AIDVISION PRO

CRITICAL DESIGN REPORT

PROTOTYPE DESIGN

NURIA BALBÁS, CARLOS CELAYA, NOELIA REDONDO, PAULA MÉNDEZ AND SANDRA PINEL

INSTRUCTORS: ANDRÉS GARCÍA, LAURA MARÍA AGUILAR, CÉSAR GARCÍA ANDCRISTINA MELERO

FRANCISCO DE VITORIA UNIVERSITY | FOURTH YEAR OF BIOMEDICAL ENGINEERING

INDEX

1. INTRODUCTION

1.1 SCOPE

The goal of this project is to develop smart glasses that help visually impaired individuals navigate their environment by detecting nearby objects, mainly cars, people and walls. These glasses are designed to enhance users' independence, so our visual and sensory identity aligns with this objective. Our logo uses blue to symbolize confidence and independence while white is used for clarity. Our sensory identity includes mint and lavender scents to provide a sense of calm and confidence.

These smart glasses are directed to customers who are totally or partially blind. We want to provide them an increased safety by allowing them to have easiness to avoid obstacles and object recognition to give them more guidance. Moreover, we presented two interviews, one with Manuel Cruz, a visually impaired person, and another one with ASPREH, an association for individuals with visual disabilities. These interviews helped us clarify our customer needs, highlighting the importance of safety and the need to increase the independence of the users and improve their quality of life.

Based on this, our glasses must detect obstacles at a distance of 1 – 1.5 meters, differentiate people and cars and alert through specific vibrations and sounds about the different obstacles that the individual could encounter. Moreover, we wanted to create comfortable, lightweight glasses with the capacity of being rechargeable and with desirable features, such as customizable vibration and sounds or the ability to detect other objects in the users' environment. To achieve this, we used both hardware and software components, which were selected taking into consideration the trade-offs.

To prove the effectiveness of our glasses we will prove them in different scenarios:

- Firstly, **navigate through an unfamiliar public building or street**. As the user walks, our product will detect walls, doors or an entire building with an ultrasound sensor so when he approaches the obstacle, the device triggers a vibration alert. This will allow the user to adjust their direction and safely walk through that unfamiliar environment.
- Another scenario is the **detection cars in the street**. The individual is in an intersection without a traffic light and doesn't know if a car is approaching. Our glasses detect it and alerts him with an audible warning. Moreover, he will know the number of cars present and if they are to the side or facing front.
- The third scenario is the **detection of people in the street or closed space**. When the user is walking on a sidewalk and a person approaches him, our device will detect it and warn him with another audible sound, so he can walk through an obstacle-free route. As in the scenario mentioned before, our customer will know how many people are on each side.
- Finally, the fourth scenario is **other objects detection (not only people and cars)**. In this case, if the user is walking through a street with a specific path for bicycles and a bike has gone out of the street to the crosswalk, the glasses will detect the bike. They will alert the user with another specific sound when the object is moving towards them and from which side does it come from.

1.2 DEFINITIONS, ACRONYMS AND ABBREVATIONS

- ASPREH: Asociación de Profesionales de la Rehabilitación de Personas con Discapacidad Visual.
- GUI: Graphical User Interface.
- SSID: Service Set Identifier.
- UDP: User Datagram Protocol.

2. CDR

2.1 SYSTEM OVERVIEW

2.1.1 Purpose, applications and objectives of the prototype

Main Purpose

- Create an assistive device in the form of glasses that will help individuals with vision loss to detect objects and obstacles in their way, which will allow users to move more safely and confidently.

Objectives

- Identifying and classifying nearby objects, such as people and cars, through image analysis.
- Provide specific information about the type, number and direction of the object detected.
- Detect nearby objects and alert the user using vibrations. Prevent collisions and help the user orient themselves in their environment.

Applications

- Daily life of blind people

The purpose of the prototype is to create an assistive device in the form of glasses that will help individuals with vision loss to detect objects and obstacles in their way, which will allow users to move more safely within their environment.

A. Ultrasound- vibration motor

The **ultrasound- vibration motor** module will provide vibrational feedback based on objects proximity, making users more aware of their surroundings. The main objectives of this module include:

- Detecting the distance to near objects with the ultrasound sensor
- Different vibration depending on the distance the obstacle is at
- Easy use ON and OFF switch to be able to stop the device

B. ESP32-CAM

The **esp32-CAM** module will provide the live-video of the surroundings of the user, allowing a better interpretation of the objects that can be found and where they are. The main objective of this module include:

- Allowing the detection of visible objects
- Sending the video to a reachable IP that can be accessed
- Obtaining basic information of the position of obstacles and tools

C. DF Player

The **DFPlayer** module will provide audio feedback to the user, enhancing the information from other modules to make it more accessible. The main objectives of this module are:

- Play pre-recorded messages to alert users to specific types, number and location of obstacles (e.g. person, cars).
- Easy use ON and OFF switch to be able to stop the device

2.1.2 Functional description of the design

KEY COMPONENTS OF THE PROTOTYPE

- Sensors

  Ultrasound sensor

  This sensor is responsible for detecting what is in front of them and measuring the distance. It works by emitting a high frequency sound wave and when it gets to the object it comes back, it measures the time it takes for the echo to return to the sensor. This time is converted to distance.

- Processors

  ESP32 Wroom-32

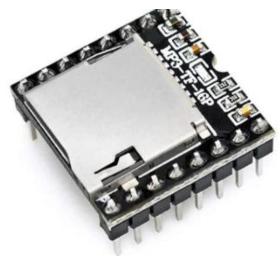
  The processor in this case is the ESP32 module. It is a microcontroller, which is the main processing unit of the device. It processes data in real time. It has low power consumption, many input and output pins which we can use for all the components, it has Wi-Fi and Bluetooth, and it has a small size, perfect for our glasses.

  - For the **ultrasound-vibration motors module**, the ESP32 receives the distance data from the ultrasound sensor processes it in real time and determines the response for how many motors to activate. It is also connected to the switch.
  - For the **DFPlayer Mini MP3 Player module and speaker**, the ESP32 controls when and which audio to play depending on the result of the YOLO detection. When the ESP32 receives the output of the detection of the algorithm it initiates a serial communication with the DFplayer which starts playing the configurated audio through the speaker.

- Other modules and components

  DFPlayer mini MP3 player + speaker

  The DFPlayer Mini MP3 Player is a small and compact module that can be used to play MP3 audio files from an external storage source such as a microSD card. It is based on the DFPlayer IC, a type of integrated circuit that can control audio playback and manage communication with an external storage source.



  This module is widely used in various applications such as sound effects, background music, voice announcements, and other audio-based projects. It is a small and easy-to-use audio module that requires minimal configuration and wiring. Using the appropriate library and codes, the module can be programmed to play audio files in the desired order and at different volumes.

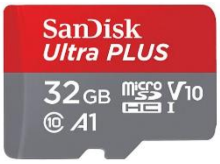
  The module we bought came with two speakers. For the moment one speaker will be used, and another one will be considered if needed.

  NOTES:

  → *microSD cards need to be in FAT16/FAT32 format*
  → *It only supports microSD card up to 32 GB*
  → *Speakers can be 3W at most*
  → *The audio recorded in the microSD card needs to be in a carpet named 'mp3'*

- ***32 GB MICROSD CARD***

  The audio that we want the DF player to play must be recorded on a microSD card. The DFplayer has a card slot where the microSD card can be introduced. When the esp32 receives data from the computer it will reproduce the corresponding sound based on the audio recorded on this card. It is very important to consider the notes mentioned in the previous section for the correct configuration and functioning of the audio.



- ***6 vibration motors***
- ***1 push button***
- ***1 power bank of 10000mAh***
- ***1 220Ω resistance***
- ***Wires***
- ***2 USB cables***
- ***PLA***
- *ESP32-CAM*

DESCRIPTION AND REPRESENTATION OF THE GENERAL ARCHITECTURE

The general architecture has **2 modules**:

1) **PROXIMITY DETECTION *(Ultrasound and Vibration Motors)***
   - → **Function:**
     Detect nearby objects and alert the user using vibrations. It is designed to prevent collisions and help the user orient themselves in their environment.
   - → **Components:**
     -) ***Ultrasound Sensor:***
       Measures the distance to nearby objects (up to 5 meters). If an object is less than that distance, a vibration pattern is activated.
     -) ***Vibration motors:***
       Activated based on the proximity of the object; the closer it is, the more motors are activated, providing tactile feedback that helps the user orient themselves.

The **ultrasound-vibration motor** module will work by using an ultrasonic sensor to measure distances. Based on the distance vibration motors on each side of the glasses will be activated. These motors will be arranged with three on each side, allowing balanced and intuitive feedback. The system will work as following:

- If an object is detected within 20cm, two vibration motors will be activated, one in each side of the glasses.
- If an object is detected within 10cm, four vibration motors will be activated, two in each side of the glasses.
- If an object is detected within 5cm, six vibration motors will be activated, three in each side of the glasses.

→ **<u>Operation</u>**

The ultrasonic sensor continuously measures the distance to objects. If an object is detected less than 5 meters away, a signal is sent to the vibration motors to activate, alerting the user of its proximity.

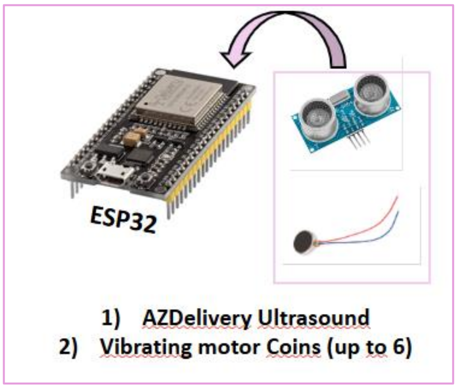


Figure 1: *Proximity Detection Module Architecture*

2) **<u>OBJECT (people/cars) DETECTION</u>** ***(Accessory camera, YOLO, Speaker Response)***

→ **<u>Function:</u>**

This module is responsible for identifying and classifying nearby objects, such as people and cars, through image analysis. It provides specific information about the type of object detected.

→ **<u>Components:</u>**

) ***Accessory camera (CAM):***
Captures images of the environment and sends them to a computer for analysis.

) ***Computer:***
Processes the images using an object detection model (YOLO) to identify whether people or cars are present.

) ***Speaker:***
Plays an audio message to the user informing them of the number of objects detected, such as "2 people" or "1 car".

→ **<u>Operation</u>**

The CAM captures images of the environment and sends them to the computer wirelessly.
- The computer analyzes the images and counts the number of people and cars detected.
- This information is sent back to the second ESP32, which activates the speaker to inform the user.

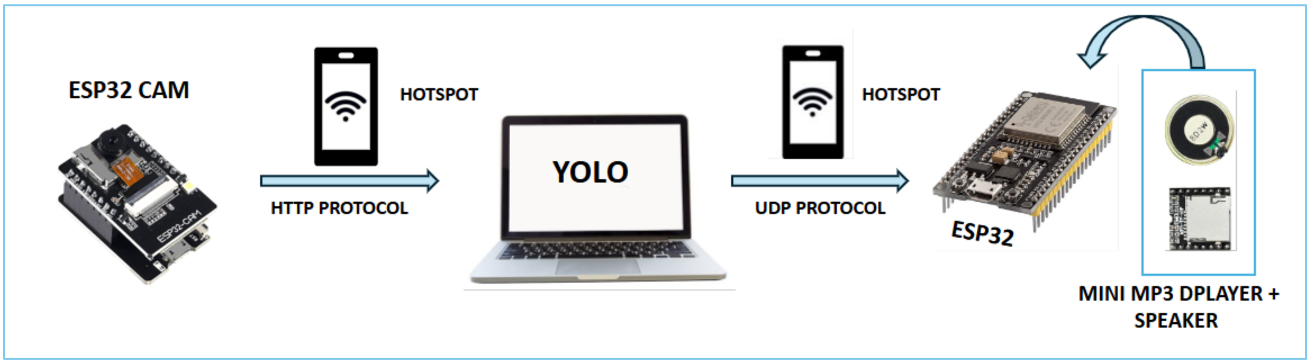


Figure 2: *Diagram of the functioning of the prototype*

STOP ACTION. *The model contains a switch that when pressed will stop the program that is running for all the accessories.*

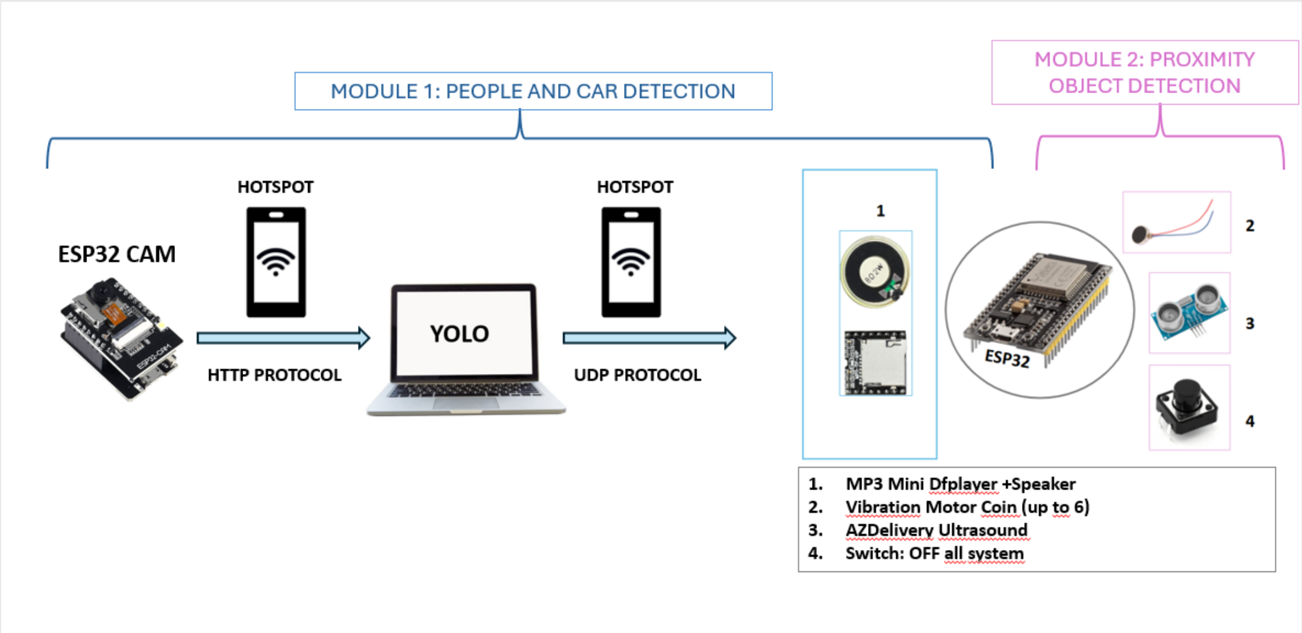


Figure 3: *Diagram of the functioning of the prototype*

3.1.3 General architecture: system block diagram

A) Ultrasound- vibration motor

The architecture of the ultrasound and vibration motors module is designed around the ESP32 microcontroller that receives distance measurements from the ultrasound and controls the vibration accordingly, and with the switch to activate or deactivate the system.

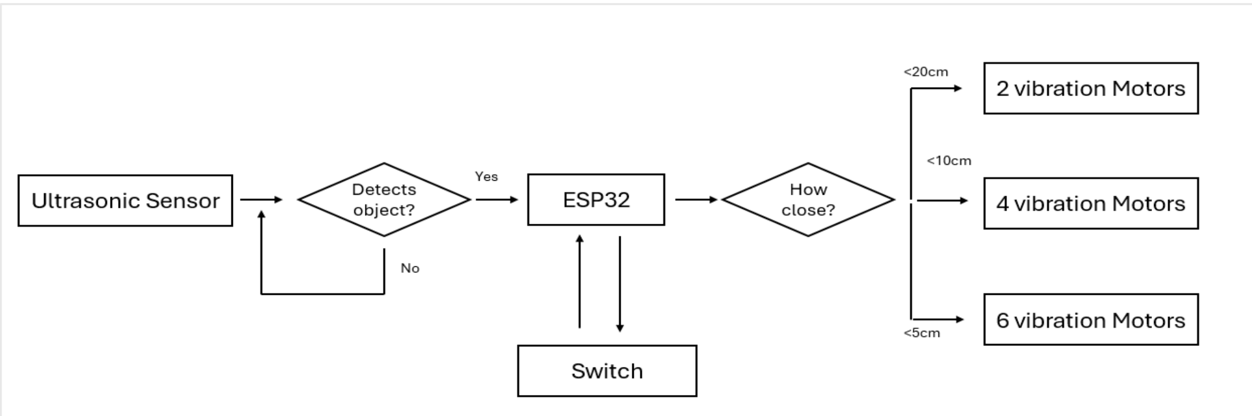


Figure 4: *Ultrasound-vibration motors System Block Diagram*

In the diagram (Figure 2) it is shown:

- The ultrasound sensor continuously measures the distance to nearby objects and sends the information to the ESP32.
- The ESP32 processes distance information and determines how many motors to activate depending on the threshold distances.

- The vibration motors vibrate until there is no object.
- The switch stops or activates again the system.

B) ESP32-CAM and DF Player

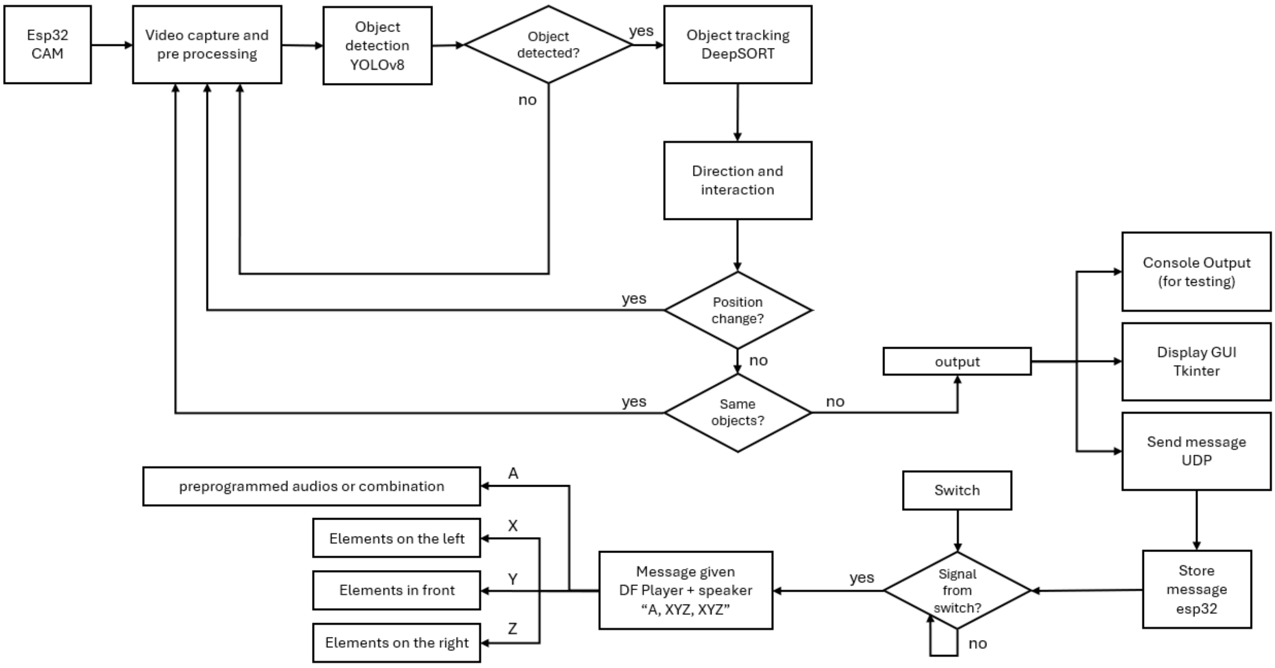


Figure 5: *Camera-computer-esp32-DfPlayer System Block Diagram*

The architecture of the esp32 CAM and DF Player is designed around the program located in the computer, that obtains the video captured by the esp32 cam and controls the audio output according to the objects detected and their respective information, being the final output an audio with the information about the objects, number and position.

