RSET
RAJAGIRI SCHOOL OF
ENGINEERING & TECHNOLOGY
(AUTONOMOUS)

Mini Project Report On

ANTI SLEEP ALARM SYSTEM FOR DRIVERS

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

ELECTRONICS AND COMMUNICATION ENGINEERING

By

Sania George (U2201183)

Under the guidance of

Ms.S.Santhi Jabarani

Electronics and Communication
Rajagiri School of Engineering & Technology (Autonomous)
(Parent University: APJ Abdul Kalam Technological University)
Rajagiri Valley, Kakkanad, Kochi, 682039
April 2025

CERTIFICATE

This is to certify that the mini project report entitled ***"ANTI SLEEP ALARM FOR DRIVERS"*** *is a bonafide record of the work done by* ***Sania George (U2201183)*** *, submitted to the Rajagiri School of Engineering & Technology (RSET) (Autonomous) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in "Electronics and Communication Engineering " during the academic year 2022-2026.*

Project Guide	**Project Coordinator**	**HOD**
Ms.S Santhi Jabarani	Ms. Neethu Radha Gopan	Dr. Jisa David
Assistant Professor	Assistant Professor	Associate Professor
Dept. of ECE	Dept. of ECE	Dept. of ECE
RSET	RSET	RSET

ACKNOWLEDGMENT

I wish to express my sincere gratitude towards **Rev.Dr.Jaison Paul Mulerikka CMI**, Principal of RSET, and **Dr Jisa David**, Head of the Department of **Electronics and Communication** for providing me with the opportunity to undertake my project, **Anti Sleep Alarm For Drivers**.

I am highly indebted to my project coordinators, **Ms. Neethu Radha Gopan** and **Ms. Soni Monica**, for their valuable support.

It is indeed my pleasure and a moment of satisfaction to express my sincere gratitude to my project guide **Ms. S. Santhi Jabarani** for her patience and all the priceless advice and wisdom she has shared with me.

Last but not the least, I would like to express my sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Abstract

In an era where road safety is paramount, the integration of advanced technologies becomes essential to mitigate the risks associated with driver drowsiness and fatigue-related accidents.In this project, we provide an anti-alarm system to detect drowsy driving, which is made up of modules: Sleep Detection, Steering Pattern Detector, and Seat Posture Detector. After identifying signs of potential fatigue from sleep patterns, steering behavior, and posture, the in-car mobilized system prompts real-time alerts to make the driver alert and avoid an accident. Unlike existing systems that only focus on a single factor like heart rate or head angle, it combines various detection methods into one, making monitoring far more accurate and trustworthy, significantly improving safety and reducing fatigue-related accidents.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Driver fatigue and drowsiness have been major causative factors for road accidents across the globe. Even with advancements in modern car safety, the human element continues to be the most variable and dangerous factor. The development of intelligent systems, especially embedded within vehicles, has made it possible to create novel tools that help counteract such risks. Among these is the anti-sleep alarm system, which proactively observes driver conduct and physiological signs to recognize signals of sleepiness or inattentiveness.

The ESP32 microcontroller, which is characterized by its flexibility and strong wireless communications support, is a perfect fit for embedded real-time applications. Through the integration of the ESP32 with sensors and computer vision libraries, it is feasible to design a comprehensive system that tracks a driver's posture, steering, and eye movements to determine alertness.

In this project, we introduce a multi-modal anti-sleep alarm system based on three force sensors placed inside the driver's seat, a potentiometer on the steering wheel, and a webcam with computer vision algorithms for real-time eye detection. The combination of these heterogeneous sensory modalities increases the accuracy and reliability of drowsiness detection, making the system more sensitive to different indicators of fatigue.

1.2 Problem Definition

Road safety is still seriously threatened by sleepy driving, which frequently goes unnoticed until it is too late. Conventional systems cause false positives or missed detections because they either oversimplify the issue or mainly rely on isolated cues.

The goal of this project is to create a multi-parameter, real-time anti-sleep alarm

system that uses ESP32, a camera, and other sensors to identify indications of driver fatigue. The objective is to develop a non-intrusive, reasonably priced, and flexible solution that can precisely identify behavior associated with fatigue and notify the driver and emergency contacts prior to a critical incident.

1.3 Scope and Motivation

This project aims at developing and implementing an anti-sleep alarm system for preventing road accidents due to driver drowsiness. The range involves hardware integration, software implementation, signal processing, and real-time data processing. The key components involved are: - ESP32 microcontroller as the core processing unit. - Three force-sensitive resistors integrated into the seat to identify posture variations. - A potentiometer fixed on the steering wheel to analyze steering patterns. - A web/USB camera interfaced through OpenCV to monitor eye movements and identify fatigue symptoms.

The motivation is due to the rising interest in road safety and the rapid use of embedded systems and IoT in daily life. With the cost-effectiveness of components such as the ESP32 and open-source libraries available, it is now possible to create smart systems even for developing countries. The project is aligned with international efforts aimed at decreasing automobile accidents and road fatalities. The project also presents opportunities for research and development of automotive safety, human-machine interfaces, and real-time embedded systems.

1.4 Objectives

The main goals of this project are:

- To create an efficient system with ESP32 for real-time driver fatigue monitoring.
- To implement a seating posture detection system with force sensors mounted in the seat cushion.
- To quantify steering behavior fluctuations with a potentiometer attached to the steering wheel.

- To create eye detection using a webcam for detecting prolonged closures or blinking irregularities.
- To sound audible alarms if symptoms of drowsiness are found to avert accidents related to drowsiness.

1.5 Challenges

The following are some of the major challenges engaged in this project:

- Attaining the credibility of posture recognition with different body types of the driver and changing seat habits.
- Potentiometer calibration in identifying slight alterations in steering drivers' habits to various drivers.
- Real-time processing and translating webcam-based detection of eyes in a way to achieve low levels of false detection.
- Effective fusion of multiple sensor data inputs on a resource-constrained platform such as ESP32.
- Mitigation of environmental conditions such as low light or sensor noise that can impact detection performance.

1.6 Assumptions

The system is designed under the following assumptions: The system is designed under the following assumptions:

1. The driver is seated during the entire operation of the vehicle.
2. The system is calibrated for a single-user profile per session.
3. The driving environment provides stable sensor readings (e.g., negligible road vibrations impacting posture sensors).
4. The webcam is correctly and unobstructedly placed for clear facial detection.
5. The system has continuous power supply when in operation.

1.7 Societal / Industrial Relevance

This system has far-reaching implications for society and industry. At the societal level, it promotes road safety through the prevention of accidents caused by driver fatigue proactively. In the transport sector, it can assist fleet operators in tracking driver alertness, possibly lowering liability, enhancing operating safety, and reducing insurance rates. Additionally, its cost-saving aspect makes it a suitable candidate for implementation in budget vehicles, thus encouraging increased adoption.

Report Organization

The report is organized as follows:

- **Chapter 1:** *Introduction* – Gives background, problem statement, scope, objectives, and significance of the project.
- **Chapter 2:** *Literature Review* – Discusses current methods of fatigue detection and points out gaps.
- **Chapter 3:** *Methodology* – Explains the integration process of sensors, coding, and development of logic and working
- **Chapter 4:** *System Design* – Explains the hardware and software design.
- **Chapter 5:** *Results and Discussions* – Outcomes, performance analysis, and system testing.
- **Chapter 6:** *Conclusion and Future Work* – Summarize findings and improvements.

Chapter 2

Literature Survey

Drowsy drivers are one of the most common causes of road accidents everywhere. Many systems have been designed which aim to monitor a driver's drowsiness level and provide alerts to the driver. This literature survey presents different methods concerning drowsiness detection technology for motor vehicle operators.

2.1 IR Sensor Anti Sleep Alarm With Arduino

P. Sandeep Chary.[4] developed an Anti-Sleep Alarm System by Utilizing an IR Sensor that focuses on eye blinking detection using a mounted sensor on spectacles. The hardware consists of an Arduino Nano, Piezo Buzzer, Micro Vibration Motor and an SPST switch. If the driver's eyes are shut for a certain time, a system-alarm is triggered which in turn, prompts vibrations to rouse the driver.
Key features:

- Performed well with spectacles in a dimly illuminated environment.
- Eye attention monitoring (open eyes or closed eyes) detecting in IR reflection mode.
- Affordable and portable, it is operational performed optimally during late-night or long-distance drives.
- However, being yawning or rubbing eyes can be additional reasons that can be mistakenly taken as positives.

2.2 Review on Design and Development of Anti-Sleep Alarm for Drivers

Ankit W. Kolarkar et al.[6] proposed a multi-feature anti-sleep alarm system with eye blink detection, alcohol detection, and vehicle speed control. The system places emphasis on safety by monitoring driver alertness and providing warnings when the driver's eyes are shut for a perceptibly long duration.

Key insights:

- Achieves non-intrusive monitoring through the use of IR-based blink sensors.
- Improves safety with the inclusion of an alcohol detection sensor.
- Proposes vehicle deceleration along with the activation of the parking lights during drowsiness events.
- Can be used in ordinary cars beyond just luxury vehicles.
- The paper also surveyed a number of existing works like:

  Detection based on Eye Closure Ratio (ECR) (Hossain et al.)

  Detection of blink/yawn using computer vision (Hitendra Garg)

  Eyelid movements pattern analysis with temporal difference images (Danghui Liu)

2.3 Drowsiness Detection and Rest Stop Suggestion

A webcam based drowsiness detection system was developed by Sabari Mathavan et al[7] using Eye Aspect Ratio (EAR) and facial landmarks with OpenCV.
Their system:

1. Detects continuous eye closure for more than 800 ms (equivalent to 48 frames on 60 FPS).
2.Emails alerts to the manager.
3.Using Google Places API, suggests nearby rest stops.
4.Logs the data to Firebase and displays it in a mobile app interface (Driver's Lounge).

Key technologies used:

- Dlib for facial landmark detection.
- EAR thresholding for drowsiness detection.
- Google APIs for services pertaining to location.
- Flutter for mobile applications development across all operating systems.

The example illustrates the application of IoT and computer vision, where driver alerting and backend monitoring happen simultaneously.

2.4 Summary and Gaps Identified

2.4.1 Summary

The table below provides a comparison of the reviewed literature based on their key features, advantages, and limitations.

