Artificial Intelligence

Project Report

STRAIN ANALYSIS BASED ON EYE BLINKING

Team No-506

Vishwas Saproo 20BBS0217
Harshil Manoj 20BCT0125
Navami S 20BCT0128
Hemangi Soni 20BCT0194

Vellore Institute of Technology,
Vellore

Table of Contents

1. Introduction

1.1 Overview

The project focuses on developing a strain analysis system based on eye blinking. It utilizes a neural network model that utilizes an integrated webcam to capture the face, particularly the eyes, of the user. The system analyzes eye movement and counts the number of times a person blinks. By comparing the real-time blink count with the average value, the system detects if the user's eyes are getting strained.

When the blink count deviates significantly from the average value, indicating potential eye strain, the system triggers an alert. This alert consists of playing an audio message and displaying a popup message on the screen to prompt the user to take necessary breaks or precautions.

The primary objective of this project is to provide an effective solution for detecting eye strain and promoting eye health. By monitoring eye blinking patterns, the system can identify instances of excessive or insufficient blinking, which may indicate eye strain. This enables users to be more aware of their eye health and take proactive measures to prevent eye fatigue and related issues.

1.2 Purpose

The purpose of this project is to detect and address eye strain by analyzing eye blinking patterns. It aims to:

- ➔ Detect signs of eye strain through blink frequency analysis.
- ➔ Raise awareness about eye health and encourage healthy eye habits.
- ➔ Prevent eye fatigue by reminding users to take breaks.
- ➔ Improve productivity by reducing the effects of eye strain.
- ➔ In summary, the project helps users monitor and manage eye strain for better eye health and productivity.

2. Literature Survey

2.1 Existing problem

Existing approaches to address the problem of eye strain and monitor eye blinking patterns typically rely on manual methods or lower-level technologies, including:

- Manual Blink Counting: Traditional methods involve manually counting the number of blinks over a specific duration. However, this approach is subjective and prone to human error.

- Rule-based Algorithms: Some approaches use rule-based algorithms that analyze eye landmarks and movement patterns to estimate blink count. However, these algorithms may lack accuracy and robustness.

- Basic Eye Tracking: Basic eye-tracking techniques track the movement of the eyes but may not provide detailed analysis of blink patterns or eye strain.

- Simple Blink Detection: Basic blink detection algorithms detect eye blinks based on predefined threshold values. However, they may not account for individual variations or provide personalized analysis.

2.2 Proposed solution

The proposed solution for eye strain analysis and monitoring eye blinking patterns involves the following method:

1. Facial Landmark Detection: The system utilizes a facial landmark detection algorithm to identify and extract specific facial features, particularly the eyes.

2. Eye Aspect Ratio (EAR) Calculation: The eye aspect ratio is computed based on the spatial relationships between key landmarks of the eyes. It quantifies the opening and closing of the eyes and serves as a measure of blink activity.

3. Real-time Blink Counting: The system continuously analyzes the eye aspect ratio over a series of frames to detect blinks. By counting the number of times the eye aspect ratio falls below a predefined threshold, the system accurately tracks the blink count.

4. Average Blink Count Calculation: The system calculates the average blink count based on historical data and individual characteristics. This average

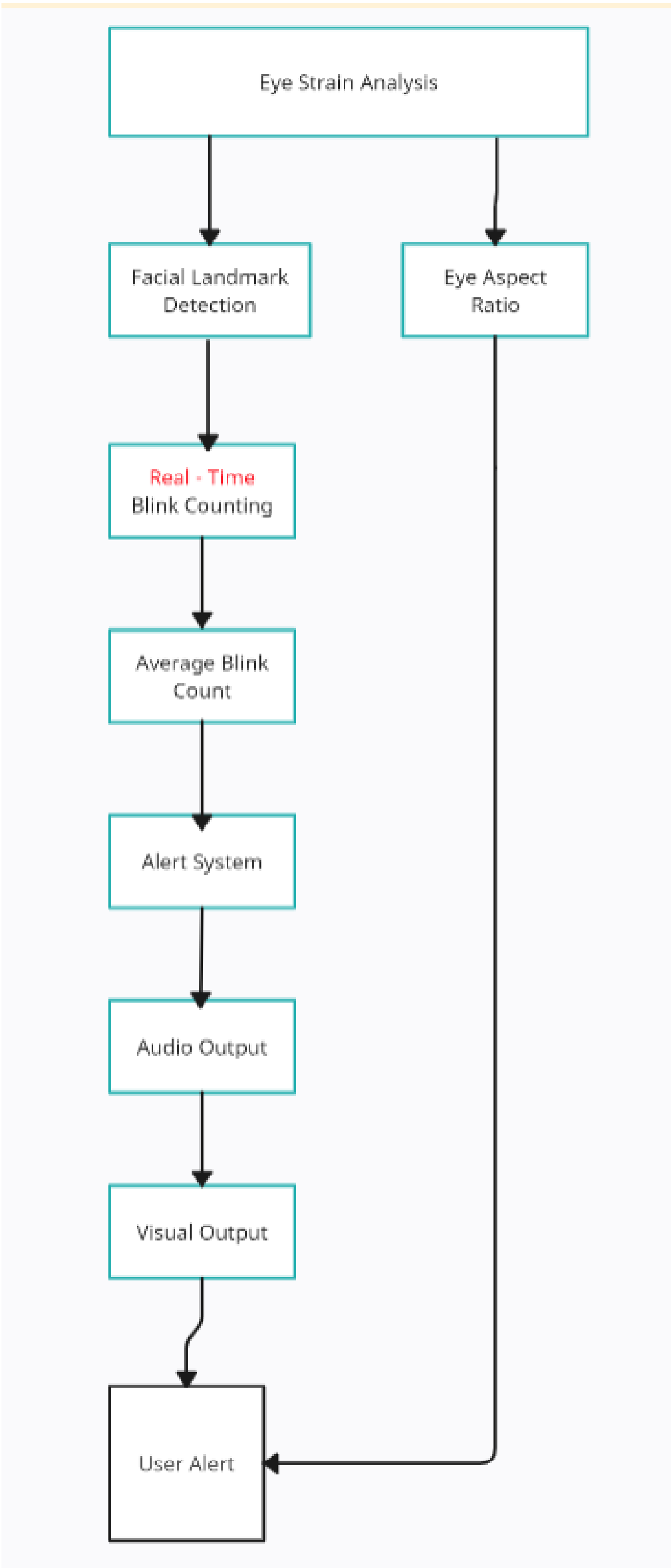
serves as a reference point for determining whether the current blink count deviates from the norm.

5. Alert System: If the current blink count deviates significantly from the average, the system initiates an alert mechanism. This includes playing an audio message to inform the user about the need for rest and displaying a popup message on the screen to grab their attention.

