**SAFE AND SECURE DRIVING SYSTEM**

**CONTENTS**

[ABSTRACT i](#_Toc138442295)

[LIST OF FIGURES ii](#_Toc138442296)

[SYMBOLS & ABBREVIATIONS iii](#_Toc138442297)

[1. INTRODUCTION 1](#_Toc138442298)

[1.1 Introduction To Broad Area Of Research 2](#_Toc138442299)

[1.2 Introduction To Specific Area Of Research 2](#_Toc138442300)

[2. LITERATURE SURVEY 3](#_Toc138442301)

[2.1 Papers Reviewed 3](#_Toc138442302)

[2.1.1 Observations in the form of Table 3](#_Toc138442303)

[2.2 Motivation 7](#_Toc138442304)

[2.3 Problem Definition 8](#_Toc138442305)

[2.4 Objectives 9](#_Toc138442306)

[2.5 Limitations 9](#_Toc138442307)

[3. ANALYSIS 10](#_Toc138442308)

[3.1 Software Requirements Specifications 10](#_Toc138442309)

[3.1.1 User Requirements 10](#_Toc138442310)

[3.1.2 Software Requirements 10](#_Toc138442311)

[3.1.3 Hardware Requirements 10](#_Toc138442312)

[3.2 Algorithms 11](#_Toc138442313)

[4. DESIGN 12](#_Toc138442314)

[5. IMPLEMENTATION OF KEY FUNCTIONS 20](#_Toc138442315)

[5.1 Drowsiness Detection System 20](#_Toc138442316)

[5.1.1 Detailed Description 20](#_Toc138442317)

[5.1.2 Result Analysis 20](#_Toc138442318)

[5.1.3 Method Of Implementation 21](#_Toc138442319)

[5.2 Palm Vein Recognition 22](#_Toc138442320)

[5.2.1 Detailed Description 22](#_Toc138442321)

[5.2.2 Result Analysis 23](#_Toc138442322)

[5.2.3 Method Of Implementation 23](#_Toc138442323)

[5.3 Iris Recognition 24](#_Toc138442324)

[5.3.1 Detailed Description 24](#_Toc138442325)

[5.3.2 Result Analysis 24](#_Toc138442326)

[5.3.3 Method Of Implementation 25](#_Toc138442327)

[5.4 Pulse Rate Detection And Proximity Sensing 26](#_Toc138442328)

[5.4.1 Detailed Description 26](#_Toc138442329)

[5.4.2 Result Analysis 27](#_Toc138442330)

[5.4.3 Method Of Implementation 28](#_Toc138442331)

[6. TESTING AND VALIDATION 30](#_Toc138442332)

[6.1 DROWSINESS DETECTION 30](#_Toc138442333)

[6.1.1 Design of test cases and scenarios 30](#_Toc138442334)

[6.1.2 Validation 31](#_Toc138442335)

[6.2 Palm Vein Recognition 32](#_Toc138442336)

[6.2.1 Design of test cases and scenarios 32](#_Toc138442337)

[6.2.2 Validation 33](#_Toc138442338)

[6.3 Iris Recognition 34](#_Toc138442339)

[6.3.1 Design of test cases and scenarios 34](#_Toc138442340)

[6.3.2 Validation 35](#_Toc138442341)

[6.4 Pulse Rate Detection And Proximity Sensing 36](#_Toc138442342)

[6.4.1 Design of test cases and scenarios 36](#_Toc138442343)

[6.4.2 Validation 37](#_Toc138442344)

[7. CONCLUSION 38](#_Toc138442345)

[7.1 Future Scope 38](#_Toc138442346)

[8. APPENDICES 39](#_Toc138442347)

[8.1 Plagiarism Report 39](#_Toc138442348)

**ABSTRACT**

The Driver Safety System detects diverse aspects of driver behaviour to identify potentially dangerous actions that can possibly cause harm to the vehicle and its driver. It manages safety in real-time by actively helping the driver to at least reduce the impact of an emergency situation. Palm vein authentication is one of the most reliable forms of biometric authentication, using the principle, the proposed system integrates security of a high degree.

It is a low-cost simple system that aids in the detection of elements that are the most contributors in causing road accidents, with respect to the driver. Using sensors of various kinds, an Arduino, and EAR and MAR, our system assists in the detection of drivers’ behaviours while on the road. Using sensors to detect eyelid movement and head alignment, proximity sensors to detect other vehicles and objects in order to prevent collision, buzzers and ERM to alert the driver in the event of drowsiness, our system can reduce the rate of road accidents. Moreover, with a biometric authentication system, which grants access by scanning the vein pattern and matching it against registered patterns.

**Keywords:** *Arduino, Head alignment, EAR, MAR, Pulse Rate, Biometric Authentication, Vein detection.*

**LIST OF FIGURES**

[Fig.1. Architecture diagram for safe driver system 12](#_Toc138441415)

[Fig.2. Flow diagram for safe driver system 13](#_Toc138441416)

[Fig.3 (i) Flow diagram of drowsiness detection 14](#_Toc138441417)

[Fig.3 (ii) Working model for drowsiness detection 15](#_Toc138441418)

[Fig.4. Flow diagram for palm recognition 16](#_Toc138441419)

[Fig.5. Flow diagram for Pulse Rate Detection 17](#_Toc138441420)

[Fig.6. Flow diagram for Proximity Sensing 18](#_Toc138441421)

[Fig.7. Flow diagram for Iris Recognition 19](#_Toc138441422)

**SYMBOLS & ABBREVIATIONS**

* EAR - Eye Aspect Ratio
* MAR - Mouth Aspect Ratio
* ERM - Eccentric Rotating Mass
* GSM - Global System for Mobile communication
* IDE - Integrated Development Environment
* LCD – Liquid Crystal Display
* IoT – Internet of Things
* LBPH – Local Binary Pattern Histogram
* VCC – Voltage at the Common Collector
* GND – Ground
* TRIG - Trigger

# INTRODUCTION

A safe and secure driver system is a set of practices, technologies, and regulations aimed at ensuring the safety and security of drivers, passengers, and other road users. The system involves various components, including vehicles, drivers, infrastructure, that work together to minimize the risk of accidents and other incidents on the road.

Biometrics refers to the measurement of biological characteristics unique to each individual for the purposes of identity verification. These characteristics include the veins in the palm or fingers of the hand or any other parts of the body whose specific patterns are unique to an individual. Biometrics has been employed in a variety of security applications like door access control to office buildings or research laboratories, PC or mobile phone login security etc. Palm vein authentication works by comparing the pattern of veins in one’s palm to a vein pattern stored in a database.

Uncovering vein pattern has been gaining more and more focus in the field of research to utilise its unique factor in various fields of application, not limited to forensics and criminology, thus increasing the number of real-world applications of the same.