In the diagram it is shown:

- The video is continuously being processed by the yolov8 and providing an output
- The message from the output is only sent if it is not the same as the previous one
- The three main outputs from the processing of the video are seen in the console, as a GUI or video in the screen with the rectangles of the objects detected and IDs and the message is sent to the esp32
- The message is only reproduced by the DFPlayer if the switch has been pressed (only reproduces the last message received)

3.2 HARDWARE DESIGN

3.2.1 Physical Design

The physical design must ensure that all parts of the glasses are optimally positioned to maximize user comfort while allowing the successful functioning of each electronic element. First, we will outline the primary requirements for the electrical components and the glasses, noting both essential and preferred features. We will then explain how we addressed these challenges, assessing our success or failure. To begin, here is a summary of the requirements our glasses design must meet and the recommendations for additional features:

- The design must be ergonomic and comfortable to wear, as the user will wear it for extended periods. Obvious comfort improvements should be considered in the design, such as minimizing or concealing wires from the electrical components.
- Optimal placement of the ultrasound sensor: The sensor should be located close to the user's eye to replicate the spatial awareness that sight provides. Additionally, the sensor's angle should provide sufficient awareness to help the user avoid head-level obstacles.
- Effective placement of the ESP Cam module: Similar to the ultrasound system, careful consideration should be given to the placement of the ESP Cam so that the user receives notifications about nearby objects. By positioning it to mimic the natural placement of eyes, evolved over time, we aim to achieve maximum efficiency.
- Strategic placement of the speakers: We decided to place the speakers near the ear, but without replicating a traditional headphone design or obstructing the user's hearing. Hearing is critical for blind individuals, so covering the ear could impair their experience and reduce the effectiveness of the device.
- Positioning of vibration motors: The motors will be attached to the inner side of the temple arms, allowing the user to feel the vibrations easily. The motors come with adhesive backing, simplifying their attachment and ensuring secure placement.
- Robustness of the design: The glasses should be as durable as possible because they will experience frequent movement and accidental drops. It would be detrimental for the user if the device broke or if the electrical components failed due to minor shocks or vibrations.
- Considerations for prototype design: As this is a prototype, every component should be as small as practical, but size reduction should not compromise functionality. For example, a smaller ultrasound system might limit functionality, so achieving full functionality is prioritized over minimal size. Optimal component arrangement is essential, but compactness is secondary in this phase.
- Aesthetic considerations: While the device's appearance is considered, it remains secondary to functionality.

Some general considerations for the design include that all parts will be 3D-printed due to the convenience of producing precise components, with PLA white as the chosen material. This material was selected for its availability at the university, along with its excellent rigidity, ease of fabrication, and cost-effectiveness. The white color also allows for easier painting in the future.

The screws used are M2, chosen for their small radius and variable length, approximately 1 to 2 cm. Screws offer a consistent way to attach components, but their application can be challenging, so nuts are used on the reverse side with a general hole diameter of 2.5 mm, including tolerances in the design. The leght of each screw depends on each part of the design.

Finally, we prioritized working with tolerances over precise fitting. For example, when screwing parts together, the lack of tolerance could result in bending and ultimately prevent proper screw use. In the image bellow you can see what happens when you try to use a screw between a space that doesn't have a minimum space or tolerance:

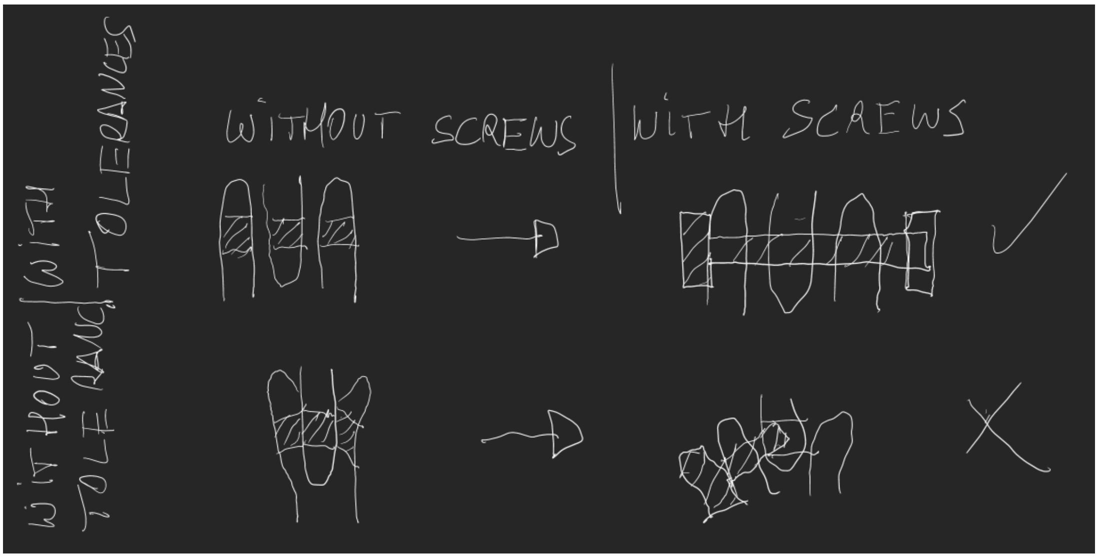


Figure 6: *tolerances followed for the design*

Figure 7: *Initial points of union for the sensors in the glasses shown in pink*

<u>3D Designs for Component Attachment</u>

We designed three different types of 3D components, each tailored to the specific electrical component and its shape:

A. Ultrasound Sensor Base:

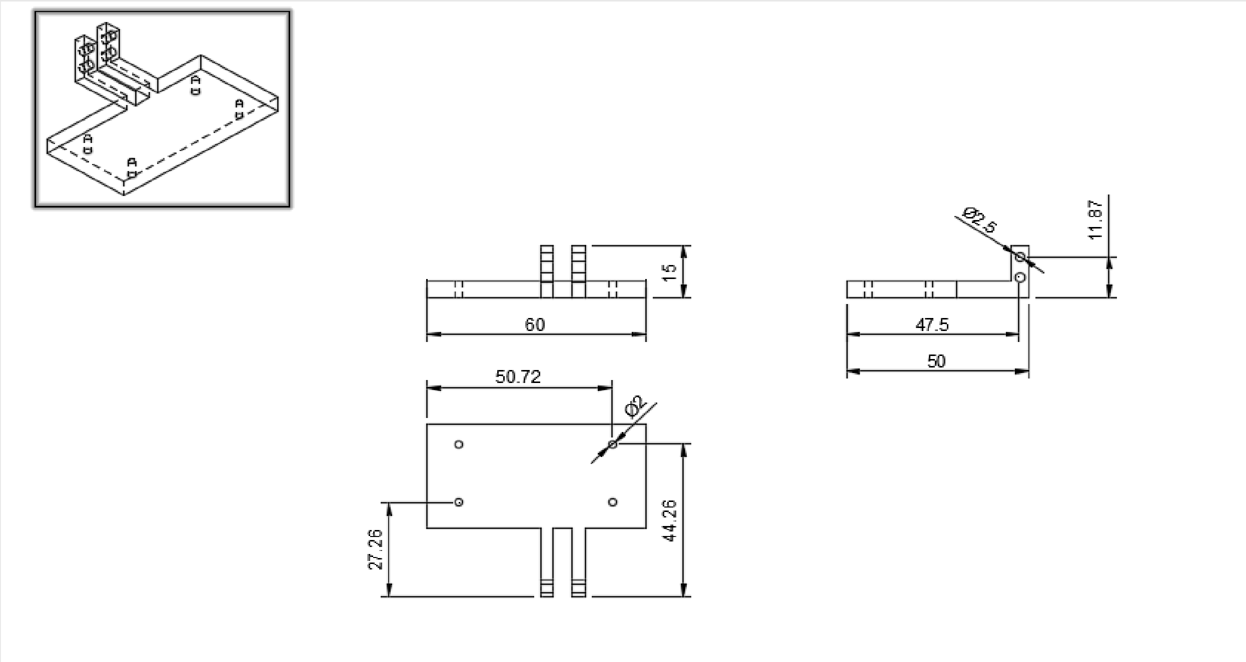


Figure 8. *Ultrasound sensor base with the measurements*

This sensor features four holes at each corner of the device to allow attachment to a base if necessary. We will use these four holes to secure the sensor to the base, and the holes on the bottom of the 3D-printed part will be used to attach the base to the sunglasses.

B. ESP32-CAM Module

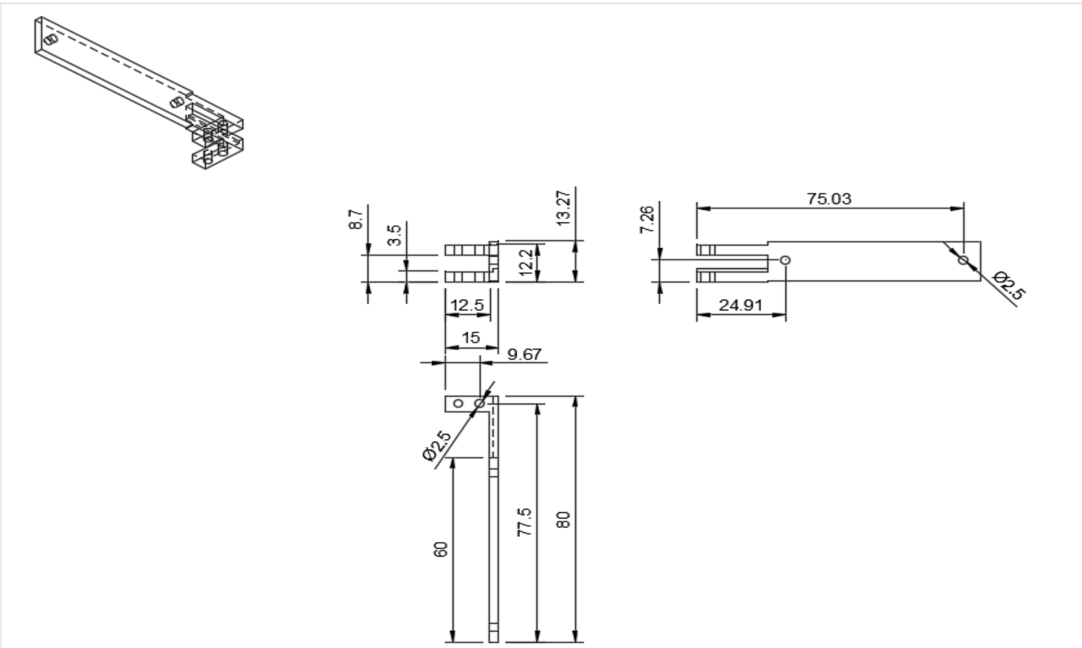


Figure 9. *Esp32-CAM module1 with measurements*

The ESP CAM module does not have any pre-existing holes or structures to facilitate attachment to the sunglasses, so we designed two parts that will wrap around the ESP CAM and attach to the sunglasses.

Although this system isn't perfect, it's the best solution we could achieve with the available tools. Ideally, we would have printed half of the frame, inserted the ESP CAM module, and continued the print to wrap the module entirely. However, due to the limitations of our 3D printers and resources, this approach may introduce potential errors. The base is simple but must be narrow enough to keep the pins of the ESP CAM exposed.

Additionally, we took special care in designing the cover for the ESP CAM, ensuring it wasn't too soft. We also created a corner specifically to accommodate the shape of the ESP CAM. However, this corner made it impossible to print the most logical piece (Frame 1), as the printer couldn't process one of the parts of the design. As a result, we redesigned it to Frame 2.

- **Frame 1:**

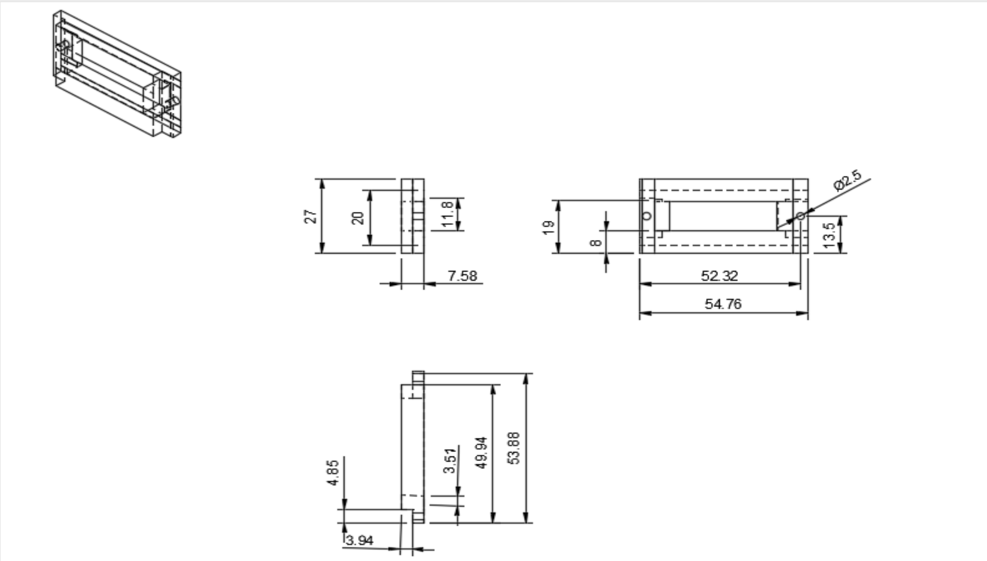


Figure 10. *Esp32-CAM frame 1 with measurements*

- **Frame 2:**

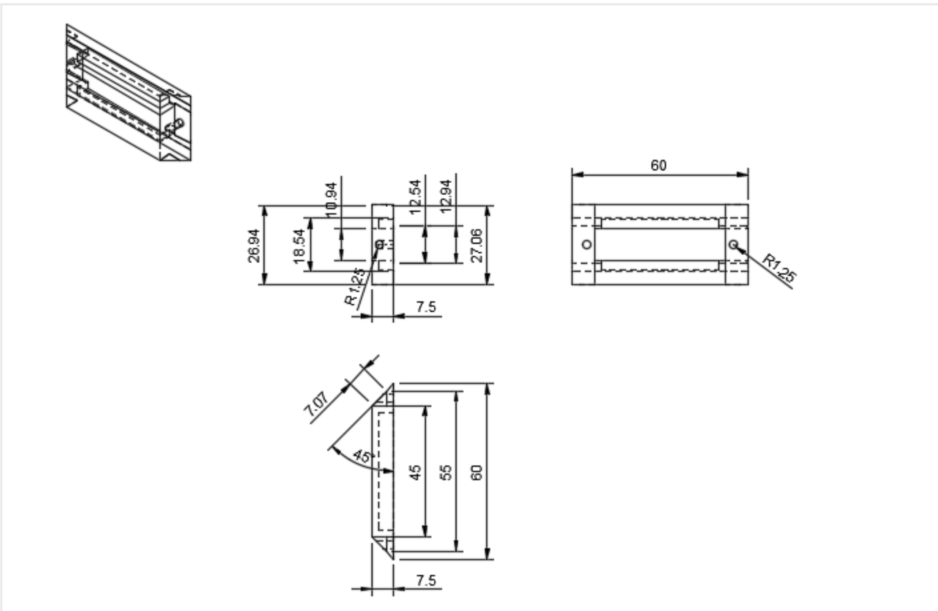


Figure 11. *Esp32-CAM frame 2 with measurements*

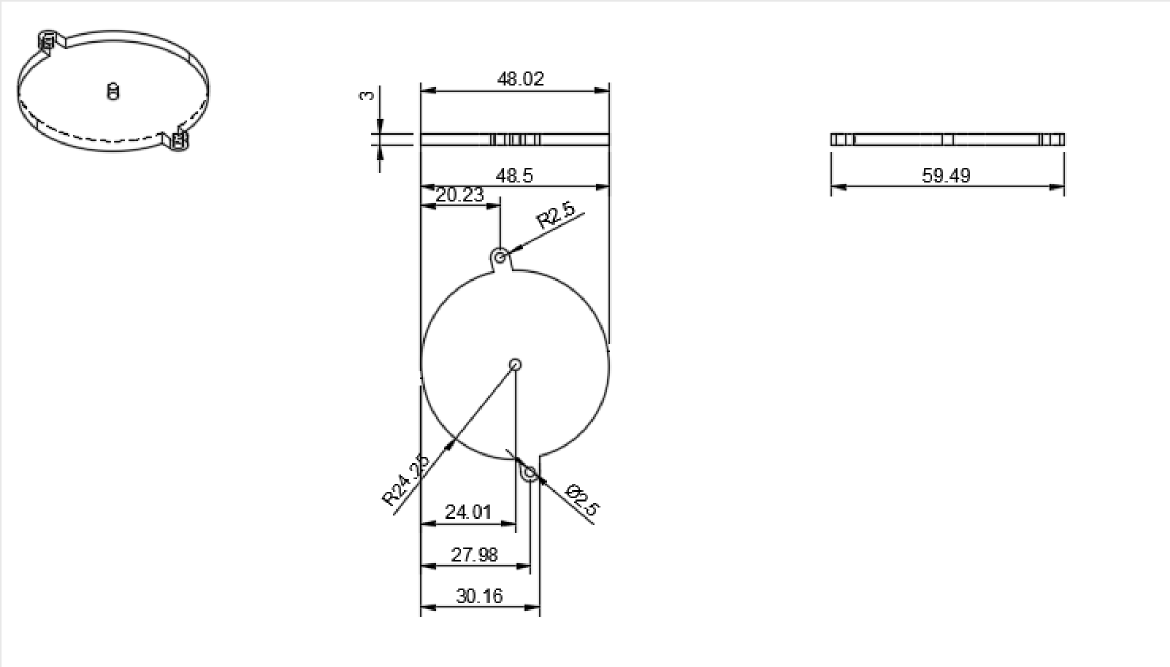


Figure 12. *Speakers' module with measurements*

C. Speakers:

The speaker attachment system is similar to the one we used for the ESP CAM, but with a key difference: the speakers do not need to be fully wrapped. Since the surface behind the functional part is flat, we can glue the speakers directly to one piece of the frame. Additionally, we used only one screw for the attachment, as the speaker's circular design allows it to rotate indefinitely without any issues.

