designs

MDPI

Article

Design of a System for Driver Drowsiness Detection and Seat Belt Monitoring Using Raspberry Pi 4 and Arduino Nano

Anthony Alvarez Oviedo [1,†], Jhojan Felipe Mamani Villanueva [1,†], German Alberto Echaiz Espinoza [2,*,†], Juan Moises Mauricio Villanueva [3,†], Andrés Ortiz Salazar [4,†] and Elmer Rolando Llanos Villarreal [5,†]

[1] Professional School of Electronics Engineering, Universidad Nacional de San Agustin de Arequipa, Arequipa 04002, Peru; aalvarezo@unsa.edu.pe (A.A.O.); jmamanivill@unsa.edu.pe (J.F.M.V.)
[2] Department of Electronics Engineering, Universidad Nacional de San Agustin de Arequipa, Arequipa 04002, Peru
[3] Department of Electrical Engineering, Center for Alternative and Renewable Energies—CEAR, Federal University of Paraíba, João Pessoa 58051-900, PB, Brazil; jmauricio@cear.ufpb.br
[4] Department of Computer Engineering and Automation, Federal University of Rio Grande do Norte (DCA-UFRN), Natal 59072-970, RN, Brazil; andres@dca.ufrn.br
[5] Department of Natural Sciences, Mathematics, and Statistics, Federal Rural University of Semi-Arid (DCME-UFERSA), Mossoró 59625-900, RN, Brazil; elmerllanos@ufersa.edu.br
* Correspondence: gechaiz@unsa.edu.pe
† These authors contributed equally to this work.

Abstract: This research explores the design of a system for monitoring driver drowsiness and supervising seat belt usage in interprovincial buses. In Peru, road accidents involving long-distance bus transportation amounted to 5449 in 2022, and the human factor plays a significant role. It is essential to understand how the use of non-invasive sensors for monitoring and supervising passengers and drivers can enhance safety in interprovincial transportation. The objective of this research is to develop a system using a Raspberry Pi 4 and Arduino Nano that allows for the storage of monitoring data. To achieve this, a conventional camera and MediaPipe were used for driver drowsiness detection, while passenger supervision was carried out using a combination of commercially available sensors as well as custom-built sensors. RS485 communication was utilized to store data related to both the driver and passengers. The simulations conducted demonstrate a high level of reliability in detecting driver drowsiness under specific conditions and the correct operation of the sensors for passenger supervision. Therefore, the proposed system is feasible and can be implemented for real-world testing. The implications of this research suggest that the system's cost is not a barrier to its implementation, thus contributing to improved safety in interprovincial transportation.

Keywords: security; raspberry; computer vision; MediaPipe; Arduino Nano; RS485; data collection

Academic Editor: Yuping He

Received: 8 November 2024
Revised: 28 December 2024
Accepted: 6 January 2025
Published: 13 January 2025

Citation: Alvarez Oviedo, A.; Mamani Villanueva, J.F.; Echaiz Espinoza, G.A.; Villanueva, J.M.M.; Salazar, A.O.; Villarreal, E.R.L. Design of a System for Driver Drowsiness Detection and Seat Belt Monitoring Using Raspberry Pi 4 and Arduino Nano. *Designs* **2025**, *9*, 11. https://doi.org/10.3390/designs9010011

1. Introduction

Road traffic crashes are a major global health concern, causing approximately 1.19 million deaths annually. Key risk factors contributing to these fatalities include non-use of safety restraints (seat belts and child restraints) and driver impairment, particularly fatigue and drowsiness. Seat belt use alone is estimated to reduce the risk of death among vehicle occupants by up to 50% [1]. Thus, in [2], the authors analyze the use of seat belts in buses through observations on 328 buses in 10 cities in Sweden, including factors such as unbuckling the seat belt, sleeping discomfort, and other factors affecting its use. In

Peru (the National Transport Administration Regulation [3]), national regulations mandate seat belt use for all passengers over 4 years of age in all vehicles. However, despite these regulations, compliance remains inconsistent, particularly in interprovincial buses, which are integral to long-distance travel but often lack effective monitoring systems to ensure compliance and maximize safety benefits for passenger safety. Concurrently, driver fatigue and drowsiness present a significant threat to road safety. According to the 2022 Road Accident Statistical Report by SUTRAN [4], Peru recorded 5449 road accidents, of which 47% were caused by collisions and 45% by reckless behavior, indicating that human factors, including driver behavior, are a primary cause of accidents, with fatigue and drowsiness being significant contributing factors. The ONSV report [5] identifies human factors—specifically fatigue and drowsiness—as the primary cause. These conditions impair cognitive function, reaction time, and decision making, dramatically increasing the risk of collisions. Traditional methods for detecting driver drowsiness, such as electroencephalograms (EEGs) and electrocardiograms (ECGs), are impractical for real-world application due to their intrusive nature and complexity. Therefore, addressing both seat belt compliance and driver drowsiness is essential for improving road safety. While these issues are often studied separately, their combined impact on accident severity highlights the need for integrated monitoring solutions. Recent advances in artificial intelligence (AI), particularly Deep Learning (DL) and machine learning (ML) coupled with computer vision (CV), offer promising avenues for developing non-invasive and cost-effective monitoring systems. However, many existing solutions require high-performance hardware, limiting their widespread adoption. This paper proposes a novel integrated system for monitoring both driver drowsiness and passenger seat belt usage in long-distance bus services using non-invasive, cost-effective methods. By combining these two critical safety aspects into a single system, we aim to provide a more holistic approach to road safety. Our system utilizes a Raspberry Pi 4 and multiple Arduino Nano microcontrollers for data acquisition and processing. Driver drowsiness is monitored using a conventional camera and the MediaPipe library, while passenger seat belt usage is detected using readily available commercial and custom-designed sensors. This architecture prioritizes portability, robustness, user-friendliness, and low implementation cost. The remainder of this paper is organized as follows:

- Section 2 reviews related work and identifies gaps in existing solutions.
- Section 3 describes the proposed methodology, including materials and methods.
- Section 4 presents results and discussions on system performance.
- Section 5 concludes the paper and summarizes the achievement of the research objectives with recommendations for future work.

2. Literature Review

The current literature and market reveal a growing interest in developing advanced in-vehicle monitoring systems, with a particular focus on seat belt usage detection, passenger identification, and vital signs monitoring. The approaches and technologies employed in these systems vary, but they all aim to enhance safety and detection accuracy. Regarding seat occupancy monitoring and seat belt usage, researchers have developed innovative sensors, [6]. Introduces a flexible resistive sensor based on interwoven conductive thread, addressing the issue of false positives found in FSR sensors by enabling differentiation between people and objects. In [7] proposes a wireless inductive sensor that detects seat occupancy by measuring changes in inductance caused by the applied weight, achieving differentiated sensitivity for various weight ranges. In another approach, [8] presents an intelligent warning system that allows a bus driver to verify if passengers have fastened their seat belts using a mobile application connected via WiFi to the seat belt buckles.

Another significant contribution is the EL20 force sensor [9], which accurately measures the force exerted by passengers on the seat belt using strain gauges configured in a low-noise Wheatstone bridge. The driver drowsiness detection field has been approached with a focus on computer vision and artificial intelligence. Internationally, commercial systems like Eye Alert [10], Eye Tracking [11], and Optalert [12] stand out, while in Peru, solutions such as VidFleet AI [13] and DMS Stonkam [14] have gained prominence. These systems detect drowsiness, distractions, and the improper use of mobile devices using infrared cameras to avoid illumination-related issues. Another notable system is Hexagon OP [15], which focuses on continuous fatigue monitoring for heavy vehicle operators through a service-based approach. Academic research has proposed various approaches for drowsiness detection using artificial intelligence and machine learning. In [16] develops a system based on OpenCV and a multi-layer perceptron (MLP) neural network, utilizing Facial Landmarks to detect eye and mouth openness and head tilt, achieving an 86.62% accuracy in drowsiness detection. In another approach, [17] employs a combination of an infrared (IR) camera for nighttime use and a Kinect camera for daytime use, complemented by facial recognition systems (FaceSDK) and data storage in MySQL databases. An additional relevant technique is presented in [18], where the Eye Aspect Ratio (EAR) is calculated using Facial Landmarks and a pre-trained Support Vector Machine (SVM) classifier to detect blinks, thereby improving detection speed. Moreover, [19] develops an emotion detection system using a Raspberry Pi 4, a V2.1 camera module, and machine learning algorithms, achieving accuracy rates above 70% for five key emotions, although with lower efficacy for anger and contempt. Then, in [20], the authors present a driver drowsiness detection system that integrates IoT with deep neural networks such as LSTM, VGG16, InceptionV3, and DenseNet. To process this information in real time, a Jetson Nano board is utilized, a specialized device that, while increasing the initial system cost, ensures optimal performance. Leveraging transfer learning and the advantages of the proposed neural networks, the accuracy of the models is enhanced by combining various factors, including eye state, mouth state, and head pose. The proposed system accurately detects drowsiness even in cases involving the use of glasses and masks. Additionally, the level of drowsiness is determined to provide early warnings, enabling drivers to prevent unintended accidents. In the case of [21], this study proposes a dual-camera system, both positioned in front of the driver's face, employing a Long Short-Term Memory (LSTM) network to detect drowsiness. The experiment was conducted at the Somnolence Laboratory of Alcohol Countermeasure Systems Corp. (ACS) in Toronto, Canada. Electroencephalography (EEG) was utilized to extract drowsiness state signals, serving as the ground truth for validating the system. The cameras focused on the driver's eyes, capturing visual data that correlate with drowsiness states determined by EEG. Using the Adam optimizer for model training, the experimental results demonstrated that the R-LSTM network achieved an accuracy of 87%, whereas the C-LSTM network reached a superior performance of 97.8%. Both LSTM architectures were directly trained using eye-state data, highlighting their potential for reliable drowsiness detection in controlled laboratory conditions. Then, in [22], the authors propose the integration of vehicular data (steering and lane-keeping) and physiological data (heart rate) in addition to the commonly used behavioral data (eye/blink measures and facial expressions) in driver monitoring systems to enhance drowsiness detection. For this study, behavioral data from a commercial DMS, vehicular data recorded in the large-motion driving simulator of the National Advanced Driving Simulator, and physiological data from a smart wristband were utilized. The results demonstrated that the inclusion of physiological data significantly improved the accuracy of the models, allowing for the earlier detection of drowsiness. However, the authors acknowledge that collecting physiological data in a real-world driving environment presents technical challenges due to variable conditions and constant motion. Finally, [23] proposes a solution based on a Raspberry Pi 3 with an 8MP camera that employs the Haar Cascade and Dlib algorithms for facial and eye detection. This system identifies the Eye Aspect Ratio and the state of the eyes (open or closed) at various testing distances.

