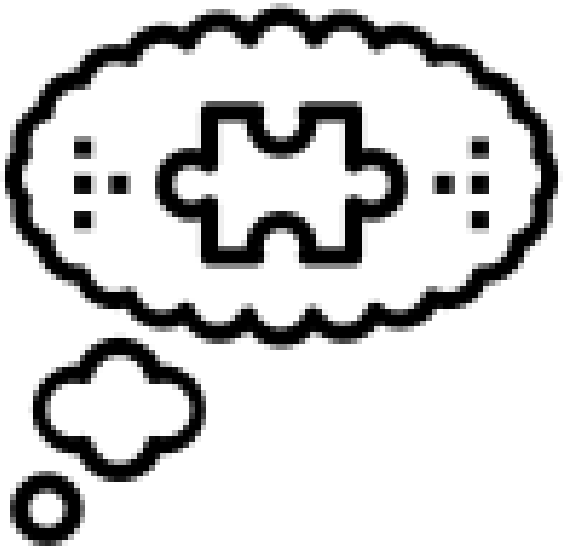
NOISE POLLUTION MONITORING

Phase 5 - Project Documentation & Submission

INTRODUCTION

In this part you will document your project and prepare it for submission.

Document the Noise Pollution Monitoring project and prepare it for submission.



Problem Definition

The project involves deploying IoT sensors to measure noise pollution in public areas and providing real-time noise level data accessible to the public through a platform or mobile app.

The primary objective is to raise awareness about noise pollution and enable informed decision-making.

This project includes defining objectives, designing the IoT sensor system, developing the noise pollution information platform, and integrating them using IoT technology and Python.

Project Objectives

Real-time Noise Pollution Monitoring: Develop a system capable of continuously and accurately measuring noise levels in real-time across designated areas, providing instant access to noise data.

Public Awareness : Launch a public awareness campaign to educate the community about noise pollution, its effects, and ways to reduce it.

Noise Regulation Compliance: Assist regulatory authorities in monitoring and enforcing noise pollution regulations, providing data as evidence for compliance or non-compliance cases..

Improved Quality of Life: Strive to improve the quality of life for residents and businesses in the monitored areas by addressing and mitigating noise pollution issues.