Iris authentication uses factors like the edge of the iris and the pupil, the centre of the pupil, and the edge of the iris to map out the unique pattern which is then compared with an existing pattern stored in the database to authenticate the user.

Road accidents are one of the leading causes of death all around the world, especially in countries with an advanced highway / freeway system. Sometimes drivers have to stay awake and drive long hours with less sleep. Be it due to driver negligence or vehicle hardware failure, a huge factor in deaths is road accidents. According to statistics, 1.3Million people die each year due to road related accidents.

Keeping in mind the ever-changing standards in automobile industries, we can incorporate certain features within the mechanism of vehicles which might aid in the prevention of road accidents. The aim of this project is to develop a safety system in vehicles for drivers to aid them in driving safely and securely, and to ensure that the roads are made more safe than before.

## 1.1 Introduction To Broad Area Of Research

Vehicles can be converted into Smart Vehicle Monitoring Systems (SVMS) by utilising the Internet of Things (IoT) to monitor driver health and prevent accidents. In vehicles, SVMS employs IoT devices that connect, communicate, and flow data using both IoT and cloud computing.​

The Internet of Things technology can then be used to monitor and maintain a safe work environment, particularly through effective record keeping. In this case, records of compliance training are reviewed, and management ensures that all employees have received adequate training, particularly in safety etiquette.​

The Internet of Things includes many new technologies, the most important of which are radio frequency identification technology (RFID), sensor technology, network communication technology, and cloud computing.​

## 1.2 Introduction To Specific Area Of Research

Sensors in autonomous vehicles (AVs) aid in the safety of drivers, passengers, and pedestrians. ​

The Driver Safety System detects diverse aspects of driver behaviour to identify potentially dangerous actions that can possibly cause harm to the vehicle and its driver. It manages safety in real-time by actively helping the driver to at least reduce the impact of emergency situation.​

It is a low cost simple system that aids in the detection of elements that are the most contributors in causing road accidents, with respect to the driver. Using sensors of various kinds and an arduino, the system assists in the detection of drivers’ behaviour while on the road.​

Using sensors and cameras to authenticate the user through biometrics such as palm vein and iris, then detect eyelid movement and head alignment, proximity sensors to detect other vehicles and objects in order to prevent collision, buzzers and vibration sensors to alert the driver in events of drowsiness, etc., our system can reduce the rate of road accidents.​

# Literature Survey

## 2.1 Papers Reviewed

### 2.1.1 Observations in the form of Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No** | **Author and year of publication** | **Title of the paper** | **Observations** |
| 1. | Chaoying Tang, Adams Wai Kin Kong, Noah Craft  June 2011 | Uncovering Vein Patterns from Color Skin Images for Forensic Analysis | - These results indicate that the uncovering vein patterns can be used for personal identification.  -Vein patterns can be uncovered from color images (evidence images) and matched to NIR images.  -Future work includes using local context of vein pixels to further improve the algorithm. |
| 2. | Hengyi Zhang, Chaoying Tang, Adams Wai-Kin Kong and Noah Craft  2012 | Matching Vein Patterns from Color Images for Forensic Investigation | Developed an automatic matching algorithm for vein identification that can handle rigid and non-rigid deformations and has an explicit pruning function to remove outliers in vein patterns.  The algorithm, based on the OBVU method, includes parameter range optimization and image intensity adjustment. |
| 3. | Chaoying Tang, Hengyi Zhang and Noah Craft  2012 | Visualising Vein Patterns from Color Skin Images based on Image Mapping for Forensics Analysis | 1. Vein patterns can be visualised using information extracted from corresponding colour values and NIR intensities.  2. The proposed NN weight adjustment is effective,  3. The final identification accuracy of the proposed algorithm is higher than the optical method, and  4. The result of the proposed algorithm from the same-type-image matching is comparable to the result from the NIR images, which are usually considered as ground truth of vein patterns |
| 4. | Rohan Don Salins and Ananth Prabhu G  November 2018 | Review on Exploring an Analytical Study on Optimal  Methods used for Identifying Sexual Offenders or  Criminal Identification | Scope of the research is not only limited to vein patterns but also lesions and other marks on the skin. Using common skin conditions which show consistency through time. It also considers androgenic hair pattern detection as a feature unique to a person. |
| 5. | Fujitsu  2005 | Palm Vein Pattern Authentication Technology | More secure than other biometric screening methods.  They use IC cards so that the vein pattern information is not stored in the park database for additional security. |
| 6. | Swati Verma and Pomona Mishra  January 2013 | A survey paper on palm prints based biometric authentication system | -Palms print scanners and preprocessing  -Verification algorithm  - Subspace-Based Approaches  -Statistical Approaches  -Coding Approaches |
| 7. | Iftikhar Ali Khan,  Syed Zubair Ahmed,  Mudassar Iqbal  March 2022 | Driver Safety System for Drowsiness, Heart Attack, Object Detection, and Internal Temperature Control of Car With Real-Time Wireless Communication | The system is affordable and can be implemented in a variety of models. |
| 8. | K. Ramesh,  Kalimuthu Krishnan,  Srinath Balasubramanian  January 2021 | Driver Assistance and Safety System for Accident Prevention Using Embedded Automotive Sensors Integration | The usage of specified sensors for every function and integration of these devices is done in a simple manner.  The emergency responses are very well thought out. |
| 9. | I. J. Mrema,  Mussa Dida  August 2020 | A Survey of Road Accident Reporting and Driver’s Behavior Awareness Systems: The Case of Tanzania | i) The mobile  application  for  reporting  road accidents  could deal with  the  current challenges  faced by the traffic police authorities in receiving accident reports, reducing the time needed to provide help to the victims.  ii)Portable  iii)Inexpensive |
| 10. | Shrinath Oza,  Dr. Sunil Rathod  June 2020 | Object Detection using IoT and Machine Learning to Avoid Accident and Improve Road Safety | In this paper, Accident Avoidance and Improving Road Safety  with  Use  of  Raspberry  Pi  for  Object  Detection system are proposed.  The camera is  used to  capture continuous  real-world images. According to the  images available  through the  camera,  these  images  can  be  transmitted  to  the raspberry pi to perform the control actions of the car. The framework  performs  preprocessing  utilizing  the  Mean Subtracted Difference Enhancement  (MSDE) strategy and afterward segmentation is performed |
| 11. | Khan,  Lee  December 2019 | Gaze and Eye Tracking: Techniques and Applications in ADAS | Present negligible intrusiveness and minimal usage difficulty.  Provide the user with enhanced freedom of head movement. |

## 2.2 Motivation

* Safety: The primary motivation for a safe and secure driver system is to enhance safety on the roads. Car accidents are a major cause of injury and death worldwide, and a driver system that can detect potential hazards and help prevent accidents can significantly reduce the number of fatalities and injuries.
* Convenience: A safe and secure driver system can make driving more convenient by reducing the need for drivers to constantly monitor the road and make split-second decisions. This can be particularly beneficial for long drives or in heavy traffic.
* Innovation: The development of a safe and secure driver system requires the use of cutting-edge technologies, which can drive innovation and advance the state of the art in fields such as artificial intelligence, computer vision, and robotics.
* Enhancement of customer satisfaction: Customers who feel safe and secure while using a vehicle are more likely to be satisfied with their experience. By offering safe and secure driver systems, vehicle manufacturers can enhance customer satisfaction and loyalty.
* Advancement of technology: Developing safe and secure driver systems requires advanced technology and innovation. By investing in these systems, vehicle manufacturers can help drive technological advancements and improve the overall quality of their products.

