FAHRER BEGLEITER

A REPORT ON THE PROJECT OF

ARTIFICIAL INTELLIGENCE

AND

MACHINE LEARNING

NAME: - RAHUL DHAR

UID: - TTLST1909

TEAMCOGNITO™

A TEAM FOR CHANGE

DECLARATION CERTIFICATE

This is to certify that the work presented in the project entitled **"Fahrer-Begleiter"** in partial fulfilment of the requirement for the award of degree of **Bachelor of Computer Application** is an authentic work carried out under my supervision and guidance.

To the best of my knowledge the content of this project does not form a basis for the award of any previous Degree to anyone else.

Date: 13-07-19

_____

Sanket Sarkar

ACKNOWLEDGEMENT

The satisfaction that accompanies that successful completion of this project would be incomplete without mentioning the name whose ceaseless co-operation made it possible, whose constant guidance and encouragement crown all efforts with success.

Our sincere regards to **Sanket Sarkar** for the guidance, inspiration and constructive suggestions that helped us in the preparation of this project. We convey our sincere thanks to him for the commendable support towards completion of our project

Name: Rahul Dhar

Student Id :- TTLST1909

CERTIFICATE OF APPROVAL

The foregoing project entitled "**Fahrer- Begleiter**" is hereby approved as Minor Project and has been presented in satisfactory manner to warrant its acceptance as prerequisite to the degree for which it has been submitted

It is understood that by this approval, the undersigned do not necessarily endorse any conclusion drawn or opinion expressed therein, but approve the project for the purpose for which it is submitted

CONTENTS

CHAPTER 1:

1.1 INTRODUCTION

Fahrer Begleiter is a Driver Alertness System, which helps to prevent accidents caused by the driver getting drowsy and fatigue. Whenever it detects that the driver is drowsy or fatigue it alerts the driver. Various studies have suggested that around **20% of all road accidents are fatigue-related, up to 50% on certain roads**. Therefore, keeping this in mind, the system is developed.

The alertness system is completely written in Python and various libraries of it namely PyQt5, openCv and playsound. It implements the abilities of Artificial Intelligence to detect the drowsiness in a person .Artificial Intelligence, sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans and other animals. These abilities are achieved using Computer Vision, which is a field of multiple disciplines that care about how computers can gain a high-level understanding from digital images/videos.

- What is PyQt5?

Qt is set of cross-platform C++ libraries that implement high-level APIs for accessing many aspects of modern desktop and mobile systems. These include location and positioning services, multimedia, NFC and Bluetooth connectivity, a Chromium based web browser, as well as traditional UI development.PyQt5 is a comprehensive set of Python bindings for Qt v5. It is implemented as more than 35 extension modules and enables Python to be used as an alternative application development language to C++ on all supported platforms including iOS and Android.PyQt5 may also be embedded in C++ based applications to allow users of those applications to configure or enhance the functionality of those applications.

- What is openCv?

OpenCV (*Open source computer vision*) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source BSD license.OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe.

- What is playsound?

'playsound' is a predefined library of Python which is use to play sound via Python program

CHAPTER 2:

2.1 SCOPE OF THE PROJECT

This system has the capability to alert the driver when he feels fatigue and drowsy. The reaction time of the system is also very fast. This system is user friendly since it has a GUI system. The system, however, can't alert if the driver is wearing spectacles/glasses.

CHAPTER 3

3.1 ALGORITHM

1. **INPUT:**
   a) User decision to start the system or not.
   b) User's image via camera.
   c) User decision to exit the system or not.

2. **OUTPUT:**
   a) Alert the user if he feels fatigue or sleepy.
   b) The image of the user.

