# Name: Mini Mp3 Player

## Team:

1.Condrea Andreea 1305A
2.Flocea Ana-Gabriela 1305A
3.Marin Olivia 1305A
4.Ghiba Răzvan-Constantin 1310A

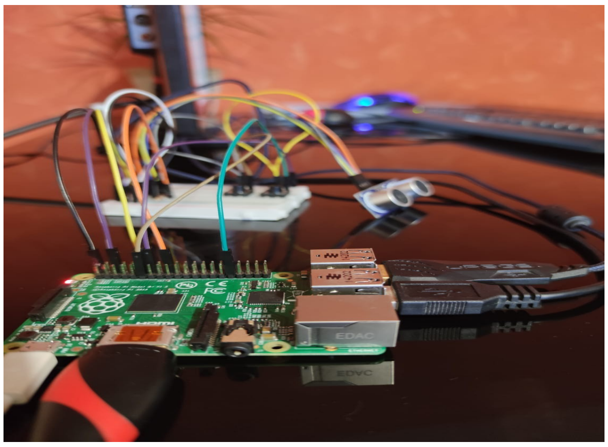
## Elevator Pitch:

What would our life be without music? Boring,isn't it?
We are convinced that, at least once, while listening music on your new Mp3 Player, you asked yourself, ``Could I make something like that?`` Well, maybe we have an answer for you. You want to find out? Then, we invite you to follow our process of creating a mini Mp3 player. Enjoy!

## Story:

Our goal was to create a mini Mp3 player, using **Raspberry Pi**, with a list of funny songs, easy to use by users, but just as easy to create by any beginner in the field of microprocessors. The volume is controlled with a simple touch of the hand.

## Cover Image:



## Components:

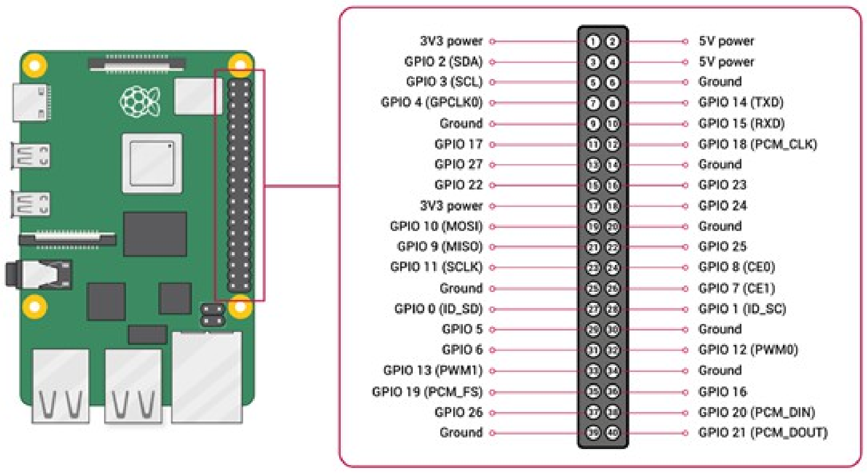
- **Hardware:**
  - o A Raspberry Pi computer

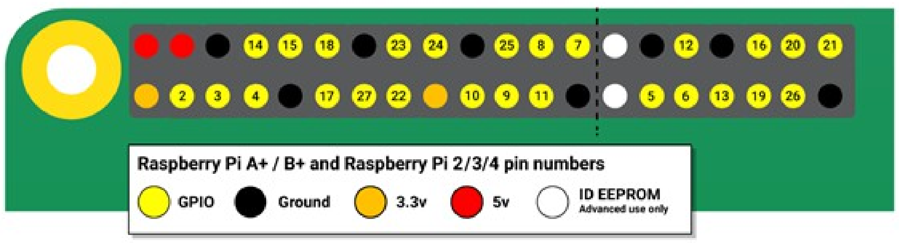
- o A breadboard
- o 1x HC-SR04 sensor.
- o Two tactile switches ( buttons )
- o 11 male-to-female jumper leads
- o 4 male-to-male jumper leads
- o 2x 1k ohm resistors
- o Tv or laptop

