CHAPTER 1
INTRODUCTION

1.1 INDRODUCTION

In this high-tech world, it's almost impossible to imagine life without vehicles. The invention of vehicles is one of the greatest human kind's inventions. vehicles have become an essential part of almost every day use for individuals of every age. In daily life, we interact and use many times with different vehicles to make our work easier. Thus, for the safety of the driver or owner the "Driver Drowsiness Detection System" has become a hot topic for research.

Drowsiness detection is a safety technology that can prevent accidents that are caused by drivers who fell asleep while driving. This system will alert the driver when drowsiness is detected. Through its driver Availability Detection System, sensors will scan the head and face to ensure that the eyes are open and the driver is alert before the cars turns over the steering wheel. If drowsiness is detected, the driver is alerted to the nearest rest stop.

In this project, drowsiness detection application will be designed and implemented using a regular webcam. To implement this,[10] we will be using OpenCV module of Python. Though there are several methods for measuring the drowsiness but this approach is completely non-intrusive which does not affect the driver in any way, hence giving the exact condition of the driver. For detection of drowsiness the per closure value of eye is considered. So when the closure of eye exceeds a certain amount then the driver is identified to be sleepy. For implementing this system several OpenCv libraries are used including Haar-cascade. The entire system is implemented using Arduino.

1.2 MOTIVATION

Driver drowsiness is the significant factor in the increasing number of accidents on today's roads and has been extensively accepted. This proof has been verified by many researches that have demonstrated ties between driver drowsiness and road accidents. Although it is hard to decide the exact number of accidents due to drowsiness, it is much likely to be underestimated. The above statement shows the significance of a research with the objective of reducing the danger of accidents anticipated to drowsiness.

1.3 OBJECTIVE

The purpose of the drowsiness detection system is to aid in the prevention of accidents passenger and commercial vehicles. [9]The system will detect the early symptoms of drowsiness before the driver has fully lost all attentiveness and warn the driver that they are no longer capable of operating the vehicle safely.

- Driver drowsiness detection is a car safety technology which helps to save the life of the driver by preventing accidents when the driver is getting drowsy.
- The main objective is to first design a system to detect driver's drowsiness by continuously monitoring retina of the eye.
- The system works in spite of driver wearing spectacles and in various lighting conditions
- To alert the driver on the detection of drowsiness by using buzzer or alarm.
- Speed of the vehicle can be reduced

1.4 FACT AND STATISTICS

Our current statistics reveal that just in 2015 in India alone, 148,707 people died due to car related accidents. Of these, at least 21 percent were caused due to fatigue causing drivers to make mistakes. This can be a relatively smaller number still, as among the multiple causes that can lead to an accident, the involvement of fatigue as a cause is generally grossly underestimated. Fatigue combined with bad infrastructure in developing countries like India is a recipe for disaster. Fatigue, in general, is very difficult to measure or observe unlike alcohol and drugs, which have clear key indicators and tests that are available easily.

Probably, the best solutions to this problem are awareness about fatigue-related accidents and promoting drivers to admits fatigue when needed. The former is hard and much more expensive to achieve, and the latter is not possible without the former as driving for long hours is very lucrative. When there is an increased need for a job, the wages associated with it increases leading to more and more people adopting it. Such is the case for driving transport vehicles at night. Money motivates drivers to make unwise decisions like driving all night even with fatigue. [2]This is mainly because the drivers are not themselves aware of the huge risk associated with driving when fatigued. Some countries have imposed restrictions on the number of hours a driver can drive at a stretch, but it is still not enough to solve this problem as its implementation is very difficult and costly.

1.5 PROBLEM STATEMENT

Figure 1.1: Problem Statement

Driver drowsiness contributes to a substantial number of fatal and nonfatal crashes. Not[1] all peoples can afford this system as it is high tech technology and car companies won't allow to install in cheaper cars. To design a model that can detect and recognize drowsiness based on facial expression and prevent further fatal accidents.

CHAPTER 2
PROJECT SCOPE

2.1 PROJECT SCOPE

There are many products out there that provide the measure of fatigue level in the drivers which are implemented in many vehicles. The driver drowsiness detection system provides the similar functionality but with better results and additional benefits. Also, it alerts the user on reaching a certain saturation point of the drowsiness measure

2.2 ADVANTAGES

- Accident victim's death can be prevented
- Easy to implement
- Small Form Factor with separately working components
- Any ONVIF and RTSP supported camera can be used
- Practically applicable and User-friendly Interface
- Low cost of equipment

2.3 APPLICATIONS

- Transportation vehicles
- Public transport
- Private vehicles
- Railways
- Industrial Monitoring
- To keep an eye on security guards

CHAPTER 3
LITERATURE REVIEW

3.1 SURVEY OF EXISTING SYSTEM

[7]Existing detection system uses iris sensor for the drowsiness drive an occasional cost and easy distributed sensors model that particularly suitable for measuring blink of the drive accident and hand position on a wheel .[3]These sensors are often utilized in automotive active safety system that aim at detection driver fatigue a serious issue to stop road accident the key point of this approach is to style a prototype of sensor units in order that it can function platform for integration different sorts of sensors into the wheel. The wheel is slowed or stopped counting on the condition. An identical existing system uses the adaptive driver for the detection of the drowsiness truck driver company car drivers and shift workers are the foremost in danger of falling asleep while driving majority of the accident occurs thanks to the drunkenness of the driving force. The misfortune of the which lies on the corporate owner as they made liable it can cause economic loss during this presentation, we present an adaptive driver and company owner alert system and an application that gives driving behaviour to the corporate owner. [4]This survey is completed to understand the requirement and prerequisite of the overall population, and to try to per se to, we went through different sites and applications and hunted for the basic data. Based on these data, an audit is carried out which helps us to get new thoughts and make different arrangements for our task. progress in this field too

3.2 LIMITATIONS OF EXISTING SYSTEM

Drowsy driving is one of the major causes that lead to fatal accidents worldwide. For the past two decades,[5] many studies have explored the feasibility and practicality of drowsiness detection using electroencephalogram (EEG) based brain-computer interface (BCI) systems. However, on the pathway of transitioning laboratory-oriented BCI into real-world environments, one chief challenge is to obtain high-quality EEG with convenience and long-term wearing comfort. Recently, acquiring EEG from non-hair-bearing (NHB) scalp areas has been proposed as an alternative solution to avoid

many of the technical limitations resulted from the interference of hair between electrodes and the skin.

