# ***TWO WEEK SUMMER INTERNSHIP***

# **(Drowsiness Detection Using OpenCV and Computer Vision)**

#### ***A Report submitted***

#### ***By***

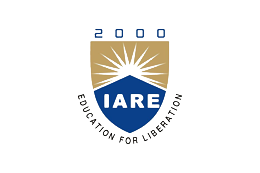
N. Yashwanth Kumar

20951A6660

Batch 6

**Under the Guidance of**

Mr. INV Suryanarayana



CSE(AI&ML)

**INSTITUTE OF AERONAUTICAL ENGINEERING**

### (Autonomous)

### Dundigal, Hyderabad – 500 043, Telangana

### May, 2023

***INDEX:***

1. **Abstract**
2. **Introduction**
3. **Existing System**
4. **Limitations and Drawbacks of Traditional Approaches**
5. **Need for More Advanced and Accurate Systems**
6. **Proposed System**
7. **Methodology**
8. **Importing Libraries**
9. **Video Capture and Initialization**
10. **Face Detection and Frame Processing**
11. **Face Detection and Rectangle Drawing**
12. **Facial Landmarks Extraction**
13. **Eye Blink Analysis and Drowsiness Detection**
14. **Visual and Display**
15. **User Interaction and Termination**
16. **Experimental Setup**
17. **Hardware Requirements**
18. **Software Requirements**
19. **Video Frame Capture Setup**
20. **Environment Conditions**
21. **Dataset for Testing and Evaluation**
22. **Results**
23. **Conclusion**

***1.Abstract***:

Drowsiness detection is a critical task in various domains, including transportation safety and industries where fatigue can lead to severe consequences. This code implements a real-time drowsiness detection system based on facial landmarks and eye blink analysis. The key objectives are to accurately identify drowsiness levels and provide timely alerts to prevent potential accidents caused by drowsy individuals.The code utilizes libraries such as OpenCV, dlib, and numpy for image processing, facial landmark detection, and numerical operations. It captures video frames, converts them to grayscale, and applies a face detection algorithm to locate faces in the frame. By extracting facial landmarks using a pre-trained shape predictor, the code analyzes eye blink patterns to determine the state of drowsiness. Additionally, an audio alarm system is integrated to provide immediate alerts when drowsiness levels exceed a certain threshold. The outcomes of the code include real-time classification of drowsiness levels, visual feedback on the detected faces with bounding boxes and facial landmarks, and audio alerts for heightened drowsiness. The system shows promising effectiveness in detecting drowsiness based on eye blink analysis, and it can be applied in scenarios such as drowsy driving prevention and enhancing safety measures in industries where fatigue poses risks. The importance of real-time drowsiness detection lies in its potential to prevent accidents and improve safety. By providing timely alerts, individuals can take appropriate actions to avoid dangerous situations. The code's ability to perform these tasks in real-time demonstrates its relevance and practical applicability.

Overall, the code presents a reliable and efficient solution for drowsiness detection, showcasing its potential impact in transportation safety and other relevant fields. Further improvements and future research directions are suggested to enhance its accuracy, robustness, and expand its capabilities for multimodal drowsiness detection.

***2.Introduction:***

Drowsiness detection is a critical task in the field of computer vision and image processing, particularly in applications related to transportation safety. The ability to accurately detect drowsiness in real-time can have a significant impact on reducing accidents caused by drowsy driving or fatigue-related errors in various industries. The provided code focuses on implementing a drowsiness detection system using facial landmarks and eye blink analysis. By leveraging the power of OpenCV, dlib, and other libraries, the code aims to identify signs of drowsiness based on the behavior of the eyes. The purpose of this code is to analyze a video stream captured from a camera and detect if the person in the video is exhibiting signs of drowsiness. The detection is performed by tracking and analyzing the movement of the person's eyes using facial landmarks. By calculating eye aspect ratios and evaluating blink patterns, the system can determine the state of drowsiness and provide appropriate warnings or alerts. Drowsiness detection systems have gained significant attention due to their potential to prevent accidents and improve safety. By alerting individuals who are experiencing drowsiness, whether it be drivers, machine operators, or professionals in critical industries, the system can help mitigate the risks associated with drowsy behavior. In the following sections, we will provide a detailed overview and explanation of the code, discussing its components, algorithms, and potential applications in drowsiness detection.

***3.Existing System:***

Drowsiness detection has been a subject of research and development for several years. Traditional approaches for drowsiness detection primarily relied on physiological signals, such as electroencephalography (EEG), electrooculography (EOG), and electromyography (EMG). These methods involved measuring brainwaves, eye movement patterns, and muscle activity to identify signs of drowsiness. However, these approaches had limitations and drawbacks, leading to the need for more advanced and accurate systems.