## 2.3 Problem Definition

Vein pattern detection is a biometric technology that uses infrared light to scan the vein patterns in a person's hand for secure authentication. It can be integrated into automobiles to create a more secure system for preventing theft. This technology could reduce the cases of grand theft auto and improve the security of automobiles.

Palm vein detection is an innovative solution to most of our security concerns. It is one of the most reliable biometric authentication techniques as it is highly difficult to replicate the vein pattern of a person. The vein pattern of each human being is unique to them, just like their fingerprint. But vein patterns are more reliable than fingerprints as it is extremely difficult to falsify.

In cases of surveillance evidence, vein detection can be applied to a body part caught on camera, for example, arm, leg etc., and match it with a potential suspect. Not only surveillance, vein pattern recognition can be beneficial in cyber crimes and illegal sexual exploitation of children, which is a major global issue.

Road mishaps are one of the major contributors of the death toll. Vehicles don’t have inbuilt mechanisms to detect and avoid certain problems, or to effectively provide security. Drivers are negligent on the road, and sometimes the journey is hectic and doesn't have any rest stops in between. These are the main factors in causing road accidents. By having a system that gives the driver periodical alerts, accidents caused by drowsiness could be reduced.

Automobile theft is a major concern, particularly for luxury cars and other high-value vehicles. Despite the availability of various anti-theft mechanisms, such as alarms and password protection, these crimes still occur at an alarming rate. Palm vein authentication could significantly reduce the cases of grand theft auto, as it would be much more difficult for thieves to bypass the authentication system. It could also reduce the cost and inconvenience of traditional anti-theft mechanisms, which can be easily bypassed by experienced thieves.

## 2.4 Objectives

* To reduce the tally of accidents on the road.
* To ensure the safety of all parties on the road.
* To reduce the risk of grand theft auto.
* To reduce the casualties which occur due to driver negligence.
* Reduction in accidents leads to less automobiles being destroyed, and the authentication process makes it less likely to be stolen, thus being sustainable.

## 2.5 Limitations

* False Alarms: Driver safety systems rely on sensors, cameras, and other technology to detect potential hazards. These sensors may sometimes interpret harmless situations as hazardous and trigger false alarms, which can be distracting and annoying for drivers.
* Reliance on Environmental Conditions: Driver safety systems may not function properly in certain environmental conditions, such as heavy rain or snow, fog, or bright sunlight. These conditions can affect the system's sensors and limit their ability to detect hazards.
* User Error: Driver safety systems are only effective if they are used properly. If drivers do not understand how to use the system or ignore alerts and warnings, they may not be able to benefit fully from the system's capabilities.

# Analysis

## 3.1 Software Requirements Specifications

### 3.1.1 User Requirements

* Reliability: Users want a system that works reliably and consistently. The system should be able to identify potential hazards and respond appropriately in a timely manner.
* Flexibility: Users may have different preferences and needs, so the system should be flexible enough to accommodate these differences. For example, it may offer different settings or levels of assistance.
* Privacy: Users want to know that their personal data is being protected and that the system is not tracking or monitoring them inappropriately.
* Security: Users want to know that the system is secure and cannot be hacked or compromised by unauthorised individuals.

### 3.1.2 Software Requirements

* Linux
* Python- Machine Learning
* IDE- Arduino IDE

### 3.1.3 Hardware Requirements

1. Arduino
2. Sensors
3. Camera
4. Pulse Sensor
5. Alarm
6. Ultrasonic Sensors
7. Eccentric Rotating Mass (ERM)
8. GSM
9. LCD

## 3.2 Algorithms

* Watershed Segmentation:

The Watershed Segmentation Algorithm is a computer vision technique used for image region segmentation.

* Euclidean Distance:

The Euclidean Distance refers to the distance between two points in the plane or 3-dimensional space. In short, we can say that it is the shortest distance between 2 points irrespective of dimensions.

* LBPH (Local Binary Patterns Histograms) Face Recognition:

LBPH is a widely used face recognition algorithm that extracts local binary pattern features from grayscale images and builds histograms to represent faces.

# DESIGN

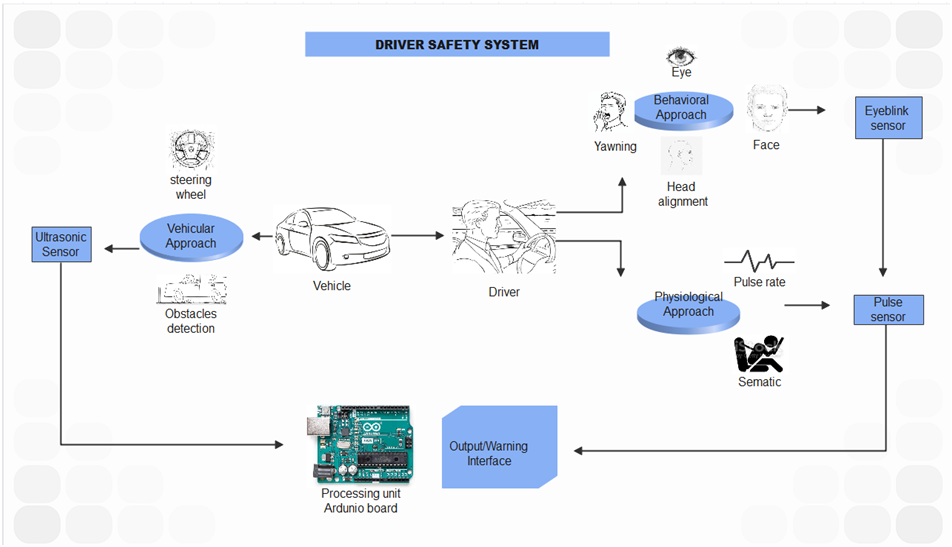


Fig.1. Architecture diagram for safe driver system

The driver safety system employs a behavioral, vehicular, and physiological approach.

