Contents

CERTIFICATE ii

COMPLIANCE CERTIFICATE iii

PAPER PUBLICATION CERTIFICATE iv

THESIS APPROVAL CERTIFICATE v

STATEMENT OF ORIGINALITY vi

Acknowledgements vii

Table of Contents x

List of Figures xi

List of Tables xii

Abbreviations xiii

Abstract xiv

Table of Contents

[Chapter 1 Proposal 5](#_Toc69377776)

[1.1 Introduction 5](#_Toc69377777)

[1.2 Background 5](#_Toc69377778)

[1.3 Problem Statement 6](#_Toc69377779)

[1.4 Aim 6](#_Toc69377780)

[1.5 Objectives 6](#_Toc69377781)

[1.6 Signiﬁcance of The Project 7](#_Toc69377782)

[1.7 Methodology 7](#_Toc69377783)

[1.8 Scope 8](#_Toc69377784)

[1.9 Deﬁnition of Key Variables 8](#_Toc69377785)

[1.10 Conclusion 9](#_Toc69377786)

[Chapter 2 Literature Review 9](#_Toc69377787)

[2.1 Introduction 9](#_Toc69377788)

[2.2 Synthesis of literature 9](#_Toc69377789)

[2.3 Conclusion 13](#_Toc69377790)

[Chapter 3 Requirements Analysis 14](#_Toc69377791)

[3.1 Introduction 14](#_Toc69377792)

[3. 2 Current System. 14](#_Toc69377793)

[3.2.1 Context Level Diagram 15](#_Toc69377794)

[3.2.2 Process Flow Diagram 16](#_Toc69377795)

[3.2.3 Use-case 17](#_Toc69377796)

[3.3 Feasibility Study 17](#_Toc69377797)

[3.3.1 Technical Feasibility 18](#_Toc69377798)

[3.3.2 Economic Feasibility 19](#_Toc69377799)

[3.4 Requirements Analysis 19](#_Toc69377800)

[3.3.1 Functional Requirements 19](#_Toc69377801)

[3.3.1.1 Context Level DFD 20](#_Toc69377802)

[3.2.1.2 DFD Level 1 20](#_Toc69377803)

[3.2.1.3 Use-case 21](#_Toc69377804)

[3.2.2 Non-functional Requirements 21](#_Toc69377805)

[3.2.2.1 Performance 21](#_Toc69377806)

[3.2.2.2 Usability 21](#_Toc69377807)

[3.2.2.3 Security 22](#_Toc69377808)

[3.3 Interface Requirements 22](#_Toc69377809)

[3.4 Technical Requirements 22](#_Toc69377810)

[3.5 Assumptions 23](#_Toc69377811)

[3.6 Conclusion 23](#_Toc69377812)

[Chapter 4 Design 23](#_Toc69377813)

[4.1 Introduction 23](#_Toc69377814)

[4.2 Proposed Solution 24](#_Toc69377815)

[4.3 Solution Architecture 24](#_Toc69377816)

[4.4 Constraints 24](#_Toc69377817)

[4.5 Security Design 24](#_Toc69377818)

[4.6 Systems Design Models 24](#_Toc69377819)

[4.6.1 UML-Activity Diagram 24](#_Toc69377820)

[4.6.2 UML-Class Diagram 25](#_Toc69377821)

[4.6.3 UML-Sequence Diagram 26](#_Toc69377822)

[4.6.4 UML-Deployment Diagram 26](#_Toc69377823)

[4.7 Database Modelling 26](#_Toc69377824)

[4.7.1 E-R Diagram 26](#_Toc69377825)

[4.7.2 Data Dictionary 26](#_Toc69377826)

[4.7.3 Relational Schema 26](#_Toc69377827)

[4.7.3.1 1st Normal Form 26](#_Toc69377828)

[4.7.3.2 2nd Normal Form 26](#_Toc69377829)

[4.7.3.3 3rd Normal Form(optional) 26](#_Toc69377830)

[4.8 Algorithm Design 26](#_Toc69377831)

[4.9 Interface Design 26](#_Toc69377832)

[4.10 Conclusion 26](#_Toc69377833)

[Chapter 5 Implementation 26](#_Toc69377834)

[5.1 Introduction 26](#_Toc69377835)

[5.2 Coding Conventions 26](#_Toc69377836)

[5.3 Coding Strategy 26](#_Toc69377837)

[5.4 Coding Review 26](#_Toc69377838)

[5.4.1 Manual Checklist 26](#_Toc69377839)

[5.5 Conclusion 29](#_Toc69377840)

[Chapter 6 Systems Testing 29](#_Toc69377841)

[6.1 Introduction 29](#_Toc69377842)

[6.2 Testing Categories and Results 29](#_Toc69377843)

[6.2.1 White Box 29](#_Toc69377844)

[6.2.2 Black Box 29](#_Toc69377845)

[6.3 Types of Testing and Results 29](#_Toc69377846)

[6.3.1 Functional Testing 29](#_Toc69377847)

[6.3.2 Non-Functional Testing 29](#_Toc69377848)

[6.4 Test Cases 29](#_Toc69377849)

[6.5 Levels of Testing and Results 29](#_Toc69377850)

[6.5.1 Unit Testing 29](#_Toc69377851)

[6.5.2 Integration Testing 29](#_Toc69377852)

[6.5.3 Validation Testing 29](#_Toc69377853)

[6.5.4 Systems Testing 29](#_Toc69377854)

[6.5.5 Acceptance Testing 29](#_Toc69377855)

[6.6 System Evaluation 29](#_Toc69377856)

[6.7 Conclusion 29](#_Toc69377857)

[Chapter 7 Conclusion 29](#_Toc69377858)

[7.1 Introduction 29](#_Toc69377859)

[7.2 Scope of Future Work 29](#_Toc69377860)

[7.3 Recommendations 29](#_Toc69377861)

# Chapter 1 Proposal

## Introduction

The automobile industry in today’s time has shown a steady rise across the globe. Consequently, the number of vehicles is increasing exponentially, which has further led to an increase in road accidents in our country. The road accidents have proved to be a menace that has majorly reduced the safety of the general public, let alone the driver**.** As a result, the fatalities and associated expenses that follow prove to be a severe threat to families across the world. The current drowsiness detection methods used are not widespread due to their high cost and less availability, thus making them unfeasible in the usual or non-luxury vehicles. Therefore, there is a growing need for a smart and viable drowsiness detection system that the numerous automobiles in the industry can quickly adapt. We are to build a drowsiness detection system that will monitor a driver’s visual features (i.e., eyes for example) and ascertain if the driver is in a drowsy state. This system will alert the driver when drowsiness is detected by this criterion in real time (using an alarm). All this will be made possible in the growing and vibrant Machine Learning area of IT.

## Background

Driver drowsiness is a significant factor in the increasing number of accidents on today’s roads and has been extensively accepted as noted in [1]. This proof has been verified by many researchers 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 dangers of accidents anticipated to drowsiness. So far, researchers have tried to model the behaviour by creating links between drowsiness and certain indications related to the vehicle and to the driver [2], [3].

The automobile business also has tried to build several systems to predict driver drowsiness but there are only a few commercial products available today [4]. The systems do not look at driver performance and overlook driver ability and external environment characteristics. Naturally, most people would agree that different people drive differently. The system being proposed aims to be able to adapt to the changes of the driver’s visual behaviour and alert accordingly thus curbing a potential road fatality.

## Problem Statement

According to the Traffic Safety Council of Zimbabwe (TSCZ) [5], over US$ 460 million is spent annually on road traffic accidents, with an average of 40,000 accidents being recorded annually.At least 15,000 people are injured and almost 2,000 killed every year in road traffic accidents, says [5].A large portion of those fatalities are attributed to human error, hence the need for such a project that curbs the effect of fatigue on drivers. Studies even show that sleepiness can impair driving performance as much as or more than alcohol. In this study, we will attempt to develop a system to detect drowsiness and alert drivers of their fatigue to allow them to take a break so that they don’t endanger themselves and other road users, thus saving lives.

## Aim

To design a system that will detect drowsiness and take necessary steps to avoid accidents. The driver drowsiness detection system being implemented in this project aims at being easily available and able to be used with different types of vehicles.

## 1.5 Objectives

The main objectives of the research are as follows:

* To take camera image input in real-time to identify facial Region of Interest (e.g., eye detection) by February 2021.
* To design a deep learning algorithm for drowsiness detection based on real-time input by April 2021.
* To alert the driver upon detection of drowsiness using an alarm sound by May 2021.