Title/Author	**Advantages**	**Disadvantages**
IR Sensor-based Eye Blink Detection (P. Sandeep Chary et al.)	- Low-cost hardware - Easy to implement - Works in low light	- Requires user to wear spectacles - Can produce false positives due to yawning
Multi-sensor Anti-Sleep Alarm (Ankit W. Kolarkar et al.)	- Combines alcohol and drowsiness detection - Slows down vehicle for safety	- More complex hardware - Less adaptable to real-time vision models
Computer Vision-based EAR Detection (Sabari Mathavan et al.)	- High accuracy using facial landmarks - Non-intrusive - Rest-stop suggestion and alert emails	- Requires decent processing power - Depends on good camera quality

Table 2.1: Summary of Reviewed Anti-Sleep Alarm Systems

2.4.2 Gaps Identified

1. IR-based systems require physical wearables, which may be inconvenient for drivers.
2. Most systems lack remote communication or alert mechanisms to notify emergency contacts.
3. Few approaches integrate real-time cloud storage and mobile monitoring for centralized tracking.
4. Vision-based models are more accurate but not widely adopted in cost-effective solutions.
5. Many systems do not distinguish well between natural blinks and drowsiness-induced eye closure.

2.4.3 Chapter Conclusion

This chapter reviewed various Anti-Sleep Alarm systems and identified the strengths and weaknesses of different technologies. While traditional IR sensor-based methods are simple and cost-effective, computer vision-based systems offer greater accuracy and adaptability. However, there is still room for improvement in terms of comfort, scalability, and real-time communication, which the proposed system aims to address.

Chapter 3

Methodology

3.1 Introduction

This part describes the approach taken for designing the Anti-Sleep Alarm System, which is a safety system with multiple sensors that is capable of detecting driver drowsiness and distraction in real time. In response to the growing number of fatigue-related road accidents, the system integrates three complementary modules: vision-based eye monitoring, seat posture detection, and steering behavior analysis. An ESP32 microcontroller serves as the core processor, interfacing with a webcam, force-sensitive resistors (FSRs), and a potentiometer. Eye closure is monitored by calculating the Eye Aspect Ratio (EAR) using Dlib's facial landmark detection. Simultaneously, FSRs detect improper seating posture, while the potentiometer tracks abnormal or absent steering movement. By fusing data from all t sensors, the system determines driver alertness and triggers warnings through an LCD, audio alerts, and cloud-based notifications via Firebase. The combined approach minimizes false positives through redundancy and threshold logic, ensuring reliability in practical driving conditions.

3.2 Block diagram

3.2.1 Explanation

The Anti-Sleep Alarm System for Drivers is designed to monitor drowsiness, inattentiveness, and improper driving posture in real time. The block diagram depicts the interconnection of core components, each serving a critical function in ensuring driver safety. The system operates through multiple sensor modules, a central

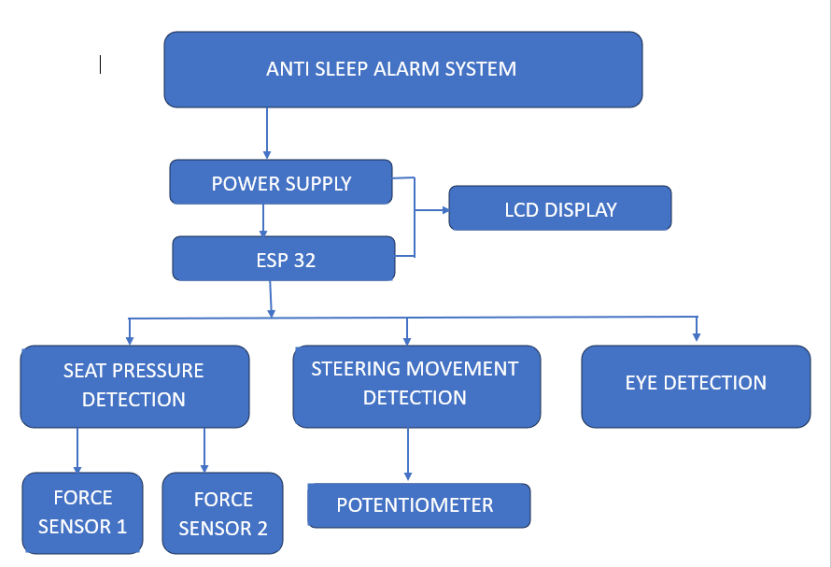


Figure 3.1: Eye Detection using cam

ESP32 microcontroller, and user interface elements such as the LED display. Each block is explained in detail below.

1. Power Supply

The power supply unit ensures all components, including sensors, display units, and the ESP32 microcontroller, receive a constant and controlled voltage. This stability is essential for consistent performance and prevents unexpected shutdowns or sensor errors.

2. ESP32 Microcontroller

The ESP32 acts as the central processing unit. It collects data from three sensor types: force sensors for posture detection, a potentiometer for steering analysis, and a webcam for eye monitoring. Based on predefined logic for each module, the ESP32 evaluates driver status and generates alarms. It also communicates with the LCD to display visual alerts and integrates with Firebase for remote monitoring.

3. Seat Pressure Detection

Driver posture is monitored using three Force Sensitive Resistors (FSRs): Force Sensor 1 (BCK) on the backrest and Force Sensor 2 (SHO) on the upper back or shoulder. Analog signals from these sensors are processed by the ESP32. A reading below 2850 indicates improper posture—such as slouching or head tilting. Alerts

like "Sit on seat" is displayed on the LCD.

4. Steering Movement Detection

Steering behavior is analyzed using a potentiometer that outputs ADC values based on wheel position. Erratic movement is indicated by ADC fluctuations exceeding 900 within a short span, while extended stable values across 55 cycles indicate attentiveness. Dangerous patterns prompt a "Steering Alert!", and safe driving is confirmed with "Steering Safe" on the LCD. This module corresponds to Alert Code 4.

5. Eye Recognition

The webcam captures real-time video of the driver's face, which is analyzed using Dlib's facial landmark detection. The Eye Aspect Ratio (EAR) is computed to detect eye closure. If EAR drops below 0.18 for 20 consecutive frames, drowsiness is confirmed. Upon detection, the LCD displays "DROWSY ALERT!", an audio message like "Wake Up!" is played, and an update is optionally sent to Firebase. The frame counter resets after each alert to ensure responsiveness.

6. LCD Screen

The LCD acts as the main user interface, showing real-time alerts based on ESP32 logic. Messages such as "DROWSY ALERT", "WARNING", or "Sit on seat" inform the driver clearly and promptly of any abnormal behavior or posture.

3.3 Project Progression

The Anti-Sleep Alarm System was designed using a stepwise and systematic approach to ensure modular functionality, accuracy, and successful integration. The project began with a phase of requirement analysis and research, during which the team identified the major causes of driver drowsiness and determined the key features to monitor—namely eye closure, seating posture, and steering wheel movement.

Suitable hardware components were selected based on these specifications, including the ESP32 microcontroller, force-sensitive resistors (FSRs), a rotary potentiometer,

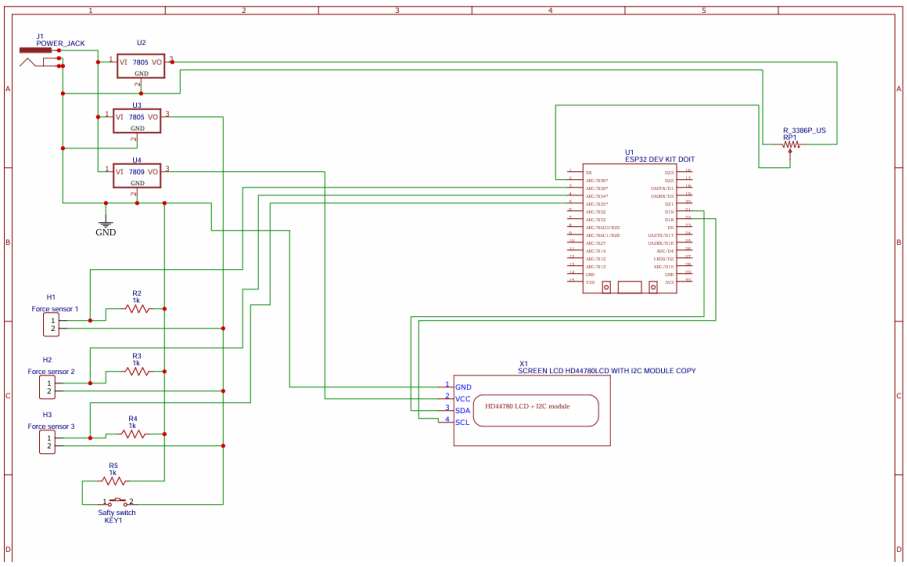


Figure 3.2: Circuit Diagram

and a webcam. Dlib, an open-source machine learning toolkit, was chosen for real-time facial landmark identification due to its robustness and Python compatibility.

Following the planning phase, each module was developed and tested independently. The eye detection module was the first to be implemented. A webcam was connected to the laptop to continuously monitor the driver's facial features. Using Dlib's 68-point facial landmark detection, the system estimated the Eye Aspect Ratio (EAR) to determine the level of eye opening. Threshold values were established to distinguish between regular blinking and prolonged eye closure, which indicates drowsiness. Alerts were displayed on an LCD screen and accompanied by an audible buzzer.

Next, the seat pressure detection module was developed using three FSRs placed at the seat base, backrest, and shoulder area. These sensors were connected to the ESP32, and their analog readings were analyzed to detect poor posture or leaning. Each situation was assigned an alert code, and the corresponding warning was displayed on the LCD.

The steering movement detection module was then created using a rotary potentiometer attached to a simulated steering wheel. The ESP32 continuously monitored angular changes by sampling ADC values.

After the successful development of each module, the system integration phase be-

gan. The ESP32 served as the central control unit, collecting input from all three modules and executing logic to generate alerts. A tiered alarm system was implemented: detection from any one module resulted in a level-1 alert, while simultaneous detection from two or more modules triggered a level-2 high-priority alert.

The final integration included data logging and alert updates to a Firebase cloud database, enabling remote monitoring.

The project's final stage involved thorough testing and validation. Multiple real-world scenarios were simulated to evaluate the system's accuracy and responsiveness. Thresholds and time intervals were fine-tuned based on performance data to reduce false positives and improve reliability. After validation, all components were assembled into a compact prototype. The system successfully detected driver drowsiness and inattention using eye tracking, posture analysis, and steering behavior assessment.

3.4 Work Division

The project development was a collaborative effort, with each team member assigned specific roles to ensure smooth progress and a balanced workload.