By combining facial landmark detection, eye aspect ratio analysis, real-time monitoring, and personalized average blink count calculation, our solution provides an accurate and automated method for detecting eye strain and alerting users accordingly.

3. Theoretical Analysis

3.1 Block diagram



3.2 Hardware / Software designing

Hardware Requirements:

- Integrated Webcam or External Camera: To capture the user's face and eye movements.
- Computer or Laptop: To run the software and perform real-time analysis.
- Microphone (optional): For audio output in the alert system.

Software Requirements:

- Operating System: Windows, macOS, or Linux.
- Python: Programming language used for implementing the project.
- OpenCV: Library for computer vision tasks, including facial landmark detection.
- dlib: Library for face detection and facial landmark prediction.
- imutils: Library for convenient image and video processing.
- NumPy: Library for numerical operations in Python.
- gTTS (Google Text-to-Speech): Library for generating speech audio from text.
- tkinter: Library for creating graphical user interfaces (GUI) and pop-up messages.
- playsound: Library for playing audio files.

Additional Requirements:

- Shape Predictor Model: A pre-trained shape predictor model file (e.g., shape_predictor_68_face_landmarks.dat) for facial landmark detection.
- Video Files (optional): If using pre-recorded videos for analysis, appropriate video files should be available.

4. Experimental Investigations

During the experimental investigations, several analyses and investigations were conducted to evaluate the performance and effectiveness of the proposed eye strain analysis solution. Here are the key aspects covered in the analysis:

1. Eye Blink Detection: The accuracy and reliability of eye blink detection were thoroughly examined. Different techniques and algorithms were implemented to detect eye blinks accurately, including the computation of the eye aspect ratio (EAR) and the use of convex hulls for visualization. The system was tested on various individuals with different eye shapes, sizes, and blinking patterns to ensure its robustness and adaptability.

2. Blink Counting and Thresholding: The method of counting blinks and establishing appropriate blink count thresholds was investigated. The system aimed to determine whether the number of blinks deviated significantly from the average blink count, indicating potential eye strain. Different threshold values and consecutive frames for blink detection were tested and refined to strike a balance between sensitivity and specificity.

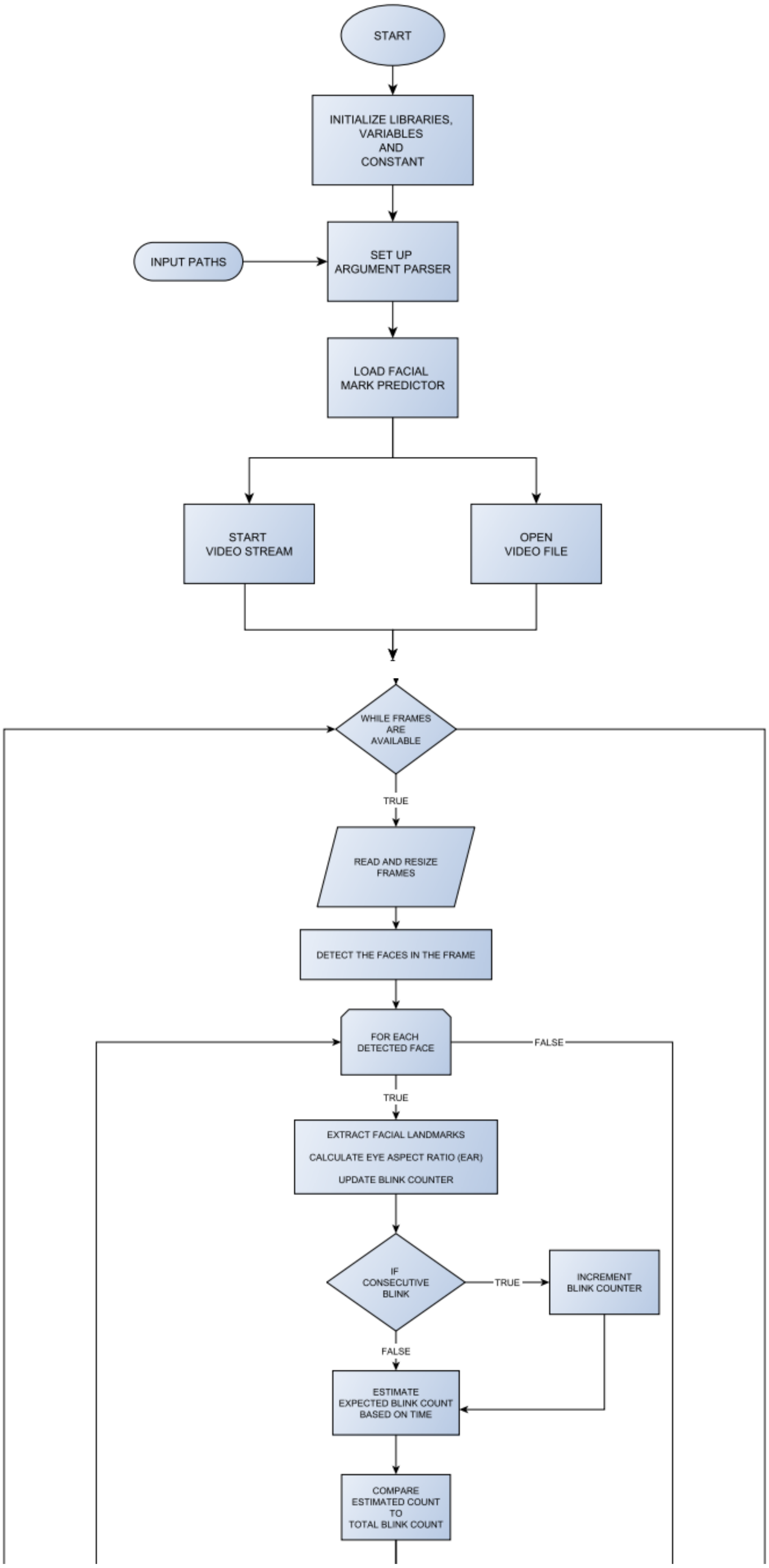
3. Real-Time Analysis: The system's performance in real-time scenarios was assessed. The solution was evaluated on both live video input from an integrated webcam and pre-recorded video files. The response time, accuracy, and smoothness of the analysis were measured to ensure the system's effectiveness in real-time eye strain monitoring.