## 1.6 Signiﬁcance of The Project

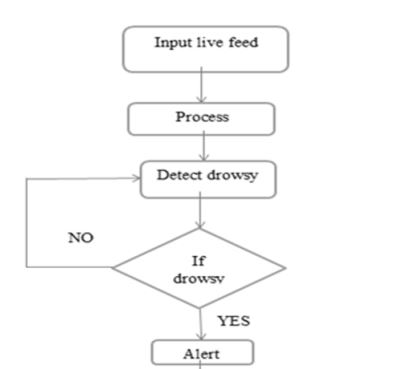
In the present creating world, advancements are impacting at each place to bolster our mankind. There has been a great headway in technological advancements of vehicles over past years and still to come. But accidents are yet occurring around us. This is because of the absence of transportation control facilities, contravention of guidelines and inattention while driving has expanded lethal rates. The research carried by the Traffic Safety Council of Zimbabwe (TSCZ) [5] shows that over US$ 460 million is spent annually on road traffic accidents, with an average of 40,000 accidents being recorded annually.At least 15,000 people are injured and almost 2,000 killed every year in road traffic accidents, says [5].

In general, the project is developed to come up with a viable solution for accidents caused by falling asleep while driving. The development of this project is necessary as it will save innocent lives caused by fatigued drivers. Fatigue is extreme tiredness brought about by not enough rest over a period of time whether from mental or physical exertion or illness. Ideally, each individual needs between seven and eight hours of good quality sleep each night so that they have less built up sleep debt, or sleep deficit. At worst, drivers with sleep debt risk nodding off, yet fatigue can impair reaction time and decision making when behind the wheel which increases the risk of being involved in an accident [6].

## 1.7 Methodology

**The proposed solution will follow the following methodology:**

#### Figure 1.7.1: Flowchart – Conceptual Design of proposed Driver Drowsiness Detection approach



**Step 1 –** Take image as input from a camera.

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

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

**Step 4 –** Classifier will categorize whether criteria for drowsiness is satisfied.

**Step 5 –** Alert the driver if drowsiness is determined.

## 1.8 Scope

* To take image as input from a camera.
* To implement eye tracking from image input.
* To categorize whether criteria for drowsiness is satisfied using a deep learning algorithm.
* To alert the driver if drowsiness is determined using an alarm.

## 1.9 Deﬁnition of Key Variables

Keywords: Real-time, Machine Learning algorithms, Image Processing, Eye Detection, Driver drowsiness, Alarm system.

## 1.10 Conclusion

Developing a Driver Drowsiness Detection system reduces the threat of road fatalities by ensuring efficient drowsiness detection in real-time and appropriate alerting of drivers. The development of this project is necessary as it will save innocent lives caused by fatigued drivers. This system will aim at being easily available and able to be used with different types of vehicles. All this will be made possible in the growing and vibrant Machine Learning area of IT.

# Chapter 2 Literature Review

## 2.1 Introduction

In an attempt to increase the efficiency over the years in drowsiness detection, several approaches have been proposed. This section discusses the previous methods and approaches to drowsiness detection in summary with their limitations to get an idea of the current state of drowsiness detection systems.

## 2.2 Synthesis of literature

**DRIVER AWARENESS**

Drivers are normally aware when they are feeling sleepy, and therefore make a conscious decision about whether to stop and rest or to continue driving while trying to fight off sleepiness and stay awake. Horne has demonstrated that most drivers involved in sleep related accidents, deny having fallen asleep. This may be due to embarrassment, fear of prosecution or loss of insurance indemnity, or to a genuine belief that they did not fall asleep. Laboratory studies have shown that if people are woken within a few minutes of falling asleep, they will have no knowledge of having fallen asleep [7].

However, even if drivers are genuinely unaware of having fallen asleep, they are fully aware of feeling sleepy beforehand. Horne used a driving simulator on which subjects whose sleep had been restricted to five hours the night before, drove for two hours in the afternoon on a monotonous road, to assess awareness of sleepiness while driving, awareness of the likelihood of falling asleep during the drive and the level of incidents due to sleepiness while driving. The study showed that drivers were well aware when they were feeling sleepy, and generally were aware that this meant they might fall asleep. The number of incidents increased as drivers grew more sleepy, and all the major incidents (where the car drifted out of the lane completely) occurred after a lengthy period in which the driver was aware of increasing sleepiness, and usually after a period of fighting sleepiness. Some drivers did not seem to realise that feeling very sleepy meant that they were likely to actually fall asleep. Another study [8]by Horne suggested that people often fall asleep more  
quickly than they realise or expect.

Sleep has a long history of research in the fields of psychology and medicine, where accurate measurements and indicators have been developed [9].

**INTRUSIVE METHODS**

*Physiological Measure*

There are Different Methods for collecting and analysing physiological data during real-world tasks to determine a stress level of persons. The different physiological parameters responsible for determining stress are Electrocardiogram, Electroencephalography, Electromyogram, skin conductance, respiration etc. One of the most prominent method is the Electroencephalography (EEG), which looks at the electrical activity of the brain. Experiments have been done earlier by different researchers taking different parameters like Electro-Oculogram (EOG), EEG, ECG, Electromyogram (EMG) and skin conductance. It has been established that the variations in the Heart Rate can detect different physical conditions including  
drowsiness.

Electroencephalograms (EEG) [10] represent the electrical changes in the brain, measured with a series of electrodes placed in the scalp. The electrodes detect small voltages  
produced in the brain cortex. These potentials form waves at several frequencies, known as delta, theta, alpha, beta and gamma waves, which are linked to different cognitive and motor processes, including drowsiness and the different sleep stages. Brain studies couple EEG with electrooculography (EOG), which detects eye movements, and electromyogram (EMG) that monitors muscular tone. These measurements provide the best data for detection of drowsiness, and as such have been used by several drowsiness detection systems, usually in conjunction with heart rate and breathing rate. The Problem of these techniques is that they are intrusive to the subject. Wiring is problem for this approach. The electrode contacts and wires will annoy the drivers and are difficult to be implemented on vehicles. They require electrodes and other sensors to be placed on the head, face and chest which may annoy the driver. They also need to be carefully placed: installing the electrodes to obtain an EEG.

Thus, this approach uses data acquired from physiological sensors, such as Electrooculography (EOG), Electrocardiogram (ECG) and Electroencephalogram (EEG) data. EEG signals provide information about the brain’s activity.

This approach can be summarized as follows:

1. After data acquisition, we have computed logarithm of energy of EEG signal as feature.

2. Higuchi’s and Petrosian’s fractal dimensions as chaotic features.

3. These features will be used to separate alertness and drowsiness levels of drivers using statistical analysis (two-tailed t tests).

This approach is one of the most accurate method to detect drowsiness with an accuracy rate of almost 90% [11],[12].

Limitation – This approach requires a lot of sensors to be attached to driver’s body which makes this approach very expensive method. Also, it can make the driver stressful because the driver will know that it is possible to get drowsy which will result in EEG data which would be a mixture of stress and drowsiness [11].

**NON-INTRUSIVE METHODS**

*Behavioural Measure*

In order to detect drowsiness, studies on driver’s performance use lane tracking, distance between driver’s vehicle and the vehicle in front of it; place sensors on components of the vehicle such as steering wheel, gas pedal and analyse the data taken by these sensors. Pilutti and Ulsoy used vehicle lateral position as the input and steering wheel position as the output and they obtained a model which can be useful to detect drowsiness.

Thus, the primitive approach was based on driving patterns, and it was highly dependent on vehicle characteristics, road conditions, and driving skills. To calculate driving pattern, deviation from a lateral or lane position or steering wheel movement should be calculated [11][12]. While driving, it is necessary to perform micro adjustments to the steering wheel to keep the car in a lane. This approach can be summarized in four steps:

1. Collect individual steering data and the associated fatigue ratings for each participant.

2. Extract relevant features from the steering data.

3. Build statistical models of the fatigue ratings based on the features.

4. Test the learned models on unseen steering data.

Using this method, detected drowsiness had 86% accuracy on the basis of correlations between micro adjustments in steering wheel and drowsiness. Also, it is possible to use deviation in a lane position to identify a driving pattern where car position respective to the lane [11].

Limitation – Approaches based on the driving pattern are highly dependent on vehicle characteristics, road conditions, and driving skills.

*Visual Measure*

An increasing research interest has focused on developing systems that detect the visual facial feature changes associated with fatigue with a video camera. These facial features include eyes, head position, face, or mouth. This approach is non-intrusive and becomes more and more practical with the rapid development of camera and computer vision technology. People in fatigue exhibit certain visual behaviours that are easily observable from changes in facial features like the eyes, head, and face. Visual behaviours that typically reflect a person’s level of fatigue include eyelid movement, head movement, gaze, and facial expression.

The method is based on facial feature extraction. It uses behaviours such as facial expression, yawning duration, head movement, and eye blinking pattern.[13][14]

The approach based on eye blinking pattern:

1. Receive an input from a colour video camera attached in front of the driver.

2. Detect the face using face recognition algorithm.

3. Checks the eye of the driver to detect the eye blink duration.

4.Set threshold time period to detect drowsiness.

This calculation considered the number of blinks per minute, assuming that it increases as the driver becomes drowsier [14]. In this approach, the accuracy rate for drowsiness detection comes out to be around 78%.

Limitation – Accuracy is little less than EEG method.

