République Algérienne Démocratique et Populaire
الجمهورية الجزائرية الديمقراطية الشعبية
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
وزارة التعليم العالي و البحث العلمي

ECOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE

المدرسة الوطنية العليا للإعلام الآلي
(المعهد الوطني للتكوين في الإعلام الآلي سابقا)
Ecole Nationale Supérieure d'Informatique
Ex : INI (Institut National de formation en Informatique)

2^{nd} Year Specialty SIQ G02, 2CS SIQ2

Intelligent and Communicating Systems, ICS Project

Deaf-friendly baby monitor

Realised by:

- *Sellal Feriel Amel*
- *Hadjer Narjas*

Supervised by:

- *Pr.Sehad*

2022-2023

Contents

List of Figures

1 Introduction

About 5% of the world's population is considered to be having hearing impairment, and more than 2 million Algerians are estimated to suffer from this disability, making it one of the most frequent disabilities in the world. Hard hearing or deaf people from all ages face a lot of issues regarding simple daily tasks, issues that are mostly invisible to people who do not suffer from this disability, making it hard for them to be totally independent in multiple life aspects.

After observing our surroundings and reflecting on potentially significant needs, we were triggered by a specific issue deaf and hard hearing people face: **taking care of toddlers and babies**. Toddlers and babies tend to express their emotions and needs primarily in a vocal way, which can be difficult for their parents to perceive, specifically at night, so most deaf or hard hearing parents or caretakers face the obligation to solicit other people, mostly family members, to be the intermediate between them and the young children in order to be able to properly respond to their needs.

Aiming to make life easier for people with disabilities should be one of our primary concerns as a society, and a matter we should work on. From that comes the motive of our idea, as part of the **ICS "Intelligent and Communicating System"** module and with the aim of creating an original, useful, efficient IOT product and taking in consideration the problematic we pointed out earlier, we present the following project idea: **A deaf-friendly baby monitor**.

Through this report, we have explained the idea of our product, its detailed operating process as well as the whole process of its creation, ranging from the exhaustive list of all its components, to their hardware wiring and the software programming of our systems. Finally, we concluded this report with all the potential prospects we consider for our product.

2 Project's Description

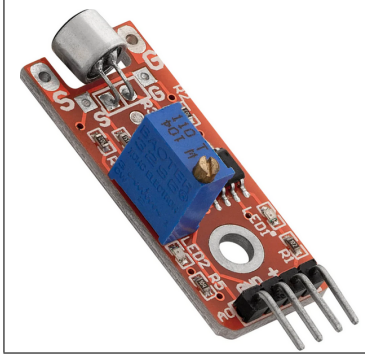
Deaf parents usually struggle in our society to be totally independent in terms of their children's care, especially at early ages. Deaf or hard-hearing parents struggle to watch their babies at night since at that age they tend to wake up a lot and cry, a sign that can not alert a deaf or hard-hearing parent. The objective of our project is to create a tool that might help deaf and hard-hearing people to gain a little more autonomy in terms of child care: by creating a deaf-friendly baby monitor, that consists of two systems: **a monitor** as a hearing device put near the babies' cribs for example, that captures their crying sounds, and **a smart bracelet** that should be worn by the parents or the caretakers of the child on their wrists. When a crying sound is captured by the monitor, a signal is sent to the bracelet which produces a vibration to alert its wearer.

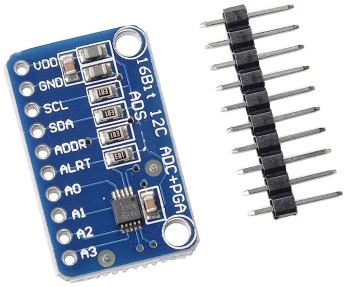
3 Inventory of Material

As mentioned before, our project consists of two systems. Both of them required a list of components that are listed below.

3.1 The monitor

The following is a list of the electronic components used to realize the monitor.