* Behavioral Approach: A Webcam will be used to detect the eyes, head, face, and yawing in the behavioral approach. The Arduino board receives the data and sends alert signals to the driver, such as buzzer sounds and vibrating seats.
* Physiological Approach: Here, pulse rate are detected using a pulse sensor; if the pulse rate drops below a certain threshold, a message is sent to a registered number.
* Vehicular Approach: In a vehicular approach, an ultrasonic sensor detects steering wheel pressure and obstacles and sends an alert signal to the driver.

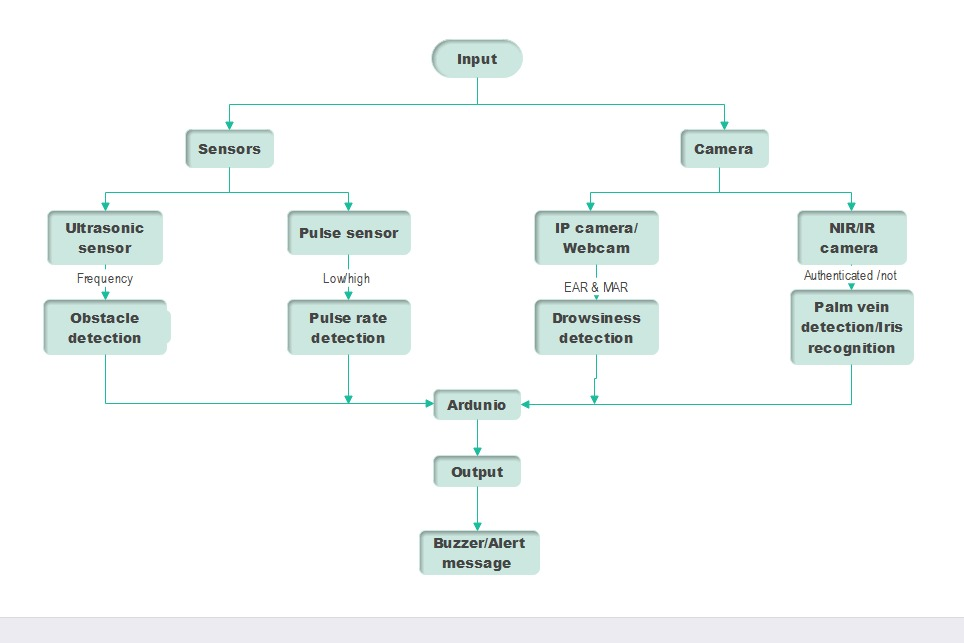


Fig.2. Flow diagram for safe driver system

The extracted frames from the webcam are analyzed for facial cues to detect drowsiness, line of sight of the driver and head alignment. In case it is detected that the driver is inattentive, an alarm is sounded or the seats are vibrated.

The pulse monitor is set to a certain threshold. If the pulse level is high or low, a message is sent to a registered number.

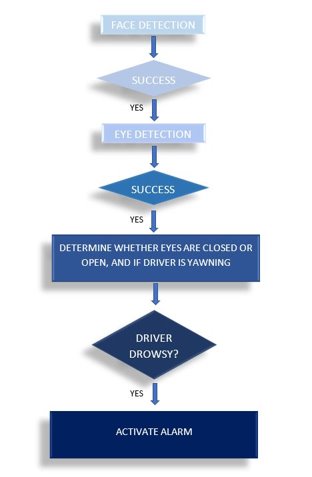


Fig.3 (i) Flow diagram of drowsiness detection

Here, in drowsiness detection of a driver we first detect the face of the driver and the eyes. After detection it checks whether the eyes of the driver are closed or open and whether the driver is yawing. If it detects that the driver is drowsy, an alarm will be activated with an alert message.