The identified innovation or research gap in the literature is the development of a low-cost, tamper-proof seat belt system. This innovation aims to overcome the limitations of current systems, which often suffer from false positives or errors in passenger identification. Additionally, in the field of drowsiness detection, a significant reduction in system costs has been achieved while maintaining high reliability in detection. These improvements contribute to the development of more accessible and effective solutions for road safety. In general, the review reveals significant progress in seat occupancy detection, seat belt usage detection, and drowsiness detection systems. However, challenges remain, particularly in terms of detection accuracy under various environmental conditions (e.g., lighting changes and camera angles) and the ability to distinguish between people and objects. These gaps highlight the need to advance toward more precise, efficient, and accessible technologies to improve road safety and enhance user experience.

3. Methodology

Currently, the proposed works are growing and providing effective solutions for driver drowsiness detection. However, all of them have their drawbacks or are solely focused on driver monitoring. Many of these drowsiness detection techniques are affected by external conditions, such as the use of glasses, changes in lighting, camera type, high computational cost, low accuracy, and slow detection speed. In the case of passenger monitoring in buses, only the buckle sensor is used. This section proposes a system for monitoring both the driver and passengers, aiming for rapid drowsiness detection and ensuring the proper use of the passenger seat belt while preventing system tampering. The system incorporates RS-485 communication between the Raspberry Pi 4 (master) and Arduino Nano (slaves). The Raspberry Pi 4 serves as the central processing unit, displaying the status of the seats on a dedicated platform, monitoring driver drowsiness, and storing data for the entire system.

Below is a point-by-point explanation of how each part of the project was implemented.

3.1. Data Collection and Passenger Monitoring

Before beginning monitoring, certain aspects are extracted from the literature, such as sensor placement, the sensing method, and seat belt usage detection. Figure 1 shows the estimated position of the sensors on a bus seat. The seat or occupancy sensor [24] is located at the base of the seat.

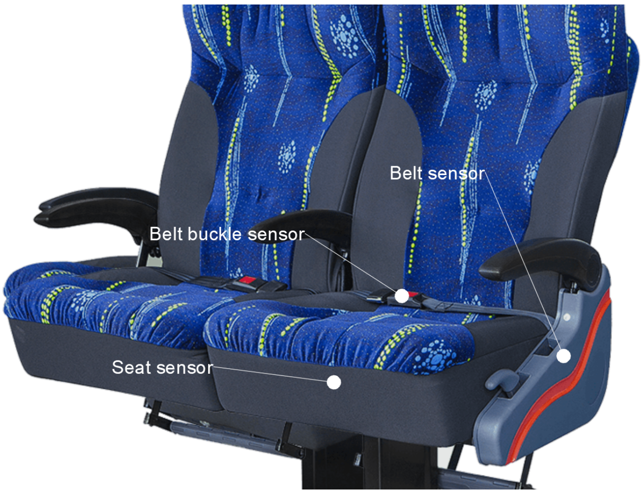


Figure 1. Sensor positioning.

The standard seat sensor consists of an arrangement of electrical resistors that help determine seat occupancy. Typically, this sensor's resistance values range from 0 to 800 ohms without faults or interruptions. A specific range of resistive values determines if a person is present on the seat. The buckle sensor [25] is already integrated into the seat belt and can be easily located. This sensor is magnetic and is found in all vehicles. Lastly, the seat belt sensor, designed specifically for this project, consists of a conductive thread seam located at the beginning of the seat belt reel (Figure 2). This conductive thread seam closes a normally open (NO) contact mounted on the reel's structure (Figure 3). The mentioned seam is located at a distance "X" from the seat belt's buckle, which closes the NO contact, indicating an attempt to bypass seat belt usage (Figure 4). If the seat belt is properly used, the seam will be at a distance greater than "X", showing it is not being tampered with. On the other hand, if someone attempts to bypass the seat belt, the NO contact will close due to the conductive thread.

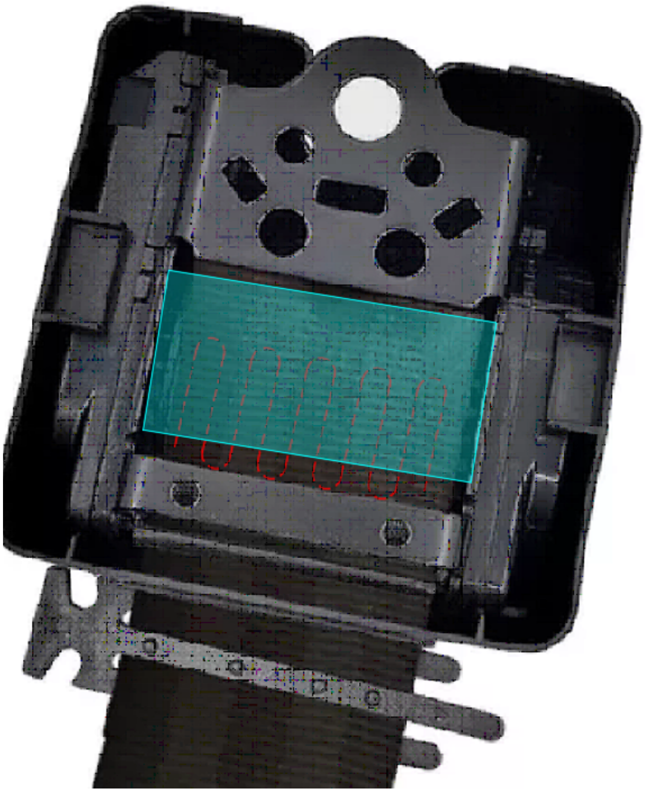


Figure 2. Conductive thread on the seat belt.

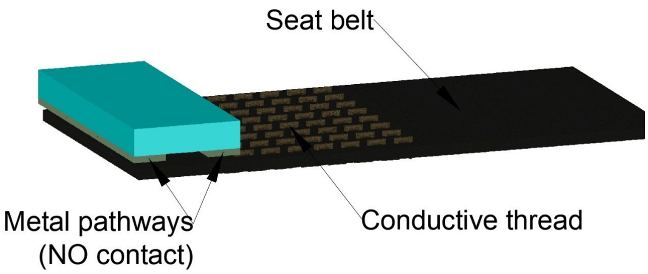


Figure 3. Three-dimensional view of the proposed sensor.

All sensors mounted on the seat, as described earlier, are connected and send their respective signals to an Arduino Nano based on the ATmega328 [26]. The signals are processed and stored for transmission to the Raspberry Pi 4 upon request. Figure 5 shows a flowchart illustrating how the signals from different sensors are registered and under what conditions the actuator, in this case, an LED, is activated.

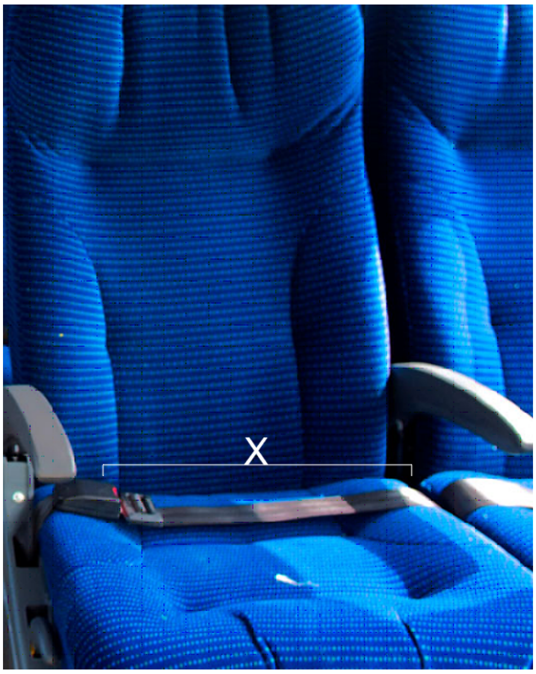


Figure 4. Distance "X" used to determine if the seat belt is being bypassed.

