NAAN MUDHALVAN PROJECT REPORT

ANNA UNIVERSITY REGIONAL CAMPUS COIMBATORE

INTERNET OF THINGS(IoT)

TOPIC: DROWSINESS DETECTION AND ALERTING SYSTEM

TEAM ID: NM2023TMD00121

FACULTY MENTOR:

Dr.S.SARAVANAKUMAR

TEAM MEMBERS:

NIVAASHINI.S

RAGAVI.R

KOWSALYA.S

RASHMIYA.J

TABLE OF CONTENTS:

INTRODUCTION

PROJECT OVERVIEW:

Drowsiness detection is the most necessary procedure to prevent any road accidents. The main aim of this project is to construct a smart alert technique for building intelligent vehicles that can automatically avoid drowsy driver impairment. Hence, it is required to design a robust alert system to avoid the cause of the mishap. In this project, we address a drowsy driver alert system that has been developed using such a technique in which the Video Stream Processing (VSP) is analysed by the eye blink concept through an Eye Aspect Ratio (EAR) and Euclidean distance of the eye. The face landmark algorithm is also used as a proper way to eye detection. When the driver's fatigue is detected, the IoT module issues a warning message along with the impact of collision and location information, thereby alerting with the help of a voice speaking.

PURPOSE:

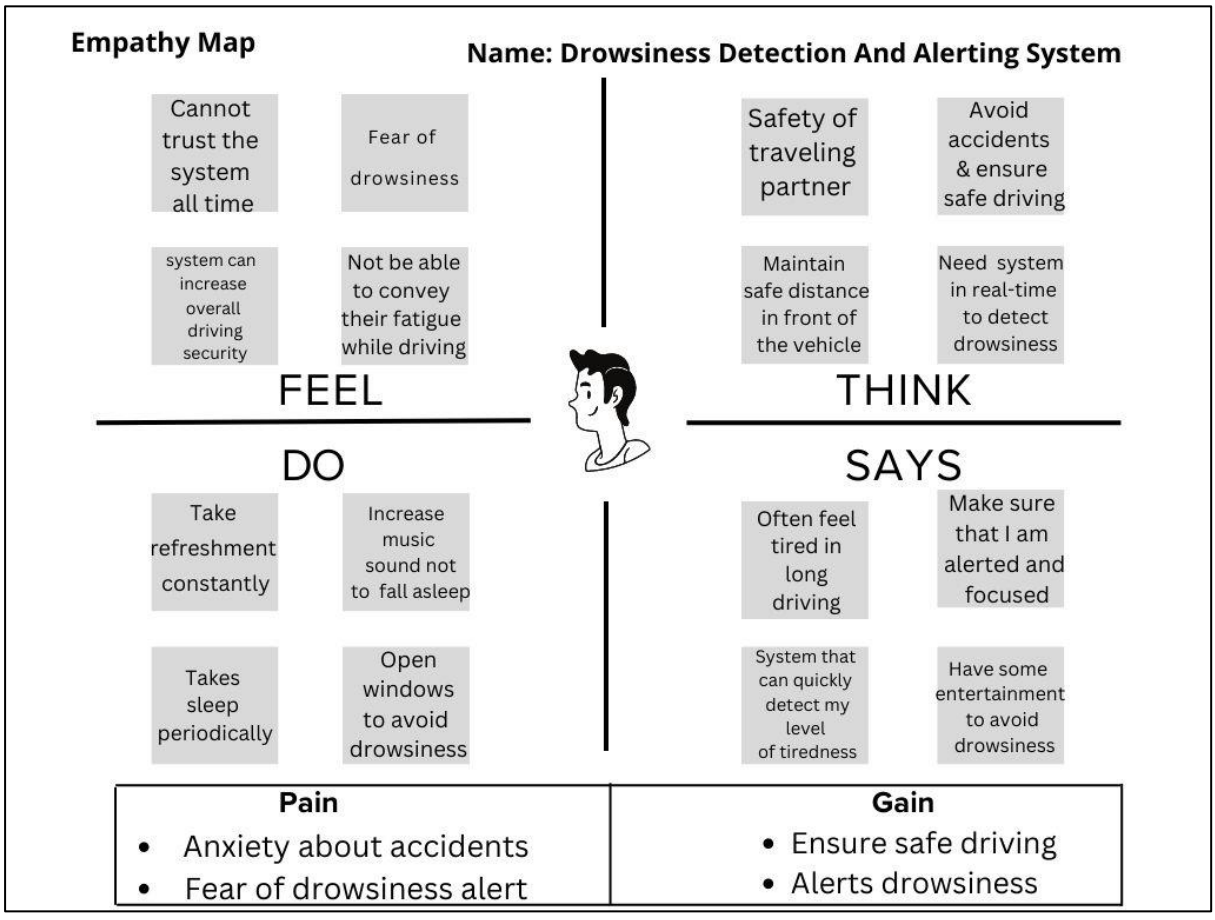
Many people drive on highway day and night. Bus drivers,truck drivers and people traveling long distance. This causes lack of sleep. Driving sleepy is very dangerous. Most of the accidents occur due to drowsiness of the driver. So to prevent these accidents we build a system using python. The objective of this python project is to built a drowsiness detection system that will detect that a person's eye are closed for few seconds. The system will alert the driver when drowsiness is detected. The purpose of this system is to aid in the prevention of accidents passenger and commercial vehicles.