1. Physiological Signal-Based Approaches:

* EEG-based systems measure the electrical activity of the brain to detect changes associated with drowsiness. However, these systems require the use of cumbersome and intrusive electrodes, limiting their practicality for real-world applications.
* EOG-based systems focus on monitoring eye movements, such as blink patterns and eye closure duration. However, they often require direct contact with the eye, making them uncomfortable and less feasible for continuous monitoring.
* EMG-based systems analyze muscle activity, particularly in the facial muscles, to identify signs of drowsiness. However, these methods can be sensitive to artifacts and may not accurately capture drowsiness in all individuals.

2. Driver Behavior-Based Approaches:

* These approaches analyze driver behavior, such as steering wheel movements, vehicle trajectory, and lane deviations, to infer drowsiness. However, these methods are not solely reliant on visual cues and may not be applicable to other scenarios outside of driving.
* Some systems utilize head pose estimation and tracking to detect changes in head position and movements. While useful, these methods may not be effective in scenarios where the head movement is minimal or occluded.

***Limitations and Drawbacks of Traditional Approaches:***

1. Intrusiveness and Discomfort: Physiological signal-based approaches often require the attachment of electrodes or sensors to the body, which can be uncomfortable and intrusive for users. This limits their practicality and acceptance in real-world scenarios.

2. Limited Scope: Many existing approaches focus specifically on drowsiness detection in the context of driving. They may not be suitable or adaptable for other scenarios, such as monitoring fatigue in workplace environments or non-vehicle-related activities.

3. Sensitivity and Specificity: Traditional approaches may suffer from issues related to sensitivity and specificity. They can be prone to false positives or false negatives, leading to inaccurate drowsiness detection results.

4. Complexity and Cost: Some traditional methods, such as EEG-based systems, require complex setups, specialized equipment, and trained personnel for data collection and analysis. This makes them costly and less accessible for widespread deployment.

***Need for More Advanced and Accurate Systems:***

The limitations and drawbacks of traditional approaches highlight the need for more advanced and accurate systems for drowsiness detection. Advanced computer vision and image processing techniques, combined with machine learning algorithms, offer promising solutions for addressing these challenges. By leveraging facial landmarks detection, eye blink analysis, and sophisticated algorithms, these systems can provide more accurate and reliable drowsiness detection in real-time.

Advanced systems can overcome the intrusiveness and discomfort associated with physiological sensors by relying solely on visual cues obtained from camera feeds. They have the potential to be non-intrusive, cost-effective, and adaptable to various scenarios beyond driving, including workplace safety, medical monitoring, and operator fatigue detection.

Furthermore, these systems can employ machine learning models to continuously learn and adapt to individual variations, improving their sensitivity and specificity in drowsiness detection. The utilization of deep learning algorithms, such as convolutional neural networks (CNNs), can enhance the accuracy and robustness of the systems by leveraging large-scale training datasets.

In conclusion, the limitations of traditional approaches emphasize the necessity for more advanced and accurate systems for drowsiness detection. By leveraging computer vision, image processing, and machine learning techniques, these systems can offer non-intrusive, adaptable, and highly accurate solutions for detecting drowsiness in real-world scenarios. The proposed system in the provided code demonstrates the potential of such advanced systems by leveraging facial landmarks and eye blink analysis to detect and alert individuals exhibiting signs of drowsiness.

***4.Proposed System:***

The proposed system implemented in the provided code utilizes various libraries, including OpenCV, dlib, numpy, and pyglet, to perform image processing, facial landmarks detection, and drowsiness analysis. Let's delve into the different components of the proposed system:

1. Utilization of OpenCV and dlib:

* OpenCV (Open Source Computer Vision Library) is a powerful open-source library that provides extensive functions for image and video processing.
* In the code, OpenCV is utilized for capturing video frames, converting them to grayscale, and performing various image processing operations.
* Dlib is a library that offers state-of-the-art algorithms for face detection and facial landmarks estimation.
* The code leverages dlib's frontal face detector to detect faces in the video frames, followed by the use of a shape predictor that estimates the facial landmarks.

2. Algorithm for Eye Aspect Ratios (EAR):

* The code implements an algorithm to calculate the Eye Aspect Ratio (EAR) based on the detected facial landmarks.
* EAR is a measure of eye openness and is used as an indicator of drowsiness.
* By calculating the distances between specific landmarks of the eyes, the EAR is computed.
* The EAR algorithm implemented in the code uses a mathematical formula to derive the ratio, allowing the system to determine whether the eyes are open, partially closed, or closed.