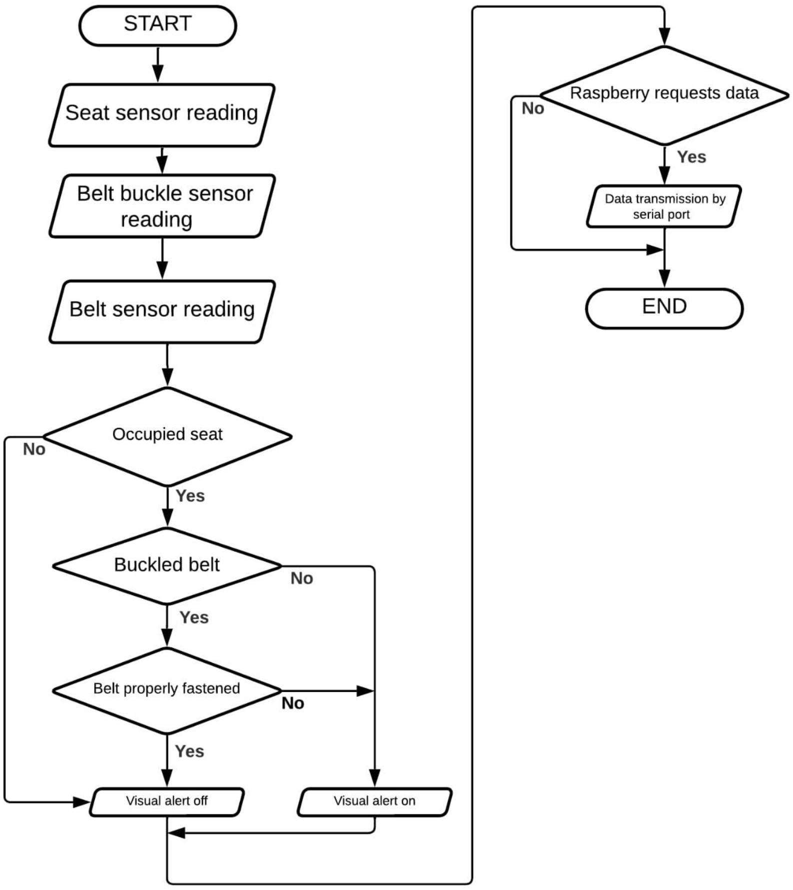


Figure 5. Flowchart executed by the Arduino. The START–END cycle repeats while the Arduino is powered.

The flowchart in Figure 5 shows the process for a single seat. Given the number of input signals that the Arduino Nano can receive and its limited input capacity, a multiplexer (CD74HC4067) is used. This multiplexer expands the Arduino's analog and digital input/output capacity [27], enabling the monitoring of up to 16 seats per Arduino. Each seat has 3 sensors, making a total of 48 input signals. The Arduino Nano receives two digital signals (the buckle sensor and seat sensor) and one analog signal (the seat sensor). Since the seat sensor provides a resistive value, it must be converted into a binary value.

Table 1 shows the binary values of the sensors and the result of their analysis. In certain cases, an Arduino output pin will activate an LED. The signal is sent through a demultiplexer, which activates the LED as a visual alert for the corresponding seat.

Figure 6 shows how the devices are interconnected. The signals sent by the sensors are divided by sensor type. Each multiplexer processes a specific sensor type, and these signals are captured by the Arduino Nano. The output uses a demultiplexer to send visual alerts to each seat. The Arduino Nano's programming was performed using the open-source Arduino IDE software. The "Mux.h" library [28] was used for managing the 74HC4067 multiplexer channels. The library creates a class called "Mux," which facilitates the declaration of inputs and outputs, as well as control over the pin selector for multiplexing or demultiplexing.

Table 1. Sensor states in the seat.

Sensor Seat(i)	Sensor Brace(i)	Sensor Belt(i)	Data LED(i)	Status
0	0	0	0	Unoccupied seat
0	0	1	0	Unoccupied seat
0	1	0	0	Unoccupied seat
0	1	1	0	Unoccupied seat
1	0	0	1	Passenger without seat belt
1	0	1	1	Passenger without seat belt
1	1	0	0	Passenger with belt correctly positioned
1	1	1	1	Passenger without seat belt

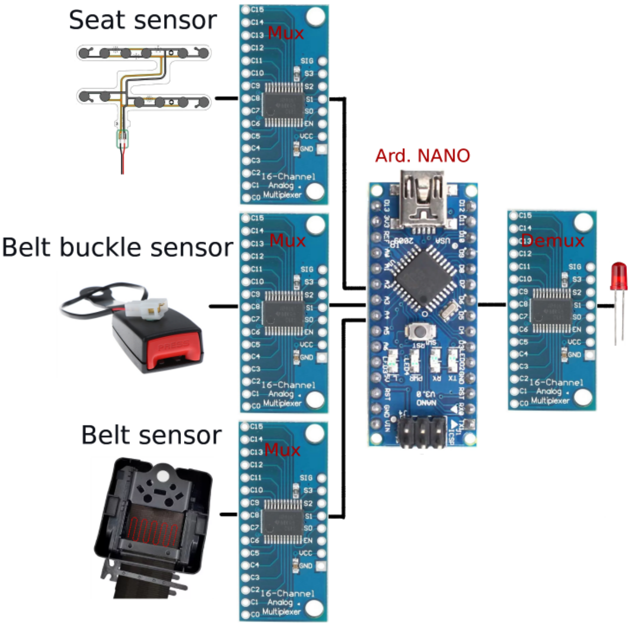


Figure 6. One-line diagram for integrating passenger monitoring devices to Arduino Nano.

3.2. Data Collection and Driver Monitoring

To develop the monitoring system, the following key points were considered based on the literature: camera position, camera distance, eye blinking, yawning, and head position. For camera position and distance, the established values range from 40 cm to 60 cm from the face, with the camera as aligned as possible with the driver's face. The system is built on the Raspberry Pi 4 platform [29]. Therefore, the code must be highly efficient, avoid the need for a graphics processor, and have a low computational cost. For this purpose, Python is used, along with the Visual Studio Code open-source IDE.

The flowchart in Figure 7 shows how the system detects the driver's face. Images are captured via a USB 2.0 camera with AVI FHD@30fps video format [30]. The face is detected using MediaPipe's FaceMesh, which identifies 486 Facial Landmarks for enhanced tracking shown in Figure 8. The system processes each frame to detect drowsiness indicators in the eyes, mouth, and head position.

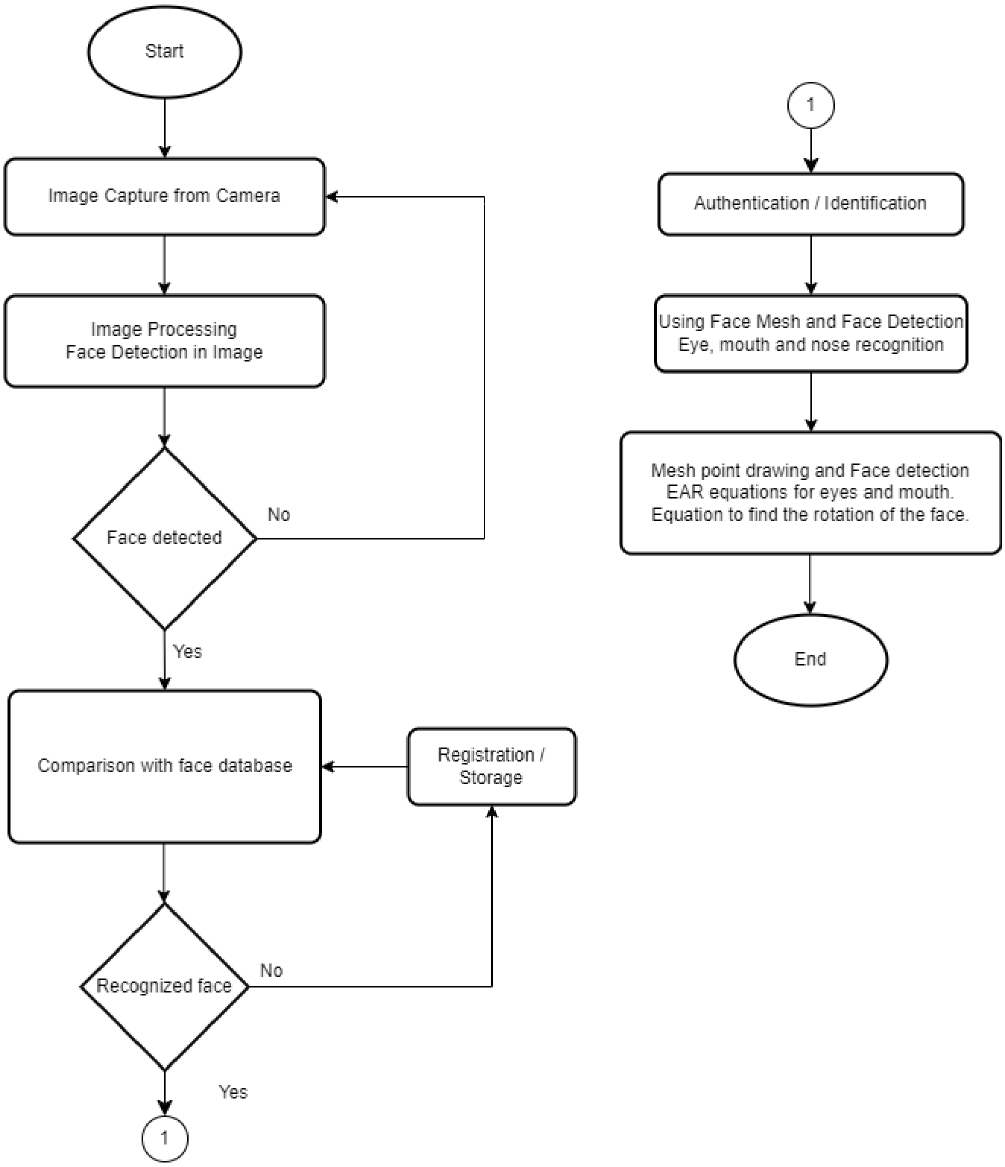

Start
Image Capture from Camera
Image Processing
Face Detection in Image
Face detected
No
Yes
Comparison with face database
Registration / Storage
Recognized face
No
Yes
1
1
Authentication / Identification
Using Face Mesh and Face Detection
Eye, mouth and nose recognition
Mesh point drawing and Face detection
EAR equations for eyes and mouth.
Equation to find the rotation of the face.
End

Figure 7. Driver data collection and monitoring operation flowchart.

