

**Pune Institute of Computer Technology
Dhankawadi, Pune**

**A MINI PROJECT REPORT
ON
DRIVER DROWSINESS DETECTION SYSTEM**

**SUBMITTED TOWARDS THE PARTIAL FULFILLMENT OF
THE REQUIREMENTS OF**

**THIRD YEAR SEMESTER II OF ENGINEERING
(Computer Engineering)
SUBMITTED BY**

Shubhankar Gaikwad Roll No. 31265

Shrut Shah Roll No. 31264

Shruti Rawate Roll No. 31279

Rasika Sonawane Roll No. 31267

**Under the guidance of
Dr. A. R. Buchade**



**DEPARTMENT OF COMPUTER ENGINEERING
Academic Year 2019-20**



DEPARTMENT OF COMPUTER ENGINEERING
Pune Institute of Computer Technology
Dhankawadi, Pune-43

CERTIFICATE

This is to certify that the Mini-Project report entitled
“DRIVER DROWSINESS DETECTION SYSTEM”

Submitted by

Shubhankar Gaikwad Roll No. 31265

Shrut Shah Roll No. 31264

Shruti Rawate Roll No. 31279

Rasika Sonawane Roll No. 31267

is a bonafide work carried out by students under the guidance of
Dr. A. R. Buchade towards the partial fulfillment of third year
Computer Engineering Semester II, Academic Year 2019-20 of
Savitribai Phule Pune University.

Dr. A. R. Buchade
Internal Guide

Prof. M.S.Takalikar
Head
Department of Computer Engineering

Place:
Date:

ACKNOWLEDGEMENT

I sincerely thank our Mini Project Guide Dr. A. R. Buchade and Head of Department Prof. M.S.Takalikar for their support.

I also sincerely convey my gratitude to Dr. R. B. Ingle, Department of Computer Engineering for his constant support, providing all the help, motivation and encouragement from beginning till end to make this project a grand success.

Contents

1	INTRODUCTION	1
2	MOTIVATION	2
3	PROBLEM DEFINITION AND SCOPE	3
3.1	Problem Definition	3
3.2	Scope	3
4	SYSTEM DESIGN	4
4.1	Working of System	4
5	MODULE INFORMATION	5
5.1	Components-	5
6	IOT DESIGN METHODOLOGY STEPS	6
6.1	Step 1: Purpose and Requirement Specification	6
6.2	Step 2: Process Specification	7
6.3	Step 3: Domain Model Specification	7
6.4	Step 4: Information Model Specification	8
6.5	Step 5: Service Specification	9
6.6	Step 6: IoT Level Specification	9
6.7	Step 7: Functional View Specification	10
6.8	Step 8: Operational View Specification	11
6.9	Step 9: Device and Component Integration	11
6.10	Step 10: App development	11
7	CIRCUIT DIAGRAM	13
8	CONCLUSION	14
8.1	Conclusion	14
8.2	Future Scope	14
9	References	15
	References	15
10	SCREENSHOTS	16
11	CODE	19

List of Tables

List of Figures

1	Raspberry Pi 3	4
2	Camera Module	5
3	IoT Design Steps	6
4	Process Specification	7
5	Information model	8
6	IoT Level 1	9
7	Functional View	10
8	Device	11
9	Tkinter App	12
10	Report Generation	12
11	Ckt	13
12	Working	13
13	Registration	16
14	Authenticated Login	16
15	Alert	17
16	Report Mailed	17
17	Local storage and analysis	18
18	Code snap 1	19
19	Code snap 2	19

Abstract

With the development of Web 3.0 and M2M communications, there has been a huge shift towards automating the devices. Web of Things and IoT have emerged as a result to provide smart solutions to day to day problems.

One such issue where IoT is being implemented is the automobile sector. IoT sensors can be used to calculate many real time measures of the automobiles. Geo-tracking, speed sensors, security systems, remote controls etc are some applications.

A major issue which causes accidents is driver drowsiness. IoT approach can be used to create smart detection systems to prevent hazards caused by driver drowsiness. We have built a level-1 IoT system for this purpose using Raspberry Pi and OpenCV

Keywords

Raspberry Pi, OpenCV, Drowsiness Detection System, IoT, Yawn Detection

1 INTRODUCTION

With the development of Web 3.0 and M2M communications, development of smart devices has been on the boom. Automobile sector has been an important shareholder in this information and embedded systems revolution. With the development of smart embedded devices for cars to the manufacturing of driverless cars like Tesla, IoT devices have many applications in this sector.

In today's world, drivers are an important part of the economy of a system. From driving cars to airplanes, trucks delivering gasoline or driving personally, safety of the driver is an important issue. Given the amount of traffic increasing day by day, it becomes even more crucial to adhere to the safety of the drivers. Usually during long journeys or when one is tired, one may tend to fall asleep. This is a major cause many road accidents. So it is essential to create a system that alerts the drivers when they are about to fall asleep.

