# Name: Sneha Jha, Lavanya Sengar

# Reg no:199301184,199301186

# Sec-CSE-C

# Project: Driver Drowsiness Detection System

1. **Introduction**:

Many accidents are caused by drowsy drivers. It is becoming one of the leading causes of traffic accidents. According to recent statistics, many accidents are caused by drowsy drivers. Thousands of lives are lost each year as a result of vehicle accidents caused by drowsy drivers. More than 30% of accidents occur due to drowsiness.

For the prevention of this, a system is required which detects the drowsiness and alerts the driver which saves the life. In this project, we present a scheme for driver drowsiness detection. In this, the driver is continuously monitored through a webcam. This model uses image processing techniques that mainly focuses on the face and eyes of the driver. The model extracts the driver's face and predicts the blinking of the eye from the eye region. We use an algorithm to track and analyse the driver's face and eyes to measure perclos. If the blinking rate is high or the drowsiness score is less then the system alerts the driver with a sound.

## **System review :**

This survey is done to comprehend the need and prerequisite of the general population, and to do as such, we went through different sites and applications and looked for the fundamental data. Based on these data, we made an audit that helped us get new thoughts and make different arrangements for our task. We reached the decision that there is a need of such application and felt that there is a decent extent of progress in this field too.

### **Technology used :**

### **Python**:

* + - * 1. Python is an interpreted, high-level, general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

## **Jupyter lab:**

* + - * 1. Project jupyter is a nonprofit organization created to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

### **Image processing** :

In computer science, digital image processing is the use of computer algorithms to perform image processing on digital images.

### **Machine learning :**

Machine learning is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly told.

## **Software requirements specification:**

### **Python:**

Python 3

* 1. **Libraries:**
     1. OpenCV
     2. Tensorflow
     3. Keras
     4. Pygame
  2. **Operating system:**

Windows or ubuntu

1. **Hardware requirements specification:**
   1. Laptop with basic hardware.
      1. Webcam
2. **Functional requirements:**
   1. **Sleep alarm:**

This function will get enabled when the perclos will be low and will produce a sound to wake the user

* 1. **Face detection:**

This will allow the software to track the user's face.

* 1. **Drowsiness score:**

A drowsiness score will be shown around the face which will tell how drowsy our user is which will decide to play the sound or not according to the score

1. **Non-functional requirements:**
   1. **Performance:**

The application should start within 5 seconds of run.

* 1. **Usability:**
     1. The website should be user-friendly with all the required browsing and data options clearly visible and available.
     2. The drowsiness detection system should be able to be used by any user without requiring any training.
     3. The user also does not need to have prior knowledge of any algorithm or techniques to use the system.
     4. User interfaces design create an effective communication medium between a user and the system.
  2. **Maintainability:**

Maintainability is the ease with which a program can be corrected if an error is encountered, adapted if its environment changes, or enhanced if the user desires a change in the requirements.

* 1. **Reliability:**
     1. The higher the mean time between failures (mtbf), the higher is the reliability of the system.
     2. The mean time to repair (mttr) should also be low for the system.
  2. **Security:**
     1. The user will be required to log in to the system to use it.
     2. The administrator will have the system rights, such as registering the users, looking over the data.
  3. **Serviceability:**
     1. Serviceability of a software system is the ability of a service/technical expert to install the software system in a real-time environment, monitor the system while it is running, identify any technical issues in the system and provide a solution to resolve the issue.
     2. Providing periodical reminder for updates to the user, providing a mechanism to debug issues, automatic recovery from failure via rollback mechanism (roll back the software system to previous working condition state).

1. **Time analysis:**

|  |  |
| --- | --- |
| Project discussion: | 1 week |
| Front end development: | 2 weeks |
| Back end development: | 2 weeks |
| Testing: | 1 week |
| Validation: | 1 week |
| Deployment: | 1 week |