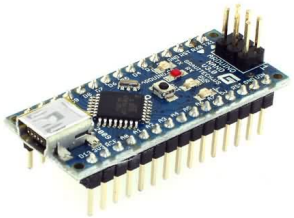
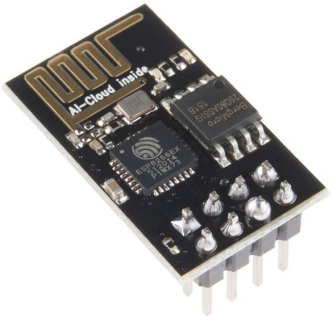
Components	Description	Objective	Quantity
	Raspberry Pi 4	The Raspberry Pi 4 is a powerful single-board computer that serves as the main micro-controller for the monitor. It provides the processing power, memory, and I/O capabilities required to run the program and interface with other components.	1
	Sound sensor KY-038	The sound sensor KY-038 Module is a sensor that captures sound. It typically consists of a microphone that converts sound waves into electrical signals. This module detects sound levels from 50Hz to 13kHz in a 1 meter diameter space and provides an analog output.	1

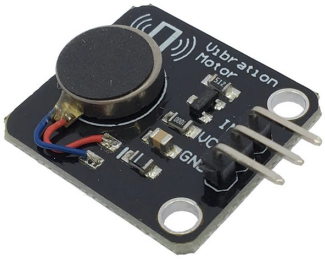
	ADC ADS1115	The ADC ADS1115 is an analog-to-digital converter. It is used to convert the analog output from the sound sensor into a digital signal that the Raspberry Pi can process. The ADS1115 provides a high-resolution of 16 bits. This means it can provide 2$\hat{1}$6 (65,536) different digital values for the range of analog input voltage it can measure. The 16-bit resolution allows for precise conversion of analog signals into digital values, offering a higher level of accuracy compared to lower-resolution ADCs.	1
	LED	One LED who is always lit when the monitor is on (power indication) and one LED that blinks when an alert is being transmitted to the bracelet.	2
	Resistors	To protect the micro-controller. (220 Ohms)	2
	Breadboard	To connect the components.	1

	Box design	A plastic black case with a refined design suitable for our monitor.	1

3.2 The smart bracelet

The following is a list of the electronic components used to realize the smart bracelet.

Components	Description	Objective	Quantity
	Arduino Nano	The micro-controller for the bracelet. The Arduino Nano is a small-sized microcontroller board based on the ATmega328P microcontroller. It has 14 digital input/output pins, 8 analog input pins, and various communication interfaces such as UART, I2C, and SPI. It operates at 5V and can be programmed using the Arduino IDE. The Nano is commonly used for prototyping and small-scale projects that require a compact microcontroller with sufficient I/O capabilities.	1
	ESP8266-01	The ESP8266-01 is a Wi-Fi module that integrates the ESP8266 chip, a low-cost Wi-Fi system-on-a-chip (SoC). It provides Wi-Fi connectivity to micro-controllers and other devices. The ESP8266-01 module offers a UART interface for communication with the host micro-controller, and it can be programmed using the Arduino IDE or other supported frameworks. It supports TCP/IP networking protocols and can connect to Wi-Fi networks, enabling devices to communicate over the internet or local network.	1

	Vibration motor	A module to create vibrations. It is a compact, cylindrical device that contains an eccentric weight and a motor. When the motor rotates, the eccentric weight generates an unbalanced force, causing the motor and the attached device to vibrate. The intensity of the vibration can be controlled by adjusting the motor speed or the weight distribution. In the context of the project, the vibration motor is used in to provide tactile feedback to the wearer.	1
	Battery	A source to power the system, between 7v and 12v.	1
	LED	A LED who is always lit when the bracelet is on (power indication).	1
	Resistors	To protect the microcontroller. (220 Ohms)	1
	Bracelet design	A comfortable and simple black bracelet design adapted to our situation.	1

4 Operating Process

Our monitor is designed to be put in a 1 meter diameter space away from the target. As the crying sound of a baby is generally between **300 Hz** and **600 Hz**, we defined a threshold of 300 Hz that should be exceeded in order for the monitor to send an alarming signal to the bracelet. Once the alert transmitted, the bracelet starts vibrating until stopped or the duration exceeds 30 seconds. The figure below resumes the general process of our system.