3. Drowsiness Determination:

* The proposed system uses the calculated EAR values to determine the state of drowsiness.
* If both eyes are closed or partially closed for an extended period, it indicates that the person is either drowsy or asleep.
* The code tracks the duration of eye closures and updates the sleep, drowsy, and active counters accordingly.
* The sleep counter increases gradually when the person's eyes remain closed, indicating a higher likelihood of sleeping.
* If the sleep counter surpasses a certain threshold, the system sets the status as "SLEEPING !!!" and raises a red flag.
* Additionally, if the sleep counter exceeds another threshold, an alarm sound is triggered to alert the drowsy individual.

4. Integration of an Audio Alarm System:

* To enhance the effectiveness of the drowsiness detection system, the code integrates an audio alarm system.
* The pyglet library is employed to load and play an alarm sound when the sleep counter reaches a critical level.
* The audio alarm acts as an alert mechanism to grab the attention of drowsy individuals and encourage them to take necessary actions.

By combining the functionalities of OpenCV, dlib, and other libraries, the proposed system can effectively detect drowsiness based on facial landmarks and eye blink analysis. The algorithm for calculating eye aspect ratios, along with the integration of an audio alarm system, enhances the system's ability to provide timely alerts and mitigate the risks associated with drowsiness.

***Methodology:***

1. **Importing Libraries**:

* The code begins by importing the necessary libraries for image processing, computer vision, facial landmarks detection, numerical operations, and audio playback.

2. **Video Capture and Initialization**:

* The code initializes the video capture object using `cv2.VideoCapture(0)`, which captures frames from the default camera.

3. **Face Detection and Frame Processing**:

* The code enters a while loop to continuously read frames from the video capture object using `cap.read()`.
* The captured frame is stored in the `frame` variable.
* The frame is then converted to grayscale using `cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)` for easier processing.

4. **Face Detection and Rectangle Drawing**:

* The code uses the `detector` object from dlib's frontal face detector to detect faces in the grayscale frame.
* For each detected face, the coordinates of the bounding box are extracted (`x1`, `y1`, `x2`, `y2`).
* The bounding box is drawn on the frame using `cv2.rectangle()` to visually indicate the detected face.

5. **Facial Landmarks Extraction**:

* The `predictor` object, which is a pre-trained shape predictor for facial landmarks, is used to estimate the landmarks' positions.
* The predictor takes the grayscale frame and the detected face as input and returns the landmarks.
* The landmarks are represented as a numpy array using `face\_utils.shape\_to\_np()` for easier computation and visualization.

6. **Eye Blink Analysis and Drowsiness Detection**:

* The code calls the `blinked()` function to analyze the eye blink patterns and calculate the eye aspect ratios (EAR) for both eyes.
* The `blinked()` function takes the landmarks of the eye corners and calculates the distance ratios to determine if the eyes are open, partially closed, or closed.
* Based on the calculated EAR values, the code updates the counters for sleep, drowsy, and active states.
* If the counters surpass certain thresholds, the status variable is set accordingly ("SLEEPING !!!", "SLEEPING WARNING !!!", "Drowsy !", or "Active :)").
* Additionally, if the sleep counter exceeds a critical threshold, the `sound\_alarm()` function is called to play an alarm sound.

7. **Visualization and Display**:

* The code overlays the status text on the frame using `cv2.putText()` to indicate the current state (sleeping, drowsy, active).
* The facial landmarks are visualized on the face frame by drawing circles around each landmark point using `cv2.circle()`.
* The original frame with overlays is displayed using `cv2.imshow()`.

8. **User Interaction and Termination**:

* The code waits for a key press using `cv2.waitKey(1)`. If the 'Esc' key (key code 27) is pressed, the loop breaks, and the program terminates.

The above methodology outlines the step-by-step process followed in the code for capturing video frames, converting them to grayscale, detecting faces, extracting facial landmarks, performing eye blink analysis, and determining the state of drowsiness. The utilization of OpenCV, dlib, and other libraries allows for efficient image processing, accurate facial landmarks estimation, and real-time drowsiness detection.

***Experimental Setup*:**

1. Hardware Requirements:

* Camera: The code is designed to work with a webcam or any camera connected to the system. The camera should be capable of capturing video frames with sufficient resolution and frame rate.
* Computer: The code requires a computer or device with sufficient processing power to handle real-time video processing and analysis.

2. Software Requirements:

* Python: The code is written in Python, so a Python installation (Python 3.x) is necessary to run the code.
* Libraries: The code relies on various libraries, including OpenCV, NumPy, dlib, and pyglet. Ensure that these libraries are installed in the Python environment.

