21BCE2817
DIXIT RAJ

Existing Solution –

Eye Tracking Based Implementation –

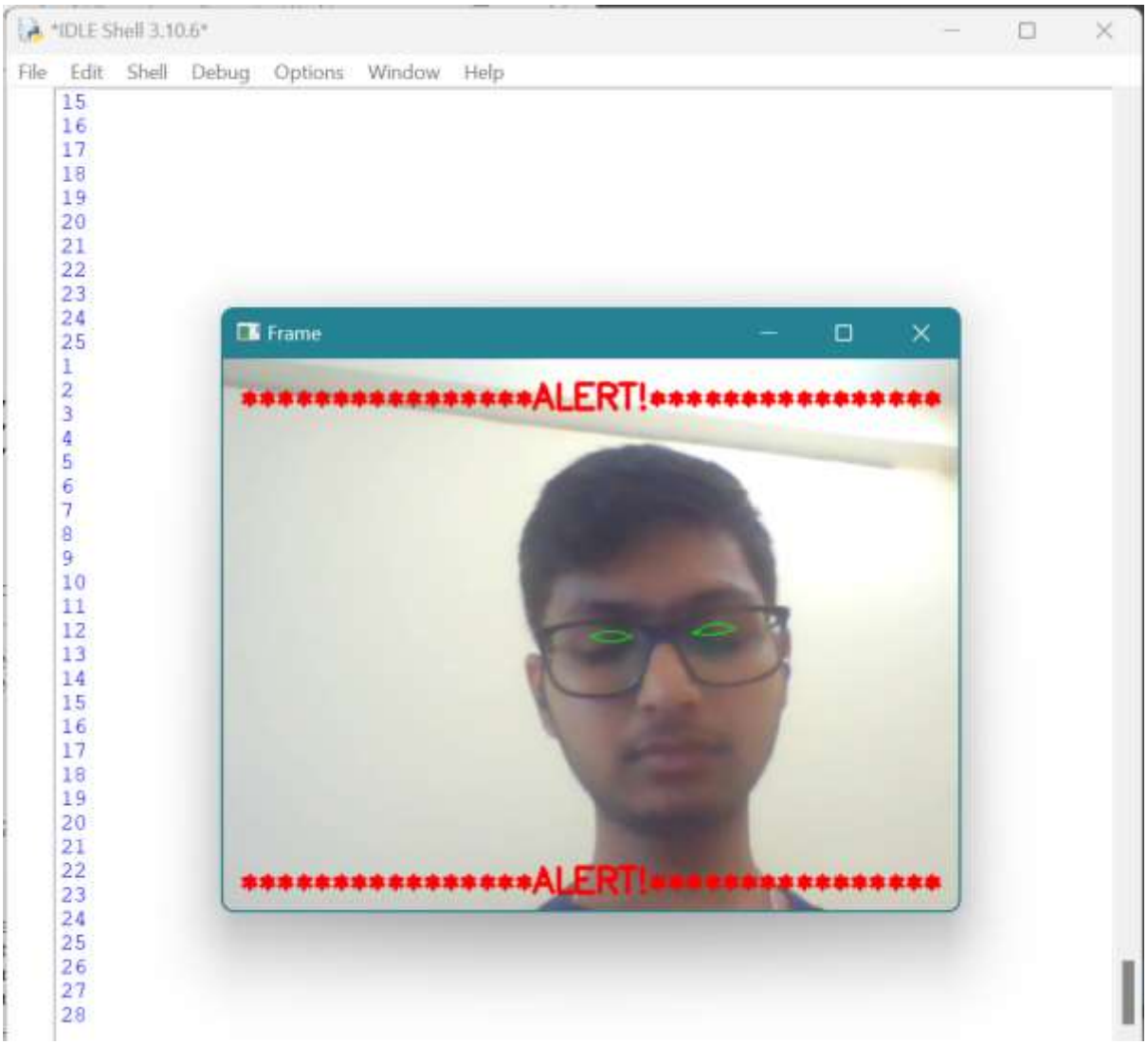
1. **Sleep Detection**

Code –

```
from scipy.spatial import distance
from imutils import face_utils
from pygame import mixer
import imutils
import dlib
import cv2


mixer.init()
        mixer.music.load("music.wav")

        def eye_aspect_ratio(eye):
        A = distance.euclidean(eye[1], eye[5])
        B = distance.euclidean(eye[2], eye[4])
        C = distance.euclidean(eye[0], eye[3])
        ear = (A + B) / (2.0 * C)
        return ear

        thresh = 0.25
        frame_check = 20
        detect = dlib.get_frontal_face_detector()
        predict = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

        (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
        (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
        cap=cv2.VideoCapture(0)
        flag=0
        while True:
        ret, frame=cap.read()
        frame = imutils.resize(frame, width=450)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        subjects = detect(gray, 0)
        for subject in subjects:
        shape = predict(gray, subject)
        shape = face_utils.shape_to_np(shape)
        leftEye = shape[lStart:lEnd]
        rightEye = shape[rStart:rEnd]
        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, (0, 255, 0), 1)
        cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
        if ear < thresh:
        flag += 1
        print (flag)
        if flag >= frame_check:
        cv2.putText(frame, "****************ALERT!****************", (10, 30),
        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        cv2.putText(frame, "****************ALERT!****************", (10,325),
        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        mixer.music.play()
        else:
        flag = 0
```

```
        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xFF
        if key == ord("q"):
        break
        cv2.destroyAllWindows()
        cap.release()
```