[6]Another limitation of it is that it is purely dependent on external factors like road marking, climatic and lightning conditions. In summary, many studies have determined that vehicle-based measures are a poor predictor of performance error risk due to drowsiness.

3.3 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.

3.4 TECHNOLOGIES USED

a. Python - 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.

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

c. Deep Learning-Deep learning is a type of machine learning and artificial intelligence (AI) that imitates the way humans gain certain types of knowledge. Deep learning is an important element of data science, which includes statistics and predictive modelling.

CHAPTER 4
SOFTWARE&HARDWARE SPECIFICATION

4.1 SOFTWARE REQUIREDMENTS SPECIFICATION

1.Python:

- Python 3

2. Libraries:

- NumPy
- Dlib
- OpenCV

3.Operating System:

- Windows

4.2 HARDWARE REQUIREMENTS SPECIFICATION

- Laptop with basic hardware or Raspberry Pi
- Webcam
- IP Camera with RTSP Support
- Arduino
- Buzzer
- Breadboard and Breadboard Power Supply

4.2.1 Laptop with basic hardware or Raspberry Pi:

- It is a low cost, credit-card sized computer which is used for implementing small projects. A monitor or TV has to be connected with it externally to visualize its operating system and operate it. We can use a key board and a mouse to provide input to it. An external memory has to be used to load its operating system. We can program it with several languages like C++, Python etc.
- Its components include the following: (1) 700 MHz processor. (2) 512 MB RAM. (3) USB ports for external devices. (4) Micro SD card slots. (5) Ethernet port. (6) HDMI port. (7) 40 GPIO pins. (8) Camera interface. (9) Display interface. (10) Power supply port.
- The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

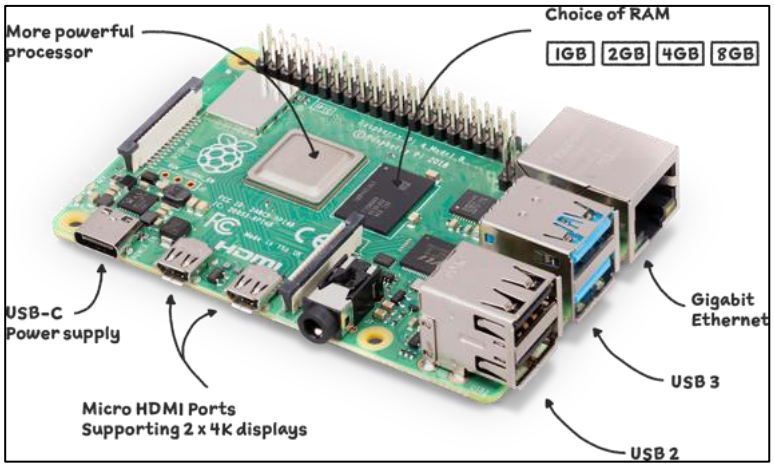


Figure 4.1: RasberryPi

4.2.2 Web Camera

- Utilizing a web camera introduced inside the automobile we can get the picture of the driver. Despite the fact that the camera creates a video clip, we have to apply the developed algorithm on each edge of the video stream.
- This paper is only focused on the applying the proposed mechanism only on single frame. The used camera is a low-cost web camera with a frame rate of 30 fps in VGA mode. Logitech Camera is used for this process

Figure 4.2: Web Camera

4.2.3 IP Camera with RTSP

RTSP stands for Real Time Streaming Protocol. RTSP allows you to pull a live video stream from your camera and view it from different devices and programs. Its primary uses are to pull a video feed from a camera to an NVR, viewing software, or even home automation solutions.

4.2.4 Arduino

Arduino UNO is a low-cost, flexible, and easy-to-use programmable open-source microcontroller board that can be integrated into a variety of electronic projects. This board can be interfaced with other Arduino boards, Arduino shields, Raspberry Pi boards and can control relays, LEDs, servos, and motors as an output.

Arduino Uno is known as an open-source development board as it allows you to use this board to interact with real-world things by uploading programs on this board. It is a microcontroller board developed by Arduino.cc and is based on the Atmega328 Microcontroller.

The first Arduino project was started in Interaction Design Institute Ivrea in 2003 by David Cuartillas and Massimo Bansi with the intention of providing a cheap and flexible way for students and professionals to learn embedded programming. Arduino UNO is a very valuable addition in electronics that consists of a USB interface, 14 digital I/O pins(of which 6 Pins are used for PWM), 6 analog pins and an Atmega328 microcontroller. It also supports 3 communication protocols named Serial, I2C and SPI protocol. It can interact with anything that is controlled by electricity in any way. It can also interact with motors, sensors, and electromagnets

a. FEATURES:

- Microcontroller: ATmega328
- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limits): 6-20V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 40 mA

- DC Current for 3.3V Pin: 50 mA
- Flash Memory: 32 KB of which 0.5 KB used by bootloader
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Clock Speed: 16 MHz

b. ARDUINO UNO BOARD DESCRIPTION:

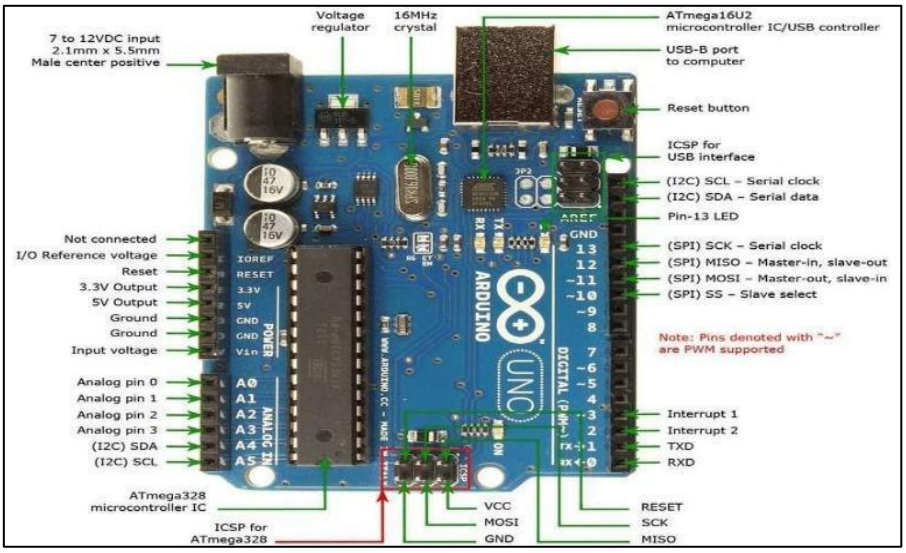


Figure 4.3: Arduino UNO Board

Arduino UNO board because it is the most popular board in the Arduino board family. In addition, it is the best board to get started with electronics and coding. Some boards look a bit different from the one given below, but most Arduinos have the majority of these components in common.

- **Power USB:** Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection.

- **Power (Barrel Jack):** Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack.

- **Voltage Regulator:** The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

- **Crystal Oscillator:** The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.

- **Arduino Reset:** You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET.

- **Pin (3.3, 5, GND, Vin):**

1) 3.3V − Supply 3.3 output volt

2) 5V − Supply 5 output volt

3) Most of the components used with Arduino boards work fine with 3.3 volt and 5volt.

4) GND (Ground) − There are several GND pins on the Arduino, any of which can be used to ground your circuit.

5) Vin − This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

- **Analog Pins:** The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.

● **Microcontroller Pins:** Each Arduino board has its own microcontroller. You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.

● **ICSP Pin:** Mostly, ICSP is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.

● **Power LED Indicator:** This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

● **TX and RX LEDs:** On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led. The TX led flashes with different speeds while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

● **Digital I/O:** The Arduino UNO board has 14 digital I/O pins (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labelled “~” can be used to generate PWM.

● **AREF:** AREF stands for Analog Reference. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

c. APPLICATIONS

The Arduino boards can work as a stand-alone project and can be interfaced with other Arduino boards or Raspberry Pi boards. Arduino UNO board is used in the following applications.

1. Weighing Machines
2. Traffic Light Countdown Timer
3. Parking Lot Counter
4. Embedded systems
5. Home Automation
6. Industrial Automation
7. Medical Instrument
8. Emergency Light for Railways

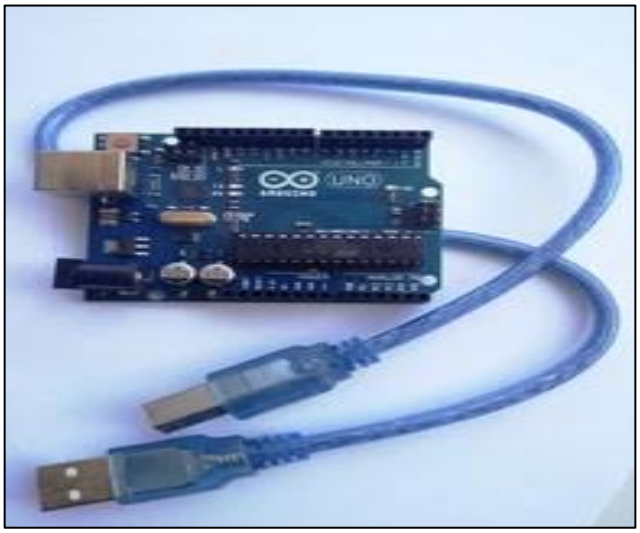


Figure 4.4: Arduino Kit

5 Buzzer

- The buzzer consists of an outside case with two pins to attach it to power and ground. When current is applied to the buzzer it causes the ceramic disk to contract or expand.
- Changing then this causes the surrounding disc to vibrate. That's the sound that you hear

Figure 4.5: Buzzer

CHAPTER 5
REQUIREMENTS ANALYSIS

5.1 REQUIREMENT ANALYSIS

A. PYTHON:

Python is the basis of the program that we wrote. It utilizes many of the python libraries. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective**.**

B. FRAMEWORK:

OpenCV:

OpenCV was started at Intel in 1999 by Gary Bradsky, and the first release came out in 2000. Vadim Pisarevsky joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with Gary Bradsky and Vadim Pisarevsky leading the project. OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day.

OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development. OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language OpenCV is a popular and open-source computer vision library that is focussed on real-time applications. The library has a modular structure and includes several hundreds of computer vision algorithms.

OpenCV includes a number of modules including image processing, video analysis, 2D feature framework, object detection, camera calibration, 3D reconstruction and more. We will be using OpenCV to capture the live video stream from webcam and to use it in the program. OpenCV means Open-Source Computer Vision Library. It is an open-source library used for computer vision and machine learning functions. OpenCV is written in C++ and has cross platform support. Its C++, Python, Java and MATLAB interfaces support Windows, Linux, Android, iOS and Mac OS. OpenCV was developed to provide a common tool for developing computer vision applications and to increase the use of machine perception in the commercial products. OpenCV aims towards real-time vision applications.

C. LIBRARIES:

1. Dlib: The dlib C++ library is a cross-platform package for threading, networking, numerical operations, machine learning, computer vision, and compression, placing a strong emphasis on extremely high-quality and portable code. For medium to large image sizes Dlib is the fastest method on CPU. But it does not detect small sized factors. So, if you know that your applications will not be dealing with very small sized faces then the HoG based Face detector is the better option. Dlib is a general purpose open-source software library and has cross platform support. It is a tool of C++ which consists of machine learning algorithms and several tools for creating complex software which can be used to solve real world problems. Its development has begun in 2002 and it has grown to include a wide variety of tools. It has features to support networking, threads, graphical user interfaces, data structures, linear algebra, machine learning, image processing, data mining, XML and text parsing, numerical optimization, Bayesian networks, and many more. Since it is open-sourced, it can be used in any app development free of cost.