PROJECT REQUIREMENTS

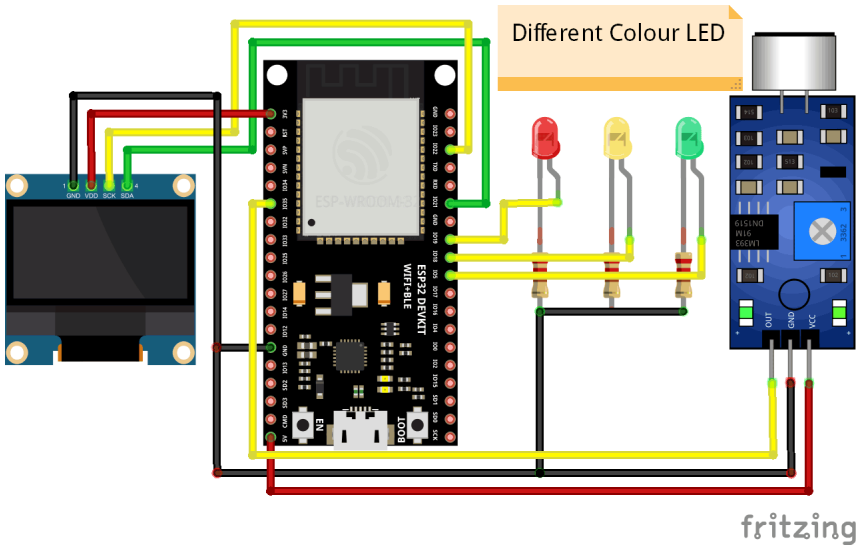
ESP32 Microcontroller

Sound Sensor like KY-038

Bread Board and Jump Wires M/F

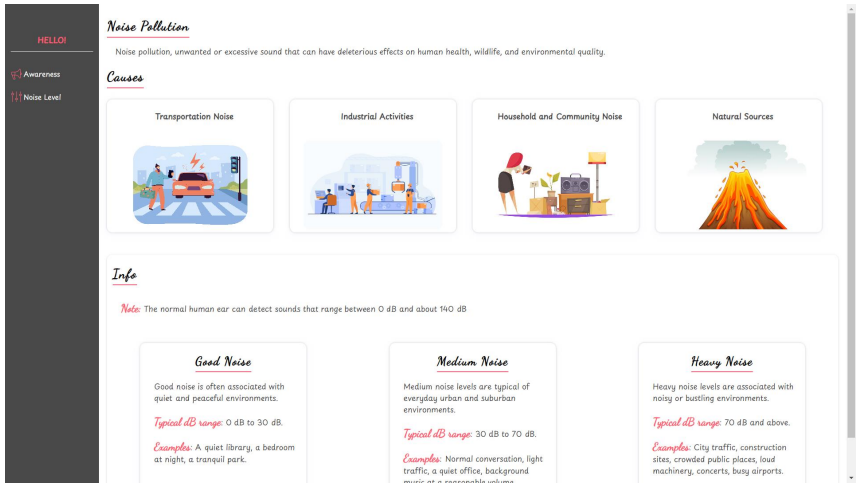
OLED Display

Arduino IDE



Sensor Deployment Steps

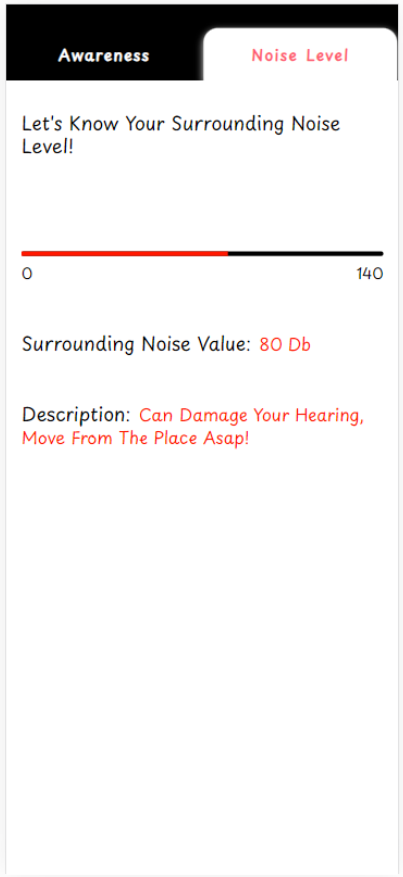
1. Place the KY-038 Sound Sensor, ESP-32 Microcontroller, And OLED Display on the Bread Board
2. Connect the Sound Sensor with the Microcontroller using the Jumper Wires
3. Connect Sound Sensor VCC pin to ESP32 3.3V or 5V
4. Connect Sound Sensor GND to ESP32 GND
5. Connect KY-038 OUT pin to a GPIO PIN on the ESP32
6. Power the ESP32 using a USB Cable



Web Platform

Technologies Used:

- ✓ HTML
- ✓ CSS
- ✓ Java Script
- ✓ Angular Framework



Mobile Application

Technologies Used:

✓ Ionic Framework

Reason To Use Ionic: Framework used to create apps which is compatible on both Android and IOS.

Code Implementation For ESP-32

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define  SCREEN_WIDTH 128
#define  SCREEN_HEIGHT 64
#define  OLED_RESET   -1
#define SENSOR_PIN 35
#define PIN_QUIET 33
#define PIN_MODERATE 25
#define PIN_LOUD 26