Individual Contribution Report

This report outlines my comprehensive contributions to the successful development and execution of the **"Anti-Sleep Alarm System for Drivers."** project. My work spanned across concept development, software coding, hardware interface design, cloud integration, and real-time monitoring implementation. The following are the major areas where I contributed:

1. **Concept Development:** I was actively involved in the early stage of conceptualization for this project. I contributed ideas and logic for detecting driver drowsiness using a multi-sensor approach combining vision, pressure, and steering data. Specifically, I proposed the use of Eye Aspect Ratio (EAR) from facial landmarks for drowsiness detection and the integration of steering

behavior monitoring using a potentiometer. I also helped in outlining how Firebase could be integrated for remote alerting and monitoring. These ideas formed the foundation for system design and block diagram planning.

2. **Code Development for Steering and Potentiometer:** I developed and tested the complete module that uses a potentiometer to monitor steering movement. The ESP32 reads the analog input from the potentiometer, and I wrote code to detect abrupt or erratic changes, which might indicate drowsiness or inattentiveness. I implemented logic to track changes in ADC values and count the number of cycles without movement to determine inattentiveness. The module also raised alerts when steering behavior was unstable. I ensured proper calibration and noise filtering in the code for accurate behavior detection.

3. **Firebase Development:**I was responsible for integrating Firebase with the ESP32 microcontroller. I created the Firebase project, configured the real-time database, and connected it to our local system via Wi-Fi. I wrote and tested the code that sends updates to Firebase whenever a drowsiness event, steering alert, or posture violation was detected. This allowed real-time remote supervision of driver status. My contribution ensured seamless cloud communication for data logging and alerting.

4. **Coding for Eye Detection:** I implemented the Eye Aspect Ratio (EAR) based drowsiness detection module using Python and the Dlib library. I wrote the code that captures video frames, extracts facial landmarks, calculates EAR, and raises alerts if the EAR remains below a set threshold for a specific number of frames (indicating closed eyes). I also added LCD and audio alerts and configured Firebase to log such events. This module plays a critical role in the core functionality of our system.

5. **Conclusion:**My involvement in this project was multi-faceted, covering both hardware interfacing and software implementation. From concept development to the final coding and integration of key modules like steering detection and eye monitoring, I ensured each component was fully functional and integrated with the cloud. The tasks I undertook contributed significantly to building a

reliable and responsive driver monitoring system.

Chapter 4

System Design

4.1 Hardware

– **Laptop with Integrated Webcam**

Used to record real-time video of the driver's face. The webcam feed is processed by Python and OpenCV libraries to track eye movements and assess the level of drowsiness. This module functions independently of the ESP32 and sends drowsiness warnings via Firebase.

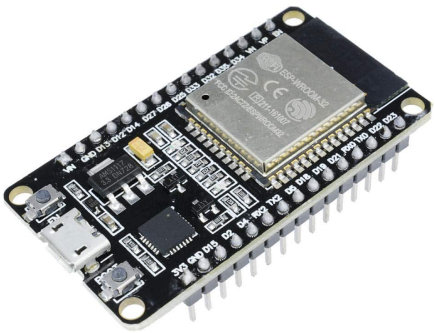


Figure 4.1: ESP32 Development Board with Wi-Fi Interface

– **ESP32 Development Board with Wi-Fi Interface**

A high-power microcontroller with integrated Wi-Fi, used to capture data from the force sensors and potentiometer. It processes analog signals to interpret seat pressure and steering motion and displays alerts on the LCD. Though it supports Wi-Fi, it is not connected to Firebase in this implementation.

Figure 4.2: Force Sensor

– **3x Force Sensors (406 series)**

  Pressure-sensitive sensors used for monitoring seat posture. Installed at the shoulder, back, and bottom of the seat, they detect improper posture or when the driver leaves the seat, and send alerts through the ESP32.

Figure 4.3: Potentiometer

– **Potentiometer (Steering Input)**

  Simulates steering wheel motion and is connected to the ESP32. It supplies variable voltage corresponding to steering angle. Sudden or erratic turns result in voltage spikes, aiding in the detection of distracted or drowsy driving.

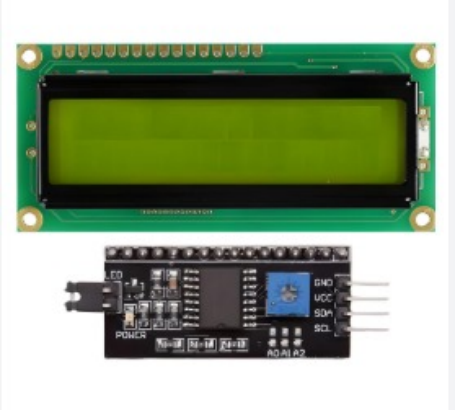


Figure 4.4: LCD Display

– **16x2 LCD Display with I2C Module**

  Displays alert messages such as “SEAT ALERT”, “STEERING ALERT”, or “DROWSINESS ALERT”. The I2C interface simplifies wiring by reducing the number of required pins, which helps create a more compact circuit.

– **Breadboard and Jumper Wires**

  Used for temporary prototyping and flexible connections between sensors, the ESP32, and the LCD display. They facilitate easy testing and adjustment before permanent implementation.

– **5V Power Supply**

  Powers the ESP32 and all attached components. A reliable 5V source is essential to ensure stable system performance.

– **7805 Voltage Regulator**

  Provides a consistent 5V output regardless of input fluctuations. It protects sensitive components like the ESP32 and sensors from voltage spikes.

– **Capacitors**

  Placed alongside the voltage regulator to stabilize output voltage and reduce noise from the power supply. They help ensure smooth power delivery and enhance overall system reliability.

4.2 Software

- **PyCharm IDE (for Eye Detection Script)**

  A robust integrated development environment (IDE) for coding, debugging, and executing the Python-based eye detection script. It supports effective code management, error tracing, and real-time testing of OpenCV and Dlib features.

- **Python Programming Language**

  The primary language used to implement the eye detection algorithm. Python is chosen for its simplicity and the wide availability of computer vision and machine learning libraries, making it ideal for rapid prototyping.

- **OpenCV Library (for Computer Vision)**

  Open Source Computer Vision Library employed to process video streams and identify facial features. OpenCV is used in this project to capture webcam input and help identify eye regions, compute eye aspect ratio (EAR), and determine drowsiness.

- **Dlib Library (for Facial Landmark Detection)**

  A machine learning library employed for facial landmark detection. Dlib assists in identifying certain facial features, like eyes, by tracing 68 landmark points on the face. This information is crucial in determining EAR accurately and whether the eyes of the driver are closed or not.

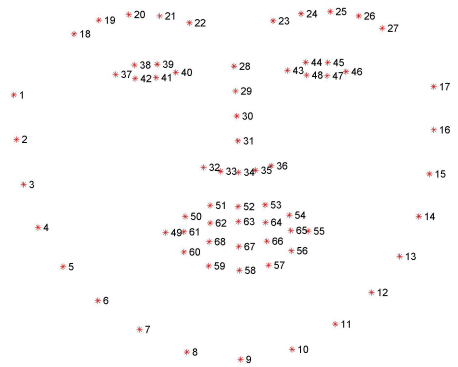


Figure 4.5: 68 Facial Landmarks

4.2.1 Facial Landmark Detection Using Dlib (68-Point Model)

Facial Landmark Detection Using Dlib (68-Point Model) We employ the Dlib 68 facial landmark detector to identify and track the driver's eye movement in real-time with the laptop's webcam. Dlib utilizes a pre-trained shape predictor model to locate 68 face points of importance. When detecting drowsiness, we concentrate on the area of the eyes: Left Eye: landmark points 36 to 41 Right Eye: landmark points 42 to 47 Points that assist us in monitoring the shape and openness of the eyes, enabling us to ascertain whether the eyes are open or closed. This is accomplished in real-time with OpenCV in Python, and there is no extra hardware needed because it is all run on the laptop. When drowsiness is detected, the alert is transmitted to Firebase, which is then utilized by the ESP32 to show the warning on the LCD.

Figure 4.6: Distance measurement

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Figure 4.7: Calculation of EAR Ratio

4.2.2 EAR ratio

Eye Aspect Ratio (EAR) – Eye Blink and Drowsiness Detection

The Eye Aspect Ratio (EAR) is a key metric used to determine if the driver's eyes are open, blinking, or closed for too long. EAR is calculated using the vertical and horizontal distances between six eye landmarks. When the eyes are open, the EAR remains high and stable. During a blink or eye closure, the EAR value drops sharply. If the EAR stays below a set threshold (e.g., 0.25) for a specific duration (e.g., 20 consecutive frames), the system flags the driver as drowsy.

In our project, EAR is continuously computed from the webcam feed. When drowsiness is detected, a `SLEEP ALERT` appears on the webcam window to provide instant visual feedback. Simultaneously, a voice warning is generated using the gTTS (Google Text-to-Speech) library. The alert is also logged to the Firebase Realtime

Database. The ESP32 then retrieves this alert and displays it on a connected LCD screen, giving the driver a real-time warning.

- **gTTS (Google Text-to-Speech)**

  Translates alert messages into speech. Upon detecting drowsiness, this module is employed to give a voice warning to the driver using an announcement such as "Wake up!" to enable the driver to wake up.

- **Firebase Realtime Database (for Cloud Alerts)**

  A NoSQL database hosted in the cloud that stores and syncs data in real-time. Firebase is solely utilized within this system with the eye detection script to transmit drowsiness notifications. The notifications can be picked up wirelessly by other devices like an LCD display.

- **Arduino IDE (for ESP32 Programming)**

  An open-source development platform employed to code and upload C/C++ onto the ESP32 board. It is utilized to read input from the force sensors and potentiometer, process the information, and provide proper alerts on the LCD display.

- **LiquidCrystal_I2C Library (for LCD Control)**

  A library specifically utilized in the Arduino environment to manage 16x2 LCD displays that utilize the I2C communication protocol. It makes it easier to send data to the LCD and enables more effective pin usage on the ESP32.

Chapter 5

Results and Discussions

5.1 Eye Blink Detection

We developed a sleep prevention device with three primary components: eye blink detection, seat pressure sensing, and steering behavior monitoring. The eye detection module utilized OpenCV and the dlib library to measure the Eye Aspect Ratio (EAR), which allows differentiating between normal blink duration and sustained eye closure. Sleepiness was determined when the EAR dropped below 0.18 for 20 consecutive frames. This ensured dependable detection without generating imprecise warnings, as a warning was only declared when eye closure was sustained. This approach minimized false positives and allowed for timely alerts when drowsiness was likely.