2. Imutils:A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and both Python 2.7 and Python 3.

3. Pygame: The pygame library is an open-source module for the Python programming language specifically intended to help you make games and other multimedia applications. Built on top of the highly portable SDL (Simple Direct Media Layer) development library, pygame can run across many platforms and operating systems

4. NumPy: NumPy stands for Numerical Python and it can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

5. Pyfirmata: There are libraries that implement the Firmata protocol in order to communicate (from a computer, smartphone or tablet for example) with Firmata firmware running on a microcontroller platform. PyFirmata is a Python interface for the Firmata protocol and runs on python3.

6. PyQT5: PyQt5 is the latest version of a GUI widgets toolkit developed by Riverbank Computing. It is a Python interface for Qt, one of the most powerful, and popular cross-platform GUI libraries. PyQt5 is a blend of Python programming language and the Qt library. This introductory tutorial will assist you in creating graphical applications with the help of PyQt.

7. SQLite3: Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases. There is no need to install this module separately as it comes along with Python after the 2.5x version.

8. SMTPLib: Python provides smtplib module, which defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon. host − This is the host running your SMTP server. You can specify IP address of the host or a domain name like example.com.

9. PyMsgBox: PyMsgBox is simple, cross-platform, purely implemented in Python for message boxes as JavaScript has. It uses Python's built-in Tkinter module for its GUI.

D. LAPTOP: Used to run our code.

E. WEBCAMERA:

The video will be recorded using the webcam **to see the transition from awake to fatigue and finally, drowsy**. The designed system deals with detecting the face area of the image captured from the video. The purpose of using the face area so it can narrow down to detect eyes and mouth within the face area.

CHAPTER 6
SYSTEM DESIGN

6.1 DETECTION STAGE:

This is the initialization stage of the system. Every time the system is started it needs to be set up and optimized for current user and conditions. The key step in this stage is successful head detection. If the driver's head is correctly located, we can proceed to extract the features necessary for setting up the system. Setup steps include: (i) extracting driver's skin color and using that information to create custom skin color model and (ii) collecting a set of open/closed eyes samples, along with driver's normal head position. To help achieve these goals, user interaction might be required. The driver might also be asked to hold their eyes closed and then open for a matter of few seconds each time. This is enough to get the system started. Over time, the system will expand the dataset of obtained images and will become more error resistant and overall, more robust

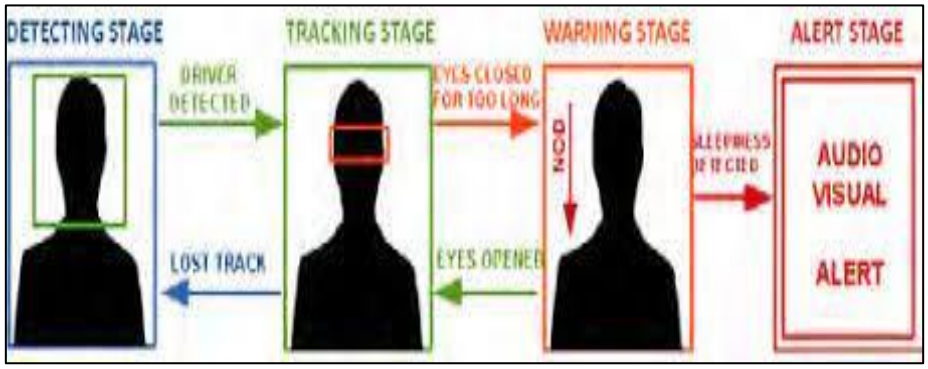


Figure 6.1: Four stages of Drowsiness Detection System

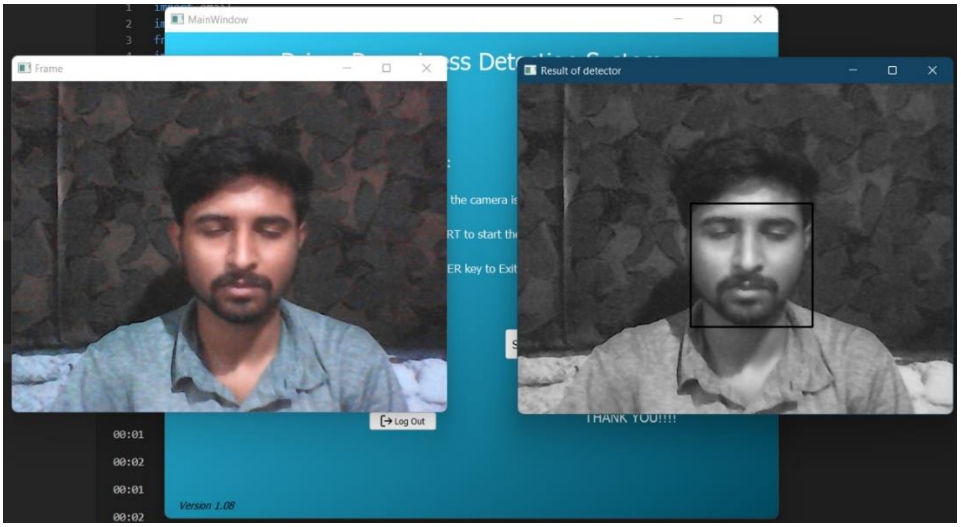


Figure 6.2: Detection stage

6.2 TRACKING STAGE:

Once the driver's head and eyes are properly located and all the necessary features are extracted, the system enters the regular tracking (monitoring) stage. A key step in this stage is the continuous monitoring of the driver's eyes within a dynamically allocated tracking area. More specifically, in order to save some processing time, the system will determine the size of the tracking area based on the previous history of eye movements. For example, if the eyes were moving horizontally to the left for a number of frames it is to be expected that that trend with continue in the following frame also. So it is logical to expand the tracking area towards the expected direction of the eyes and shrink the area in other three directions. During this stage, the system must also determine the state of the eyes. All these tasks must be carried out in realtime; depending on the processor's abilities and current load, it might be necessary to occasionally skip a few frames, without sacrificing algorithmic accuracy.