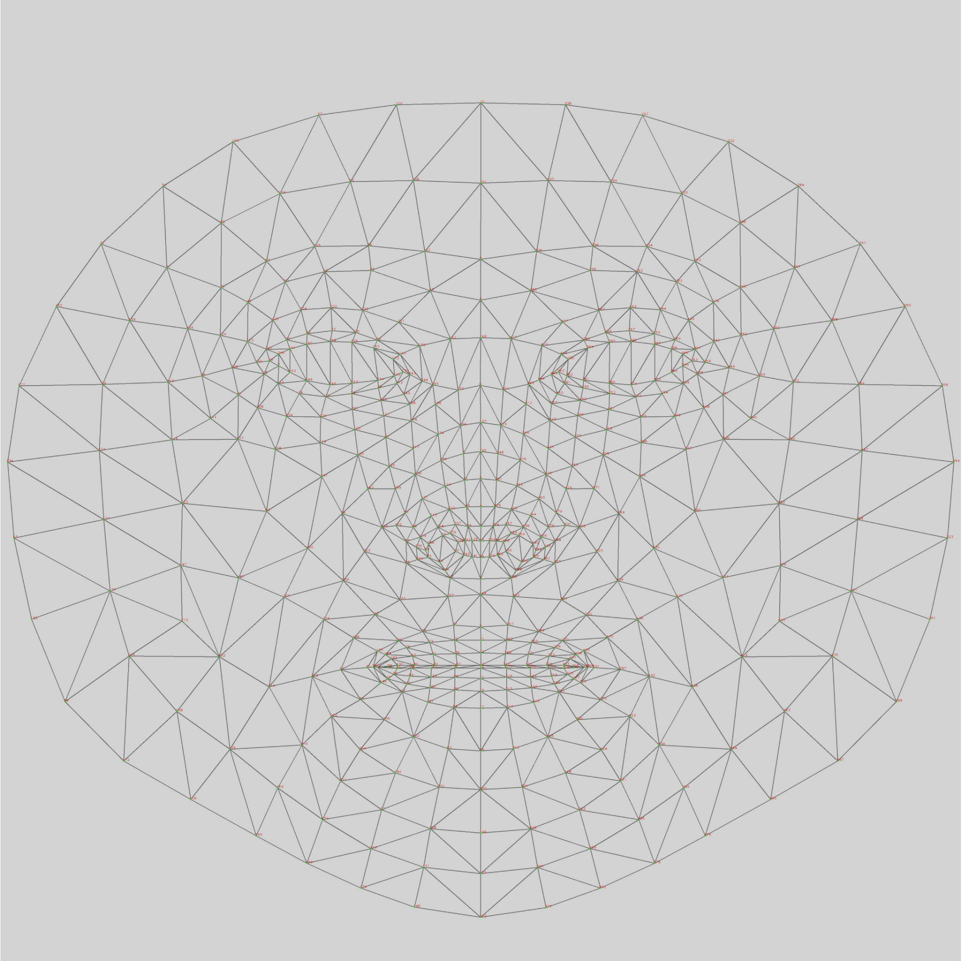


Figure 8. MediaPipe FaceMesh by stack overflow.

3.2.1. Eye Monitoring

Facial Landmarks from FaceMesh are used to identify eye position. Considering the camera's 30 fps frame rate and the duration of a blink (300–400 ms), a threshold of 13 frames (433 ms) is set for drowsiness detection. An LSTM neural network with PyTorch dynamically adjusts the *EAR* (Eye Aspect Ratio) threshold every 5 seconds to account for user-specific variations. If a long blink (over 13 frames) is detected, the system records the duration, date, and time of the incident.

Figure 9 explains how to obtain the *EAR* between the values of height and width, taking as a reference the equation given in [9] (Real-Time Eye Blink Detection using Facial Landmarks). Considering Equation (1), we have the distance A (*DA*), distance B (*DB*), and distance C (*DC*), and therefore, it can be concluded that if the eye is closed or close to being closed, the value that indicates us is close to zero, and it is true that there is variation depending on the user; hence, during the application of the neural network, the average is calculated to find the *EAR* (Eye Aspect Ratio) using the value of both eyes and thus achieve the detection of drowsiness. An alarm is activated at the moment of reaching 13 frames, and if the blink is long enough, a record is created with the duration, date, and time of the incident:

$$EAR = \frac{|[DB] + [DC]|}{2|[DA]|} \tag{1}$$

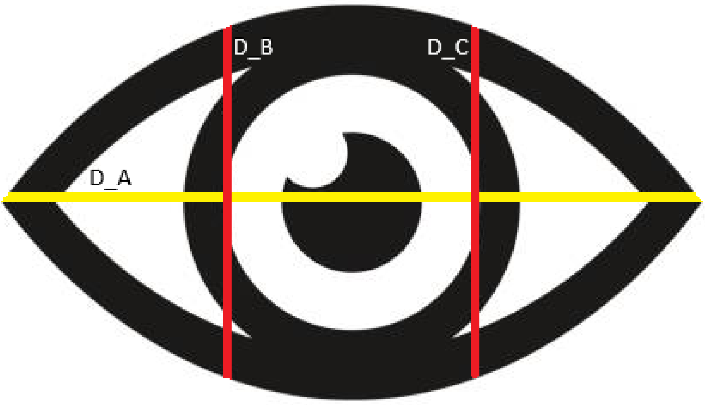


Figure 9. EAR between width and height.

3.2.2. Mouth Monitoring

MediaPipe's Facial Landmarks are also used to monitor mouth position. Points P1, P2, P3, and P4 (lip corners) are tracked, and the angle between these points is calculated to detect yawning. Yawning is identified if the mouth remains open for 5 seconds [31].

Figure 10 shows the mesh of points that MediaPipe uses for monitoring.

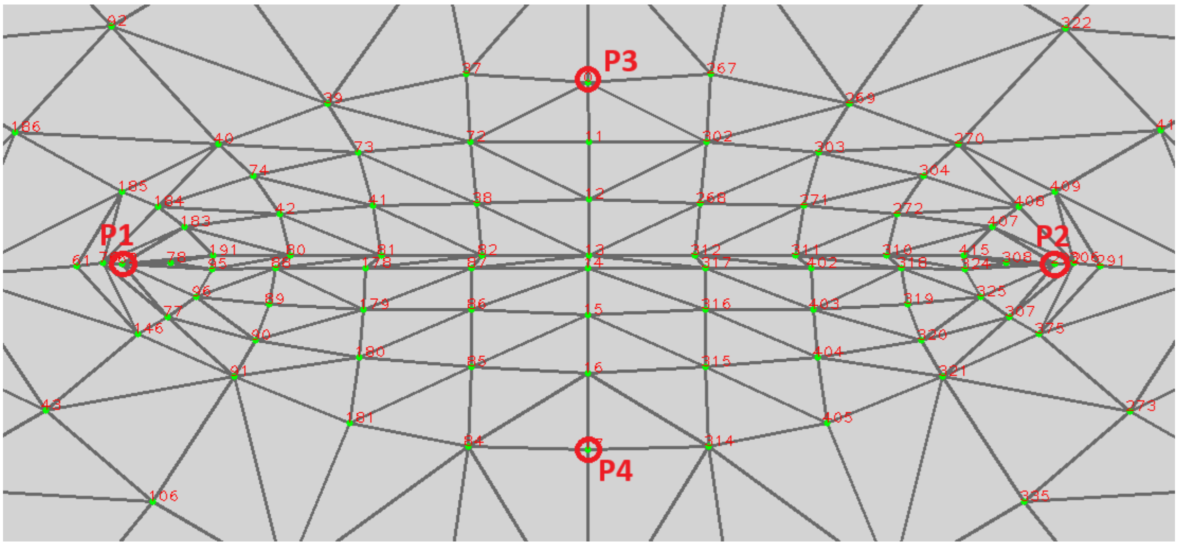


Figure 10. Points taken in the MediaPipe mesh. Source: stack overflow.

3.2.3. Head Monitoring

Head position is monitored using two FaceMesh points: one on the nose tip and another on the cheek. The angle between these points is calculated to detect head nodding, which indicates drowsiness [32]. Real-time alerts are triggered, and data on duration, date, and time are recorded. An RTC module [33] is used to store the date and time in case the Raspberry Pi lacks internet access. An LCD screen [34] displays the driver status and bus seat conditions. A Kalman filter is applied to avoid false positives due to lighting changes. The value of the arctangent function formed by the 2 points is taken into account, and with this, we achieve what is observed in Figure 11, where the values above the line are positive and those below are negative, so that by placing the camera in the correct position, it is possible to monitor microsleeps or some nodding that occurs at the time of drowsiness and tiredness, which will alert the person in real time based on the number of frames and saving the data of the nodding with duration time, date, and time. Also, when a certain time elapses, it will issue an alert.

It is necessary that the monitoring has a time and date; by default, the Raspberry Pi 4 needs an internet connection for updating, so an RTC module is placed, and that way the battery stores the time and date. To visualize the monitoring of the driver where not only the face to be monitored but also a drowsiness and incidents counter is observed, an LCD screen is used; likewise, this screen is used to display the status of the bus seats, as shown in Figure 12, which also shows the data memory where the system is stored. It is also worth mentioning that a Kalman filter is used in the driver monitoring to avoid false positives due to light changes.