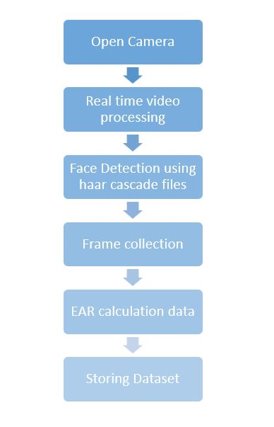
~~~~

Fig.3 (ii) Working model for drowsiness detection

Drowsiness detection using webcam:

Webcam is used to detect driver’s drowsiness where it uses haar cascade function to detect the face of the driver with real time video processing. Once it detects a face it uses a shape predictor of 68 landmarks to detect whether a driver is yawning or sleepy by EAR and MAR calculation. If it detects signs of drowsiness, an alarm is activated with the alert message. Here the frames of driver are collected and stored in the dataset.

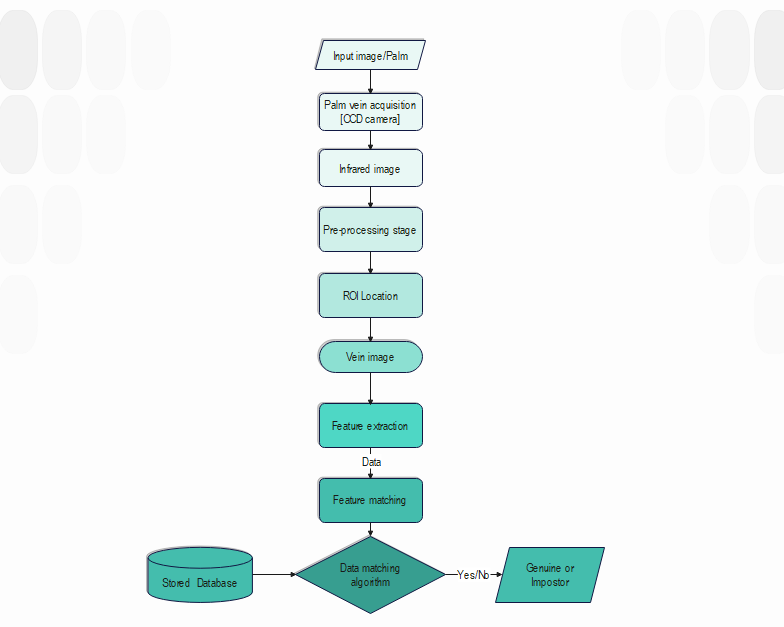


Fig.4. Flow diagram for palm recognition

There are many ways to retrieve palm images. In the design we are using camera with infrared rays, which undergoes pre-processing, where it finds vein image. The feature extraction will be matched with the data stored in the database using algorithm. Once after the matching is successful it will be able to find whether the driver is genuine or impostor.

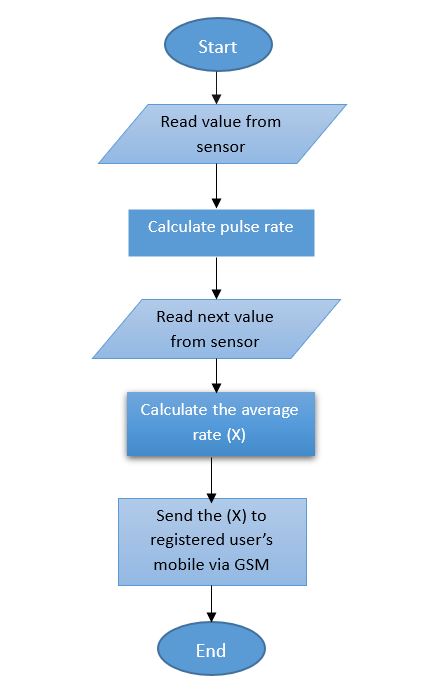


Fig.5. Flow diagram for Pulse Rate Detection

When the driver places his/her finger on the pulse sensor, the sensor detects the pulse rate and sends this data to the microcontroller. The microcontroller then processes the data and sends it to the display unit, where the driver can see their pulse rate.

In vehicles, pulse sensors are often used to monitor the driver's health and alert the driver in case of any emergency. For example, if the driver's pulse rate exceeds a certain limit, the system sends a message to a registered number via 4g-sim GSM.

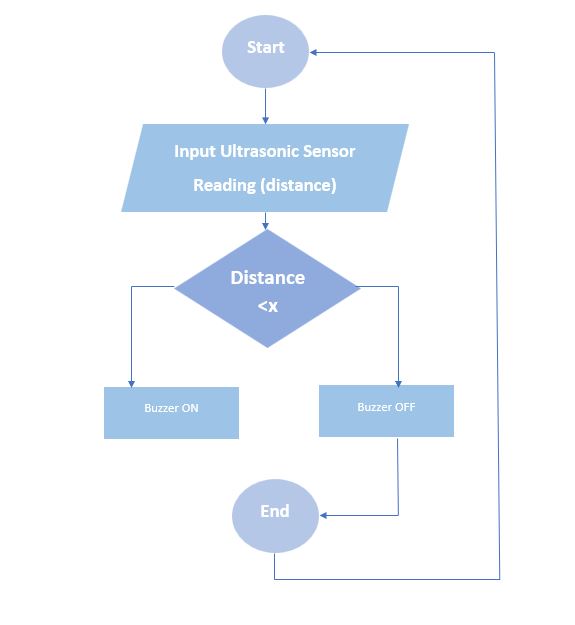


Fig.6. Flow diagram for Proximity Sensing

Emitting waves: Ultrasonic sensors generate high-frequency sound waves.

Receiving echoes: The sensor detects the echoes when the waves bounce off objects.

Measuring distance: By calculating the time it takes for the waves to return, the sensor determines the distance to the object.

Activating the Buzzer : The Buzzer turns on if the object appears to be closer to the vehicle.

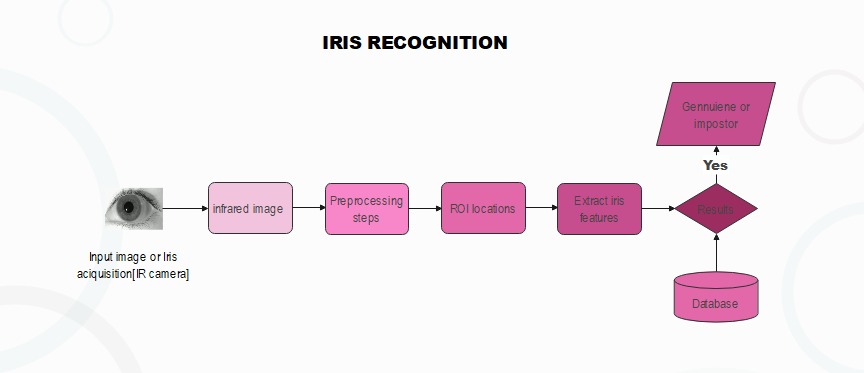


Fig.7. Flow diagram for Iris Recognition

The iris recognition system trains a model using iris images stored in a database directory by performing preprocessing, iris segmentation, normalization, and feature extraction. The encoded iris features are saved in a file. The code then tests the model by preprocessing a test image, extracting its iris features, and comparing them with the stored encodings. If the distance between the features is below a threshold, it returns the matched person's name and a confidence percentage based on the distance. Otherwise, it returns "Unmatched" and a confidence of 0.0.

# IMPLEMENTATION OF KEY FUNCTIONS

## 5.1 Drowsiness Detection System

### 5.1.1 Detailed Description

Driver drowsiness detection using webcam real-time video processing is a computer vision-based approach that utilizes a camera to monitor the driver's facial features and analyze them in real-time to detect signs of drowsiness.

The system typically follows the following steps:

1. Capture live video feed from the webcam using OpenCV.
2. Convert each frame of the video feed to grayscale.
3. Use Haar Cascades to detect eyes and mouth in the grayscale frame.
4. Calculate the aspect ratio of the eyes and mouth to check for drowsiness. The aspect ratio of the eyes is calculated as the ratio of the distance between the vertical eye landmarks to the distance between the horizontal eye landmarks. If the aspect ratio falls below a certain threshold, the driver is considered drowsy. The aspect ratio of the mouth is calculated as the ratio of the distance between the upper lip and lower lip landmarks to the distance between the left corner and right corner of the mouth landmarks. If the aspect ratio falls below a certain threshold, the driver is considered yawning.
5. Alert the driver using a sound or vibration if drowsiness or yawning is detected.

### 5.1.2 Result Analysis

* After taking video input it starts calculating EAR and MAR of the face.
* Once the values are calculated, it stores the frames as datasets in the designated folder.
* If the EAR and MAR coordinates changes drastically, it concludes that the driver is drowsy and alerts the driver by sounding an alarm and activating the ERM.

### 5.1.3 Method Of Implementation

The platform currently used in the implementation is Anaconda.

       Step 1: Update conda

       Step 2: Update anaconda

       Step 3: Create a virtual environment

We create an environment to integrate all the components.

       Step 4: Activate the virtual environment

       Step 5: Install dlib

       Step 6: Code of drowsiness detection

       Step 7: Download shape predictor as per the requirement

       Step 8: Creating application to open webcam

       Step 9: Begin frame by frame implementation.

## 5.2 Palm Vein Recognition

### 5.2.1 Detailed Description

1. Dataset Loading:

* The code begins by specifying the directory path where the palmprint dataset is stored.
* It initializes empty lists for labels and images.

1. Iterating through the Dataset:

* The code iterates over each label in the dataset directory.
* Within the label directory, it iterates over each image file.
* For each image file, it checks if it is a valid image file (with extensions .png, .jpg, .jpeg).
* If the file is not a valid image, it skips to the next iteration, or else it reads the image using OpenCV's imread function in grayscale format.
* The image is appended to the images list, and the corresponding label is appended to the labels list.

1. Converts the  Lists to NumPy Arrays
2. Recognizer Creation and Label Mapping:

* The code creates an instance of the LBPH (Local Binary Patterns Histograms) face recognizer using OpenCV's LBPHFaceRecognizer\_create function.
* It defines a label dictionary that maps non-numeric labels (person names) to integer values.

1. Converts Labels to Integer Data Type using a list comprehension and the label dictionary.
2. The code trains the Recognizer.
3. Testing the Recognizer:

* The code specifies the path of a test image in the test\_image\_path variable.
* If the test image is read successfully, the code uses the recognizer's predict function to predict the label and confidence of the test image.
* Finally, the predicted person's name and confidence level are printed as output, or an error message, if there’s an error.

### 5.2.2 Result Analysis

* The code successfully loads the palmprint dataset and processes the images for training and testing.
* The LBPH face recognizer is trained using the dataset and is able to predict the person's name based on a test image.
* The predicted person and confidence score are printed, indicating the recognized individual and the confidence level of the prediction.
* The code provides a basic implementation for palmprint recognition using LBPH, but the accuracy and performance may vary depending on the dataset and specific use case.
* It would be beneficial to evaluate the performance of the recognizer using different test images and potentially explore other recognition algorithms or techniques to improve accuracy.