The proposed driver drowsiness and yawn detection system is one such solution to partially solve the issue and make the system smart. Using machine learning and OpenCV, video analysis can be done to note the eye and mouth readings.

Threshold levels are set and beep alerts are sent to the driver when he closes his eyes or yawns. This system also sends email alerts and daily report logs. Raspberry Pi is used with camera modules for the working project.

2 MOTIVATION

In our country there are a lot of accidents happening on the road due the miscalculations and faults of drivers. In a country where public transports is in the heart of the economy, safety cant be compromised. Thus this mini project is an attempt to embed IoT devices into automobiles and use them to warn the drivers for drowsiness and yawns.

3 PROBLEM DEFINITION AND SCOPE

3.1 Problem Definition

To develop an effective system to send alerts to drivers for yawn and drowsiness using OpenCV and RaspberryPi

3.2 Scope

The project model can be used in personal vehicles or commercial transportation automobiles for the safety purpose. System can be customized to specific alert messages and alarms as needed.

4 SYSTEM DESIGN

4.1 Working of System

- Start the tkinter app on machine/ mobile.
- User registration and login
- Start recording the driver using webcam/ camera module.
- Alarm on yawn or drowsiness detection OpenCV image processing.
- Save the data to SQLite Database.
- Send email for specified date.



Figure 1: Raspberry Pi 3

5 MODULE INFORMATION

5.1 Components-

- Raspberry Pi- Raspbian OS
- Python App - tkinter
- Camera Module / Web cam.
- SQLite Database for local analysis.
- Report generation in app.
- Email facility to report based on date.
- Login and registration.



Figure 2: Camera Module

6 IOT DESIGN METHODOLOGY STEPS

IoT Design Methodology - Steps



Figure 3: IoT Design Steps

6.1 Step 1: Purpose and Requirement Specification

- To send alerts to drivers for drowsiness and yawn.
- Automatic behavior detection using video processing.
- Send alerts and detailed reports.
- Local data storage and analysis.
- Local python application with authentication feature.

6.2 Step 2: Process Specification

Use case-

- If eyelid distance less than threshold, send drowsiness alert.
- If mouth is wide open than threshold, send yawn alert.

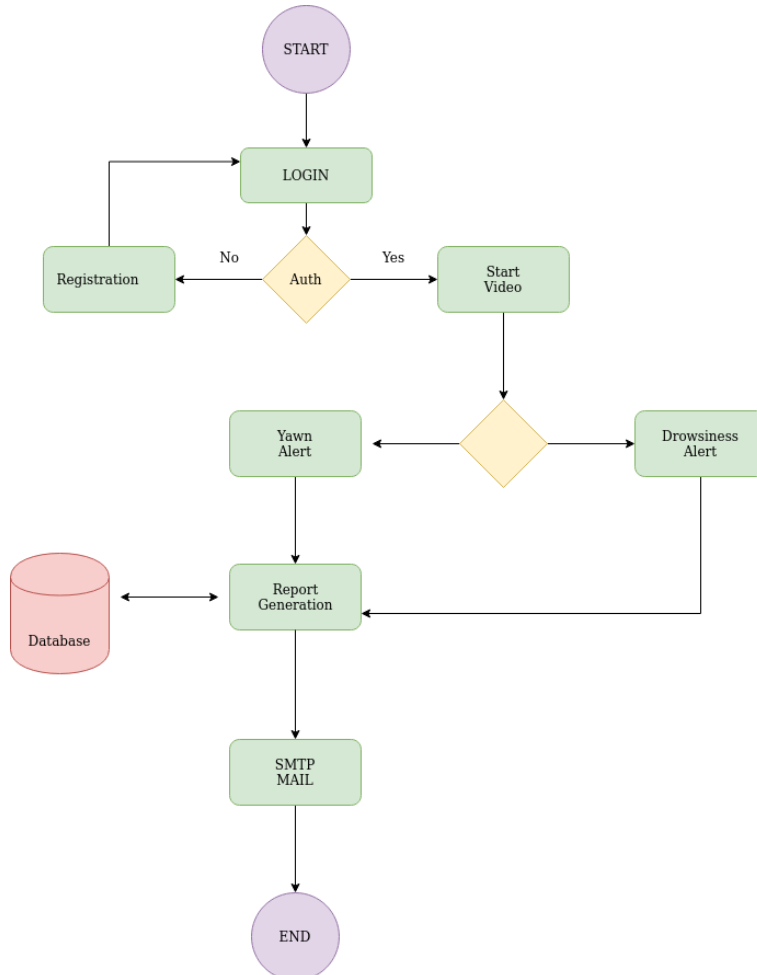


Figure 4: Process Specification

6.3 Step 3: Domain Model Specification

- Physical entity- Driver, Car, Detection System
- Virtual entity- Capture of real time video.
- Device- Raspberry Pi, Camera module, phone/PC.
- Resource- Python libraries, OpenCV .
- Service- Python app for interaction, SMTP mailing service, local information storage.