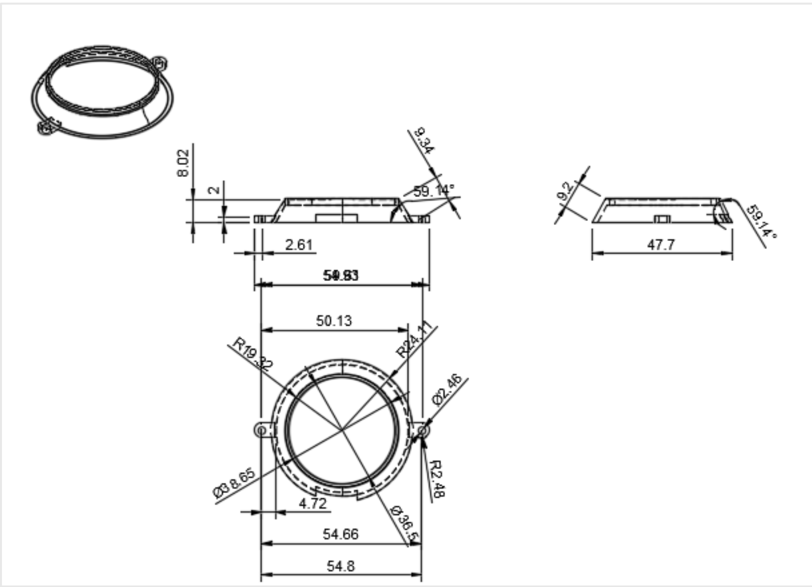


Figure 13. *Cover for the speaker with measurements*

We also designed a cover for the speaker in case a more compact solution is required.

Once these changes were finalized, we proceeded to assemble all the components into the sunglasses, and here is the result:

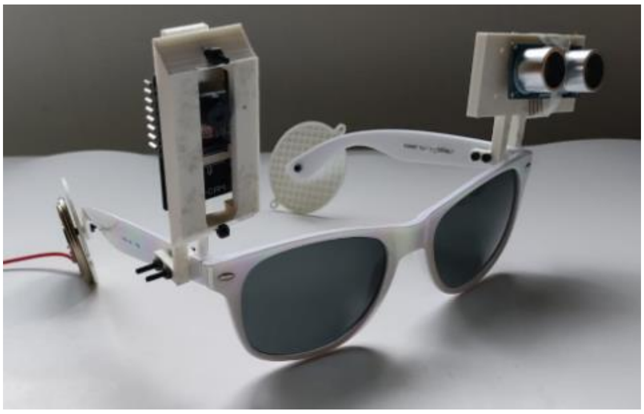


Figure 14. *Version 1 of the glasses*

As you can see, the design is successful in many aspects, but there are still areas for improvement, such as robustness and the fact that the wires are exposed. This exposes the user to potential risks, and the product is more likely to break. For this reason, we explored an alternative development that will be shown in the final design section.

Some of the previous discussions highlighted that the design had several issues, such as exposed wires and slight movements of the components, which could impair the functionality of the glasses. To address this, we propose an entirely new design for the glasses, aiming to resolve both of these problems by manufacturing the frame from scratch.

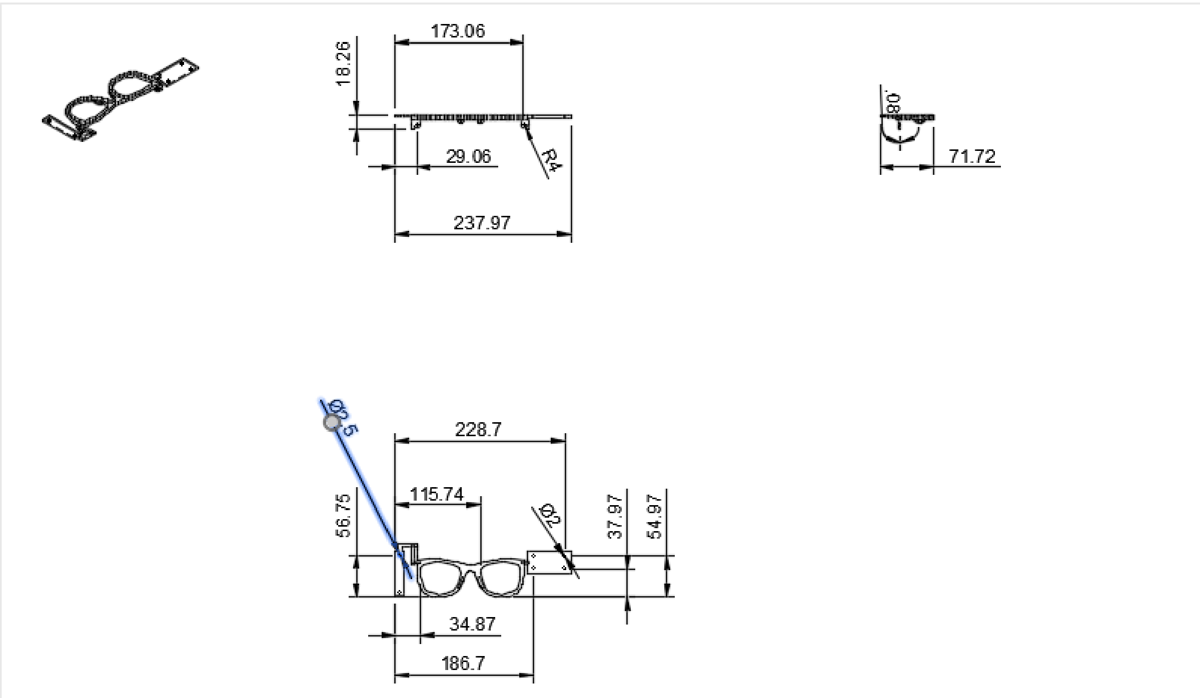


Figure 15. **S***econd design for the glasses with measurements*

- **Monture design:** The redesigned frame incorporates a base for the ultrasound sensor, which helps reduce vibration due to minimized tolerances. Other than this addition, the frame remains largely similar to the initial design.

- **Temple (Branch) Design:** The temple design now allows for internal routing of wires and includes other improvements, such as the addition of a button on each side. As previously mentioned, the constraints of 3D printing required us to modify the manufacturing process: one part will serve as the base, and the other will be placed on top. This adjustment avoids the need for supports in 3D printing, making assembly easier. Below is the final result for the base and top parts:

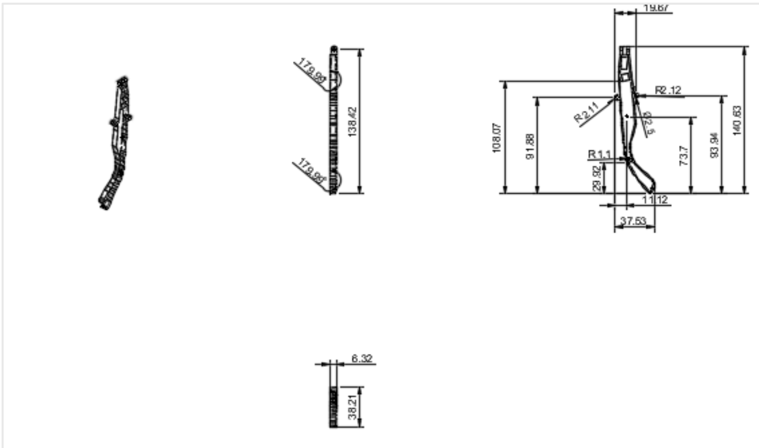


Figure 16. *Second design of the glasses (temple) with measurements*

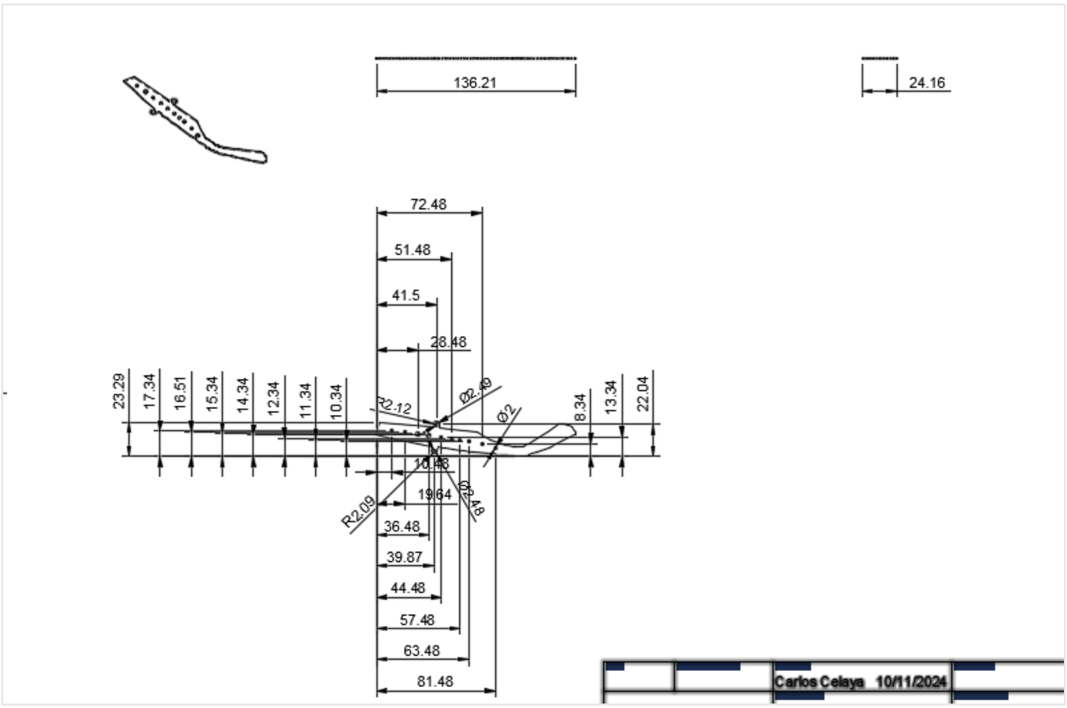


Figure 17. *Second design of the glasses (temple) with measurements*

- **ESP32-CAM Module Frame Redesign:** We also decided to redesign the frame for the ESP CAM module. After initial printing, we realized that the second design didn't fit the camera well, so we reverted to the first design. This adjustment allows for additional screw length and simplifies the attachment of nuts.

Here is our final design:

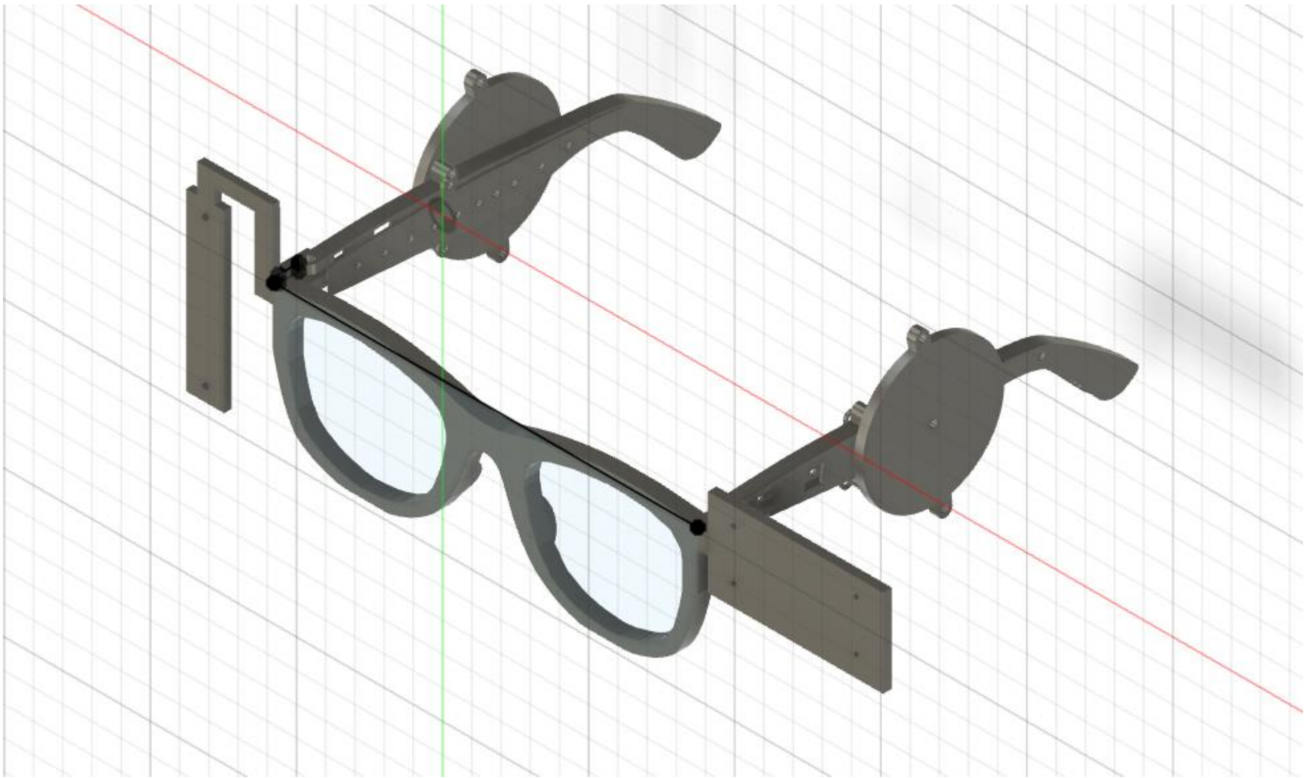


Figure 18. *Final design of the glasses seen in 3D*

Finally we need to see how the components ESP32 and battery are going to be carried. For the ESP 32 we plan to put some strap that joins the two ends of the branches and in that strap we will insert the ESP32 with a two pieces and just one design that will grip as we did with other components the strap and the ESP32. For the strap we have decided to choose the Nylon straps due to the strength of this material and its stress resistance. Also in this product it includes some adjustable buckles which makes it ideal for doing an ergonomic and easy to wear product.

Figure 19. *Strap used for the glasses*

The 3D part that included the design, is thought to work in the same way as the other components, it will wrap the ESP32. Needing to print it in 3D twice, and place the ESP32 in the middle:

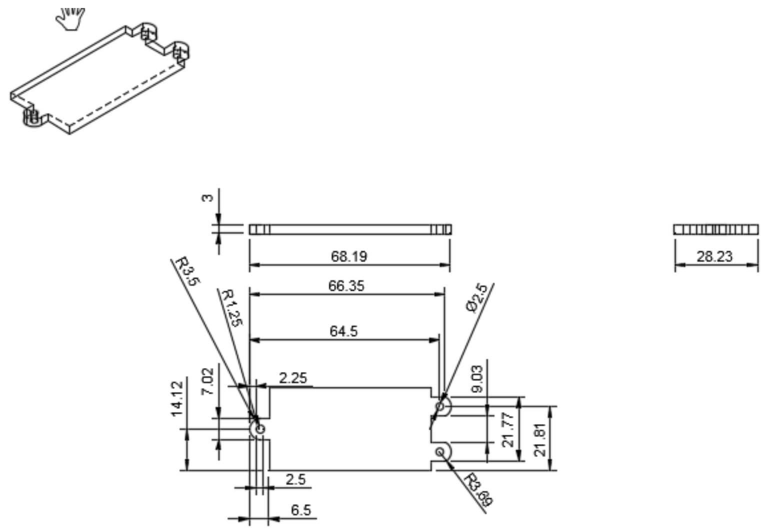


Figure 20. *Esp32 holder with measurements*

The battery on the other hand will have a strap covering it and it won't need a 3D printed body because we will use the strap to encapsulate it , and attach it to the hip.

Software used for the design

Designs considered

3.2.2 Electronic design: Component connection diagrams

For the comprehension of the connection between the ESP32 and the rest of components, first we have to have in mind the Pin Diagram of the ESP32 that we used:

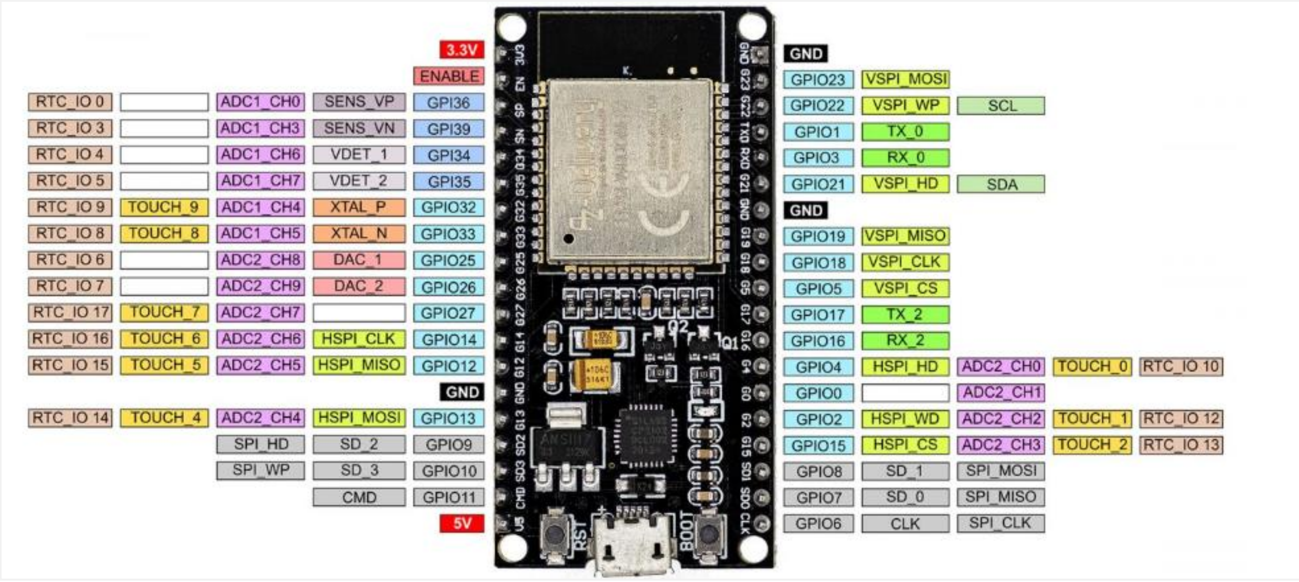


Figure 21. *Esp32 WROOM 32 NodeMCU module pin diagram*

Circuit 1: ESP32 + Ultrasound + Vibration motors

A) Ultrasound – Vibration motors

the component diagram of the ultrasound-vibration motors shown in Figure 2, illustrates the essential connections of the system:

- **Ultrasound sensor:**
  - **Trigger Pin:** It is connected to Pin19 on the ESP32. It sends a pulse to initiate the distance measurement, it is configured as an output.
  - **Echo Pin:** It is connected to Pin 18 on the ESP32. It receives the return signal to measure the time taken for the pulse to come back. It is configured as an input to allow the ESP32 to read the signal and calculate the distance.
  - **Vcc:** connected to the 5V to power the ultrasound
  - **GND:** connected to the ground Pin on the ESP32

- **Vibration motors:** They are organized in the same pins for the pairs of motors that activate at the same time:
  - **Motor pair 1:** Connected to Pin2 and ground on the ESP32, these two motors only activate when something is detected within 5 cm.
  - **Motor pair 2:** Connected to Pin 15 and ground, they activate when an object is detected within 10 cm (which means when something is detected within 5 cm they will also be activated).
  - **Motor pair 3:** We have to add the last pair of motors.
- **Switch:** It is connected with a 220 ohms resistance to Pin 34 and ground of the ESP32. It is configured as an input, and when the switch is pressed the pin detects changes from HIGH or LOW and turns off or on the system.

