A PROJECT REPORT
ON

"REAL-TIME DRIVER-DROWSINESS DETECTION SYSTEM USING FACIAL FEATURES"

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

BACHELOR OF ENGINEERING
(COMPUTER ENGINEERING)

SUBMITTED BY

MR. HARSHAL CHAUDHARI	B150614221
MR. PRATIK NICHITE	B150614277
MR. ROHIT PAGAR	B150614284
MR. PRASAD JADHAV	B150614243

UNDER THE GUIDNACE OF
DR. VIVEK N. WAGHMARE

SANDIP
FOUNDATION

DEPARTMENT OF COMPUTER ENGINEERING

SANDIP INSTITUTE OF TECHNOLOGY AND RESEARCH CENTRE,
NASHIK – 422213

AY 2021-2022

SANDIP FOUNDATION

Certificate

This is to certify that the project report entitles

"Real-Time Driver-Drowsiness Detection System Using Facial Features"

Submitted by

Mr. Harshal Chaudhari	B150614221
Mr. Pratik Nichite	B150614277
Mr. Rohit Pagar	B150614284
Mr.Prasad Jadhav	B150614243

Are the bonafide student of this Institute and the work has been carried out under the supervision of **Dr. Vivek N. Waghmare** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

Dr. Vivek N. Waghmare
Guide

Dr. Amol Potgantwar
Head

Dr. S. T. Gandhe
Principal

Department of Computer Engineering,
Sandip Institute of Technology and Research Centre, Nashik

AY 2021-2022

SANDIP FOUNDATION

Certificate of Approval

This is to certify that the project report entitles

"Real-Time Driver-Drowsiness Detection System Using Facial Features"

Submitted by

Mr. Harshal Chaudhari	B150614221
Mr. Pratik Nichite	B150614277
Mr. Rohit Pagar	B150614284
Mr.Prasad Jadhav	B150614243

are the bonafide student of this Institute and the work has been carried out under the supervision of **Dr. Vivek Waghmare** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

Internal Examiner

Name:
Date:

External Examiner

Name:
Date:

DEPARTMENT OF COMPUTER ENGINEERING
Sandip Institute of Technology and Research Centre, Nashik
Nashik-422213

AY 2021-2022

ACKNOWLEDGEMENT

We would like to take this opportunity to thank **Dr. Vivek N. Waghmare** (Guide) for giving us all help and guidance we needed. We are really grateful to them for their king support. Their valuable suggestions were very helpful.

We are also grateful to **Dr. Amol Potgantwar,** Head of the Department of Computer Engineering, Sandip Institute of Technology and Research Centre for his indispensable support, suggestions.

Mr. Harshal Chaudhari
Mr. Pratik Nichite
Mr. Rohit Pagar
Mr. Prasad Jadhav

ABSTRACT

Drowsiness of the driver is one of the big reasons for road accidents from the past few years. This turns out to be a big problem not only for the driver but also for the people who use that road. With the improvement in technology, various accident prevention technologies are evolving. The primary objective of avoidance of road accidents can be achieved through real-time drowsiness detection of a driver using video capturing with face detection. After capturing and detecting the drowsiness by using a camera, the system triggers buzzer from an Arduino Nano board. The position of head and blinking of eyes are used as the features to detect whether the driver is drowsy or not.

Keywords: *Alarm, Android Studio, Convolutional Neural Network, Deep Learning, Drowsiness Detection, Eye Detection, Face Detection, Java, Machine Learning, Mobile Vision API, Supervised Learning, Unsupervised Learning.*

TABLE OF CONTENTS

LIST OF ABBREVATIONS i
LIST OF FIGURES ii
LIST OF TABLES iv

Sr. No.	Title of Chapter	Page No.
01	**Introduction**	1
1.1	Motivation	5
1.2	Objectives	5
1.3	Problem Statement	6
1.4	Problem Definition	6
1.5	Project Domain	6
02	**Literature Survey**	7
03	**Software Requirements Specification**	11
3.1	Introduction	11
3.1.1	Project Scope	11
3.1.2	User Classes and Characteristics	11
3.1.3	Assumptions and Dependencies	12
3.2	Functional Requirements	12
3.3	External User Interface Requirements	13
3.3.1	User Interface	13
3.3.2	Hardware Interface	13
3.3.3	Software Interface	13
3.4	Nonfunctional Requirements	14
3.4.1	Performance Requirements	14
3.4.2	Safety Requirements	14
3.4.3	Security Requirements	14
3.4.4	Software Quality Attributes	14
3.5	System Requirements	15
3.5.1	Hardware Requirements	15
5.5.2	Software Requirements	15
3.6	Analysis Model	16
3.7	System Implementation Plan	19
04	**System Design**	21
4.1	Introduction	21
4.2	System Architecture	22
4.3	Data Flow Diagrams	23
4.4	ER Diagram	25
4.5	UML Diagrams	26
05	**Implementation Details**	34
5.1	Methodology	34
5.2	Flowchart	49

	5.3	Results & Screenshots	50
	5.4	Result Analysis	53
06		**Other Specifications**	55
	6.1	Advantages	55
	6.2	Limitations	55
	6.3	Applications	55
07		**Conclusions and Future Work**	56
		References	57

LIST OF ABBREVATIONS

ABBREVIATION	ILLUSTRATION
NHTSA	National Highway Traffic Safety Administration
EEG	Electroencephalogram
ECG	Electrocardiogram
PLSR	Partial Least Square Regression
HP	Head Pose
EI	Eye Index
PA	Pupil Activity
HMM	Hidden Markov Model
SDLC	Software Development Life Cycle
CNN	Convolutional Neural Network
DFD	Data Flow Diagrams
ER	Entity Relation
UML	Unified Modeling Language
API	Application Programming Interface
EAR	Eye Aspect Ratio

LIST OF FIGURES

FIGURE	ILLUSTRATION	PAGE NO.
1.1	Driver Drowsiness Detection System	1
1.2	Face Tracking System	4
3.1	Waterfall Model	17
4.1	System Design Factors	21
4.2	Proposed Architecture	22
4.3	DFD – Level 0	23
4.4	DFD – Level 1	24
4.5	DFD – Level 2	24
4.6	Entity Relationship Diagram	25
4.7	Class Diagram	27
4.8	Component Diagram	28
4.9	Activity Diagram	29
4.10	Sequence Diagram	32
4.11	Use Case Diagram	33
5.1	Face Landmarks	36
5.2	Eye State Classification Using Eye Landmarks	37
5.3	Face Tracking	38
5.4	Open Eye Landmarks	40
5.5	Closed Eye Landmarks	40
5.6	Arduino Nano	42
5.7	Pin Diagram of Arduino Nano	43
5.8	Bluetooth Module (HC-05)	45
5.9	Bluetooth Module HC-05 Pin Diagram	46
5.10	Piezo Buzzer	48
5.11	Flowchart of the System	49
5.12	Home Screen	50

5.13	Drowsiness Detected Output	51
5.14	Proposed System Hardware	52
5.15	Graph of Result Analysis	54

LIST OF TABLE

TABLE	ILLUSTRATION	PAGE NO.
3.3.2	Hardware Interface	13
3.3.3	Software Interface	13
3.7.1	System Implementation Plan	20
5.1	Result Analysis	53

Chapter 1
Introduction

Drowsiness, simply defined as “a state of near sleep due to fatigue”. It is technically distinct from fatigue, which has been defined as a “disinclination to continue performing the task at hand”. The effects of sleepiness and fatigue are very much the same. Fatigue affects mental alertness, decreasing an individual’s ability to operate a vehicle safely and increasing the risk of human error that could lead to fatalities and injuries. Sleepiness slows reaction time, decreases awareness, and impairs judgment. Fatigue and sleep deprivation impact all transportation operators (for example: airline pilots, truck drivers, and railroad engineers). In both conditions, driver can’t focus on primary task of driving which may enhance the likelihood of crash occurrence. With the ever-growing traffic conditions, this problem will further deteriorate. For this reason, it is necessary to develop driver alertness system for accident prevention due to Driver Drowsiness as shown in Figure 1.1.

Figure 1.1: Driver Drowsiness Detection System

Interaction between driver and vehicle such as monitoring and supporting each other is one of the important solutions for keeping ourselves safe in the vehicles. Although active safety systems in vehicles have contributed to the decrease in the number of deaths occurring in traffic accidents, the number of traffic accidents is still increasing.

The National Highway Traffic Safety Administration (NHTSA) estimates that approximately 100,000 crashes each year are caused primarily by driver drowsiness or fatigue in the United States. In Japan, attention lapse, including that due to driving while drowsy, was the primary reason for traffic accidents in 2008. The Ministry of Economy, Trade and Industry in Japan reports that number of such accidents has increased 1.5 times in the 12 - year period from 1997 to 2008. Indian government also passed a law named 'Motor Bill' to improve safety on roads caused by driver drowsiness. The Bill is aimed at bringing down fatalities in road accidents by two lakhs in the first five years in a scenario where India reports around 5 lakh road accidents annually.

A suitable quantity of sleep is essential for a sound body to be active and healthy. Reduced or no sleep can lead to a variety of ailments, as well as tiredness and weariness. Sleep deprivation also signals a person's degree of activity. Sleeplessness can be caused by a variety of factors. Anxiety, arrhythmia, and other factors are possible reasons. Lack of a balanced diet is one of the key culprits. Sleep debt occurs when a person does not get enough sleep for two nights in a row. A person's tiredness and exhaustion are directly proportional to their sleep debt. Sleep deprivation or regressive exercise might cause drowsiness. Our bodies are stressed and muscle tissue is broken down during exercise. Our bodies require more energy to recover from this situation, therefore the energy that enters our brain is reduced, resulting in sleepiness.

Certain lifestyle factors may lead to increased drowsiness, such as working very long hours or switching to a night shift. Drowsiness can also be a result of your mental, emotional, or psychological state. Some medical conditions can cause drowsiness. One of the most common of these is diabetes. Many medications, particularly antihistamines, tranquilizers, and sleeping pills, list drowsiness as a possible side effect. Excessive drowsiness without a known cause can be a sign of a sleeping disorder. There's a range of sleeping disorders, and each has its own unique effects. A sudden onset of drowsiness can indicate a life-threatening condition, especially if it is related to a head injury, exposure to extreme cold, or a medication overdose.

Methods

One solution to this serious problem is the development of an intelligent vehicle that can predict driver drowsiness and prevent drowsy driving. The percentage of eyelid closure over the pupil over time (PERCLOS) is one of the major methods for the detection of the driver's drowsiness. Physiological measurements like electroencephalogram (EEG), electrocardiogram (ECG) [1], capturing eye closure, facial features [2] [3], or driving performance (such as steering characteristics, lane departure, etc.) [4], [5] are used for drowsiness detection. When drowsiness is detected while driving, audible sound, vibrations [6], [7], or warning messages on a display are generally used to warn the driver to concentrate on driving or to take a rest.

Face Tracking

Face Tracking Technology detects and tracks the presence of a human face in a digital video frame. This technology can be incorporated into computer and mobile applications, and can even be used in robotics. Face Tracking Technology can be used online or offline. Face Tracking enables the development of technologies such as face analysis and facial recognition.

When it comes to face analysis, Face Tracking makes it possible to:

- Follow a particular face as it moves within a video stream
- Count the number of people in a video frame or live video stream
- Determine the direction in which a face is looking
- Recognize facial expressions and perform sentiment analysis

Face Tracking can give greater accuracy compared to previous biometric recognition methods like iris and fingerprint recognition. It has also proven to be more secure and harder

to hack, making it increasingly important for use in security systems. It plays a powerful role, especially in applications such as access control.

How does Face Tracking work?

A face tracking camera captures video images that are transmitted to the Face Tracking software. The software then uses AI algorithms to detect faces. Once a face has been detected, the software is able to follow that face around within the video stream and to analyze facial features and expressions in real time.

Face tracking system as shown in Figure 1.2, must be robust to head movement, rotation, pose variation and illumination changes. To achieve this goal, we propose a method to use face detection and object tracking systems simultaneously. This combination gives us the opportunity to utilize advantages of two programs together.

Figure 1.2: Face Tracking System

Eye Detection

Locating the position of eye is difficult task due to many factors such as lighting condition, expression, facial shadowing, etc. Using eye features, different measures can be calculated with percentage of eyelid closure, maximum closure duration, blink frequency, average opening level of the eyes, opening velocity of the eyes, and closing velocity of the eye and an effective driver drowsiness detection model can be created which can work under

varying unconstraint and luminance conditions. After the position of face has been obtained, locating the eye can be done with better accuracy. As what was said before this method is not robust to pose variation. However, we use this disability as an advantage in distraction detection system. If eye could not be detected, we can assume that the driver is not looking at front. So this situation can be categorized in distraction state and must alarms the driver.

1.1 Motivation

The National Highway Traffic Safety Administration (NHTSA) estimates that approximately 100,000 crashes each year are caused primarily by driver drowsiness or fatigue in the United States. In Japan, attention lapse, including that due to driving while drowsy, was the primary reason for traffic accidents in 2008. The Ministry of Economy, Trade and Industry in Japan reports that number of such accidents has increased 1.5 times in the 12-year period from 1997 to 2008. Indian government also passed a law named 'Motor Bill' to improve safety on roads caused by driver drowsiness. The Bill is aimed at bringing down fatalities in road accidents by two lakhs in the first five years in a scenario where India reports around 5 lakh road accidents annually. Our Motive is to reduce accidents on the road due to sleepiness of drivers.

1.2 Objectives

The main objective, to overcome the problem related to accidents that related to drivers experiencing fatigue leads to a need to design a system that keeps the driver focused on the road. It also to prepare a prototype of real-time driver drowsiness detection system that alerts the driver when he is drowsy and sleepy. This is achieved by the use of Machine Learning model to detect the face of the driver using a camera and analyzing the state of the driver. Therefore, many designs and prototypes have been implemented in automobile to avoid such accidents by keeping whole focus and concentration on accurately monitoring the open and closed state of the driver's eye in real time. Same model and techniques can be

used for various other uses like Netflix and other streaming services can detect when the user is asleep. It can also be used in application that prevents user from sleeping.

1.3 Problem Statement

Many people drive Car, Truck, Bus and etc. for long distance which may lead lack of sleep. This suffering because of less sleep can cause accidents. To avoid such kind of accidents we build this system.