**Cost analysis:**

**Hardware:**

A desktop with cpu @3.0ghz,ram 8gb, ssd 256gb

**Software :**

Windows operating system 10, vs code ide

**Supply cost:**

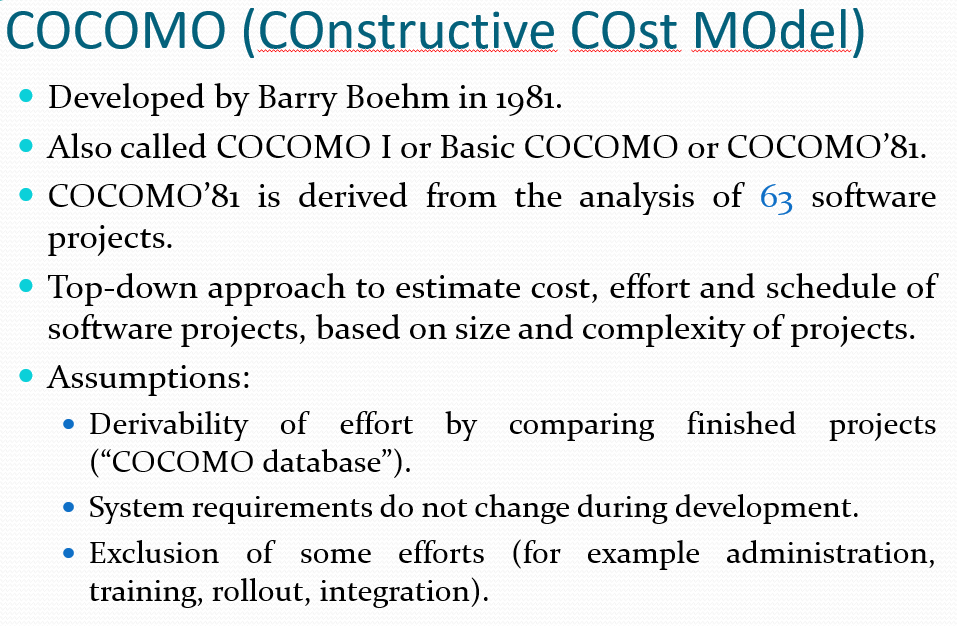
Rs 2000, for all stationery and papers required.