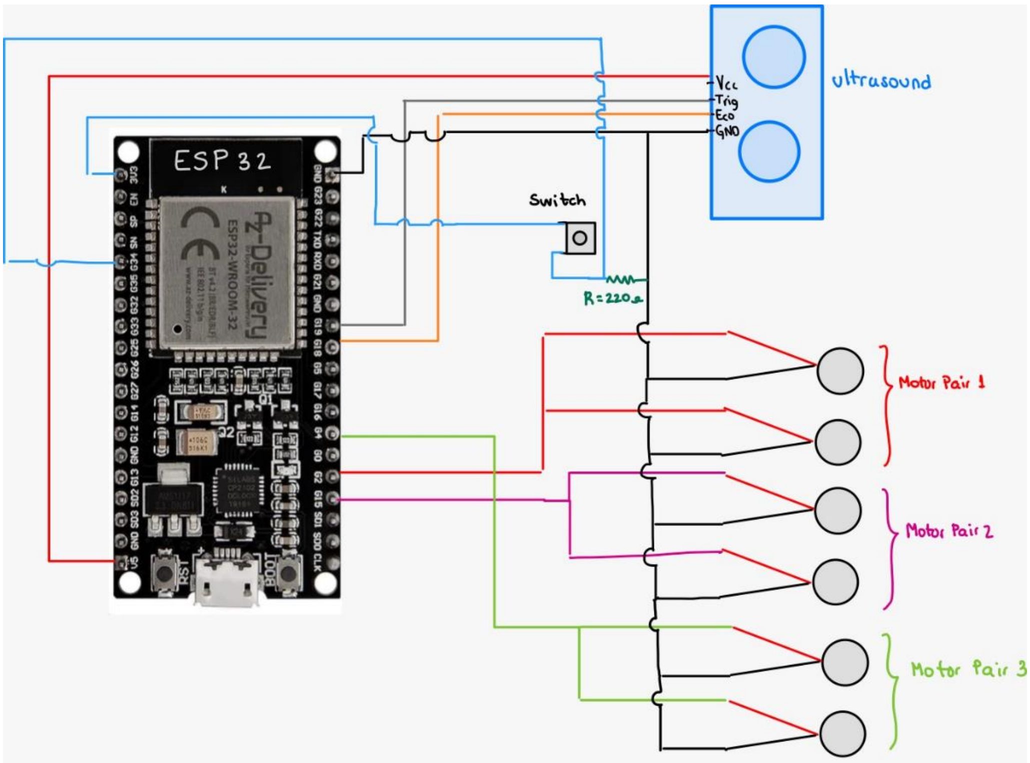


Figure 22. *Diagram of the ultrasound, vibration motors and switch*

Real life photos:

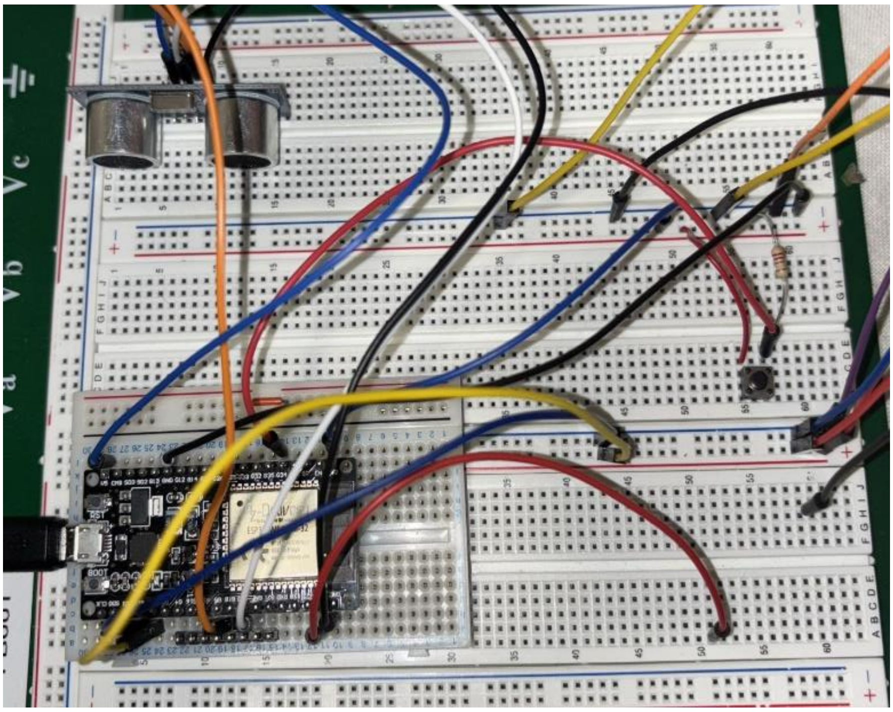


Figure 23. *Ultrasound-Vibration motor component image*

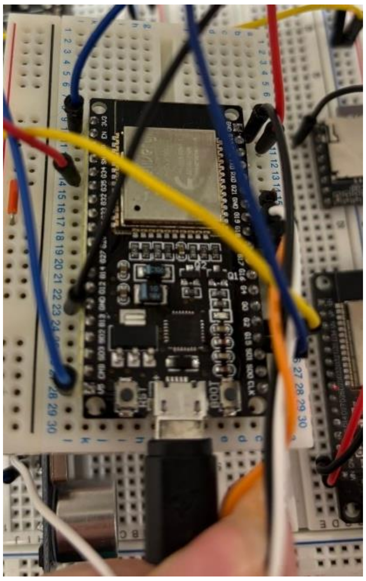


Figure 24: ESP32 Microcontroller

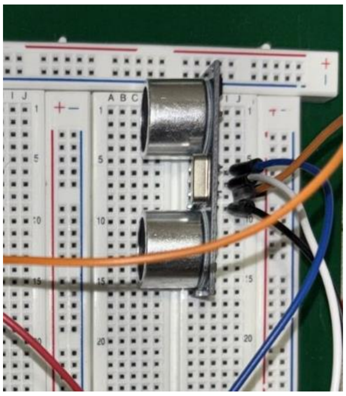


Figure25 : Ultrasound

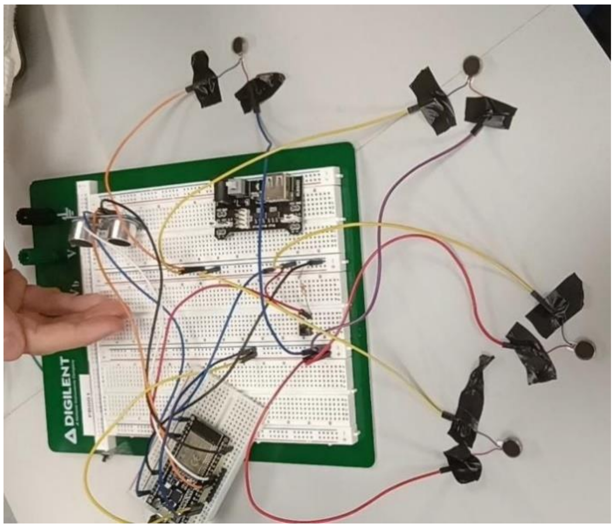
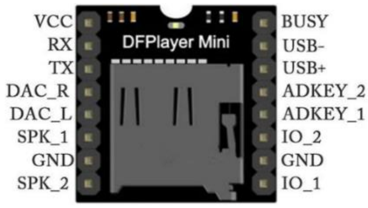


Figure 26: Vibration motors

Circuit 2: ESP32 + DFPlayer Mini MP3 Player + Speaker

B) DFPlayer mini mp3 player + Speaker
The Speaker has only 2 pins ('+' and '- '). However, the DFplayer has 16 pins, In Figure 'X' we can see the function and description of each of them:



No.	Pin	Description	Note
1	VCC	Input Voltage	DC 3.2-5.0V; Typical: DC4.2
2	RX	UART serial input	
3	TX	UART serial output	
4	DAC_R	Audio output right channel	Drive earphone and amplifier
5	DAC_L	Audio output left channel	Drive earphone and amplifier
6	SPK1	Speaker+	Drive speaker less than 3W
7	GND	Ground	Power Ground
8	SPK2	Speaker-	Drive speaker less than 3W
9	IO1	Trigger port 1	Short press to play previous(long press to decrease volume)
10	GND	Ground	Power Ground
11	IO2	Trigger port 2	Short press to play next(long press to increase volume)
12	ADKEY1	AD port 1	Trigger play first segment
13	ADKEY2	AD port 2	Trigger play fifth segment
14	USB+	USB+ DP	USB Port
15	USB-	USB- DM	USB Port
16	BUSY	Playing Status	Low means playing\High means no

Figure 27. *DFPlayer Mini MP3 Player Pin Diagram*

Looking at the pin diagrams of the ESP32 (**Figure**) and the DFPlayer (**Figure**), as well as the speaker's - and + pins, we obtain the general diagram of the pin connections between these 3 different components (Figure 'x').

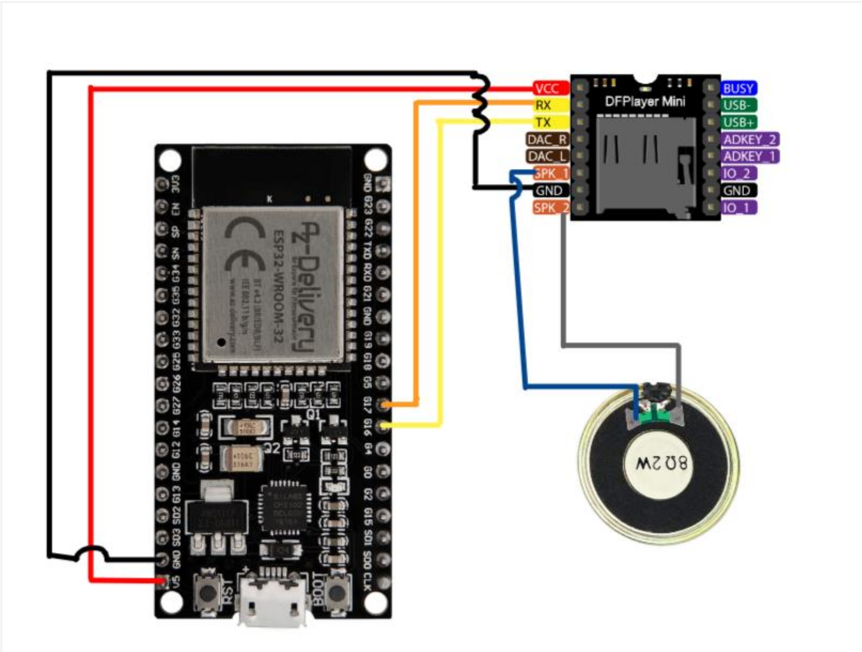


Figure 28. *ESP32 Wroom 32, DFPlayer and Speaker Connection Diagram*

<u>As we can see the pins used for this part where:</u>

- **ESP32, 4 Pins used**: VCC, GND, GPIO16 (RX2=pin16) and GPIO 17 (TX2=pin17)
- **DFPLayer, 6 Pins used**: VCC, GND, RX, TX, SPK1 and SPK2
- **Speaker, 2 Parts used**: '+' and '-' parts

<u>Description</u>

- **Electric connections**:

  VCC and GND are the terminals used to provide electrical power to the components in a circuit to establish common voltage reference for all components. **VCC** refers to the positive power terminal (input **voltage** pin). The **GND** refers to the ground terminal (0 V).

  For this circuit we connected

  - VCC pin of the ESP32 (5V) with the VCC pin of the DFplayer (pin 1), shown with a red wire in **Figure .**
  - GND pin of the ESP32 with the GND pin of the DFPlayer (pin 7), shown with a black wire in **Figure .**
- **Data connections**

  RX and TX or UART (Universal Asynchronous Receiver/Transmitter) **ports**: they are serial ports used for serial protocol communication, that means, that they are the **pins used for** data transmission between components. TX is the one which transmits the data and the RX the one who receives it. That means that the **connection** for component data transmission **has to be crossed**: the RX of one component has to be connected to the TX of the other component and vice versa. That is why the **RX of the DFPLayer is connected to the TX2 of the ESP32 and the DFPlayer TX is connected to the RX2 of the ESP32**.

Point: The ESP32 has 3 UART ports available:

- **UART 0** (TX0=GPIO3= pin 3; RX0=GPIO1= pin 1). *It is generally used to communicate with the ESP32 for programming and during reset/boot.*
- **UART 1** (TX1=GPIO 9; RX1=GPIO10). *Is unused and can be used for your projects. However, some boards use this port for SPI flash access, so you have to redefine other pins for its use.* That is exactly what happen with our board, as we can notice in the esp32 Pin Diagram (**Figure)** pins 9 and 10 are not accessible.
- **UART2** (TX2=GPIO17= pin 17; RX2=GPIO16= pin 16). *Is unused and can be used for your projects.*

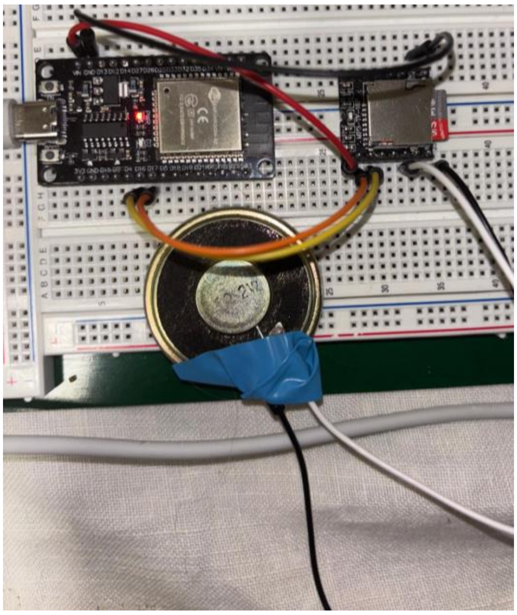
Taking into account all this information, we can notice that we used UART2 for this project.

****Note: The power supply for the ESP32 is 5V, but it distributes 3.3V to its components. Since the DFPlayer's logic control operates at 3.3V, we don't need to use resistors on the RX-TX2 and TX-RX2 connections to reduce the input voltage, as we would with Arduinos (which use 5V logic and supply 5V to components).***

- **Speaker connections**

DFPlayer Mini **SPK1** is **connected to** the **'+' (**positive**) terminal of** the **speaker.**
DFPlayer Mini **SPK2** is **connected to** the **'-'(**negative) **terminal of the speaker.**
These two pins (SPK1 and SPK2) provide the audio output from the DFPlayer Mini to the speaker, allowing it to play audio.

Real Life Photos of the Components Connection Circuit



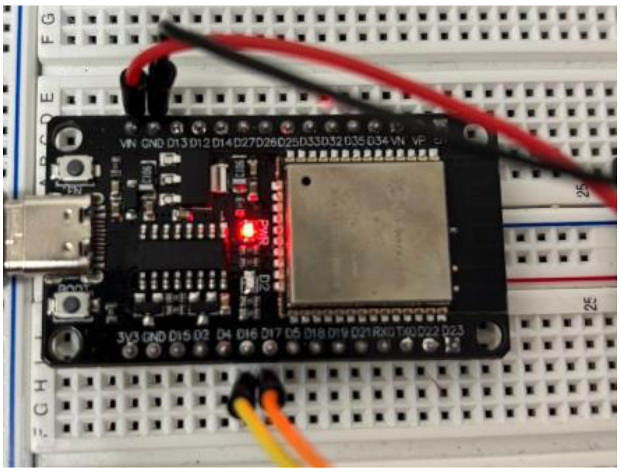


Figure 29. *4 pins ES32 used in the circuit*

Circuit 3: ESP32 CAM + FTDI Adaptor + Processor (Computer)

As the esp32 CAM does not have a USB port to upload the program to the camera and connect it to the battery, an FTDI adaptor can be found as a solution, being the main key points to take into account that the TX and RC pins are intertwined so that the TX port of the adaptor can communicate with the RX port of the esp32 CAM and vice versa, and a bridge between GPIO 0 and GND must be set to allow the programming of the esp32 cam, as shown below.

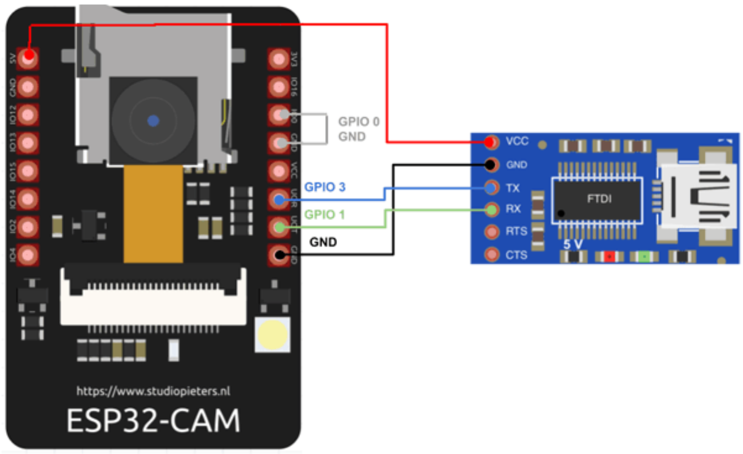


Figure 30. *Esp32CAM+FTDI adaptor connections*

However, the same connections were achieved by using the esp32-CAM-cm development board with micro-USB.