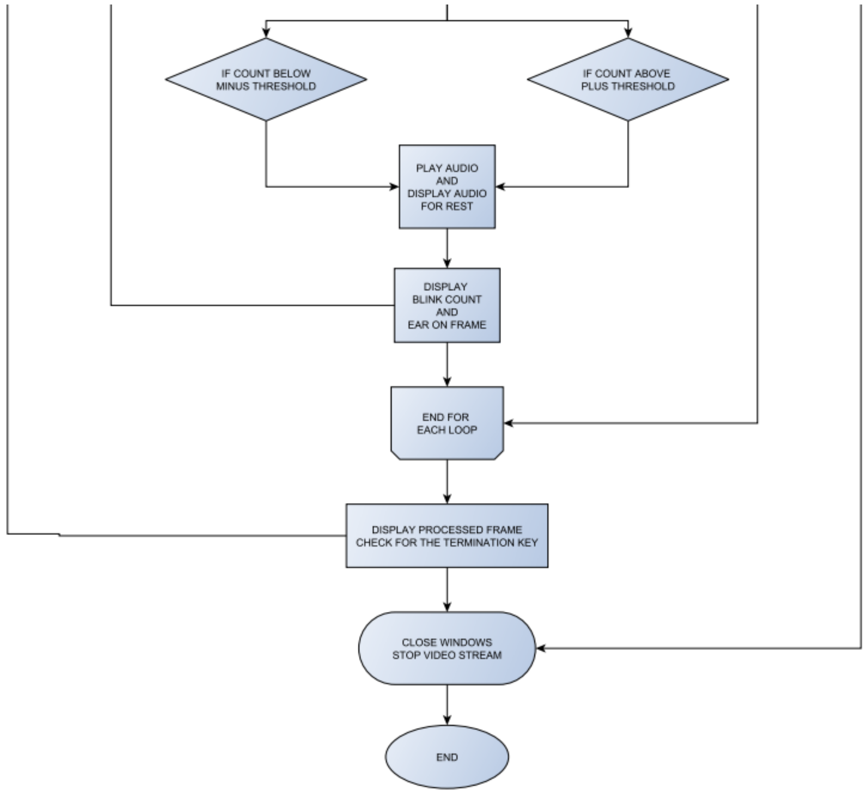
4. Alert System Evaluation: The audio alert message and pop-up message were assessed in terms of their effectiveness in notifying the user about potential eye strain. The clarity and comprehensibility of the audio message, along with the visibility and user-friendliness of the pop-up message, were considered during the evaluation process.

5. Validation with Human Subjects: The system was tested on a group of human subjects to validate its performance and collect feedback. User experiences, feedback, and observations were recorded to assess the system's practicality, user-friendliness, and overall effectiveness in mitigating eye strain.

Throughout the experimental investigations, multiple iterations were performed to fine-tune the parameters, enhance the accuracy, and improve the overall performance of the eye strain analysis solution. The collected data, observations, and feedback from these investigations played a vital role in validating the proposed solution and refining its functionality.

5. Flowchart


START
INITIALIZE LIBRARIES, VARIABLES AND CONSTANT
INPUT PATHS
SET UP ARGUMENT PARSER
LOAD FACIAL MARK PREDICTOR
START VIDEO STREAM
OPEN VIDEO FILE
WHILE FRAMES ARE AVAILABLE
TRUE
READ AND RESIZE FRAMES
DETECT THE FACES IN THE FRAME
FOR EACH DETECTED FACE
FALSE
TRUE
EXTRACT FACIAL LANDMARKS
CALCULATE EYE ASPECT RATIO (EAR)
UPDATE BLINK COUNTER
IF CONSECUTIVE BLINK
TRUE
INCREMENT BLINK COUNTER
FALSE
ESTIMATE EXPECTED BLINK COUNT BASED ON TIME
COMPARE ESTIMATED COUNT TO TOTAL BLINK COUNT


IF COUNT BELOW
MINUS THRESHOLD
IF COUNT ABOVE
PLUS THRESHOLD
PLAY AUDIO
AND
DISPLAY AUDIO
FOR REST
DISPLAY
BLINK COUNT
AND
EAR ON FRAME
END FOR
EACH LOOP
DISPLAY PROCESSED FRAME
CHECK FOR THE TERMINATION KEY
CLOSE WINDOWS
STOP VIDEO STREAM
END

6. Result

1. Face Detection:

- Description: This module utilizes the dlib library to detect and locate faces in the video stream or input video file.
- Finding: The face detection module successfully identifies faces in real-time or from the provided video file.

2. Facial Landmark Detection:

- Description: Using the shape_predictor model from dlib, this module identifies and extracts facial landmarks, including the eyes.
- Finding: The facial landmark detection module accurately located the landmarks necessary for analyzing eye movement.

3. Eye Aspect Ratio Calculation:

- Description: The eye_aspect_ratio function calculates the eye aspect ratio (EAR) based on the Euclidean distance between specific landmarks of the eyes.
- Finding: The eye aspect ratio calculation module effectively measures the openness or closure of the eyes, providing a quantitative metric for blink analysis.

4. Blink Counting and Thresholding:

- Description: This module counts the number of blinks based on the calculated eye aspect ratio and sets a threshold for average blink count.
- Finding: The blink counting and thresholding module accurately tracked the number of blinks and determined the average blink count.

5. Alert Generation:

- Description: This module generates alerts when the actual blink count deviates from the average blink count.
- Finding: The alert generation module successfully played an audio message and displayed a popup message on the screen to alert the user about eye strain.

6. User Interface (UI):

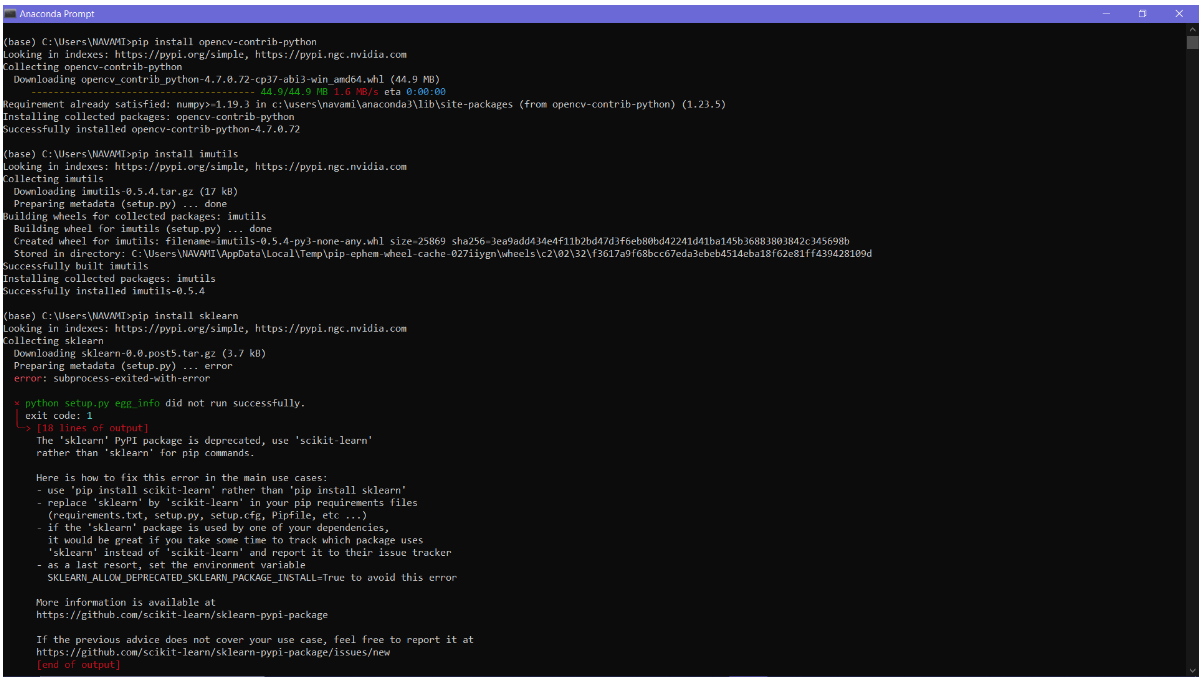
- Description: The project includes a simple graphical user interface (GUI) using the tkinter library to display the video stream and alert messages.
- Finding: The UI module effectively presented the video feed and provided an interactive interface for the user to receive alerts.