6.4 Step 4: Information Model Specification

Structure /attributes

- Physical entity- Driver, Car, Detection System
- Virtual entity- Capture of real time video.
- Device- Raspberry Pi, Camera module, phone/PC.
- Resource- Python libraries, OpenCV .
- Service- Python app for interaction, smtp mailing service, local information storage.

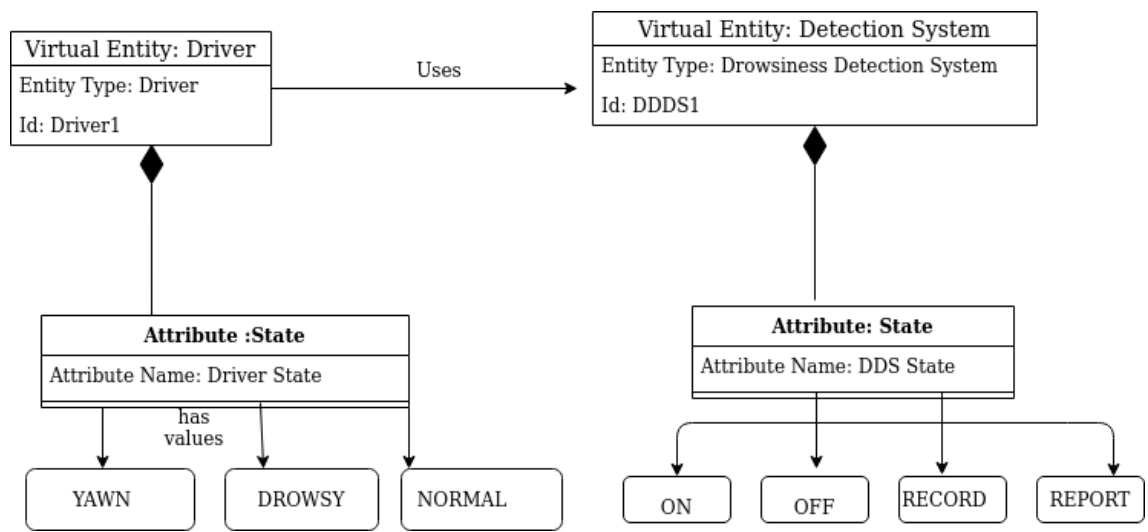


Figure 5: Information model

6.5 Step 5: Service Specification

- The camera module is used to record real time video of driver and analyse for drowsiness.
- Alert if eyes are closed or yawns.
- Mail service based on user input of date.

6.6 Step 6: IoT Level Specification

IoT Level 1 deployment.

- Local storage and analysis of data- SQLite
- Local application for control- tkinter python app.
- Single device node and controller.
- Remote access to RaspberryPi using phone/PC over WiFi