Figure 31. *Esp32-CAM with the esp32-CAM-cm development board with micro-USB*

C) GENERAL ARCHITECTURE

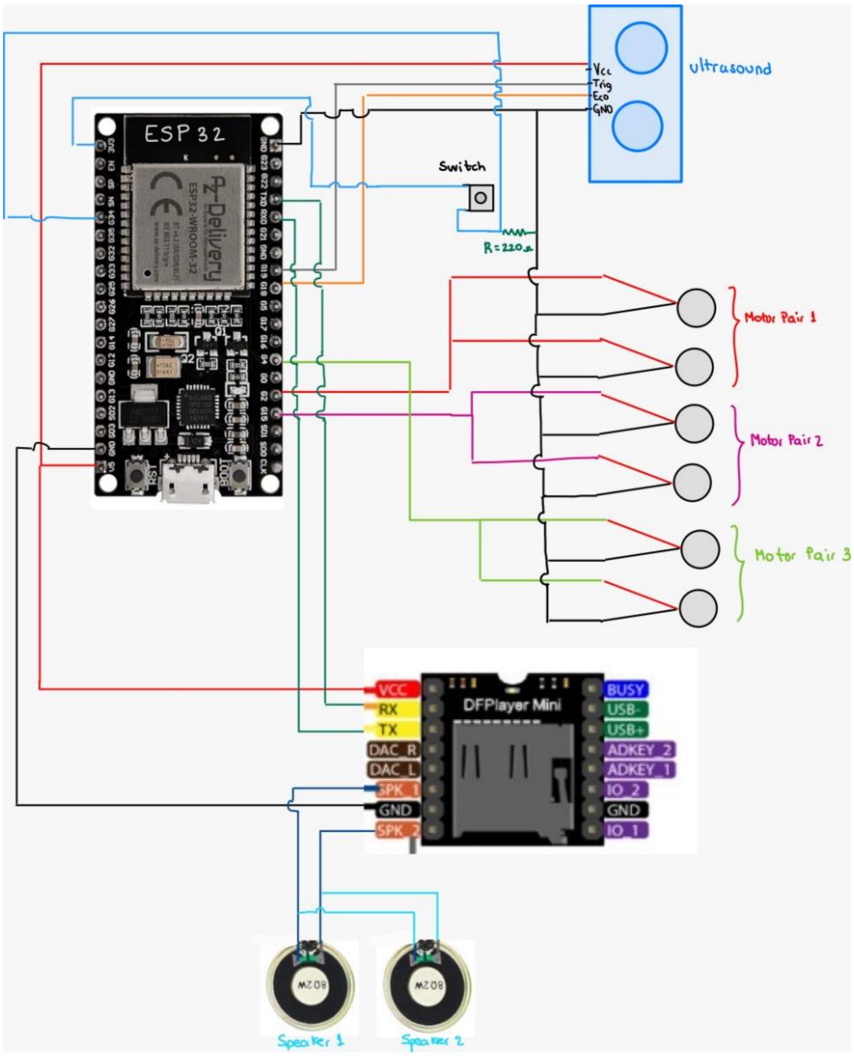


Figure 32. *General diagram*

3.2.3 Power supply

Battery: The system was first tested with the laptop, so we could see the distances in real time in the Serial Monitor from Arduino IDE. After making sure it worked correctly by testing which motors vibrated at different distances, we checked the system worked with the uploaded code with a power bank. We have also calculated the battery that consumes each component, and we have seen that the total consume is 740 mA, our battery has a capacity of 10000mAh so the total theoretical time that the device could work is 13.5h

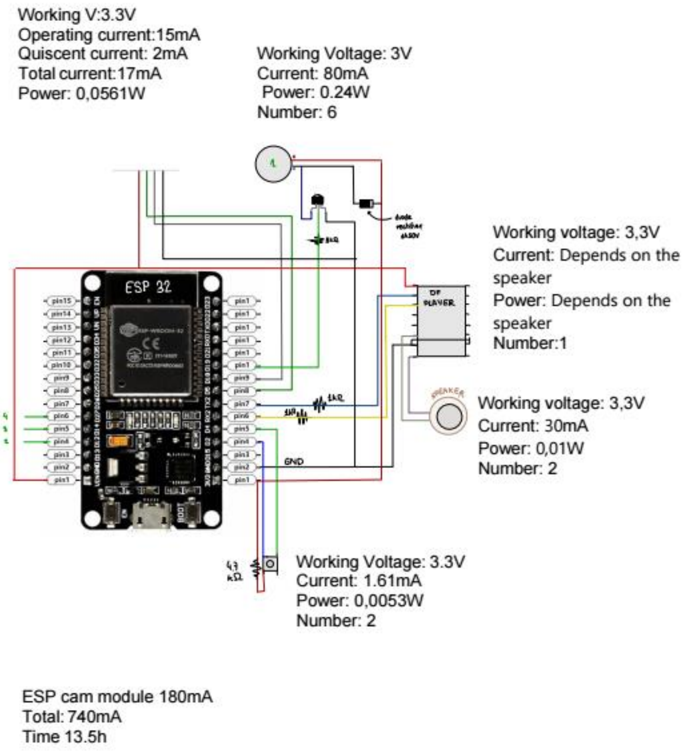


Figure 33. *Power consumption of each element*

3.3 SOFTWARE DESIGN

3.3.1 Arduino IDE

Circuit 1: ESP32 + Ultrasound + Vibration motors

A. **ESP32 connection**

Firstly, we connected the ESP32 to our laptop. However, we had some problems with this because the laptop did not recognize the ESP32, therefore we needed to install a CP2102 controller.

B. **Testing components**

- **Ultrasound sensor:** We started by connecting this device and making sure it detected distance correctly. However, we had some problems because, due to its position, it detected the protoboard, so we had to lif it up.
- **Vibration motors:** We started by connecting 1 vibration motor configured at a distance of less than 5 centimeters. Then, we added another one in a different pin. Since we wanted to add 6 motors, adding each one in one pin would use too many pins that we could need for other components. Therefore, as the motors vibrate in pairs, we used one pin for each pair of motors.
- **Switch:** Initially, the switch was configured to require a continuous pressing to stop the system from working. However, this configuration wasn't practical, so we changed it to turning the system on or off with one single press.

This setup represents the initial design. We are considering adding a diode in the motor to protect the system.

C. Arduino Code

The final code used for two pairs of vibration motors (we plan to add one more pair, and decide the optimal distances for each pair) and for the on and off switch configuration is the following:

```
// Pins for the ultrasonic sensor and vibration motors
int DISTANCIA = 0; // Variable to store the measured distance
int pinMotor1 = 2; // Pin for the first group of vibration motors
int pinMotor2 = 15; // Pin for the second group of vibration motors
int pinMotor3= 4; // Pin for the third group of vibration motors
int pinEco = 18; // Pin to receive the echo signal from the ultrasonic sensor
int pinTrig = 19; // Pin to send the trigger signal to the ultrasonic sensor
int pinInterruptor = 34; // Pin for the switch to turn the system on/off

bool sistemaActivo = false; // System state: false = off, true = on
bool ultimoEstadoInterruptor = HIGH; // Last state of the switch

// Function to measure distance using the ultrasonic sensor
long readUltrasonicDistance(int triggerPin, int echoPin) {
  pinMode(triggerPin, OUTPUT); // Set trigger pin as output
  digitalWrite(triggerPin, LOW); // Set trigger pin to LOW
  delayMicroseconds(2); // Small delay before sending pulse
  digitalWrite(triggerPin, HIGH); // Set trigger pin to HIGH to send a pulse
  delayMicroseconds(10); // Keep the pulse for 10 microseconds
  digitalWrite(triggerPin, LOW); // Set trigger pin back to LOW
  pinMode(echoPin, INPUT); // Set echo pin as input
  return pulseIn(echoPin, HIGH); // Measure the time for the pulse to return
}

void setup() {
  Serial.begin(115200); // Initialize serial communication for debugging

  // Configure pins
  pinMode(pinMotor1, OUTPUT); // Set motor1 pin as output
  pinMode(pinMotor2, OUTPUT); // Set motor2 pin as output
  pinMode(pinMotor3, OUTPUT); // Set motor3 pin as output
  pinMode(pinInterruptor, INPUT_PULLUP); // Configure switch pin as input with pull-up resistor

  Serial.println("System ready. Press the switch to turn on/off the system.");
}

void loop() {
  // Read the current state of the switch
  int estadoInterruptor = digitalRead(pinInterruptor);
```

```
    // Check if the switch state has changed from HIGH to LOW
    if (estadoInterruptor == LOW && ultimoEstadoInterruptor == HIGH) {
      sistemaActivo = !sistemaActivo; // Toggle the system state (on/off)
      delay(200); // Small delay to prevent bouncing (debounce)
    }

    // Update the last switch state
    ultimoEstadoInterruptor = estadoInterruptor;

    // Run the system if it’s active
    if (sistemaActivo) {
      // Measure the distance using the ultrasonic sensor
      DISTANCIA = 0.01723 * readUltrasonicDistance(pinTrig, pinEco); // Convert time to distance in cm
      Serial.print("Distance: ");
      Serial.println(DISTANCIA);

      // Activate motors based on distance
      if (DISTANCIA < 5) {
        digitalWrite(pinMotor1, HIGH); // Turn on motor group 1
        digitalWrite(pinMotor2, HIGH); // Turn on motor group 2
        digitalWrite(pinMotor3, HIGH); // Turn on motor group 3
      } else if (DISTANCIA < 10) {
        digitalWrite(pinMotor1, LOW); // Turn off motor group 1
        digitalWrite(pinMotor2, HIGH); // Keep motor group 2 on
        digitalWrite(pinMotor3, HIGH); // Keep motor group 3 on
      } else if (DISTANCIA < 20) {
        digitalWrite(pinMotor1, LOW); // Turn off motor group 1
        digitalWrite(pinMotor2, LOW); // Keep motor group 2 on
        digitalWrite(pinMotor3, HIGH); // Keep motor group 3 on
      } else {
        digitalWrite(pinMotor1, LOW); // Turn off motor group 1
        digitalWrite(pinMotor2, LOW); // Turn off motor group 2
        digitalWrite(pinMotor3, LOW); // Turn off motor group 3
      }
    } else {
      // If the system is off, deactivate all motors
      digitalWrite(pinMotor1, LOW);
      digitalWrite(pinMotor2, LOW);
      digitalWrite(pinMotor3, LOW);
      Serial.println("System off. Press the switch to turn on.");
    }

    delay(500); // Wait half a second before the next measurement
  }
```

D. **Switch programming**

The switch is configured as an input with a pull-up resistor, Its pin is in a HIGH state when the button is not pressed; and when it is pressed it pulls it to LOW, which turns off the system.

There is also a delay of 200 milliseconds to prevent bouncing after detecting a button press. Bouncing happens when the switch oscillates between HIGH and LOW states, which can cause multiple state changes.

E. **Vibration motors programming:**

The motors are controlled by sending LOW and HIGH signals from the ESP32 to the specific pins. When a HIGH signal is sent to a motor pin the connected motors vibrate, and the same with a LOW signal.

- If distance is less than 5 cm, both motor pins are set to HIGH, which means all motors will vibrate
  - If distance is between 5 and 10 cm.
    - The first pair of motors (connected to pin 2) is set to LOW (they will not vibrate).
    - The second pair of motors (connected to pin 15) remain at HIGH (they will vibrate.
    - There is a third pair of motors to be configured.
  - In any other case (if distance is over 10 cm) both pair of motors are set to LOW.

Circuit 2: ESP32 + DFPLAYER MINI + SPEAKER

For this part we want to achieve the DFPlayer to reproduce a specific audio depending of the type of the detection of the YOLO algorithm.

Progress So Far:

We've managed to get the speaker to play sound. However, we still need to decide the way we are going to associate the detection results with the corresponding audio files and implement the programming logic for the DFPlayer to play the audio correctly (we have an idea, will be presented later but we need to specify). Here are some key points to consider for that part:

- Each time audio information is received, it's stored in a variable (the last command from the computer). This variable is updated with new information from the computer while sound is playing. It's possible that multiple audio commands are queued, with the most recent one being the highest priority.
- It's essential to identify when the last audio command has been received and know when to stop playing to avoid continuously repeating the last audio. Another variable (set to 0 or 1) will help manage this.
- This setup ensures that the latest audio commands are played without interruptions or crashes. The audio output must be reliable.

Setup and Logic

The videos we found on programming the DFPlayer to play sound consistently used the <DFRobotDFPlayerMini.h> library, but this library did not work with our module. It appears our DFPlayer module is not the original model by DFRobot but instead uses a different chip (an AAXXXX from Jieli (JL)), so that library was incompatible. We used a library called <DFPlayerMini_Fast.h> instead.
The examples also used the <SoftwareSerial.H> library, which only works with Arduino, not with the ESP32, since the ESP32 has hardware serial ports (which can be configured using libraries like HardwareSerial, though we didn't need them here).
We also learned that the ESP32 has three serial ports, and the ones that worked for us were UART2 (TX2 and RX2, corresponding to pins 17 and 16, respectively). This was crucial for initializing serial communication with Serial2 (the "2" was important here).
We had to adjust some functions as well. For example, DFPlayer.next (to play the next MP3 track) didn't exist in the Fast library, so we modified it to use currentTrack.
Additionally, we needed to purchase an SD card under 32 GB so the DFPlayer could read it and ensure it was formatted to FAT32.

```
#include <DFPlayerMini_Fast.h>  // Librería alternativa

// Definir pines de comunicación
#define RXD2 16
#define TXD2 17

// Crear un objeto DFPlayer
DFPlayerMini_Fast myDFPlayer;

int currentTrack = 1;  // Pista inicial (1 corresponde a "0001.mp3")

void setup() {
  // Configuración del monitor serial para depuración
  Serial.begin(115200);

  // Inicializar el puerto serial de comunicación con DFPlayer
  Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);  // Usa el puerto Serial2 del ESP32

   // Imprimir los pines RXD2 y TXD2
  Serial.println("Serial Txd está en el pin: " + String(TXD2));
  Serial.println("Serial Rxd está en el pin: " + String(RXD2));


  // Intentar inicializar DFPlayer
  if (!myDFPlayer.begin(Serial2)) {
    Serial.println("Error al inicializar DFPlayer. Verifica la conexión.");
    while (true);  // Se queda en un bucle infinito si hay error
  }

  Serial.println("DFPlayer inicializado correctamente.");

  // Intentar leer el primer archivo de la tarjeta SD
  myDFPlayer.play(currentTrack);  // Reproduce la primera pista ("0001.mp3")

  // Verificar si se está reproduciendo correctamente
  if (myDFPlayer.isPlaying()) {
    Serial.println("Tarjeta SD detectada y funcionando correctamente.");
  } else {
    Serial.println("Error al reproducir la música. Verifica la tarjeta SD.");
    while (true);  // Bucle infinito si no se reproduce
  }

  // Configurar el volumen (0-30)
  myDFPlayer.volume(20);
}

void loop() {
  // Espera 9 segundos antes de reproducir la siguiente pista
  delay(9000);  // 9 segundos de espera

  // Incrementar el número de pista y reproducirla
  currentTrack++;
  myDFPlayer.play(currentTrack);  // Reproduce la siguiente pista

  // Si hemos alcanzado el final de las pistas, reiniciar al principio
  if (currentTrack > 10) {  // Suponiendo que tienes 10 pistas, ajusta según el número de pistas
    currentTrack = 1;  // Volver a la primera pista
  }
}
```

Taking all of this into account, here is the final code:

Circuit 3: ESP32 CAM

First, a YOLO v7 was tested by filming videos to see if the objects could be recognized well. The result is shown on Figure

Figure 34. *Testing of yolo v7 with a video*

4.1.3 ESP32 CAM

In terms of the software used for the esp32 CAM, the main key points that had to be considered were the following:

- Wi-Fi connection: one of the main reasons the esp32 CAM was chosen was due to the integration of a module capable to transmit via wifi the data needed, in this case being said data a video on stream. A hotspot is used, being said hotspot produced from the phone, the reasons why this was chosen being:

1. Network restrictions and security policies

- Trials done with the Wi-Fi from the university have proven to result in errors. This can mostly be due to the network policy, which includes firewalls, restricted IP addresses, as well as port blocking to prevent unauthorized devices from connecting, being the computer used an unauthorized device. Another reason for the failed trials can be explained from the isolation between the devices that are on university networks, which prevents one from access the ESP32-CAM's stream from another device. IT approval would therefore be needed in order to use the university's Wi-Fi, which can be a restrictive and lengthy process. In the case of the hotspot there are no network restrictions or security policies.

2. IP Address Management

- On university networks dynamic IP allocation is often used, which results in different IP addresses for the esp32-CAM each time the connection occurs, posing a challenge to maintain a stable connection to the camera as one would have to find the new IP with every new connection, and then change it in the code uploaded to both esp32 and python code from the computer. The hotspot on the other hand provides a stable and specific IP address.

3. Data Privacy and Security

- The stream from the camera could potentially be accessible to other users of the same network, raising privacy concerns, especially if it is not a secured stream (which it is not). By using a hotspot the phone user of the group can control who accesses the network and limits the stream visibility.

4. Signal Strength and Reliability

- When it comes to the signal strength and reliability, public wifi is prone to congestion and its signal strength may vary depending on factors that cannot be controlled such as the time of day or number of students connected. The hotspot can ensure a stronger and more consistent signal, and the tests done can more effectively offer an idea of the strength of the signal on the day of the presentation.

5. Simpler Setup and Testing

- Setting up a direct connection to a mobile hotspot is faster and more straightforward, and troubleshooting is both simpler and quicker.

For the setup of the hotspot within the Arduino camera code the ssid and the corresponding password were set.

- Storage of the video: the video from the Esp32 CAM was set to send the video to a server, with a name derived from the IP. This video can then be obtained from the server using PyCharm.

- Changes to the code that had to be done: From the initial code obtained from the examples of Arduino UNO, some of the main changes consisted on the defining the specific ssid and password for the hotspot, defining the type of camera being used, finding and changing the local IP, and finding the IP of just the video that is streaming, as the initial code provided the IP to a server that did not have a video that could be recognized by the program run in PyCharm.

- Connection of the computer to the IP: It is of importance to remark that in order to access the IP the camera is sending the video to the computer must also be connected to the same hotspot.

- Modules that have to be installed: Arduino IDE does not directly recognize the esp32 or esp32 CAM, a software module had to be installed in the Arduino IDE.

- Antenna use and trials: For the initial trials an antenna was used to ensure the connection between the esp32 CAM and the hotspot was made. Even though tests have been conducted to ensure the antenna is not currently needed, being able to transmit the video without the connection to the antenna, further tests need to be done.

- Face recognition: one option that was also considered was a direct recognition of the faces from the esp32 CAM, possible with the Arduino IDE, however, the function of face recognition is not available for esp32 boards as it takes up from 15seconds to process a single frame, and is only possible with PSRAM. This is a function has however been replaced by the processing of the image and video with the yolov8 implemented in Pycharm, and explained within the Python section.

- Uploading the program: In order to upload the program to the esp32 CAM, a programming module ESP32-CAM-CM with micro-USB included in the order for the camera was used. However, as the esp32 CAM was received first, the alternative used was an Arduino UNO. The first component used was however an Arduino Mega, which was unsuccessful due to the fact that it did not have the required module that the Arduino UNO did have. After trying to install an alternative (software solution) which due to it being outdated did not work, an Arduino UNO was used. Once the specific programming module from the ESP32-CAM was received it was inserted into the esp32 CAM and used to upload the newer programs as well as to connect the camera to the power supply via a usb cable.