It detects the early symptoms of drowsiness before the driver has fully lost all attentiveness and warn the driver that they no longer capable of operating the vehicle safely.

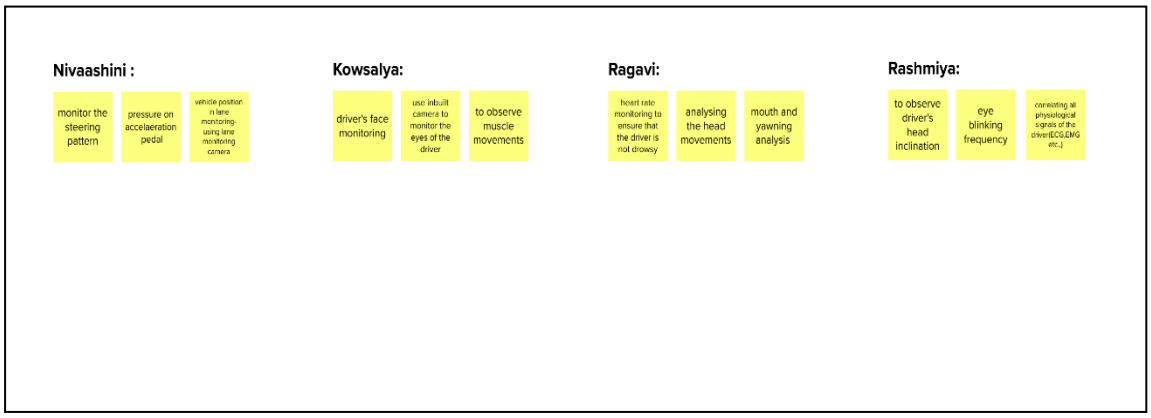
IDEATION AND PROPOSED SOLUTION

Customer Problem Statement:

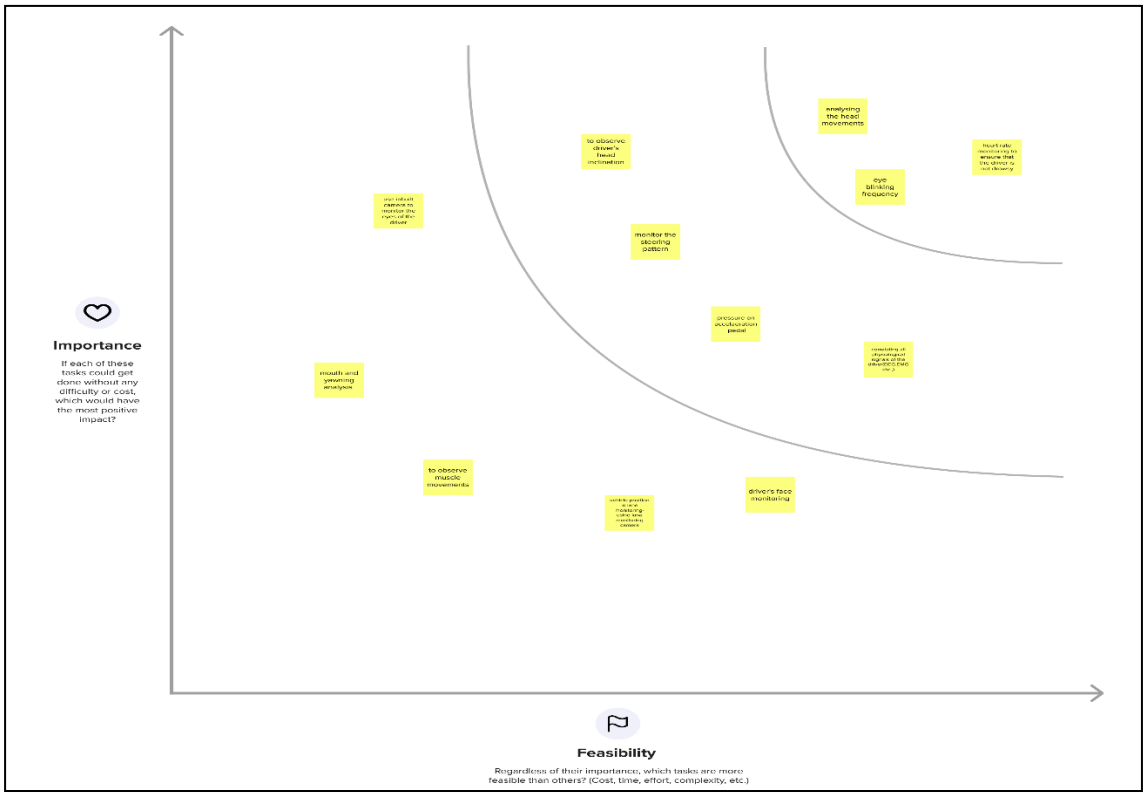
Empathy Map:



Brainstorm, Idea Listing and Grouping:



Idea Prioritization:



Proposed Solution:

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Drowsiness and fatigue are main reasons for most of the unfortunate accidents.
2.	Idea / Solution description	A method for detecting and alerting tiredness is created. Using an algorithm, this device tracks the driver's eyelid movement and records the information in the cloud. And the data from the cloud starts an alarm sound in the driver's mobile app to inform him when the eyelid is identified at sleepiness level.
3.	Novelty / Uniqueness	The uniqueness of this system is that it gives instantaneous alarm to the driver to be aware. Also, it is user-friendly.
4.	Social Impact / Customer Satisfaction	This approach ensures safe travel and has a positive effect on living securely while travelling. This ensures consumer pleasure and an improved journey.
5.	Business Model (Revenue Model)	According to the survey, long-distance travellers prefer night time travel, and commodities are transported by large vehicles at night for extended periods of time. As a result, the drivers could experience driving fatigue. This device is therefore in demand for all drivers who commute at night. As a result, this product is in high demand because it guarantees people's safety in their daily lives.
6.	Scalability of the Solution	Increasing the accuracy of the system improves the system performance.

REQUIREMENT ANALYSIS

Functional Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	The user can Register through Email.
FR-2	User Confirmation	The entire user can confirm through OTP via Email.
FR-3	Authentication Requirement	The user is given proper credentials to access the device. User set password to protect his data.
FR-4	Authorization levels	Authorization determines the user access right via password or Email verification.
FR-5	External interfaces	A GPS is connected to determine the location and route information. A camera is used to detect the eyelid movements of the driver. Collected data is stored in an external IBM cloud.
FR-6	Reporting requirement	If the driver feels drowsy an alerting system alerts the driver by detecting his eyelid movements and data from the cloud.

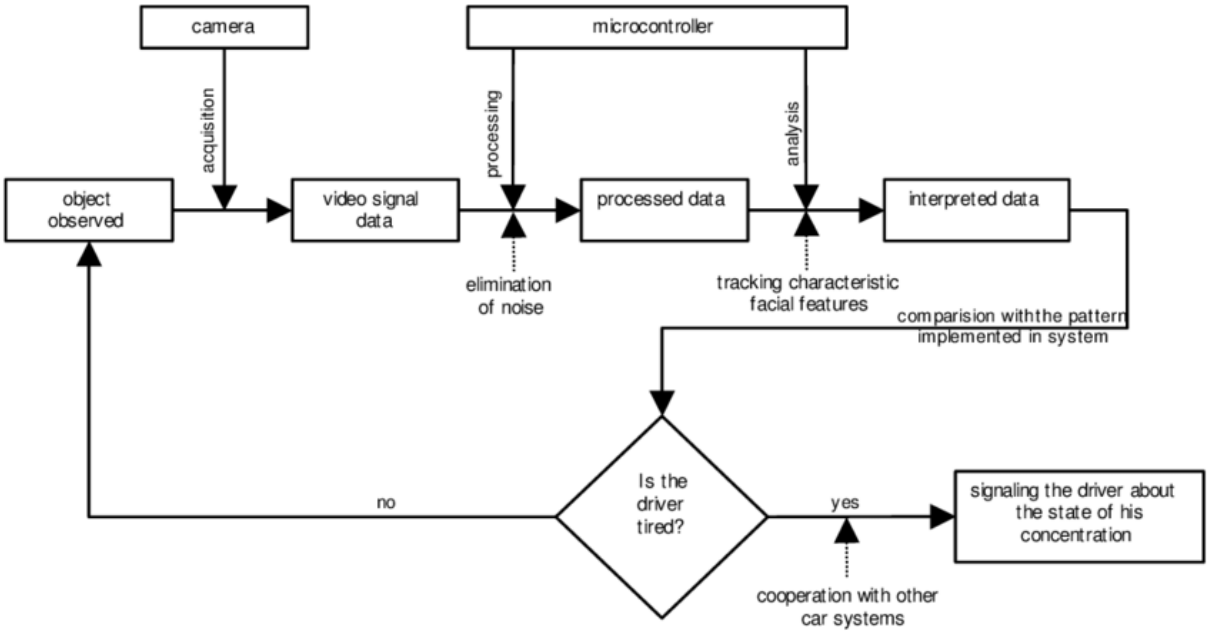
Non-functional Requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Extremely User friendly, Installation of the system is easy.
NFR-2	**Security**	Ensures high level security of the system and the data received are being stored in the cloud and is protected.
NFR-3	**Reliability**	System works very efficiently unless the internet connection is unavailable.
NFR-4	**Performance**	System is highly responsive as far as there is the internet connectivity for the information collected from the user being stored in the cloud and the result is processed back to user as an alert message.