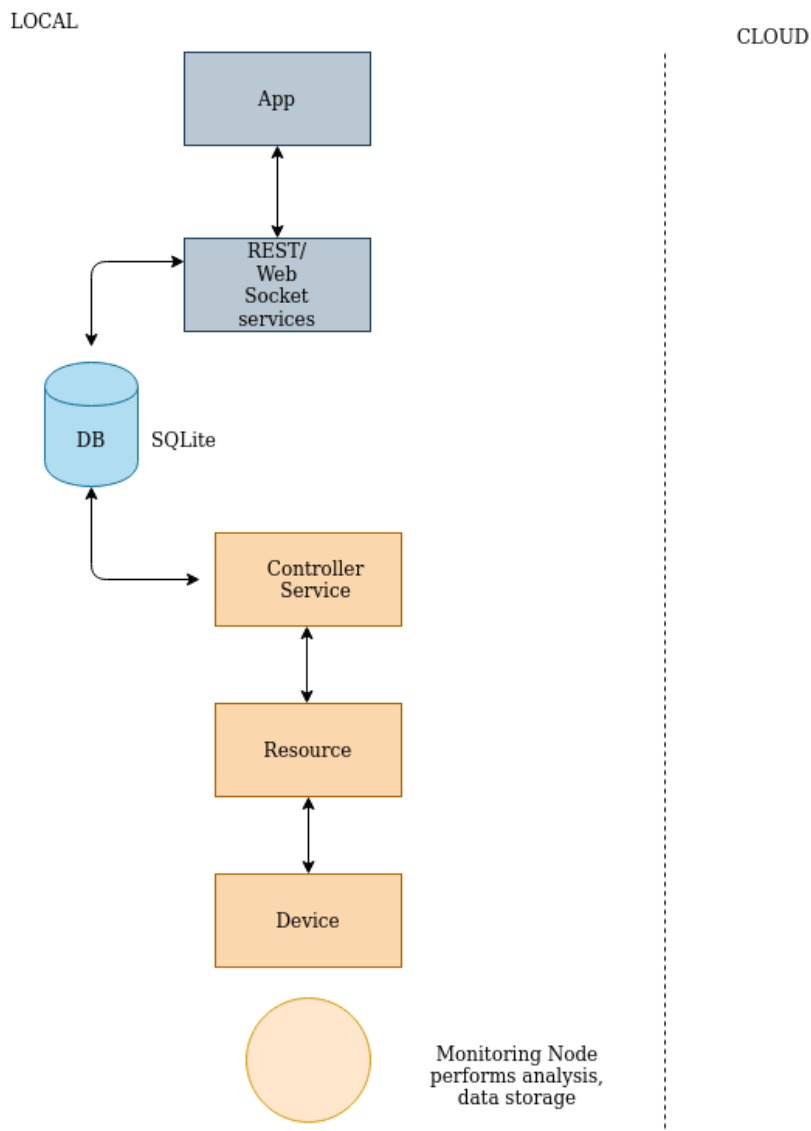


Figure 6: IoT Level 1

6.7 Step 7: Functional View Specification

- Driver starts the app.
- New user registration or login authentication.
- Start the video capturing through app and camera module.
- Send email based on date provided by user.
- Report generation using local data analysis from SQLite.

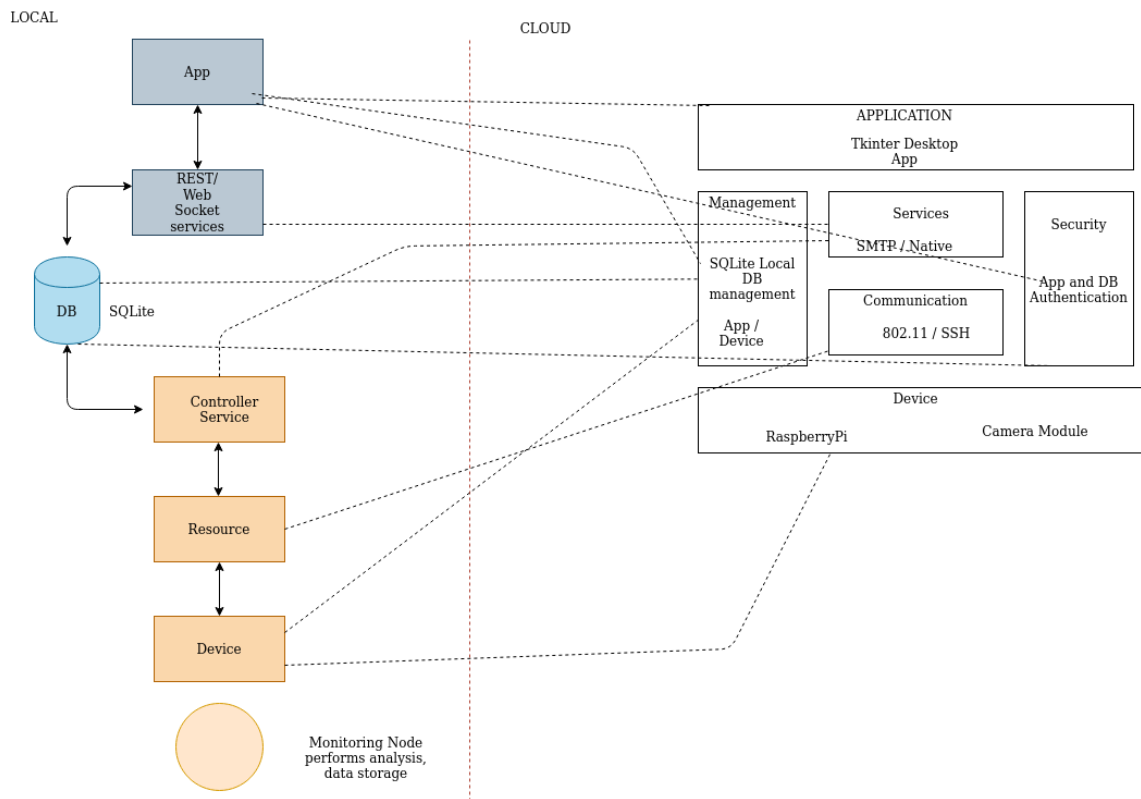


Figure 7: Functional View

6.8 Step 8: Operational View Specification

- App deployment using Python tkinter.
- Database - SQLite local storage.
- SMTP mailing servie.
- User authorization and registration.
- 802.11 WiFi and SSH to control device remotely from phone or PC.