3. PROCESS (PSEUDOCODE):

STEP 1: START
STEP 2: Initialize the value of the GUI variables namely title, left, top, width, height to "Fahrer-Beigleiter: An alertness system for drivers",10,10,320,200 respectively.
STEP 3: Set the GUI window
STEP 4: Ask the user whether he want to start the system or not
STEP 5: IF (user wants to start the system) THEN,
STEP 5.1: Display "Yes Clicked"
STEP 5.2 Play sound to notify the user that the system has started
STEP 5.3: face_cascade ← the contents of the file 'face.xml'
STEP 5.4: left_eye_cascde ← the contents of the file 'leftEye.xml'
STEP 5.5: right_eye_cascade ← the contents of the file 'rightEye.xml'
STEP 5.6: IF (Either of the above mentioned 3 files is not found) THEN
STEP 5.6.1: Raise an IOError stating 'Unable to load the haarcascade file/s'
ENDIF
STEP 5.7: cap ← Capturing of the image from the selected camera
STEP 5.8: ds_factor ← 0.5
STEP 5.9: REPEAT
STEP 5.9.1: r,frame← Frame captured by the selected camera
STEP 5.9.2: frame ← Resize the frame captured
STEP 5.9.3: gray ← Convert the frame captured to grayscale
STEP 5.9.4: faces← Detect whether there is face in the captured image with help of the variable face_cascade
STEP 5.9.5: IF (no. of faces detected =0) THEN
STEP 5.9.5.1: Alert the driver that he is sleeping
ENDIF
STEP 5.9.6: DOWHILE (x, y, w, h <=length of faces)
STEP 5.9.6.1 Draw Rectangle around the faces
ENDDO

STEP 5.9.7: leftEye ← detect whether there is the left eye in the captured face
STEP 5.9.8: rightEye ← detect whether there is the right eye in the captured face
STEP 5.9.9: IF (length of leftEye and rightEye =0) THEN
STEP 5.9.9.1: Alert the driver that his eyes are closed for a long time
ENDIF
STEP 5.9.10: DOWHILE (x_eye,y_eye,w_eye,h_eye <= length of leftEye)
STEP 5.9.10.1: Draw Circle around the leftEye
ENDDO
STEP 5.9.11: DOWHILE (x_eye,y_eye,w_eye,h_eye <= length of rightEYE)
STEP 5.9.11.1: Draw circle around the right Eye
ENDDO

STEP 5.9.12: Show the modified captured frame to the user in a separate window on the monitor

STEP 5.9.13: c ← Value entered by the user

UNTIL (FALSE)

STEP 5.10: Release the window and the camera

STEP 5.11: Destroy all the windows

STEP 5.12: Display the message "\nThank You for using Fahrer Begleiter\nThis system is developed by Rahul Dhar"