1.4 Problem Definition

Fatigue is a safety problem that has not yet been deeply tackled by any country in the world mainly because of its nature. 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 admit 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.

1.5 Project Domain

Deep learning and mobile vision

Chapter 2

Literature Survey

A Partial Least Squares Regression-Based Fusion Model for Predicting the Trend in Drowsiness, proposed by Hong Su and Gangtie Zheng [8]. They proposed a new technique of modeling driver drowsiness with multiple eyelid movement features based on an information fusion technique - partial least squares regression (PLSR), with which to cope with the problem of strong collinear relations among eyelid movement features and, thus, predicting the tendency of the drowsiness. It also provides a novel way of fusing multi-features together for enhancing our capability of detecting and predicting the state of drowsiness.

Another Camera-based Drowsiness Reference for Driver State Classification under Real Driving Conditions' approach proposed by Fabian Friedrichs and Bin Yang [9]. In this approach the driver's eyes are capable to detect drowsiness under simulator or experiment conditions. The performances of the latest eye tracking based in-vehicle fatigue prediction measures are evaluated. These measures are assessed statistically and by a classification method based on a large dataset of 90 hours of real road drives. The results show that eye-tracking drowsiness detection works well for some drivers as long as the blinks detection works properly. Even with some proposed improvements.

Limitations of existing system are overcome by M.J. Flores and et. al. [10] in 'Driver Drowsiness Detection System under Infrared Illumination for an Intelligent Vehicle'. In this approach an Artificial intelligence algorithms used to process the visual information in order to locate, track and analyze both the driver's face and eyes to compute the drowsiness and distraction indexes. This real-time system works during nocturnal conditions as a result of a near-infrared lighting system.

Wei Zhang and et. al. [11] proposed an approach for driver drowsiness recognition based on computer vision. They presented a nonintrusive drowsiness recognition method using eye-tracking and image processing. A robust eye detection algorithm is introduced to

address the problems caused by changes in illumination and driver posture. Six measures are calculated with percentage of eyelid closure, maximum closure duration, blink frequency, average opening level of the eyes, opening velocity of the eyes, and closing velocity of the eyes. Six participants in driving simulator experiments demonstrate the feasibility of this video-based drowsiness recognition method that provided 86% accuracy.

R. O. Mbouna and et. al. [12] presented visual analysis of eye state and head pose for continuous monitoring of alertness of a vehicle driver. Most existing approaches to visual detection of non-alert driving patterns rely either on eye closure or head nodding angles to determine the driver drowsiness or distraction level. The proposed scheme uses visual features such as eye index (EI), pupil activity (PA), and HP to extract critical information on non-alertness of a vehicle driver. Proposed scheme offers high classification accuracy with acceptably low errors and false alarms for people of various ethnicity and gender in real road driving conditions.

Eyosiyas Tadesse and et. al. [13] proposed a new method of analyzing the facial expression of the driver through Hidden Markov Model (HMM) based dynamic modeling to detect drowsiness. They have implemented the algorithm using a simulated driving setup. Experimental results verified the effectiveness of the proposed method.

Tiberiu Vesselenyi and et. al. [14] introduce the possibility to develop a drowsiness detection system for car drivers based on three types of methods: EEG and EOG signal processing and driver image analysis. In previous works the authors have described the researches on the first two methods. In this paper they have introduce the possibility to detect the drowsy or alert state of the driver based on the images taken during driving and by analysing the state of the driver eyes: opened, half-opened and closed. For this purpose, two kinds of artificial neural networks were employed: a 1 hidden layer network and an autoencoder network.

Arun Sahayadhas and et. al. [15] presented three measures as to the sensors used and discuss the advantages and limitations of each. The various ways through which drowsiness has been experimentally manipulated is also discussed. We conclude that by designing a

hybrid drowsiness detection system that combines non-intusive physiological measures with other measures one would accurately determine the drowsiness level of a driver. A number of road accidents might then be avoided if an alert is sent to a driver that is deemed drowsy.

Driver Drowsiness Detection System proposed by Chisty and et. al. [16] . They proposed Based on the type of data used, drowsiness detection can be conveniently separated into the two categories of intrusive and non-intrusive methods. During the survey, non-intrusive methods detect drowsiness by measuring driving behaviour and sometimes eye features, through which camera based detection system is the best method and so are useful for real world driving situations. This paper presents the review of existed drowsiness detection techniques that will be used in this system like Circular Hough Transform, FCM, Lab Color Space etc.

Sakshi Botwe and et. al. [17] proposed The best way to escape accidents caused by drivers' drowsiness is to detect drowsiness of the driver and warn him before fall into sleep. To detect drowsiness many method like eye retina detection, facial feature recognition has been used. Here in this paper, they propose a method of detecting driver drowsiness using eye retina detection and accident exposure of the driver. In this report, they propose a more precise drowsiness detection method which is a hybrid approach of eye retina detection and accident detection.

Development of drowsiness detection system developed by H. Ueno and et. al. [18]. They developed a system that uses image processing technology to analyze images of the driver's face taken with a video camera. Diminished alertness is detected on the basis of the degree to which the driver's eyes are open or closed. This detection system provides a noncontact technique for judging various levels of driver alertness and facilitates early detection of a decline in alertness during driving.

MarcoJavier Flores and et. al. [19] presented a new module for Advanced Driver Assistance System (ADAS) which deals with automatic driver drowsiness detection based on visual information and Artificial Intelligence is presented. The aim of this system is to locate, track, and analyze both the drivers face and eyes to compute a drowsiness index,

where this real-time system works under varying light conditions (diurnal and nocturnal driving). Examples of different images of drivers taken in a real vehicle are shown to validate the algorithms used.

Chapter 3

Software Requirements Specification

3.1 Introduction

With advancement in technology, transport industries are also evolving consistently and along with this rate of accidents are increasing. One of the major reasons for road accidents is drowsiness of the driver. It is becoming more important for us to understand and implement proper safety technologies to prevent accidents caused by drowsiness of the driver. So we aim to develop a prototype which will provide safety majors to avoid road accidents caused by drowsiness. Use of latest concepts of machine learning, deep learning and convolutional neural networks will prove to be cost effective and efficient.

3.1.1 Project Scope

The purpose of this project work is to provide a prototype of a real time driver's drowsiness detection system that alerts the driver when he is drowsy or sleepy.

This project is first deployed on android. The system will use phone's camera to take input which then we feed to the trained model from the Mobile Vision API. This trained model will detect the state of eye, then system will trigger an alarm from Arduino board if state of eye is closed for longer duration of time. For the future scope, this can be deployed in an embedded camera in a car with android system on it. Also various other features can be implemented depending on the requirements.

3.1.2 User Classes and Characteristics

User Classes:

- Driver

Characteristics:

- Face Detection
- Eye Tracking
- Drowsiness Detection
- Drowsiness Alert

3.1.3 Assumptions and Dependencies

- The device which is executing the application will have the required resources for the application to perform good.
- Another assumption is that the software and hardware components work the same way they worked while developing the project.
- The input video will have enough lighting so that the system can detect face and eyes which is the essential part needed to detect drowsiness.
- The quality of input will be good enough to be processed and not create any distortions while processing.

3.2 Functional Requirements

- REQ-1:
  System should be able to detect face

- REQ-2:
  System should be able to extract features from the face

- REQ-3:
  System should be able to classify state of the eye using features

- REQ-4:
  System should be able to calculate drowsiness

- REQ-5:
  System should be able to trigger alarm if drowsiness is detected

- REQ-6:
  System should be easy to use and should not have complicated user interface.

3.3 External User Interface Requirements

3.3.1 User Interface

1. Frontend: Android Application Using Java
2. Backend: Mobile Vision API

3.3.2 Hardware Interface

Device	Android
RAM	4 GB
Processor	Snapdragon 330 or any of similar/higher processing power
Camera	Phone's Camera
Controller Board	Arduino Nano
Sound Module	Piezo Buzzer
Communication Module	Bluetooth Module (HC05)

3.3.3 Software Interface

Operating System	Android (API version 9 or above)

3.4 Nonfunctional Requirements

3.4.1 Performance Requirements

1. The system detects the face and eye immediately.
2. System alerts the driver when eyes are closed for longer than usual eye blinking in real-time to avoid accidents.
3. System gives high accuracy and minimal time delay in drowsiness detection.

3.4.2 Safety Requirements

1. Nobody is harmed during development of the system.
2. System does not distract the driver while driving.
3. System is easy to use.

3.4.3 Security Requirements

1. System does not cause any distractions to the driver.
2. System does not harm the driver.

3.4.4 Software Quality Attributes

- Better performance
- Maximum accuracy
- Reliable
- Maintainable
- Scalable

3.5 System Requirements

3.5.1 Hardware Requirements

- CPU Speed: 2.4 GHz
- Operating System: Windows / Linux / MacOS
- RAM: 8GB
- Processor: I5 or higher
- Hard Disk Space: minimum 10GB
- Other Devices: Android device
- Controller Board: Arduino Nano
- Sound Module: Buzzer
- Other Modules: Bluetooth Module (HC05)

3.5.2 Software Requirements

- Programming Language: Java
- IDE for Application Development: Android Studio
- IDE for Arduino Programming: Arduino IDE

3.6 Analysis Model

SDLC Model to be applied

Waterfall Model:

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In “The Waterfall" approach, the process of software development is divided into phases. The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process inwhich progress is seen as owing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase mustbe completed before the next phase can begin and there is no overlapping in the phases.The Waterfall model is the earliest SDLC approach that was used for softwaredevelopment.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order. The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage. The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase

is complete. In this waterfall model, the phases do not overlap. The waterfall model is a sequential design process inwhich progress is seen as owing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. The classical waterfall model is the basic software development life cycle model. It is very simple but idealistic. Earlier this model was very popular but nowadays it is not used. But it is very important because all the other software development life cycle models are based on the classical waterfall model. This model is named "Waterfall Model", because its diagrammatic representation resembles a cascade of waterfalls.

Waterfall approach was first SDLC Model to be used widely in Software Engineeringto ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the nextphase sequentially. The following illustration is a representation of the different phases of the WaterfallModel.

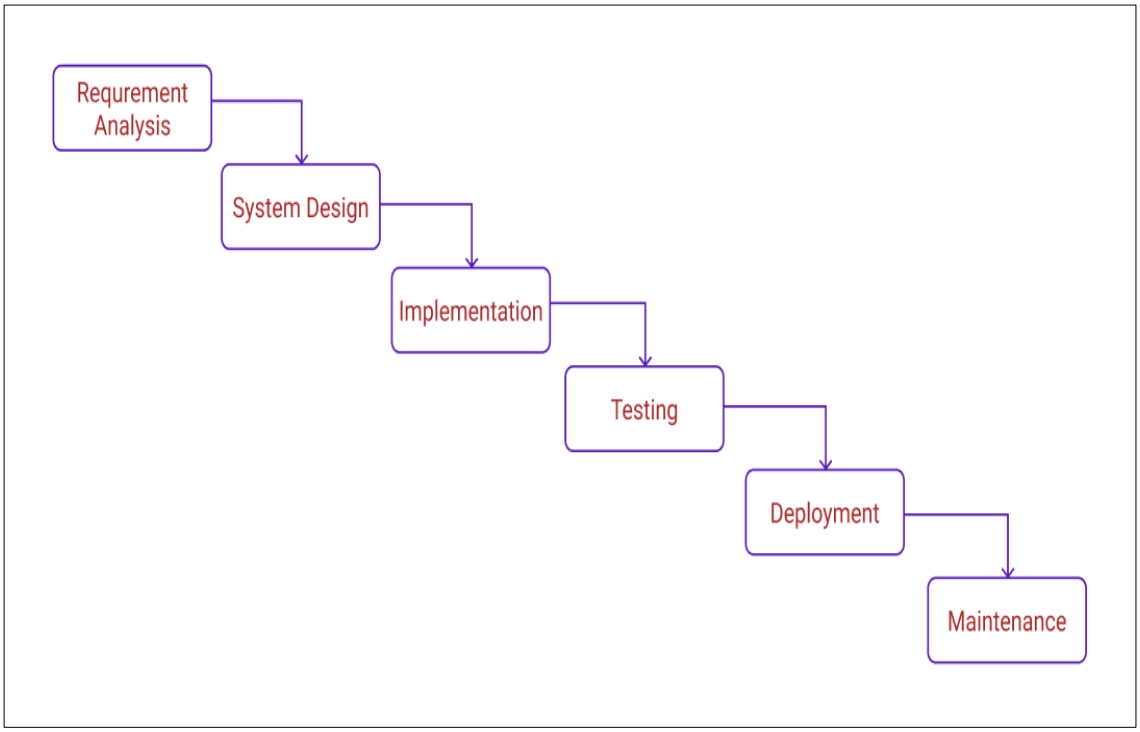

Figure 3.1: Waterfall Model

1. Requirement Analysis

This is the first phase of waterfall model in which all of the possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

This phase involves understanding what needs to design and what is its function, purpose, etc. Here, the specifications of the input and output or the final product are studied and marked.

2. System Design

In this phase system requirement specifications from the first phase are studied and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

3. Implementation

After system design phase is done, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

4. Testing

In this phase each unit from the implementation phase is tested and added to the system. The software designed, needs to go through constant software testing to find out if there are any errors. Testing is done so that the client does not face any problem during the installation or during the software being used.

5. Deployment

After testing the testing is done, the product is then deployed for the users to use. The deployment phase involves making the software live in the production/real environment after it tested for its tested thoroughly in the previous phase.

6. Maintenance

This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change re-quests initiated by the customer, or defects uncovered during live use of the system. The client is provided with regular maintenance and support for the developed software.

All these phases are cascaded to each other in which progress is seen as owing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for the previous phase and it is signed off, so the name "Waterfall Model".

3.7 System Implementation Plan

For developing the project, we have followed the phases of software development life cycle steps with planning schedule and performed the required tasks with respect to that planning schedule. As shown in table there are 6 phases which are Requirement Analysis, System Design, Implementation, Testing, Documentation and Deployment. We can manage the project to achieve the expected results and modify approaches as needed. Following table depicts the 6 phases followed with the timelines in days.