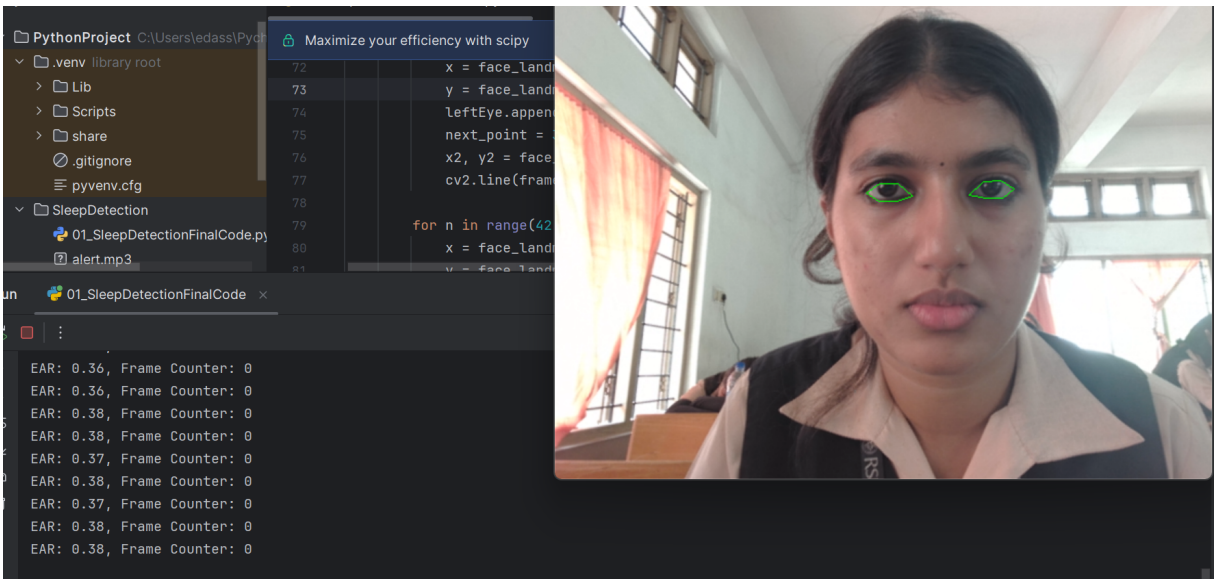


Figure 5.1: When eyes are open

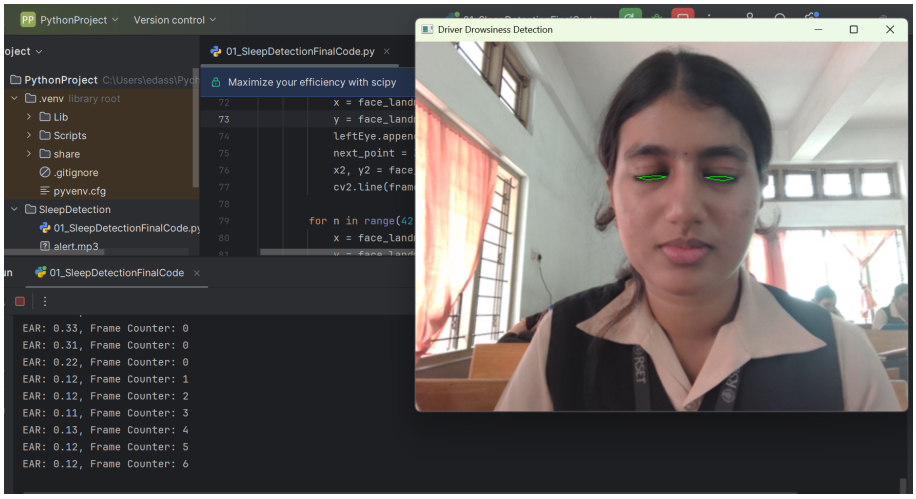


Figure 5.2: Counter increasing when eyes are closed

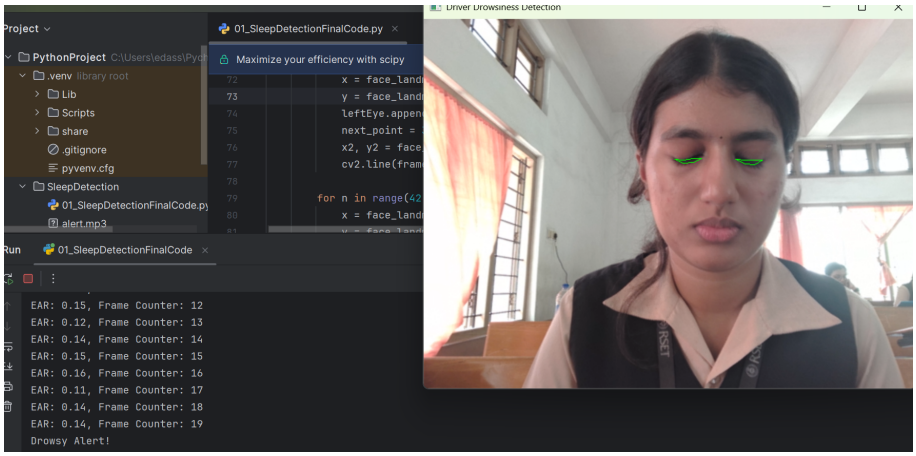


Figure 5.3: When counter reach 20

5.2 Seat Pressure Sensing

The seat pressure sensing modality used three force sensors attached in defined positions: the seat base, seat backrest, and the shoulder location. These sensors continuously measured the driver's posture throughout the duration of a trip. An alert was presented to the driver if any sensor's reading fell below 2850 ADC units, indicating improper posture or a lack of contact with the seat. This subsystem was used to guarantee that the driver maintained a proper seated position, helping reduce inactivity or loss of alertness associated with slouching or posture changes.

Figure 5.4: Warning 1

Figure 5.5: Warning 2

5.3 The steering pattern monitoring

The steering pattern monitoring system utilized a potentiometer to monitor steering behavior changes in real-time. A sudden change in ADC value greater than 900 indicated erratic behavior and triggered alerts. This sensor was especially useful at identifying sudden or inconsistent steering that might indicate drowsiness or distraction. The monitoring system also monitored both consistent and smooth steering behavior to minimize false alerts, maintaining a balance of sensitivity under reliability.

Figure 5.6: Steering alert

Figure 5.7: Safe Conditions

5.4 Challenges Faced

Although implementing the multi-layered approach yielded success, several challenges were noted throughout development. The ESP32 microcontroller experienced power supply instability issues, which led to intermittent performance fluctuations and occasional sensor communication interruptions. The force sensors proved to be sensitive, with a risk of damage during soldering due to exposure to high heat, which compromised their structural integrity.

Additionally, considerable testing time was required to calibrate threshold values for

each detection module, ensuring that they were both accurate and resistant to misclassification. These calibration efforts were essential to fine-tune the system for real-time responsiveness without triggering false alarms.

5.5 Limitations and Observations

The processing capacity of the ESP32-CAM was insufficient to support real-time eye detection, necessitating a shift to a more capable laptop webcam for the computer vision module. Environmental factors such as low lighting conditions and road vibrations further impacted the performance of the system, particularly in the vision and pressure sensing modules. These conditions occasionally interfered with the consistency and accuracy of detection.

Furthermore, a GPS-based alert system for emergency contact and location sharing was intended but could not be implemented due to the unavailability of the GPS module during the development phase.

5.6 Conclusion

Despite challenges, the system effectively detected key signs of driver drowsiness. Using affordable components kept costs low without sacrificing functionality. The multi-layered detection strategy added reliability, as one module's failure could be offset by others. This redundancy makes it suitable for real-world use, especially for long-distance and night-time drivers. It's an affordable, practical solution for enhancing road safety.

Chapter 6

Conclusions & Future Scope

6.1 Conclusion

The Anti-Sleep Alarm for Drivers project was developed to address the critical issue of driver drowsiness and its contribution to road accidents. By leveraging computer vision techniques to monitor eye blink patterns through a laptop camera, the system offers a non-intrusive and efficient method to detect signs of fatigue in real time. Once drowsiness is detected, an alert signal is sent from the ESP32 to the buzzer and a warning is displayed. The buzzer is stopped once the driver is awake hence avoiding accidents.

This project offers a comprehensive method of driver monitoring by effectively integrating computer vision with embedded systems and communication modules. It is accessible for additional development and real-world implementation due to the utilization of open-source tools and widely accessible components.

6.2 Future scope

A GPS module could be added to the project to transmit the driver's precise location when drowsiness is detected. Additionally, by taking into account variables like head position and yawning, machine learning models can be trained for more precise and customized drowsiness detection. Dashboards for fleet management monitoring in real time can be made possible through integration with cloud platforms. To give the user logs, system status, and alerts, a mobile application could also be created. Lastly, it may be possible to reduce future iterations to a single embedded unit that can be directly integrated into car dashboards.

Reference

1. Y.-K. Cheong, "Potentiometer-type steering sensor," U.S. Patent 5,754,091, May 19, 1998.

2. P. C. Sekhar, K. S. Reddy, K. S. V. Reddy, G. S. Bhargavi, and D. Manasa, "Anti-Sleep Alarm for Drivers," *Int. J. Res. Appl. Sci. Eng. Technol. (IJRASET)*, vol. 11, no. 11, pp. 580–584, Nov. 2023, doi: 10.22214/ijraset.2023.56505.

3. "Interfacing Pressure Sensor with Arduino," *STEMpedia*, Jul. 25, 2023. [Online]. Available: `https://thestempedia.com`

4. P. S. Chary, S. Pranay, N. S. Kishore, and M. Ravi Kumar, "Anti-Sleep Alarm for Drivers," *J. Eng. Sci.*, vol. 14, no. 6, pp. 257–260, 2023. [Online].

5. D. Pandey, "Eye Aspect Ratio (EAR) and Drowsiness detector using dlib," *Medium*, Apr. 21, 2021. [Online]. Available: `https://medium.com`

6. A. W. Kolarkar, N. R. Sontakke, G. A. Kukadkar, Y. K. Gujar, A. Kamble, N. Dhande, and A. K. Singh, "Review on to Design and Develop an Anti-Sleep Alarm for Drivers," *International Journal of Scientific Research & Engineering Trends*, vol. 10, no. 1, pp. 31–33, Jan.–Feb. 2024.

7. R. B. S. Mathavan, V. Rohitram, C. Ashhwath, and P. Sasikumar, "Drowsiness Detection and Rest Stop Suggestion," *Journal of Physics: Conference Series*, vol. 2115, no. 1, 012028, 2021, doi: `https://doi.org/10.1088/1742-6596/2115/1/012028`.

