
```
11      _, frame = cam.read()
12      frame = cv2.flip(frame, 1)
13      rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
14      output = face_mesh.process(rgb_frame)
15      landmark_points = output.multi_face_landmarks
16      frame_h, frame_w, _ = frame.shape
17
18      if landmark_points:
19          landmarks = landmark_points[0].landmark
20
21          for id, landmark in enumerate(landmarks[474:478]):
22              x = int(landmark.x * frame_w)
23              y = int(landmark.y * frame_h)
24              print(x,y)
25              cv2.circle(frame, (x, y), 3, (0, 255, 0))
```

PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE

```
390 187
380 179
370 187
380 195
390 188
380 179
371 187
380 195
390 187
381 179
370 187
380 195
391 188
381 179
372 187
381 196
390 188
381 179
371 187
381 196
391 188
381 179
```



Step 7:
Extraction of Left Eye Landmarks out of Detected Facial Landmarks for the Click event



Expected Outcomes:

1) To provide a cheap eye-tracking system.

2) To control the cursor of a computer with eyes.

3) Allow physically disabled people to use computers efficiently.

4) To provide a real-time accurate eyes gesture mouse control system.

Continued:

5) To provide a hand free mouse control system.

6) To provide a complete wire free mouse control system.

7) Easy to control cursor movement of a mouse.

- https://code.visualstudio.com/
- https://www.w3schools.com/python/
- https://opencv.org/
- https://mediapipe.dev/
- https://pyautogui.readthedocs.io/en/latest/

Thank You