6.9 Step 9: Device and Component Integration

- The camera module is used to record real time video of driver and analyse for drowsiness.
- Alert if eyes are closed or yawns.
- Mail service based on user input of date.



Figure 8: Device

6.10 Step 10: App development

IoT Level 1 deployment.

- Local storage and analysis of date- SQLite
- Local application for control- tkinter python app.
- Single device node and controller.

- Remote access to Raspberry Pi using phone/PC over WiFi



Figure 9: Tkinter App

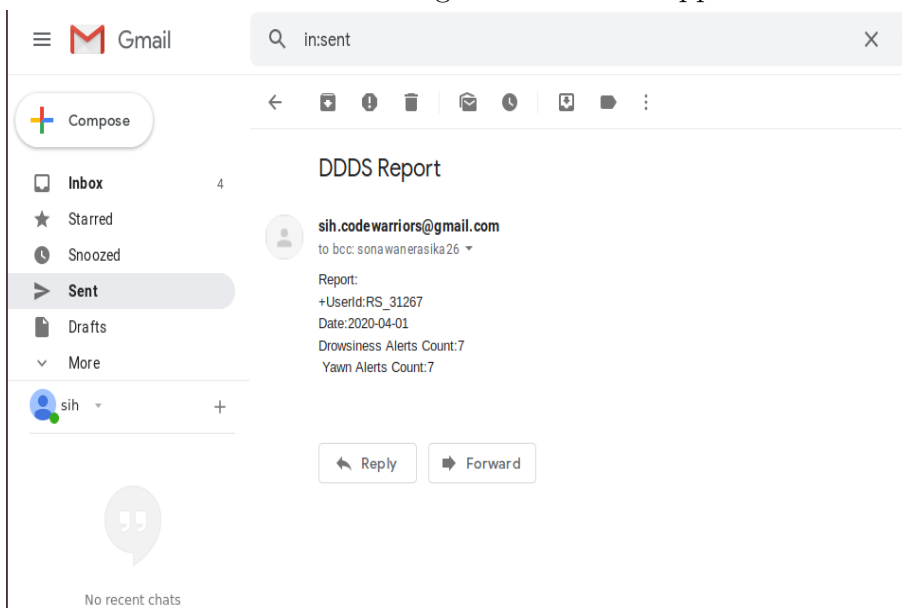


Figure 10: Report Generation

7 CIRCUIT DIAGRAM

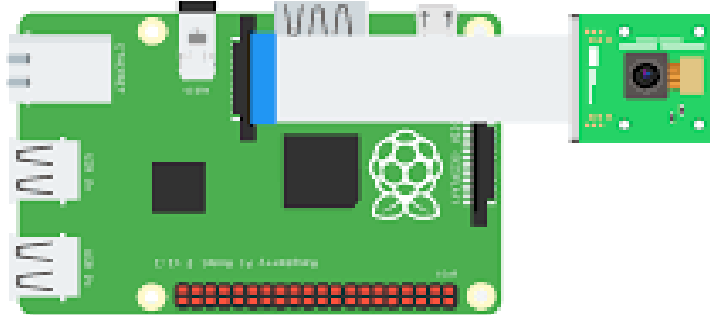


Figure 11: Ckt

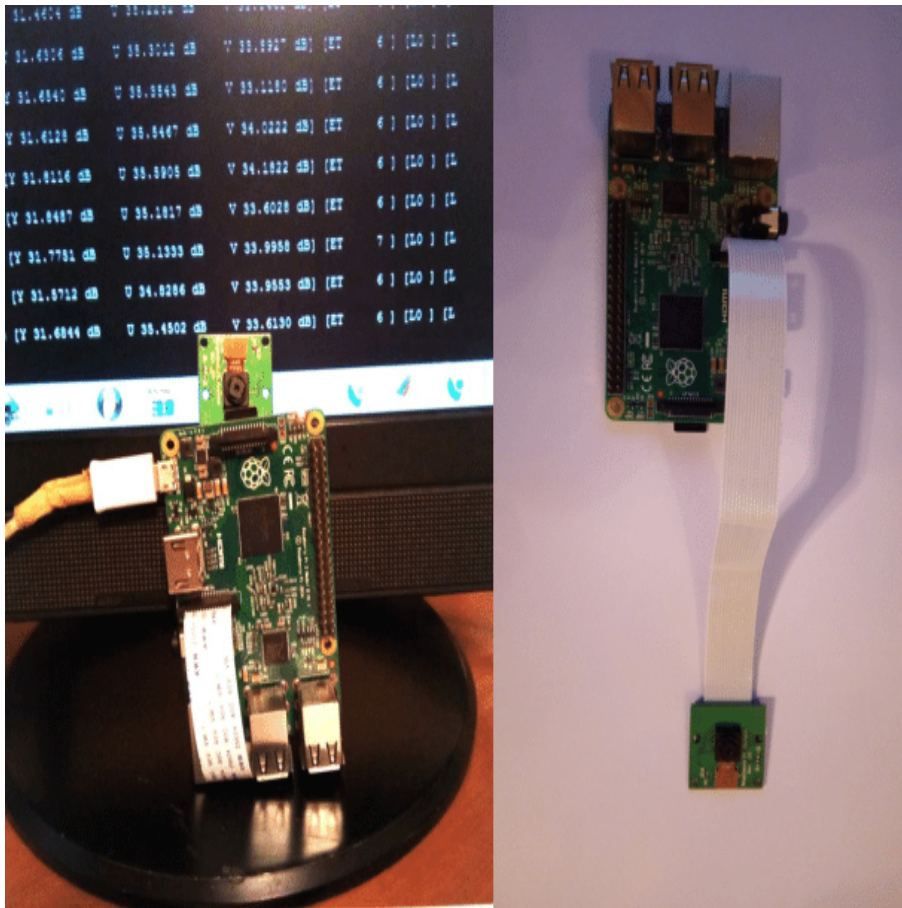


Figure 12: Working

8 CONCLUSION

8.1 Conclusion

Implemented the driver drowsiness detection system using RaspberryPi and camera module. OpenCV was used to analyse videos and python app to send messages.

8.2 Future Scope

Can be integrated with speed, lane change detection modules to make a wholesome automobile smart device integration. Basic framework can be used to monitor drowsiness during meetings, classes etc.

9 References

References

- [1] Raspberry Pi Projects *<https://projects.raspberrypi.org/en>*
- [2] OpenCV documentations. *<https://opencv.org/>*
- [3] Tkinter documentation. *<https://docs.python.org/2/library/tkinter.html>*
- [4] Driver Drowsiness Detection System using OpenCV-PyImage
<https://www.pyimagesearch.com/2017/05/08/drowsiness-detection-opencv/>

10 SCREENSHOTS



The screenshot shows a web application window titled "Driver Drowsiness Detection System". It has a menu bar with "Tools" and "About". The main content area has a light blue background and a central "Registration form" section. The form contains six input fields: "FullName" (Rasika Sonawane), "UserName" (RS_31267), "Email" (nerasika26@gmail.com), "Phone" (7559107706), and "Password" (masked with asterisks). Below the fields are two buttons: a grey "Submit" button and a purple "Go back to start page" button.

Figure 13: Registration



The screenshot shows the same web application window. The main content area has a dark blue background with a glowing blue geometric pattern. At the top, the text "DRIVER DROWSINESS DETECTION SYSTEM" is displayed in large, bold, black capital letters. Below this, there are two input fields: "UserName" (RS_31267) and "Password" (masked with asterisks). At the bottom, there are two buttons: a grey "Register" button and a grey "Login" button.