• **Software:**

- o Raspberry PI Raspbian
- o Python 2

## Schematics:

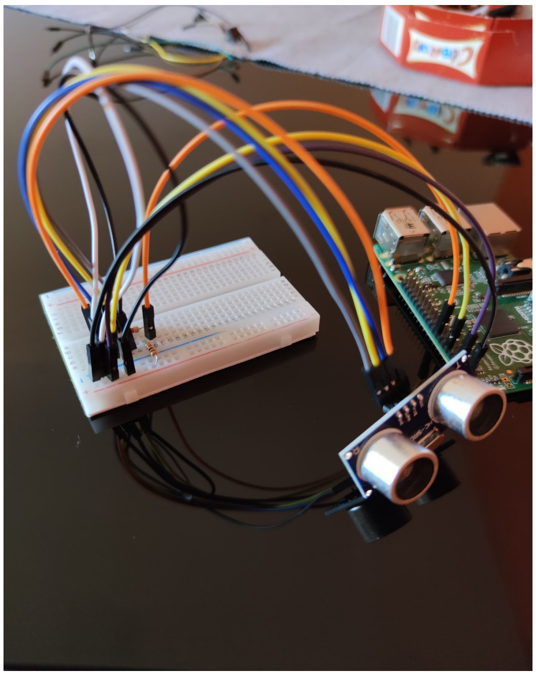


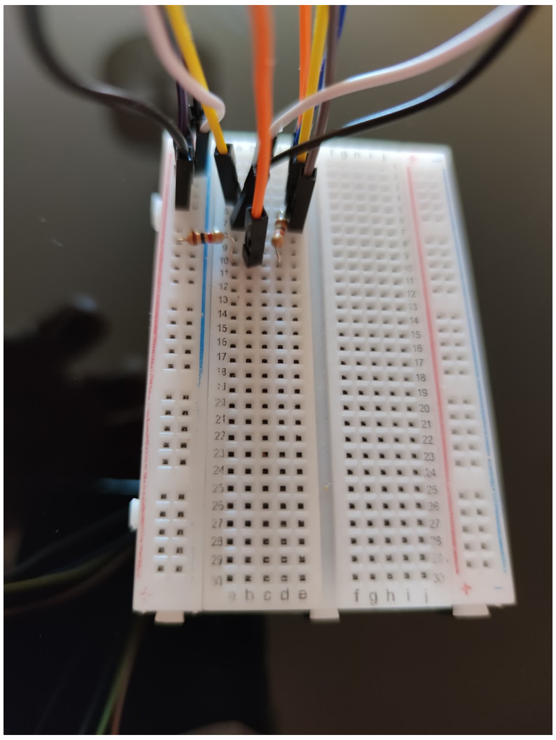


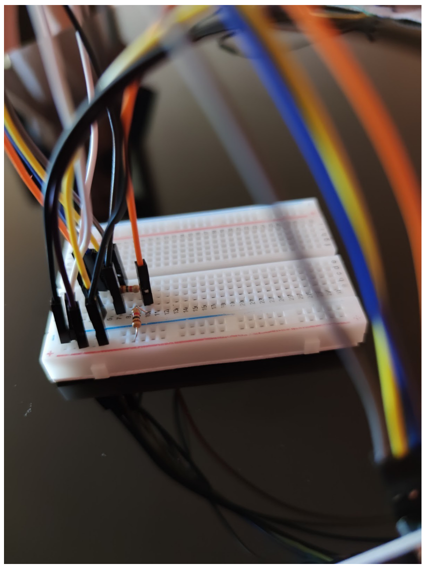
## Steps:

• First of all, we have to assemble the circuit. Below we have some images suggestive for assembly, very easy to follow and understand.