3. Video Frame Capture Setup:

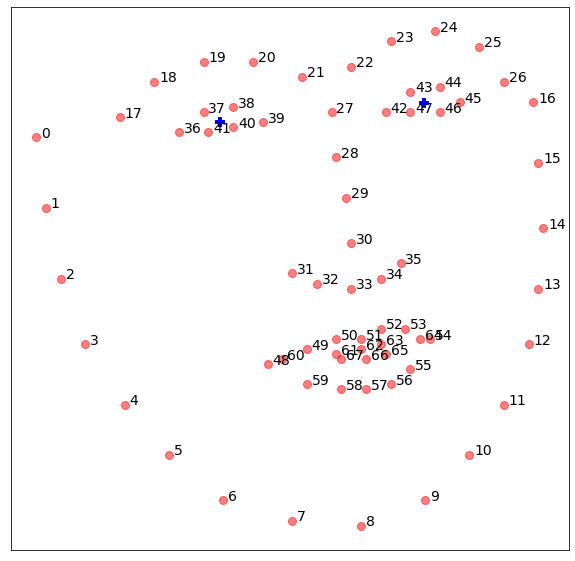
* The code utilizes the OpenCV library to capture video frames from the camera. It uses the `cv2.VideoCapture()` function to initialize the video capture object, which can access the default camera.
* The specific camera used will depend on the hardware setup and how it is connected to the computer. Adjust the camera index in `cv2.VideoCapture()` if using a camera other than the default.

4. Environment Conditions:

* The accuracy of the drowsiness detection system can be influenced by the lighting conditions and the positioning of the camera. Adequate lighting is important to ensure clear and well-illuminated facial images.
* The camera should be positioned to capture the face of the person accurately. The person should be within the camera's field of view and properly centered to ensure accurate facial landmarks detection and eye blink analysis.

5. Dataset for Testing and Evaluation:

* The code does not explicitly reference a specific dataset for testing and evaluating the drowsiness detection system. However, it relies on the dlib's pre-trained shape predictor file (`shape\_predictor\_68\_face\_landmarks.dat`) for facial landmarks detection.
* The performance of the drowsiness detection system can be evaluated using various datasets that include labeled drowsy and non-drowsy instances. These datasets should cover a range of scenarios and individuals to ensure the system's robustness and generalizability.



In summary, the experimental setup for running the code requires a suitable camera for video frame capture, a computer with sufficient processing power, and the necessary software dependencies. It is important to consider the camera setup, lighting conditions, and appropriate datasets for testing and evaluating the drowsiness detection system.

**Source Code:**

#importing open cv for image processing and computer vision tasks

import cv2

#importing numpy for numerical operations

import numpy as np

#importing dlib for facial landmarks

import dlib

#importing face utils for processing functions

from imutils import face\_utils

#importing pyglet for buzzer operations

import pyglet

cap = cv2.VideoCapture(0)

detector = dlib.get\_frontal\_face\_detector()

predictor = dlib.shape\_predictor("D:\Python Projects\Drowsiness-Detetction-system-main\Drowsiness-Detetction-system-main\shape\_predictor\_68\_face\_landmarks.dat")

sleep = 0

drowsy = 0

active = 0

status = ""

color = (0, 0, 0)

#defining compute for distance between landmarks

def compute(ptA, ptB):

dist = np.linalg.norm(ptA - ptB)

return dist

#defining for buzzer sounds

def sound\_alarm(path):

music = pyglet.media.load(r"D:\Python Projects\Drowsiness-Detetction-system-main\Drowsiness-Detetction-system-main\alarm.wav")

music.play()

#defing blinked for different phases.

def blinked(a, b, c, d, e, f):

up = compute(b, d) + compute(c, e)

down = compute(a, f)

ratio = up/(2.0\*down)

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)

face\_frame = frame.copy()

for face in faces:

x1 = face.left()

y1 = face.top()

x2 = face.right()

y2 = face.bottom()

cv2.rectangle(face\_frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

landmarks = predictor(gray, face)

landmarks = face\_utils.shape\_to\_np(landmarks)

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])

if (left\_blink == 0 or right\_blink == 0):

sleep += 1

drowsy = 0

active = 0

if (sleep > 6):

status = "SLEEPING !!!"

color = (255, 0, 0)

if (sleep > 45):

status = "SLEEPING WARNING !!!"

color = (255, 0, 0)

sound\_alarm(r"D:\Python Projects\Drowsiness-Detetction-system-main\Drowsiness-Detetction-system-main\alarm.wav")

#defining the different phases and buzzer starts when the person sleeps for more thank 5sec.

elif (left\_blink == 1 or right\_blink == 1):

sleep = 0

active = 0

drowsy += 1

if (drowsy > 6):

status = "Drowsy !"

color = (0, 0, 255)

else:

drowsy = 0

sleep = 0

active += 1

if (active > 6):

status = "Active :)"

color = (0, 255, 0)

cv2.putText(frame, status, (100, 100),

cv2.FONT\_HERSHEY\_DUPLEX, 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)