Appendix A: Presentation

ANTI SLEEP ALARM FOR DRIVERS

GUIDE : Ms.S.SANTHI JABARANI

TEAM MEMBERS:
Neha Maria Thomas - U2201150
Niya Paul - U2201156
Raiga Mariyam Regi - U2201170
Sania George - U2201183

TABLE OF CONTENTS:

INTRODUCTION

BACKGROUND

- Drowsy driving is a leading cause of road accidents, contributing to thousands of fatalities worldwide.
- Fatigue impairs reaction time, decision making and awareness, increasing accident risks.
- Traditional counter measures like rolling down windows or drinking coffee provide only temporary relief.
- Modern driver monitoring systems aim to detect drowsiness early and prevent accidents.

WHY IS THIS IMPORTANT ?

- Enhancing Road Safety : Detecting fatigue early helps prevent accidents and save lives.
- Multi-Layered Detection : Our project uses multiple indicators – Eye blinking , steering pattern and posture detection.
- Cost-Effective Solution : Unlike expensive commercial systems, our design is affordable and easy to implement in any vehicle.
- Real-World Applications : Beneficial for long–haul truck drivers, taxi drivers and individuals prone to fatigue while driving.

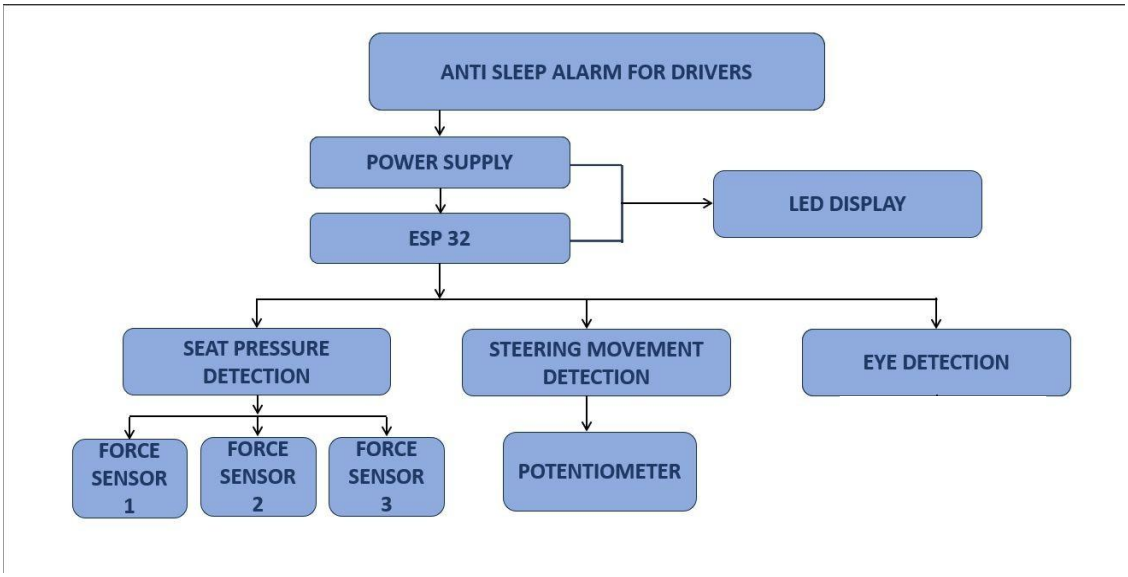
OBJECTIVE

Our solution is a multi-layered cost effective driver drowsiness detection system that enhances road safety by integrating multiple detection mechanisms.

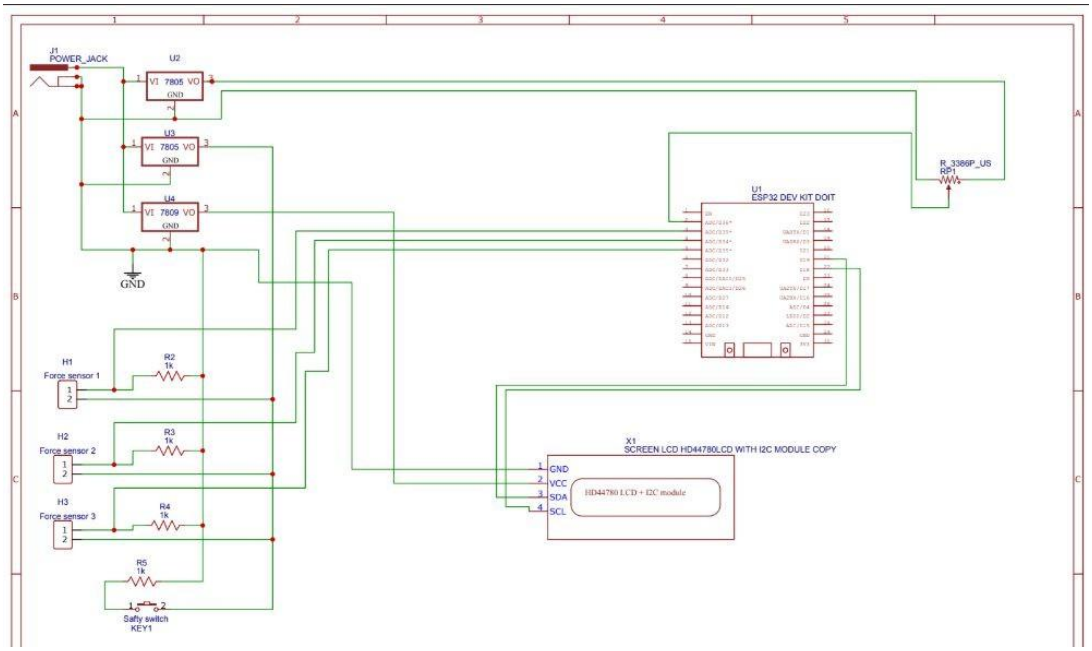
- Three independent detection methods : Eye blink tracking, Steering behaviour and posture detection.
- This redundancy ensures that if one method fails(e.g., poor lighting affecting eye detection), the other mechanisms can still detect drowsiness, reducing false negatives.

METHODOLOGY

BLOCK DIAGRAM



CIRCUIT DIAGRAM



1.EYE BLINK TRACKING

WORKING PRINCIPLE:

- Dlib detects 68 facial landmarks, including the eye region.
- The Eye Aspect Ratio (EAR) is calculated using the distance between specific eye landmarks to measure openness.
- If EAR drops below 0.18 for 20 consecutive frames, the system confirms drowsiness.
- This prevents false alerts from blinking and ensures only sustained eye closure triggers an alarm.

SYSTEM RESET AFTER ALERT :

- After detecting drowsiness and triggering an alert, the frame counter is reset to 0.
- This prevents the system from continuously raising alarms while allowing it to detect new drowsiness events.
- Once the driver opens their eyes and EAR returns to normal, the system resumes monitoring for the next drowsiness detection.

ALERTING MECHANISM:

- LCD Display: Shows "DROWSY ALERT!“
- Audio Alert: Plays "Wake up!“
- Firebase Update: Sends a sleep alert for remote monitoring.

2.SEAT PRESSURE DETECTION(Force Sensor Based)

WORKING PRINCIPLE:

- Sensors measure seat pressure at three key points: Seat (SET), Backrest (BCK), and Shoulder (SHO). If the pressure drops below 2850 ADC value, it indicates improper posture or movement.
- Real-time monitoring ensures continuous tracking of the driver’s seating position.

IMPROPER SEATING DETECTION:

- If SET sensor is low → Driver not seated properly.
- If BCK sensor is low → Driver not sitting straight.
- If SHO sensor is low → Driver leaning or head tilted.

ALERTING MECHANISM:

- LCD Display: Shows warnings like "Sit Straight" or "Keep Head Straight".
- Alert Codes: Values 1 to 3 indicate different seating violations.

3.STEERING MOVEMENT DETECTION (Potentiometer based)

WORKING PRINCIPLE:

- The potentiometer tracks real-time steering rotation using ADC values.
- A sudden change (>900 ADC) indicates erratic steering, triggering an alert.
- If consistent erratic movements are detected, it signals loss of control or drowsiness.
- Gradual changes in ADC values are ignored to prevent false alarms.

IMPROPER STEERING DETECTION:

- If ADC change > 900 → Sudden steering movement detected (erratic driving).
- If steering remains unstable for multiple cycles → Possible loss of control or drowsiness.
- If no abrupt movements for 55 cycles → Steering considered safe.

ALERTING MECHANISM:

- LCD Display: Shows messages like "Steering Alert!" or "Steering Safe".
- Alert Code 4: Indicates unsafe or sudden steering movements.

OVERALL WORKING

- Imagine a driver is on a long journey, navigating highways late at night. To ensure safety, the vehicle is equipped with a system that monitors posture, steering movements, and eye behavior in real time.
- As the driver settles in, the seat pressure sensors continuously check if they are sitting correctly. If they start to slouch or shift into an incorrect posture, the system detects the change. A warning appears on the LCD screen saying "Sit straight" or "Sit on seat," and a buzzer sounds to alert them.
- While driving, the steering sensor tracks movement. If the driver suddenly jerks the wheel or shows irregular steering behavior, the system recognizes the unusual pattern. A "Steering Alert" is activated,the LCD flashes a warning, and an alarm sounds to regain their attention.

- Meanwhile, the camera captures real-time video of the driver’s face. Using OpenCV and dlib, the system analyzes eye movements. If the driver's Eye Aspect Ratio (EAR) drops below 0.25 for more than 20 frames, it indicates drowsiness. Instantly, a voice message plays, warning them to wake up.
- If any unsafe condition is detected—whether poor posture, erratic steering, or drowsiness—the system responds immediately. A buzzer sounds, a voice message plays, and the LCD screen displays a warning.
- This real-time monitoring helps keep the driver alert, significantly reducing the risk of accidents caused by fatigue or inattentiveness.

TOOLS USED

HARDWARE COMPONENTS

- ESP32 Development Board
- Voltage regulator
- Capacitor
- Force Sensors
- Potentiometer
- I2C LCD Display
- Reset button
- Buzzer

SOFTWARE COMPONENTS

- Arduino IDE
- ESP32 Board Manager
- LiquidCrystal_AIP31068_I2C Library
- pycharm
- dlib library
- open cv
- scipy library

ALGORITHM FOR STEERING PATTERN AND POSTURE DETECTION

1. Initialize the system by setting up the serial communication, LCD display, and timers. The initial values from the seat and steering sensors are recorded.
2. Continuously monitor seat pressure sensors to check if the driver is sitting properly. If the seat is empty, or if the driver is in an incorrect posture, an alert is triggered.
3. Monitor steering movement using a potentiometer. If sudden or unusual movements are detected, a **"Steering Alert"** is activated.