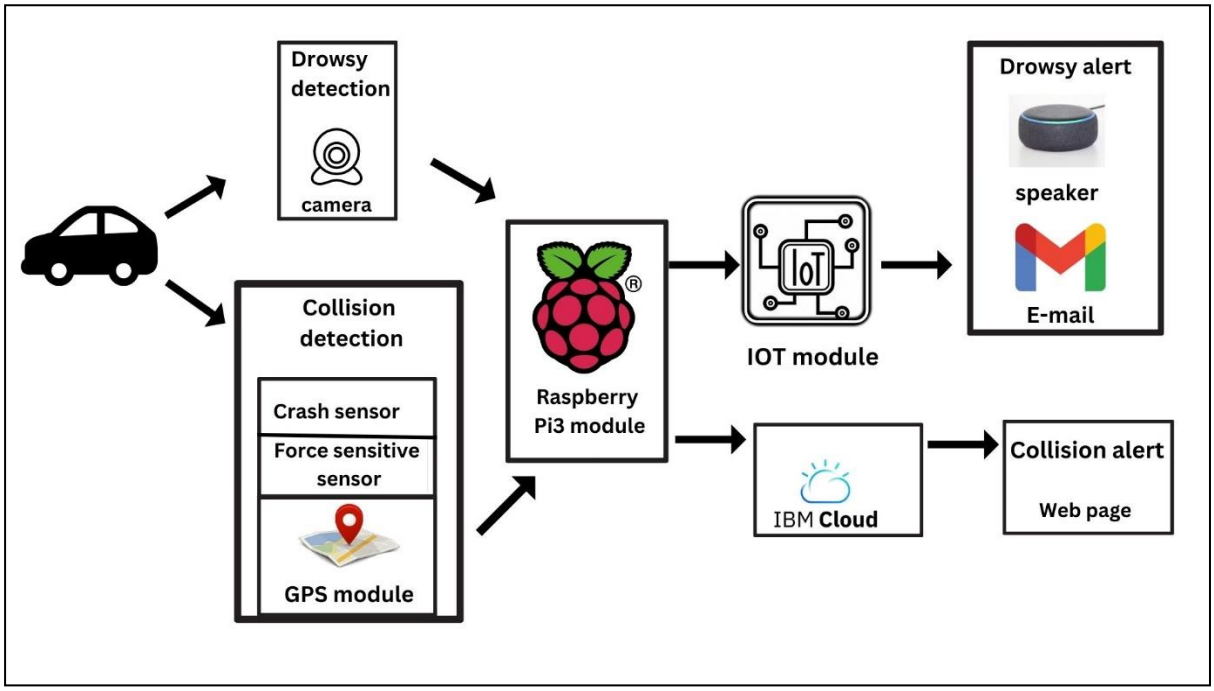
NFR-5	**Availability**	The system's functionality is available all the time while the driver is driving.
NFR-6	**Scalability**	Updating the version of the system any further will not have a negative impact on the performance.

PROJECT DESIGN

Data Flow Diagrams:



Solution and technical Architecture:



User stories:

Table-1: Components & Technologies:

S. No	Component	Description	Technology
1.	User Interface	.The eye movement and face movement of the driver is captured continuously using a camera.	Web camera connected with internet.
2.	GPS module	A GPS module is used to track the location of the driver.	Raspberry Pi GPS Module.
3.	Cloud Database	Database service on cloud to store the collected data.	IBM cloud.
4.	Speaker and email	Used to alert the driver and send relevant message to the user regarding it.	Registered email, audible speaker.
5.	Logical application-1	The code is written using python programming.	Python IDLE 3.7.0

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	**Usability**	Extremely User friendly, Installation of the system is easy.	Web camera.
2.	**Security**	Ensures high level security of the system and the data received are being stored in the cloud and is protected.	IBM cloud storage.
3.	**Reliability**	System works very efficiently unless the internet connection is unavailable.	Wi-Fi or Internet connectivity medium
4.	**Performance**	System is highly responsive as far as there is the internet connectivity for the information collected from the user being stored in the cloud and the result is processed back to user as an alert message.	Speaker or E-mail.