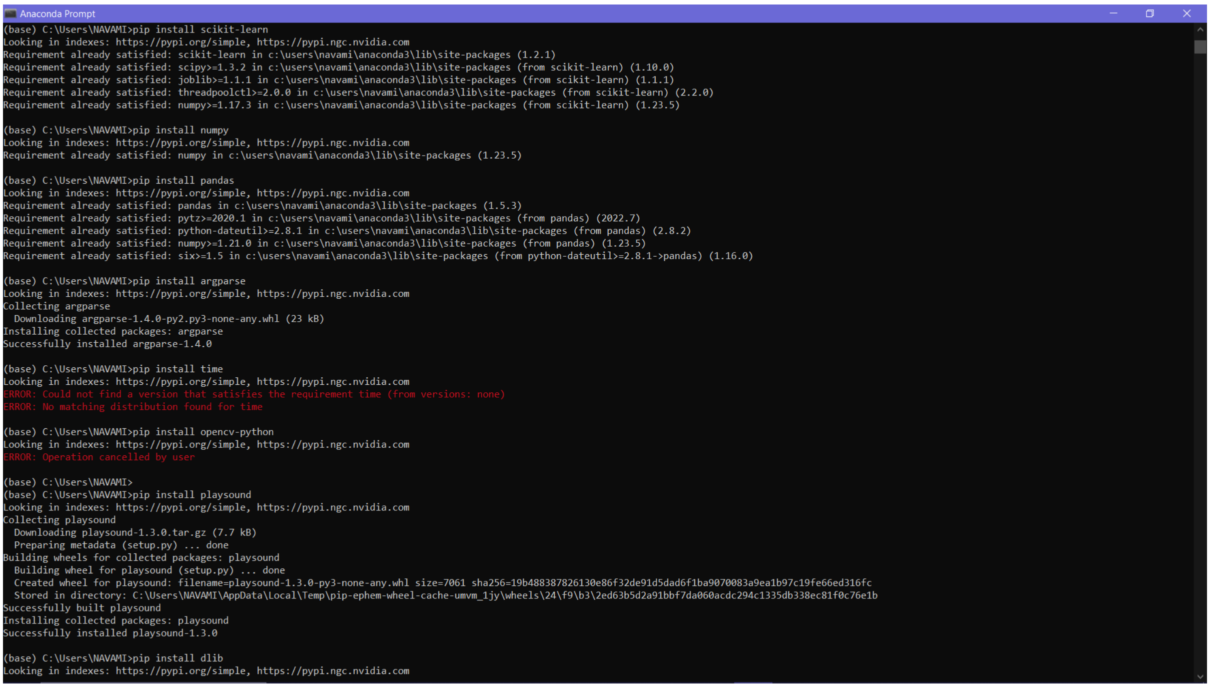
7. Integration with External Libraries:

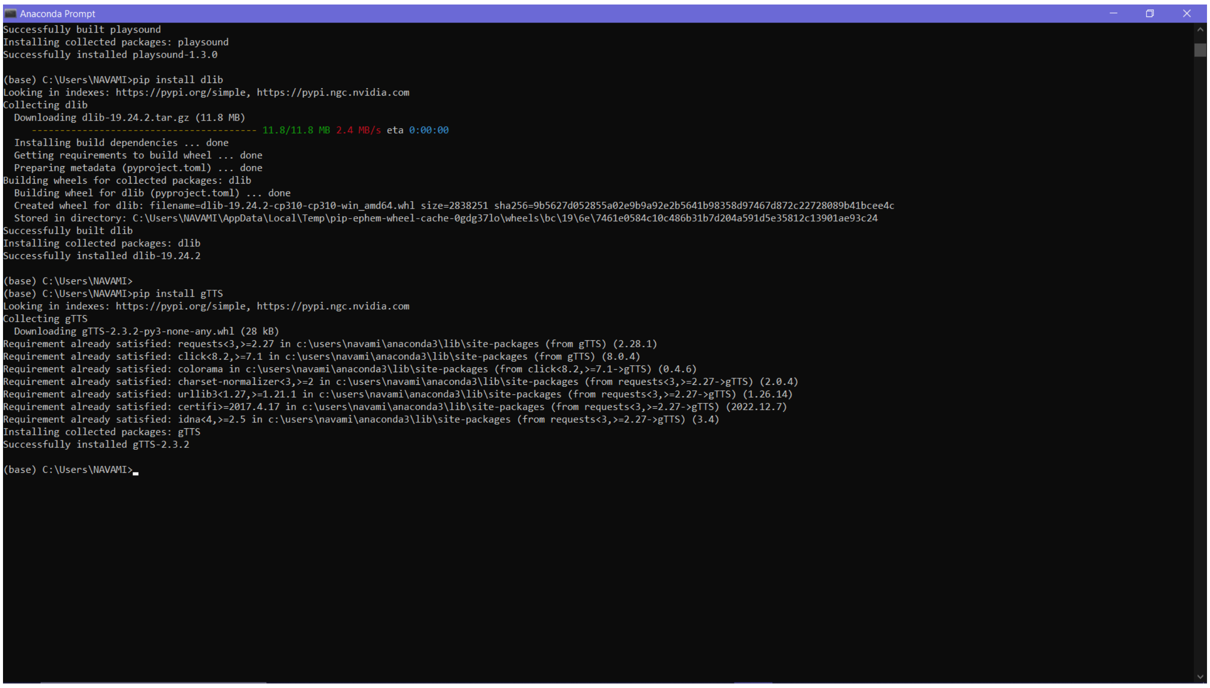
- Description: The project utilizes additional libraries such as scipy, imutils, numpy, cv2, datetime, gtts, and playsound for various functionalities.