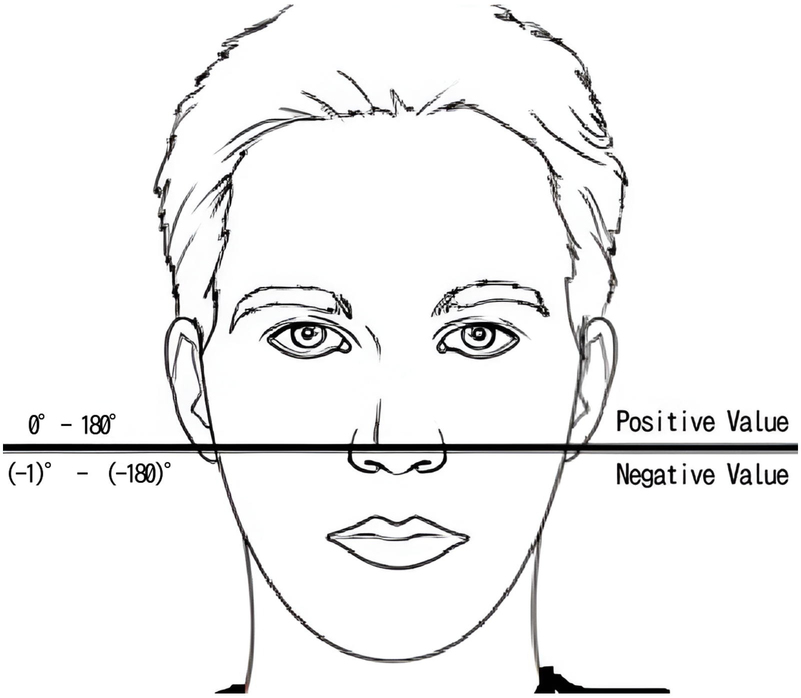


Figure 11. Head orientation.

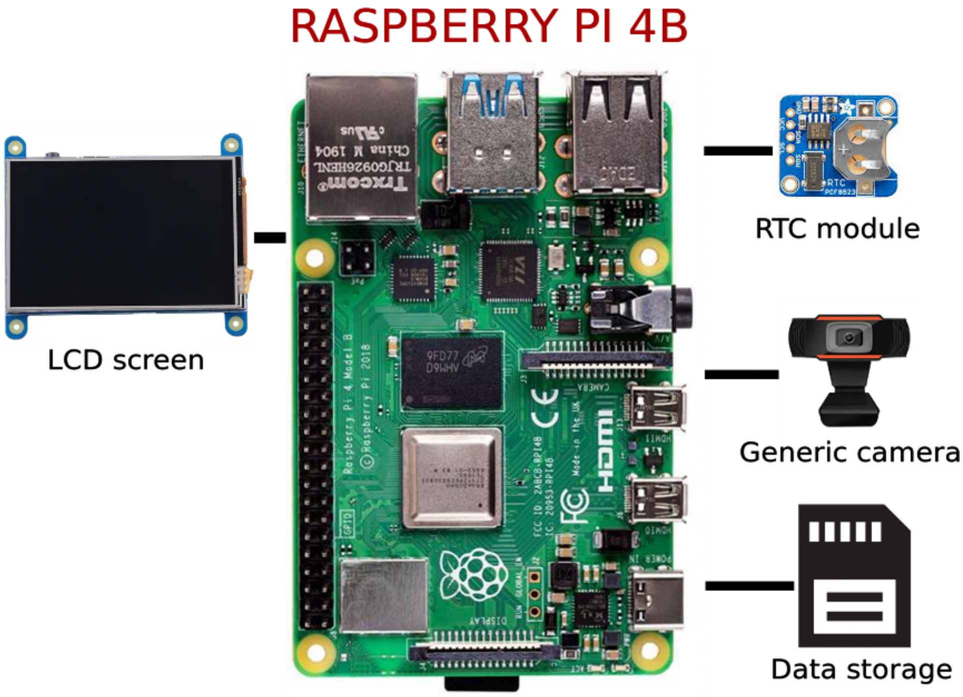


Figure 12. One-line diagram of Raspberry Pi 4B and devices to be used.

3.3. *System Integration*

The system's operation relies on the Raspberry Pi 4, which serves as the master and communicates with the Arduino Nano slaves via an RS-485 interface. This communication standard supports two- or three-wire connections with 22 or 24 AWG cables over a maximum distance of 1200 m [35].

Figure 13 shows the interconnection between the Raspberry Pi 4 and Arduino Nanosystems. The RS-485 communication is half-duplex. The Arduino Nano uses the MAX485 module [36], while the Raspberry Pi 4 uses the RS485 Can Hat module [37]. Data are sent every 5 min or when the update button is pressed. The LCD screen shows the status of the bus seats. Data from seat and driver monitoring are stored separately in text files.

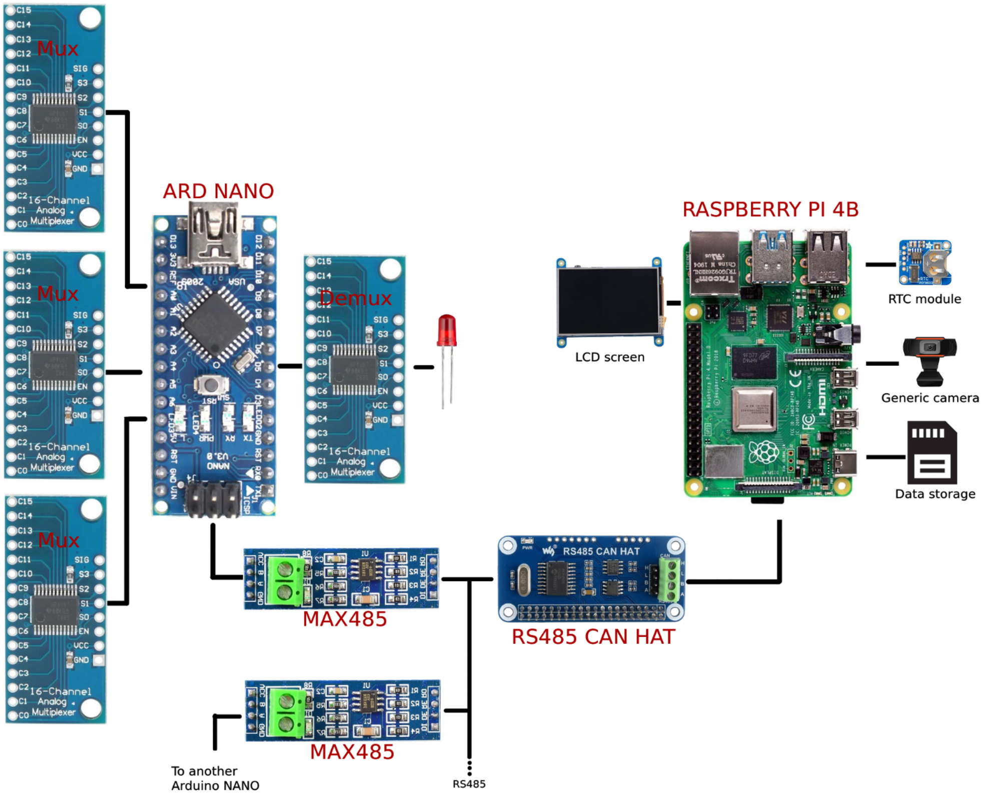


Figure 13. One-line diagram for communication between Arduino Nano and Raspberry Pi 4B.

4. Results and Discussion

According to the development process, tests were carried out to verify the system. For this reason, simulations were conducted in parts: a simulation of the passenger-oriented system using the Proteus software, and tests of the driver-oriented system with the Raspberry Pi 4. A communication test was also performed between the Arduino and Raspberry Pi to verify the correct storage of the collected data in a text file.

4.1. *Simulation*

The first part focuses on the simulation of passenger monitoring, conducted using Proteus V8.13, an electronic design automation software.

Figure 14 shows the seat sensor represented by a potentiometer (SE-01), which is set to 0 Ω, indicating that no person is occupying the seat. As a result, the Arduino sends a logical OFF signal to the corresponding seat LED through the "OUTPUT-LED" multiplexer.

Figure 15 illustrates the scenario of a person occupying the seat but not fastening the seat belt (value 0 on the "SEAT BELT BUCKLE SENSOR" multiplexer). Consequently, the

Arduino sends a logical ON signal to the corresponding seat LED through the “OUTPUT-LED” multiplexer. Finally, the simulation of a passenger occupying the seat and properly wearing the seat belt is shown in Figure 16.

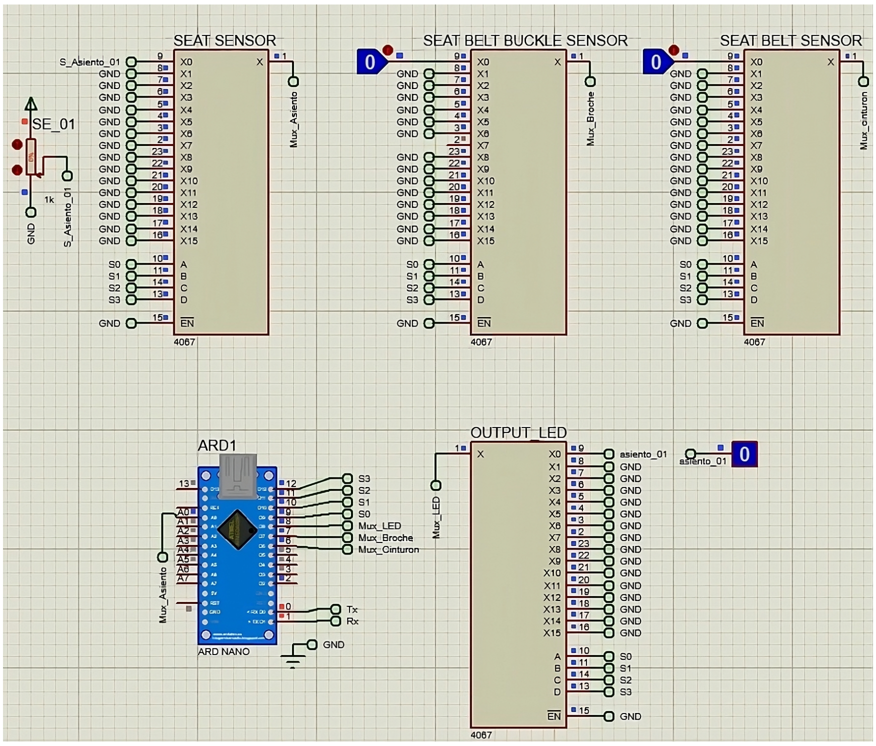


Figure 14. Passenger monitoring simulation. The simulation illustrates a scenario where the seat is unoccupied.