Table 3.7.1: System Implementation Plan

Phases	Task	Time (Days)
Phase 1	Information Gathering and Requirement Analysis	35
Phase 2	System Designing	35
Phase 3	Implementing System	60
Phase 4	Testing	25
Phase 5	Documentation	30
Phase 6	Project Finalization & Deployment	25
	Total	**210**

Chapter 4

System Design

4.1 Introduction

The process of establishing system aspects such as modules, architecture, components and their interfaces, and data for a system based on specified requirements is known as systems design. When faced with a System Design Problem, you should address it methodically. The problem may appear large at first glance, and one may be unsure where to begin. While designing a system, there is no one-size-fits-all solution. Following are the steps to approach a system design problem:

1. Breaking down the problem: When you're given a Design Problem, break it down into manageable pieces. These components might be Services or Features that you need to add to the system. A System that is being created may have a large number of features.
2. Communicate your ideas: Using flowcharts and diagrams, clearly demonstrate your design to the team members. Describe your ideas to your group members, including how you propose to solve the scalability challenge, how you will construct your database, and so on.

While designing system there are few things we need keep in mind. They are shown in the figure below:

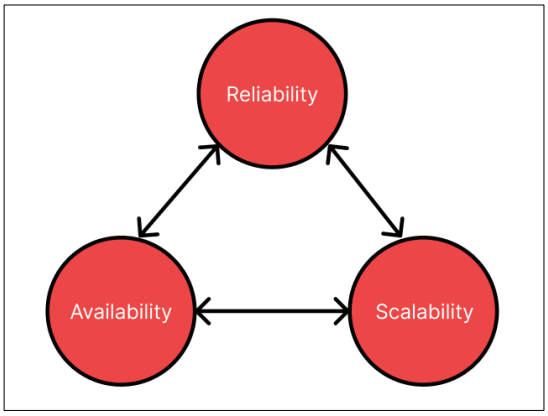

Figure 4.1: System Design Factors

4.2 System Architecture

System architecture is a conceptual model that describes the structure and behavior of multiple components and subsystems like multiple software applications, network devices, hardware, and even other machinery of a system. It is Architecture Description Language (ADL) which helps to describe the entire system architecture. System architecture can be broadly categorize into centralized and decentralized architectural organizations.

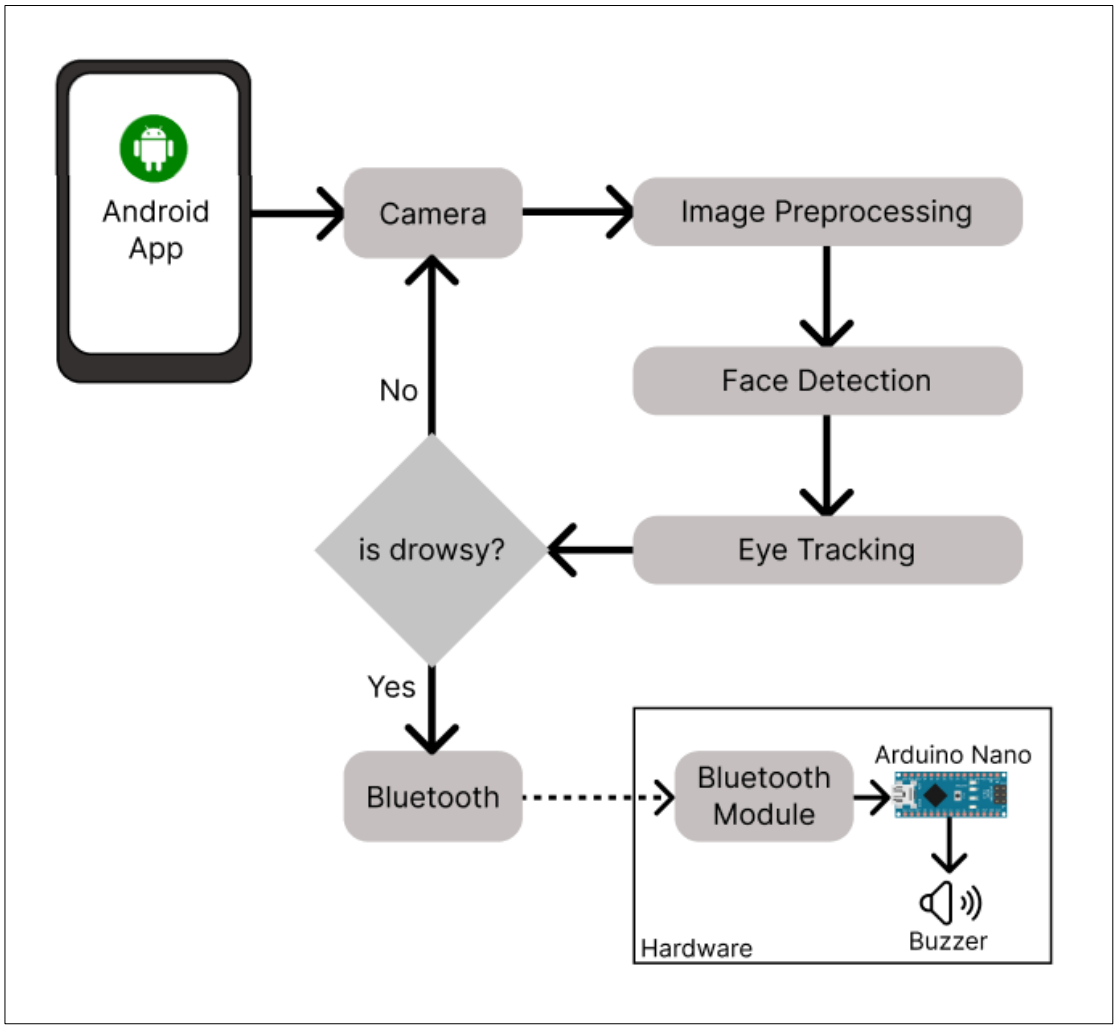

Figure 4.2: Proposed Architecture

4.3 Data Flow Diagrams

A data-flow diagram is a way of representing the flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. There are several levels of dataflow diagrams such as Level 0, Level 1 and Level 2. The higher the level of diagram the more processes, inputs and output it describes. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.

4.3.1 DFD - Level 0

Highest abstraction level DFD is known as Level 0 DFD, which depicts the entire information systems as one diagram concealing all the underlying details. Level 0 DFD are also known as context level DFDs.

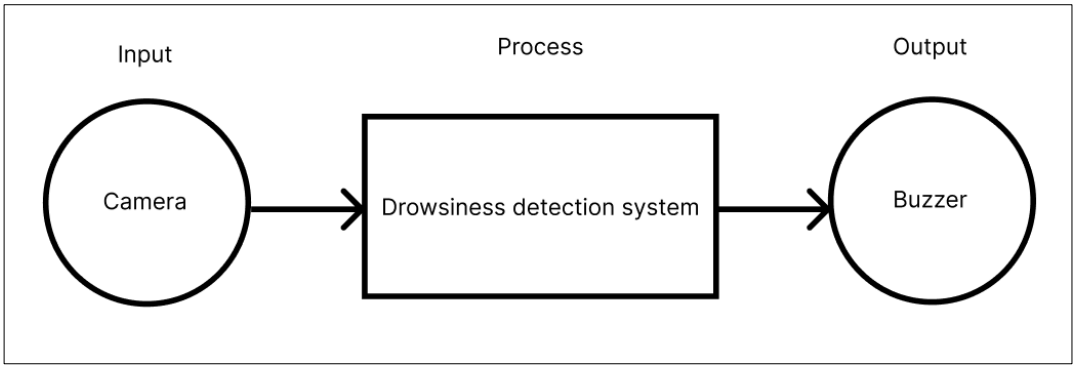

Figure 4.3: DFD Level 0

4.3.2 DFD – Level 1

The Level 0 DFD is broken down into more specific, Level 1 DFD. Level 1 DFD depicts basic modules in the system and flow of data among various modules. Level 1 DFD also mentions basic processes and sources of information. Here the context diagram is decomposed into multiple processes.

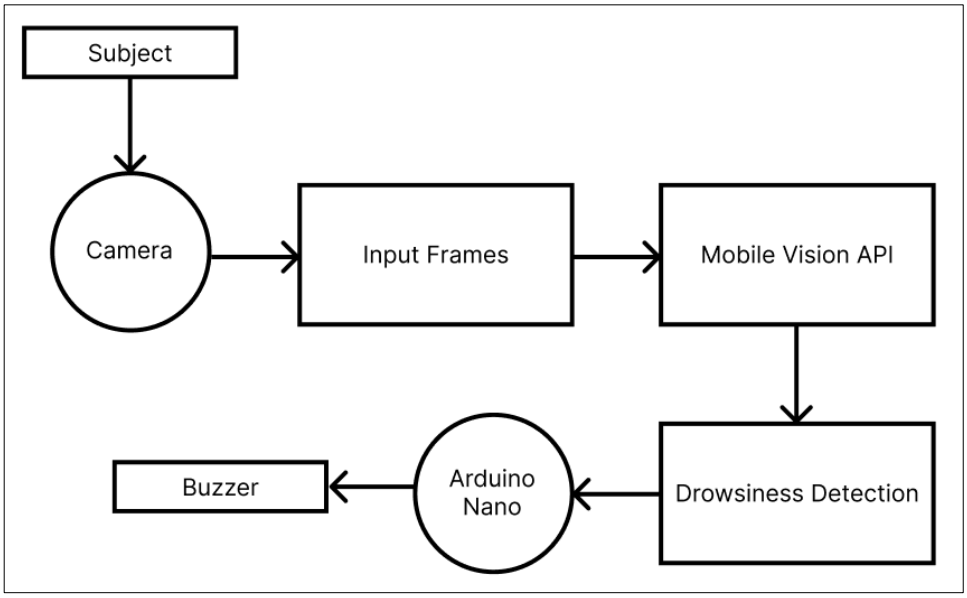

Figure 4.4: DFD Level 1

4.3.3 DFD – Level 2

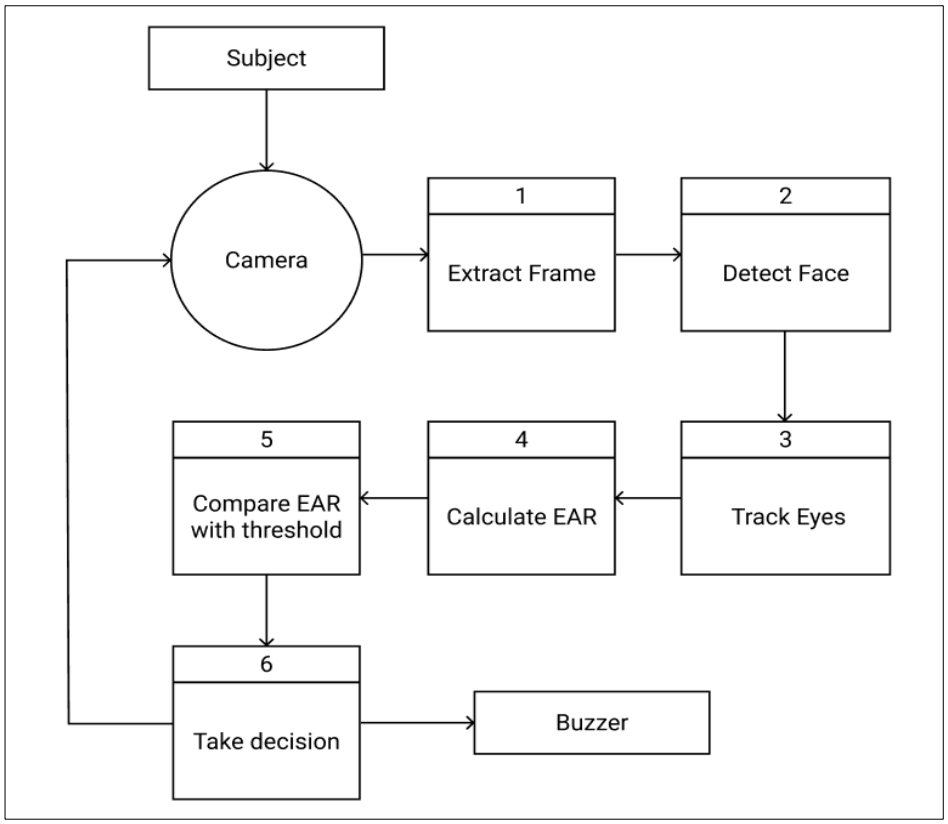

Figure 4.5: DFD Level 2

4.4 ER Diagram

ER-modeling is a data modeling method used in software engineering to produce a conceptual data model of an information system. Diagrams created using this ER-modeling method are called Entity-Relationship Diagrams or ER diagrams.

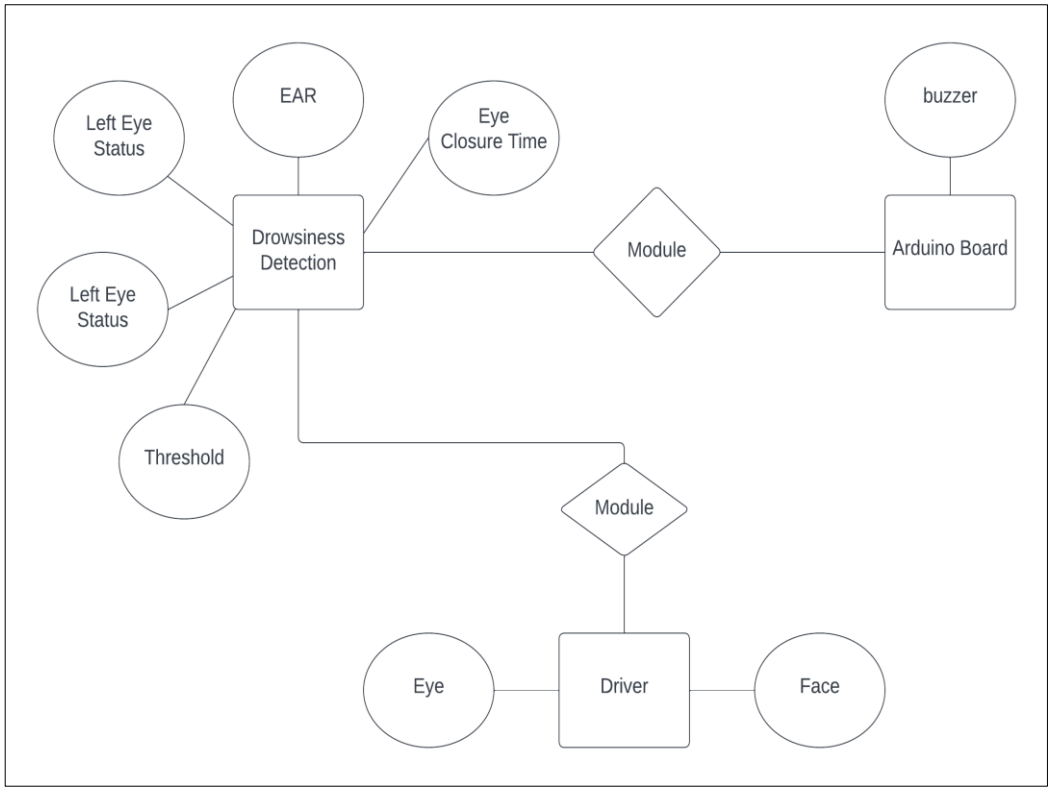

Figure 4.6: Entity Relationship Diagram

Components of ER diagram:

- **Entities:** An entity can be a real-world object, either animate or inanimate, that can be merely identifiable. An entity is denoted as a rectangle in an ER diagram.
- **Attributes:** Entities are denoted utilizing their properties, known as attributes. All attributes have values.
- **Relationships:** The association among entities is known as relationship. Relationships are represented by the diamond-shaped box.