5.	**Availability**	The system's functionality is available all the time while the driver is driving.	Wi-Fi connectivity
6.	**Scalability**	Updating the version of the system any further will not have a negative impact on the performance.	IOT

CODING & SOLUTIONS

Code:

```
import cv2
import dlib
import numpy as np
import pyttsx3
import sys
from scipy.spatial import distance
from imutils import face_utils
import ibmiotf.device
import pygame

#Provide your IBM Watson Device Credentials
organization = "2w8p10"
deviceType = "Drowsy"
deviceId = "12052001"
authMethod = "token"
authToken = "03121975"

def ibmstart(x):

    def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data['command'])
        print(cmd)
```

```
    try:
      deviceOptions = {"org": organization, "type": deviceType, "id":
deviceId, "auth-method": authMethod, "auth-token": authToken}
      deviceCli = ibmiotf.device.Client(deviceOptions)
      #.....................................


    except Exception as e:
      print("Caught exception connecting device: %s" % str(e))
      sys.exit()


    deviceCli.connect()
    data = { 'Status' : x}
    #print data
    def myOnPublishCallback():
        print ("Published Status = %s" % x, "to IBM Watson")


    success = deviceCli.publishEvent("DD", "json", data, qos=0,
on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")



    deviceCli.commandCallback = myCommandCallback
    deviceCli.disconnect()
```

```
# INITIALIZING THE pyttsx3 SO THAT
# ALERT AUDIO MESSAGE CAN BE DELIVERED
engine = pyttsx3.init()

# SETTING UP OF CAMERA TO 1 YOU CAN
# EVEN CHOOSE 0 IN PLACE OF 1
cap = cv2.VideoCapture(0)

# FACE DETECTION OR MAPPING THE FACE TO
# GET THE Eye AND EYES DETECTED
face_detector = dlib.get_frontal_face_detector()

# PUT THE LOCATION OF .DAT FILE (FILE FOR
# PREDECTING THE LANDMARKS ON FACE )
dlib_facelandmark =
dlib.shape_predictor("C:/Users/kowsa/AppData/Local/Programs/Pyt
hon/Python311/shape_predictor_68_face_landmarks.py")

# FUNCTION CALCULATING THE ASPECT RATIO FOR
# THE Eye BY USING EUCLIDEAN DISTANCE FUNCTION
def Detect_Eye(eye):
  poi_A = distance.euclidean(eye[1], eye[5])
  poi_B = distance.euclidean(eye[2], eye[4])
  poi_C = distance.euclidean(eye[0], eye[3])
  aspect_ratio_Eye = (poi_A+poi_B)/(2*poi_C)
```

```
  return aspect_ratio_Eye


# MAIN LOOP IT WILL RUN ALL THE UNLESS AND
# UNTIL THE PROGRAM IS BEING KILLED BY THE USER
while True:

  null, frame = cap.read()
  flag=0
  gray_scale = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

  faces = face_detector(gray_scale)

  for face in faces:
    face_landmarks = dlib_facelandmark(gray_scale, face)
    leftEye = []
    rightEye = []

    # THESE ARE THE POINTS ALLOCATION FOR THE
    # LEFT EYES IN .DAT FILE THAT ARE FROM 42 TO 47
    for n in range(42, 48):
      x = face_landmarks.part(n).x
      y = face_landmarks.part(n).y
      rightEye.append((x, y))
      next_point = n+1
```