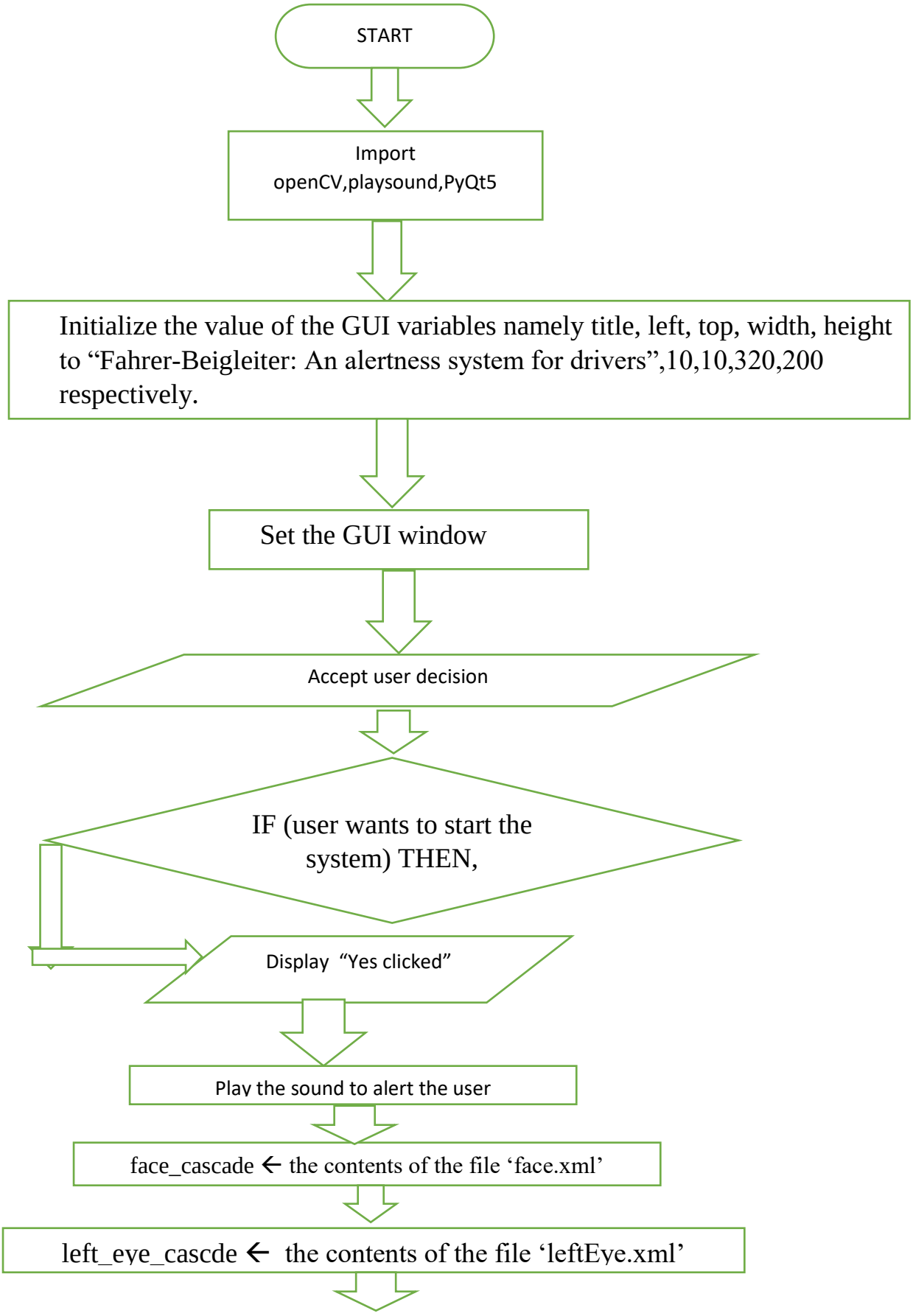
STEP 6: ELSE

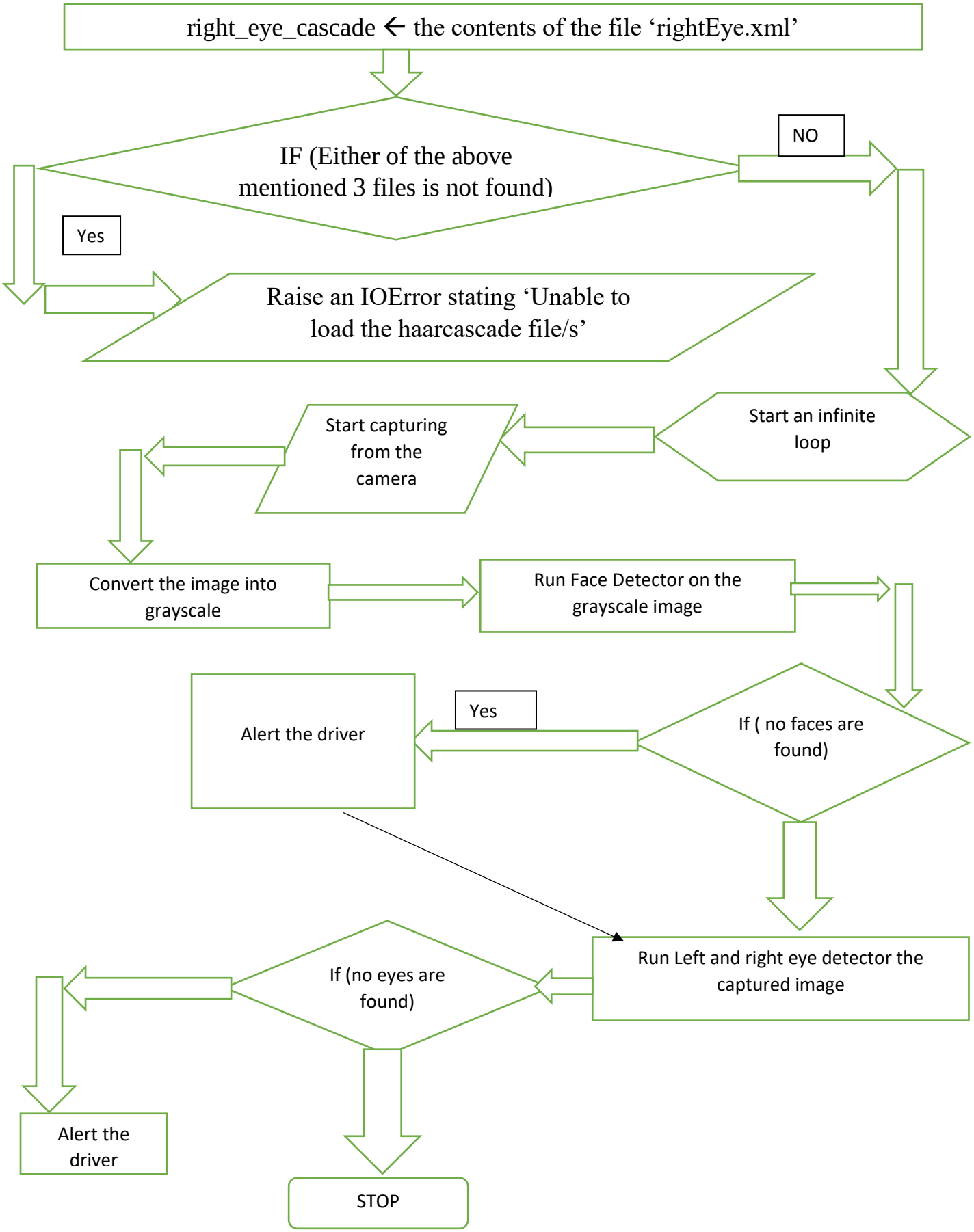
STEP 6.1 Exit from the system

ENDIF

STEP 7: STOP

3.2 FLOWCHART




right_eye_cascade ← the contents of the file 'rightEye.xml'
IF (Either of the above mentioned 3 files is not found)
NO
Yes
Raise an IOError stating 'Unable to load the haarcascade file/s'
Start an infinite loop
Start capturing from the camera
Convert the image into grayscale
Run Face Detector on the grayscale image
If (no faces are found)
Yes
Alert the driver
Run Left and right eye detector the captured image
If (no eyes are found)
Alert the driver
STOP

CHAPTER 4

4.1 REQUIREMENT ANALYSIS

The requirements for this system to run most effectively is:

- Python (version >=3.6)
- Python Libraries:
    1. openCv
    2. playsound
    3. PyQt5
- Camera Support
- Sound Support
- Platform (OS) used: Windows 7 (Minimum Requirement)
- Hard Disk: - 12KB
- Technology used: Artificial Intelligence using Python

CHAPTER 5

5.1 SOURCE CODE

```
'''
# AIM: - TO ALERT THE DRIVER WHEN HE IS FEELING FATIGUE OR TIRED
#PROGRAM DEVELOPED BY: RAHUL DHAR
#VERSION: - 1.0.2.10.07.2019
'''
# To import the required packages
import cv2
import playsound
import sys
import os
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QMessageBox
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import pyqtSlot

#To declare the class
class App(QWidget):
        def __init__(self):
                super().__init__()