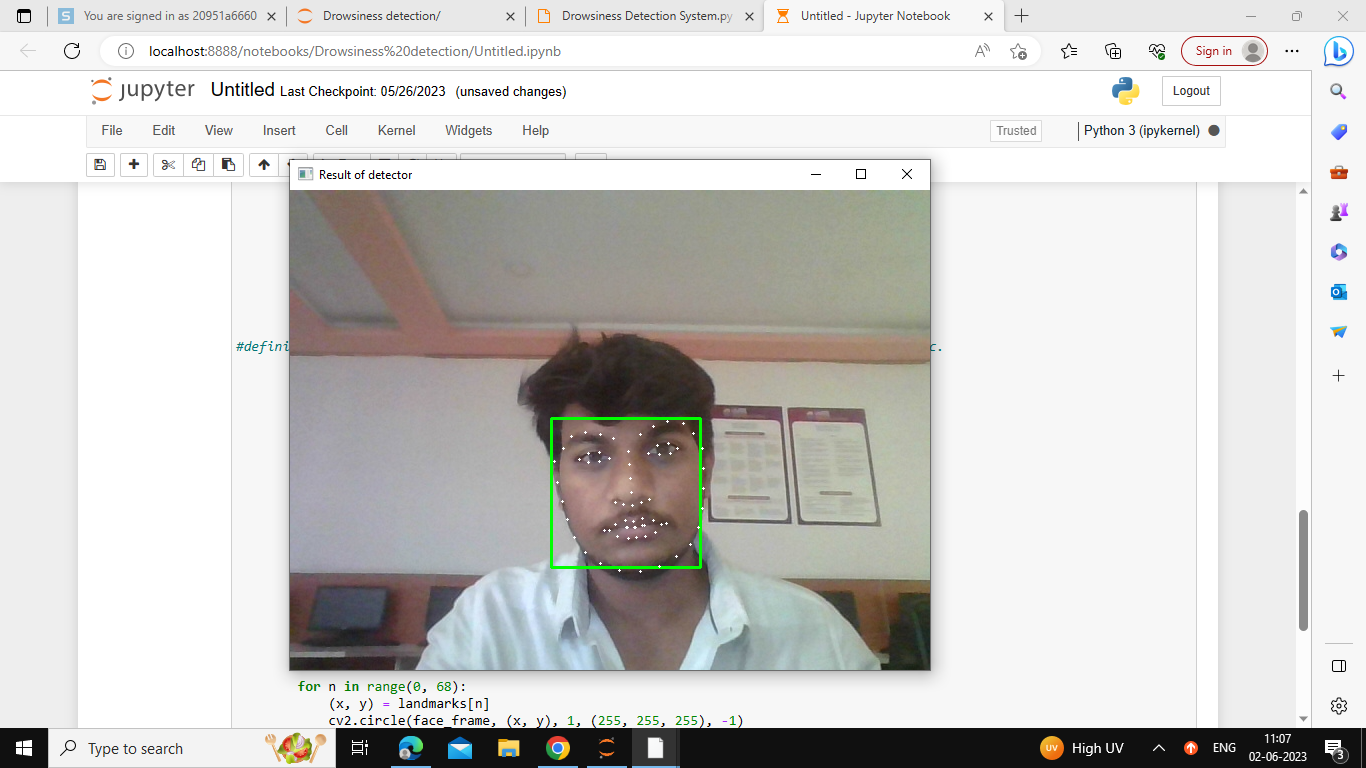
key = cv2.waitKey(1)