4. Compare sensor readings with predefined thresholds to determine whether the driver is in a safe or drowsy state.
5.Update the LCD display to show relevant warnings, such as **"Sit on seat"**, **"Sit straight"**, or **"Steering Alert!"**, depending on detected conditions.

6.Log real-time data on the serial monitor for debugging and system analysis.

7.Repeat the monitoring process continuously to detect drowsiness in real-time and ensure safety.

ALGORITHM FOR EYE DETECTION

1. **Initialize Firebase** for real-time drowsiness alerts.
2. **Set up voice notification** using gTTS for audio warnings.
3. Import libraries (OpenCV, dlib, NumPy, SciPy, pyrebase) for image processing and cloud updates.
4. **Load dlib's 68 facial landmarks model** for detecting eye positions.
5. **Define thresholds:** EAR_THRESHOLD=0.18 (eye closure) and FRAME_LIMIT=20(sustained closure).
6. **Convert frames to grayscale** for faster processing.

7. **Detect faces** using dlib's face detector.
8. **Identify and draw eye landmarks** for visualization.
9. **Calculate Eye Aspect Ratio (EAR)** to determine if eyes are open or closed.
10. **Increase frame counter if EAR < 0.18,** else reset it.

OUTPUT

1. ALERT 1 : SHOULDER SUPPORT LOST
- No pressure detected on the shoulder-level sensor.
- Driver's posture is incorrect. Please adjust your seating position.
- Eye status : Checking

2. ALERT 2 : BACKREST SUPPORT LOST
- No pressure detected on the backrest sensor.
- Improper posture detected. Maintain proper back support for safe driving.
- Eye status : Checking

3.ALERT 3 : SEAT OCCUPANCY LOST

- No pressure detected on the seat sensor .Possible movement or absence detected. Ensure proper seating to avoid distractions.
- Eye status : Checking

4.ALERT 4 : SUDDEN STEERING ALERT DETECTED

- Rapid steering movement identified.
- Potential loss of control detected. Drive carefully to maintain stability.
- Eye status : Checking

5.DROWSINESS DETECTED

- Eye Status :Eyes closed for more than 20 frames

LIMITATIONS AND CHALLENGES FACED

1. ESP32 Compatibility & Power Issues : Unstable power supply caused performance fluctuations and communication failures.
2. Coding issue: Debugging and setting the threshold values took time.
3. While FRC cables are flexible, repeated bending or excessive force can break internal wires.
4. Soldering of force sensors: Force sensors are made of flexible polymer materials that melt or get damaged under excessive heat.
5. Eye detection using ESP32 is not possible. Had to switch to webcam.

WORKDIVISION

RAIGA MARIYAM REGI	Assemble Circuit Components, Circuit Wiring, Hardware Debugging,concept development for eye detection and database research
NIYA PAUL	Circuit diagram, Circuit Wiring and Technical Research,database research and debugging
SANIA GEORGE	Concept development , Code Development for steering and potentiometer , firebase development and coding for eye detection
NEHA MARIA THOMAS	Concept development , Code Testing , Design Presentation,coding for eye detection And debugging

CONCLUSION

Eye Detection: Detects drowsiness using EAR, triggering alerts.
Seat Pressure Monitoring: Identifies improper posture using force sensors.
Steering Detection :Tracks sudden movements via a potentiometer.
Real-Time Alerts: ESP32 & Firebase enable instant notifications.
Impact: Prevents fatigue-related accidents and enhances road safety.

FUTURE IMPLEMENTATION

- EMERGENCY CONTACT AND LOCATION SHARING SYSTEM(using GPS module).
- POSTURE ACCURACY using in built datasets.
- REDUCE SPEED in order to prevent accidents.

REFERENCE

1. Y.-K. Cheong, "Potentiometer-type steering sensor," *U.S. Patent* 5,754,091, May 19, 1998.
2. P. C. Sekhar, K. S. Reddy, K. S. V. Reddy, G. S. Bhargavi, and D. Manasa, "Anti-Sleep Alarm for Drivers," *Int. J. Res. Appl. Sci. Eng. Technol. (IJRASET)*, vol. 11, no. XI, pp. 580–584, Nov. 2023, doi: 10.22214/ijraset.2023.56505.
3. Interfacing Pressure Sensor with Arduino," *STEMpedia*, Jul. 25, 2023. [Online].
4. P. S. Chary, S. Pranay, N. S. Kishore, and M. Ravi Kumar, "Anti-Sleep Alarm for Drivers," *J. Eng. Sci.*, vol. 14, no. 6, pp. 257–260, 2023. [Online]
5. D. Pandey, "Eye Aspect Ratio(EAR) and Drowsiness detector using dlib," Medium, Apr. 21, 2021. [Online].