**Summary**

As per the analysis, the author concluded that despite having more accuracy in the intrusive physiological approach still it is not economically feasible. Hence, we go with the action-based visual measure approach for drowsiness detection. Recently, the deep learning approaches, especially the Convolutional Neural Networks. (CNNs) methods, have gained prominence in resolving challenging classification problems. CNNs clubbed with action-based approach can help us increase the accuracy of drowsiness detection making it one of the best methods to be used for drowsiness detection.

## 2.3 Conclusion

This literature review began with a brief summary of literature pertaining to driver awareness and fatigue indicators. It went on to cover the intrusive and non-intrusive methods of drowsiness detection currently available and evaluated their shortcomings. An analysis of these allowed the researcher to conclude on the method to be used for the study using visual measures, machine learning concepts and real-time input for eye tracking.

# Chapter 3 Requirements Analysis

## 3.1 Introduction

In this chapter we will discuss what is needed in coming up with the system; this includes functional and non-functional requirements. It also specifies whether it is feasible to come up with the system and whether the technologies available are sufficient to come up with the system through a thorough assessment.

## 3. 2 Current System.

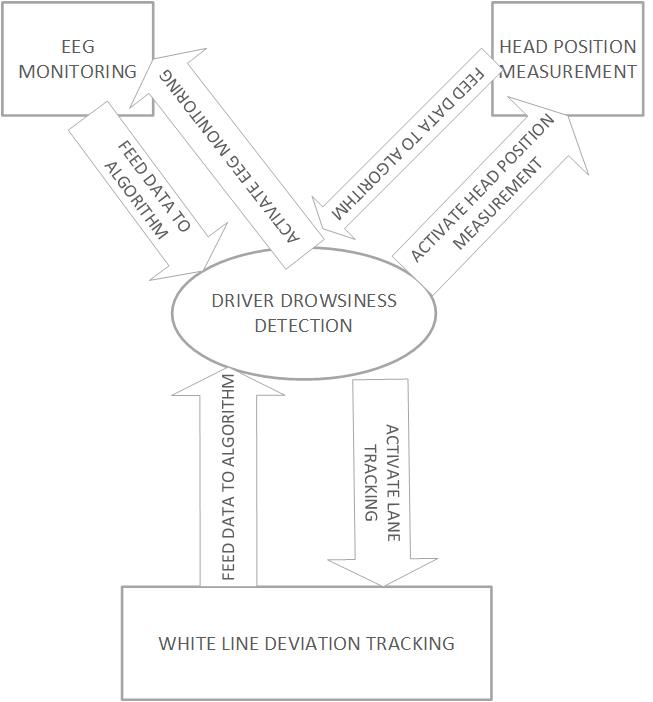
A significant factor for causing mishaps in rush hour gridlock is the driver's exhaustion. Numerous nations are occupied with research here effectively now. The major thrust is to reduce the threat of auto collisions and road fatalities. Currently there are a variety of intrusive methods in place.

Head position measurements featuring weakness degree studies are a common example. At the point when the driver is exhausted, the head will in every case descend slanting. As indicated by measurements, the connection coefficient of head position and weakness degree is about 0.8. However, some driver's head stance won't change essentially, the connection coefficient will be negative and the framework's results and early detection will be flawed.

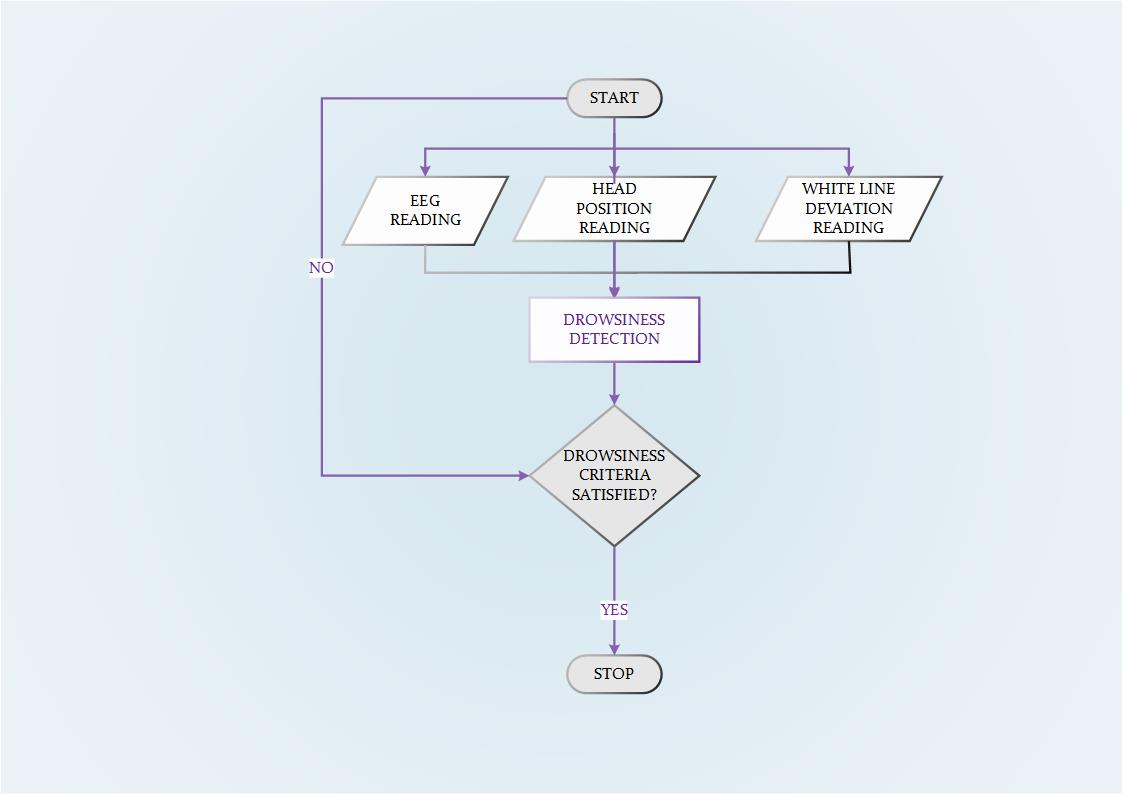
Another framework identifies the driver's psychological state by checking guiding wheel's developments and examples. This technique screens the time and the deviation level of vehicles leaving from the white lines by introducing camera in a similar viewpoint with the driver on the vehicle. However. this estimation requires that the white line must exist and be clear enough, thus the impedance of outside conditions is significant making it unreliable.

EEG studies are another measure used currently. It uses the EEG's recurrence dispersion and waveforms measured from the driver’s vitals, to monitor the status of the cerebrum action and guess whether the driver is exhausted. However, the EEG is defenceless against impedance from outer components (electrical interference) and there are a significant number of contrasts in individual physiological reaction that would make readings inaccurate.

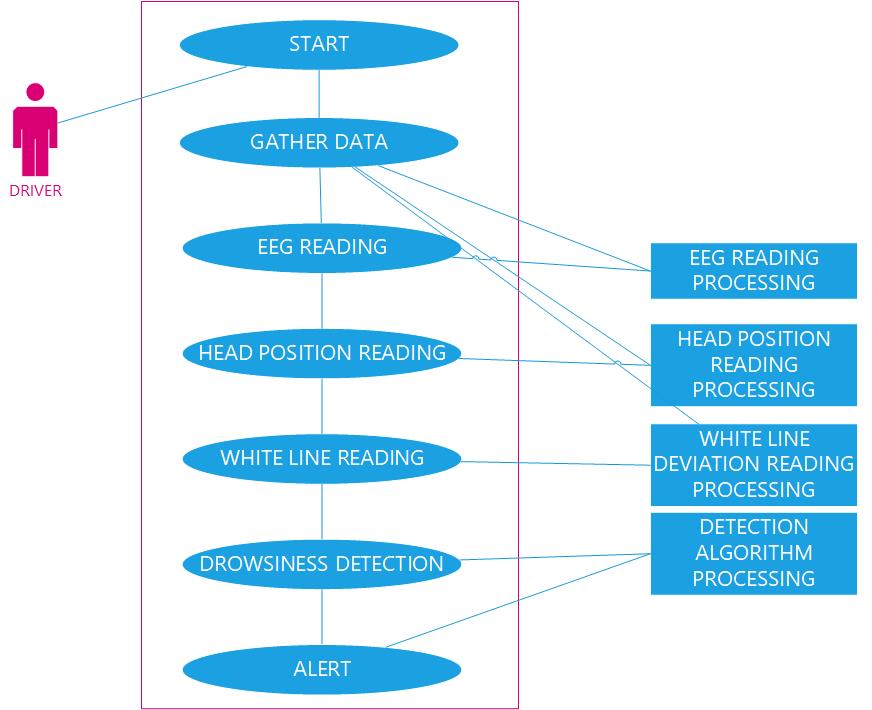
### 3.2.1 Context Level Diagram



### 3.2.2 Process Flow Diagram



### 3.2.3 Use-case



## 3.3 Feasibility Study

During the course of the research, an assessment was done which was undertaken to determine whether the suggested system can be developed and implemented considering the current economic conditions, financial and technological constraints. This was done to ensure that the proposed system is feasible and can be implemented.