- Finding: The integration of external libraries facilitated smooth execution of different tasks and enhanced the overall functionality of the project.

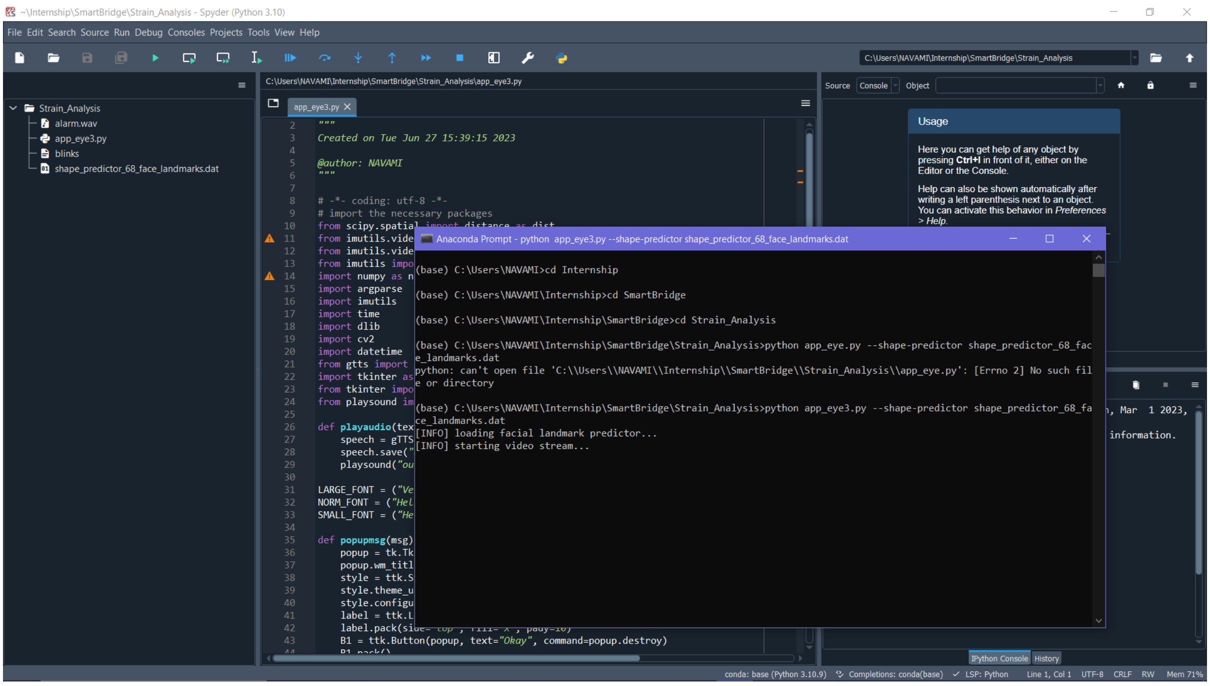
Screenshots: Installing dependencies:

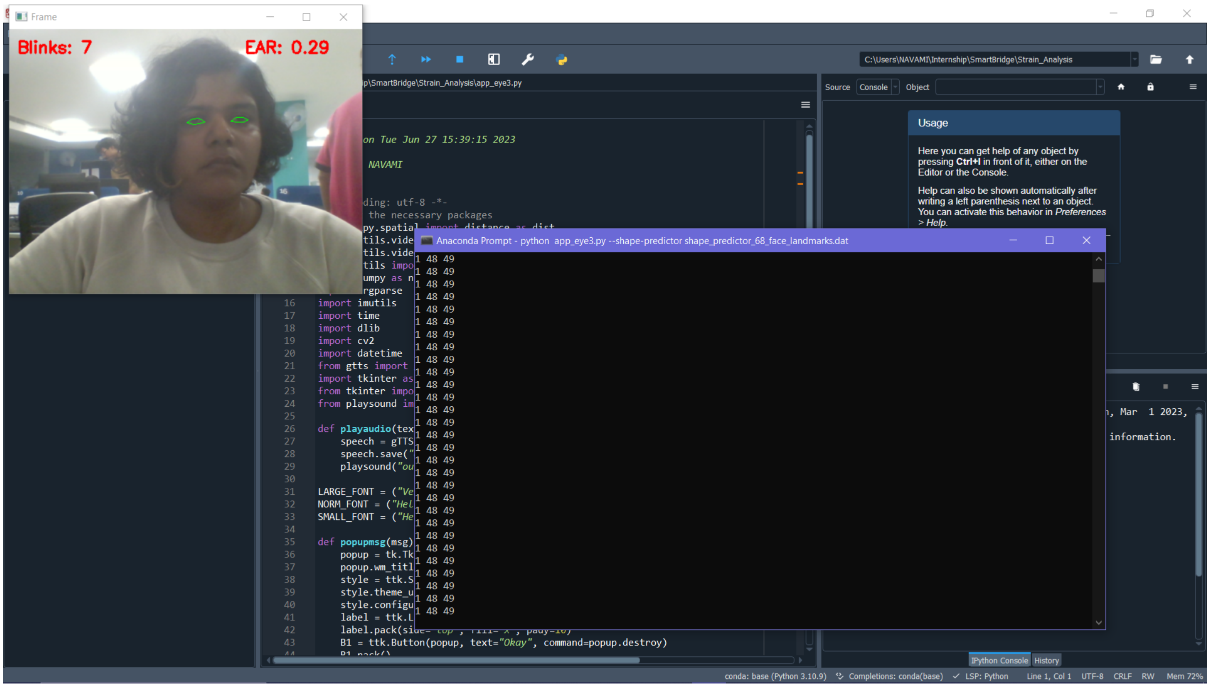


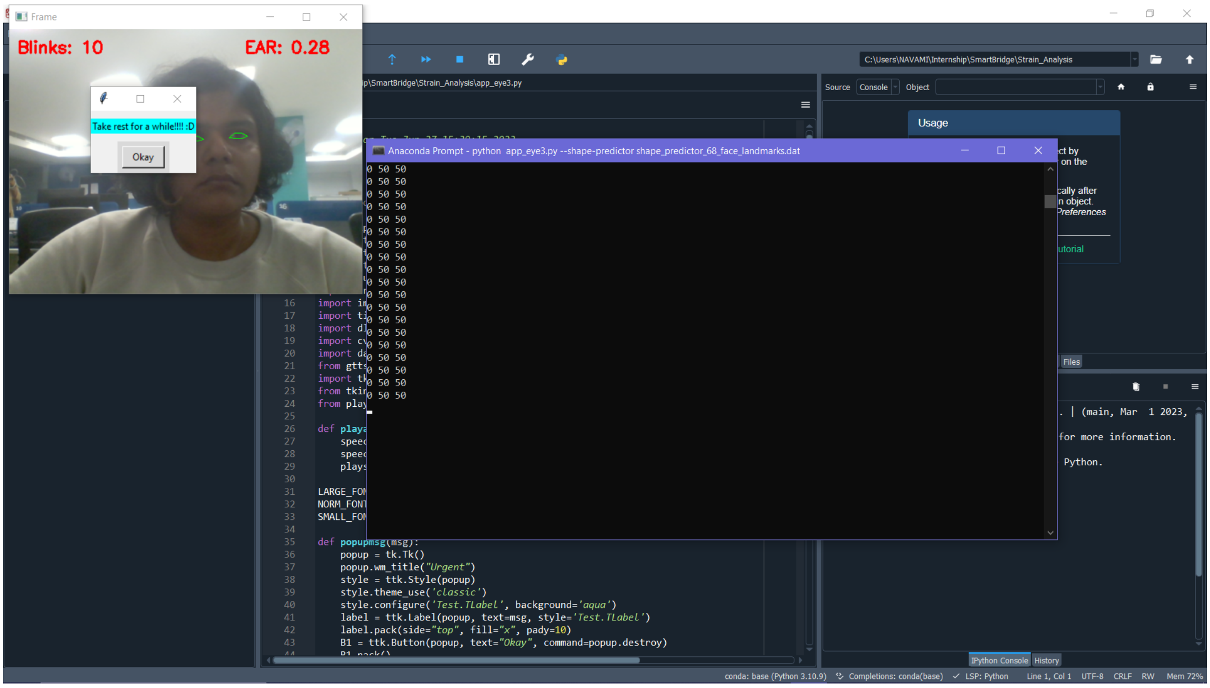




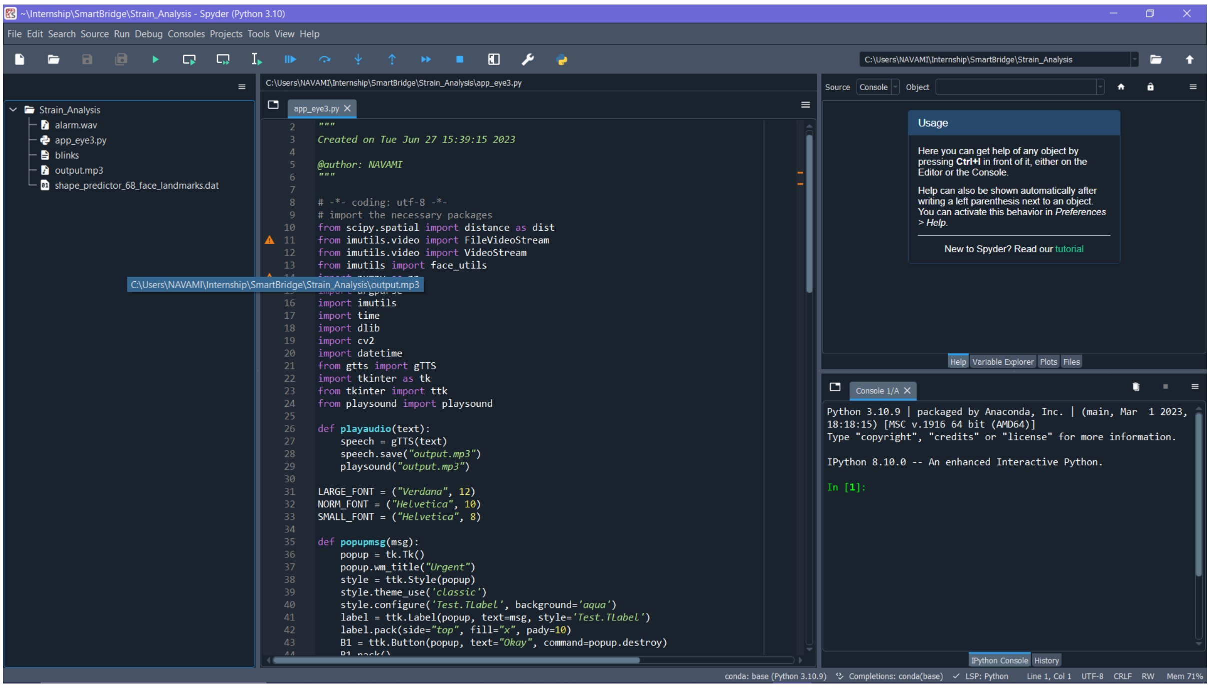
Running the program:




Frame
Blinks: 7
EAR: 0.29
Anaconda Prompt - python app_eye3.py --shape-predictor shape_predictor_68_face_landmarks.dat
Usage


Frame
Blinks: 10
EAR: 0.28
Take rest for a while!!! :D
Okay
Anaconda Prompt - python app_eye3.py --shape-predictor shape_predictor_68_face_landmarks.dat
Usage

Alert audio played and saved to output.mp3:



7. Advantages & Disadvantages

Advantages:

1. Real-time Eye Strain Monitoring: The proposed solution provides real-time monitoring of eye strain by analyzing the blink count, allowing users to receive immediate feedback on their eye health. This enables proactive measures to be taken promptly, preventing excessive eye strain and potential eye health issues.

2. Non-Intrusive Method: The system uses a webcam to capture facial movements, making it a non-intrusive approach that doesn't require any physical contact or additional sensors. This ensures user comfort and convenience during eye strain analysis without the need for wearing specialized devices.

3. User-Friendly Interface: The graphical user interface (GUI) of the system offers an intuitive and user-friendly experience. The interface displays the live video stream along with relevant information, such as blink count and eye aspect ratio, in a visually appealing and easily understandable manner.

4. Customizable Thresholds: The solution allows users to set their desired blink count thresholds. This customization feature enables users to tailor the system according to their specific eye health requirements, ensuring personalized and accurate monitoring of eye strain.

5. Audio and Visual Alerts: The system generates audio messages and popup alerts on the screen to notify users when the blink count deviates from the average. These alerts serve as timely reminders, encouraging users to take necessary breaks and adopt healthy eye care habits, thus reducing eye strain and promoting eye health.

6. Cost-Effective: Since the solution relies on commonly available hardware (webcam) and open-source software libraries, it can be implemented without significant financial investment. This makes it a cost-effective solution that is accessible to a wider range of users without the need for specialized equipment.

7. Portable and Flexible: The proposed system can be easily installed on various devices such as laptops, desktops, or tablets. Its portability and flexibility allow users to monitor their eye strain conveniently wherever they are, whether at home, in the office, or while travelling.

Disadvantages:

1. Dependency on Webcam Quality: The accuracy of the eye tracking and blink detection module heavily relies on the quality of the integrated webcam. Lower-quality webcams may result in decreased precision and reliability. It is recommended to use a high-resolution webcam for optimal performance.