### 5.2.3 Method Of Implementation

Step 1: Make sure you have Python installed on your laptop

Step 2: Install OpenCV [OpenCV is a required library for computer vision tasks in the code]

Step 3: Create a directory on your laptop and organize your palmprint dataset according to the code's expected structure.

Step 4: Save a code of Plamprint detection in the same directory as dataset with the extension .py.

Step 5: Run file.

## 5.3 Iris Recognition

### 5.3.1 Detailed Description

1. Preprocessing:

* The image is loaded and converted to grayscale.
* Histogram equalization is performed to enhance contrast.
* Gaussian blur is applied to reduce noise.

1. Iris Segmentation and Normalization:

* Watershed algorithm is applied to separate the iris region.
* Normalization is performed by multiplying the image with the iris region.

1. Gabor filtering is applied to extract features from the normalized image.
2. Training:

* The code iterates through the labeled directories in the database.
* For each image file in a label directory, preprocessing and feature extraction are performed.

1. The iris encodings dictionary is saved to a pickle file.
2. Testing:

* The encoding model is loaded from the pickle file.
* The test image is preprocessed and its iris features are extracted.
* The Euclidean distance is calculated between the test features and each encoding in the model.
* The best match with the minimum distance is determined as the result.

1. Confidence Calculation:

* If the minimum distance is below a threshold, the result is considered a "Matched" iris.
* If the distance is above the threshold, the result is "Unmatched" with 0.0% confidence.

1. Printing the Result:

* The code prints the result as "Matched" or "Unmatched".
* If the result is a match, the person's name and the confidence value  is printed.

### 5.3.2 Result Analysis

* The code implements an iris recognition system using image processing techniques.
* It trains the model by extracting iris features from a database of labeled iris images and saving the encodings.
* The trained model is then used to test a new iris image by comparing its features with the stored encodings.
* The best match is determined based on the Euclidean distance between the test features and the stored encodings.
* If the distance is below a specified threshold, the result is considered a match.
* The code calculates a confidence value as a percentage based on the distance, representing the similarity between the test image and the best match.
* The result, person's name, and confidence value are printed as the output.
* The code allows for iris recognition and identification with a measure of confidence in the match.

### 5.3.3 Method Of Implementation

Step 1: Make sure you have Python installed on your laptop.

Step 2: install opencv, numpy, pickle, os, scipy, distance, gabor\_kernel, watershed and few more packages as per the requirements.

Step 3: Create a directory on your laptop and organize your palmprint dataset according to the code's expected structure.

Step 4: Create another directory called “encodingModel “in the same directory dataset is stored.

Step 5: Create a file named ” IrisEncoding.pickle “ inside encodingModel.

Step 6: Create a file inside the main directory and save the Iris code inside it with the .py extension.

Step 7: Set the paths properly in the code to avoid errors in the execution of the code.

Step 8: Run the code.

## 5.4 Pulse Rate Detection And Proximity Sensing

### 5.4.1 Detailed Description

5.4.1.1. Proximity Sensing Algorithm:

* The code uses an ultrasonic sensor to measure the distance to an obstacle.
* To calculate the distance, it measures the duration of the pulse received by the echo pin using the ‘pulseIn()’ function.
* It then uses the speed of sound (0.034 cm/microsecond) to calculate the distance based on the duration.
* Depending on the calculated distance, different actions are taken, like, different beeps are produced from the buzzer.

5.4.1.2. Pulse Rate Monitoring Algorithm:

* The code uses the ‘PulseSensorPlayground’ library to monitor the user's pulse rate.
* It checks for the start of a heartbeat using the ‘pulseSensor.sawStartOfBeat()’ function.
* If a heartbeat is detected, the code receives the pulse rate (BPM) using the ‘pulseSensor.getBeatsPerMinute()' function.
* It then checks if the obtained pulse rate is within a reasonable range (between 60 and100 BPM).
* Depending on the result, if the BPM exceeds the threshold continuously for a certain amount of time, the GSM module sends a message to a registered number, informing them of the situation.
* Additionally, it displays a cautionary message on the LCD for the driver to see.

Overall, the code combines basic sensor input reading (ultrasonic sensor and pulse sensor) with simple control flow and calculations. While it does not employ complex algorithms, it utilises underlying principles to achieve proximity sensing and pulse rate monitoring functionalities.

### 5.4.2 Result Analysis

5.4.2.1 Proximity Sensing:

* An ultrasonic sensor is used to measure the distance to an obstacle.
* It calculates the distance based on the duration of the pulse received by the echo pin.
* The distance is calculated using the formula: ‘distance = duration \* 0.034 / 2’
* If the distance is less than 15 cm, the buzzer plays a short beep.
* If the distance is between 15 cm and 25 cm, the buzzer plays a longer beep.
* If the distance is between 25 cm and 35 cm, the buzzer plays an even longer beep.
* The buzzer beeps are controlled using the ‘tone()’ and ‘noTone()’ functions with appropriate delay durations.

5.4.2.2 Pulse Rate Monitoring:

* The ‘PulseSensorPlayground’ library is used to monitor the user's pulse rate.
* It checks for the beginning of a heartbeat using the ‘pulseSensor.sawStartOfBeat()’ function.
* If a heartbeat is detected, the pulse rate (BPM) is obtained using the ‘pulseSensor.getBeatsPerMinute()’ function.
* If the pulse rate is within the range of 60 to 100 BPM, it is printed to the Serial Monitor.
* If the pulse rate is not within the range for a continuous period of time, a message is sent to the registered number informing them of the driver’s pulse abnormality.
* A warning message is printed on an LCD for the driver to see.

The code provides a simple integration of proximity sensing and pulse rate monitoring functionalities, along with buzzers, GSM, and LCD. It continuously loops through the ‘loop()’ function, performing both tasks simultaneously. When an obstacle is detected within a certain range, the buzzer emits different beeps based on the distance. Additionally, if a heartbeat is detected within the specified range, or outside the specified threshold, the necessary functions are executed.

### 5.4.3 Method Of Implementation

5.4.3.1 Proximity Sensing:

Step 1: Gather the necessary hardware:

* Arduino UNO Board
* Ultrasonic Sensor (HC-SR04)
* Jumper cables (male-to-male and male-to-female)
* Buzzer

And install Arduino IDE

Step 2: Connect the hardware components to the board:

* VCC pin of the ultrasonic sensor to the 5V pin on the Arduino board.