4.5 UML Diagrams

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

The two broad categories that encompass all other types are Behavioral UML diagram and Structural UML diagram. As the name suggests, some UML diagrams try to analyze and depict the structure of a system or process, whereas other describe the behavior of the system, its actors, and its building components.

4.5.1 Structural Diagrams

4.5.1.1 Class Diagram

A class diagram depicts a class's properties and actions, as well as the system's constraints. Because class diagrams are the only UML diagrams that can be directly mapped with object-oriented languages, they are utilized generally in the modelling of object-oriented systems. A collection of classes, interfaces, associations, collaborations, and constraints are shown in a class diagram. It is also known as a structural diagram.

Class UML diagram is the most common diagram type for software documentation. Since most software being created nowadays is still based on the Object-Oriented Programming paradigm, using class diagrams to document the software turns out to be a common-sense solution. This happens because OOP is based on classes and the relations between them.

In a nutshell, class diagrams contain classes, alongside with their attributes (also referred to as data fields) and their behaviours (also referred to as member functions). More specifically, each class has 3 fields: the class name at the top, the class attributes right below the name, the class operations/behaviors at the bottom. The relation between different classes (represented by a connecting line), makes up a class diagram.

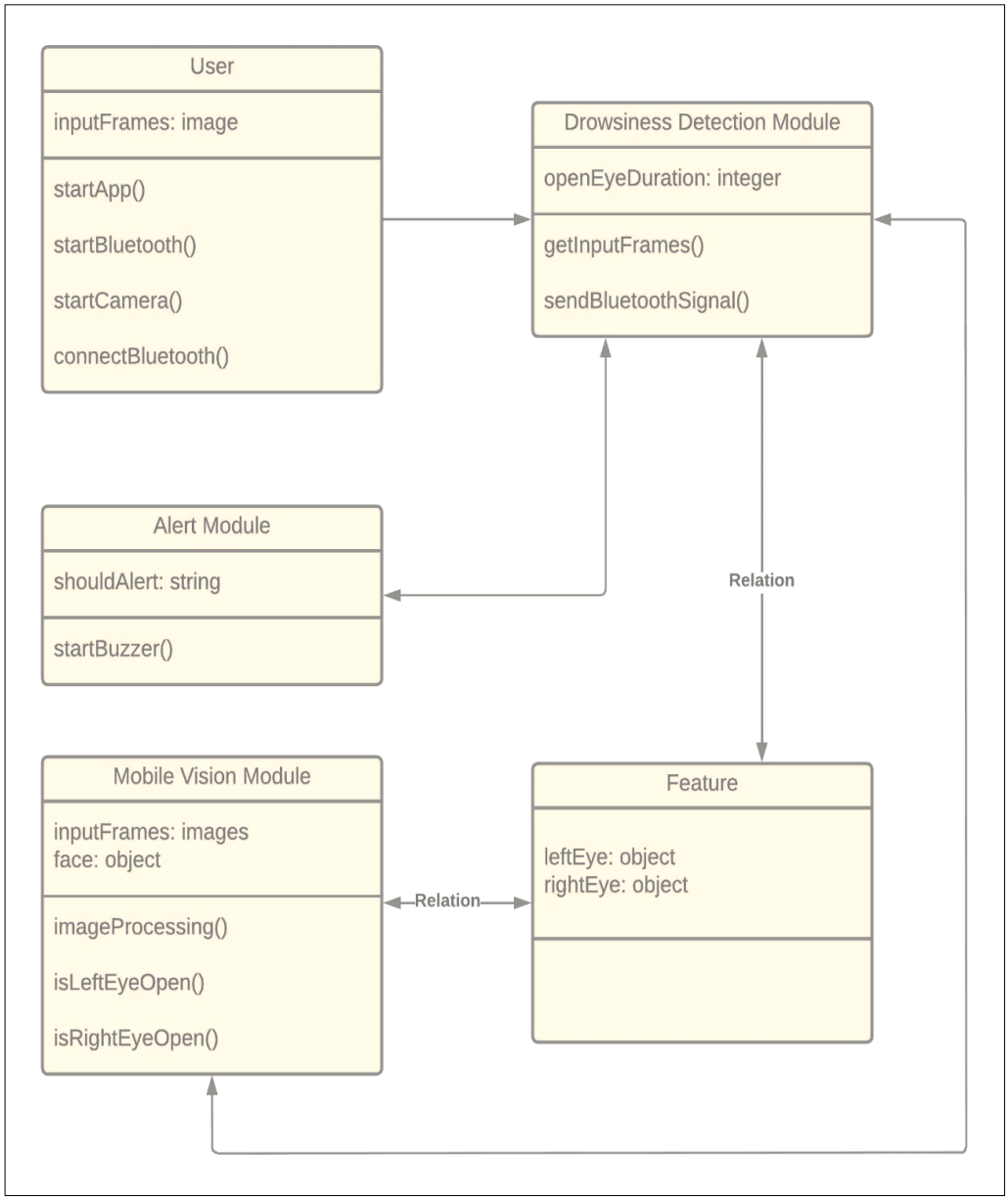

Figure 4.7: Class Diagram

4.5.1.2 Component Diagram

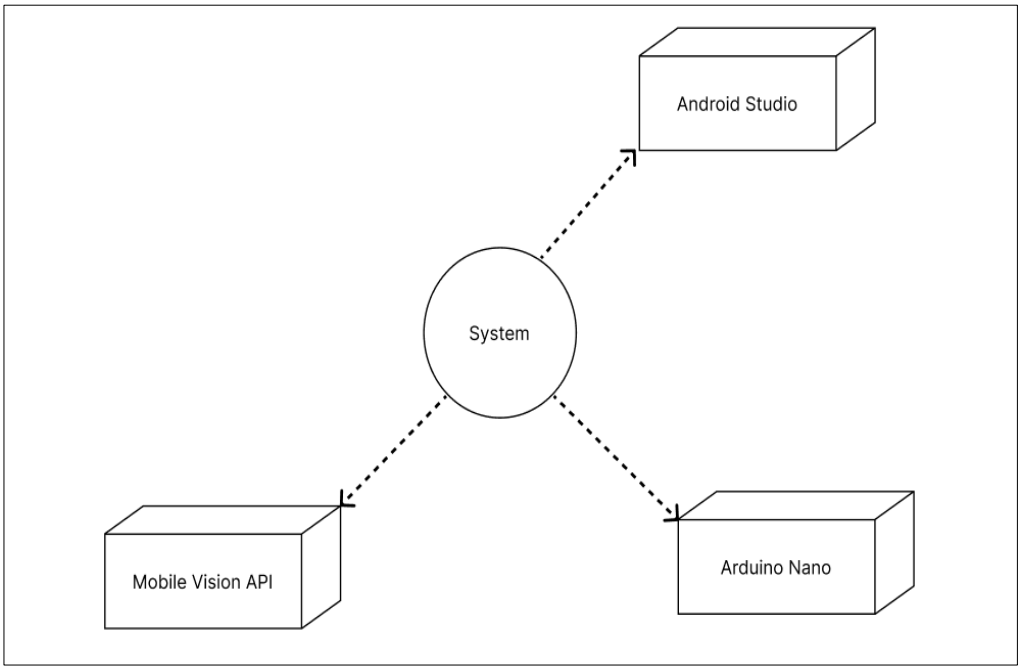

Figure 4.8: Component Diagram

When dealing with documentation of complex systems, component UML diagrams helps break down the system into smaller components. The above component diagram describes the main backend components of the system.

4.5.2 Behavioral Diagrams

4.5.2.1 Activity Diagram

Activity diagrams are probably the most important UML diagrams for doing business process modeling. In software development, it is generally used to describe the flow of different activities and actions. These can be both sequential and in parallel. They describe the objects used, consumed or produced by an activity and the relationship between the different activities. All the above are essential in business process modeling.

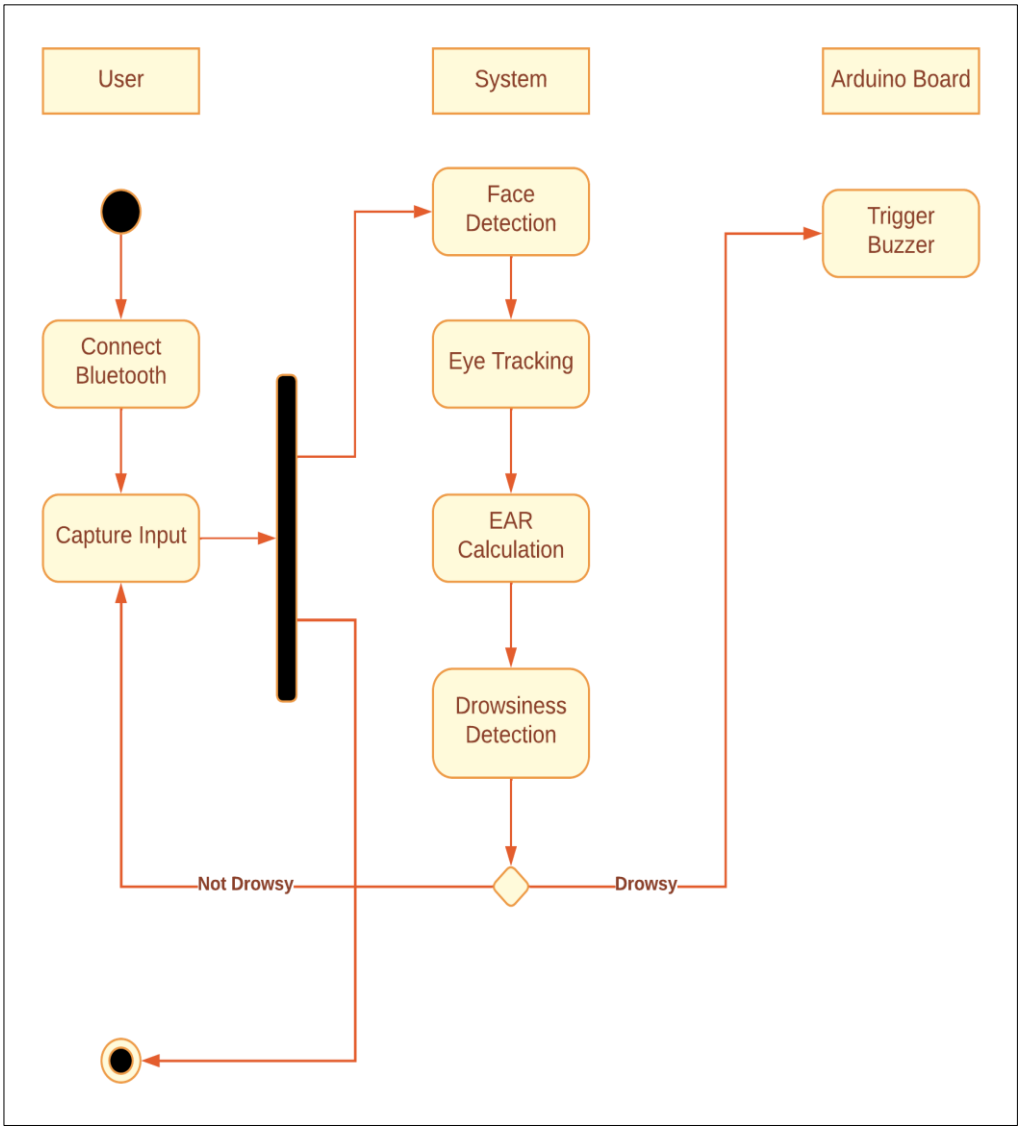

Figure 4.9: Activity Diagram

4.5.2.2 Sequence Diagram

The sequence diagram, also known as an event diagram, depicts the flow of messages through the system. It assists in visualising a variety of dynamic scenarios. It depicts communication between any two lifelines as a time-ordered series of events, with these lifelines participating at the same time. The message flow is represented by a vertical dotted line that extends across the bottom of the page in UML, whereas the lifeline is represented by a vertical bar. It encompasses both iterations and branching.

Notions used in sequence diagrams are as follows:

- **Lifeline**

An individual participant in the sequence diagram is represented by a lifeline. It is positioned at the top of the diagram.

- **Actor**

A role played by an entity that interacts with the subject is called as an actor. It is out of the scope of the system. It represents the role, which involves human users and external hardware or subjects. An actor may or may not represent a physical entity, but it purely depicts the role of an entity. Several distinct roles can be played by an actor or vice versa.

- **Activation**

It is represented by a thin rectangle on the lifeline. It describes that time period in which an operation is performed by an element, such that the top and the bottom of the rectangle is associated with the initiation and the completion time, each respectively.

- **Messages**

The messages depict the interaction between the objects and are represented by arrows. They are in the sequential order on the lifeline. The core of the sequence diagram is formed by messages and lifelines.