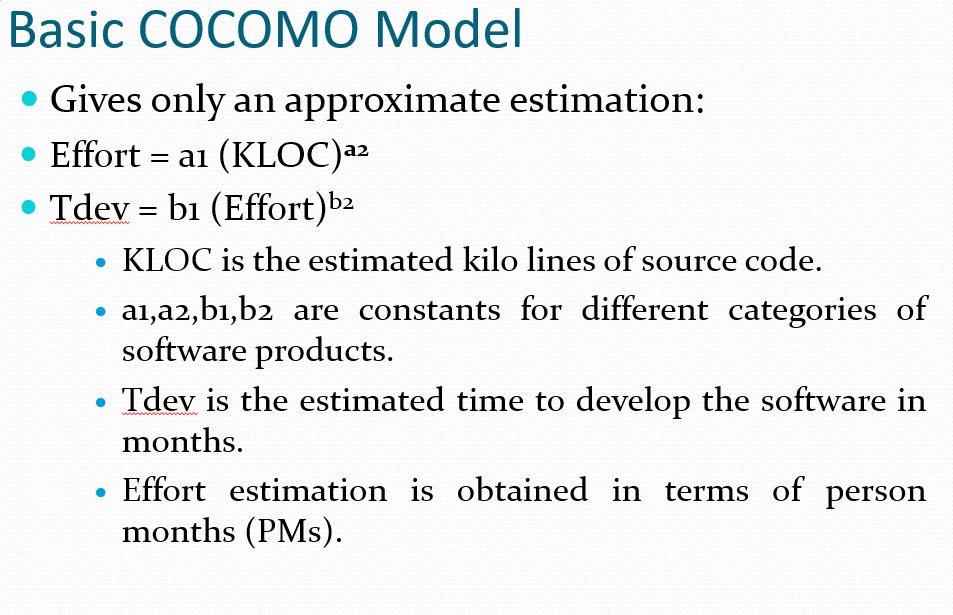
**Personal cost :**

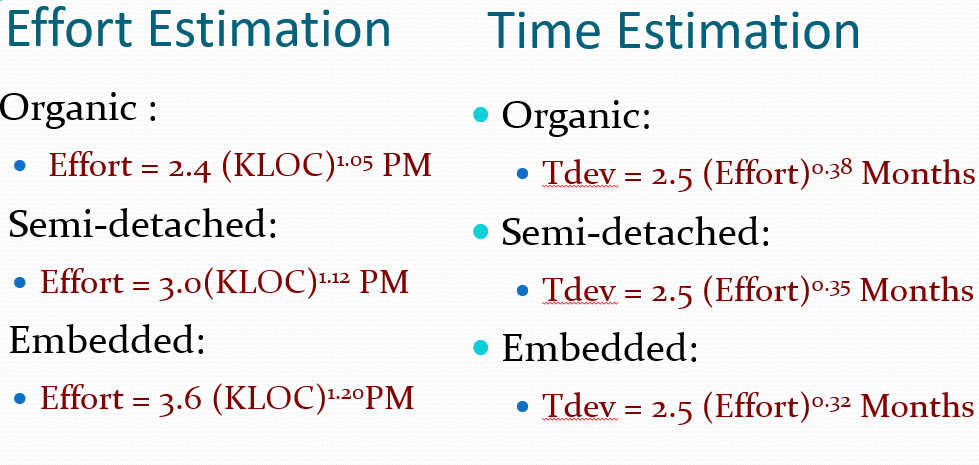
Rs 2000/month , for all other expanses.

**Operational cost:**

For the workstation, and its maintenance.

1. 





Our project is semi-detached:

Estimated 5000 loc required

**Effort:**

=3.0\*(5)^1.12

=3.0\*6.065

=18.195 pm

**Time estimation:**

=2.5\*(efforts)^0.35 months

=2.5\*(18.195) ^0.35 months

=2.5\*2.76 months

=6.9 months

We have used basic cocomo (cost constructive model) for effort and duration estimation.

Cocomo is applied on three classes of software projects:

* Organic
* Semi detached
* Embedded

Our project falls under the category of organic software project.

According to the basic cocomo model:

Effort(e) = ab \* (kloc)^(bb)

Duration(d) = cb\*(e)^(db)

Where, ab = 2.4, bb = 1.05, cb = 2.5and db = 0.38

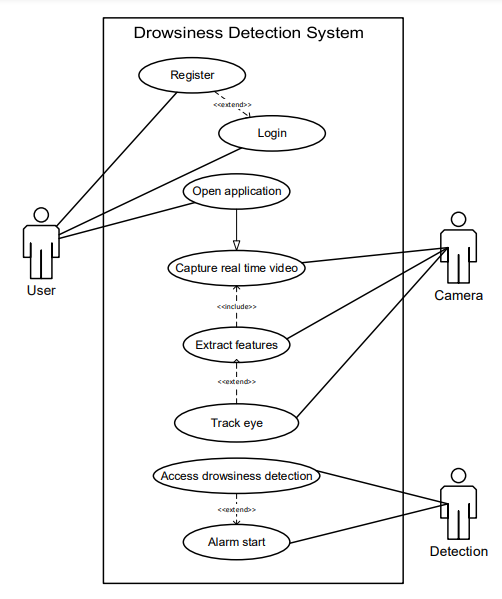
We have assumed that our project will have about 2k loc.

Therefore, using the above equations, we get:

E = 7.60 pm

D = 3.79 pm

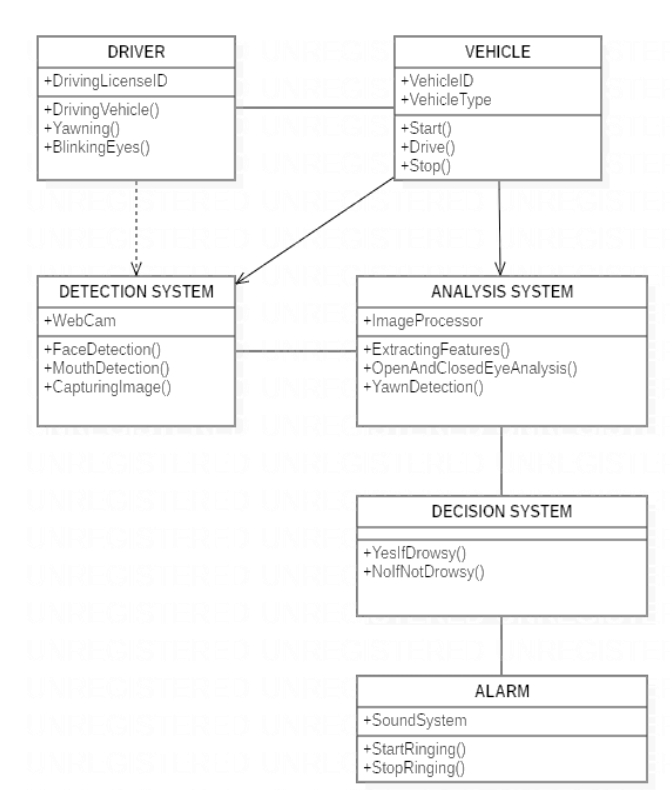
1. **Use case diagram:**



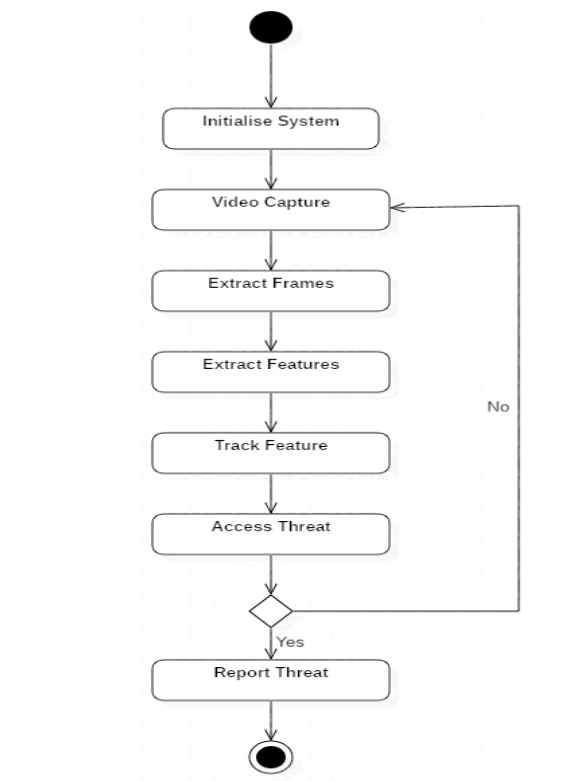
1. **Use case scenario:**

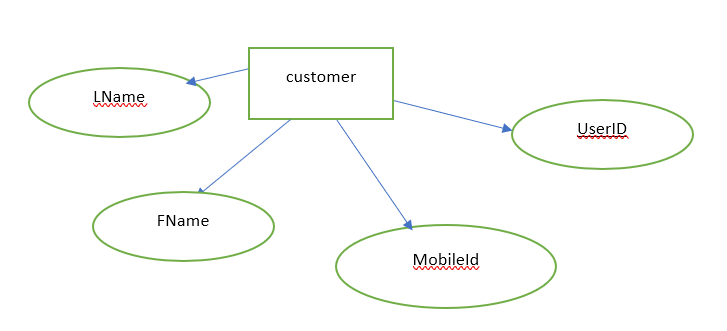
|  |  |
| --- | --- |
| Use case name | Drowsiness detection system |
| Authors | Lavanya, Sneha |
| Actors | User |
| Uses | Accident prevention |
| Basic course/main scenario | 1. User first registers and then logs in. 2. The camera takes real time video as input. 3. The algorithm extracts features and tracks eye movements. 4. The drowsiness score will be measured. 5. The system will start the alarm to alert the user. |
| Other scenarios | User could select the alarm sound |
| Post conditions | A message will be displayed on the screen along with the alarm alert. |

1. **Class diagram:**



1. **Activity model:**



1. **ER model:** 
2. **Project Code:**
   1. **Main.py:**

import cv2

import os

from keras.models import load\_model

import numpy as np

from pygame import mixer

import time

mixer.init()

sound = mixer.Sound('alarm.wav')

face = cv2.CascadeClassifier('haar cascade files\haarcascade\_frontalface\_alt.xml')

leye = cv2.CascadeClassifier('haar cascade files\haarcascade\_lefteye\_2splits.xml')

reye = cv2.CascadeClassifier('haar cascade files\haarcascade\_righteye\_2splits.xml')

lbl=['Close','Open']

model = load\_model('models/cnncat2.h5')

path = os.getcwd()

cap = cv2.VideoCapture(0)

font = cv2.FONT\_HERSHEY\_COMPLEX\_SMALL

count=0

score=0

thicc=2

rpred=[99]