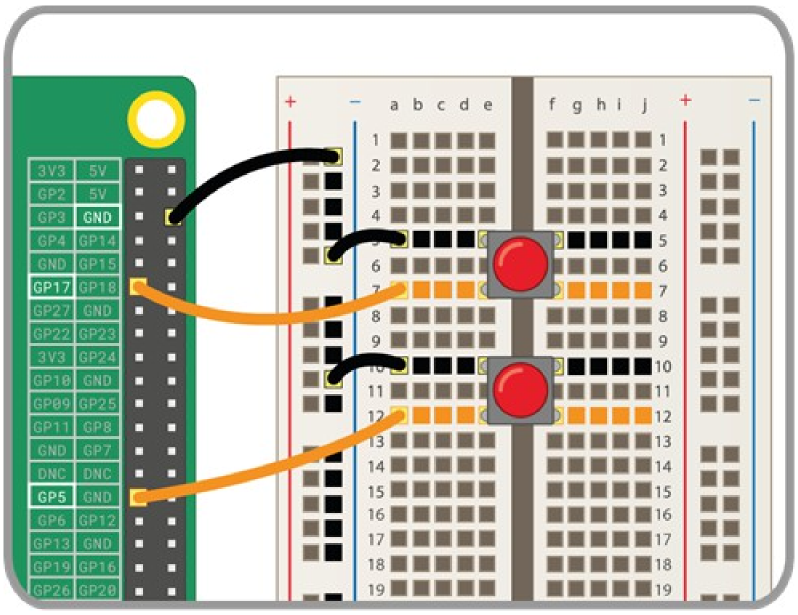
o The sensor aims to take the distance from the user's hand and adjust the volume accordingly.







o We have two buttons. The first is to turn on the mp3 player and change the song. The second stops the current song.



• Now that we have assembled the circuit, we can move on to the software development part.

o Firstly, we will download a few songs to make the list.

Be careful: Files must be in .wav format!

o In order to work with such files we need the pygame library:

**sudo apt-get install python-pygame**

We'll also need the alsaaudio library, to set the volume:

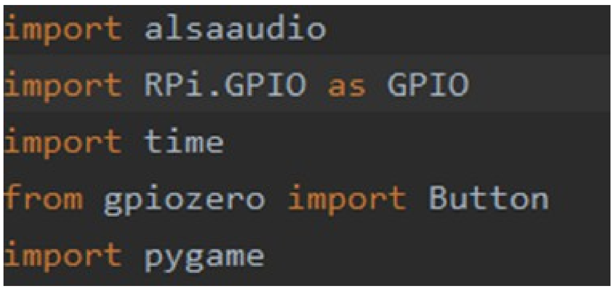
**sudo -H pip3 install pyalsaaudio**

and GPIO library, to connect the microcontroller to our electronic devices:

**sudo apt install python3-gpiozero**

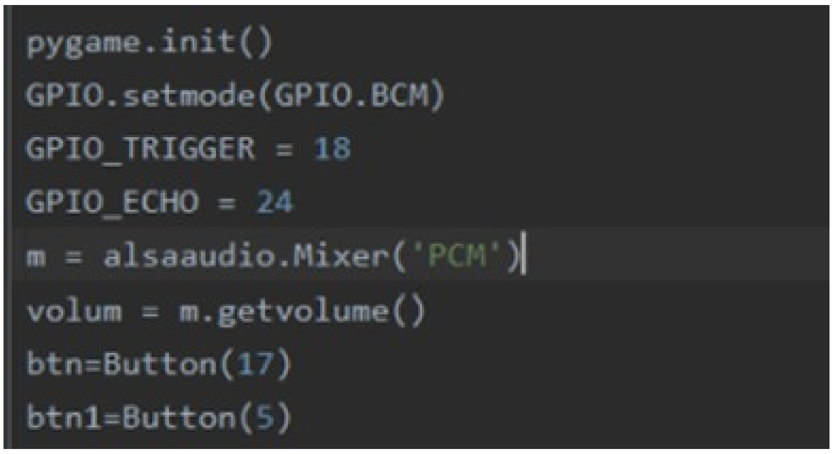
o After we download all the libraries, we can start to write the code:

- we import all the libraries



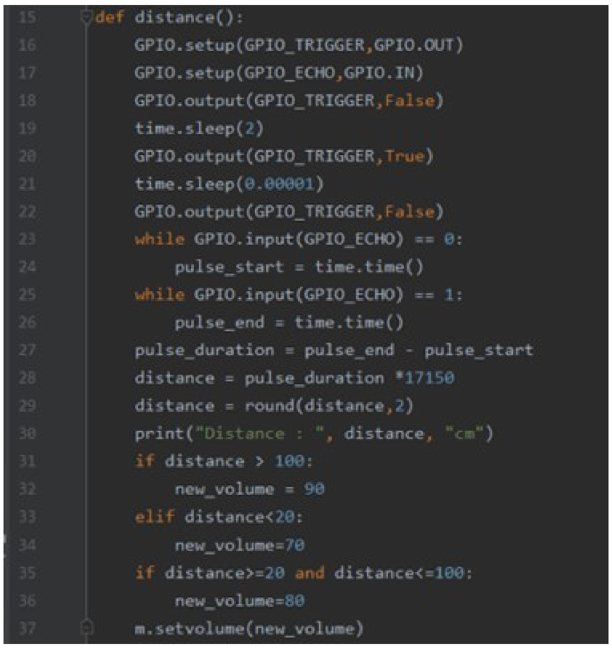
```
import alsaaudio
import RPi.GPIO as GPIO
import time
from gpiozero import Button
import pygame
```

- we set our GPIO pin numbering using setmode(). We'll name our output pin(trigger the sensor) GPIO_TRIGGER( Pin 18) and our input pin GPIO_ECHO(Pin 24).
  **btn** and **btn1** are corresponding to the two buttons: **btn** to stop the song, and **btn1** change the song.



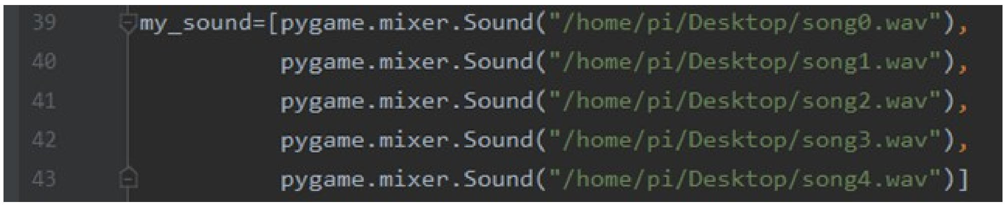
```
pygame.init()
GPIO.setmode(GPIO.BCM)
GPIO_TRIGGER = 18
GPIO_ECHO = 24
m = alsaaudio.Mixer('PCM')
volum = m.getvolume()
btn=Button(17)
btn1=Button(5)
```

- **distance()** is the function which calculate the distance between the user's hand and the sensor. Also, in this function we set the volume in correspondence with the calculated distance.