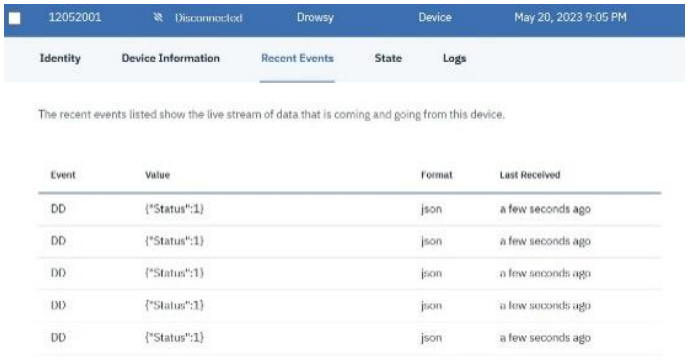
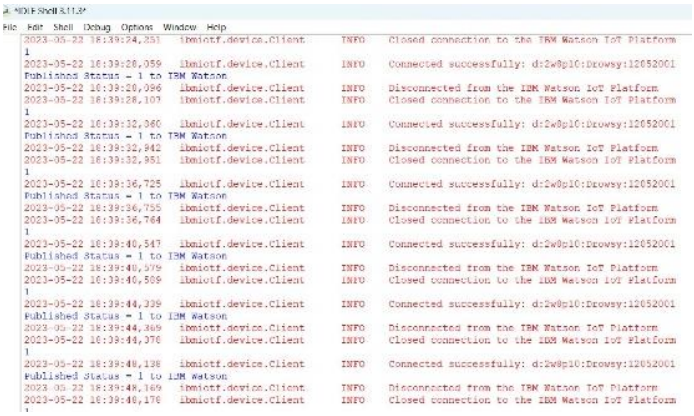
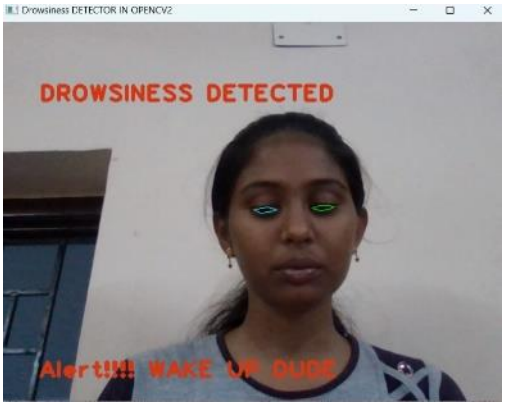
```
  if n == 47:
    next_point = 42
  x2 = face_landmarks.part(next_point).x
  y2 = face_landmarks.part(next_point).y
  cv2.line(frame, (x, y), (x2, y2), (0, 255, 0), 1)

# THESE ARE THE POINTS ALLOCATION FOR THE
# RIGHT EYES IN .DAT FILE THAT ARE FROM 36 TO 41
for n in range(36, 42):
  x = face_landmarks.part(n).x
  y = face_landmarks.part(n).y
  leftEye.append((x, y))
  next_point = n+1
  if n == 41:
    next_point = 36
  x2 = face_landmarks.part(next_point).x
  y2 = face_landmarks.part(next_point).y
  cv2.line(frame, (x, y), (x2, y2), (255, 255, 0), 1)

# CALCULATING THE ASPECT RATIO FOR LEFT
# AND RIGHT EYE
right_Eye = Detect_Eye(rightEye)
left_Eye = Detect_Eye(leftEye)
Eye_Rat = (left_Eye+right_Eye)/2
```