lpred=[99]

while(True):

    ret, frame = cap.read()

    height,width = frame.shape[:2]

    gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

    faces = face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25))

    left\_eye = leye.detectMultiScale(gray)

    right\_eye =  reye.detectMultiScale(gray)

    cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) , thickness=cv2.FILLED )

    for (x,y,w,h) in faces:

        cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )

    for (x,y,w,h) in right\_eye:

        r\_eye=frame[y:y+h,x:x+w]

        count=count+1

        r\_eye = cv2.cvtColor(r\_eye,cv2.COLOR\_BGR2GRAY)

        r\_eye = cv2.resize(r\_eye,(24,24))

        r\_eye= r\_eye/255

        r\_eye=  r\_eye.reshape(24,24,-1)

        r\_eye = np.expand\_dims(r\_eye,axis=0)

        rpred = model.predict\_classes(r\_eye)

        if(rpred[0]==1):

            lbl='Open'

        if(rpred[0]==0):

            lbl='Closed'

        break

    for (x,y,w,h) in left\_eye:

        l\_eye=frame[y:y+h,x:x+w]

        count=count+1

        l\_eye = cv2.cvtColor(l\_eye,cv2.COLOR\_BGR2GRAY)

        l\_eye = cv2.resize(l\_eye,(24,24))

        l\_eye= l\_eye/255

        l\_eye=l\_eye.reshape(24,24,-1)

        l\_eye = np.expand\_dims(l\_eye,axis=0)

        lpred = model.predict\_classes(l\_eye)

        if(lpred[0]==1):

            lbl='Open'

        if(lpred[0]==0):

            lbl='Closed'

        break

    if(rpred[0]==0 and lpred[0]==0):

        score=score+1

        cv2.putText(frame,"Closed",(10,height-20), font, 1,(255,255,255),1,cv2.LINE\_AA)

*#if(rpred[0]==1 or lpred[0]==1):*

    else:

        score=score-1

        cv2.putText(frame,"Open",(10,height-20), font, 1,(255,255,255),1,cv2.LINE\_AA)

    if(score<0):

        score=0

    cv2.putText(frame,'Score:'+str(score),(100,height-20), font, 1,(255,255,255),1,cv2.LINE\_AA)

    if(score>15):

*#person is feeling sleepy so we sound the alarm*

        cv2.imwrite(os.path.join(path,'image.jpg'),frame)

        try:

            sound.play()

        except:  *#isplaying = False*

            pass

        if(thicc<16):

            thicc= thicc+2

        else:

            thicc=thicc-2

            if(thicc<2):

                thicc=2

        cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thicc)

    cv2.imshow('frame',frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):

        break

cap.release()

cv2.destroyAllWindows()

* 1. **Driver Drowsiness.py:**

import os

from keras.preprocessing import image

import matplotlib.pyplot as plt

import numpy as np

from keras.utils.np\_utils import to\_categorical

import random,shutil

from keras.models import Sequential

from keras.layers import Dropout,Conv2D,Flatten,Dense, MaxPooling2D, BatchNormalization

from keras.models import load\_model

def generator(dir, gen=image.ImageDataGenerator(rescale=1./255), shuffle=True,batch\_size=1,target\_size=(24,24),class\_mode='categorical' ):

    return gen.flow\_from\_directory(dir,batch\_size=batch\_size,shuffle=shuffle,color\_mode='grayscale',class\_mode=class\_mode,target\_size=target\_size)

BS= 32

TS=(24,24)

train\_batch= generator(r'C:\Users\Lavanya\Desktop\SE Lab\Project\data\train',shuffle=True, batch\_size=BS,target\_size=TS)

valid\_batch= generator(r'C:\Users\Lavanya\Desktop\SE Lab\Project\data\test',shuffle=True, batch\_size=BS,target\_size=TS)

SPE= len(train\_batch.classes)//BS

VS = len(valid\_batch.classes)//BS

print(SPE,VS)