6.3 WARNING STAGE:

If the driver keeps his eyes closed for prolonged period of time or starts to nod, alertness has to be raised. The key step within this stage is close monitoring of driver's eyes. The system must determine whether the eyes are still closed, and what is the eyes' position relative to previously established thresholds. We cannot afford to skip frames in this stage. In practice, tracking of eyes is performed much in the same way as in the tracking stage with the addition of the following processes: calculation of velocity and trajectory of the eyes and threshold monitoring. These additional computations are required to improve the system's ability to determine whether the driver is drowsy or not.

6.4 ALERT STAGE:

Once it has been determined that the driver appears to be in an abnormal driving state, the system has to be proactive and alert the driver of potential dangers that can arise. Combination of audio/visual alerts are used to attract the driver's attention and raise their alertness level. Alerting has to be implemented in such a way as not to cause the opposite effect of intended and startle the driver into causing an accident.

6.5 USE CASE DIAGRAM:

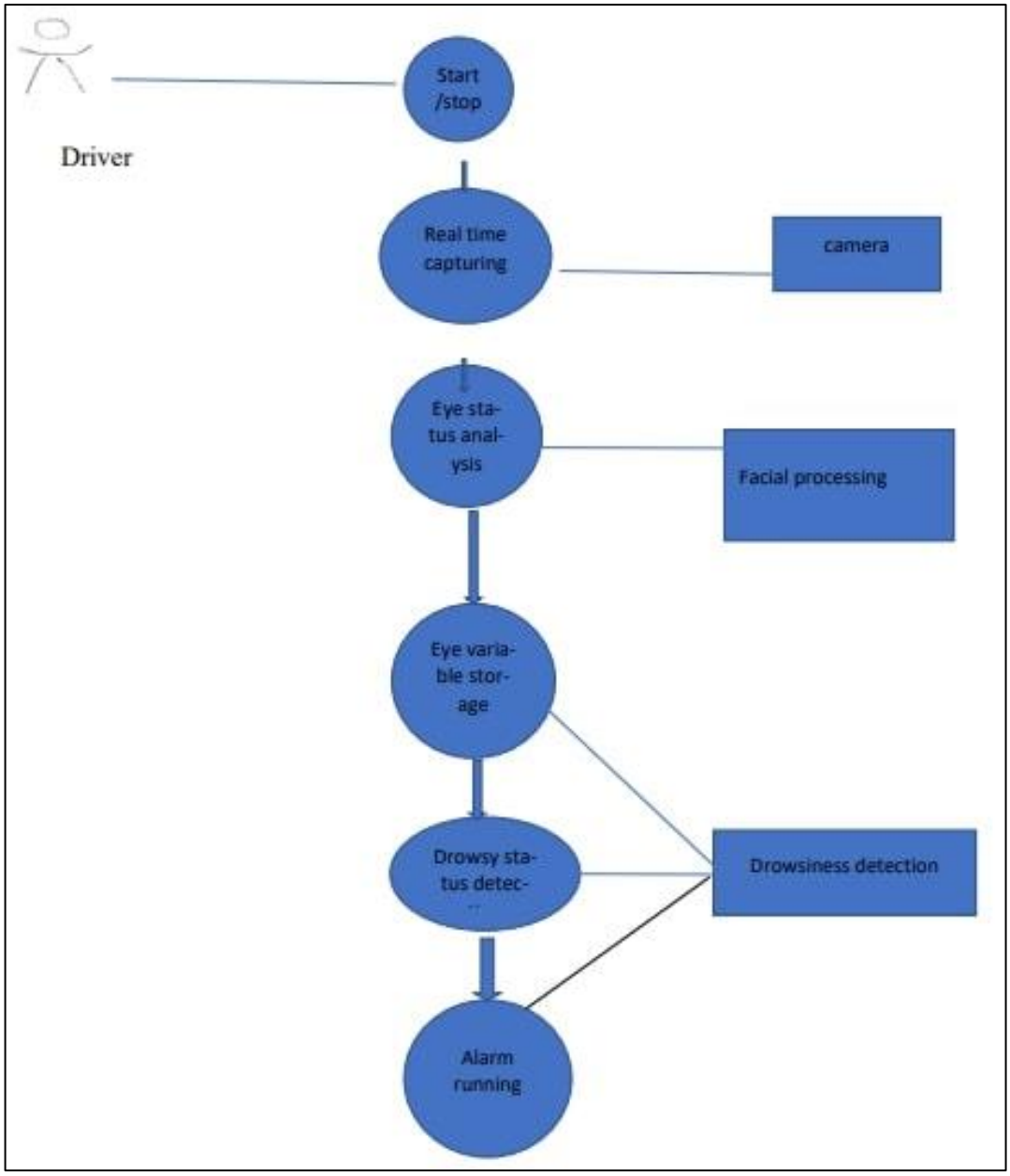


Figure 6.3: Use Case Diagram

CHAPTER 7
WORKING

7.1 IMPLEMENTATION

In this Python project, we have used OpenCV for gathering the images from webcam and feed them into the deep learning model which will classify whether the person's eyes are 'Open' or 'Closed'. The approach we have used for this Python project is as follows:

Step 1 – Take image as input from a camera.

With a webcam, we will take images as input. So to access the webcam, we made an infinite loop that will capture each frame. We use the method provided by OpenCV to access the camera and set the capture object will read each frame and we store the image in a frame variable.

Step 2 – Detect the face in the image and create a Region of Interest (ROI).

To detect the face in the image, we need to first convert the image into grayscale as the OpenCV algorithm for object detection takes grey images in the input. We don't need color information to detect the objects.

Then we perform the detection. It returns an array of detections with x, y coordinates, and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face.

Step 3 – Detect the eyes from ROI and feed it to the classifier.