Following are types of messages enlisted below:

- **Call Message:** It defines a particular communication between the lifelines of an interaction, which represents that the target lifeline has invoked an operation.
- **Return Message:** It defines a particular communication between the lifelines of interaction that represent the flow of information from the receiver of the corresponding caller message.
- **Self Message:** It describes a communication, particularly between the lifelines of an interaction that represents a message of the same lifeline, has been invoked.
- **Recursive Message:** A self message sent for recursive purpose is called a recursive message. In other words, it can be said that the recursive message is a special case of the self message as it represents the recursive calls.
- **Create Message:** It describes a communication, particularly between the lifelines of an interaction describing that the target (lifeline) has been instantiated.
- **Destroy Message:** It describes a communication, particularly between the lifelines of an interaction that depicts a request to destroy the lifecycle of the target.
- **Duration Message:** It describes a communication particularly between the lifelines of an interaction, which portrays the time passage of the message while modeling a system.

- **Note**

A note is the capability of attaching several remarks to the element. It basically carries useful information for the modelers.

- **Sequence Fragments**

1. Sequence fragments have been introduced by UML 2.0, which makes it quite easy for the creation and maintenance of an accurate sequence diagram.

2. It is represented by a box called a combined fragment, encloses a part of interaction inside a sequence diagram.
3. The type of fragment is shown by a fragment operator.

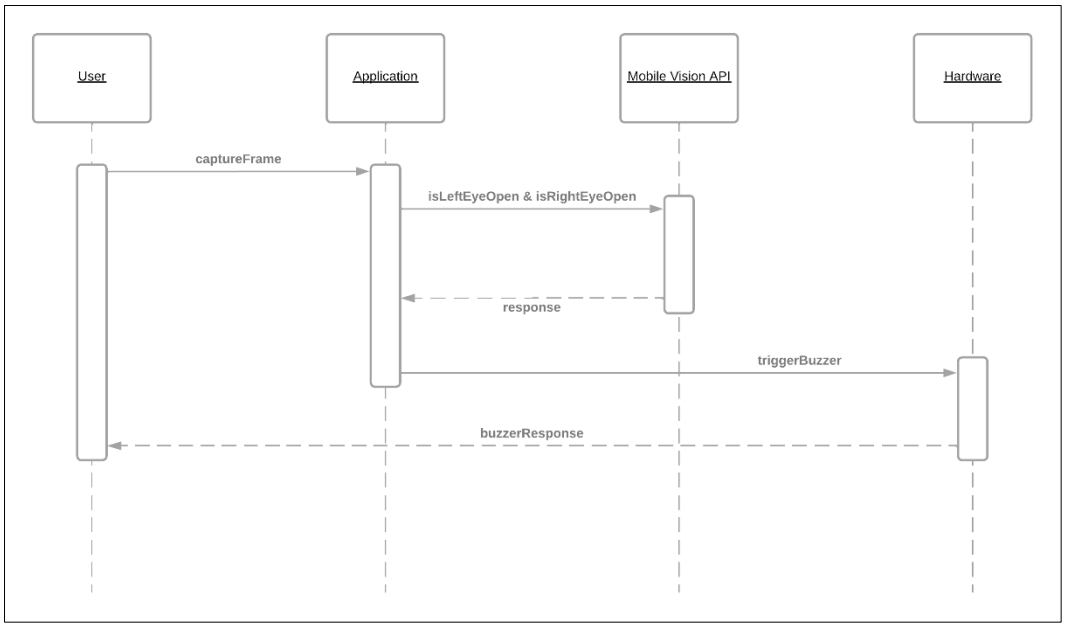

Figure 4.10: Sequence Diagram

4.5.2.3 Use Case Diagram

Use Case diagrams are used to analyze the system's high-level requirements. These requirements are expressed through different use cases. There are three main components of this UML diagram:

- **Functional requirements –** represented as use cases; a verb describing an action.
- **Actors –** they interact with the system; an actor can be a human being, an organization or an internal or external application
- **Relationships between actors –** represented using straight arrows.
- **Use cases –** represented using eclipses.

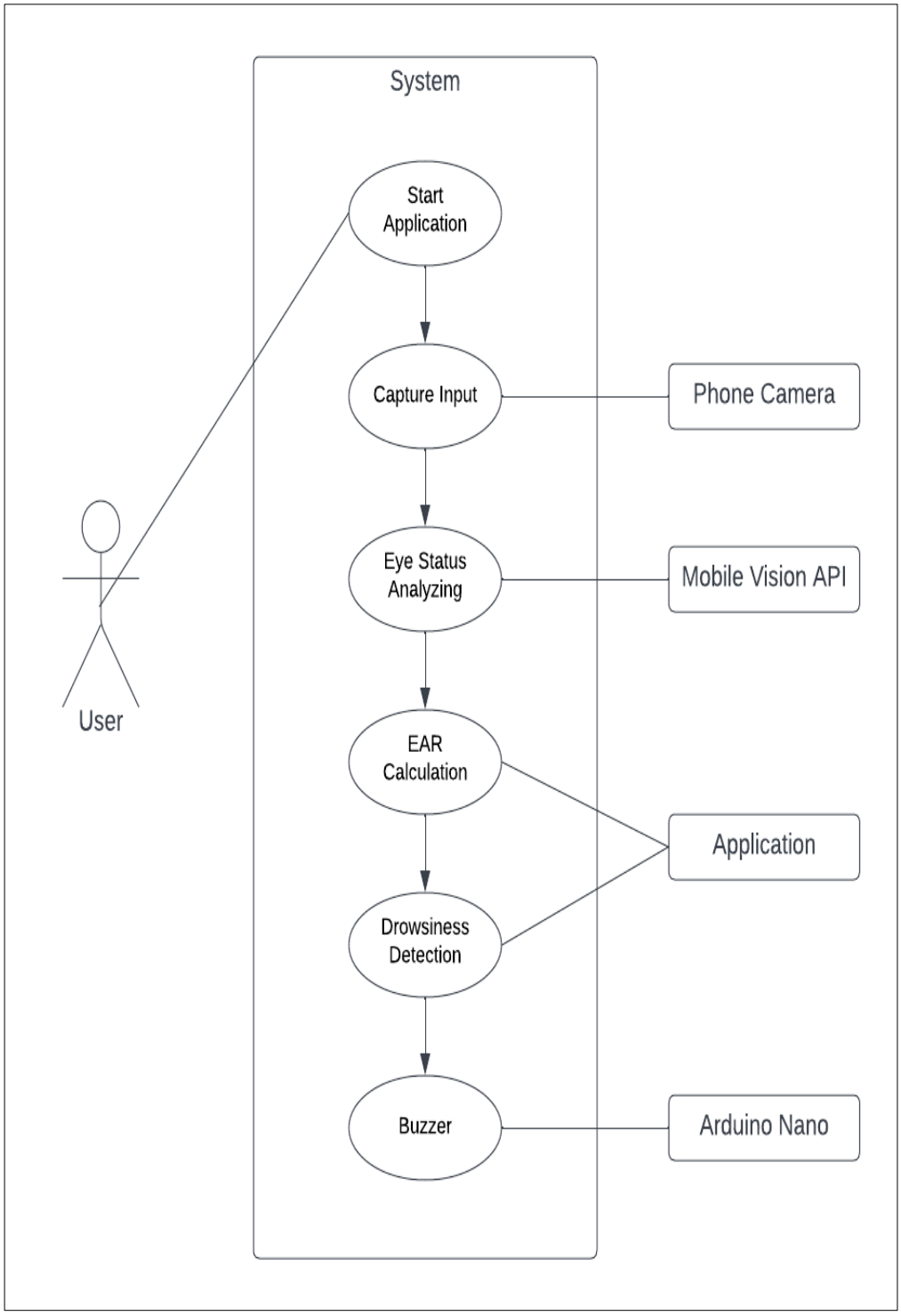

Figure 4.11: Use Case Diagram

Chapter 5
Implementation Details

5.1 Methodology

- The methodology of the proposed system for eye tracking, eye closure detection and drowsiness detection is described.
- The system will help significantly in detection of drowsiness while driving vehicles.
- The whole architecture can be divided into several phases such as:
    - Development of the system to capture input frames.
    - Face detection and eye blink tracking.
    - Drowsiness detection.
    - Alerting driver.

5.1.1 System Development

This section describes the implementation details and programming techniques that have been used to enable the system to work effectively. In this work, an application has been proposed for the same purpose of drowsiness detection using a smartphone to process a stream of picture frames for facial analysis. The application will timely and consistently monitor driver fatigue while the person is driving by checking the duration of eye blinks. The application alerts the driver in case of positive detection of drowsiness analyzed before and while driving. The project aims at using Arduino Nano as external hardware and a Bluetooth HC 05 module and a buzzer module.

The application is built using Java. Android studio is an Integrated Development Environment (IDE) for Android's operating system which is constructed on JetBrain's IntelliJ IDEA software and is a flexible Gradle-based build system. It allows developers to build and test applications in various devices with a feature- rich emulator and allows programming in Java and even Kotlin in Android Studio 4.0 or late.

The system developed collects the input frames and then analyzes the frames using the Mobile Vision API. For face recognition, we should use an image with dimensions of at least

480x360 pixels. For Mobile Vision API to accurately detect faces, input images must contain faces that are represented by sufficient pixel data. In general, each face we want to detect in an image should be at least 100x100 pixels. If we want to detect the contours of faces, Mobile Vision API requires higher resolution input: each face should be at least 200x200 pixels. Poor image focus can also impact accuracy.

5.1.2 Face detection and eye blink tracking

Face detection locates human faces in visual media such as digital images or video. When a face is detected it has an associated position, size, and orientation; and it can be searched for landmarks such as the eyes and nose.

Face tracking extends face detection to video sequences. Any face that appears in a video for any length of time can be tracked from frame to frame. face tracking only makes inferences based on the position and motion of the faces in a video sequence.

A landmark is a point of interest within a face. The left eye, right eye, and base of the nose are all examples of landmarks. Mobile Vision API provides the ability to find landmarks on a detected face.

A contour is a set of points that follow the shape of a facial feature. Mobile Vision API provides the ability to find the contours of a face.

Classification determines whether a certain facial characteristic is present. For example, a face can be classified by whether its eyes are open or closed, or if the face is smiling or not. Classification is a certainty value. It indicates the confidence that a facial characteristic is present. Both of these classifications rely upon landmark detection.

After collecting input images, we then use face detector from Mobile Vision API to detect face from the input image.

- **Mobile Vision API:**

Mobile Vision is an API which help us to find objects in photos and video, using real-time on-device vision technology by Google. This framework has the capability to detect objects in photos and videos.

This is the most powerful API in all, as it has human face detection capabilities. It is a perfectly suited for any face filter or camera app, as it can perform the analysis on the device itself once the package is downloaded. Interestingly it not only recognizes a face but can also extract the facial features of that face, including eyes nose and mouth etc. Since this API as of now does not support face recognition it cannot identify similarity between two faces, but can still classify the features like, if the eyes are open or not. The functionalities that Mobile Vision Face Detection supports are:

- **Landmark Detection:**

  Face API understands the human face in terms of landmarks. When a face is scanned via this API, it identifies that face via landmarks. In simple terms face landmarks are: nose, mouth, left eye, and right eye etc. By using this API, we can actually extract the position of all these landmarks. Following figure shows the landmarks detected by the face API.

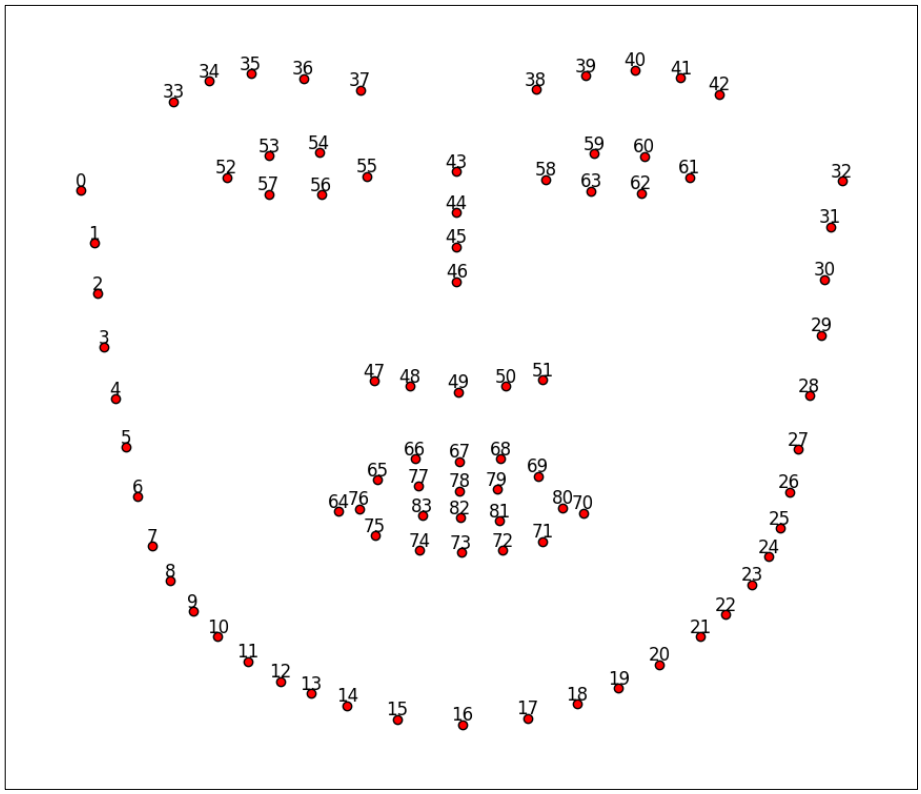

Figure 5.1: Face Landmarks

- **Classification:**

  This API does not only scan a face but can also apply some basic logic and identify certain characteristics on the scanned face. For e.g. with this feature we can find out whether the face has its eyes open or not. Also we can find out the probability of a smile on that face. The following figure depicts the classification of closed and open eyes using landmarks.