* GND pin of the ultrasonic sensor to the GND pin on the Arduino board.
* TRIG pin of the ultrasonic sensor to digital pin 9 on the Arduino board.
* ECHO pin of the ultrasonic sensor to digital pin 10 on the Arduino board.
* Connect the positive terminal of the buzzer to digital pin 6 on the Arduino board and the negative terminal to the GND pin on the Arduino board.

Step 3: Install the Arduino IDE and open a new sketch.

Step 4: Type a code for the connections, along with the necessary libraries.

Step 5: Upload the code to the Arduino board:

* Connect the Arduino UNO board to your computer using a USB cable.
* Wait for the code to be compiled and uploaded.

Step 6: Power the System:

* Power up the Arduino board by connecting it to a power source or using the USB cable connected to your computer.

Step 7: Monitor the output:

* Make sure the baud rate is set to 9600.
* The Serial Monitor will display the distance of the detected obstacle when it is within the specified ranges (15-25 cm, 25-35 cm, and less than 15 cm).

Step 8: Test the proximity sensing:

* Place an obstacle within the sensing range of the ultrasonic sensor and observe the output in the Serial Monitor.
* The buzzer should emit different beeps depending on the distance of the obstacle.
* Verify that the distance readings and buzzer beeps correspond to the expected output based on the code.

5.4.3.2 Pulse Rate Monitoring:

  Step 1: Gather the necessary hardware:

* Arduino UNO Board
* Pulse Sensor Module
* Jumper cables (male-to-male and male-to-female)
* GSM Module (GSM800C)
* LCD (16x2)

Step 2: Connect the hardware components to the board:

* VCC pin of the Pulse Sensor to the 5V pin on the Arduino board.
* GND pin of the Pulse Sensor to the GND pin on the Arduino board.
* Purple wire (PulseWire) pin of the Pulse Sensor to the A0 pin on the Arduino board.

Step 3: Install the Arduino IDE and open a new sketch.

Step 4:  Install the PulseSensorPlayground library.

Step 5: Type a code for the connections, along with the necessary libraries.

Step 6: Upload the code to the Arduino board:

Step 7: Power up the system:

Step 8: Power up the Arduino board by connecting it to a power source or using the USB cable connected to your computer.

Step 9: Monitor the output:

* Make sure the baud rate is set to 9600.
* The Serial Monitor will display the pulse rate (BPM) when a heartbeat is detected.
* Additionally, if the pulse rate is within the range of 60 to 120 BPM, it will be displayed as "BPM: XXX" (where XXX is the pulse rate).
* If a heartbeat is detected but the pulse rate is not within the specified range for a continuous period of time, a message is sent to the registered number informing them of the driver’s pulse abnormality and a warning message is printed on an LCD for the driver to see.

Step 10: Test the pulse rate monitoring:

* Place your finger on the Pulse Sensor module to detect your heartbeat.
* The pulse rate (BPM) should be displayed when a heartbeat is detected.
* Verify that the pulse rate readings and messages correspond to the expected behaviour based on the code.

# TESTING AND VALIDATION

## 6.1 DROWSINESS DETECTION

### 6.1.1 Design of test cases and scenarios

* Test Case 1: Verify Eye Aspect Ratio and Mouth Aspect Ratio Calculations
* Inputs: Known Eye Aspect Ratio and Mouth Aspect Ratio values
* Expected Output: System should calculate EAR and MAR values and verify.
* Actual Output: Pass
* Test Case 2: Verify if the system is able to recognise a drowsy driver, Eye Aspect Ratio and Mouth Aspect Ratio Calculations
* Inputs: With Eye Aspect Ratio and Mouth Aspect Ratio values, check if the values are below threshold.
* Expected Output: Alarm should be activated only if the EAR and MAR values are lesser than threshold values.
* Actual Output: Pass
* Test Case 3: Verify if the alarm is triggered correctly.
* Inputs: If the driver is said to be drowsy, the alarm should be activated.
* Expected Output: Alarm should be activated on time.
* Actual Output: Pass

### 6.1.2 Validation

For the driver drowsiness detection system, validation seeks to determine whether or not it works as intended and is capable of providing accurate results. A variety of both real-time and pre-recorded tests can be conducted to ensure that the system detects drowsy drivers accurately. This includes testing the accuracy of the calculators for Eye Aspect Ratio and Mouth Aspect Ratio by comparing the actual output of the calculations to the expected output. Additionally, user feedback and reviews can be analysed to assess the accuracy of the system in detecting drowsiness and effectively alerting the driver in time.

Furthermore, the data used for training the system can also be assessed for reliability and accuracy. By cross-checking the data with existing records, the system can be trained to become more reliable and accurate. Finally, the system performance can be monitored in real-time to detect any discrepancies and errors in order to make the necessary adjustments.

 With all the testing and evaluation conducted, it can be concluded that the driver drowsiness detection system is reliable and effective. The system is equipped with advanced technology to accurately detect drowsiness and alert drivers on time, improving safety on roads significantly. In addition, users can rest assured knowing that the system is regularly monitored and updated to ensure maximum accuracy.

## 6.2 Palm Vein Recognition

### 6.2.1 Design of test cases and scenarios

* Test Case 1:
* Input: Valid dataset directory with a set of labeled images.
* Expected Output:

The code should successfully load the dataset, resize the images to a specified size (e.g., 200x200 pixels), and print the shapes of the images and labels arrays.

* Actual Output:

Images shape: (N, H, W) where N is the number of images in the dataset, H is the height of the resized images, and W is the width of the resized images.

Labels shape: (N,) where N is the number of images in the dataset.

* Test Case 2:
* Input: Valid test image path from the dataset.
* Expected Output: The code should successfully read the test image, predict the person in the image using the trained recognizer, and print the predicted person's name and confidence level.
* Actual Output:

Predicted person: [Person's Name] (the name of the recognized person in the test image).

Confidence: A floating-point value indicating the confidence level of the prediction (e.g., 78.9%).

* Test Case 3:
* Input: Invalid dataset directory or non-existent directory.
* Expected Output: The code should print an error message indicating that the dataset directory is invalid or does not exist.
* Actual Output: Error reading dataset directory: [Directory Path]

### 6.2.2 Validation

1. Dataset Loading and Preprocessing:
   * The code should successfully load and preprocess the palmprint dataset, storing images and labels.
   * The shapes of the images and labels arrays should be printed correctly.
   * Errors in loading or preprocessing should be handled and appropriate error messages displayed.
2. Recognition Model Training:
   * The code should create a recognizer using the LBPH Face Recognizer algorithm and train it using the dataset.