The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for eyes in **leye** and **reye** respectively then detect the eyes. Now we need to extract only the eyes data from the full image. This can be achieved by extracting the boundary box of the eye and then we can pull out the eye image from the frame

Step 4 – Classifier will categorize whether eyes are open or closed.

We are using CNN classifier for predicting the eye status. To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with

Step 5 – Calculate score to check whether the person is drowsy.

The score is basically a value we will use to determine how long the person has closed his eyes. So, if both eyes are closed, we will keep on increasing score and when eyes are open, we decrease the score. We are drawing the result on the screen using cv2.putText() function which will display real time status of the person.

7.2 SAMPLE SCREENSHOT

- **Active state:**

This is the initialization stage of the system. Every time the system is started it needs to be set up and optimized for current user and conditions. In this state check the position of driver that is in Active state and gives frame of Active state.

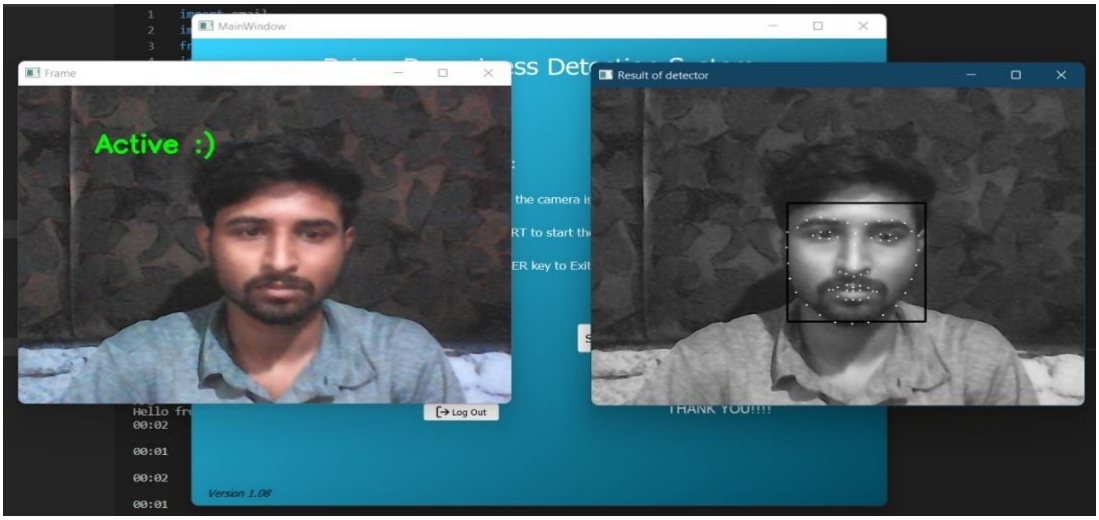


Figure 7.2.1: Active State

- **Drowsy State:**

In this figure gives alert if driver eye become drowsy that gives drowsy state. If drivers could be warned before they became too drowsy to drive safely, some of these crashes could be prevented. In order to reliably detect the drowsiness, it depends on the presentation of timely warnings of drowsiness.

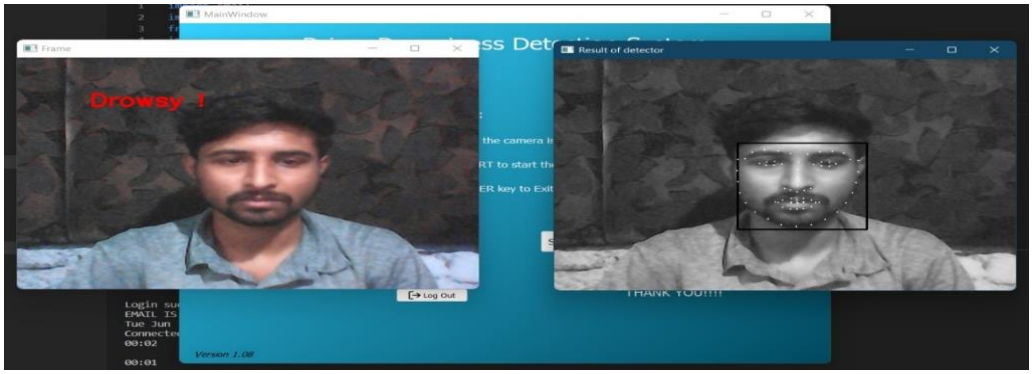


Figure 7.2.2: Drowsy State

- **Detection stage:**

The driver might be asked to sit comfortably in its normal driving position so that system can determine upper and lower thresholds needed for detecting potential nodding. The driver might also be asked to hold their eyes closed and then open for a matter of few seconds each time

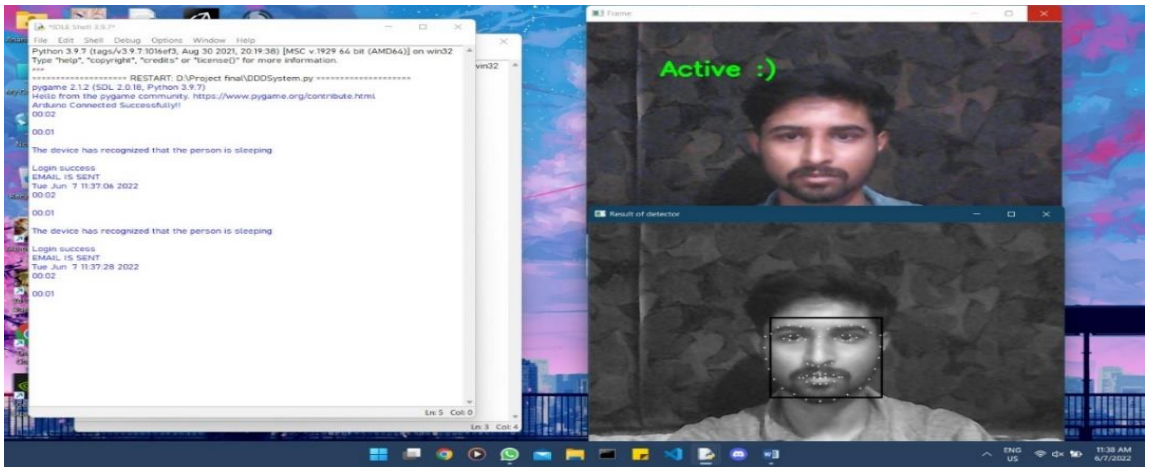


Figure 7.2.3: Detection stage

- **After all stage and Buzzer stage:**

After all stages are performed if driver is in drowsy stage, it gives result of drowsy state and buzzer can work as a alert for driver.