Figure 14: Authenticated Login

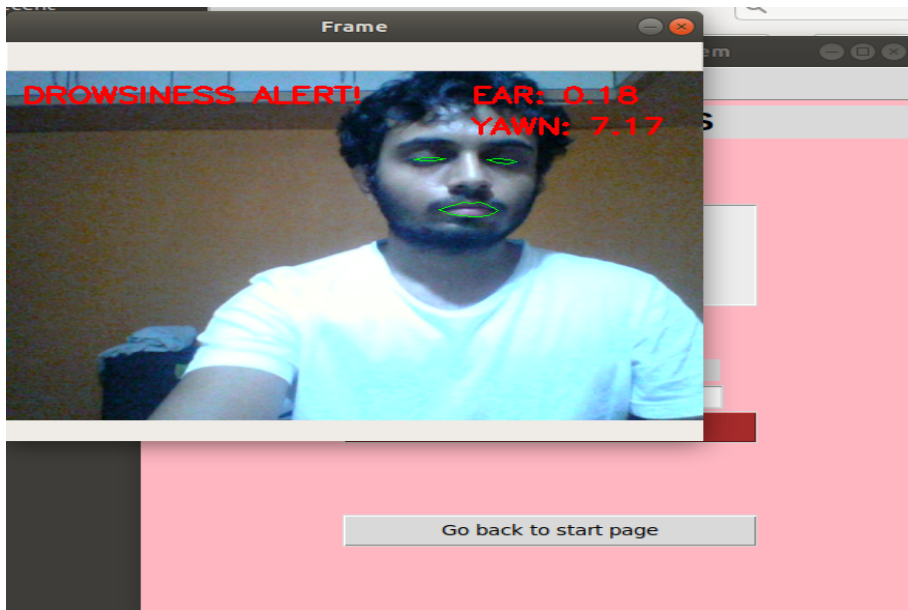


Figure 15: Alert

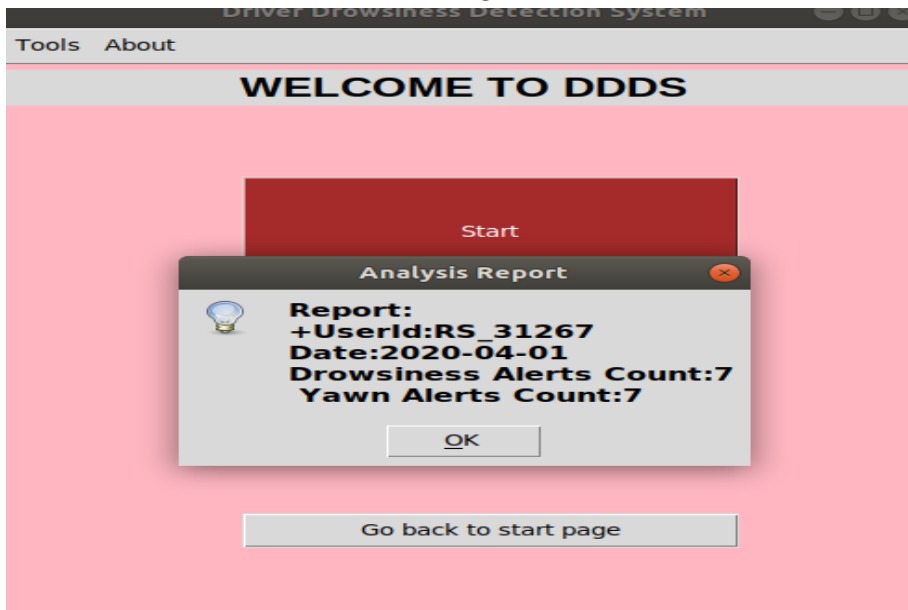


Figure 16: Report Mailed

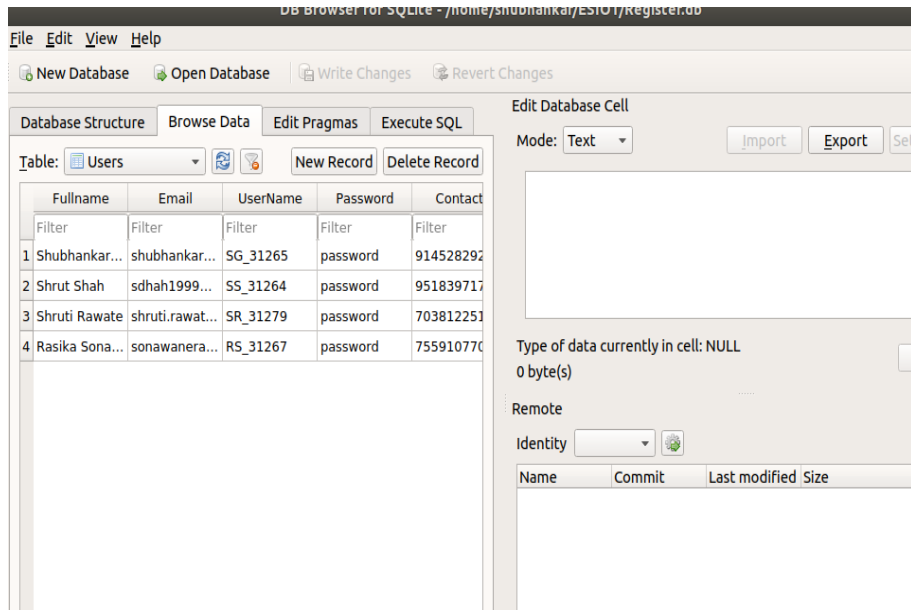
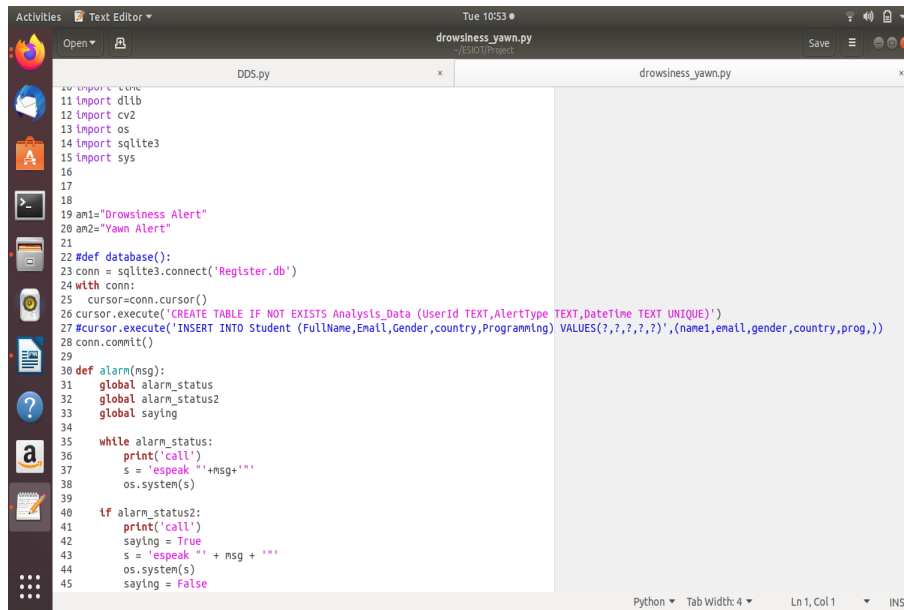