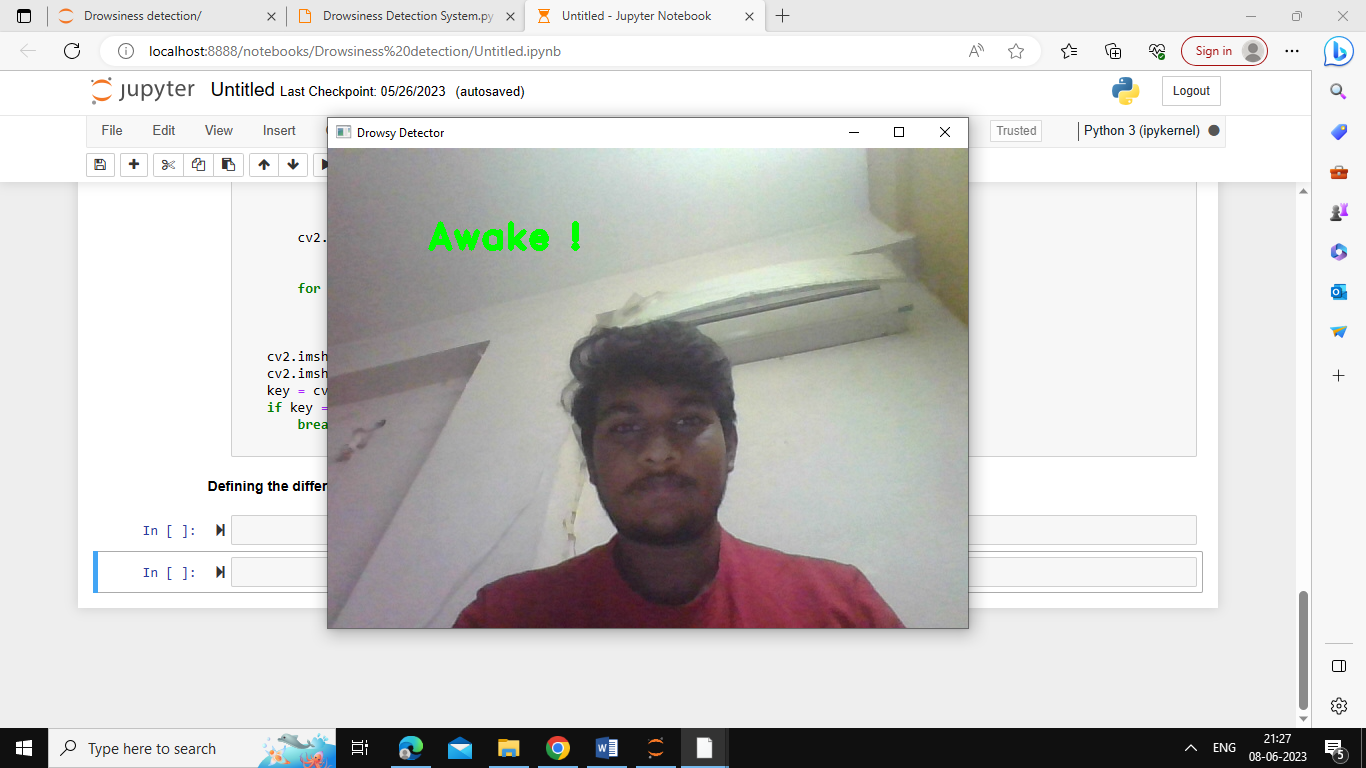
if key == 27:

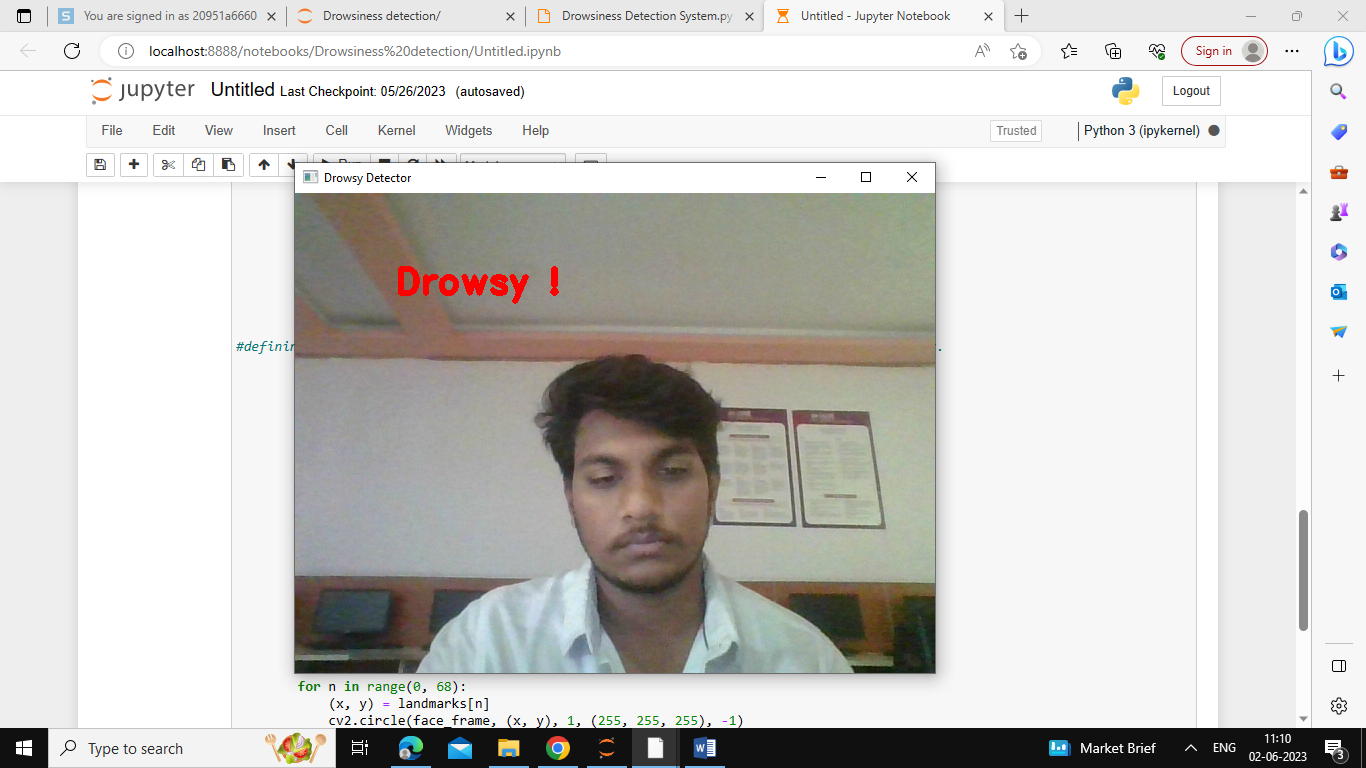
break

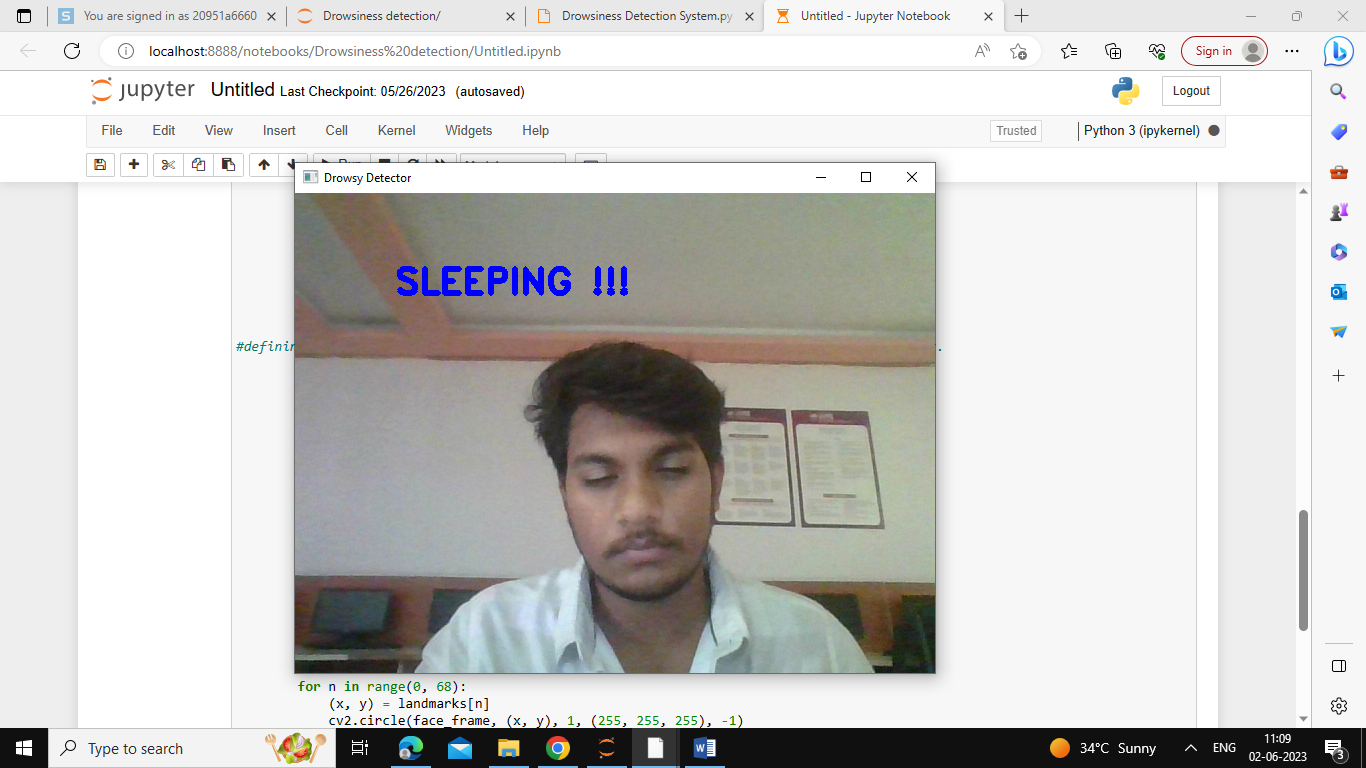
***5.RESULTS:***

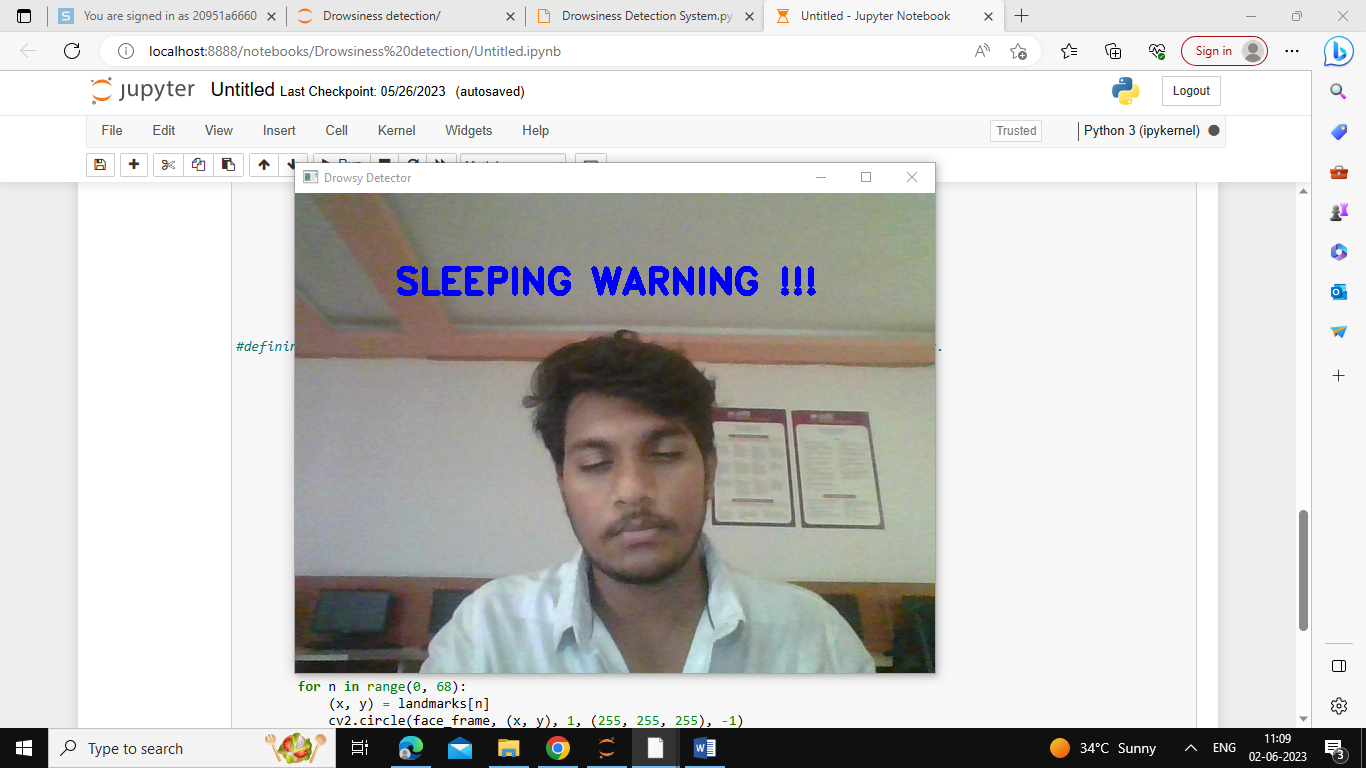
The results show us the following faces whether a person is awake, Sleeping, or drowsy











***6.Conclusion*:**

In conclusion, the implemented drowsiness detection system based on facial landmarks and eye blink analysis shows promise in accurately identifying drowsiness in individuals. The key findings and contributions of the research are as follows:

1. Effectiveness and Reliability:

* The system successfully detects facial landmarks using the dlib library, allowing for precise analysis of eye blink patterns.
* By calculating eye aspect ratios and analyzing blink patterns, the system can identify drowsiness levels in real-time.
* The integration of an audio alarm system adds an additional layer of alertness, helping to mitigate potential risks associated with drowsiness.

2. Potential Impact and Significance:

* The drowsiness detection system holds significant potential in various fields, particularly in transportation safety.
* It can be implemented in vehicles, helping to prevent accidents caused by drowsy driving.
* In industries where fatigue can lead to severe consequences, such as healthcare or aviation, this system can enhance safety measures and prevent accidents due to human error caused by drowsiness.

3. Recommendations for Further Improvements:

* Enhance robustness to varying lighting conditions and head movements to improve accuracy under different scenarios.
* Consider additional features or machine learning techniques to capture more comprehensive drowsiness indicators, such as facial expressions or head pose.
* Conduct extensive testing on diverse datasets with a larger sample size to validate and fine-tune the system's performance.

4. Future Research Directions:

* Explore the integration of other physiological signals, such as heart rate or EEG, to create a multimodal drowsiness detection system for improved accuracy and reliability.
* Investigate the use of deep learning techniques, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), to automatically learn discriminative features for drowsiness detection.
* Further investigate real-time implementation optimizations to improve the system's efficiency and make it more suitable for resource-constrained environments.

In summary, the implemented drowsiness detection system demonstrates promising results and contributes to the field of computer vision and image processing in detecting drowsiness. With further refinements and advancements, this system has the potential to greatly impact transportation safety and other industries where drowsiness poses a significant risk.