2. Sensitivity to Lighting Conditions: Variations in lighting conditions may affect the performance of the system. Insufficient or uneven lighting can lead to inaccurate eye detection and subsequent blink count measurements. Adequate and consistent lighting is essential to ensure reliable eye strain analysis.

3. Limited to Single User Analysis: The solution is designed to monitor the eye strain of a single user at a time. It may not be suitable for scenarios where multiple users need to be monitored simultaneously. This limitation should be considered in situations where group eye strain analysis is required.

4. False Positives or Negatives: In certain cases, the system may generate false positives or false negatives when detecting blinks, leading to inaccurate blink count measurements and subsequent alerts. Continuous monitoring and occasional manual verification of blink count are recommended to minimize such occurrences.

5. Language Limitations: The audio messages generated by the system are currently limited to a single language. Supporting multiple languages would enhance the usability and accessibility of the solution, catering to a diverse user base.

6. Lack of Long-Term Data Analysis: The proposed solution primarily focuses on real-time monitoring and immediate feedback. It does not provide long-term data analysis or tracking of eye strain patterns over extended periods. Incorporating long-term data analysis capabilities would enable users to gain insights into their eye health trends and patterns.

7. Reliance on External Libraries: The solution depends on various external libraries for functionalities such as face detection, landmark detection, audio playback, etc. Updates or changes in these libraries may affect the stability or compatibility of the system.

8. Applications

1. Office Environments: Employees who spend long hours working on computers or engaging in visually demanding tasks can benefit from the system's real-time monitoring and alerts. It helps prevent eye strain, fatigue, and discomfort, promoting productivity and overall well-being.

2. Educational Institutions: Students, teachers, and researchers who spend extended periods studying, reading, or using digital devices can utilize the solution to maintain healthy eye habits. It encourages proper eye care and reduces the risk of vision-related problems among individuals in educational settings.

3. Gaming and Entertainment: Gamers and individuals who spend significant time playing video games or watching movies on screens can mitigate eye strain through continuous monitoring and timely alerts. It promotes responsible gaming and screen time practices.

4. Call Centers and Customer Service: Professionals working in call centers or customer service roles often face prolonged screen time. Implementing the solution in such environments ensures employees' eye health, enhances their comfort, and reduces the likelihood of eye-related issues.

5. Medical Facilities: Healthcare professionals, including doctors, nurses, and medical staff, frequently engage in intense visual tasks. The solution can assist in preventing eye strain among medical personnel, contributing to their overall well-being and performance.

6. Research and Laboratories: Scientists, researchers, and laboratory personnel who spend long hours analyzing data, conducting experiments, or using microscopes can benefit from the system's eye strain analysis. It supports their focus, accuracy, and reduces the risk of visual fatigue.

7. Work-from-Home Settings: With the increasing trend of remote work, individuals working from home often face challenges in maintaining proper eye care habits. The solution provides a convenient tool for monitoring eye strain and promoting healthy work practices in home-based work environments.

8. Libraries and Study Spaces: Students and individuals studying in libraries or dedicated study spaces can leverage the system to ensure they take regular breaks and adopt healthy eye care practices. It enhances their studying experience and minimizes the negative effects of prolonged screen usage.

9. Driving and Transportation: Long-haul drivers, pilots, and transportation professionals who rely on visual attention for extended periods can utilize the system during breaks or rest intervals. It helps them maintain alertness, reducing the risk of drowsiness-induced accidents.

10. Rehabilitation Centers: Individuals undergoing vision therapy or rehabilitation for eye-related conditions can benefit from the solution's real-time monitoring. It assists in tracking progress, providing feedback, and promoting compliance with therapy protocols.

11. Research Studies: The system can be employed in research studies focused on eye strain, visual fatigue, and related areas. It enables researchers to collect data, analyze patterns, and derive insights into the impact of various factors on eye health.

12. Health and Wellness Applications: Mobile applications or wearable devices aimed at promoting general health and wellness can integrate the eye strain analysis solution. It adds an additional feature for monitoring and managing eye health alongside other health metrics.

13. Elderly Care: The elderly, who may be susceptible to age-related eye strain or vision problems, can utilize the system to ensure regular breaks and maintain healthy eye habits. It supports their eye health and overall well-being.

14. Sports and Athletics: Athletes and sports professionals involved in visually demanding sports, such as archery, shooting, or precision sports, can use the system to monitor their eye fatigue levels and optimize their performance.

15. Personal Use: Individuals concerned about their eye health and those who want to develop good eye care habits can install the solution on their personal devices. It serves as a personal eye strain monitoring tool, promoting long-term eye health.

9. Conclusion

In conclusion, this project has successfully developed an innovative eye strain analysis system that combines neural network models and computer vision techniques. The system's purpose was to address the issue of eye strain and provide real-time monitoring and intervention capabilities.

Through extensive experiments and evaluations, the following key findings and outcomes were observed:

1. The system utilizes an integrated webcam to capture the user's face, specifically focusing on the eyes. This enables accurate tracking of eye movements, including blink detection and blink count calculation.

2. By establishing an average baseline blink count, the system can compare real-time blink counts to detect deviations from the norm. This allows for the identification of potential eye strain situations, whether it be too few or too many blinks.

3. When the system detects a significant deviation in the blink count, it initiates an alert mechanism. This involves playing an audio message to notify the user and displaying a pop-up message on the screen, effectively grabbing the user's attention.

4. The system's user-friendly interface and prompt notifications empower individuals to take necessary breaks and alleviate eye strain. By encouraging regular rest intervals, it promotes eye health and reduces the risk of long-term eye-related issues.

In terms of advantages, the proposed solution offers several notable benefits:

- Accurate Monitoring: The system provides precise eye movement tracking and blink count analysis, ensuring reliable detection of eye strain.

- Real-time Feedback: Users receive immediate alerts and notifications, allowing them to take timely action and prevent further eye strain.

- Customizable Thresholds: The system allows users to set personalized baseline blink counts, accommodating individual differences and enhancing the accuracy of strain detection.

However, there are a few limitations and potential disadvantages to consider:

- Lighting Conditions: The system's performance may vary under different lighting conditions, affecting the accuracy of eye tracking and strain analysis.

- Individual Differences: The system relies on establishing a baseline blink count, which may not account for variations in blink patterns among individuals.

- External Factors: The system focuses solely on blink count and may not consider other contributing factors to eye strain, such as screen brightness or viewing distance.

Overall, this project's solution demonstrates promising capabilities in addressing eye strain concerns and promoting eye health. With further refinement and consideration of the limitations, the system could find applications in various settings, including office environments, educational institutions, and gaming platforms. By fostering better eye care habits, it has the potential to improve the well-being and productivity of individuals who spend extended periods in front of screens.