In consideration of the system’s feasibility two main categories were assessed:

**Technical Feasibility** –is it possible to build the system with the available technical resources.

**Economic Feasibility** –is it feasible to develop the system with the current economic situation and will it be cost effective.

### 3.3.1 Technical Feasibility

The system is considered technically feasible if it has the necessary software, equipment, technology and personnel to develop, install, operate and maintain the proposed system.

Table 3.1 Desktop Hardware Requirements – System Development

|  |  |  |
| --- | --- | --- |
| Component | Minimum Required | Available |
| Processor | Intel® Core™ i3  Processor 2.00GHz | Intel® Core™ i5  Processor 2.7GHz |
| Memory (RAM) | 2GB | 4GB |
| Storage (Hard drive) | 200GB | 1TB |
| HD Webcam | HD Webcam | HD Webcam |

Table 3.2 Desktop Software Requirements – System Development

|  |  |  |
| --- | --- | --- |
| Software | Minimum Version Required | Version Available |
| Operating System | Windows 7 | Windows 10 |
| Microsoft Office | Office 2007 | Office 2016 |
| Anaconda | 2 | 3 |
| Python | 2 | 3.6 |
| OpenCV | 3.2.0.8 | 3.2.0.8 |
| Tkinter | 8.5 | 8.5 |

The available desktop hardware and software can support the development of the system.

### 3.3.2 Economic Feasibility

For any system if the expected benefits equal or exceed the expected costs, the system can be judged to be economically feasible. In economic feasibility, cost benefit analysis is done in which expected costs and benefits are evaluated. Economic analysis is used for evaluating the effectiveness of the proposed system. In economic feasibility, the most important is [cost-benefit analysis](https://www.freetutes.com/systemanalysis/sa3-cost-benefit-analysis.html). As the name suggests, it is an analysis of the costs to be incurred in the system and benefits derivable out of the system.

**Cost Benefit Analysis (CBA)**

If the cost can be quantified, they are tangible, otherwise they are intangible.

* **Tangible CBA** – a user friendly User Interface will require minimum training of the person making use of the system. Hence, time needed to carry out the training is reduced, as the system is designed to have as much minimum interaction with the driver as possible so as not to disturb normal driving.
* **Development CBA** – as mentioned earlier in the technical feasibility study, minimum development costs will be incurred, thus overall costs are reduced.
* **Intangible CBA** – increased use of the Driver Drowsiness Detection system even in commercial everyday vehicles, resulting in reliable driver safety on our roads and accident reduction.

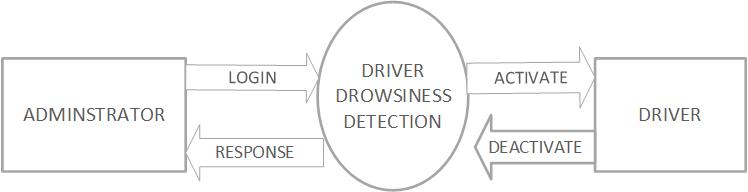
Deriving from the above analysis, the developers can implement the system since its economically sound to undertake the project.

## 3.4 Requirements Analysis

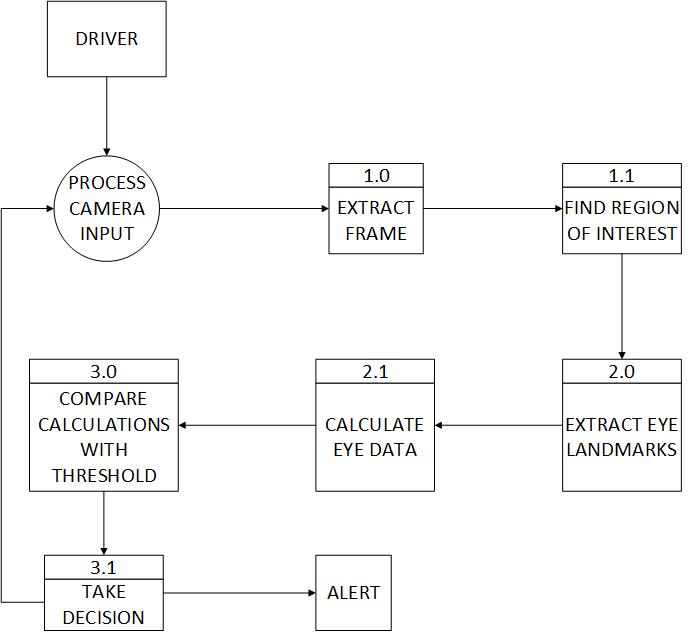
### 3.3.1 Functional Requirements

The system will allow the driver to start/stop the camera manually.

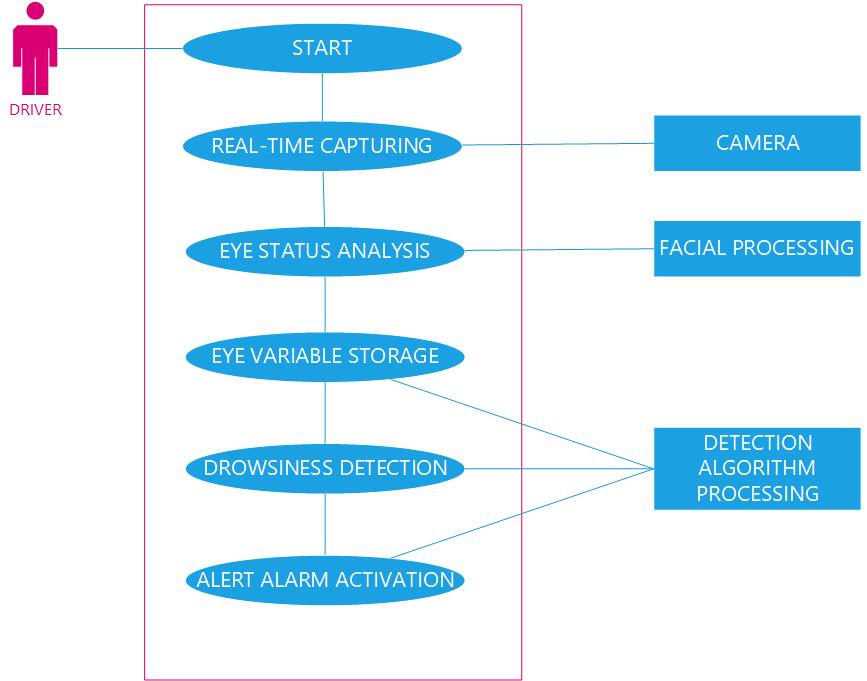
### 3.3.1.1 Context Level DFD



### 3.2.1.2 DFD Level 1



### 3.2.1.3 Use-case



### 3.2.2 Non-functional Requirements

A specification of how well the system must function.

**Availability -**The system should be available at all times when switched on to take camera input of the driver in real-time.

**Maintainability-**It shall be easy to maintain and any modification to algorithm, service of camera and other connecting components must be done by professional personnel only.

### 3.2.2.1 Performance

The performance requirements of Driver Drowsiness Detection are the following:

* It should process the Eye tracking algorithm in less than 0.9 seconds.
* It should alert fatigued driver with an alarm after 5 seconds of detected drowsiness criteria satisfaction.

### 3.2.2.2 Usability

The usability requirements of Driver Drowsiness Detection are the following:

* The system must provide a simple and user-friendly GUI in the event of a manual override by authorised personnel.
* System must be robust to extreme road conditions.
* The number of physical interactions should be minimum.
* The camera position must be well aligned to take real-time input of driver.

### 3.2.2.3 Security

* User also need to activate notification access permission on device manually.
* Authorised User's credentials shall not be accessed or reached by anyone else and authorised credentials will only be used in the event of a manual override.

### 3.3 Interface Requirements

* The system will have an onboard interface in the vehicle that will show the status of the system when it goes on and off.
* A camera will be required to take real-time input of the driver.
* OpenCV library will be used for Driver Drowsiness Detection.

### 3.4 Technical Requirements

**Hardware Requirements**

|  |  |
| --- | --- |
| Component | Minimum Required |
| Processor | Intel® Core™ i3  Processor 2.00GHz |
| Memory (RAM) | 2GB |
| Storage (Hard drive) | 200GB |
| HD Webcam | HD Webcam |

**Software Requirements**

|  |  |
| --- | --- |
| Software | Minimum Version Required |
| Operating System | Windows 7 |
| Anaconda | 3 |
| Python | 3 |
| OpenCV | 3.2.0.8 |
| Tkinter | 8.5 |

### 3.5 Assumptions

It is assumed that:

* There will be no reflective surfaces in the view of the camera to obstruct its view of the driver.
* The camera will have good night vision capabilities for night-time driving.
* The camera will be securely fastened so as to avoid disturbance in harsh road conditions.