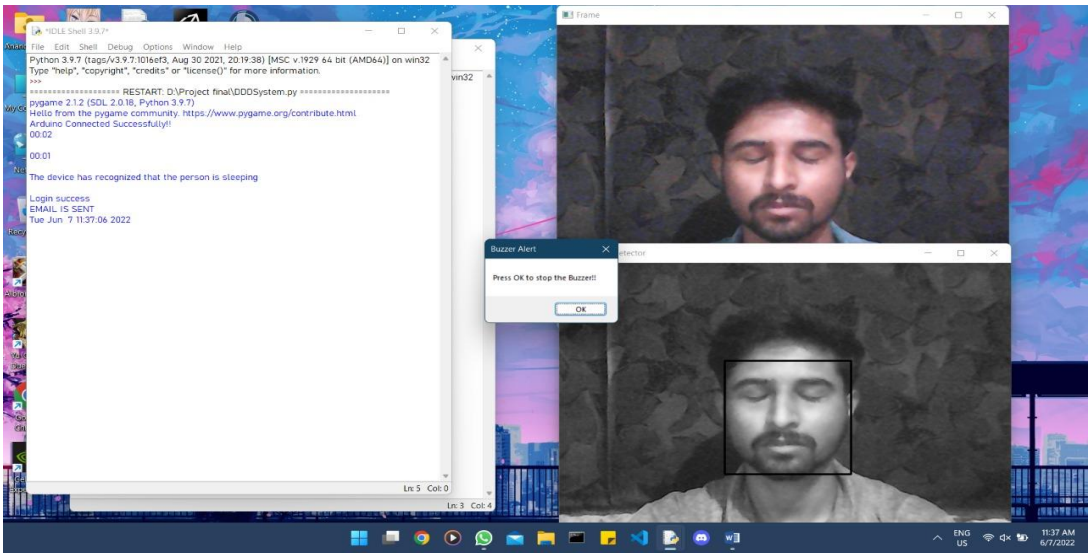


Figure 7.2.4: After all stage and Buzzer stage

- **Back to detection stage 1 after several iteration:**

After all stages are performed than start from initial stage that is detection stage and performed all step one by one this step continues performed.

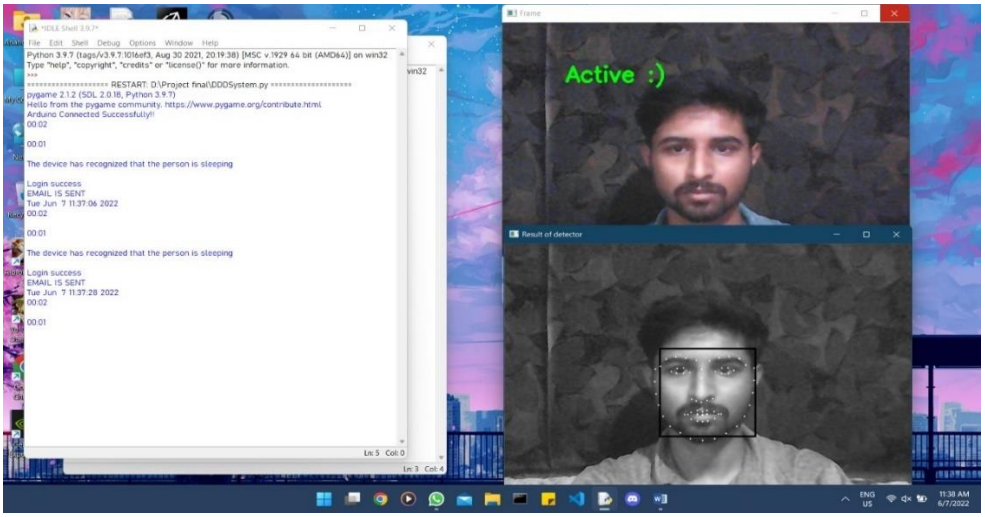
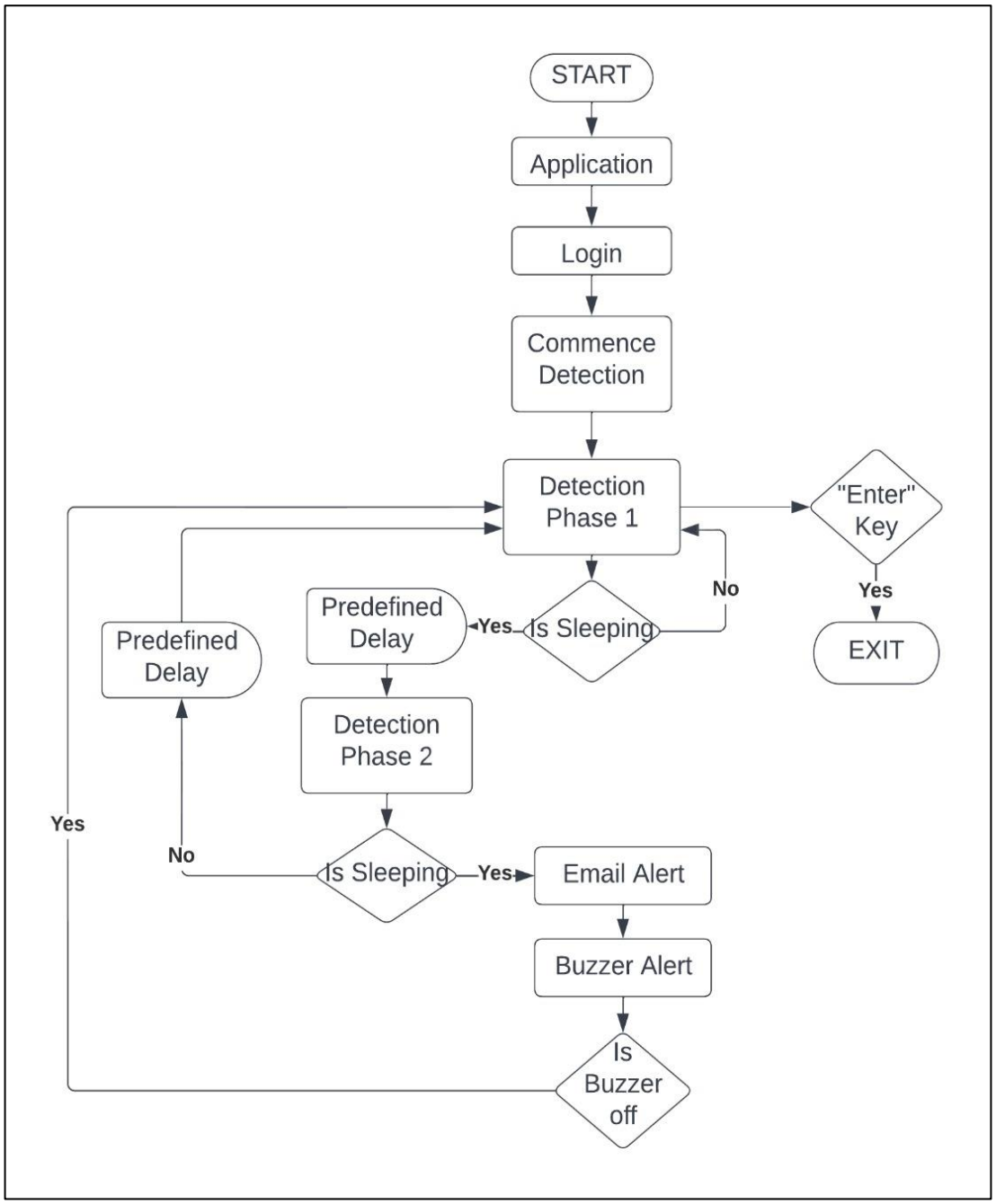