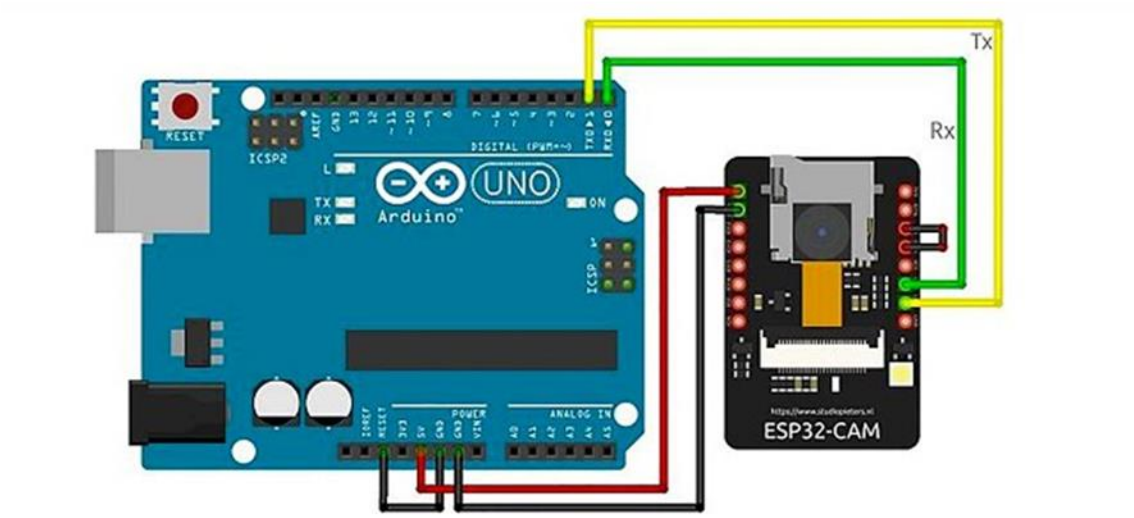


Figure 35. *Setup used to upload the program to the esp32-CAM*

The final files used consist on two .h files (one to define the pins of the camera and another one for the index), 1 .cpp file and 1 .ino file (CameraWebServer.ino), which is the file where the changes were implemented.

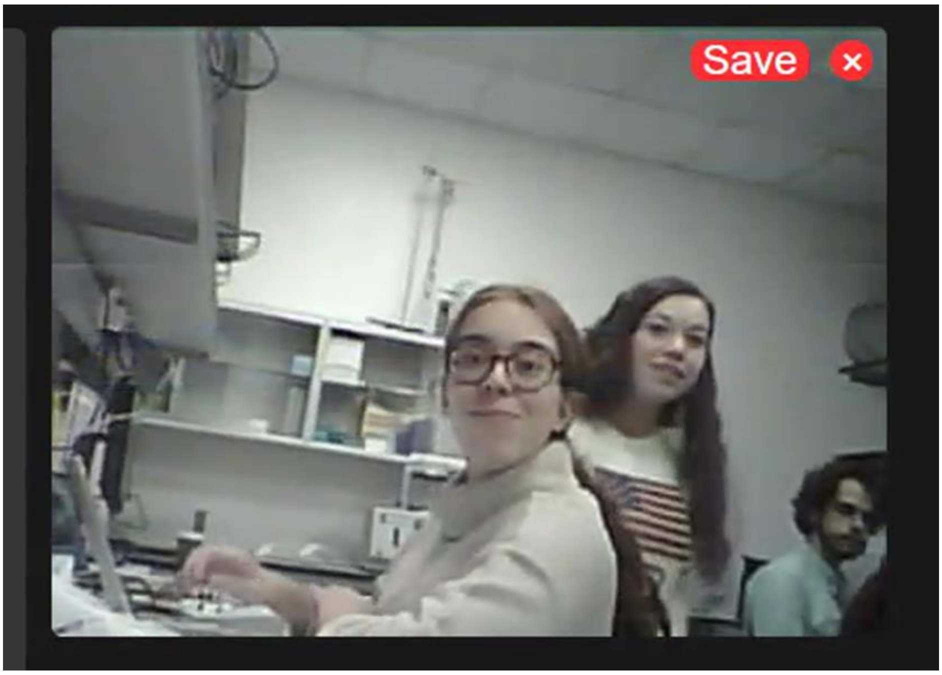


Figure 36. *Video of the esp32 CAM seen when accessing the server*

4.2 Python

4.2.1 PyCharm

The main musts and wants that we wanted the artificial intelligence to have in order to detect specific objects were:

- Musts:
  - It shall detect cars and people that are in front of the user and at the sides.
  - It shall count how many objects are detected.
  - It shall detect the objects with slight movement in the image.
- Wants:
  - The glasses may detect cars and people that are very far away.
  - They may recognize other objects from daily life (cats, dogs…).
  - They may detect objects even when the user is moving a lot, and the image is not completely clear.

The final program uses both YOLO object detection as well as DeepSORT tracking to detect, track and announce the positions of people and cars (more objects can be added) in a video feed from an ESP32-CAM. It also plays a message whenever a detected person is in a specific direction relative to the camera view.

The steps followed to obtain the final program are the following:

1. Initial Setup for Object Detection
   - Used as a base the necessary code for the implementation of a YOLO v8 with the names of the objects that can be detected, the yolov8n.pt and the basic code to implement yolo v8 (with the creation of a VideoCapture object(cap) to read the feed and the code to read the classes from coco.names and assign random colors using numpy and processing each frame to detect objects and draw bourning boxes around them)
   - Imported essential libraries (tkinter, PIL, cv2, and YOLO from ultralytics for object detection)

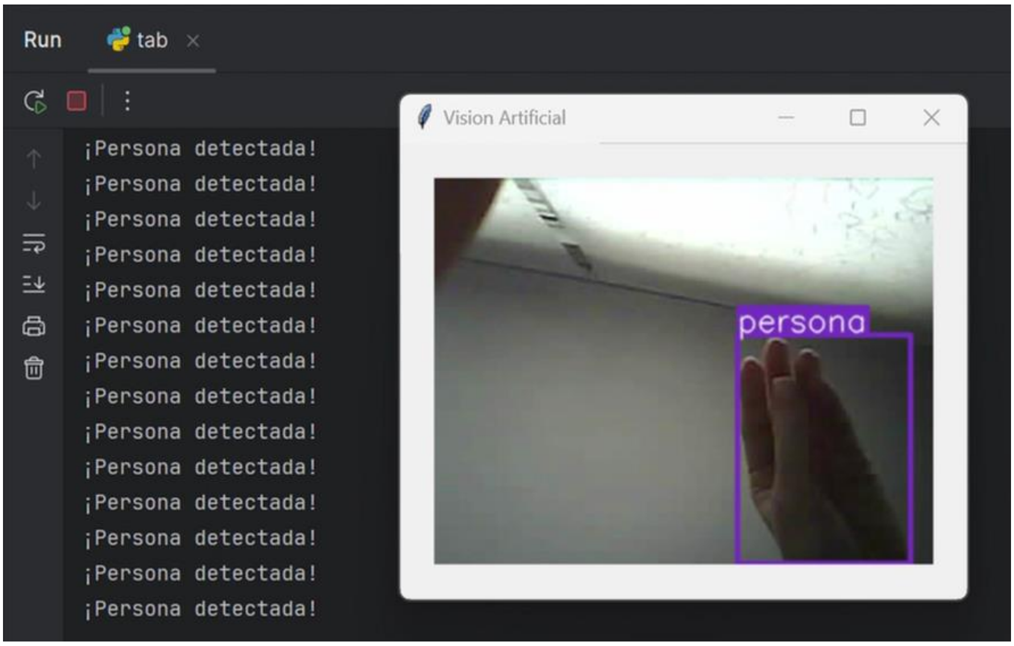


Figure 37. *Initial functioning of the yolo v8 implementation*

2. Error Handling and Debugging (after testing multiple errors were encountered)

- Increased the time before obtaining a timeout while connecting to the required server.
- Ensured that label.configure was correctly called with an actual ImageTk.PhotoImage object
- Annotator.box_label required color to be a hashable type but it was an np.ndarray. To solve it COLORS was converted to a tuple by wrapping it in tuple() to ensure it was hashable when passed to box_label()
- The variable r was accessed before assignment in some cases. To solve it r was properly assigned within the loop before being used for labelling.

3. Detection of additional objects

- The detection of people was expanded to include cars, and can be expanded to include additional objects such as bicycles, phones and others.

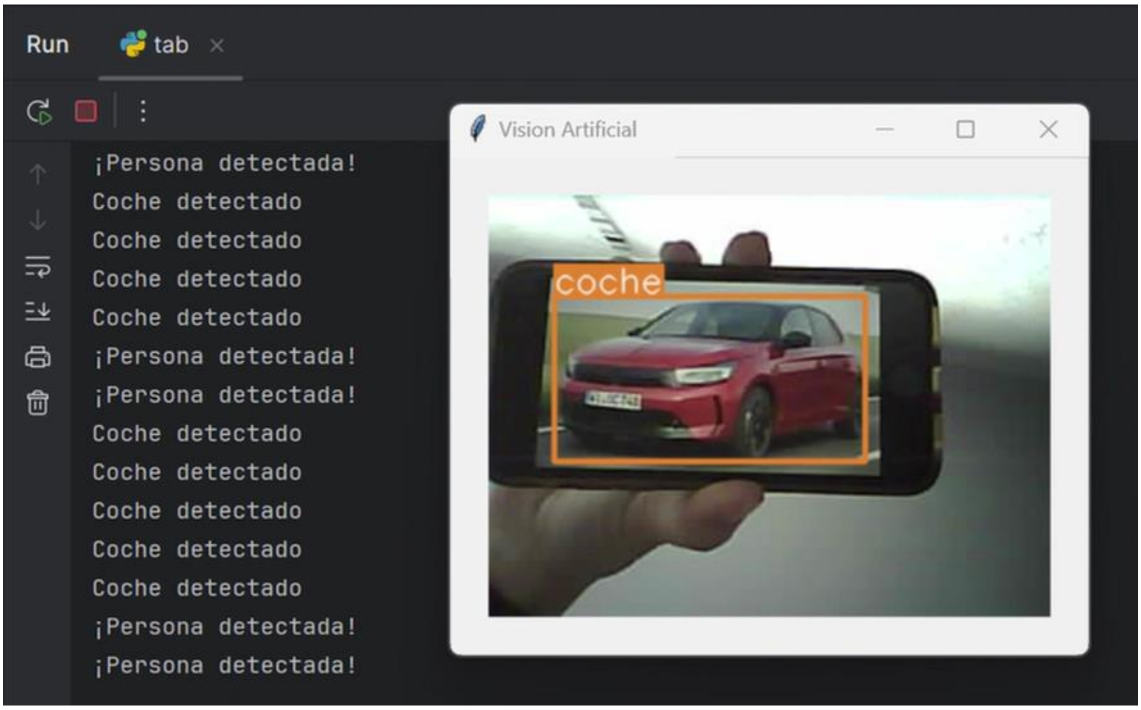


Figure 38. *Implementation of the detection of cars*

4. Integration of DeepSORT for Object Tracking

- The objective was to add object tracking to correctly count the number of each object in the frame, as without it the objects kept appearing and disappearing. In addition, this function allows the monitorization of object paths in real-time. The steps where as follows:

  - Instalation of DeepSORT dependencies: DeepSORT as well as any additional dependencies were installed to integrate tracking with YOLOv8. The initialization for DeepSORT was then added to allow for persistent object tracking across frames

  - Formatted detections for DeepSORT by creating a detections list that stores each detected object's bounding box, confidence and class information

  - Applied processing for the tracking results to maintain object IDs over time, which enabled the program to track unique individuals and vehicles as well as their locations across frames.

5. Additional enhancements

- Global variables to track changes: the code was enhanced to detect when the person or vehicle counts change, in order to allow a more dynamic feedback

- Audio message function: a placeholder function was integrated, which can be expanded to trigger audio messages that is used for the esp32.

6. Optimizations

- Addition of global variables to maintain the last known count of detected objects across frames

- Use of Tk.after() for a smoother regular frame updates to ensure that the real-time display is responsive

The key changes made to the initial base code adapted basic object detection with YOLO to a more advanced setup that includes DeepSORT tracking, directional tracking and additional interaction capabilities, the changes made have increased responsiveness and enhanced the program to the needs required.

Figure 39. *Output when implementing DeepSORT to the yolo v8*

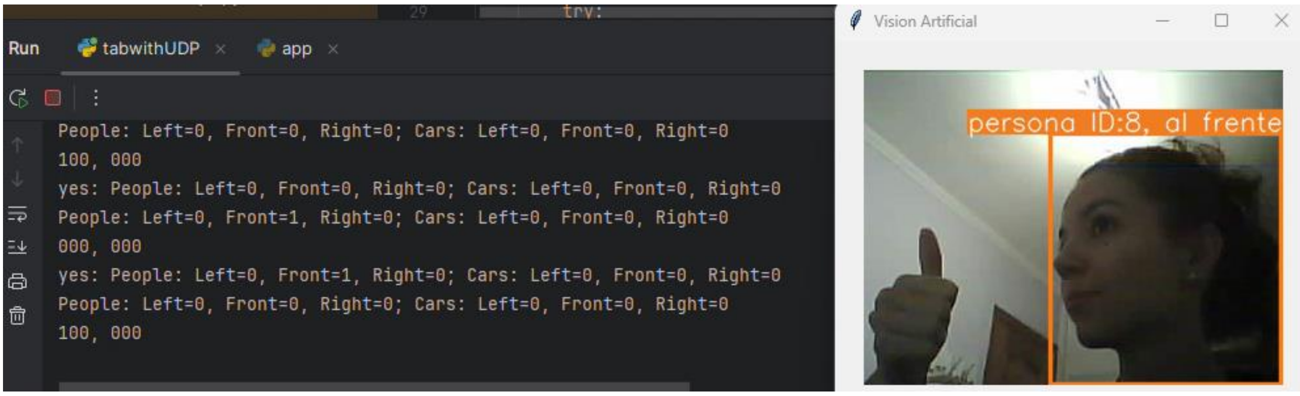


Figure 40. *Output after including the DeepSORT tracker and adding the positions*

3.3.2 Communication between the different components

1) WIFI CONNECTION

The communication between the ESP32-Cam, the computer (running as a processor), and the ESP32 WROOM-32 is achieved by creating a Wi-Fi hotspot using a phone. As shown in the figure, it's essential to set up a local network that connects these three components. We accomplish this by connecting the ESP32-Cam, the computer, and the ESP32 WROOM-32 to the phone's Wi-Fi network.
Connecting the computer to the Wi-Fi network is straightforward; you just need to select the phone's Wi-Fi network from your computer's network settings. However, connecting the two ESP32 devices requires code that instructs them to join the same Wi-Fi network:

```
#include <WiFi.h>  // Needed to connect and set up an access point

// SSID and password that are going to be used for the Access Point you will create
const char* ssid = "PHONE";  // Replace with your SSID
const char* password = "PASSWORD";   // Replace with your Wi-Fi password

void setup() {
  Serial.begin(115200);  // Initialize serial port for debugging

  // Connecting to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);  // Wait for 1 second before checking the connection again
    Serial.println("Establishing connection to Wi-Fi...");
  }

  Serial.println("Wi-Fi connected");

  // Print the ESP32 IP address once connected to the Wi-Fi network
  Serial.print("Connected to the network with IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  // Add your repetitive code here if needed
}
```

We included code lines to print the IP address of the ESP32, which is essential for properly establishing the communication protocols that will be explained in the following section.

2) COMMUNICATIONS PROTOCOLS

A) HTTP: ESP32-CAM – PROCESSOR
B) UDP: PROCESSOR – ESP32 -DFPLAYER-SPEAKER

Objective

Send the output/result of detection (number of people or cars) from the YOLO algorithm to the ESP32, so that it can play the result through a speaker.
Notes: The YOLO algorithm is running on a Python program in PyCharm. Therefore, a connection needs to be established between the Python program and the ESP32. In both codes, the chosen communication protocol's connection configuration needs to be set up: in Python to send data and on the ESP32 to receive it. When the ESP32 receives the result with the number of people or cars, it will be programmed to play the corresponding audio through a speaker, based on the detection's count and object type.

That is why, it is crucial to capture the detection result (number of people or cars) and determine how send that information to the ESP32, since this data will determine the audio to be played and serve as the 'trigger' for the speaker.

CONNECTING THE YOLO OUTPUT TO THE ESP32 TO ACTIVATE COMPONENTS

The first steps were:

- Establishing and deciding which communication protocol is appropriate for the connection between the YOLO program and the ESP32.

- Configuring the communication protocol so that the YOLO program sends data and the ESP32 receives it to activate the speaker and the corresponding audio.

When looking for communication protocols, HTTP and WebSockets appear the most. Although these protocols appeared in most videos, they did not fit what we needed. After several failed attempts, we found a way to do it through the UDP protocol.

The first step was to use the UDP protocol only with the computer and esp32 connected to a buzzer. In order to do so the esp32 was first connected to both the buzzer and computer via a cable and the buzzer was activated using the esp32 without the UDP protocol, to ensure the connexion between components was optimal.

The next step consisted on using the UDP protocol with the esp32 already connected to the MP3 player.

Once the previous steps were successful, the final step was using the UDP protocol to send to the esp32 the outputs from the YOLO and DeepSORT program.

In order to do so a logic for the UDP signal was established, being the format used for each message ”123, 456”, two numbers of three digits each separated by a coma.

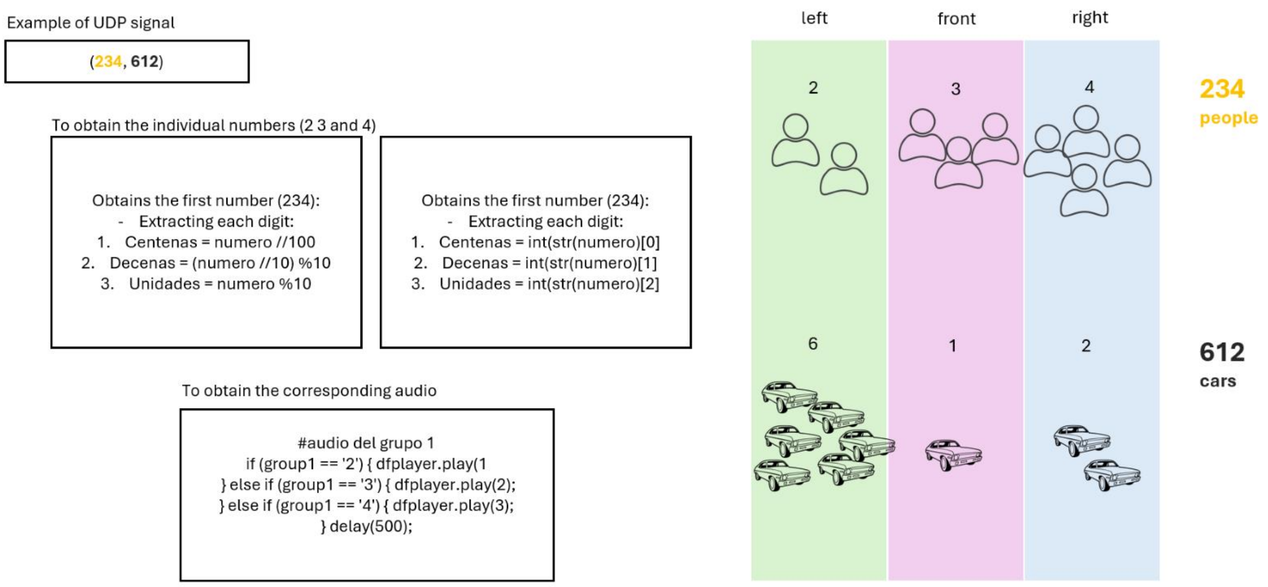


Figure 41. *Logic followed for the UDP message sent*

For the first trials what was used were pre-recorded audios with ”one person on the right”, ”one person in front of you”, ”one person on the left” and so on for both people and cars.