*# img,labels= next(train\_batch)*

*# print(img.shape)*

model = Sequential([

    Conv2D(32, kernel\_size=(3, 3), activation='relu', input\_shape=(24,24,1)),

    MaxPooling2D(pool\_size=(1,1)),

    Conv2D(32,(3,3),activation='relu'),

    MaxPooling2D(pool\_size=(1,1)),

*#32 convolution filters used each of size 3x3*

*#again*

    Conv2D(64, (3, 3), activation='relu'),

    MaxPooling2D(pool\_size=(1,1)),

*#64 convolution filters used each of size 3x3*

*#choose the best features via pooling*

*#randomly turn neurons on and off to improve convergence*

    Dropout(0.25),

*#flatten since too many dimensions, we only want a classification output*

    Flatten(),

*#fully connected to get all relevant data*

    Dense(128, activation='relu'),

*#one more dropout for convergence' sake :)*

    Dropout(0.5),

*#output a softmax to squash the matrix into output probabilities*

    Dense(4, activation='softmax')

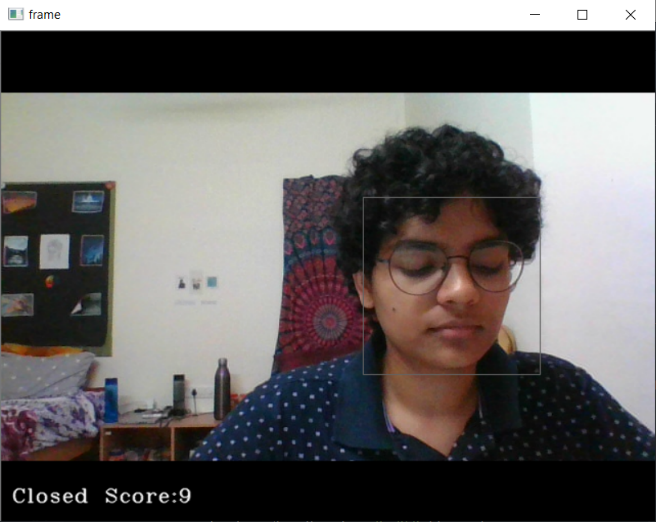
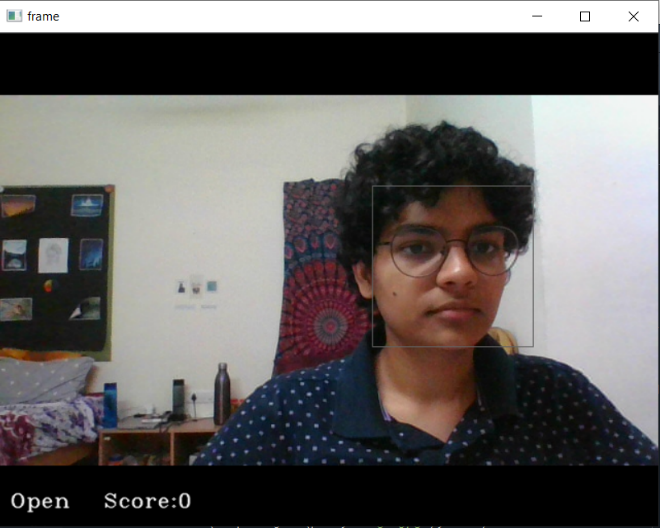
])

model.compile(optimizer='adam',loss='categorical\_crossentropy',metrics=['accuracy'])

model.fit\_generator(train\_batch, validation\_data=valid\_batch,epochs=15,steps\_per\_epoch=SPE ,validation\_steps=VS)

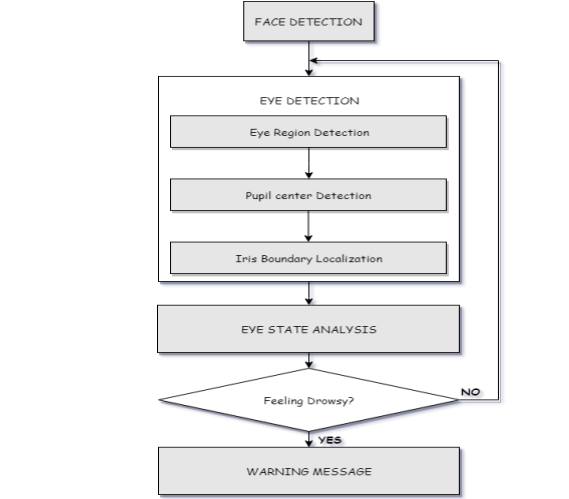
model.save('models/cnnCat2.h5', overwrite=True)