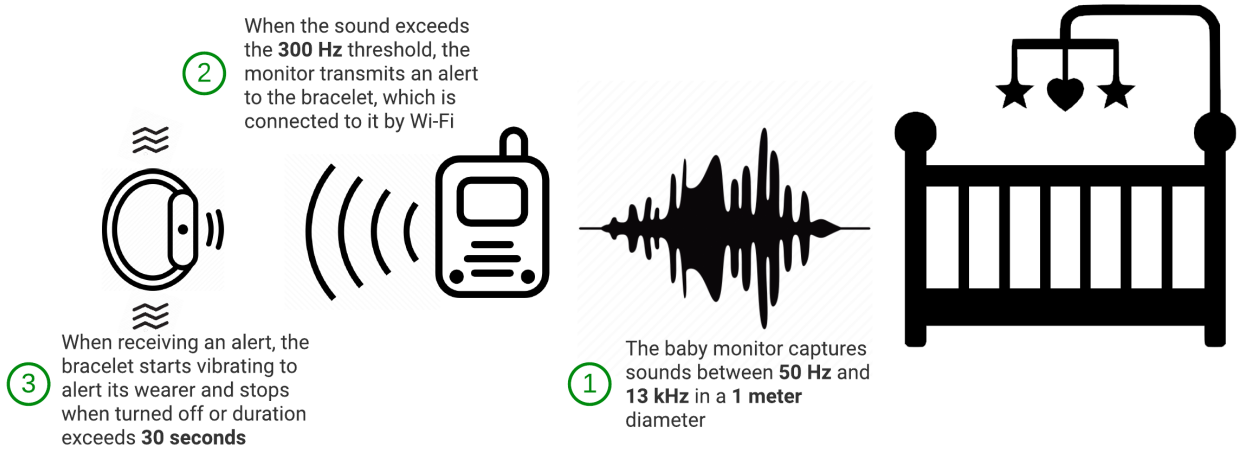


Figure 1: Global architecture of the system

In order to create the wanted effect, we needed to find an efficient way to connect our components as the communication between the monitor and the bracelet had to be in real-time, quick and reliable. For that, we choose to connect the two systems by Wi-Fi over the **MQTT** protocol.
The monitor hosts the MQTT broker to which the sound sensor module publishes on a topic named **"Sound"**. In order to reduce the number of publications, we programmed the publication to be carried out only if the sound's frequency exceeds 300 Hz.
Our bracelet being the subscriber to the **"Sound"** topic, it gathers the alarms and alerts the wearer of the bracelet of the situation.
As we choose the Arduino Nano to be the bracelet's micro-controller, we had to add a Wi-Fi module (ESP8266) to it in order to get access to the broker. The Wi-Fi module is set to communicate with the Arduino micro-controller using the **UART** protocol.
The following diagram summarizes the global communication process and means used in our product.