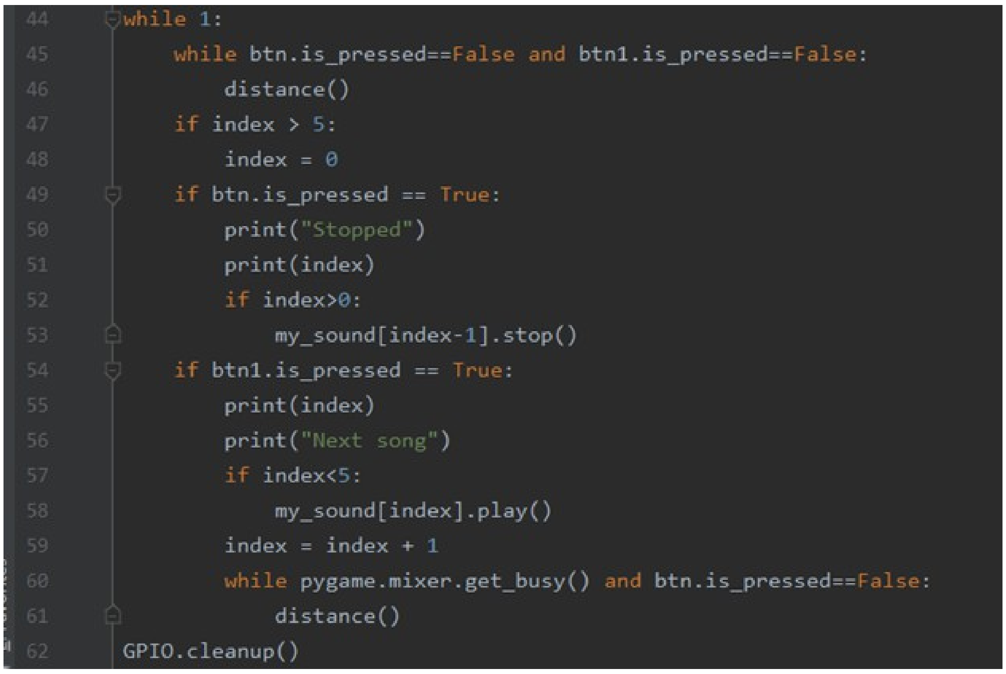
```python
def distance():
    GPIO.setup(GPIO_TRIGGER,GPIO.OUT)
    GPIO.setup(GPIO_ECHO,GPIO.IN)
    GPIO.output(GPIO_TRIGGER,False)
    time.sleep(2)
    GPIO.output(GPIO_TRIGGER,True)
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER,False)
    while GPIO.input(GPIO_ECHO) == 0:
        pulse_start = time.time()
    while GPIO.input(GPIO_ECHO) == 1:
        pulse_end = time.time()
    pulse_duration = pulse_end - pulse_start
    distance = pulse_duration *17150
    distance = round(distance,2)
    print("Distance : ", distance, "cm")
    if distance > 100:
        new_volume = 90
    elif distance<20:
        new_volume=70
    if distance>=20 and distance<=100:
        new_volume=80
    m.setvolume(new_volume)
```

- we create a list with all of our songs.



```python
my_sound=[pygame.mixer.Sound("/home/pi/Desktop/song0.wav"),
          pygame.mixer.Sound("/home/pi/Desktop/song1.wav"),
          pygame.mixer.Sound("/home/pi/Desktop/song2.wav"),
          pygame.mixer.Sound("/home/pi/Desktop/song3.wav"),
          pygame.mixer.Sound("/home/pi/Desktop/song4.wav")]
```

- the most important part of the program is located in **while(1) loop**. The first while check the state of each button and begins to calculate the distance. If **btn** is pressed then the song is stopped. We press **btn1** after stopping the previous song, and we move to the next song.



```
while 1:
    while btn.is_pressed==False and btn1.is_pressed==False:
        distance()
    if index > 5:
        index = 0
    if btn.is_pressed == True:
        print("Stopped")
        print(index)
        if index>0:
            my_sound[index-1].stop()
    if btn1.is_pressed == True:
        print(index)
        print("Next song")
        if index<5:
            my_sound[index].play()
        index = index + 1
        while pygame.mixer.get_busy() and btn.is_pressed==False:
            distance()
GPIO.cleanup()
```

In the end, we must clean our GPIO pins to ensure that all inputs/outputs are reset.

## Demo: https://drive.google.com/file/d/1-3vOaNoz0fSG9gBzLKcrkjuPFrUEePIJ/view