Figure 7.2.5: Back to detection stage 1 after several iteration

7.3 FLOWCHART



Flowchart 7.3: Flowchart for Driver Drowsiness Detection System.

CHAPTER 8
CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

In this paper, we have reviewed the various methods available to determine the drowsiness state of a driver. Although there is no universally accepted definition for drowsiness, the various definitions and the reasons behind them were discussed. This paper also discusses the various ways in which drowsiness can be manipulated in a simulated environment. The various measures used to detect drowsiness include subjective, vehicle-based, physiological and behavioural measures; these were also discussed in detail and the advantages and disadvantages of each measure were described. Although the accuracy rate of using physiological measures to detect drowsiness is high, these are highly intrusive. However, this intrusive nature can be resolved by using contactless electrode placement. Hence, it would be worth fusing physiological measures, such as ECG, with behavioural and vehicle-based measures in the development of an efficient drowsiness detection system. In addition, it is important to consider the driving environment to obtain optimal results

8.2 FUTURE SCOPE:

The future works may focus on the utilization of outer factors such as vehicle states, sleeping hours, weather conditions, mechanical data, etc, for fatigue measurement. Driver drowsiness pose a major threat to highway safety, and the problem is particularly severe for commercial motor vehicle operators. Twenty-four-hour operations, high annual mileage, exposure to challenging environmental conditions, and demanding work schedules all contribute to this serious safety issue. Monitoring the driver's state of drowsiness and vigilance and providing feedback on their condition so that they can take appropriate action is one crucial step in a series of preventive measures necessary to address this problem. Currently there is not adjustment in zoom or direction of the camera during operation. Future work may be to automatically zoom in on the eyes once they are localized.

REFERENCES

REFERENCES

1. National Highway Traffic Safety Administration. Drowsy Driving. (Accessed on 10 May 2021).
2. National Institutes of Health. Drowsiness. (Accessed on 10 May 2021)
3. Kundinger, T.; Sofra, N.; Riener, A. Assessment of the potential of wrist-worn wearable sensors for driver drowsiness detection. Sensors 2020, 20, 1029.
4. Gwak, J.; Hirao, A.; Shino, M. An investigation of early detection of driver drowsiness using ensemble machine learning based on hybrid sensing. Appl. Sci. 2020, 10, 2890.
5. Kashevnik, A.; Lashkov, I.; Ponomarev, A.; Teslya, N.; Gurtov, A. Cloud-based driver monitoring system using a smartphone. IEEE Sens. J. 2020, 20, 6701–6715
6. Tamanani, R.; Muresan, R.; Al-Dweik, A. Estimation of Driver Vigilance Status Using Real-Time Facial Expression and Deep Learning. IEEE Sens. Lett. 2021, 5, 1–4.
7. Sunagawa, M.; Shikii, S.; Nakai, W.; Mochizuki, M.; Kusukame, K.; Kitajima, H. Comprehensive Drowsiness Level Detection Model Combining Multimodal Information. IEEE Sens. J. 2020, 20, 3709–3717.
8. Ouabida, E.; Essadike, A.; Bouzid, A. Optical correlator-based algorithm for driver drowsiness detection. Optik 2020, 204, 164102.
9. Kiashari, S.E.H.; Nahvi, A.; Bakhoda, H.; Homayounfard, A.; Tashakori, M. Evaluation of driver drowsiness using respiration analysis by thermal imaging on a driving simulator. Multimed. Tools Appl. 2020, 79, 17793–17815.
10. Pranjali R. Hiwale.; Anand A. Kalsait.; Kishori Y. Choukade.; Aishwarya S. Puri Design and Implementation of Driver Drowsiness Detection System IJRTI Vol.7 Issue 6, June-2022.