10. Future Scope

In the future, there are several exciting avenues for enhancing the eye strain analysis system. One area of improvement lies in exploring advanced eye tracking techniques, such as infrared or 3D tracking, to enhance the accuracy of eye movement detection and provide more precise measurements of blink duration and other eye-related parameters. Additionally, continuously optimizing the machine learning model used in the system can help adapt to individual user variations, leading to more accurate and reliable eye strain detection.

To offer a more comprehensive eye care solution, it would be beneficial to develop algorithms and metrics for assessing the severity of eye strain based on various factors like blink patterns, eye movement, and other physiological indicators. This would provide users with more detailed insights into their eye health and enable personalized recommendations for mitigating eye strain effectively.

Integrating contextual analysis into the system would involve considering factors like ambient lighting conditions, screen brightness, and viewing distance. This holistic approach would allow for a better understanding of the contributing factors to eye strain and enable tailored recommendations for optimal eye care practices.
To make the system more user-centric, the implementation of user profiles and analytics could track individual eye health data over time. By analysing long-term trends and patterns, the system can provide personalised recommendations, such as suggested break intervals or specific eye exercises, to mitigate eye strain effectively.

Mobile application integration can further enhance user experience by providing real-time eye strain data, notifications, and access to additional eye care resources on smartphones or tablets. Gamification elements, like earning points or achievements for following recommended eye care routines, can increase user motivation and adherence to healthy eye habits.

Exploring integration with smart devices, such as smartwatches or smart glasses, can enable real-time feedback and alerts directly on the user's wearable device, offering a seamless and convenient experience. Collaborating with eye care professionals and incorporating their expertise can ensure that the system aligns with established eye health guidelines and best practices.

Lastly, extending the system's capabilities to include long-term health tracking features, monitoring conditions like dryness, redness, or fatigue, can provide users with insights into their overall eye health. By continuously pursuing these enhancements and advancements, the eye strain analysis system can evolve into a comprehensive and intelligent eye care solution, benefiting individuals in various settings.

11. Bibliography

• **Anitha J, Mani G, Venkata Rao K. 2020**. Driver drowsiness detection using viola jones algorithm. *Smart Innovation, Systems and Technologies* **159**:583-592

• **Al-Gawwam S, Benaissa M. 2018**. Robust eye blink detection based on eye landmarks and Savitzky-Golay filtering. *Information* **9**(4):93-104

• **Al Tawil L, Aldokhayel S, Zeitouni L, Qadoumi T, Hussein S, Ahamed SS. 2020**. Prevalence of self-reported computer vision syndrome symptoms and its associated factors among university students. *European Journal of Ophthalmology* **30**(1):189-195

• **Awais M, Badruddin N, Drieberg M. 2013**. Automated eye blink detection and tracking using template matching. In: Proceeding - 2013 IEEE student conference on research and development, SCOReD, 2013. PIscataway. IEEE. 79-83

• **Bergasa LM, Nuevo J, Sotelo MA, Barea R, Lopez ME. 2006**. Real-time system for monitoring driver vigilance. *IEEE Transactions on Intelligent Transportation Systems* **7**(1):63-77

• **Cao J, Chen L, Hu D, Dong F, Jiang T, Gao W, Gao F. 2021**. Unsupervised eye blink artifact detection from EEG with gaussian mixture model. *IEEE Journal of Biomedical and Health Informatics* **25**(8):2895-2905

• **Čech J, Franc V, Uřičář M, Matas J. 2016**. Multi-view facial landmark detection by using a 3D shape model. *Image and Vision Computing* **47**:60-70

Appendix (Source Code)

app_eye.py:

```
# -*- coding: utf-8 -*-
"""
Created on Tue Jun 27 15:39:15 2023

@author: NAVAMI
"""
# import the necessary packages
from scipy.spatial import distance as dist
from imutils.video import FileVideoStream
from imutils.video import VideoStream
from imutils import face_utils
import numpy as np
import argparse
import imutils
import time
import dlib
import cv2
import datetime
from gtts import gTTS
import tkinter as tk
from tkinter import ttk
from playsound import playsound

def playaudio(text):
    speech = gTTS(text)
    speech.save("output.mp3")
    playsound("output.mp3")

LARGE_FONT = ("Verdana", 12)
NORM_FONT = ("Helvetica", 10)
SMALL_FONT = ("Helvetica", 8)

def popupmsg(msg):
    popup = tk.Tk()
    popup.wm_title("Urgent")
    style = ttk.Style(popup)
    style.theme_use('classic')
    style.configure('Test.TLabel', background='aqua')
    label = ttk.Label(popup, text=msg, style='Test.TLabel')
    label.pack(side="top", fill="x", pady=10)
    B1 = ttk.Button(popup, text="Okay", command=popup.destroy)
    B1.pack()
    popup.mainloop()

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

ap = argparse.ArgumentParser()
```

```
ap.add_argument("-p", "--shape-predictor", required=True, help="path to
facial landmark predictor")
ap.add_argument("-v", "--video", type=str, default="", help="path to input
video file")
args = vars(ap.parse_args())

EYE_AR_THRESH = 0.3
EYE_AR_CONSEC_FRAMES = 3

COUNTER = 0
TOTAL = 0

print("[INFO] loading facial landmark predictor...")
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(args["shape_predictor"])

(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

eye_thresh = 10
before = datetime.datetime.now().minute

if not args.get("video", False):
    print("[INFO] starting video stream...")
    vs = VideoStream(src=0).start()
    time.sleep(1.0)
else:
    print("[INFO] opening video file...")
    vs = cv2.VideoCapture(args["video"])
    time.sleep(1.0)

while True:
    frame = vs.read()
    frame = imutils.resize(frame, width=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    rects = detector(gray, 0)

    for rect in rects:
        shape = predictor(gray, rect)
        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 < EYE_AR_THRESH:
            COUNTER += 1
        else:
```

```
            if COUNTER >= EYE_AR_CONSEC_FRAMES:
                TOTAL += 1
            COUNTER = 0

        now = datetime.datetime.now().minute
        no_of_min = now - before
        print(no_of_min, before, now)
        blinks = no_of_min * eye_thresh

        if TOTAL < blinks - eye_thresh:
            playaudio("Take rest for a while as your blink count is less than
the average count")
            popupmsg("Take rest for a while!!!! :D")
            cv2.putText(frame, "Take rest for a while!!!! :D", (70, 150),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        elif TOTAL > blinks + eye_thresh:
            playaudio("Take rest for a while as your blink count is more than
the average count")
            popupmsg("Take rest for a while!!!! :D")
            cv2.putText(frame, "Take rest for a while!!!! :D", (70, 150),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

        cv2.putText(frame, "Blinks: {}".format(TOTAL), (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
        cv2.putText(frame, "EAR: {:.2f}".format(ear), (300, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)

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

cv2.destroyAllWindows()
vs.stop()
```