Figure 17: Local storage and analysis

11 CODE

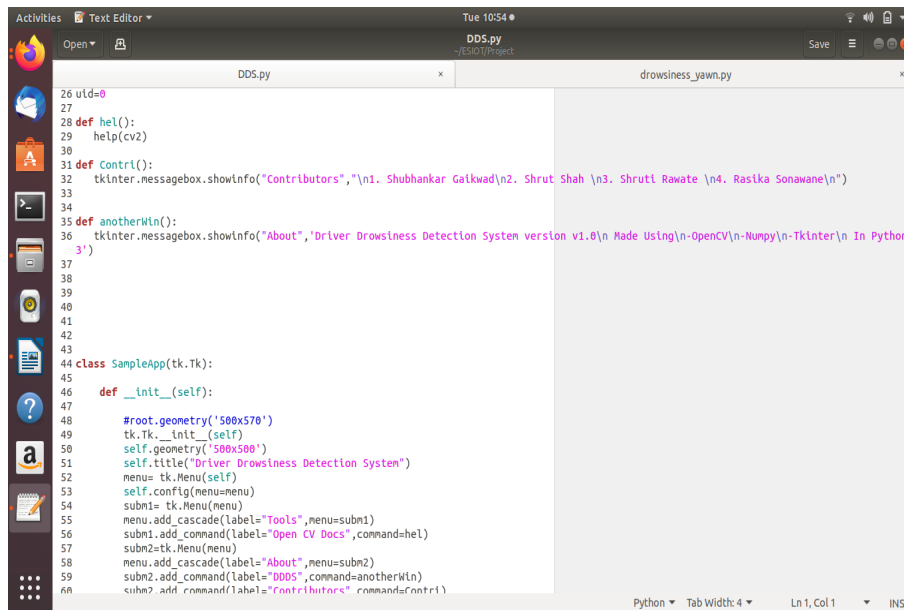


```

20 import cv2
21 import dlib
22 import cv2
23 import os
24 import sqlite3
25 import sys
26
27
28 am1="Drowsiness Alert"
29 am2="Yawn Alert"
30
31 #def database():
32 conn = sqlite3.connect('Register.db')
33 with conn:
34     cursor=conn.cursor()
35     cursor.execute('CREATE TABLE IF NOT EXISTS Analysis_Data (UserID TEXT,Alerttype TEXT,dateTime TEXT UNIQUE)')
36 #cursor.execute('INSERT INTO Student (FullName,Enail,Gender,country,Programming) VALUES(?,?,?,?),(name1,enail,gender,country,prog,))
37 conn.commit()
38
39 def alarm(msg):
40     global alarm_status
41     global alarm_status2
42     global saying
43
44     while alarm_status:
45         print('call')
46         s = 'espeak '+msg+' '
47         os.system(s)
48
49     if alarm_status2:
50         print('call')
51         saying = True
52         s = 'espeak ' + msg + ' '
53         os.system(s)
54         saying = False

```

Figure 18: Code snap 1



```

26 uid=0
27
28 def hel():
29     help(cv2)
30
31 def Contri():
32     tkinter.messagebox.showinfo("Contributors", "\n1. Shubhankar Gaikwad\n2. Shrut Shah \n3. Shruti Rawate \n4. Rasika Sonawane\n")
33
34
35 def anotherWin():
36     tkinter.messagebox.showinfo("About", "Driver Drowsiness Detection System version v1.0\n Made Using\n-OpenCV\n-Numpy\n-Tkinter\n In Python
37 ')
38
39
40
41
42
43
44 class SampleApp(tk.Tk):
45     def __init__(self):
46         #root.geometry('500x570')
47         tk.Tk.__init__(self)
48         self.geometry('500x500')
49         self.title("Driver Drowsiness Detection System")
50         menu= tk.Menu(self)
51         self.config(menu=menu)
52         subm1= tk.Menu(menu)
53         menu.add_cascade(label="Tools", menu=subm1)
54         subm1.add_command(label="Open CV Docs", command=hel)
55         subm2=tk.Menu(menu)
56         menu.add_cascade(label="About", menu=subm2)
57         subm2.add_command(label="DDDS", command=anotherWin)
58         subm2.add_command(label="Contributors" command=Contri)

```

Figure 19: Code snap 2