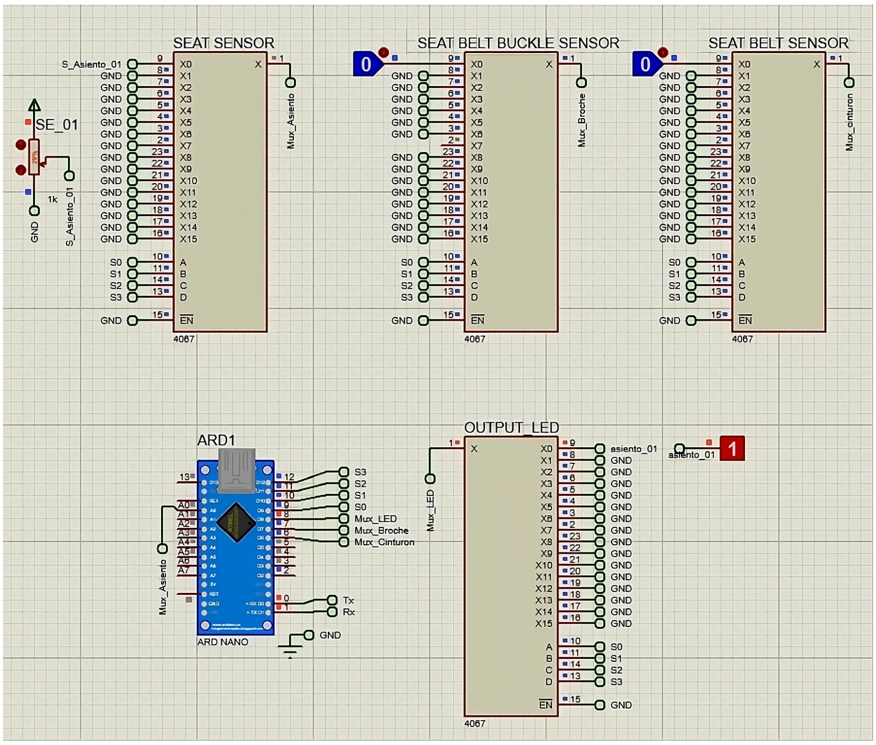


Figure 15. Simulation of a passenger without a seat belt.

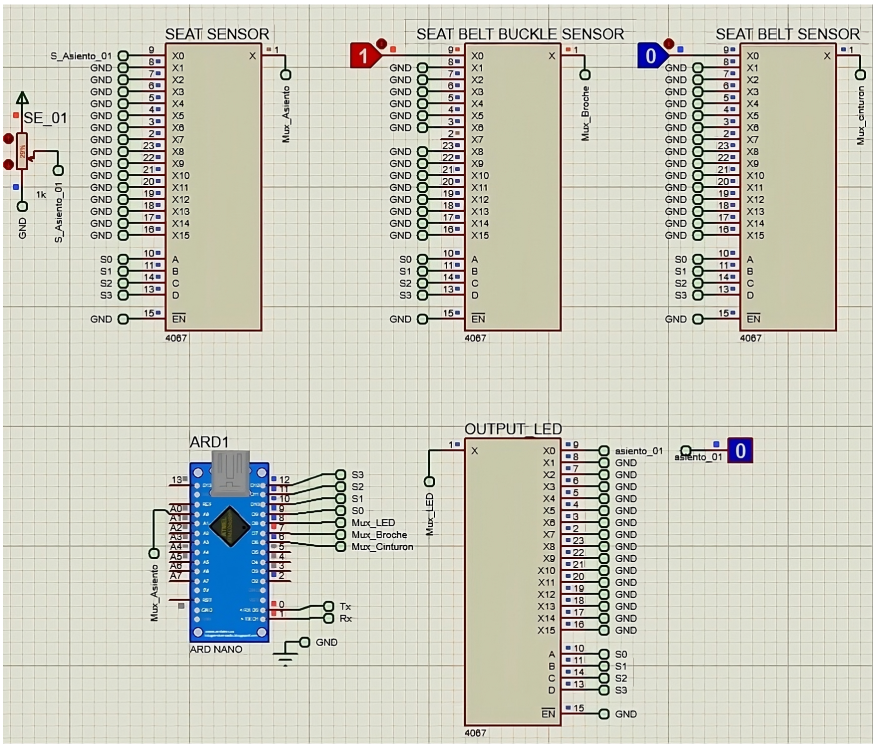
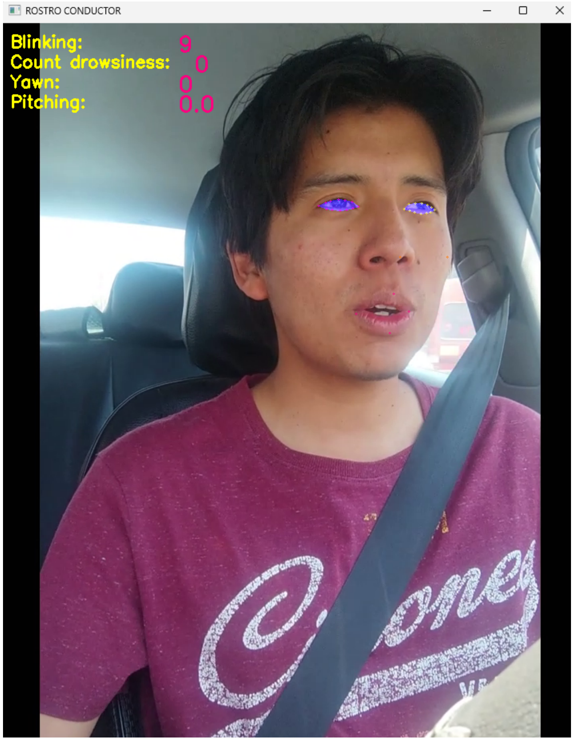


Figure 16. Passenger wearing the seat belt correctly.

The second part of the simulation focuses on the drowsiness detection system during an actual driving test.

Figure 17 shows the driver and the yellow-highlighted items that display a count of blinks, drowsiness episodes, yawns, and head pitch. The head pitch is measured in degrees. All incidents recorded during monitoring are displayed in an incident log, showing the date, time, duration, and type of incident.



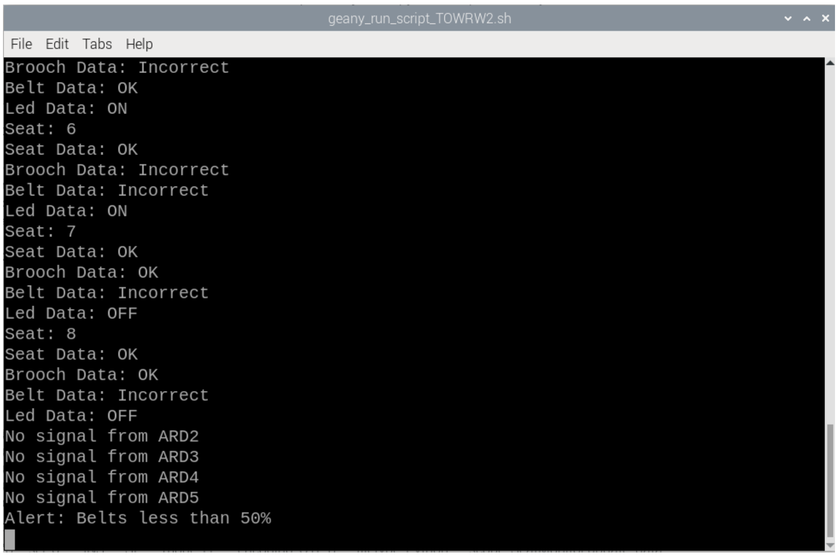
Blinking: 9
Count drowsiness: 0
Yawn: 0
Pitching: 0.0

Figure 17. Blink, yawn, and vertical pitch detection system.

4.2. Results

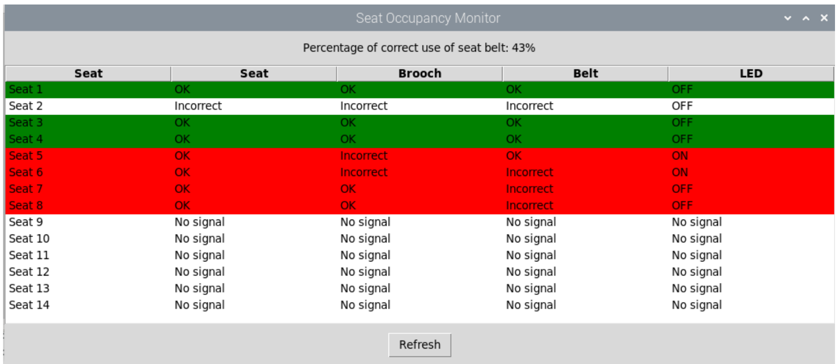
Regarding the tests performed during the simulation phase, several data points were collected and stored. When the code is initiated, a window like the one in Figure 18 is displayed, showing the data obtained. This section also indicates if there is a connection between the Arduino Nano and the Raspberry Pi 4. In the example shown, the system was set up with five Arduino Nano devices, but only one is connected.