### 3.6 Conclusion

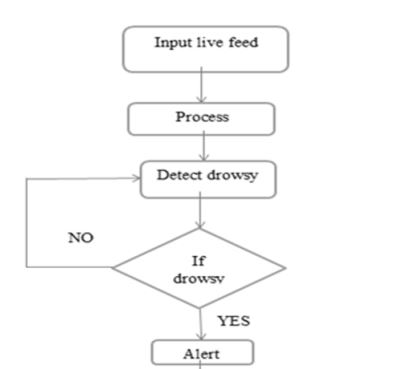
This requirements analysis began with a look at the current systems in place in the problem space. It went on to cover the overall feasibility of the project by assessing whether technical and economic needs could be met. An analysis of these allowed the researcher to conclude that the project is indeed feasible and thus functional and non-functional re4quirements were adequately determined.

# Chapter 4 Design

## 4.1 Introduction

The chapter is focused on the design of the proposed system. It highlights on the physical design, architecture design and interface design of the proposed system and pays close attention to the constraints, guidelines and principles of design.

## 4.2 Proposed Solution



**Step 1 –** Take image as input from a camera.

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

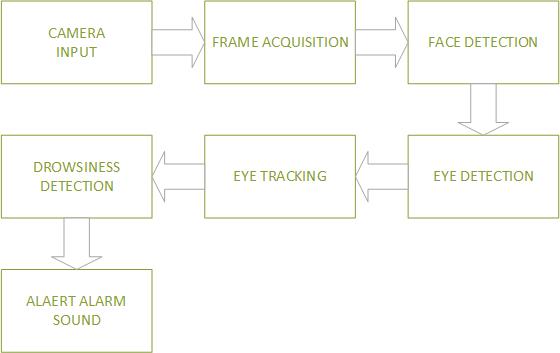
**Step 3 –** Detect the eyes from ROI and feed it to the classifier for tracking.

**Step 4 –** Classifier will categorize whether criteria for drowsiness is satisfied.

**Step 5 –** Alert the driver if drowsiness is determined.

## 4.3 Solution Architecture

This includes translation of the abstract logical model into the specific technical design. It is the design of the hardware and software that the proposed system is going to operate on. The design involves use of a camera to input the real-time image sequence to the drowsiness detection system. The system will perform face detection, eye detection, eye state tracking and perform the necessary calculations to ascertain if the driver is drowsy or not. In the event of a drowsy condition, the driver is then alerted via a sound alarm.



## 4.4 Constraints

* Space: The solution needs to be implemented in a space-efficient manner. It must not interfere with the existing controls of the car.
* Power: There will be a limited power source so the solution needs to be designed so that it can operate properly on limited power requirements.

## 4.5 Security Design

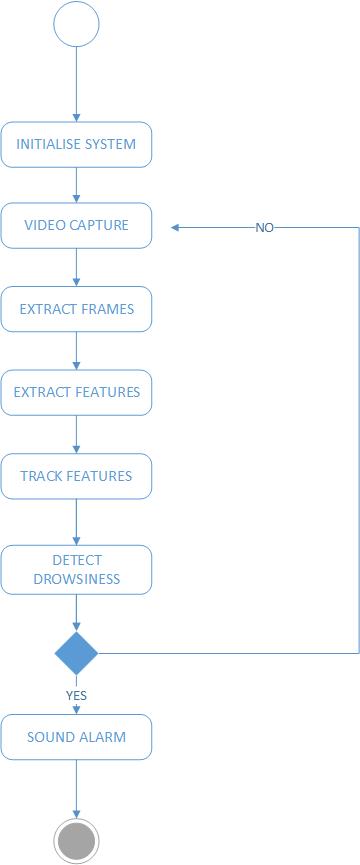
The system is offline to reduce the amount of security threats that can occur. This places particular attention to the security of access to the system outside the confines of the user and solely on the approved administrator in the case of manually disabling the system.

**User security**

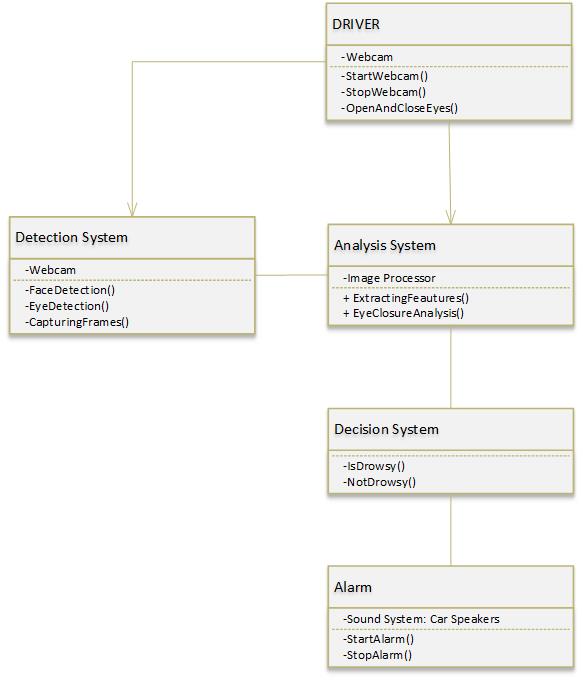
The system will have the facility of using passwords so that authorised administrators will be allowed to access the system and disable it in the case of malfunction. Each administrator will have a unique username and password for logging into the system. Only correct passwords are accepted. Administrators should have unique passwords, should not share passwords and passwords should not be documented.

## 4.6 Systems Design Models

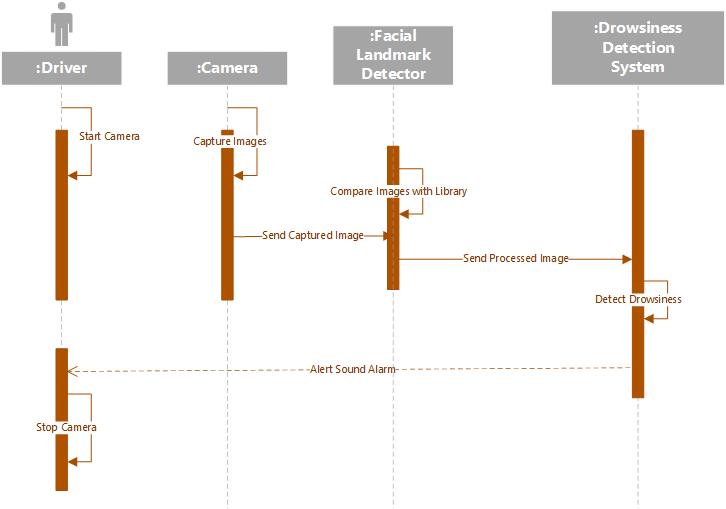
### 4.6.1 UML-Activity Diagram



### 4.6.2 UML-Class Diagram



### 4.6.3 UML-Sequence Diagram



### 4.6.4 UML-Deployment Diagram

## 4.7 Database Modelling

### 4.7.1 E-R Diagram

### 4.7.2 Data Dictionary

### 4.7.3 Relational Schema

### 4.7.3.1 1st Normal Form

### 4.7.3.2 2nd Normal Form

### 4.7.3.3 3rd Normal Form(optional)

## 4.8 Algorithm Design

The various drowsiness detection system algorithm stages are discussed as:

### 4.8.1 Face Detection

For face detection it uses Haar feature-based cascade classifiers, which is an effective object detection method proposed by [15]. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

A cascaded Adaboost classifier with the Haar-like features is exploited to find out the face region. First, the compensated image is segmented into numbers of rectangle areas, at any position and scale within the original image. Due to the difference of facial feature, Haar-like feature is efficient for real-time face detection. These can be calculated according to the difference of sum of pixel values within rectangle areas. The features can be represented by the different composition of the black region and white region. A cascaded Adaboost classifier is a strong classifier which is a combination of several weak classifiers. Each weak classifier is trained by Adaboost algorithm. If a candidate sample passes through the cascaded Adaboost classifier, the face region can be found. Almost all of face samples can pass through and nonface samples can be rejected.

### 4.8.2 Eye detection

In the system we have used facial landmark prediction for eye detection. Facial landmarks are used to localize and represent salient regions of the face, such as:  
• Eyes  
• Eyebrows  
• Nose  
• Mouth  
• Jawline

Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. In the context of facial landmarks, our goal is detecting important facial structures on the face using shape prediction methods. Detecting facial landmarks is therefore a two-step process:

• Localize the face in the image.

• Detect the key facial structures on the face ROI.