3. Testing the Recognizer:
   * The code should successfully read a test image.
   * The trained recognizer should predict the person in the test image.
   * The predicted person's name and confidence level should be printed correctly.
   * Errors in reading the test image should be handled and an appropriate error message displayed.

## 6.3 Iris Recognition

### 6.3.1 Design of test cases and scenarios

* Test Case 1: Verify image preprocessing
* Input: Image path
* Expected Output: Preprocessed image (blurred)
* Actual Output: Preprocessed image (blurred)
* Test Case 2: Verify iris feature extraction
* Input: Preprocessed image
* Expected Output: Iris features (numpy array)
* Actual Output: Iris features (numpy array)
* Test Case 3: Verify training of iris model
* Input: Database directory, encoding model path
* Expected Output: Iris encodings saved to the encoding model file
* Actual Output: Iris encodings saved to the encoding model file
* Test Case 4: Verify testing of iris model (matched case)
* Input: Encoding model path, test image path, threshold
* Expected Output: Matched label, confidence value
* Actual Output: Matched label, confidence value
* Test Case 5: Verify testing of iris model (unmatched case)
* Input: Encoding model path, test image path, threshold
* Expected Output: "Unmatched" label, 0.0 confidence value
* Actual Output: "Unmatched" label, 0.0 confidence value
* Test Case 6: Verify printing of results
* Input: Matched label, confidence value
* Expected Output: Result: Matched, Person's name: [matched label], Confidence: [confidence value]%
* Actual Output: Result: Matched, Person's name: [matched label], Confidence: [confidence value]%

### 6.3.2 Validation

For the iris detection system, validation aims to ensure that the system accurately matches iris images and provides reliable results.

Here are three concise points for validating the iris detection system:

1. Image Matching Validation: Compare the output labels and confidence values of the system with expected labels for a set of test images with known labels. Calculate the accuracy of the system's iris matching capability.
2. Threshold Evaluation: Test the system using different threshold values and analyze the impact on matching results. Find the optimal threshold value that balances accuracy and reliability, considering false positives and false negatives.
3. Cross-Validation and User Feedback: Conduct cross-validation using a diverse dataset and evaluate the system's ability to generalize and match unseen iris images accurately. Gather user feedback to assess matching accuracy and overall satisfaction with the system, making improvements based on reported issues.

These three points cover the essential aspects of validating the iris detection system, including matching accuracy, threshold optimization, and feedback analysis.

## 6.4 Pulse Rate Detection And Proximity Sensing

### 6.4.1 Design of test cases and scenarios

* Test Case 1:
* Input:
* Proximity Sensor: Distance = 20 cm
* Pulse Sensor: BPM = 80
* Expected Output:
* Buzzer should sound for 0.1 seconds.
* Serial Output:
* “The distance of the obstacle is: 20”
* “Your pulse rate is: 80”
* Actual Output: Pass
* Test Case 2:
* Input:
* Proximity Sensor: Distance = 30 cm
* Pulse Sensor: BPM = 110
* Expected Output: Buzzer should sound for 0.5 seconds.
* Serial Output:
* “The distance of the obstacle is: 30”
* “Your pulse rate is: 110”
* Actual Output: Pass
* Test Case 3:
* Input:
* Proximity Sensor: Distance = 10 cm
* Pulse Sensor: BPM = 130 (continuously for a period of time)
* Expected Output:
* Buzzer should sound for 0.5 seconds.
* The GSM module should send a message.
* The LCD should display a warning.
* Serial Output:
* “The distance of the obstacle is: 10”
* “Your pulse rate is: 130”
* Actual Output: Pass

### 6.4.2 Validation

1. Ensure that the hardware components (Proximity Sensor, Pulse Sensor, Buzzer, etc.) are properly connected to the Arduino board as per the code and wiring instructions.
2. Upload the code to the Arduino board using the Arduino IDE.
3. Open the Serial Monitor in the Arduino IDE to monitor the serial output.
4. For proximity sensing validation, place objects at various distances within the sensing range of the Proximity Sensor and observe the buzzer sound and corresponding distance readings on the Serial Monitor. Ensure that the buzzer sounds for the specified durations and that the distance readings are accurate.
5. For pulse rate monitoring validation, place your finger on the Pulse Sensor and check if the pulse rate (BPM) is displayed correctly on the Serial Monitor when a heartbeat is detected. Verify that the buzzer does not sound for pulse rate monitoring.
6. Ensure that the GSM module has a signal and sends messages properly.
7. Check the display of the LCD is functioning properly and displays the messages accurately.
8. Repeat the tests with different inputs to cover various scenarios and verify the expected outputs.

# Conclusion

The "Safe and Secure Driver System" is a technology designed to monitor the behaviour of the driver while driving. It uses various sensors and algorithms to identify potentially harmful behaviours that could endanger not only the driver but also the vehicle and its surroundings. By detecting these behaviours in real-time, the system can proactively intervene to prevent or minimise the impact of an emergency situation.

One of the key features of the system is its ability to actively assist the driver in responding to an emergency situation. For example, if the system detects that the driver is about to collide with an obstacle or another vehicle, it alerts the driver to avoid the collision. This not only helps to prevent accidents but also reduces the severity of the impact if one does occur.

In addition to its safety features, the "Safe and Secure Driver System" also incorporates vein pattern and iris authentication, making it highly secure and less susceptible to theft. This technology uses infrared light to scan the vein patterns in the driver's hand, which are unique to each individual and difficult to replicate. For iris recognition, the features are matched to authenticate the driver. This ensures that only authorised drivers can operate the vehicle, preventing theft and unauthorised use.

Overall, the "Safe and Secure Driver System" is an advanced technology that enhances driver safety and security on the road. Its ability to detect and respond to potentially harmful behaviors in real-time, coupled with its advanced authentication technology, makes it a valuable addition to any vehicle.

## 7.1 Future Scope

Artificial intelligence: Future safety systems will be able to predict and respond to potential hazards more accurately and quickly than ever before.

Vehicle-to-vehicle communication: Connected cars will be able to communicate with each other and share information, improving overall safety for all drivers.

Biometric monitoring: In addition to analysing the driver's behaviour and authenticating the driver, future systems may also incorporate biometric monitoring to detect signs of drowsiness or distraction more rapidly and respond accordingly.