A visual interface was designed to display the status of bus seats, as shown in Figure 19. For testing purposes, only one Arduino Nano was used to send data to the main system, and only eight seats were monitored. The interface uses different colors for each row to represent seat status. Green indicates that the passenger is correctly using the seat belt, red indicates incorrect seat belt usage, and white (or uncolored) indicates that no passenger is detected in the seat. If no signal is received from the Arduino, the interface shows "No Signal". Additionally, a percentage is displayed at the top of the window, indicating the percentage of seats with correctly used seat belts. If this percentage is less than 50%, an alert is triggered.



```
Brooch Data: Incorrect
Belt Data: OK
Led Data: ON
Seat: 6
Seat Data: OK
Brooch Data: Incorrect
Belt Data: Incorrect
Led Data: ON
Seat: 7
Seat Data: OK
Brooch Data: OK
Belt Data: Incorrect
Led Data: OFF
Seat: 8
Seat Data: OK
Brooch Data: OK
Belt Data: Incorrect
Led Data: OFF
No signal from ARD2
No signal from ARD3
No signal from ARD4
No signal from ARD5
Alert: Belts less than 50%
```

Figure 18. Console displaying a connected Arduino Nano.

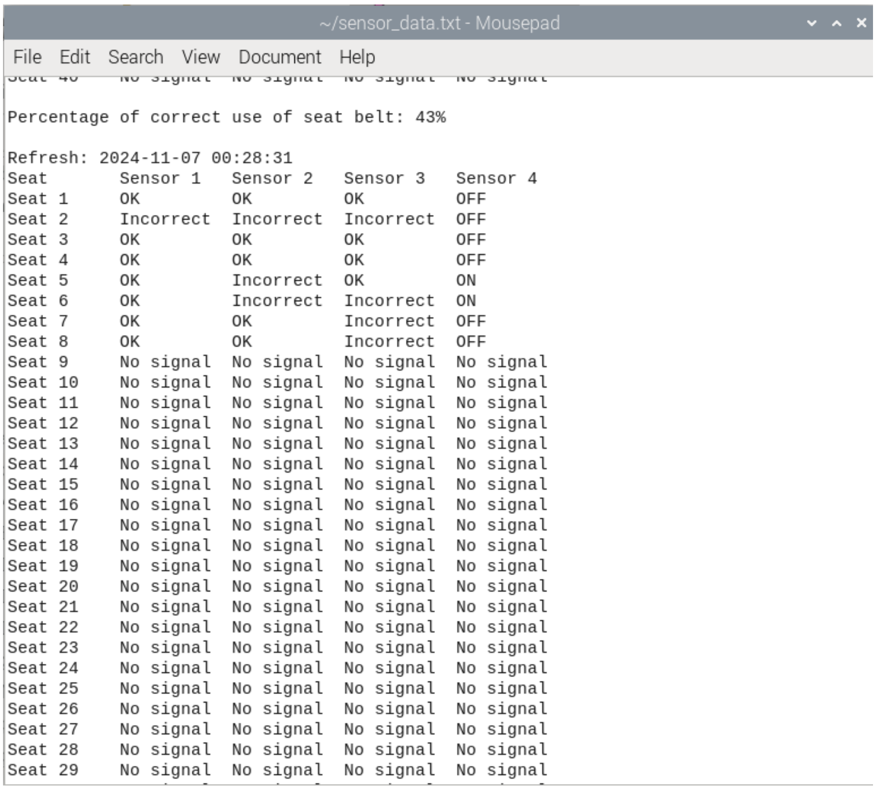


Percentage of correct use of seat belt: 43%

Seat	Seat	Brooch	Belt	LED
Seat 1	OK	OK	OK	OFF
Seat 2	Incorrect	Incorrect	Incorrect	OFF
Seat 3	OK	OK	OK	OFF
Seat 4	OK	OK	OK	OFF
Seat 5	OK	Incorrect	OK	ON
Seat 6	OK	Incorrect	Incorrect	ON
Seat 7	OK	OK	Incorrect	OFF
Seat 8	OK	OK	Incorrect	OFF
Seat 9	No signal	No signal	No signal	No signal
Seat 10	No signal	No signal	No signal	No signal
Seat 11	No signal	No signal	No signal	No signal
Seat 12	No signal	No signal	No signal	No signal
Seat 13	No signal	No signal	No signal	No signal
Seat 14	No signal	No signal	No signal	No signal

Figure 19. Seat occupancy monitor interface.

Figure 20 shows how information from the seat sensors is stored in a text file, including the time, date, and percentage of correctly used seat belts. A similar process is carried out for computer vision tests. A text file called the incident log is generated to record nodding, microsleeps, and yawns detected by the system, along with the date and time of each incident (Figure 21).

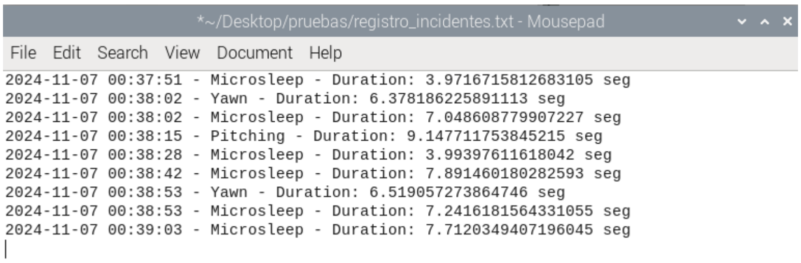

~/sensor_data.txt - Mousepad
File Edit Search View Document Help

```
Seat 40     No signal  No signal  No signal  No signal

Percentage of correct use of seat belt: 43%

Refresh: 2024-11-07 00:28:31
Seat        Sensor 1   Sensor 2   Sensor 3   Sensor 4
Seat 1      OK         OK         OK         OFF
Seat 2      Incorrect  Incorrect  Incorrect  OFF
Seat 3      OK         OK         OK         OFF
Seat 4      OK         OK         OK         OFF
Seat 5      OK         Incorrect  OK         ON
Seat 6      OK         Incorrect  Incorrect  ON
Seat 7      OK         OK         Incorrect  OFF
Seat 8      OK         OK         Incorrect  OFF
Seat 9      No signal  No signal  No signal  No signal
Seat 10     No signal  No signal  No signal  No signal
Seat 11     No signal  No signal  No signal  No signal
Seat 12     No signal  No signal  No signal  No signal
Seat 13     No signal  No signal  No signal  No signal
Seat 14     No signal  No signal  No signal  No signal
Seat 15     No signal  No signal  No signal  No signal
Seat 16     No signal  No signal  No signal  No signal
Seat 17     No signal  No signal  No signal  No signal
Seat 18     No signal  No signal  No signal  No signal
Seat 19     No signal  No signal  No signal  No signal
Seat 20     No signal  No signal  No signal  No signal
Seat 21     No signal  No signal  No signal  No signal
Seat 22     No signal  No signal  No signal  No signal
Seat 23     No signal  No signal  No signal  No signal
Seat 24     No signal  No signal  No signal  No signal
Seat 25     No signal  No signal  No signal  No signal
Seat 26     No signal  No signal  No signal  No signal
Seat 27     No signal  No signal  No signal  No signal
Seat 28     No signal  No signal  No signal  No signal
Seat 29     No signal  No signal  No signal  No signal
```

Figure 20. Method for saving seat entry data in a text file.


*~/Desktop/pruebas/registro_incidentes.txt - Mousepad
File Edit Search View Document Help

```
2024-11-07 00:37:51 - Microsleep - Duration: 3.9716715812683105 seg
2024-11-07 00:38:02 - Yawn - Duration: 6.378186225891113 seg
2024-11-07 00:38:02 - Microsleep - Duration: 7.048608779907227 seg
2024-11-07 00:38:15 - Pitching - Duration: 9.147711753845215 seg
2024-11-07 00:38:28 - Microsleep - Duration: 3.99397611618042 seg
2024-11-07 00:38:42 - Microsleep - Duration: 7.891460180282593 seg
2024-11-07 00:38:53 - Yawn - Duration: 6.519057273864746 seg
2024-11-07 00:38:53 - Microsleep - Duration: 7.2416181564331055 seg
2024-11-07 00:39:03 - Microsleep - Duration: 7.7120349407196045 seg
```

Figure 21. Method for saving incident log data in a text file.

Finally, Table 2 presents the reliability of the driver monitoring system. This table is based on two video recordings: one for driver monitoring, as mentioned in the simulation section, and another to test head nodding and yawning.

As shown in Table 2, the system achieves a reliability of 87.27% for blink detection, 94% for yawn detection, and 92% for head nodding detection. This results in an overall average system reliability of 91.09%.

Table 2. Reliability of the passenger monitoring system.

Action	Test 1	Test 2	Test 3	Test 4	Test 5	Reability
Blink. Real.	66	66	66	66	66	100%
Blink Rece.	58	56	58	58	58	87.27%
Yawn Real.	10	10	10	10	10	100%
Yawn Rece.	10	9	9	10	10	94%
Pitch. Real.	10	10	10	10	10	100%
Pitch. Rece.	10	8	10	10	8	94%

5. Conclusions

The proposed design meets the objective of monitoring driver drowsiness, supervising the correct use of seat belts by passengers, and locally storing the data on a MicroSD memory card. The proposed system is compact, portable, reliable, easy to install, and cost effective. However, it is necessary for the drowsiness monitoring system to use a high-performance IR camera to function effectively throughout the driver's workday. The system is already prepared for the integration of the IR camera. Some driver monitoring systems on the market use two alert methods: an audible alarm and vibrating seats. This system uses only an audible alarm. One of the objectives was to achieve accuracy and speed in detecting signs of drowsiness, which was accomplished through code optimization, achieving an overall reliability of 91.09%. However, this test did not consider the effect of drivers wearing glasses, which is a point to address in future work. For future work, it is necessary to develop a system with a higher level of CPU and GPU processing to enable using more robust methods, thereby achieving greater reliability. Additionally, the implementation of the proposed system is essential for collecting data that will be useful for future modifications. Another relevant point is the transfer of collected data, which could be automatically stored on servers via Ethernet or WiFi. This database should be uploaded to the server once the bus reaches the terminal.