                self.title = 'Fahrer-Beigleiter : An alertness system for drivers'#To set the title
                self.left = 10#To set the left border
                self.top = 10#To set the top border
                self.width = 320#To set the width
                self.height = 200#To set the height
                self.initUI()#To call the initUI() method

        def initUI(self):
                self.setWindowTitle(self.title)
                self.setGeometry(self.left, self.top, self.width, self.height)
                buttonReply = QMessageBox.question(self , 'Fahrer Beigleiter' , "Do you want to use Fahrer-
Beigleiter ?", QMessageBox.Yes | QMessageBox.No,QMessageBox.No)#To ask whether he wants to use the system
                if buttonReply==QMessageBox.Yes: #If the user clicks 'Yes'
                        print('Yes clicked')
                        playsound.playsound('service-bell.wav')
                        face_cascade=cv2.CascadeClassifier('face.xml')#To load the face haarcascade file
                        right_eye_cascade=cv2.CascadeClassifier('rightEye.xml')#To load the right-eye
haarcascade file
                        left_eye_cascade=cv2.CascadeClassifier('leftEye.xml')#To load the left-eye haarcascde
file
                        if face_cascade.empty() or right_eye_cascade.empty() or left_eye_cascade.empty():
                                raise IOError('Unable to load the required haarcascade file/s')#To raise an
error if the required haarcascade file isn't available
```