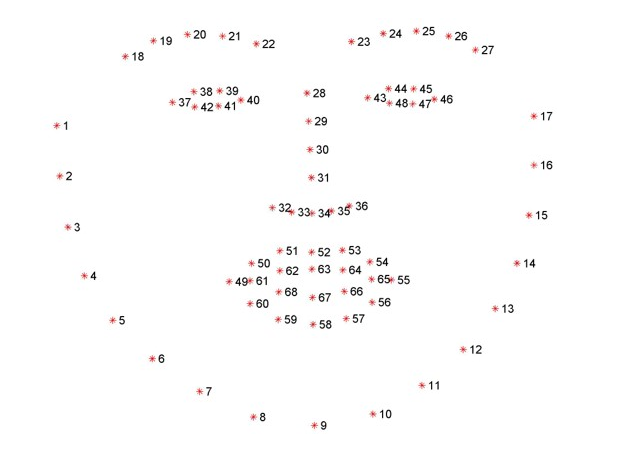
**Localize the face in the image**: The face image is localized by Haar feature-based cascade classifiers which was discussed in the first step of our algorithm i.e., face detection.

**Detect the key facial structures on the face ROI**: There are a variety of facial landmark detectors, but all methods essentially try to localize and label the following facial regions:  
• Mouth  
• Right eyebrow  
• Left eyebrow  
• Right eye  
• Left eye  
• Nose

The facial landmark detector included in the dlib library is an implementation of the ideas used by [16] for facial landmark detection. This method starts by using:

1. A training set of labelled facial landmarks on an image. These images are manually labelled, specifying specific (x, y*)*-coordinates of regions surrounding each facial structure.

2. Priors, of more specifically, the probability on distance between pairs of input pixels. The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face. The indexes of the 68 coordinates can be visualized on the image below:



#### Figure 4.8.2.1: The 68 facial landmark coordinates

In this picture you can see a training set of 68 labelled facial points with specific coordinates that surround certain parts of the face.

* JAWLINE POINTS: 1 – 17
* RIGHT EYEBROW POINTS: 17 – 22
* LEFT EYEBROW POINTS: 22 – 27
* NOSE BRIDGE POINTS: 27 – 31
* LOWER NOSE POINTS: 31 – 36
* RIGHT EYE POINTS: 36 – 42
* LEFT EYE POINTS: 42 – 48
* LEFT EYE POINTS: 42 – 48
* MOUTH INNER POINTS: 61 – 68

We can detect and access both the eye region by the following facial landmark index shown below as:

• The right eye using [36, 42].

• The left eye with [42, 48].

These annotations are part of the 68-point iBUG 300-W dataset which the dlib facial landmark predictor was trained on.

### 4.8.3 Recognition of Eye's State

The eye area can be estimated from optical flow, by sparse tracking or by frame-to-frame intensity differencing and adaptive thresholding and a decision is made whether the eyes are opened or closed. A different approach is to infer the state of the eye opening from a single image e.g., by correlation matching with open and closed eye templates, a heuristic horizontal or vertical image intensity projection over the eye region, a parametric model fitting to find the eyelids, or active shape models. A major drawback of the previous approaches is that they usually implicitly impose too strong requirements on the setup, in the sense of a relative face-camera pose (head orientation), image resolution, illumination, motion dynamics, etc. Therefore, we propose a simple but efficient algorithm to detect eye blinks by using a recent facial landmark detector. A single scalar quantity that reflects a level of the eye opening is derived from the landmarks. Finally, having a per-frame sequence of the eye-opening estimates, the eye blinks are found by an SVM classifier that is trained on examples of blinking and nonblinking patterns.

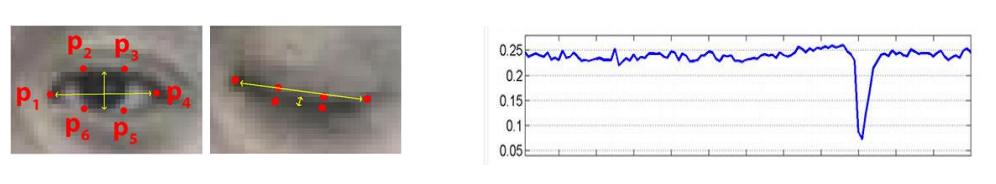
**Eye Aspect Ratio Calculation**

For every video frame, the eye landmarks are detected. The eye aspect ratio (EAR) between height and width of the eye is computed.

EAR = ||p2 − p6|| + ||p3 − p5|| (1)

2||p1 − p4||

where p1, . . ., p6 are the 2D landmark locations, depicted in Fig. 4.8.2. The EAR is mostly constant when an eye is open and is getting close to zero while closing an eye. It is partially person and head pose insensitive. Aspect ratio of the open eye has a small variance among individuals, and it is fully invariant to a uniform scaling of the image and in-plane rotation of the face. Since eye blinking is performed by both eyes synchronously, the EAR of both eyes is averaged.



#### Figure 4.8.3.1: Open and closed eyes with landmarks p(i) automatically detected for use in EAR calculations and EAR reading for one blink.

### 4.8.4 Eye State Determination:

Finally, the decision for the eye state is made based on EAR calculated in the previous step. If the distance is zero or is close to zero, the eye state is classified as “closed” otherwise the eye state is identified as “open”.

### 4.8.5 Drowsiness Detection:

The last step of the algorithm is to determine the person’s condition based on a pre-set condition for drowsiness. The average blink duration of a person is 100-400 milliseconds (i.e., 0.1-0.4 of a second). Hence if a person is drowsy his eye closure must be beyond this interval. We set a time frame of 5 seconds. If the eyes remain closed for five or more seconds, drowsiness is detected and the driver is alerted using an alarm sound.

## 4.9 Interface Design

This phase will produce the graphical user interface (GUI). This is used as the communication platform between the user and the system.

**User Interface Design Principles and Guidelines**

The system design follows the three **principles** for a user interface design below:

1. **Learnability**: The interface must be easy to learn and understand.
2. **Flexibility**: The interface must support a variety of interaction mechanism.
3. **Robustness**: The interface must provide proper feedback to let the user understand what is going on in the system.

**Guidelines**

The system corresponds to the following guidelines for designing the user interface:

* The user must be asked before any destruction action.
* The amount of information that must be remembered between two actions must be **minimal**.
* The number of inputs from the user required during interaction must be reduced.
* The commands that are irrelevant to the current action, must be deactivated.
* To categorize the type of activities different windows should be used.

## 4.10 Conclusion

The design phase of the system consisted of a detailed description of the proposed system in terms of its functionality in the real operating environment. The physical layout of the proposed system was illustrated to show a clear picture of how the software and hardware will interact. The design phase also included the system design models such as the activity and sequence diagram which gave a clear indication of what the system should do and the correct sequence of steps to be followed in the design phase.

# Chapter 5 Implementation

## 5.1 Introduction

This chapter presents the structured approach undertaken to implement the proposed system and the standards followed to ensure it achieves set objectives while fitting to the needs of the individual end user. It also outlines the stages taken to implement the systems, processes undertaken at each stage and conclusions ascertained.

## 5.2 Coding Conventions

**NB: See code samples in appendix section showing compliance to below Python conventions.**

### 5.2.1 Naming Conventions

**Naming Styles**

The table below outlines some of the common naming styles in Python code and when you should use them:

| **Type** | **Naming Convention** | **Examples** |
| --- | --- | --- |
| Function | Use a lowercase word or words. Separate words by underscores to improve readability. | function, my\_function |
| Variable | Use a lowercase single letter, word, or words. Separate words with underscores to improve readability. | x, var, my\_variable |
| Class | Start each word with a capital letter. Do not separate words with underscores. This style is called camel case. | Model, MyClass |
| Method | Use a lowercase word or words. Separate words with underscores to improve readability. | class\_method, method |
| Constant | Use an uppercase single letter, word, or words. Separate words with underscores to improve readability. | CONSTANT, MY\_CONSTANT, MY\_LONG\_CONSTANT |
| Module | Use a short, lowercase word or words. Separate words with underscores to improve readability. | module.py, my\_module.py |
| Package | Use a short, lowercase word or words. Do not separate words with underscores. | package, mypackage |

These are some of the common naming conventions and examples of how to use them. But in order to write readable code, you still have to be careful with your choice of letters and words. In addition to choosing the correct naming styles in your code, you also have to choose the names carefully. Below are a few pointers on how to do this as effectively as possible.

### 5.2.2 Blank Lines

Vertical whitespace, or blank lines, can greatly improve the readability of your code. Code that’s bunched up together can be overwhelming and hard to read. Similarly, too many blank lines in your code makes it look very sparse, and the reader might need to scroll more than necessary. Below are three key guidelines on how to use vertical whitespace.

**Surround top-level functions and classes with two blank lines.** Top-level functions and classes should be fairly self-contained and handle separate functionality. It makes sense to put extra vertical space around them, so that it’s clear they are separate.

**Surround method definitions inside classes with a single blank line.** Inside a class, functions are all related to one another. It’s good practice to leave only a single line between them.

**Use blank lines sparingly inside functions to show clear steps.** Sometimes, a complicated function has to complete several steps before the return statement. To help the reader understand the logic inside the function, it can be helpful to leave a blank line between each step.