Appendix B:CODE

```
import cv2
import dlib
from scipy.spatial import distance
from gtts import gTTS
import os
import time
import pyrebase
import tempfile

# ------------------------------------ Firebase Initialization ------------------------------------------------

firebaseConfig = {
  "apiKey": "AIzaSyCLU2w5XcrPmMYDCGxlSlZou55tv1nGfHQ",
  "authDomain": "dispmeddem.firebaseapp.com",
  "databaseURL": "https://dispmeddem-default-rtdb.firebaseio.com",
  "projectId": "dispmeddem",
  "storageBucket": "dispmeddem.firebasestorage.app",
  "messagingSenderId": "77501478469",
  "appId": "1:77501478469:web:50c261c14b7ac27f6f952c",
  "measurementId": "G-KW3CVHRJ10"
}
firebase = pyrebase.initialize_app(firebaseConfig)
db = firebase.database()

# ------------------------------------ Voice Notification Setup ------------------------------------------------

def play_audio_message(message):
    with tempfile.NamedTemporaryFile(delete=False, suffix=".mp3") as temp_audio:
        temp_filename = temp_audio.name
    tts = gTTS(text=message, lang='en', slow=False)
    tts.save(temp_filename)
    os.system(f"start {temp_filename}")  # Windows

# ------------------------------------ EAR Calculation Function ------------------------------------------------
def calculate_EAR(eye):
    A = distance.euclidean(eye[1], eye[5])
    B = distance.euclidean(eye[2], eye[4])
    C = distance.euclidean(eye[0], eye[3])
    ear_aspect_ratio = (A + B) / (2.0 * C)
    return ear_aspect_ratio

# ------------------------------------ Firebase Update Function ------------------------------------------------
def firebase_update_sleep():
    data = {"sleep": "ALERT"}
```

```
    db.child("KL05S6628").set(data)

# ------------------------------------ Sleep Detection Setup ---------------------------------------------------

cap = cv2.VideoCapture(0)  # Change to the appropriate camera index or RTSP URL
hog_face_detector = dlib.get_frontal_face_detector()
dlib_facelandmark = dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")

# Drowsiness detection parameters
EAR_THRESHOLD = 0.18  # Adjust this based on testing
FRAME_LIMIT = 20  # Number of consecutive frames required to confirm sleep
frame_counter = 0

while True:
    ret, frame = cap.read()
    if not ret:
        break  # Exit loop if the camera is not providing frames

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = hog_face_detector(gray)

    for face in faces:
        face_landmarks = dlib_facelandmark(gray, face)
        leftEye = []
        rightEye = []

        for n in range(36, 42):
            x = face_landmarks.part(n).x
            y = face_landmarks.part(n).y
            leftEye.append((x, y))
            next_point = 36 if n == 41 else n + 1
            x2, y2 = face_landmarks.part(next_point).x, face_landmarks.part(next_point).y
            cv2.line(frame, (x, y), (x2, y2), (0, 255, 0), 1)

        for n in range(42, 48):
            x = face_landmarks.part(n).x
            y = face_landmarks.part(n).y
            rightEye.append((x, y))
            next_point = 42 if n == 47 else n + 1
            x2, y2 = face_landmarks.part(next_point).x, face_landmarks.part(next_point).y
            cv2.line(frame, (x, y), (x2, y2), (0, 255, 0), 1)

        left_ear = calculate_EAR(leftEye)
        right_ear = calculate_EAR(rightEye)
```

```
        EAR = round((left_ear + right_ear) / 2, 2)

        if EAR < EAR_THRESHOLD:
            frame_counter += 1  # Increase frame count if EAR remains low
        else:
            frame_counter = 0  # Reset if eyes are open

        if frame_counter >= FRAME_LIMIT:  # Trigger alert after sustained low EAR
            cv2.putText(frame, "DROWSY ALERT!", (20, 100), cv2.FONT_HERSHEY_SIMPLEX, 2,
(0, 0, 255), 4)
            cv2.putText(frame, "Wake up!", (20, 400), cv2.FONT_HERSHEY_SIMPLEX, 2, (0, 0,
255), 4)
            print("Drowsy Alert!")
            firebase_update_sleep()
            play_audio_message("Wake up! You are feeling sleepy.")
            frame_counter = 0  # Reset after alert

        print(f"EAR: {EAR}, Frame Counter: {frame_counter}")

    cv2.imshow("Driver Drowsiness Detection", frame)

    key = cv2.waitKey(1)
    if key == 27:  # Press ESC to exit
        break

cap.release()
cv2.destroyAllWindows()
```

```
  "apiKey": "AIzaSyCLU2w5XcrPmMYDCGxlSIZou55tv1nGfHQ",
  "authDomain": "dispmeddem.firebaseapp.com",
  "databaseURL": "https://dispmeddem-default-rtdb.firebaseio.com",
  "projectId": "dispmeddem",
  "storageBucket": "dispmeddem.firebasestorage.app",
  "messagingSenderId": "77501478469",
  "appId": "1:77501478469:web:50c261c14b7ac27f6f952c",
  "measurementId": "G-KW3CVHRJ10"
*****************************************************************************************************/
#include<Arduino.h>
#include <SoftSPIB.h>
#include <LiquidCrystal_AIP31068_I2C.h>
#include <Arduino.h>
#if defined(ESP32)
  #include <WiFi.h>
#elif defined(ESP8266)
  #include <ESP8266WiFi.h>
#endif
#include <Firebase_ESP_Client.h>
#include <addons/TokenHelper.h>
#include <addons/RTDBHelper.h>
#define WIFI_SSID "Embedded123"
#define WIFI_PASSWORD "Embedded54321"
#define API_KEY "AIzaSyCLU2w5XcrPmMYDCGxlSIZou55tv1nGfHQ"
#define DATABASE_URL "dispmeddem-default-rtdb.firebaseio.com"
#define USER_EMAIL "jbcproject2020@gmail.com"
#define USER_PASSWORD "Embedded54321"

#define BP 13
#define STR 35
#define SET 34
#define BCK 39
#define SHO 36
#define SW 32
#define THRS 1100

LiquidCrystal_AIP31068_I2C lcd(0x3E,16,2);
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
/*****************************************************************************************************/
 int s1,s2,s3;
```

```
 int alert=0;
 int seat_alert =0;
 int sa_1,sa_2,sa_3,sa_4;
 int steering_alert =0;
 int adc,adc_pre;
 int pt,ot,tc=0;
 int pt1,ot1,tc1=0;
 int bpt,bot,btc=0;
 int SL = 0;

 String msg;
/*************************************************************************************************************/
void readSeatpreassure();
void steering();
void disp();
void Connect_WiFi();
void Firebase_Store(String PATH,String MSG);
String Firebase_getString(String PATH);
void test();
void key();
/*************************************************************************************************************/
void setup()
{
   Serial.begin(9600);
   pinMode(BP,OUTPUT);
   pinMode(SW,INPUT);
   digitalWrite(BP,HIGH);
   Serial.println("Driver drowsiness Detection ");
   lcd.init();
   lcd.setCursor(0,0);
   lcd.print("Driver drowsiness");
   lcd.setCursor(0, 1);
   lcd.print("   Detection");
   pt1=ot1=millis()/100;
   adc_pre = adc = analogRead(STR);
   delay(1000);
   digitalWrite(BP,LOW);
   Connect_WiFi();
}
/*************************************************************************************************************/
void loop()
{
   readSeatpreassure();
   steering();
```

```
    disp();
    Serial.println();
    test();
    key();
    if(SL == 1)
    {
      digitalWrite(BP,HIGH);
      delay(60);
      digitalWrite(BP,LOW);
      delay(50);
    }
}
/*****************************************************************************************************************/
void readSeatpreassure()
{
    s1=analogRead(SET);
    s2=analogRead(BCK);
    s3=analogRead(SHO);

    Serial.print(s1);
    Serial.print("   ");
    Serial.print(s2);
    Serial.print("   ");
    Serial.print(s3);
    Serial.print("   ");
    delay(100);

    if(s1>3450)
     {
      sa_1=1;
     }
     else
     {
      sa_1=0;
     }

     if(s2>3450)
     {
      sa_2=1;
     }
     else
     {
      sa_2=0;
     }
```

```
    /*if(s3>3450)
    {
     sa_4=1;
    }
    else
    {
     sa_4=0;
    }*/
    if(sa_1 == 1)
    {
      Serial.print("  Please sit on seat");
      seat_alert =1;
      alert=3;
    }
    else if(sa_2==1)
    {
       Serial.print("  Please sit Straight");
       seat_alert =2;
       alert=2;
    }
    /*else if(sa_4 == 1)
    {
       Serial.print("  Please keep head Straight");
       seat_alert =3;
       alert=1;
    }*/
    else
    {
     seat_alert =0;
    }

}
/*************************************************************************************************************/
void steering()
{
  int diff;
  pt = millis()/100;
  if(pt - ot > 1)
  {
      tc++;
      ot=pt;
      adc = analogRead(STR);
      if(adc >= adc_pre)
```

```
      {
        diff = adc - adc_pre;
        adc_pre = adc;
      }
      else
      {
        diff = adc_pre - adc;
        adc_pre = adc;
      }
      Serial.print("   ");
      Serial.print(diff);
      Serial.print("   ");

      if(diff > THRS)
      {
          Serial.print("Steering Alert !");
          alert=4;
          steering_alert =1;
          tc=0;
      }
  }
  if(tc >55)
  {
    tc =0;
    steering_alert =0;
    Serial.print("Steering safe");
  }

}
/*************************************************************************************************************/
void disp()
  {
    pt1=millis()/100;
    if(pt1-ot1>=10)
    {
      tc1++;
        Serial.print("Timer count \n");
        Serial.println(tc1);
        ot1=pt1;
        lcd.begin(16, 2);
        lcd.setCursor(14, 0);
        lcd.print(tc1);
        if(tc1 <= 3)
        {
```

```
if(seat_alert == 1)
{
  lcd.setCursor(0, 0);
  lcd.print("sit on seat");
  digitalWrite(BP,HIGH);
  delay(60);
  digitalWrite(BP,LOW);
}
else if(seat_alert == 2)
{
  lcd.setCursor(0, 0);
  lcd.print("Sit Straight");

  delay(60);
  digitalWrite(BP,LOW);
}
else if(seat_alert == 3)
{
  lcd.setCursor(0, 0);
  lcd.print("head Rest");

  digitalWrite(BP,HIGH);
  delay(60);
  digitalWrite(BP,LOW);
}
else
{
  lcd.setCursor(0, 0);
  lcd.print("Seat Safe");
}
if(steering_alert == 1)
{
  lcd.setCursor(0, 1);
  lcd.print("Steering ALERT !");

  digitalWrite(BP,HIGH);
  delay(60);
  digitalWrite(BP,LOW);
}
else if(steering_alert == 0)
{
  lcd.setCursor(0, 1);
  lcd.print("Steering safe");
}
```

```
}
else if(tc1 == 4)
{
  lcd.setCursor(0, 0);
   lcd.print("Alert Record");
   lcd.setCursor(0,1);
   if(alert>0)
   {
       if(SL == 1)
       {
          lcd.print("Sleep");
       }
       else if(alert == 3)
       {
          lcd.print("Sit on seat");
       }
       else if(alert == 4)
       {
          lcd.print("Steering ALERT !");
       }
       else if(alert == 2)
       {
          lcd.print("Sit Straight");
       }
       else if(alert == 1)
       {
          lcd.print("Head Rest");
       }
   }
    else
   {
      lcd.print("Safe");
   }


}
else if(tc1 <= 5)
{
  tc1=0;
  lcd.setCursor(0, 0);
  lcd.print("Geting Sleep info");
  msg = Firebase_getString("/KL05S6628/sleep");
  Serial.println(msg);
  delay(50);
```

```
        lcd.setCursor(0, 1);
        if(msg == "ALERT")
        {
          lcd.print("Driver Sleep");
          alert=5;
          SL=1;
          delay(500);
        }
        else
        {
          lcd.print("Driver Active");
          SL=0;
        }
      }
   }
 }
/*********************************************************************************************/
void Connect_WiFi()
   {
     int i=0;
        Serial.begin(9600);
        delay(100);
        WiFi.disconnect();
        delay(800);
        lcd.begin(16, 2);
        lcd.setCursor(0, 0);
        lcd.print("WiFi Connecting");
        lcd.setCursor(0, 1);
        Serial.println("Connecting to Wi-Fi");
        WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
        Serial.print("Connecting to Wi-Fi");
        delay(100);
        while (WiFi.status() != WL_CONNECTED)
        {
          i++;
          if(i>15)
          {
            lcd.begin(16, 2);
            lcd.setCursor(0, 0);
            lcd.print("WiFi Connecting");
            lcd.setCursor(0, 1);
          }
          digitalWrite(BP,HIGH);
          delay(90);
```

```
          digitalWrite(BP,LOW);
          Serial.print(".");
          lcd.print("*");
          delay(300);
        }
        Serial.println();
        Serial.print("Connected with IP: ");
        Serial.println(WiFi.localIP());
        Serial.println();
        Serial.printf("Firebase Client v%s\n\n", FIREBASE_CLIENT_VERSION);
        config.api_key = API_KEY;
        auth.user.email = USER_EMAIL;
        auth.user.password = USER_PASSWORD;
        config.database_url = DATABASE_URL;
        config.token_status_callback = tokenStatusCallback;
        #if defined(ESP8266)
          fbdo.setBSSLBufferSize(2048 /* Rx buffer size in bytes from 512 - 16384 */, 2048 /* Tx
buffer size in bytes from 512 - 16384 */);
        #endif
        Firebase.begin(&config, &auth);
        Firebase.reconnectWiFi(true);
        Firebase.setDoubleDigits(5);
        config.timeout.serverResponse = 10 * 1000;
        lcd.begin(16, 2);
        lcd.setCursor(0, 0);
        lcd.print("WiFi   {OK}");
    }
/*****************************************************************************/
    void Firebase_Store(String PATH,String MSG)
    {
        Serial.print("Uploading data \" ");
        Serial.print(MSG);
        Serial.print(" \"  to the location \" ");
        Serial.print(PATH);
        Serial.println(" \"");
        Firebase.RTDB.setString(&fbdo, PATH, MSG);
        delay(50);
    }
/*****************************************************************************/
    String Firebase_getString(String PATH)
    {
      String msg = (Firebase.RTDB.getString(&fbdo, PATH) ? fbdo.to<const char *>() :
fbdo.errorReason().c_str());
      delay(50);
```

```
    return msg;
  }
/*********************************************************************/
void test()
  {
   if(Serial.available())
      {
         char x;
         x=Serial.read();
         if(x=='a')
         {
           Firebase_Store("/KL05S6628/sleep","SAFE");
         }
         else if(x=='b')
         {
           int z;
           msg = Firebase_getString("/KL05S6628/sleep");
           Serial.println(msg);
         }
      }
  }
/*********************************************************************/
void key()
{
 if(digitalRead(SW) == 0)
 {
   digitalWrite(BP,HIGH);
   Firebase_Store("/KL05S6628/sleep","SAFE");
   digitalWrite(BP,LOW);
   SL=0;
   while(digitalRead(SW) == 0)
   {
     delay(50);
     SL=0;
   }
 }
}
/*****************************************************************************************************/
```

Appendix C:Queries

- Why was the ESP32 not used for implementing the eye detection module in this project? The ESP32, while suitable for handling sensor inputs and basic data processing, lacks the necessary computational resources to perform real-time eye detection tasks. Eye detection involves capturing and processing video frames, detecting facial landmarks, and calculating the Eye Aspect Ratio (EAR), which are computationally intensive operations. These are better handled by a laptop equipped with a built-in webcam and software tools like Python, OpenCV, and Dlib. Therefore, the eye detection module was implemented on a laptop to ensure accurate and real-time performance, while the ESP32 was used for monitoring seat pressure and steering movement.

- How were the threshold values of 2850 (for force sensors), 900 (for potentiometer), and 0.18 (for EAR) determined in this project?