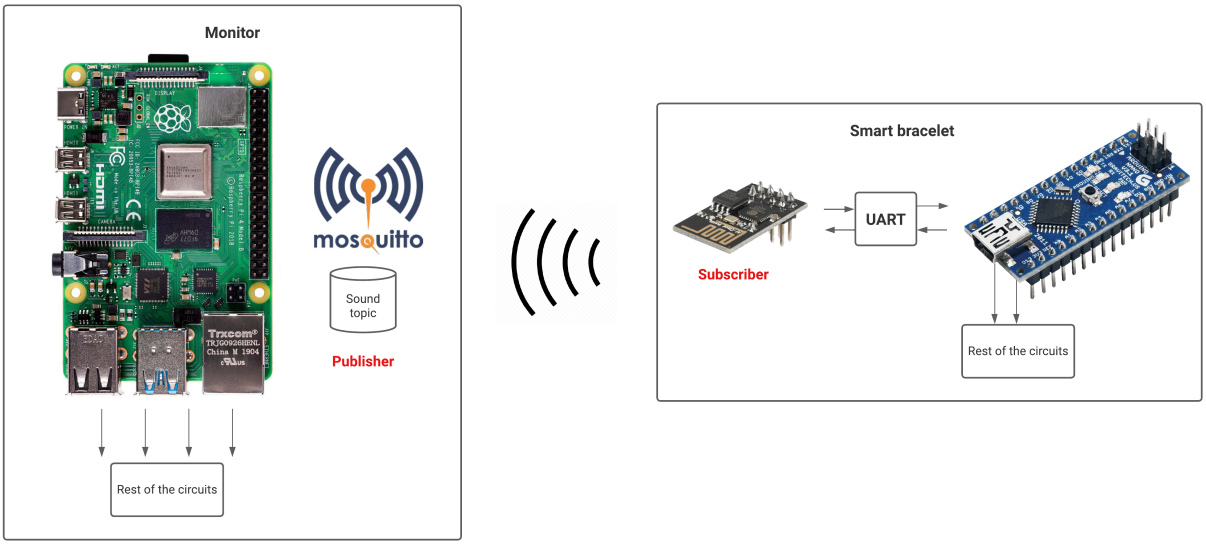


Figure 2: Global communication process between the systems

5 Electronic Schemes

Now that all of the components have been listed, the global working process of our product known and the communication protocols discovered, we can finally address the matter of the wiring of our different components. You can see below both the circuit in the monitor and the one in the smart bracelet.