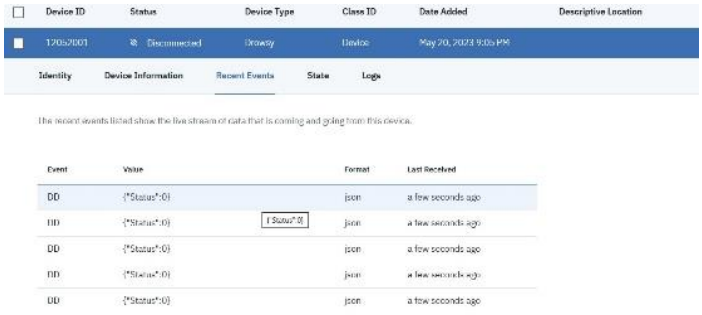
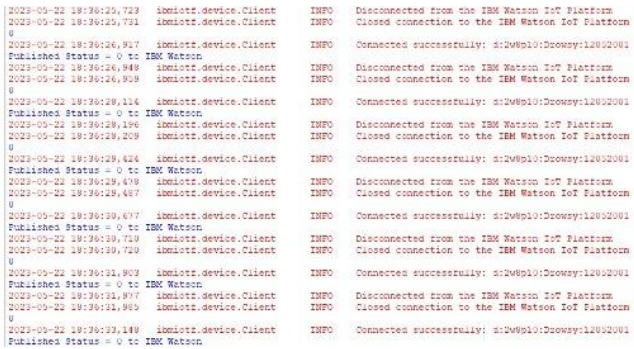
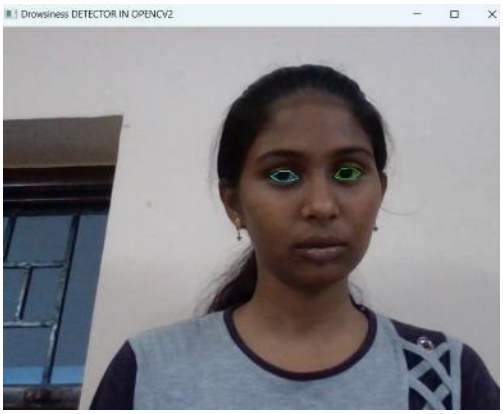
```
 # NOW ROUND OF THE VALUE OF AVERAGE MEAN
 # OF RIGHT AND LEFT EYES
 Eye_Rat = round(Eye_Rat, 2)

 # THIS VALUE OF 0.25 (YOU CAN EVEN CHANGE IT)
 # WILL DECIDE WHETHER THE PERSONS'S EYES ARE CLOSE OR NOT
 if Eye_Rat < 0.25:
   cv2.putText(frame, "DROWSINESS DETECTED", (50, 100),
       cv2.FONT_HERSHEY_PLAIN, 2, (21, 56, 210), 3)
   cv2.putText(frame, "Alert!!!! WAKE UP DUDE", (50, 450),
       cv2.FONT_HERSHEY_PLAIN, 2, (21, 56, 212), 3)

   # CALLING THE AUDIO FUNCTION OF TEXT TO
   # AUDIO FOR ALERTING THE PERSON
   engine.say("Alert!!!! WAKE UP DUDE")
   flag=1
   engine.runAndWait()



cv2.imshow("Drowsiness DETECTOR IN OPENCV2", frame)
print(flag)
ibmstart(flag)
'''
while True:
```

```
            data = { 'Status' : x}
            #print data
            def myOnPublishCallback():
               print ("Published Status = %s" % x, "to IBM Watson")

            success = deviceCli.publishEvent("DD", "json", data, qos=0,
on_publish=myOnPublishCallback)
            if not success:
               print("Not connected to IoTF")
            time.sleep(1)

            deviceCli.commandCallback = myCommandCallback
 '''
 #r1 =
requests.get('https://api.thingspeak.com/update?api_key=SEWZDEK
7APG3P0P8&field1='+str(flag))
 #print(r1.status_code)
 key = cv2.waitKey(9)
 if key == 20:
   break

# Disconnect the device and application from the cloud
#deviceCli.disconnect()
cap.release()
cv2.destroyAllWindows()
```

RESULTS

Performance Metrics:

Parameter	Values	Screenshot
Drowsiness detected	**IBM CLOUD OUTPUT**  12052001 Disconnected Drowsy Device May 20, 2023 9:05 PM Identity Device Information Recent Events State Logs The recent events listed show the live stream of data that is coming and going from this device. Event Value Format Last Received DD {"Status":1} json a few seconds ago DD {"Status":1} json a few seconds ago DD {"Status":1} json a few seconds ago DD {"Status":1} json a few seconds ago DD {"Status":1} json a few seconds ago **PYTHON OUTPUT**  IDLE Shell 3.11.3 File Edit Shell Debug Options Window Help 2023-05-22 16:39:24,251 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform 1 2023-05-22 16:39:28,059 ibmiotf.device.Client INFO Connected successfully: d:2w8p10:Drowsy:12052001 Published Status = 1 to IBM Watson 2023-05-22 16:39:28,096 ibmiotf.device.Client INFO Disconnected from the IBM Watson IoT Platform 2023-05-22 16:39:28,107 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform 1 2023-05-22 16:39:32,360 ibmiotf.device.Client INFO Connected successfully: d:2w8p10:Drowsy:12052001 Published Status = 1 to IBM Watson 2023-05-22 16:39:32,942 ibmiotf.device.Client INFO Disconnected from the IBM Watson IoT Platform 2023-05-22 16:39:32,951 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform 1 2023-05-22 16:39:36,725 ibmiotf.device.Client INFO Connected successfully: d:2w8p10:Drowsy:12052001 Published Status = 1 to IBM Watson 2023-05-22 16:39:36,755 ibmiotf.device.Client INFO Disconnected from the IBM Watson IoT Platform 2023-05-22 16:39:36,764 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform 1 2023-05-22 16:39:40,547 ibmiotf.device.Client INFO Connected successfully: d:2w8p10:Drowsy:12052001 Published Status = 1 to IBM Watson 2023-05-22 16:39:40,579 ibmiotf.device.Client INFO Disconnected from the IBM Watson IoT Platform 2023-05-22 16:39:40,589 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform 1 2023-05-22 16:39:44,339 ibmiotf.device.Client INFO Connected successfully: d:2w8p10:Drowsy:12052001 Published Status = 1 to IBM Watson 2023-05-22 16:39:44,369 ibmiotf.device.Client INFO Disconnected from the IBM Watson IoT Platform 2023-05-22 16:39:44,370 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform 1 2023-05-22 16:39:48,138 ibmiotf.device.Client INFO Connected successfully: d:2w8p10:Drowsy:12052001 Published Status = 1 to IBM Watson 2023-05-22 16:39:48,169 ibmiotf.device.Client INFO Disconnected from the IBM Watson IoT Platform 2023-05-22 16:39:48,170 ibmiotf.device.Client INFO Closed connection to the IBM Watson IoT Platform	EYES CLOSED  Drowsiness DETECTOR IN OPENCV2 DROWSINESS DETECTED Alert!!!! WAKE UP DUDE