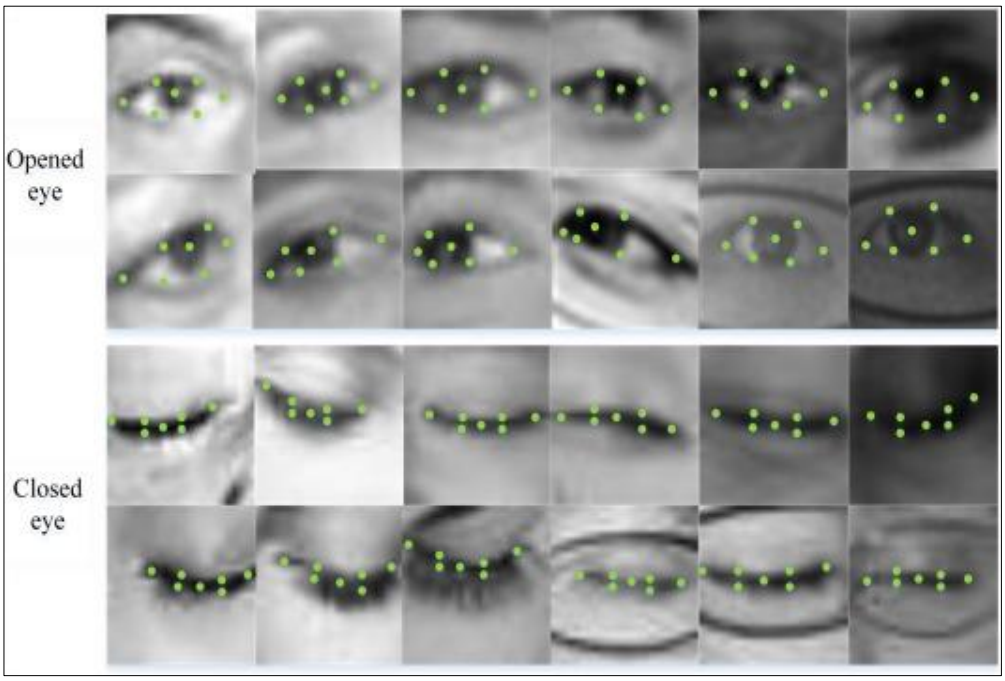

Figure 5.2: Eye State Classification Using Eye Landmarks

- **Tracking:**

  This is the most interesting feature of this API, we can actually track a face in a video sequence through this API. This feature of Mobile Vision library can be used to identify and track a face in a video. It tracks the face through movement of that particular face in the video.

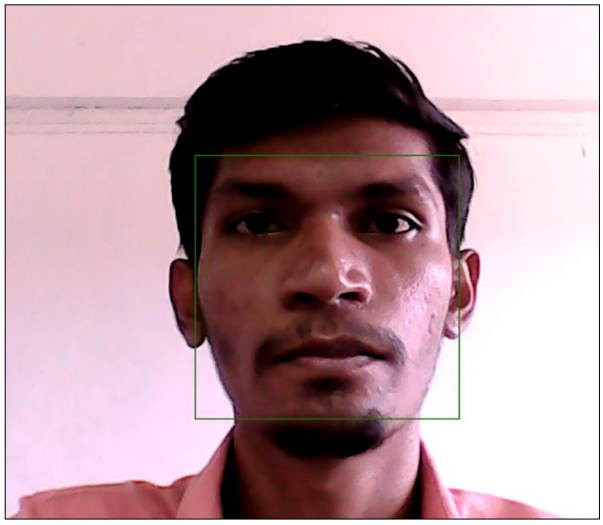

Figure 5.3: Face Tracking

The Mobile Vision uses models that are trained on large datasets. This interns increases accuracy of extracting features from the input data. There are different categories of models depending upon the learning method that are used to train the models:

- **Supervised Learning:** Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labeled data. Even though the data needs to be labeled accurately for this method to work, supervised learning is extremely powerful when used in the right circumstances.

  In supervised learning, the ML algorithm is given a small training dataset to work with. This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with. The training dataset is also very similar to the final dataset in its characteristics and provides the algorithm with the labeled parameters required for the problem.

  The algorithm then finds relationships between the parameters given, essentially establishing a cause and effect relationship between the variables in the dataset. At the end of the training, the algorithm has an idea of how the data works and the relationship between the input and the output.

This solution is then deployed for use with the final dataset, which it learns from in the same way as the training dataset. This means that supervised machine learning algorithms will continue to improve even after being deployed, discovering new patterns and relationships as it trains itself on new data.

- **Unsupervised Learning:** Unsupervised machine learning holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.

  In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures. Relationships between data points are perceived by the algorithm in an abstract manner, with no input required from human beings.

  The creation of these hidden structures is what makes unsupervised learning algorithms versatile. Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures. This offers more post-deployment development than supervised learning algorithms.

5.1.3 Drowsiness Detection

In order to detect the drowsiness, we need to calculate the eye blinks which we can do so using the provided methods by Mobile Vision API. The methods are as follows:

- getIsLeftEyeOpenProbability: this method gives probability of left eye being open.
- getIsRightEyeOpenProbability: this method gives probability of right eye being open.

We then compare this probability with the threshold value to get Eye Aspect Ratio.

- **Algorithm to detect drowsiness**

  - Starting at the left-corner of the eye and going clockwise around the rest of the region, each eye is represented by six coordinates as shown in the following figure.

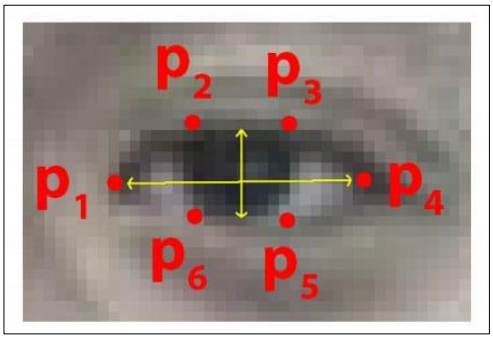

Figure 5.4: Open eye landmarks

  - These coordinates have a relationship between their width and height. The eye aspect ratio (EAR) can then be calculated using the equation. Where p1,p2,… p6 are facial landmarks.
  - The distance between vertical eye landmarks is computed in the numerator, while the distance between horizontal eye landmarks is computed in the denominator, with the denominator weighted correctly because there is only one set of horizontal points but two sets of vertical ones.
  - When the eye is open, the aspect ratio is roughly constant, but when the eye blinks, it quickly drops to zero.

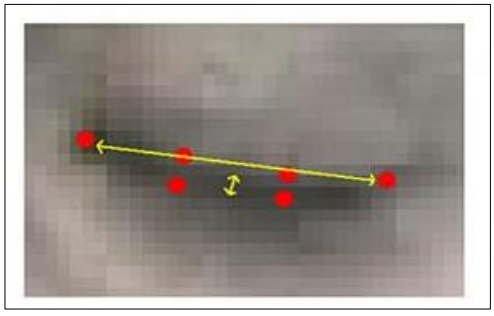

Figure 5.5: Closed eye landmarks

- The Mobile Vision API calculates the eye open probability which we then compare with the threshold value. If the probability of being eyes open falls below 5% then we can safely say that the person has closed his eyes.
- We calculate this for both left and right eye to detect drowsiness.
- When the eye is closed we start counting the time duration for which eyelid is closed. To determine whether the blinking pattern is normal or drowsy.
- We have set a maximum time limit of 2 seconds which is enough to be able to calculate drowsiness of the driver.
- When the eyelid closure durations hit 2 seconds we generate a bluetooth signal that will be transmitted through bluetooth to the hardware part for alerting the driver.

5.1.4 Alerting Driver

After detecting drowsiness there comes the main part of alerting the driver. We are making use of hardware for this task since it is more reliable. Taking hardware part into consideration, we have an Arduino board to process the input signal, a bluetooth module to receive the signal and a buzzer to alert the driver.

- **Arduino Nano:**

  After detecting drowsiness using the eye landmarks and calculating Eye Aspect Ratio, the application needs to alert the driver. This is done by sending bluetooth signal to a controller board called Arduino Nano through the bluetooth of the android device in which the application is running.

  The below figure shows the Arduino Nano board. Arduino Nano is a small, compatible open-source electronic development board based on an 8-bit AVR microcontroller. Two versions of this board are available, one is based on ATmega328p, and the other on Atmega168.

  Arduino Nano is the cheap controller board that can be used in small projects or for light weight jobs. Arduino Nano is the cheapest of all other controller boards such as Arduino UNO etc.

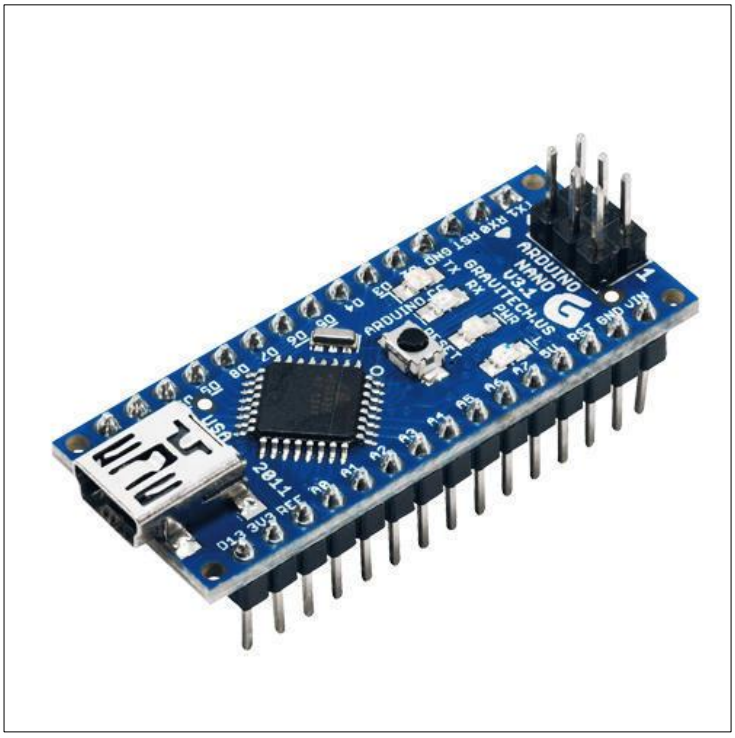

Figure 5.6: Arduino Nano

Arduino Nano can perform some functions similar to other boards available in the market, however, it is smaller in size and is a right match for projects requiring less memory space and fewer GPIO pins to connect with.

Like other Arduino boards, the operating voltage of this device is 5V, while input voltage ranges between 6V to 20V while the recommended input voltage ranges from 7V to 12V.

All Arduino boards can be programmed using Arduino IDE (Integrated Development Environment) Software - An official software introduced by Arduino. cc. All you need is a code to burn into the board to make it work as per the instructions fed into the board.

The best thing about Arduino boards is they can work as a stand-alone project or as a part of other electronic projects. You can interface Arduino Nano with other Arduino boards and Raspberry Pi boards. No technical expertise is required to use Arduino boards and anyone with little to no technical knowledge can make amazing projects with these units.

Following is the pin diagram of Arduino Nano board:

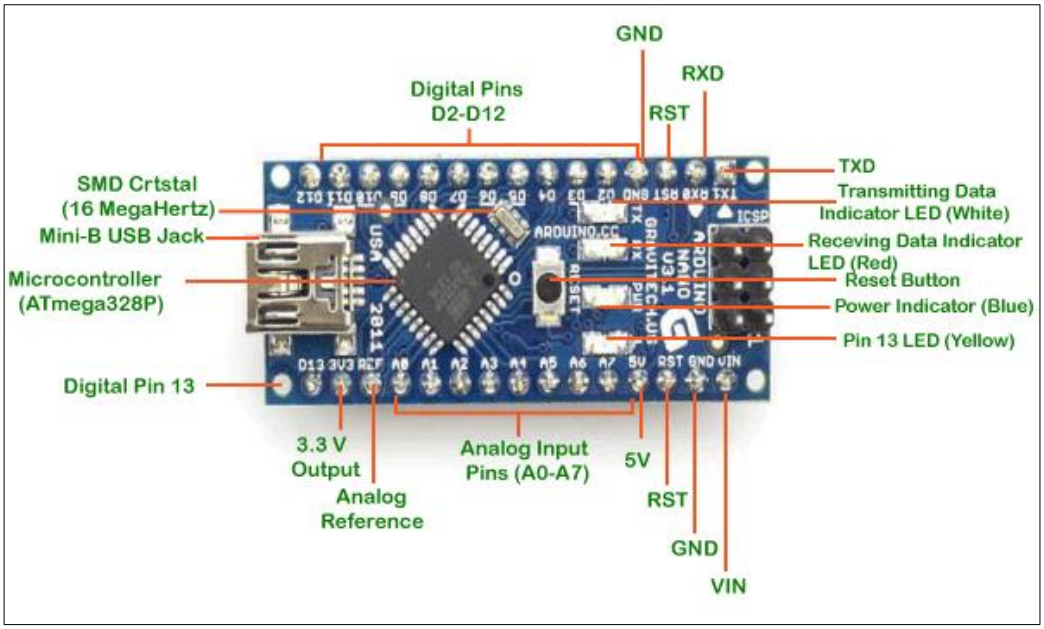

Figure 5.7: Pin Diagram of Arduino Nano

Pin description:

- **RXD and TXD:**

  TXD and RXD pins are used for serial communication. The TXD is used for transmitting the data, and RXD is used for receiving the data. It also represents the successful flow of data from computer to the board.

- **Digital Pins:**

  There are 14 digital I/O pins. The six pins from the set of digital pins are PWM (Pulse Width Modulation) pins numbered D3, D5, D6, D9, D10, and D11. The digital pins have the value either HIGH or LOW.

- **Analog Pins:**

  There are eight analog pins numbered from A0 to A7. The function of Analog pins is to read the value of analog sensor used in the connection. It can also act as GPIO (General Purpose Input Output) pins.

- **Analog Reference or (AREF):**

  The AREF pin acts as a reference voltage to feed the Arduino from an external power supply voltage.

- **Vin:**

  It is defined as the input voltage, which is applied to the Arduino Board when it is using an external power source.

- **3V3:**

  The 3V3 pin works as the output regulated voltage of 3.3V.

- **5V:**

  The 5V pin works as the output regulated voltage of 5V. The power source of 5V for the Arduino Nano board are USB connector, DC power jack, and the Vin. The power can be supplied to the board from either of the above specified sources.

Following are the main applications of Arduino Nano Board:

- Medical Instruments
- GSM Based Projects
- Embedded Systems
- Industrial Automation
- Android Applications
- Virtual Reality Applications
- Real-Time Face Detection
- Automation and Robotics

In the proposed system we have programmed the Arduino board such that when it receives a specific signal from bluetooth module, it triggers the buzzer.

- **Bluetooth Module:**

In communication, the Bluetooth technology has become very popular and it is one of the fastest growing fields in the wireless technology. Hence it is important to learn how the HC 05 Bluetooth module interfacing with the microcontroller. Nowadays, demands of mobile phones and personal communication the bandwidth are easy and convenient to use. The Bluetooth technology manages the communication channel of the wireless part. The Bluetooth modules can transmit and receives the data wirelessly by using two devices. The Bluetooth module can receive and transmits the data from a host system with the help of the host controller interface (HCI). The bluetooth module can be integrated and it gives the best possible solutions for the bluetooth embedded systems.