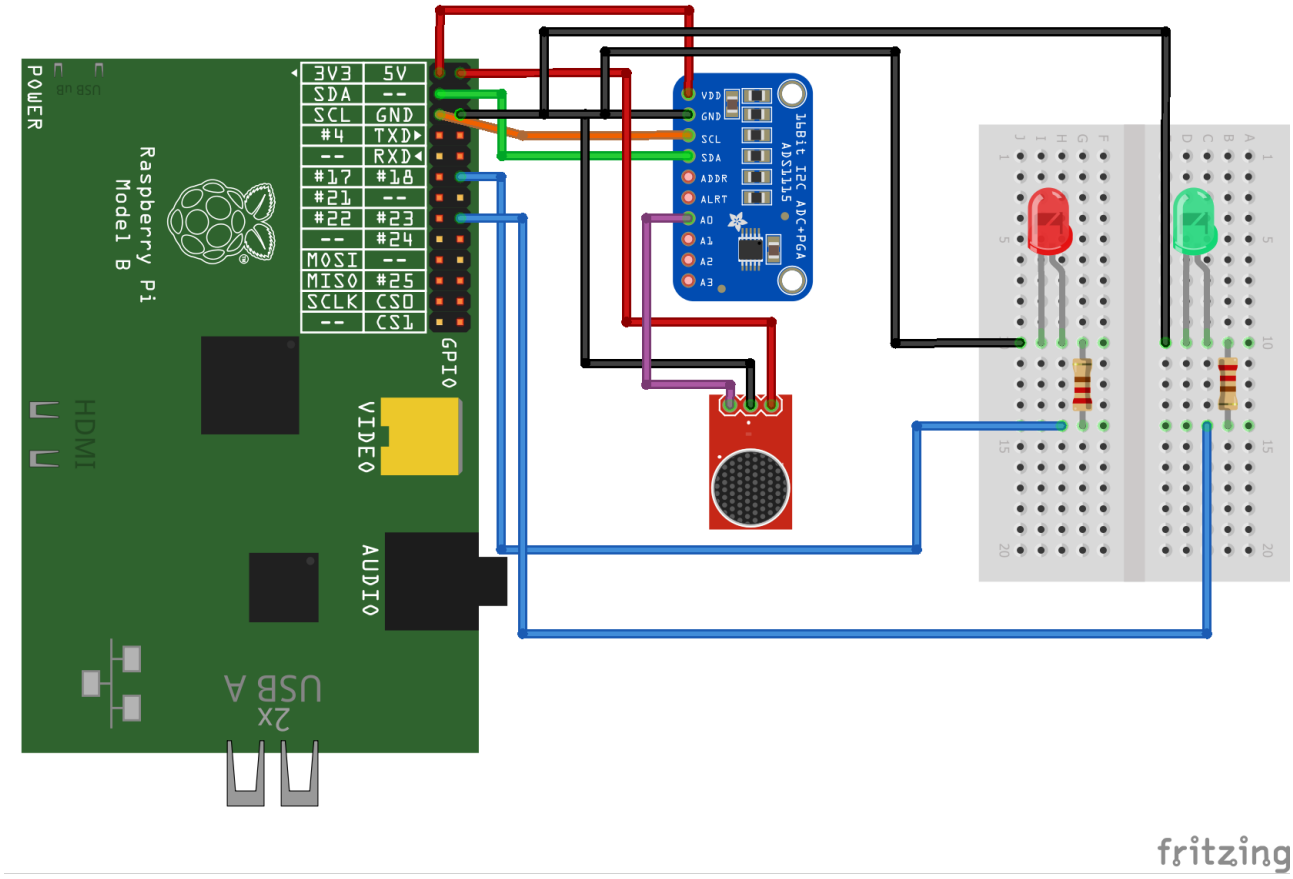


Figure 3: Circuit in the monitor

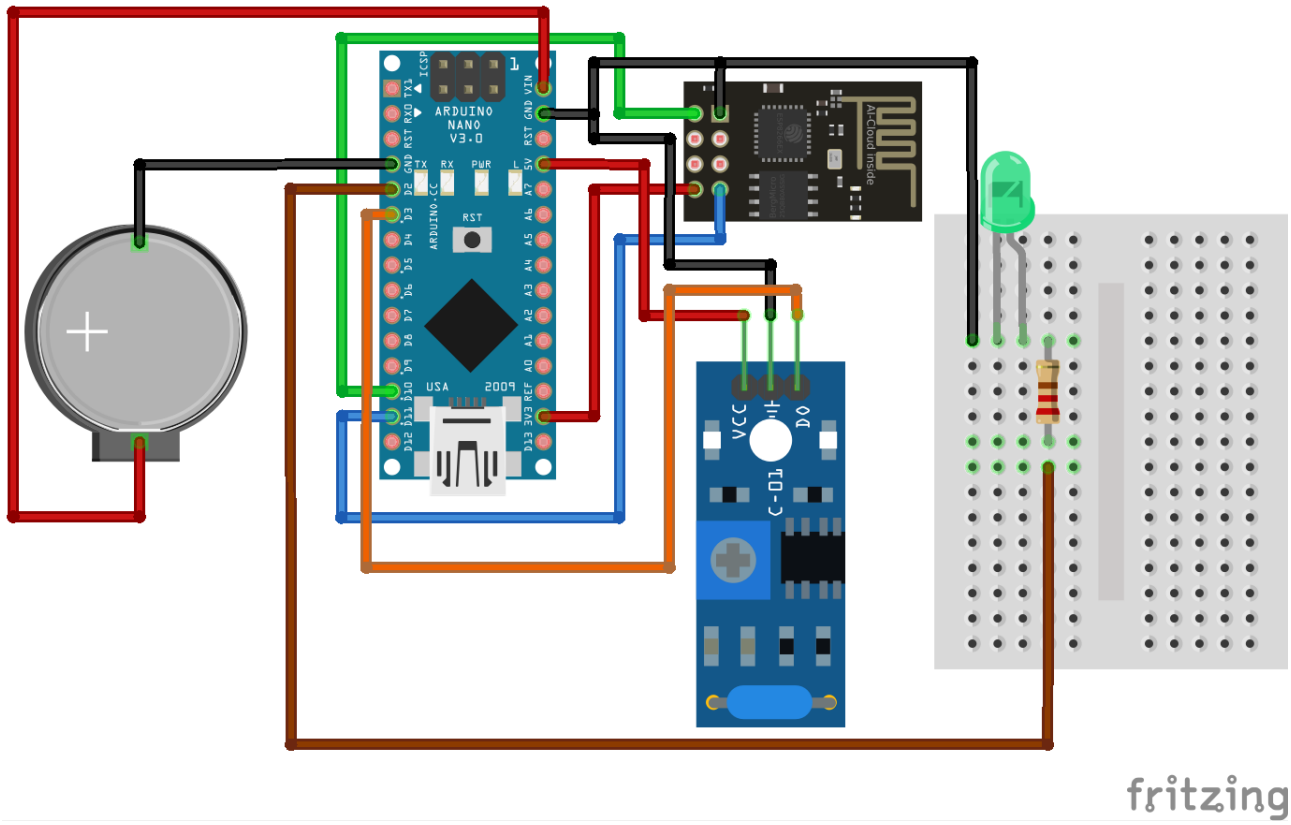


Figure 4: Circuit in the smart bracelet

6 Code

The previous listed micro-controllers needed to be implemented with the correct software to realize the wanted tasks. For that, we produced a code for the **Raspberry Pi 4** micro-controller, the **Arduino nano** one and the **ESP8266** Wi-Fi module.

```python
import time
import paho.mqtt.client as mqtt
import RPi.GPIO as GPIO
import Adafruit_ADS1x15


# MQTT network configuration
mqttServer = "MQTT_Broker_IP_Address"  # Replace with MQTT broker IP address
mqttPort = 1883  # Default MQTT port
mqttTopic = "sound"


# GPIO configuration
led_pin = 18  # GPIO pin number used for the LED
GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT) # Setup the led pin as output
GPIO.output(led_pin, GPIO.LOW)  # Turn off the LED at startup


# MQTT client configuration
client = mqtt.Client()


# MQTT client connection
def on_connect(client, userdata, flags, rc):
    client.subscribe(mqttTopic)


def on_message(client, userdata, msg):
    pass  # this function empty cause we don't receive MQTT messages

```

```
32
33 # MQTT client publishing function
34 def publish_message(msg):
35     client.publish(mqttTopic, msg)
36
37
38 client.on_connect = on_connect
39 #client.on_message = on_message
40
41
42 client.connect(mqttServer, mqttPort, 60)
43 client.loop_start()
44
45
46 def read_sound_level():
47     adc = Adafruit_ADS1x15.ADS1115()
48     gain = 1  # Gain factor (1x)
49     sound_level = adc.read_adc(0, gain=gain) * (3300.0 / 32767.0)  # Read analog value in Hz
       and convert to mV
50     return sound_level
51
52
53 threshold = 300  # Crying threshold in Hz
54 blink_duration = 1  # Duration in seconds during which the LED will blink
55
56
57 def blink_led():
58     start_time = time.time()
59     end_time = start_time + blink_duration
60     while time.time() < end_time:
61         GPIO.output(led_pin, GPIO.HIGH)  # Turn on the LED
62         time.sleep(0.5)  # LED on time
63         GPIO.output(led_pin, GPIO.LOW)  # Turn off the LED