### 5.2.3 Maximum Line Length and Line Breaking

PEP 8 suggests lines should be limited to 79 characters. This is because it allows you to have multiple files open next to one another, while also avoiding line wrapping.

Of course, keeping statements to 79 characters or less is not always possible. PEP 8 outlines ways to allow statements to run over several lines.

Python will assume line continuation if code is contained within parentheses, brackets, or braces:

### 5.2.4 Indentation

Indentation, or leading whitespace, is extremely important in Python. The indentation level of lines of code in Python determines how statements are grouped together.

The key indentation rules laid out by PEP 8 are the following:

* Use 4 consecutive spaces to indicate indentation.
* Prefer spaces over tabs.

### 5.2.5 Comments

You should use comments to document code as it’s written. It is important to document your code so that you, and any collaborators, can understand it. When you or someone else reads a comment, they should be able to easily understand the code the comment applies to and how it fits in with the rest of your code.

Here are some key points to remember when adding comments to your code:

* Limit the line length of comments and docstrings to 72 characters.
* Use complete sentences, starting with a capital letter.
* Make sure to update comments if you change your code.

### 5.2.6 Block Comments

Use block comments to document a small section of code. They are useful when you have to write several lines of code to perform a single action, such as importing data from a file or updating a database entry. They are important as they help others understand the purpose and functionality of a given code block.

PEP 8 provides the following rules for writing block comments:

* Indent block comments to the same level as the code they describe.
* Start each line with a # followed by a single space.
* Separate paragraphs by a line containing a single #.

## 5.3 Coding Strategy

## 5.4 Coding Review

### 5.4.1 Manual Checklist

The code was checked under the following key headings to ascertain whether it conforms to the following expectations and it passed all items with a **YES**:

1. **Structure**

* Does the code completely and correctly implement the design?
* Does the code conform to any applicable coding standards?
* Is the code well-structured, consistent in style, and consistently formatted?
* Are there any uncalled or unneeded procedures or any unreachable code?
* Are there any leftover stubs or test routines in the code?
* Can any code be replaced by calls to external reusable components or library functions?
* Are there any blocks of repeated code that could be condensed into a single procedure?
* Are any modules excessively complex and should be restructured or split into multiple routines?

1. **Variables**

* Are all variables properly defined with meaningful, consistent, and clear names?
* Do all assigned variables have proper type consistency or casting?
* Are there any redundant or unused variables?

1. **Loops and Branches**

* Are all loops, branches, and logic constructs complete, correct, and properly nested?
* Are the most common cases tested first in IF- -ELSEIF chains?
* Are loop termination conditions obvious and invariably achievable?
* Are indexes or subscripts properly initialized, just prior to the loop?
* Can any statements that are enclosed within loops be placed outside the loops?
* Does the code in the loop avoid manipulating the index variable or using it upon exit from the loop?

1. **Defensive Programming**

* Are indexes, pointers, and subscripts tested against array, record, or file bounds?
* Are imported data and input arguments tested for validity and completeness?
* Are all output variables assigned?
* Are the correct data operated on in each statement?
* Are timeouts or error traps used for external device accesses?
* Are files checked for existence before attempting to access them?
* Are all files and devices are left in the correct state upon program termination?

## 5.5 Conclusion

The implementation and evaluation process was conducted for the proposed system. All the objectives were met and all the correct coding conventions for the various languages used where followed. The coding strategy used involved extensive planning on most of the complex modules and trial and error on the other small components of the system. This was done in order to come up with a high quality and fully functional system. The code was also reviewed by various people to ensure the quality of the code.

# Chapter 6 Systems Testing

## 6.1 Introduction

In this chapter the various testing methods and results undertaken to ensure the system is performing as it should are done. Testing is done to discover errors in intended system performance. It is the process of evaluation in the software product to detect the differences between the given requirements and the expect output. It also assesses the quality of the software. There are various types of tests as shown below. Each test type addresses a specific testing requirement.

## 6.2 Testing Categories and Results

### 6.2.1 White Box

White Box Testing was done. The internal structure/ design/ implementation of the item system was made known to the tester. Mainly applicable to lower levels of testing especially Unit Testing. The basis for white box testing is Detail Design. The tester chose inputs to exercise paths through the code and determined the appropriate outputs. This testing has been uses to find in the following categories:

* Execute internal data structures to ensure their validity.
* Guarantee that all independent paths have been executed.
* Execute all logical decisions on their true and false sides.

Sample results from white box testing are given below.

*Table 1: White box test results*

|  |  |  |  |
| --- | --- | --- | --- |
| **Condition** | **Input data** | **Expected results** | **Observed results** |
| if key == ord("q"):              break      cap.release()      cv2.destroyAllWindows() | Keypress ‘q’ | Exit program | Exit program |
| if EAR<MIN\_EAR:                  COUNTER+=1                  if COUNTER>=EYE\_AR\_CONSEC\_FRAMES:                      if not ALARM\_ON:                          ALARM\_ON = True                          t = Thread(target=sound\_alarm,args=('alarm.wav',))                          t.daemon = True                          t.start()                          t.start() | Consecutive frame EAR < Set threshold | Sound alarm | Sound alarm |

### 6.2.2 Black Box

Black Box Testing was done. The internal structure/ design/ implementation of the system was **NOT** known to the tester. Mainly applicable to higher levels of testing: Acceptance Testing and System Testing. The basis for black box testing is Requirement Specifications. This testing has been used to find out the following categories:

* Initialization and termination errors.
* Incorrect or missing functions
* Performance errors

*Table 2: black box test results*

|  |  |
| --- | --- |
| **Code testing scenario** | **Results** |
| Initiate real-time camera input | Success |
| Manual override shutdown by administrator. | Success |
| Facial detection. | Success |
| Eye detection. | Success |
| Eye state monitoring. | Success |
| Drowsiness detection and sounding of alarm. | Success |

## 6.3 Types of Testing and Results

### 6.3.1 Functional Testing

Functional testing was used to test whether the system is performing required functions, which was found to be true .The system was tested for validation of input and user action. Essentially, this testing was carried out to ensure that the system accepted only the required input conditions and carried out the appropriate detection actions e.g., to ensure that drowsiness detection and sounding of alarm occurred when a facial region of interest had been ascertained (focusing on eye detection).

*Table 3: Functional testing results*

|  |  |
| --- | --- |
| **Functional requirements** | **Result** |
| Allow the administrator to log into the system and do a manual override to shut down the system. | Success |
| Allow driver to launch the camera. | Success |
| Allow camera to detect and monitor the facial region of interest of the driver. | Success |
| Allow classifier to keep track of eye state. | Success |
| Alert the driver when drowsiness has been detected (via sounding of alarm) | Success |
| Allow driver to stop the camera | Success |

### 6.3.2 Non-Functional Testing

Non-functional testing verifies the attributes of the system such as performance, usability, and robustness of the system. The tests carried out focused on testing these non-functional requirements which reflect on the quality of the system. The following test cases were made on the non-functional requirements of the system.

*Table 4: Non-functional testing results*

|  |  |  |
| --- | --- | --- |
| **Domain** | **Test Case** | **Results** |
| Efficiency | System should process the Eye tracking algorithm in less than 0.9 seconds. | Success |
| Performance | System should alert fatigued driver with an alarm after 5 seconds of detected drowsiness criteria satisfaction. | Success |
| Usability: | All user interface pages should be easily accessible. | Success |
| Security | Only the administrator can manually override the system to shut it down. | Success |

## 6.4 Test Cases

The main test cases performed on Drowsiness Detection system are of the objectives and other main operations that are expected to be performed by the system. During the test, sample users were tracked by the system to determine if it was able to detect drowsiness and sound the alarm only when the set conditions were met.

*Table 5: Test cases*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case # | Eyes Detected | Eye Closure | Expected Results | Actual Results | Status |
| 1 | Not Detected | Not Detected | No result. | No result. | Success |
| 2 | Not Detected | Not Detected | No result. | No result. | Success |
| 3 | Detected | Not Detected | No alarm | No alarm | Success |
| 4 | Detected | Detected | Alert sound alarm. | Alert sound alarm. | Success |

## 6.5 Levels of Testing and Results

### 6.5.1 Unit Testing

Unit testing was carried out on individual modules of the system to ensure that they are fully functional units. We did this by examining each unit, for example the Manual Override page. It was checked to ensure that it functions as required and that it authenticates the administrator credentials and allows the system to be shutdown manually. The success of each individual unit gave us the go ahead to carryout integration testing. All identified errors were dealt with. . Each component was tested individually without other system components. The following test cases were conducted.