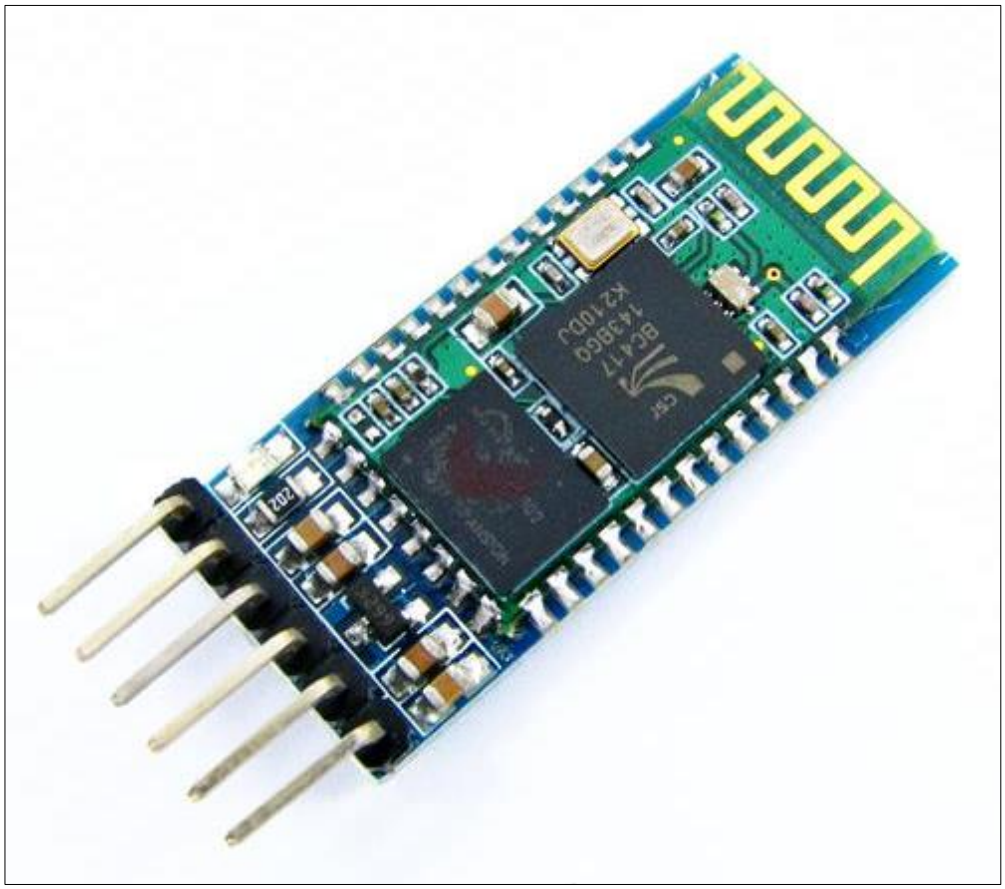

Figure 5.8: Bluetooth Module (HC-05)

The above figure shows the bluetooth module HC-05. HC-05 Bluetooth is a wireless communication protocol; it is used in two devices as a sending and receiving the information. The Bluetooth is free to use in the wireless communication protocol as the range of the bluetooth is less than the other wireless communication protocols like WiFi and Zigbee. The bluetooth operates at the frequency of the 2.41 GHz and also used in many small ranges of applications.

The HC-05 Bluetooth module is the most popular module in the Indian market and this module is mostly used in the embedded projects. The HC-05 Bluetooth modules are easy to use & simple, its price is low and these types of modules are interfaced with the Arduino, Rasberry Pi, and Microcontroller through the serial UART interface. These modules are designed for the transparent wireless connection setup and it is very easy to use in the Bluetooth serial port protocol.

Following diagram depicts the pin description of the Bluetooth HC 05 module:

Figure 5.9: Bluetooth Module HC-05 Pin Diagram

Pin description:

1. Key/EN: It is used to bring Bluetooth module in AT commands mode. If Key/EN pin is set to high, then this module will work in command mode. Otherwise by default it is in data mode. The default baud rate of HC-05 in command mode is 38400bps and 9600 in data mode. HC-05 module has two modes,

a. Data mode: Exchange of data between devices.

b. Command mode: It uses AT commands which are used to change setting of HC-05. To send these commands to module serial (USART) port is used.

2. VCC: Connect 5 V or 3.3 V to this Pin.

3. GND: Ground Pin of module.

4. TXD: Transmit Serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin).

5. RXD: Receive data serially (received data will be transmitted wirelessly by Bluetooth module).

6. State: It tells whether module is connected or not.

- **Buzzer Module:**

Buzzer is a small electromagnetic hardware device that produces sound. There are many types of buzzer available in the market. Buzzers are used in many small projects or jobs such as a washing machine, oven etc. An Arduino buzzer is also called a piezo buzzer. It is basically a tiny speaker that you can connect directly to an Arduino. You can make it sound a tone at a frequency you set. The buzzer produces sound based on reverse of the piezoelectric effect. Piezo buzzers are simple devices that can generate basic beeps and tones. They work by using a piezo crystal, a special material that changes shape when voltage is applied to it. If the crystal pushes against a diaphragm, like a tiny speaker cone, it can generate a pressure wave which the human ear picks up as sound.

Figure 5.10: Piezo Buzzer

As the name suggests, the buzzer creates an audible tone when you apply electrical energy to it. The most common buzzers you find are Piezo buzzers which are easy to use. The main element in a buzzer is a piezo (a Greek word meaning to squeeze), which oscillates and creates a tone when you apply DC power to it.

Piezo buzzers are very reliable and come in various form factors. We can see applications of piezo buzzers in following:

- Toys
- microwave ovens
- washing machines
- hardware projects
- other appliances

The piezo buzzer has two pins. Positive (Red) and negative (black) wire. we connect the negative wire to the GND pin and the positive wire to a PWM pin of the Arduino.

5.2 Flowchart

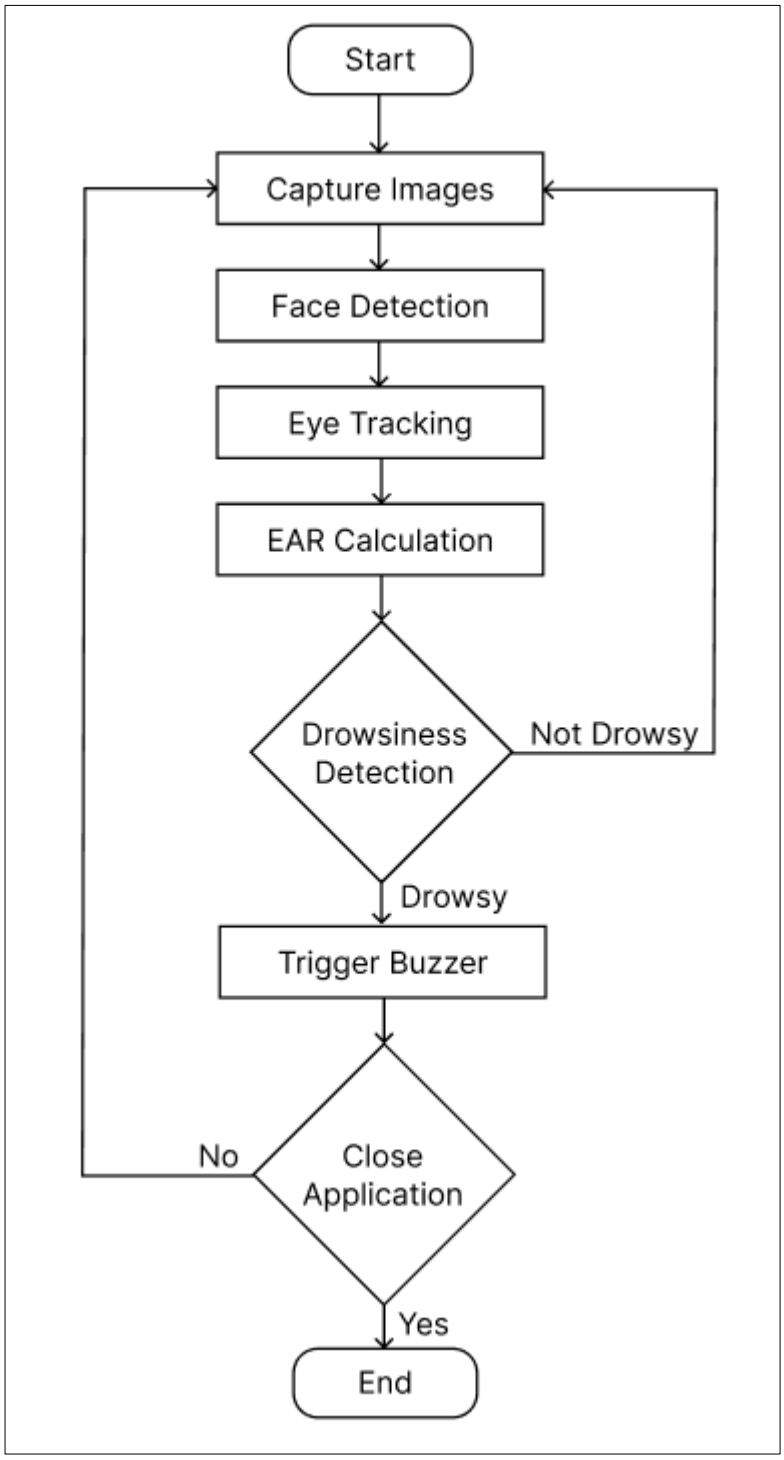

Figure 5.11: Flowchart of the System

5.3 Results & Screenshots:

- Home Screen:

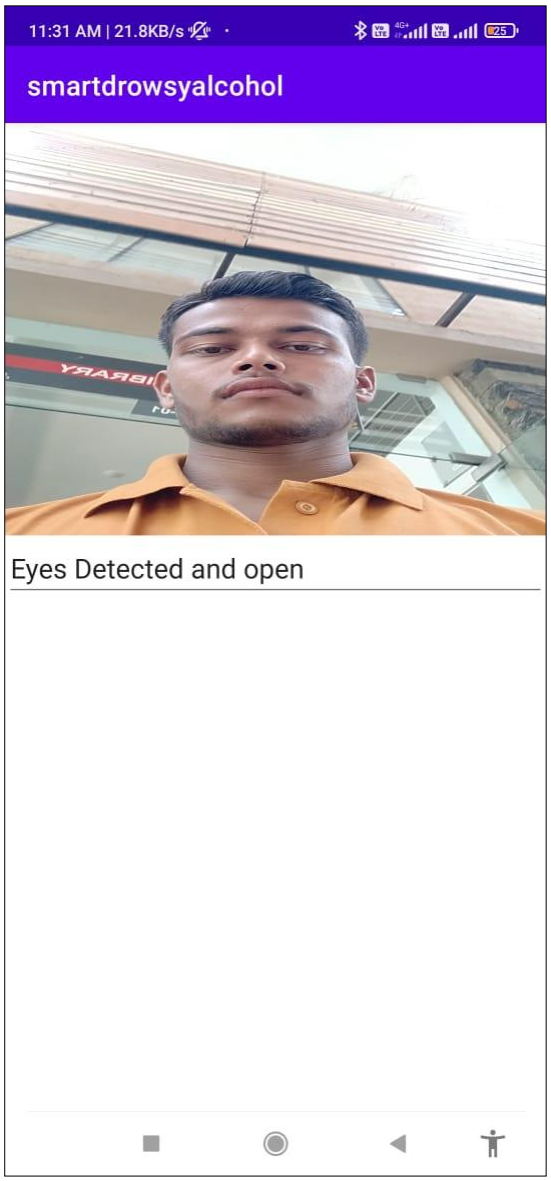

Figure 5.12: Home Screen

- This is the home screen of our application where user can see the camera input and below that there is a text which updates depending on whether face is detected or not and if person has closed his eyes or not.

- Updated Output Screen:

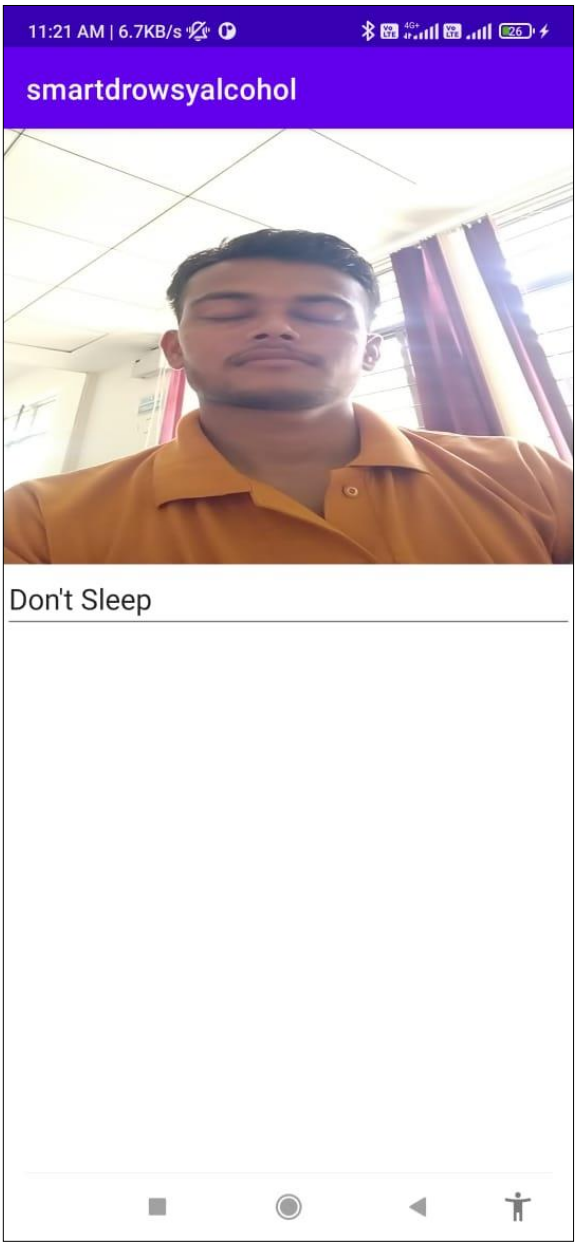

Figure 5.13: Drowsiness Detected Output

When user's face is detected the text below changes from "Face Not Detected" to "Eyes Detected and Open" if user has eyes open. When the eyes are closed the text changes to "Don't Sleep".