64         time.sleep(0.5)  # LED off time
65
66
67 try:
68     while True:
69         sound_level = read_sound_level()  # Get the sound level
70         if sound_level > threshold:  # Check if sound level exceeds the threshold
71             blink_led()  # Blink the LED
72             publish_message("Sound Detected!")  # Publish an MQTT message
73         time.sleep(1)  # Wait for 1 second
74 except KeyboardInterrupt:
75     client.loop_stop()
76     client.disconnect()
77     GPIO.cleanup()  # Clean up used GPIOs
```

Listing 1: Raspberry Pi program

```
1  #include <ESP8266WiFi.h>
2  #include <PubSubClient.h>
3
4  // WiFi
5  const char* ssid = "SSID";
6  const char* password = "PASSWORD";
7
8  // MQTT Broker
9  const char *mqtt_broker = "IP_ADDRESS";
10 const char *topic = "Sound";
11 const int mqtt_port = 1883;
12
13 WiFiClient espClient;
14 PubSubClient client(espClient);
15
16 void setup() {
17   // Set software serial baud to 115200;
18   Serial.begin(9600);
19   // connecting to a WiFi network
20   WiFi.begin(ssid, password);
21   while (WiFi.status() != WL_CONNECTED) {
22       delay(500);
23       Serial.println("Connecting to WiFi..");
24   }
25   Serial.println("Connected to the WiFi network");
26   //connecting to a mqtt broker
```

```
  client.setServer(mqtt_broker, mqtt_port);
  client.setCallback(callback);
  while (!client.connected()) {
      String client_id = "esp8266-client-";
      client_id += String(WiFi.macAddress());
      Serial.printf("The client %s connects to the public mqtt broker\n", client_id.c_str());
      if (client.connect(client_id.c_str(), NULL,NULL)) {
          Serial.println("Public emqx mqtt broker connected");
      } else {
          Serial.print("failed with state ");
          Serial.print(client.state());
          delay(2000);
      }
  }
  // Publish and subscribe
  client.publish(topic, "hello emqx");
  client.subscribe(topic);
}

void callback(char *topic, byte *payload, unsigned int length) {
// Send the Serial line content
  for (int i = 0; i < length; i++) {
      Serial.print((char) payload[i]);
  }
}

void loop() {
  client.loop();
}
```