1. **Project Snippets:**

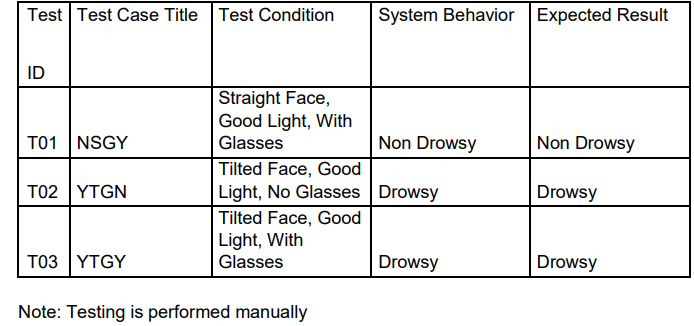


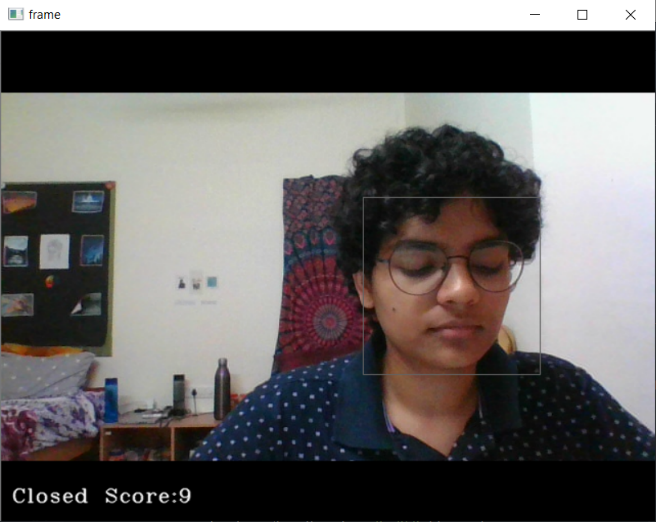
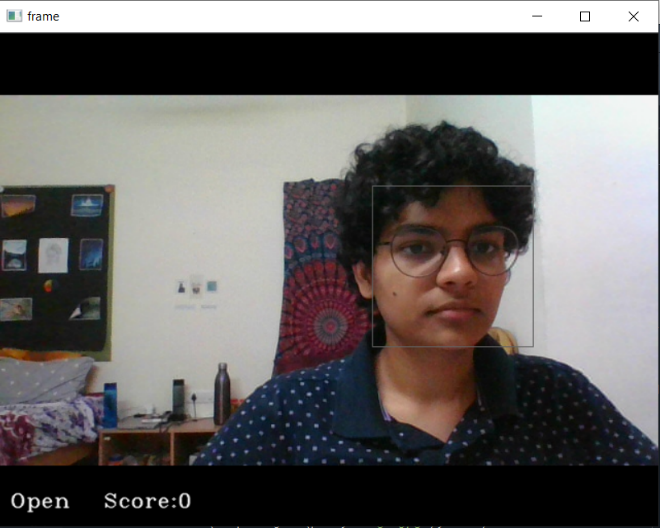
1. **Project Planning :**
   1. **System Model :**

The framework is created utilizing the incremental model. The center model of the framework is first created and afterwards augmented in this way in the wake of testing at each turn. The underlying undertaking skeleton was refined into expanding levels of ability. At the following incremental level, it might incorporate new execution backing and improvement



1. **System Testing:** 
   1. **Test Cases and Test Results:**





1. **Conclusion and Future Scope :**
   1. **Conclusion :**

It completely meets the objectives and requirements of the system. The framework has achieved an unfaltering state where all the bugs have been disposed of. The framework cognizant clients who are familiar with the framework and comprehend it's focal points and the fact that it takes care of the issue of stressing out for individuals having fatigue-related issues to inform them about the drowsiness level while driving.

* 1. **Future Scope :**

The model can be improved incrementally by using other parameters like blink rate, yawning, state of the car, etc. If all these parameters are used it can improve the accuracy by a lot. We plan to further work on the project by adding a sensor to track the heart rate in order to prevent accidents caused due to sudden heart attacks to drivers. Same model and techniques can be used for various other uses like Netflix and other streaming services can detect when the user is asleep and stop the video accordingly. It can also be used in application that prevents user from sleeping