Format of the signal sent: (num1, num2)
mode = 1
num1 = 123
num2 = 123

mode = type of logic followed for the audio
num1= people
num2 = cars

Digit 1 → number of instances on the left
Digit 2 → number of instances in front
Digit 3 → number of instances on the right

Audio output:
3 audio groups→ audios with numbers (group 1), audios with the objects (group 2), audios with the position (group 3)

Figure 42. *Logic followed by the UDP protocol established*

Once the count exceeds a certain number such as 8 of the same objects with the same position, it is programmed to say "multiple" instead of the exact number of objects.

The full integration of the camera, processing and audio output has been tested and proved to be feasible.

3.4 USER INTERFACE DESIGN

3.4.1 Application design

For the creation of the app, an initial design was made using Canva. The main idea was to offer the caregivers of the user the possibility to aid the person when there is an emergency. The caregiver can access the video when given permission through a signal emitted by sending a UDP message from the esp32 when a button is pressed. The aim is to offer more guidance, get a better understanding of the situation the legally blind user is facing, and allowing the caregiver to give additional directions. In order to do so the main functions of the app that were described were an emergency button, button to pair the glasses, and access to notifications, where one can access the time at which the button was pressed. The first diagram can be seen below.

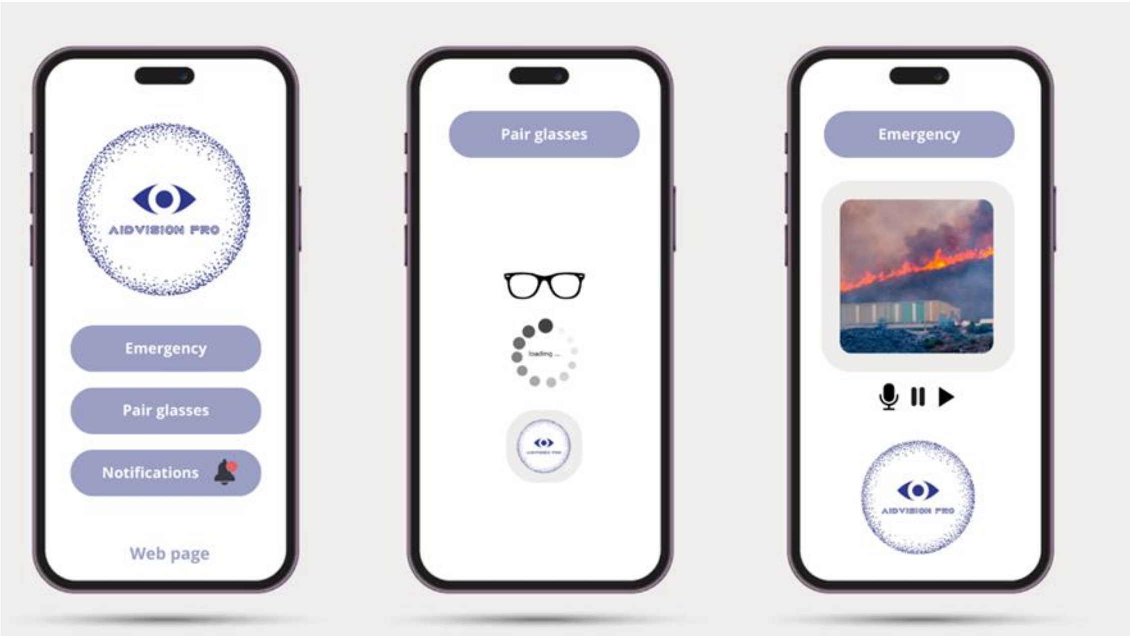


Figure 43. *Initial app design of the application in Canva*

Once the aim was established and the design defined, the program was coded using Python in Pycharm from scratch. To create the program the following main considerations were taken into account:

1. Connectivity: defining the IP and creating the socket to obtain the signal from the video
2. Definition of the Graphic interface used: use of Tkinter within PyCharm to create the graphic interface by creating a new window where the information is displayed
3. Creation of the main page: adding the 3 main buttons (emergency, pair glasses and notifications), setting the logo as a photo in the upper part of the page
4. Displaying the video: the video was set to appear only when the emergency button was pressed
5. Return button: As it has to be possible to return to the main page, a “Go Back” button was included and used to return from the notifications, pair glasses and emergency options. Within this button it was defined the components that needed to reappear and the ones that would disappear from the screen
6. Use of functions defined in Python at the beginning of the code and use of comments (#) to facilitate the implementation of changes in the code.
7. Additional button “Simulate Emergency”: as at the time of the creation of the application the esp32 was being used for other purposes within the project, an additional button was created to simulate the receival of the corresponding UDP signal to test the program and allow test runs for debugging

8. Speaker button: When in the emergency page, the option to send an audio to the esp32 and reproduce it can be done by using the speaker button, which offers 3 different options for the audios that can be sent: “Everything is okay”, “I called 112” and “Don’t worry, be happy”
9. Knowing when the emergency button was pressed: For the caregiver to know when the emergency happened, a register is created with the time at which each signal was received.
10. Change in appearance when receiving the signal: when the emergency signal is received a dynamic message is displayed in the screen until a button is pressed, and the emergency button turns red until the user accesses the video, to alert the user of the app. In addition, a sound alarm similar to a ring tone is played.

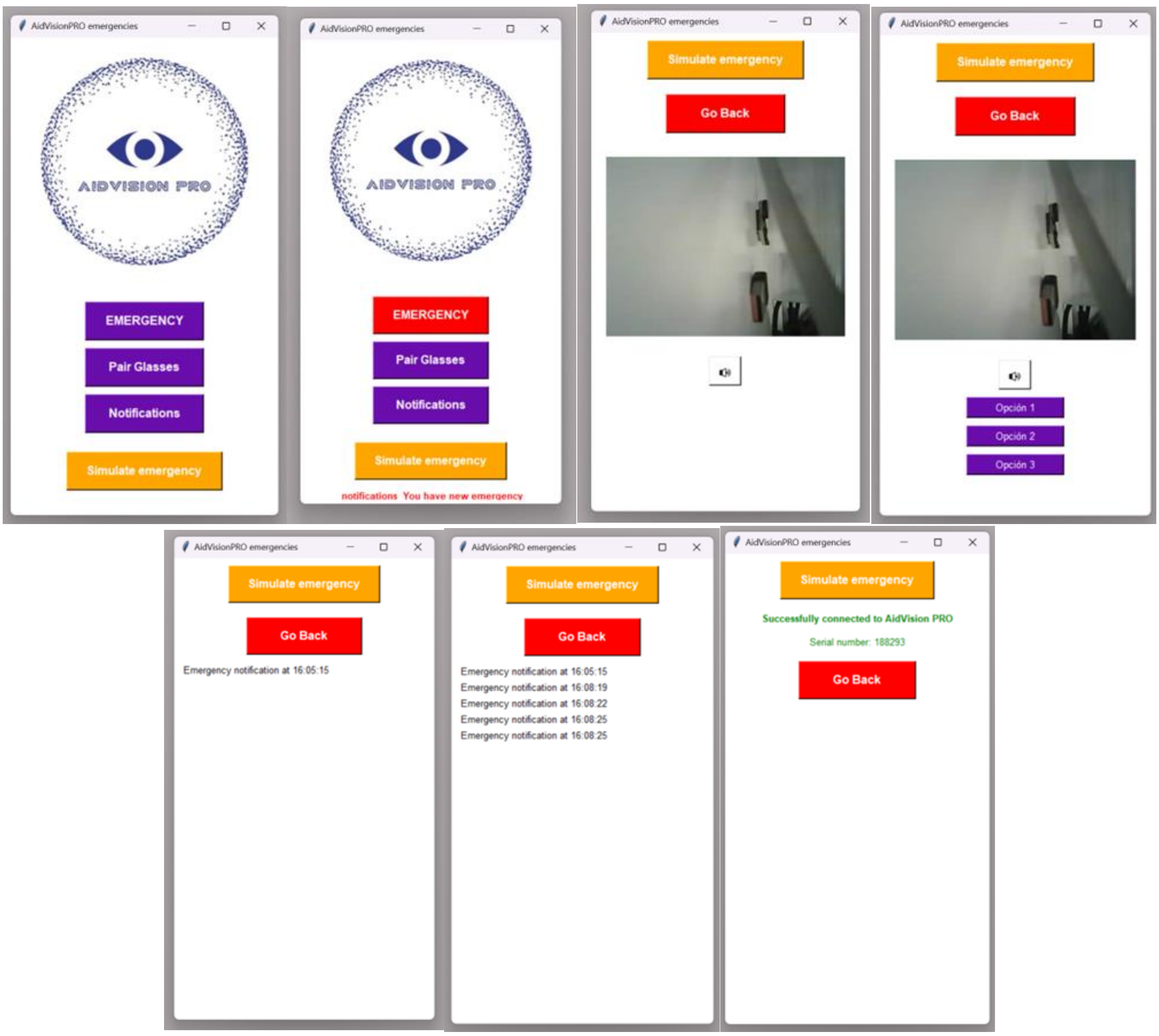


Figure 44. *Functional design in PyCharm within the computer*

This program aims to act as an app but is currently being run in the computer, serving as a prototype of how the app will be seen and how it will act.

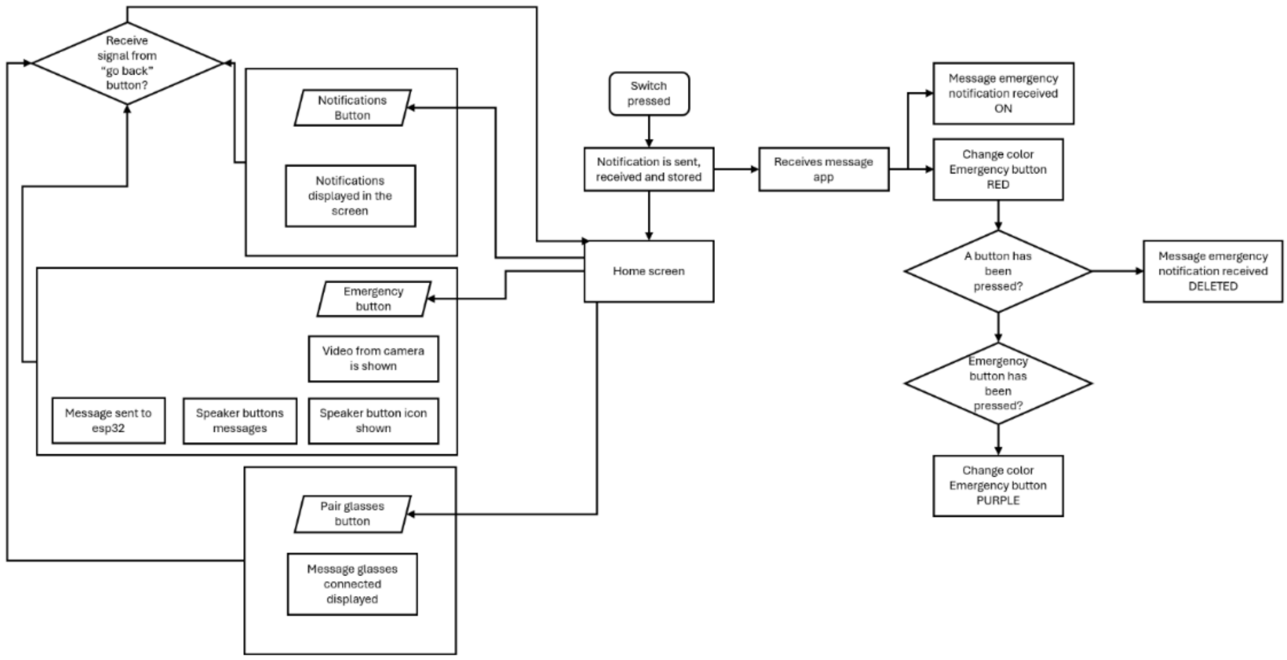


Figure 45. *Block diagram how the app functions*

3.5 APPENDICES

3.5.3 PDR

3.5.3.1 GENERAL GOAL OF THE PROJECT

Purpose Definition and Social Mission

The **goal** of this project is **to develop smart glasses that assist individuals with visual disabilities in navigating their environment by detecting nearby objects** (cars and people).

"A Vision in which millions of visually impaired individuals gain access to greater independence and calm in their daily lives, enhancing their overall well-being".

Visual and sensory identity

Figure 46. *Logo of our company, AidVision Pro.*

Our visual identity is crucial for us. Firstly, our logo and its colors, represented in **Figure 1**, have an important meaning, blue symbolizes confidence, harmony and independence while white symbolizes simplicity, seriousness and clarity. Moreover, our slogan: “The technology that transforms darkness into clarity” emphasizes on our scope for this project. To have also a sensory identity we chose mint (fresh air), as it could enhance confidence and calm. It also reflects the freedom and independence that our product aims for. Moreover, we chose lavender to represent the calm, peace of mind and confidence that our glasses will give users to navigate their surroundings.

Elevator Pitch

Imagine losing most of your eyesight—unable to walk safely or perform daily tasks. Wouldn't you do everything possible to regain your independence?

At AidVision Pro, we've developed an affordable, cutting-edge solution to transform the lives of those with vision loss. Our advanced glasses not only extend peripheral vision through fisheye lenses, but also use AI-powered technology to detect objects in your surroundings—whether they're cars or people—and announce through a built-in speaker the exact number of each detected object, providing you with real-time environmental awareness. If an object is within 5 meters, a vibration motor alerts the wearer, ensuring safer navigation.

We're also proud to introduce a new feature: an AI that reads and announces any text in front of you, making everyday tasks even easier.

We invite you to be part of this breakthrough. Together, we can empower millions to see and navigate the world with renewed confidence.

Customer needs

Our product is directed to customers who are totally or partially blind people.

- Increased safety.
- Easiness to avoid obstacles at the line of sight.
- Complement the use of guide dogs with a more suitable system.
- Object recognition to provide more guidance.
- Assistance for users who might need to make a videocall with a relative to get help.

These customer needs were selected due to the interviews we made in the preview project:

− Interview to Manuel Cruz: a person with glaucoma and macular degeneration

Manuel Cruz was selected for our interview due to his personal experience of living with visual impairment. He knows the daily challenges faced by individuals with vision loss and can offer valuable insights on the needs of our smart glasses.

- What is exactly your vision impairment? *I have macular degeneration and glaucoma. Also, I have undergone cataracts surgery but ended up with diopter.*

- Are you currently using glasses? *I am using progressive glasses, but they no longer serve me.*

- In which daily activities do you notice the vision loss the most? *I am a person who likes to read, but nowadays even with a magnifying glass I am not able to do it. Moreover, I don't feel confident to walk out of my house because I might fall or drive my car.*

- Have you ever experienced situations where vision loss could lead to an accident? *For example, difficulty seeing traffic lights or detecting vehicles approaching at high speeds. I have tripped and fallen, though it didn't have serious consequences.*

- If these glasses could have functionalities you desire, what would you choose? For example, motion detector, bill identifiers, AI narrating what is being seen, object identifier… *Those functionalities that* *help me to perceive the reality of my surroundings because I can distinguish bills although only when I approximate them to my face.*

- In what situations or daily activities do you think they could make a real difference? *I can't go to any place so it will make a real difference in my daily life. I am recruited in my house because I must take care of my wife, however I can't prepare food. With the glasses all those actions that make me feel burdensome, will no longer make me feel that.*
- Do you think there could be challenges in terms of the acceptance or purchase of these glasses by people with blindness? *I don't think so as any innovative product is very appealing to us.*
- Which media channels do you find most effective for advertising these glasses? *I don't believe television is the adequate because I have complications watching TV. I have hearing aids connected to the mobile phone. In order to read the newspaper, I have someone reading it for me.*

Manuel Cruz highlighted how vision impairment limits his independence in his daily life, especially for reading and moving safely, he mentioned his actual glasses don't help, and he believes smart glasses could help him improve his quality of life.

- Interview to ASPREH
  - What does your association consist of? *ASPREH is an association of rehabilitation professionals for individuals with visual disabilities, operating within the Spanish context. It consists of professionals such as ophthalmologists, optometrists, occupational therapists, psychologists, rehabilitation technicians…*
  - How do you help people with legal blindness? *We provide guidance to individuals with visual disabilities (whether they are legally blind or they have low vision), offering them strategies to enhance their functionality in daily life…*
  - What difficulties and challenges do people with visual disabilities face in their day-to-day lives in Spain? *It varies for each person based on their lifestyle and needs, but challenges include carrying out daily activities, orientation and mobility, access to the job market, general accessibility issues…*
  - Do you know anyone with peripheral vision loss? If so, what are their difficulties? *They often face issues with orientation and mobility, poor night vision, low contrast sensitivity (difficulty adapting to changes in lighting conditions) …*
  - Do you think these innovative glasses could improve the quality of life for people with visual disabilities? *It should be clear in which cases they could be useful and tested beforehand with real patients.*
  - In what situations or daily activities do you think they could make a real difference? *It depends on the user, their visual condition, and the goals set for the individual.*
  - What social and emotional impact do you think these glasses could have on users? *Especially in terms of their autonomy or independence. It should be determined if the device meets the user's needs and various other variables (weight, comfort, price…).*
  - Do you think there could be challenges in terms of the acceptance or purchase of these glasses by people with blindness? *Certainly, it is a field with a lot of room for improvement, in terms of technology. Therefore, ensuring feasibility and conducting a prior market study of possible similar devices already in the market and gathering feedback, especially from patients, would be crucial.*

The interview to ASPREH also highlighted the daily challenges faced by individuals with vision loss and emphasized the importance of evaluating in which cases the glasses could be helpful as well as comparing them to similar devices, gathering feedback, and testing them in real patients.

3.5.3.2. CONOPS

→ **Scenario 1: Navigating an unfamiliar public space**
The user is in a public building, such as a post office or hospital, or in the street with an unfamiliar layout. As the user walks, the glasses detect obstacles such as walls, counters, and doors or in the case of the street a building, using ultrasonic and vision sensors. When the user approaches an obstacle, the device triggers a vibration alert, allowing the user to adjust their direction and safely navigate an unfamiliar environment.

→ **Scenario 2: Detect cars and advise the user in the street**
The user is in an intersection without traffic light. The glasses detect a car that is approaching and gives an audible warning to the user to wait before passing. Moreover, the glasses will advise the user how many cars are and if it is to the side or facing front.

→ **Scenario 3: Detect people and advise the user in the street**
The is walking on the sidewalk and a person approaches him. The device detects it and warns the person, so he can walk along an obstacle- free route. The glasses will advise the customer if that person is positioned to the side or facing forward. Additionally, the user will be informed of how many people are there on each side.

→ **Scenario 4: Object detection (not only person and cars)**
The user is walking in a busy street with a route for bicycles. The glasses detect a bike that has slightly gone out of his path and is going to cross with the visually impaired individual. Moreover, they will alert the user with a specific sound when the object is moving towards them.