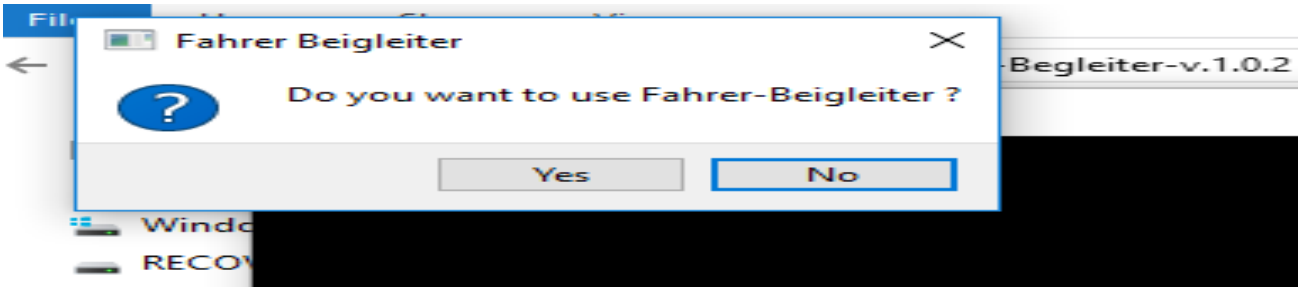
```
cap=cv2.VideoCapture(0)#To set the camera for capture
ds_factor=0.5#To set the scaling factor
while True:#To start an infinite loop
        r,frame=cap.read()#To start the capture from the camera
        frame=cv2.resize(frame, None, fx=ds_factor, fy=ds_factor,
interpolation=cv2.INTER_AREA)#To resize the frame
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)#To convert the image into grey
scale
        faces = face_cascade.detectMultiScale(gray, 1.3, 5)#To run the face detector
on the grayscale image
        if len(faces) == 0:#To check if no faces are found
                playsound.playsound('airhorn.wav')#To alert the driver when no faces
are found
        for (x,y,w,h) in faces:
                cv2.rectangle(frame, (x,y), (x+w,y+h), (210,255,210), 3)#To draw the
rectangle around the face
        roi_gray = gray[y:y+h, x:x+w]# Extract the gray face ROI
        roi_color = frame[y:y+h, x:x+w]# Extract the color face ROI
        leftEye = left_eye_cascade.detectMultiScale(roi_gray)#To detect the left eye
from the grayscale face
        rightEye=right_eye_cascade.detectMultiScale(roi_gray)#To detect the right eye
from the grayscale face
        if len(leftEye)==0 and len(rightEye)==0:#To check if both the eyes are closed
or not
                playsound.playsound('Smoke Alarm.wav')#To alert the driver if both of
the eyes are closed for a long time
        for (x_eye,y_eye,w_eye,h_eye) in leftEye:
                center = (int(x_eye + 0.5*w_eye), int(y_eye + 0.5*h_eye))
                radius = int(0.3 * (w_eye + h_eye))
                color = (0,255,0)
                thickness =  3
                cv2.circle(roi_color, center, radius, color, thickness)#To draw the
circles around the left eye
        for (x_eye,y_eye,w_eye,h_eye) in rightEye:
                center = (int(x_eye + 0.5*w_eye), int(y_eye + 0.5*h_eye))
                radius = int(0.3 * (w_eye + h_eye))
                color = (0,0,255)
                thickness =  3
                cv2.circle(roi_color, center, radius, color, thickness)#To draw the
circle around the right eye

        cv2.imshow(' Fahrer Begleiter--An alertness system for Drivers ',frame)#To
show the captured frame to the user
        c = cv2.waitKey(1)#To accept user's input to continue or not
        if c == 27:#To check if the 'esc' key is pressed
                break
cap.release()#To release the allocated memory space
cv2.destroyAllWindows()#To destroy all the windows
```

```
                        print("\nThank You for using Fahrer Begleiter\nThis system is developed by Rahul
Dhar")
                else:
                        exit()#To exit from the system
if __name__ == '__main__':
    app = QApplication(sys.argv)
    ex = App()
    sys.exit(app.exec_())
```

CHAPTER 6

6.1 RESULT

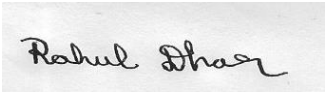


The GUI Interface

The Captured Frame

6.2 CONCLUSION

I, hereby would like to conclude the report, by stating that the system is very efficient. The loading time of the system is one second which is very first. Also, the system is very efficient since it can detect that slight moment of the driver and alert the driver very fast if he feels fatigue and doze off. It can be extremely great if it can be implemented in a real vehicle model.



RAHUL DHAR

-------XXXXXXXX----------