Output –



Dataset Used – shape_predictor_68_face_landmarks.data

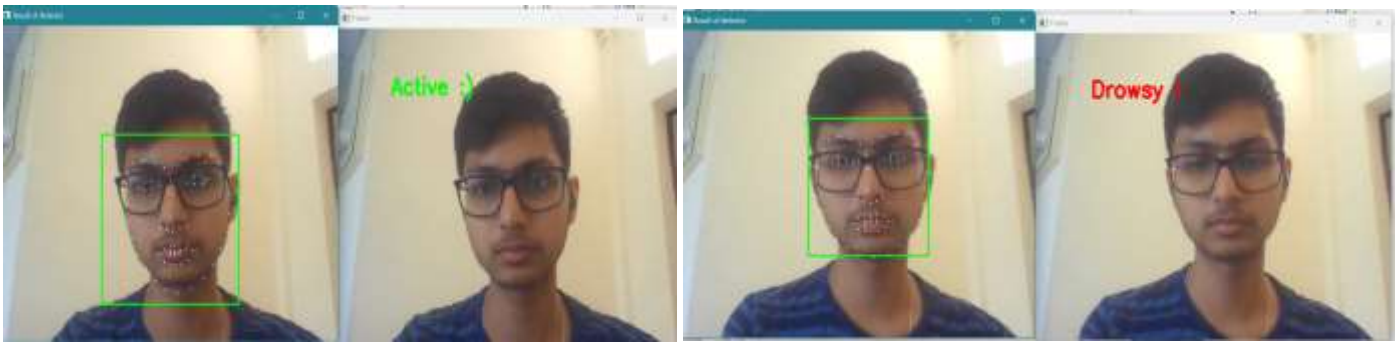
2. Sleep, Drowsy and Active state Detection

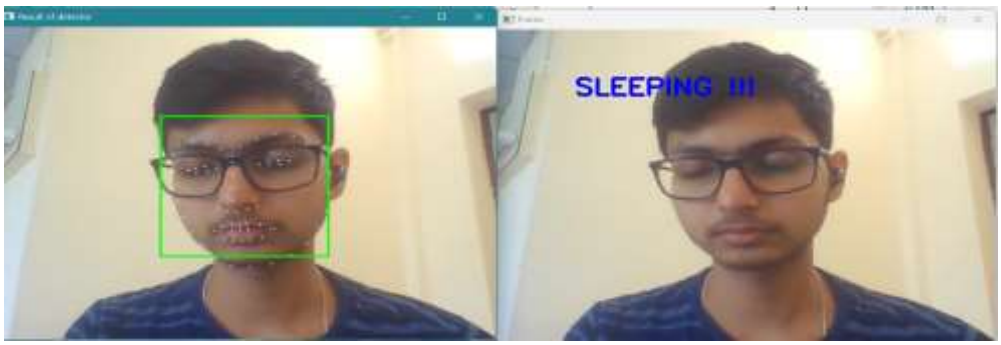
Code –

```
import cv2
import numpy as np
import dlib
from imutils import face_utils
cap = cv2.VideoCapture(0)
        detector = dlib.get_frontal_face_detector()
        predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
        sleep = 0
        drowsy = 0
        active = 0
        status=""
        color=(0,0,0)

        def compute(ptA,ptB):
        dist = np.linalg.norm(ptA - ptB)
        return dist

        def blinked(a,b,c,d,e,f):
        up = compute(b,d) + compute(c,e)
        down = compute(a,f)
        ratio = up/(2.0*down)

        #Checking if it is blinked
        if(ratio>0.25):
        return 2
        elif(ratio>0.21 and ratio<=0.25):
        return 1
        else:
        return 0
        while True:
        _, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        faces = detector(gray)
        #detected face in faces array
        for face in faces:
        x1 = face.left()
        y1 = face.top()
        x2 = face.right()
        y2 = face.bottom()

        face_frame = frame.copy()
        cv2.rectangle(face_frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

        landmarks = predictor(gray, face)
        landmarks = face_utils.shape_to_np(landmarks)

        #The numbers are actually the landmarks which will show eye
        left_blink = blinked(landmarks[36],landmarks[37],
        landmarks[38], landmarks[41], landmarks[40], landmarks[39])
        right_blink = blinked(landmarks[42],landmarks[43],
        landmarks[44], landmarks[47], landmarks[46], landmarks[45])

        #Now judge what to do for the eye blinks
        if(left_blink==0 or right_blink==0):
        sleep+=1
        drowsy=0
        active=0
        if(sleep>6):
        status="SLEEPING !!!"
        color = (255,0,0)
```

```
        elif(left_blink==1 or right_blink==1):
        sleep=0
        active=0
        drowsy+=1
        if(drowsy>10):
        status="Drowsy !"
        color = (0,0,255)

        else:
        drowsy=0
        sleep=0
        active+=1
        if(active>10):
        status="Active :)"
        color = (0,255,0)

        cv2.putText(frame, status, (100,100), cv2.FONT_HERSHEY_SIMPLEX, 1.2, color,3)

        for n in range(0, 68):
        (x,y) = landmarks[n]
        cv2.circle(face_frame, (x, y), 1, (255, 255, 255), -1)

        cv2.imshow("Frame", frame)
        cv2.imshow("Result of detector", face_frame)
        key = cv2.waitKey(1)
        if key == 27:
        break
```

Output –





Dataset Used – shape_predictor_68_face_landmarks.data