3.5.3.3 LIST OF REQUIREMENTS

Top level requirements

- The system shall help visually impaired in their day-to-day life
- The system shall detect a wall
- The system shall vibrate when it detects a wall
- The system shall distinguish cars and make a specific buzzle
- The system shall distinguish people and make another specific buzzle

Component requirements

- The ultrasound sensor shall detect walls before 1 - 1.5 m
- The vibrator motor shall vibrate when a wall is at a distance of 1 – 1.5 m
- The camera shall obtain video images in real time and have the capacity of connecting it with the computer
- The camera shall obtain video images in real time with enough resolution (720p)
- The camera shall obtain video images in real time with enough frames per second and resolution (50 fps)
- The software (AI) shall distinguish a car from other objects
- The software (AI) shall distinguish people from other objects
- The buzzer shall make a buzzle in a specific way when a car is near
- The buzzer shall make a different buzzle when a person is near

3.5.3.4 TRADE OFFS

Glasses

- **Musts**
  The glasses with the integrated devices must be comfortable and not too heavy.
  They should help the user walk more confidently.
  They must have a user-friendly interface that is easy to understand.
  They should be able to be charged.
- **Wants**
  They may include customization settings like vibration or sound intensity.

Obstacle detection system

- **Musts**
  It must detect objects at the height of ultrasound.
  When an object is detected, the glasses should vibrate to alert the user.
- **Wants**
  It may detect objects that are lower or higher than ultrasound.
  It may have customizable vibration intensity.

Artificial intelligence that detects specific objects

- **Musts**
  It shall detect cars and people that are directly in front of the user.
  It shall detect objects with slight movement in the image.
  When cars or people are detected, the glasses should emit a specific sound alert.
  The speaker must have the option to be turned off.
- **Wants**
  The glasses may detect cars and people that are very far away or not at plain sight.
  Recognize other objects from day life (cats, dogs…).
  It may detect objects even when the user is moving a lot, and the image is not completely clear.

-Camera module → ESP32-CAM

Decision statement:		Camera module														
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3			Alternative 4			Alternative 5		
		ESP 32-CAM			ESP 8266 + OV7670			Raspberry Pi Zero W + Camera			Arduino MKR Wifi 1010 + OV7670			AI- Thinker A9G + Camera		
Musts (Go/ No-GO)																
Connect to Arduino/ Rasberry		GO			GO			GO			GO			GO		
Min: 720p		GO			GO			GO			GO			NO GO		
Min: 24 fps		GO			GO			GO			GO			NO GO		
Weight < 50g		GO			GO			GO			GO			GO		
Wifi		GO			GO			GO			GO			NO GO		
High processing capacity		GO			NO GO			GO			GO			NO GO		
Price < 25		GO			GO			GO			NO GO			NO GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	9	13,99	8	72			0	22.99	4	36			0			0
Dimensions	8	27mm x 40 mm	7	56			0	65mm x 30 mm	4	32			0			0
Energy Consumption	7	0,5-1W	9	63			0	1,5-2,15W	5	35			0			0
Easy use	5	Easy	8	40			0	More steps	6	30			0			0
MAX Score (10xW)	290															
Total Score		231			0			133			0			0		

We will use a ESP32-CAM and a ESP32 without the camera module to receive the information.

-Ultrasound sensor → AZ Delivery Sensor

Decision statement:		Ultrasound sensor											
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3			Alternative 4		
		AZ Delivery Sensor			Grove - Ultrasonic Ranger			SRF05 Sensor			MaxBotix MB1010		
Musts (Go/ No-GO)													
Detection range > 1,5 m		GO			GO			GO			GO		
Connect to Arduino/ Rasberry		GO			GO			GO			GO		
Weight < 20g		GO			GO			GO			GO		
Detection angle > 10º		GO			GO			GO			GO		
Price < 10		GO			NO GO			GO			NO GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	9	5,99	8	72			0	4,95	9	81			0
Dimensions	8	45 x 20 x 15 mm	7	56			0	40 x 20 x 20 mm	7	56			0
Energy Consumption	7	0,075 W	9	63			0	0,1 W	9	63			0
Easy use	5	Easy	9	45			0	More configurations	7	35			0
Precision	8	± 3 mm	9	72			0	± 1 cm	6	48			0
MAX Score (10xW)	370												
Total Score		308			0			283			0		

- Vibration motor → Mini vibration motor coin

Decision statement:		Vibration motor								
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3		
		Shaftless Vibration Motor of Sparkfun			5pcs Flat Vibrating Motor DC3V			Mini motor vibrador moneda Plano 10x2mmpara Arduino Raspberry pi		
Musts (Go/ No-GO)										
Connect to Arduino/ Rasberry		GO			GO			GO		
Weight < 30g		GO			GO			GO		
Small size		GO			GO			GO		
Price < 10		GO			GO			GO		
Good vibration intensity		GO			GO			GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	8	2,25	9	72	9,96	6	48	1,28	10	80
Dimensions	8	D=10mm, H=3,3mm	9	72	D=10mm, H=2mm	10	80	D=10mm, H=2mm	10	80
Energy Consumption	7	Low	9	63	Low	9	63	Low	9	63
Easy use	5	Easy	8	40	Easy	8	40	Easy	8	40
MAX Score (10xW)	280									
Total Score		247			231			263		

-Speaker→ KeeYees Mini MP3 DFPlayer

Decision statement:		Speaker											
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3			Alternative 4		
		Eageroo Super Mini Altavoz Bluetooth			KeeYees Mini MP3 DFPlayer			Cablepelado Mini Altavoz			Mini Metal Speaker w/ Wires - 8 ohm 0.5W		
Musts (Go/ No-GO)													
Connect to Arduino/ Rasberry		NO GO			GO			GO			GO		
Weight < 50g		GO			GO			GO			GO		
Small size		GO			GO			GO			GO		
Price < 10		NO GO			GO			GO			NO GO		
Min: 2W		GO			GO			GO			NO GO		
Plays MP3 files		GO			GO			NO GO			GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	9			0	9.99	7	63			0			0
Dimensions	8			0	D=4cm,H=0.5cm	9	72			0			0
Energy Consumption	7			0	Low	9	63			0			0
Easy use	5			0	Easy	9	45			0			0
MAX Score (10xW)	290			0									
Total Score		0			243			0			0		

- Step-down transformer → LM2596S

Decision statement:		Step-down transformer								
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3		
		LM2596S			LM317			XL4015		
Musts (Go/ No-GO)										
Input voltage > 3V		GO			GO			GO		
Variable ouput voltage		GO			GO			GO		
Price < 15		GO			GO			GO		
Output current > 1,5 V		GO			GO			GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	9	8,99	9	81	10,49	7	63	13,99	5	45
Efficiency	6	High	8	48	Low	5	30	High	8	48
Easy integration with arduino	7	High	8	56	Low	5	35	Moderated	7	49
Refrigeration	6	Not required	9	54	Required (high dissipator)	4	24	Required	5	30
Input voltage range	9	4-40 V	8	72	3-40 V	9	81	5-36 V	7	63
Output voltage range	9	1,25-35 V	8	72	1,25-37 V	8	72	1,25-32 V	7	63
MAX Score (10xW)	460									
Total Score		311			200			205		

-Battery charger controller → TP4056

Decision statement:	Battery charger controller	Alternative 1	Alternative 2	Alternative 3	Alternative 4
Evaluation Criteria:	TP4056	MCP73831	BQ24072	MAX1811	CN3065
Musts (Go/ No-GO)					
Capacity to charge LiPo	GO	GO	GO	GO	GO
Overcharge protection	GO	GO	GO	GO	GO
Low voltage cutoff	GO	NO GO	GO	NO GO	GO
Thermal protection	GO	NO GO	GO	GO	GO
Reverse polarity protection	GO	NO GO	GO	NO GO	GO
Wants	Weight	Comments	Raw (R)	R*W	Comments
Price	8	1,50 USD	5	40	4 USD
Charge current flexibility	9	Up to 1A	5	45	Up to 500mA
Ease of use (components)	7	Simple	5	35	Simple
Input voltage range	7	5V (USB)	5	35	5V to 6V
Protection circuit simplicity	8	Built-in	5	40	External components needed
MAX Score (10xW)	450				
Total Score	195		153	163	155

-Wires à Fermerry 20 AWG Wire

Decision statement:	Battery															
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3			Alternative 3			Alternative 4		
		Rechargable battery Lipo Gens ace 7,4V 1000mAh 25C			Litium ion battery CASEA 3,7V 6000mAh			NiMh Basics Pilas			Alcaline batteries 9V			Charmast 10400mAh Power Bank		
Musts (Go/ No-GO)																
Price <30		GO			GO			GO			GO			GO		
Weight < 300g		GO			GO			GO			GO			GO		
Rechargable		GO			GO			GO			NO GO			GO		
Durability		GO			GO			NO GO			NO GO			GO		
Capacity		GO			NO GO			NO GO			NO GO			GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	9	20,93	9	81			0			0			0	12.99	10	90
Dimensions	9	103x19x11mm	8	72			0			0			0	184 x 101 x 22mm	8	72
Difficulty	10	Complicated	3	72			0			0			0	Easy use	10	100
Charge time	7	Fast	9	63			0			0			0	Fast	8	56
Use life	8	150 uses	9	72			0			0			0	Long lasting	9	72
MAX Score (10xW)	410															
Total Score		360			0			0			0			390		

- Battery → 10400mAh Power Bank

Decision statement:		Wires														
Evaluation Criteria:		Alternative 1			Alternative 2			Alternative 3			Alternative 4			Alternative		
		Fermerry 20 AWG Wire			UL1007 22 AWG Wire			Silicone Wire 24 AWG			Dupont 20 AWG Wire			18 AWG Solid Copper Wire		
Musts (Go/ No-GO)																
Capacity to handle electrical current		GO			GO			GO			GO			GO		
Flexibility (easy handling)		GO			GO			GO			GO			NO GO		
Proper insulation		GO			GO			GO			GO			GO		
Heat resistance		GO			NO GO			GO			NO GO			GO		
Wants	Weight	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W	Comments	Raw (R)	R*W
Price	8	25,99	5	40				28,99	4	32			0			0
Flexibility (ease of use)	7	flexible	5	35				flexible	5	35			0			0
Heat resistance	9	-60ºC to 200ºC	5	45				-60ºC to 200ºC	5	45			0			0
Durability of insulation	6	provides insulation	5	30				provides insulation	5	30			0			0
Variety of colors	5	6 colors	5	25				2	3	15			0			0
MAX Score (10xW)	450															
Total Score		175			0			157			0			0		

3.5.3.5 BUCKET LIST

AID VISION PRO					
Unit Price Without VAT	**Cantidad**		**Total Price With VAT**	**Description**	**Link**
€ 11,56	1	€ 11,56	€ 13,99	ESPCAM 32	https://acortar.link/VBMQWS
€ 9,50	1	€ 9,50	€ 11,50	ESP32 WROOM-32	https://acortar.link/rkaQuM

€ 8,26	2	€ 16,52	€ 19,99	Mini Mp3 DFplayer + Speaker	**https://acortar.link/GMrvsh**
				MicroSD card	https://acortar.link/cGnFmB
€ 6,79	2	€ 11,22	€ 13,58	Proximity sensor	https://acortar.link/yf614z
€ 2,95	6	€ 8,23	€ 9,96	Vibration motor	
€ 21,48	1	€ 21,48	€ 25,99	Wires	
€ 20	1	€ 20		Battery	
TOTAL PRICE WITH VAT 230					

3.6 REFERENCES

3.6.1 For the customer needs and project definition

- *ASPREH*. (n.d.). Retrieved November 5, 2023, from https://www.aspreh.org/
- *Certificación ISO 13485 - Gestión de productos sanitarios | NQA*. (n.d.). Retrieved December 1, 2023, from https://www.nqa.com/es-es/certification/standards/iso-13485/implementation
- *Cuatro millones de indios dejan de ser ciegos por un cambio de criterio legal | Sociedad Home | EL MUNDO*. (n.d.). Retrieved September 27, 2023, from https://www.elmundo.es/sociedad/2017/04/20/58f8d28a268e3ee34c8b4682.html
- *Mapa de países y estimaciones de la pérdida de visión - Agencia Internacional para la Prevención de la Ceguera*. (n.d.). Retrieved September 27, 2023, from https://www.iapb.org/es/learn/vision-atlas/magnitude-and-projections/countries/
- *Startup española quiere sustituir los perros guía para ciegos por unas gafas futuristas con IA y robótica - Informativo Entre Todos*. (n.d.). Retrieved October 10, 2023, from https://entretodos.com.mx/noticias/startup-espanola-quiere-sustituir-los-perros-guia-para-ciegos-por-unas-gafas-futuristas-con-ia-y-robotica/
- *Las gafas que pueden ayudar a ver a personas con visión muy baja | Tecnología | EL PAÍS*. (n.d.). Retrieved October 10, 2023, from https://elpais.com/tecnologia/2017/02/22/actualidad/1487765942_466913.html
- *Ayudas ópticas, no ópticas y electrónicas para personas con resto visual — Web de la ONCE*. (n.d.). Retrieved October 5, 2023, from https://www.once.es/servicios-sociales/autonomia-personal/paginas-rehabilitacion/ayudas-opticas-no-opticas-y-electronicas-para-personas-con-resto-visual
- *mdr, 2017 (1)*. (n.d.).
- *These Glasses Could Help the Blind See | Smithsonian*. (n.d.). Retrieved October 10, 2023, from https://www.smithsonianmag.com/innovation/these-glasses-could-help-blind-see-180955797/
- *India: Economía y demografía 2024 | Datosmacro.com*. (n.d.). Retrieved September 27, 2023, from https://datosmacro.expansion.com/paises/india#
- *Unos 314 millones de personas sufren discapacidad visual en el mundo, de las que 45 millones son ciegas, según la OMS – Solidaridad Intergeneracional*. (n.d.). Retrieved September 27, 2023, from https://solidaridadintergeneracional.es/wp/unos-314-millones-de-personas-sufren-discapacidad-visual-en-el-mundo-de-las-que-45-millones-son-ciegas-segun-la-oms/
- Bourne, R. R. A., Steinmetz, J. D., Saylan, M., Mersha, A. M., Weldemariam, A. H., Wondmeneh, T. G., Sreeramareddy, C. T., Pinheiro, M., Yaseri, M., Yu, C., Zastrozhin, M. S., Zastrozhina, A., Zhang, Z. J.,

Zimsen, S. R. M., Yonemoto, N., Tsegaye, G. W., Vu, G. T., Vongpradith, A., Renzaho, A. M. N., … Vos, T. (2021). Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness in relation to VISION 2020: The Right to Sight: An analysis for the Global Burden of Disease Study. *The Lancet Global Health*, *9*(2), e144–e160. https://doi.org/10.1016/S2214-109X(20)30489-7

- *La India es el país con más discapacitados visuales del mundo - Medicina Responsable*. (n.d.). Retrieved September 17, 2023, from https://medicinaresponsable.com/sabias-que.../india-discapacitados-visuales-mundo
- *These AI-Powered Smart Glasses are Changing the Lives of Blind People*. (n.d.). Retrieved October 12, 2023, from https://www.intelligentliving.co/ai-powered-smart-glasses-changing-lives-blind-people/
- *Los ciegos podrán “ver” con lentes con inteligencia artificial, pero son muy caros - Buena Vibra*. (n.d.). Retrieved November 3, 2023, from https://buenavibra.es/entretodos/lentes-inteligentes-para-ciegos/
- *Crean las gafas basadas en IA para personas con ceguera*. (n.d.). Retrieved October 29, 2023, from https://www.mundodeportivo.com/urbantecno/tecnologia/gafas-envision-describir-personas-ciegas
- *Las gafas para ciegos que transforman lo que te rodea en sonido*. (n.d.). Retrieved November 10, 2024, from https://www.elespanol.com/omicrono/tecnologia/20180328/gafas-ciegos-transforman-rodea-sonido/295471816_0.html

- *Una startup española quiere sustituir los perros guía para ciegos por unas gafas futuristas con IA y robótica*. (n.d.). Retrieved November 7, 2023, from https://computerhoy.20minutos.es/tecnologia/biel-glasses-gafas-inteligencia-artificial-realidad-aumentada-discapacidad-visual-1185444
- *eSight - Electronic eyewear for the visually impaired*. (n.d.). Retrieved November 7, 2023, from https://www.esighteyewear.com/

3.6.2 For the software and hardware design

- *Usa un Push boton con el ESP32 - YouTube*. (n.d.). Retrieved October 21, 2024, from https://www.youtube.com/watch?v=US3aFqL0BRA
- *ESP32CAM Detección de personas con YOLOV8 - YouTube*. (n.d.). Retrieved October 10, 2024, from https://www.youtube.com/watch?v=QYzCBD62QN4
- *DFplayer Mini to ESP32 Connection Tutorial - Easy Steps | 2024 - YouTube*. (n.d.). Retrieved October 21, 2024, from https://www.youtube.com/watch?v=4PExe2b1Ckw
- *DFPlayer Mini Interface with ESP32: Audio Playback Tutorial | Add voice to ESP32 - YouTube*. (n.d.). Retrieved October 21, 2024, from https://www.youtube.com/watch?v=9w_AalwlsE4
- *Formas de alimentar una ESP32 DEVKIT 1 - YouTube*. (n.d.). Retrieved October 26, 2024, from https://www.youtube.com/watch?v=vTIlNM_Flck
- *UdpCommunication - Python Wiki*. (n.d.). Retrieved November 4, 2024, from https://wiki.python.org/moin/UdpCommunication
- *ArduinoIoT Distance Detection Light Module full explaination #iot #trending #technology #innovation - YouTube*. (n.d.). Retrieved October 21, 2024, from https://www.youtube.com/watch?v=hBh_GfDVNs0

- *Configura y Programa 🔥 el CrowPanel Pico Display de Elecrow: Arduino IDE y SquareLine Studio - YouTube*. (n.d.). Retrieved November 4, 2024, from https://www.youtube.com/watch?v=Veg2064v3B4
- *Esp32 - **Electrodaddy***. (n.d.). Retrieved November 4, 2024, from https://electrodaddy.com/esp32/
- *ESP32CAM Clasificacion de frutas y vegetales en tiempo real - YouTube*. (n.d.). Retrieved October 21, 2024, from https://www.youtube.com/watch?v=JoJQWIjKYI8
- *Ejecutar script python como servicio en la Raspberry Pi - YouTube*. (n.d.). Retrieved October 21, 2024, from https://www.youtube.com/watch?v=U8pLVzBrM38

3.6.3 For the user interface

- *Flutter - Build apps for any screen*. (n.d.). Retrieved November 7, 2024, from https://flutter.dev/
- *GitHub - flet-dev/flet: Flet enables developers to easily build realtime web, mobile and desktop apps in Python. No frontend experience required.* (n.d.). Retrieved November 7, 2024, from https://github.com/flet-dev/flet
- *Building Flutter Apps in Python | Making 2 Apps From Scratch - YouTube*. (n.d.). Retrieved November 7, 2024, from https://www.youtube.com/watch?v=Vkm0O4B-vts
- *Crea una APP Android, iOS y Web usando FLUTTER - YouTube*. (n.d.). Retrieved November 7, 2024, from https://www.youtube.com/watch?v=7N-wDxe18PA