When the eyes are closed for more than 2 seconds the application sends a specific message to Arduino board through the bluetooth module. Then Arduino Nano after receiving this signal, triggers the buzzer to alert the driver. Below is the picture of the hardware implemented.

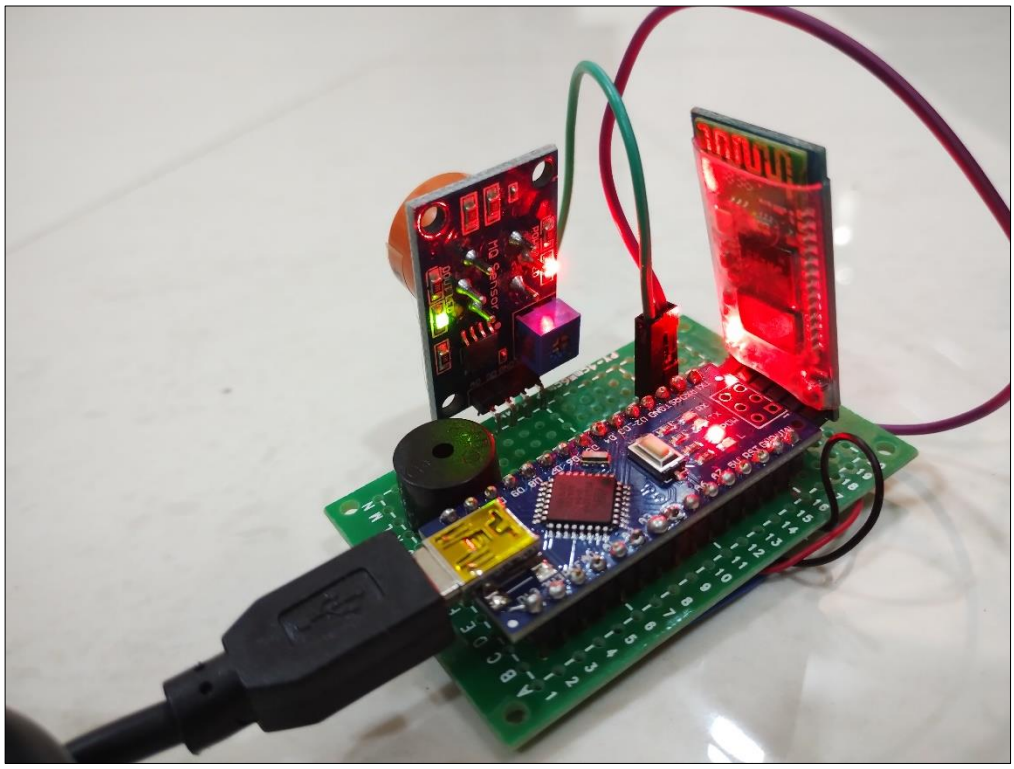

Figure 5.14: Proposed System Hardware

5.4 Result Analysis

Test Subject No.	Eye Detection Accuracy (%)	Drowsiness Detection Accuracy (%)
1	97.3	96
2	93	92.5
3	98	97.2
4	93.5	92

Table 5.1: Result Analysis

Formulae:

- **Eye Detection Accuracy** = $\frac{\textbf{Total No.of Times Eye Detected}}{\textbf{Total No.of Times Test Performed}}$ **x** 100
- **Drowsiness Detection Accuracy** = $\frac{\textbf{Total No.of Times Drowsiness Detected}}{\textbf{Total No.of Times Test Performed}}$ **x** 100

Result Analysis Graph:

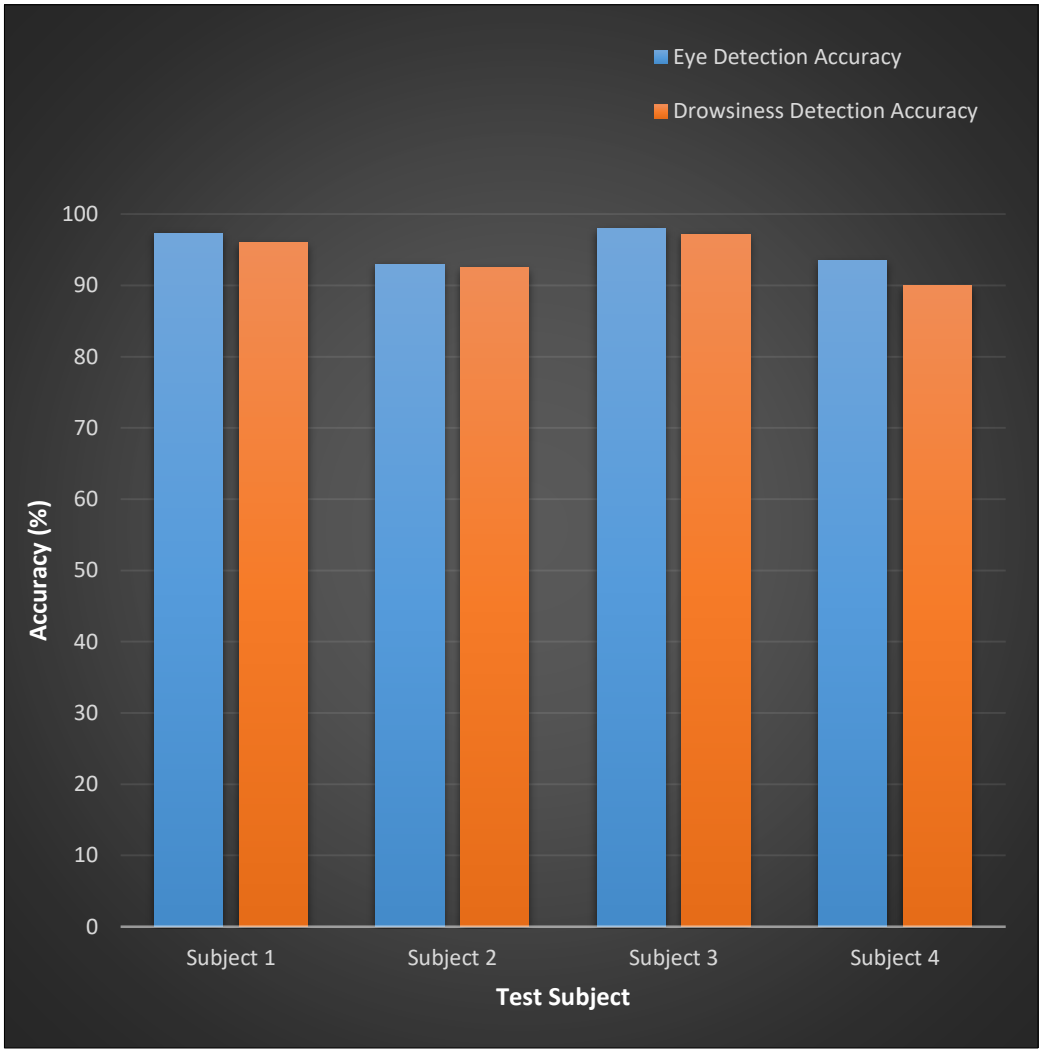

Figure 5.15: Graph of Result Analysis

We tested our system on 4 different people, 50 times for each person. The above graph depicts the results of those test cases. From the above graph we can observe that the average accuracy of the system was around 95%. The tests were conducted during different timelines so to check how the system works under different conditions. We have observed that the system performs better in daylight, where as the performance of the system decreases in darkness if there is not much of light on the user's face.

Chapter 6

Other Specifications

6.1 Advantages

- Easy to use.
- Efficient and reliable.
- Convenient approach to detect the drowsiness.
- Economically better with the technologies used.
- Decision Making / Problem Solving.
- Alarm automatically stops beeping after opening the eyes.

6.2 Limitations

Accuracy of the system decreases if there is low lighting on the face which makes it hard for algorithm to detect eyes.

6.3 Applications

Our system will help drivers by detecting their drowsiness and alert them so that they can focus on driving. It is applicable to all types of 4 wheeler vehicles.

Chapter 7
Conclusions and Future Work

Conslusions:

The drowsiness detection and correction system we are developing is capable of detecting drowsiness in a rapid manner. Our proposed system can differentiate normal eye blink and drowsiness, which can prevent the driver from entering the state of sleepiness while driving.

During the monitoring, it decides if the eyes are opened or closed. When the eyes have been closed for too long, a warning signal is issued. Processing judges the driver's alertness level on the basis of continuous eye closures.

This system can be used to reduce the amount of road accidents that happens to great extent. This can save a lot of lives, which is a main motive of this system.

This system does not need any complex system to work effectively. Taking the facts into consideration "Driver Drowsiness Detection System" is the future of road safety.

Future Work:

Future research may focus on measuring weariness using external parameters such as vehicle statuses, sleeping hours, weather conditions, mechanical data, and so on. Driver sleepiness is a serious hazard to highway safety, and it is especially problematic for commercial motor vehicle operators. This major safety concern is exacerbated by 24-hour operations, high yearly mileage, exposure to hazardous environmental conditions, and demanding work schedules. One key stage in a series of preventative measures required to solve this problem is to monitor the driver's level of sleepiness and attentiveness and provide feedback on their condition so that they may take appropriate action.

References

[1] A. Murata and Y. Hiramatsu, “Evaluation of drowsiness by HRV measures-Basic study for drowsy driver detection,” in Proc. 4th Int. Workshop Comput. Intell. Appl., 2008, pp. 99–102.

[2] J. Batista, “A drowsiness and point of attention monitoring system for driver vigilance,” in Proc. Intell. Transp. Syst. Conf., 2007, pp. 702–708.

[3] L. M. Bergasa, A. Member, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez, “Real-time system for monitoring driver vigilance,” IEEE Trans. Intell. Transp. Syst., vol. 7, no. 1, pp. 63–77, Mar. 2006.

[4] C. R. Jung and C. R. Kelber, “A lane departure warning system based on a linear–parabolic lane model,” in Proc. IEEE Intell. Veh. Symp., 2004, pp. 891–895.

[5] J. M. Clanton, D. M. Bevly, and A. S. Hodel, “A low-cost solution for an integrated multisensor lane departure warning system,” IEEE Trans. Intell. Transp. Syst., vol. 10, no. 1, pp. 47–59, Mar. 2009.

[6] R. Kawamura,M. S. Bhuiyan, H. Kawanaka, and K. Oguri, “Simultaneous stimuli of vibration and audio for in-vehicle driver activation,” in Proc. 14th Int. IEEE Conf. Intell. Transp. Syst., 2011, pp. 1710– 1715.

[7] N. Azmi, A. S. M. M. Rahman, S. Shirmohammadi, and A. El Saddik, “LBP-based driver fatigue monitoring system with the adoption of haptic warning scheme,” in Proc. IEEE Int. Conf. Virtual Environ., Human- Comp. Interfaces Meas. Syst., 2011, pp. 1–4.

[8] Hong Su and Gangtie Zheng, “A Partial Least Squares Regression-Based Fusion Model for Predicting the Trend in Drowsiness” IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 38, NO. 5, SEPTEMBER 2008.

[9] Fabian Friedrichs and Bin Yang, “Camera-based Drowsiness Reference for Driver State Classification under Real Driving Conditions” 2010 IEEE Intelligent Vehicles Symposium University of California, San Diego, CA, USA June 21-24, 2010.

[10] M.J. Flores J. Ma Armingol A. de la Escalera, “Driver drowsiness detection system under infrared illumination for an intelligent vehicle” Published in IET Intelligent Transport Systems Received on 13th October 2009 Revised on 1st April 2011.

[11] Zhang, Wei; Cheng, Bo; Lin, Yingzi,” Driver drowsiness recognition based on computer vision technology.” Published in: Tsinghua Science and Technology (Volume: 17, Issue: 3) Page(s):354 - 362 Date of Publication: June 2012

[12] Ralph Oyini Mbouna, Seong G. Kong, Senior Member, IEEE, and Myung-Geun Chun,” Visual Analysis of Eye State and Head Pose for Driver Alertness Monitoring.” IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 14, NO. 3, SEPTEMBER 2013.

[13] Eyosiyas Tadesse, Weihua Sheng, Meiqin Liu,” Driver Drowsiness Detection through HMM based Dynamic Modeling.” 2014 IEEE International Conference on Robotics & Automation (ICRA) Hong Kong Convention and Exhibition Center May 31 - June 7, 2014. Hong Kong, China.

[14] Tiberiu Vesselenyi, S. Moca, Alexandru Rus, Tudor Mitran "Driver drowsiness detection using ANN image processing", IOP Conf. Series: Materials Science and Engineering 252 (2017).

[15] Arun Sahayadhas, Kenneth Sundaraj & Murugappan Murugappan, “Detecting Driver Drowsiness Based on Sensors”, Sensors (Basel, Switzerland) vol. 12,12 16937-53. 7 Dec. 2012.

[16] Chisty, Jasmeen Gill, “Driver Drowsiness Detection System”, IJCST, Volume 3, Issue 4, Jul-Aug 2015.

[17] Sakshi Botwe, Vaishnavi Vispute, Prajakta Nagare and Dhruti Patil,"DRIVER DROWSINESS AND ACCIDENT DETECTION SYSTEM",IJARIIE-ISSN(O)-2395-4396,Vol-8 Issue-2 2022.

[18] H. Ueno, M. Kaneda and M. Tsukino, "Development of drowsiness detection system," Proceedings of VNIS'94 - 1994 Vehicle Navigation and Information Systems Conference, 1994, pp. 15-20.

[19] Marco Javier Flores, José María Armingol and Arturo de la Escalera, “A. Driver Drowsiness Warning System Using Visual Information for Both Diurnal and Nocturnal Illumination Conditions”. EURASIP J. Adv. Signal Process. 2010, 438205 (2010).

PUBLICATIONS

Sr. No.	Paper / Chapter Title	Journal / Conference / Book Details	Status (Published / Accepted / Submitted)
1	A Review On Driver's Drowsiness Detection Using Facial Feature	**Journal:** International Journal of Emerging Technologies and Innovative Research Date: 11th May 2022	Published