Listing 2: ESP8266 program

```
#include <SoftwareSerial.h>
#define ON_LED 2 //LED pin
#define VIBRATION 3 //Vibration motor pin

//This is the serial communication line between the Arduino Nano and the ESP8266
SoftwareSerial mySerial(10, 11); //RX, TX

void setup() {
  //Open serial communications and wait for port to open
  Serial.begin(9600);
  /*Set the data rate for the SoftwareSerial port*/
  mySerial.begin(9600);
  pinMode(ON_LED, OUTPUT); /*Define the LED pin as output*/
  pinMode(VIBRATION, OUTPUT); /*Define the vibration motor pin as output*/
  digitalWrite(ON_LED, HIGH); /*As long as the bracelet is on, the LED is on*/
}

void loop() { //Run over and over
  //If an element is on the serial line
  if (mySerial.available()) {
    char ser = mySerial.read();
    digitalWrite(VIBRATION, HIGH); //Produce a vibration
    delay(2000);
    digitalWrite(VIBRATION, LOW); //Stop the vibration
    }
}
```

Listing 3: Arduino Nano program

7 Prospects

Our product is a product set to be developed. We can consider many improvements for it, so it can be more useful and versatile such as the latter:

- **Smaller bracelet**: Our bracelet size could easily be reduced with smaller components, like replacing the Arduino Nano with an Arduino Lilly Pad, which is designed to be put on smart bracelets and watches.
- **Mobile application**: We can design a mobile application in order to centralize the data collected by each bracelet and personalize a dashboard for the parents and caretakers with statics about the baby's activity at night, the hours at which it wakes up more...

- **Wider range**: Our product is set to be used on relatively small areas, but the latter could be improved by having more robust components in terms of distance and intensity range.
- **Multiple purpose system**: In fact, by adding other sensors and modules to our system, we could make a wider use of our product. For example we can add a movement sensor to indicate when the target is moving, a temperature sensor to perceive the temperature in the room...
- **Extend features on the bracelet**: We can extend our product's features by adding other components like a LCD on the bracelet, that would display messages for its wearer, we can also add a rechargeable battery and its module to diminish the battery use...

All of this ideas can easily be implemented with the adequate electronic components, more precise means of wiring and software programming.

8 Conclusion

In conclusion, our project aims to address the challenges faced by deaf and hard hearing parents in caring for their young children. We recognized the difficulties they encounter in perceiving their babies' vocal cues, especially during nighttime. To enhance their independence and provide a solution to this issue, we proposed the development of a deaf-friendly baby monitor.

The deaf-friendly baby monitor consists of two components: a monitor placed near the baby's crib to capture crying sounds, and a smart bracelet worn by the parents or caretakers. When the monitor detects a crying sound, it sends a signal to the bracelet, which generates a vibration to alert the wearer.

By creating this innovative IoT product, we strive to empower deaf and hard hearing parents, enabling them to respond promptly to their children's needs without relying heavily on others for assistance. The aim is to enhance their overall quality of life, promoting independence and inclusivity.

Throughout this report, we provided a detailed description of the project, including the conceptualization, operational process, component lists, hardware wiring, and software programming. Our focus was on creating an original, useful, and efficient solution within the context of the **"Intelligent and Communicating System"** module.

As we conclude this report, we are optimistic about the potential prospects for our product. It has the capacity to make a positive impact on the lives of deaf and hard hearing parents, fostering their autonomy and enhancing their ability to care for their children effectively. Our hope is that this project will contribute to a more inclusive society, where the needs of individuals with disabilities are addressed with compassion and innovation.