  - **2850 (Force Sensors):** Multiple postures were tested. Readings ranged from 3000–3500 when seated properly and dropped to 2700–2800 when leaning or standing. A threshold of 2850 was set to detect absence or movement.
  - **900 (Potentiometer):** Normal steering gave values of 400–800, while sharp turns exceeded 900. Thus, 900 was chosen to detect sudden steering activity.
  - **0.18 (EAR):** EAR values for open eyes were 0.25–0.3 and 0.15–0.17 for closed eyes. After trials, 0.18 was finalized to detect drowsiness without false positives.

- Is it necessary to place a force sensor in the seat, since we already know the person is seated? No, placing a force sensor in the seat is not strictly necessary. While the original design included a seat sensor to monitor posture, feedback from our instructor highlighted that it did not add meaningful data. As a result, we removed the seat sensor and retained only two force sensors—on the back and shoulder—to detect posture changes like slouching or leaning forward, which are more relevant to drowsiness detection.

- Why was yawning detection not included in the project? Yawning detection was initially planned using ESP32, but real-time facial feature analysis for yawning is computationally demanding. The ESP32 lacks the necessary processing power and

resolution. As our EAR-based eye detection on a laptop was already effective and resource-efficient, we decided to drop yawning detection and focus on eye closure, seat posture, and steering behavior.

- Is there a unit for the threshold values used in the project? The threshold values used in our project are unitless:
  - **Force sensors and potentiometer:** These give raw ADC values (0–4095) when read by ESP32. The thresholds are scaled sensor outputs, not standard units.
  - **EAR (Eye Aspect Ratio):** A unitless ratio derived from pixel distances between landmarks around the eye.
- Since a laptop cannot be placed inside a car for eye detection, how can this system be made practical? To make the system practical for real-world use, the eye detection module can be implemented on devices like Raspberry Pi or NVIDIA Jetson Nano. These support Python, OpenCV, and Dlib, allowing the detection script to run independently with a USB or Pi camera. However, we used a laptop for this project due to cost and availability constraints, especially for student-level implementations.

Appendix D: Vision, Mission, Programme Outcomes and Course Outcomes

Vision, Mission, Programme Outcomes and Course Outcomes

Institute Vision

To evolve into a premier technological institution, moulding eminent professionals with creative minds, innovative ideas and sound practical skill, and to shape a future where technology works for the enrichment of mankind.

Institute Mission

To impart state-of-the-art knowledge to individuals in various technological disciplines and to inculcate in them a high degree of social consciousness and human values, thereby enabling them to face the challenges of life with courage and conviction.

Department Vision

To evolve into a center of excellence in electronics and communication engineering, moulding professionals having Inquisitive, Innovative and Creative minds with sound practical skills who can strive for the betterment of mankind.

Department Mission

To impart state-of-the-art knowledge to students in Electronics andCommunication Engineering and to inculcate in them a high degree of social consciousness and a sense of human values, thereby enabling them to face challenges with courage and conviction.

Programme Outcomes (PO)

Engineering Graduates will be able to:

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and Team work: Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.

10. Communication: Communicate effectively with the engineering community and with society at large. Be able to comprehend and write effective reports documentation. Make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Programme Outcomes (PO)
Engineering Graduates will be able to:

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and Team work: Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.

10. Communication: Communicate effectively with the engineering community and with society at large. Be able to comprehend and write effective reports documentation. Make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Programme Specific Outcomes (PSO)

PSO1	Demonstrate their skills in designing, implementing and testing analogue and digital electronic circuits, including microprocessor systems, for signal processing, communication, networking, VLSI and embedded systems applications.
PSO2	Apply their knowledge and skills to conduct experiments and develop applications using electronic design automation (EDA) tools.
PSO3	Demonstrate a sense of professional ethics, recognize the importance of continued learning, and be able to carry out their professional and entrepreneurial responsibilities in electronics engineering field giving due consideration to environment protection and sustainability.

Program Specific Outcomes (PSOs)

Course Outcomes (CO)

CO1	**Students will be able to practice acquired knowledge within the selected area of technology for project development.**
CO2	Students will be able to identify, discuss and justify the technical aspects and design aspects of the project with a systematic approach.
CO3	Students will be able to reproduce, improve and refine technical aspects for engineering projects.
CO4	Work as a team in development of technical projects.
CO5	Communicate and report effectively project related activities and findings.

Course Outcomes for Project Development

Project Outcomes

CO No.	**Project Outcome Statement**
P1	Was able to apply acquired knowledge on technical and design aspects to implement an Anti-Sleep Alarm System.
P2	Was able to use appropriate tools and techniques to develop an Anti-Sleep Alarm System.
P3	Was able to analyze and troubleshoot technical issues encountered during the development of the Anti-Sleep Alarm System.
P4	Was able to demonstrate effective communication and teamwork skills during the design and implementation of the Anti-Sleep Alarm System.
P5	Was able to evaluate the performance and reliability of the developed Anti-Sleep Alarm System.

Project Outcomes

Appendix E:CO–PO–PSO Mapping

Course Outcome – Program Outcome Mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3	3	3	2	3	3	2					2
CO2	3	3	3	2	3	3	2			3	2	2
CO3	3	3	3	2	3	3	2			3	2	2
CO4					2			3	3	3	2	2
CO5					2			3	3	3	2	2

Mapping of Course Outcomes (COs) with Program Outcomes (POs)

Course Outcome – Programme Specific Outcome Mapping

Programme-specific Outcomes (PSOs)			
	PSO1	**PSO2**	**PSO3**
CO1	3	3	2
CO2	3	3	2
CO3	3	3	2
CO4	2	2	3
CO5	2	2	3

Mapping of Course Outcomes (COs) with Programme Specific Outcomes (PSOs)

Project Outcome – Program Outcome Mapping

	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**
P1	3	3	3	2	3	3	2					2
P2	2	2	2	2	2	2	2			3	2	2
P3	2	2	2	2	2	2	2			3	2	2
P4				2				3	3	3	2	2
P5				2				3	3	3	2	2

Mapping of Project Outcomes (P1–P5) with Programme Outcomes (PO1–PO12)

Justification Table: PO Mapping with Project Outcomes

Project Outcome	PO	Level	Justification
P1	PO1	3	Applies engineering knowledge to develop Anti-Sleep Alarm components.
P1	PO2	3	Analyzes drowsiness-related issues and develops logic-based solutions.
P1	PO3	3	Designs integrated system using sensors, microcontrollers, and alarms.
P1	PO4	2	Conducts testing and compares alternate designs.
P1	PO5	3	Uses Arduino IDE, ESP32, and modeling tools in development.
P1	PO6	2	Considers safe assembly and voltage handling.
P1	PO7	2	Selects energy-efficient components and sustainable practices.
P1	PO12	2	Demonstrates self-learning in embedded systems and protocols.
P2	PO1	2	Applies core engineering to select appropriate modules.
P2	PO2	2	Analyzes trade-offs in implementation techniques.
P2	PO3	2	Customizes design logic for the intended functionality.
P2	PO4	2	Uses performance tests to improve the system.
P2	PO5	3	Uses real-time tools for debugging and testing.
P2	PO10	3	Clearly communicates technical decisions and process.
P2	PO12	2	Researches and learns new component behaviors.
P3	PO1	2	Uses previous knowledge to improve the current prototype.
P3	PO2	2	Diagnoses system bottlenecks and errors.
P3	PO3	2	Refines integration for higher accuracy and reliability.
P3	PO4	2	Tests system rigorously to identify design flaws.
P3	PO5	2	Updates firmware and runs simulations.
P3	PO10	3	Effectively presents logs and version upgrades.
P3	PO11	2	Plans project stages and tracks iterations.
P3	PO12	2	Learns updated technologies for optimization.
P4	PO5	2	Uses platforms like GitHub and shared tools for development.
P4	PO8	3	Follows ethical guidelines in documentation and design reviews.
P4	PO9	3	Collaborates with multi-role team members effectively.
P4	PO10	3	Maintains consistent communication via meetings and reports.

Project Outcome	PO	Level	Justification
P4	PO11	2	Allocates resources and sets task deadlines.
P4	PO12	2	Adapts and grows through peer interactions and feedback.
P5	PO5	2	Uses modern documentation and analysis tools.
P5	PO8	3	Upholds ethical reporting practices.
P5	PO9	3	Distributes documentation responsibilities across the team.
P5	PO10	3	Demonstrates technical writing and presentation skills.
P5	PO11	2	Manages tasks and final compilation efficiently.
P5	PO12	2	Takes initiative in preparing final reports and evaluations.

Justification Table for Mapping of Project Outcomes (P1–P5) with Programme Outcomes (PO1–PO12)

Project Outcome – PSO Mapping

Programme-specific Outcomes (PSOs)			
	PSO1	**PSO2**	**PSO3**
P1	3	2	2
P2	3	3	1
P3	3	2	2
P4	2	2	3
P5	1	2	3

Mapping of Project Outcomes (P1–P5) with Programme Specific Outcomes (PSO1–PSO3)

Justification Table: PSO Mapping with Project Outcomes

Project Outcome	**PSO**	**Justification**
P1	PSO1	Demonstrates core knowledge of electronics and embedded systems for designing the Anti-Sleep Alarm System using sensors and microcontrollers.
P1	PSO2	Applies electronic design tools for simulation, testing, and hardware realization of circuits involved.
P1	PSO3	Reflects awareness of sustainable design and professional responsibilities in system development.
P2	PSO1	Implements relevant tools and embedded systems to achieve desired real-time response.
P2	PSO2	Uses EDA tools and hardware platforms to prototype and refine the alarm system design.
P2	PSO3	Applies professional responsibility in component selection and energy-efficient design.
P3	PSO1	Identifies hardware issues and updates control logic for optimized system performance.
P3	PSO2	Uses tools and methods to analyze and troubleshoot technical faults.

Project Outcome	**PSO**	**Justification**
P3	PSO3	Emphasizes continuous improvement and ethical problem-solving.
P4	PSO1	Collaborates across teams to integrate circuit design with system implementation.
P4	PSO2	Uses collaborative platforms and test setups to verify subsystem functionality.
P4	PSO3	Demonstrates teamwork ethics, responsibility, and commitment during system testing.
P5	PSO1	Tests the final system to ensure electronic performance parameters are met.
P5	PSO2	Analyzes results using suitable tools and generates structured reports.
P5	PSO3	Promotes lifelong learning and responsibility through post-project documentation and reflection.

Justification Table: PSO Mapping with Project Outcomes