Not drowsy	**IBM CLOUD OUTPUT**  **PYTHON OUTPUT** 	EYES OPEN 


```
2023-05-22 19:36:25,723   ibmiotf.device.Client      INFO    Disconnected from the IBM Watson IoT Platform
2023-05-22 19:36:25,731   ibmiotf.device.Client      INFO    Closed connection to the IBM Watson IoT Platform
0
2023-05-22 19:36:26,917   ibmiotf.device.Client      INFO    Connected successfully: d:2w0p10:Drowsy:1205/2001
Published Status = 0 to IBM Watson
2023-05-22 19:36:26,948   ibmiotf.device.Client      INFO    Disconnected from the IBM Watson IoT Platform
2023-05-22 19:36:26,959   ibmiotf.device.Client      INFO    Closed connection to the IBM Watson IoT Platform
0
2023-05-22 19:36:28,114   ibmiotf.device.Client      INFO    Connected successfully: d:2w0p10:Drowsy:1205/2001
Published Status = 0 to IBM Watson
2023-05-22 19:36:28,196   ibmiotf.device.Client      INFO    Disconnected from the IBM Watson IoT Platform
2023-05-22 19:36:28,209   ibmiotf.device.Client      INFO    Closed connection to the IBM Watson IoT Platform
0
2023-05-22 19:36:29,424   ibmiotf.device.Client      INFO    Connected successfully: d:2w0p10:Drowsy:1205/2001
Published Status = 0 to IBM Watson
2023-05-22 19:36:29,478   ibmiotf.device.Client      INFO    Disconnected from the IBM Watson IoT Platform
2023-05-22 19:36:29,487   ibmiotf.device.Client      INFO    Closed connection to the IBM Watson IoT Platform
0
2023-05-22 19:36:30,677   ibmiotf.device.Client      INFO    Connected successfully: d:2w0p10:Drowsy:1205/2001
Published Status = 0 to IBM Watson
2023-05-22 19:36:30,719   ibmiotf.device.Client      INFO    Disconnected from the IBM Watson IoT Platform
2023-05-22 19:36:30,729   ibmiotf.device.Client      INFO    Closed connection to the IBM Watson IoT Platform
0
2023-05-22 19:36:31,909   ibmiotf.device.Client      INFO    Connected successfully: d:2w0p10:Drowsy:1205/2001
Published Status = 0 to IBM Watson
2023-05-22 19:36:31,977   ibmiotf.device.Client      INFO    Disconnected from the IBM Watson IoT Platform
2023-05-22 19:36:31,989   ibmiotf.device.Client      INFO    Closed connection to the IBM Watson IoT Platform
0
2023-05-22 19:36:33,148   ibmiotf.device.Client      INFO    Connected successfully: d:2w0p10:Drowsy:1205/2001
Published Status = 0 to IBM Watson
```

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- The drowsiness detection system is capable of detecting drowsiness in quickly. The system which can differentiate normal eye blink and drowsiness can prevent the driver from entering the state of sleepiness while driving.

DISADVANTAGES:

- Aging of sensors are main disadvantage of this system and misconception of eye movement may be a disadvantage of this system.

CONCLUSION:

Thus we conclude that, this drowsiness detection and alerting system is used if the drivers eyes are closed cumulatively more than a standard value, the system draws the conclusion that the driver is falling asleep, and then it will activate an alarm sound to alert the driver. A non-invasive system to localize the eyes and monitor fatigue was developed.

FUTURE SCOPE:

- The system can be made more accurate using various other parameters such as State of the Car, Detecting Foreign Substances on Face etc.
- An application can be developed where it can alert or prevent the user from sleeping.
- It can be used to develop an IOT device that can be installed in the car to detect driver's drowsiness.
- Similar models and techniques can be used for various other uses such as Netflix, Hotstar and other streaming service platforms can detect whether the person is sleeping and stop the video accordingly.

Demo Video link:

https://github.com/naanmudhalvan-SI/PBL-NT-GP--1585-1680517958/blob/fdacf0891ff88b76dd0424d28a40ba6617557b08/Final%20Deliveries/Project%20Report/Demo%20video.mp4