*Table 6: Unit testing results*

|  |  |  |
| --- | --- | --- |
| **Function** | **Test case 1** | **Test case 2** |
| Initiate camera | Camera launched | Success |
| Manual override and shutdown system | Success | Success |
| Detect facial region of interest | Success | Success |
| Track eye state | Success | Success |
| Classify satisfaction of drowsiness criteria | EAR < Threshold | Success |
| Alert driver using an alarm sound | Success | Success |

### 6.5.2 Integration Testing

Integration is testing in which a group of components are combined to produce output. Also, the interaction between software and the hardware is tested in integration testing if software and hardware components have any relation. This testing was done to ensure if the camera was able to take the camera image input in real-time and feed it to the drowsiness detection system for classification.

*Table 7: Integration testing results*

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case Objective** | **Test Case Description** | **Expected Outcome** | **Result** |
| Check the interface link between front end login and back-end login. | Enter the username and password and click the Login button. | To be directed to the Manual override screen on verification from the backend. | Success |
| Check the navigation links to all the pages. | Press all the existing navigation links and check where they lead. | All the links should lead to the appropriate page. | Success |
| Check the launching of the camera | Press Launch Camera on the interface. | The camera should be launched, and immediate monitoring should occur. | Success |
| Check the stopping of the camera | Press the keyboard key ‘q’ to shut down the camera. | The camera should immediately shutdown. | Success |

### 6.5.3 Validation Testing

Validation Testing is the process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. It was done to ensure that the product met the project requirements. It was also done in order to demonstrate that the product fulfilled its intended use when deployed on the appropriate environment. All the system requirements were met and the system fulfilled its intended purpose.

*Table 8: Validation testing results*

|  |  |  |
| --- | --- | --- |
| **Domain** | **Expected Results** | **Actual Results** |
| Functional Testing | * The system modules should function as expected. * The system should be easily accessible and user friendly. * Error messages should be displayed on the system. | As expected |
| Integration Testing | All the integrated modules should work together flawlessly. | As expected |
| System Testing | All the components of the system should function properly. | As expected |
| Acceptance Testing | The system should meet user requirements and system objectives. | As expected |

### 6.5.4 Systems Testing

Complete and integrated software was tested. The reference document for this process is the requirements document, and the goal is to see if software meets its requirements. Here entire project has been tested against requirements of the project and it was checked whether all requirements of project have been satisfied or not. The system was ascertained to meet all specified user requirements.

*Table 9: System testing results*

|  |  |  |
| --- | --- | --- |
| **Domain** | **Expected Result** | **Actual Result** |
| Black Box Testing | The system should accept user input and produce the desired result. | As expected |
| Functional Testing | The system modules should function as expected and produce the desired output.  All the functional requirements should be met. | As expected |
| Non-functional Testing | All the non-functional requirements have been met.  The system is secure and efficient. | As expected |

### 6.5.5 Acceptance Testing

System was tested for acceptability. We evaluated the system’s compliance with the business requirements and assessed whether it is acceptable for delivery. The system under consideration was tested for user acceptance by constantly consulting with prospective system users at the time of developing and making changes whenever required. This was done with regard to the following points:

* Manual override screen design
* Output alert system design
* Menu driven design

*Table 10: Acceptance testing results*

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Test Result(Accepted or Rejected)** | **Comments** |
| The system should take camera input. | Accepted | The camera was able to launch and immediately start monitoring the driver. |
| The system should detect the facial region of interest and do nothing when no face is detected. | Accepted | The driver’s facial features were able to be detected and nothing was done when no face was detected. |
| The system should keep track of the state of the eyes in consecutive frames. | Accepted | Eye tracking is being done successfully. |
| The system should classify if drowsiness criteria has been satisfied. | Accepted | The classifier is adequately evaluating whether criteria for drowsiness has been satisfied. |
| The system should alert the driver in the event of satisfaction of drowsiness criteria by sounding an alarm sound. | Accepted | Alarm sound was initiated thus alerting the driver. |
| The system should allow an authorised administrator to login and manually override the system to shut it down in the event of malfunction. | Accepted | Administrator was able to login and deactivate the system. |

## 6.6 System Evaluation

This was done to assess the performance of the [completed system](http://www.modustrading.com/SevenElements.htm) and to discover how it is likely to perform in a live market environment. The developed solution successfully passed the diverse testing paradigms, and the sample users accepted the system. Most of the users agreed to the notion that the system is dependable, sustainable, and a useful solution that can be used to adequately warn drivers of impending drowsiness for the prevention of road accidents.

## 6.7 Conclusion

In this chapter, various tests were carried out from unit testing to acceptance testing. This was done in order to find out whether the developed system was accurate or not whilst taking into consideration how the system was developed by utilizing the different testing methods. All the objectives were met and the test cases were successful.

# Chapter 7 Conclusion

## 7.1 Introduction

## 7.2 Scope of Future Work

## 7.3 Recommendations

Bibliography

[1] Economic Commission for Africa, *UN rolls out Zimbabwe road safety performance review in bid to reduce carnage,* Communications Section*,* Addis Ababa, Ethiopia. Accessed on Nov 15, 2020. [Online]. Available: <https://www.uneca.org/stories/un-rolls-out-zimbabwe-road-safety-performance-review-bid-reduce-carnage>

[2] *Steering Wheel Behavior Based Estimation of Fatigue* Jarek Krajewski University of Wuppertal, Germany David Sommer University of Applied Sciences, Schmalkalden, Germany Udo Trutschel Circadian Technologies Inc., Stoneham, MA Dave Edwards Caterpillar Inc., Peoria, IL Martin Golz University of Applied Science, Schmalkalden, Germany. Mardi Z, Ashtiani SN, Mikaili M.

[3] Rukhsar Khan, Shruti Menon, Shivraj Patil, Suraj Anchan, Saritha L. R – *Human Drowsiness Detection System* [International Journal of Engineering and Advanced Technology (IJEAT) Volume-8 Issue-4, April, 2019].

[4] TecDay Real Life Safety, *ATTENTION ASSIST: Drowsiness-detection system warns drivers to prevent them falling asleep momentarily,* TecDay Real Life Safety*,* Leipzig, Germany. Accessed on Nov 16, 2020. [Online]. Available: [https](file:///C:\Users\pcRUX\Downloads\https)[://web.archive.org/web/20120226083842/http://media.daimler.com/dcmedia/0-921-658892-1-1147698-1-0-0-1147922-0-1-11702-0-0-1-0-0-0-0-0.html?TS=1266506682902](https://web.archive.org/web/20120226083842/http:/media.daimler.com/dcmedia/0-921-658892-1-1147698-1-0-0-1147922-0-1-11702-0-0-1-0-0-0-0-0.html?TS=1266506682902)

[5] Traffic Safety Council Of Zimbabwe, *TSCZ 2018,* Traffic Safety Council Of Zimbabwe*,* Harare, Zimbawe. Accessed on Nov 19, 2020. [Online]. Available: <http://www.trafficsafety.co.zw/index.php?option=com_content&view=article&id=70&Itemid=78>

[6] Transport Accident Commission, *Fatigue statistics,* State of Victoria*,* Geelong, Australia. Accessed on Nov 19, 2020. [Online]. Available: <http://www.tac.vic.gov.au/road-safety/statistics/summaries/>[fatigue-statistics](http://www.tac.vic.gov.au/road-safety/statistics/summaries/fatigue-statistics)

[7] J A Horne and L A Reyner, *Driver Sleepiness*, Journal of Sleep research Vol. 4, 1995

[8]Jim Horne and Louise Reyner, *Vehicle Accidents Related to Sleep*: A Review,  
Occupational Environmental Medicine, Vol. 56, 1999

[9] A. Rechtschaffen, “*Current perspectives on the function of sleep*,”  
Perspectives in Biology and Medicine, vol. 41, no. 3, pp.359–  
390,1998.

[10] S. B. Klein and B. M. Thorne, *Biological Psychology*, Worth Pub, 2007

[11] *EEG-based drowsiness detection for safe driving using chaotic features and statistical tests*. Journal of medical signals and sensors 2011; 1:130–137

[12] *EEG-Based Drowsiness Detection for Safe Driving Using Chaotic Features and Statistical Tests* Zahra Mardi, Seyedeh Naghmeh Miri Ashtiani, Mohammad

[13] Arun Sahayadhas, Kenneth Sundaraj and Murugappan Murugappan -- *Detecting Driver Drowsiness Based on Sensors: A Review* [ PMC Published online 2012 Dec 7].

[14]MikailiDanisman T, Bilasco IM, Djeraba C, Ihaddadene N. *Drowsy driver detection system using eye blink patterns. Machine and Web Intelligence* (ICMWI) IEEE 2010;230-233.

[15] Paul Viola, Michael Jones, "*Rapid Object Detection using a Boosted Cascade of Simple Features*", 2001

[16] Aigenexpert, *Detecting Facial Landmarks With OpenCV And DLIB,* Accessed on Feb 19, 2021. [Online]. Available: https://aigenexpert.com/blog/detecting-facial-landmarks-with-opencv-and-dlib/

Appendix A

APPENDIX I - User Manual

Appendix B

APPENDIX II - Project Proposal

Appendix C

APPENDIX III - Survey Paper

Appendix D

APPENDIX IV - Technical Paper