Author Contributions: A.A.O., J.F.M.V. and G.A.E.E. conceived and designed the study; A.A.O., J.F.M.V., G.A.E.E. and J.M.M.V. were responsible for the methodology; A.A.O., J.F.M.V. and J.M.M.V. performed the simulations and experiments; A.A.O., J.F.M.V., J.M.M.V. and E.R.L.V. reviewed the manuscript and provided valuable suggestions; A.A.O., J.F.M.V., J.M.M.V. and E.R.L.V. wrote the paper; G.A.E.E. and A.O.S. were responsible for supervision. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is contained within the article.

Acknowledgments: We thank the National University of San Agustín, our alma mater, for providing us with the necessary knowledge to carry out this project.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LSTM	Long Short-Term Memory
EAR	Eye Aspect Ratio
RTC	Real-Time Clock
LCD	Liquid-Crystal Display
LFD	Learning from Demonstration

References

1. National Transportation Administration Regulations. 2020. Available online: https://www.sutran.gob.pe/wp-content/uploads/2020/08/Reglamento-Nacional-de-Administracion-de-Transporte-E28093-DS-NC2BA-017-2009MTCmodificado.pdf (accessed on 10 December 2024).
2. Seat Belt Usage in Buses—An Observation Study of Usage and Travellers' Perspectives. Available online: https://www.sciencedirect.com/science/article/pii/S0001457523001859?via%3Dihub (accessed on 25 December 2024).
3. Trauma Caused by Traffic. Available online: https://www.who.int/es/news-room/fact-sheets/detail/road-traffic-injuries (accessed on 24 December 2024).
4. Statistical Report on Road Accidents 2022. Gob.pe. Available online: https://cdn.www.gob.pe/uploads/document/file/4489498/Reporte%20Estad%C3%ADstico%20de%20Siniestros%20Viales%202022.pdf (accessed on 24 August 2024).

5. NATIONAL ROAD SAFETY OBSERVATORY, [no Date]. ONSV—Road Accident Report and Actions to Promote Road Safety. Gob.pe. Available online: https://www.onsv.gob.pe/post/informe-de-siniestralidad-vial-y-las-acciones-para-promover-la-seguridad-vial/ (accessed on 15 December 2024).
6. Martínez-Estrada, M.; Gil, I.; Fernández-García, R. Automotive Seat Occupancy Sensor Based on e-Textile Technology. *Eng. Proc.* **2023**, *30*, 7. [CrossRef]
7. Kisic, M.G.; Blaz, N.V.; Babkovic, K.B.; Zivanov, L.D.; Damnjanovic, M.S. Detection of Seat Occupancy Using a Wireless Inductive Sensor. *IEEE Trans. Magn.* **2017**, *53*, 4001204. [CrossRef]
8. Cheng, H.C.; Chang, C.C.; Wang, W.J. An Effective Seat Belt Detection System on the Bus. In Proceedings of the 2020 IEEE International Conference on Consumer Electronics—Taiwan (ICCE-Taiwan), Taoyuan, Taiwan, 28–30 September 2020; pp. 1–2. [CrossRef]
9. Belt Force Sensors. Sensing. Online Measurement Sensors, 2017. Available online: https://sensores-de-medida.es/catalogo/sensores-de-fuerza-para-cinturon/ (accessed on 24 May 2024).
10. Eye Alert—Distracted Drivg and Fatigue Monitors—Highway Safety Group—Products, [no Date]. Available online: https://eyealert.com/index.html (accessed on 12 December 2024).
11. SMI Eye Tracking Glasses, [No Date]. iMotions. Available online: https://imotions.com/hardware/smi-eye-tracking-glasses/ (accessed on 12 December 2024).
12. Optalert—Drowsiness, OSA and Neurology Screening, [No Date]. Available online: https://www.optalert.com/ (accessed on 15 December 2024).
13. TSO Mobile Miles Ahead, [No Date]. Online. Available online: https://tsomobile.com.pe/blog/sensor-de-fatiga-para-flotas/ (accessed on 10 December 2024).
14. Driver Monitoring System DMS-STONKAM CO., LTD., [No Date]. Stonkam. Available online: https://es.stonkam.com/products/Driver-Status-Detection-System-DMS31.html/ (accessed on 12 December 2024).
15. Operator Alert System Hexagon OP, [No Date]. Hexagon. Available online: https://hexagon.com/es/products/hexagon-op-operator-alertness-system (accessed on 12 December 2024).
16. Berlanga, J.M.J. Detección de Somnolencia y Síncope en Conductores Mediante Visión Artificial. Available online: https://openaccess.uoc.edu/bitstream/10609/132366/7/jmjberlangaTFM0621memoria.pdf (accessed on 24 July 2024).
17. Muños, E.L.B.; Mendez, M.M.M. Sistema Basado en la Detección y Notificación de Somnolencia Para Conductores de Autos. Montería: Universidad de Córdoba. 2015. Available online: https://repositorio.unicordoba.edu.co/server/api/core/bitstreams/2ac71ade-9e9b-47e8-b787-66e0b3721aae/content (accessed on 24 July 2024).
18. Soukupova, T.; Cech, J. Real-Time Eye Blink Detection Using Facial Landmarks. Uni-lj.si. Available online: https://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf (accessed on 24 July 2024).
19. Madruga, J.M. Sistema de Detección de Emociones Faciales Mediante Técnicas de Machine Learning Adaptado a ROS Para un robot de Bajo Coste Basado en Raspberry Pi. España: Universidad Rey Juan Carlos, 2022. Available online: https://gsyc.urjc.es/jmvega/teaching/tfgs/2021-22_JavierMartinez.pdf (accessed on 24 July 2024).
20. PHAN, Anh-Cang, TRIEU, Thanh-Ngoan y PHAN, Thuong-Cang, 2023. Driver Drowsiness Detection and Smart Alerting Using Deep Learning and IoT. Internet of Things. Online. 2023. Vol. 22, no. 100705, p. 100705. Available online: https://www.sciencedirect.com/science/article/abs/pii/S2542660523000288 (accessed on 24 July 2024).
21. Quddus, A.; Zandi, A.S.; Prest, L.; Comeau, F.J.E. Using long short term memory and convolutional neural networks for driver drowsiness detection. *Accid. Anal. Prev.* **2021**, *156*, 106107. [CrossRef] [PubMed]
22. Chris, S.; John, G.; Yousefian, R. Multi-sensor driver monitoring for drowsiness prediction. *Traffic Inj. Prev.* **2023**, *24* (Suppl. S1), S100–S104. [CrossRef]
23. Zain, Z.M.; Roseli, M.S.; Abdullah, N.A. Enhancing Driver Safety: Real-Time Eye Detection for Drowsiness Prevention Driver Assistance Systems. *Eng. Proc.* **2023**, *46*, 39. [CrossRef]
24. Xinjiejia, SBR JYJ-105. Diaphragm Pressure Shenzhen Xinjie Jia Electronic Thin Film Switch co. Available online: http://en.szxjj.com/index.php?m=content&c=index&a=lists&catid=13 (accessed on 24 July 2024).
25. Seat Belt Proximity Sensor, [no Date]. Directindustry.es. Online. Available online: https://trends.directindustry.es/soway-tech-limited/project-161356-168240.html (accessed on 16 July 2024).
26. Arduino Documentation. Arduino.cc. Available online: https://docs.arduino.cc/hardware/nano/ (accessed on 24 July 2024).
27. 74HC4067 Multiplexer Analog-Digital Module 16ch. NaylampMechatronics-Perú. Available online: https://naylampmechatronics.com/circuitos-integrados/644-modulo-74hc4067-multiplexor-analogico-16ch.html (accessed on 24 July 2024).
28. Chizzolini, S. Arduino-Ad-Mix-Lib. 2020. Available online: https://github.com/stechio/arduino-ad-mux-lib (accessed on 24 July 2024).
29. Raspberry Pi 4. Available online: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ (accessed on 24 July 2024).

30. Cámara Web Full Hd 1920 X 1080p Con Micrófono Usb Pc Laptop. Available online: https://articulo.mercadolibre.com.pe/MPE-670143902-camara-web-full-hd-1920-x-1080p-con-microfono-usb-pc-laptop-_JM? (accessed on 24 July 2024).
31. Medrano, J. Vivir, Bostezar, Morir. Rev. Asoc. Esp. Neuropsiq. **2013**, *33*, 117. Available online: https://scielo.isciii.es/pdf/neuropsiq/v33n117/11.pdf (accessed on 24 July 2024).
32. Sanchez, S. Detección de Rotacion del Rostro | Deteccion de Rostros en 3D con Phython y Opencv. 2022. Available online: https://youtu.be/cTpTjGK8HME?si=DZcaUixHaZQ7-iZd (accessed on 20 July 2024).
33. Gizmo Mechatronics Central. tINYrtc i2c mODULE. 2016. Available online: https://pdf.direnc.net/upload/tinyrtc-i2c-modul-datasheet.pdf (accessed on 24 July 2024).
34. Overview, 800 x. 480 Dfr0550. Available online: https://www.mouser.com/pdfDocs/ProductOverview_DFRobot-DFR0550-2.pdf (accessed on 24 July 2024).
35. RS, R.Y. FUNDAMENTAL CONCEPTS OF. Novusautomation.com. Available online: https://cdn.novusautomation.com/downloads/conceptos%20fundamentales%20de%20rs485%20y%20rs422%20-%20espa%C3%B1ol.pdf (accessed on 24 July 2024).
36. Maxim Integrated. MAX481/MAX483/MAX485/ MAX487–MAX491/MAX1487. 2014. Available online: https://www.analog.com/media/en/technical-documentation/data-sheets/MAX1487-MAX491.pdf (accessed on 24 July 2024).
37. Waveshare RS485 Can Hat User Manual. Available online: https://www.waveshare.com/w/upload/2/29/RS485-CAN-HAT-user-manuakl-en.pdf (accessed on 24 July 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.