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

const int sampleWindow = 50;
unsigned int sample;
```

```
void setup(){
  pinMode(SENSOR_PIN, INPUT); // Set the signal pin as input
  pinMode(PIN_QUIET, OUTPUT);
  pinMode(PIN_MODERATE, OUTPUT);
  pinMode(PIN_LOUD, OUTPUT);
  digitalWrite(PIN_QUIET, LOW);
  digitalWrite(PIN_MODERATE, LOW);
  digitalWrite(PIN_LOUD, LOW);
  Serial.begin(115200);
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;); // Don't proceed, loop forever
  }
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.display();
  digitalWrite(PIN_LOUD, LOW);
}
```

Code Implementation For ESP-32

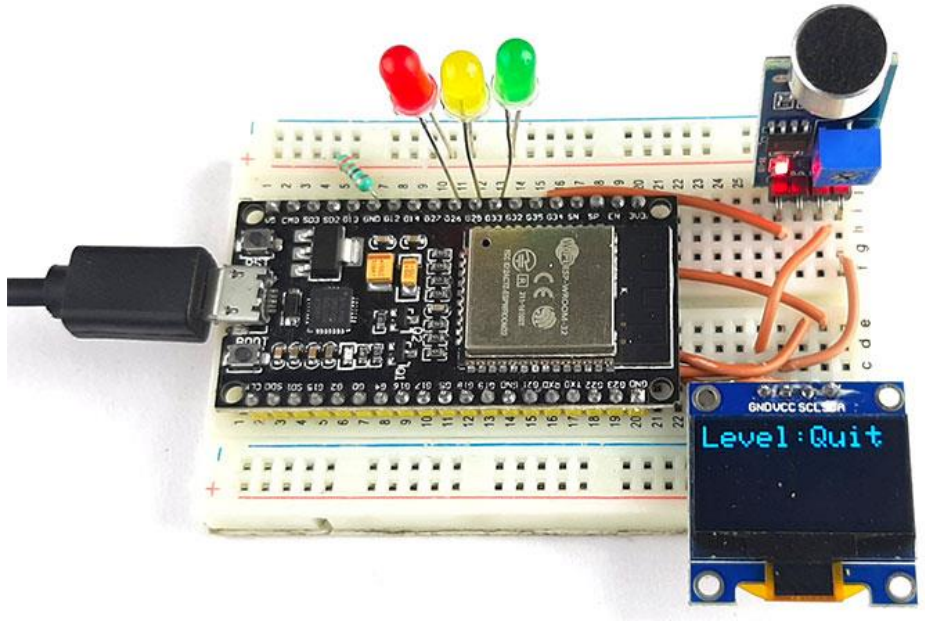
```
unsigned long startMillis = millis();
 float peakToPeak = 0;
 unsigned int signalMax = 0;
 unsigned int signalMin = 1024;
 // collect data for 50 mS
 while (millis() - startMillis < sampleWindow) {
  sample = analogRead(SENSOR_PIN);
  if (sample < 1024) {
   if (sample > signalMax) {
    signalMax = sample;
   }
   else if (sample < signalMin)  {
    signalMin = sample;
   }
  }
 }
peakToPeak = signalMax - signalMin;
 int db = map(peakToPeak, 0, 900, 49, 90);
 Serial.print("\t");
 Serial.println(db);
 display.setCursor(0, 0);
 display.print("Loudness: ");
 display.print(db);
 display.print("dB");
 digitalWrite(PIN_LOUD, LOW);
```

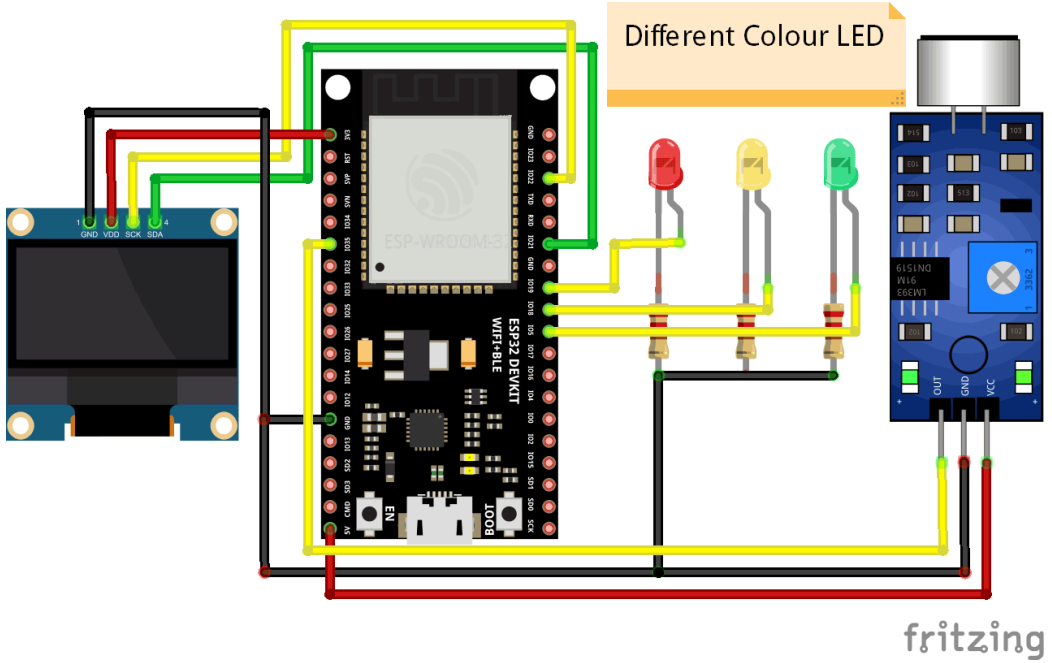
Code Implementation For ESP-32

```
else if (db > 60 && db < 85){
  display.clearDisplay();
  display.setCursor(0, 1);
  display.print("Level:Moderate");
  display.display();
  digitalWrite(PIN_QUIET, LOW);
  digitalWrite(PIN_MODERATE, HIGH);
  digitalWrite(PIN_LOUD, LOW);
}
 else if (db >= 85 && db <= 90){
  display.clearDisplay();
  display.setCursor(0, 1);
  display.print("Level:High");
  display.display();
  digitalWrite(PIN_QUIET, LOW);
  digitalWrite(PIN_MODERATE, LOW);
  digitalWrite(PIN_LOUD, HIGH);
}
```

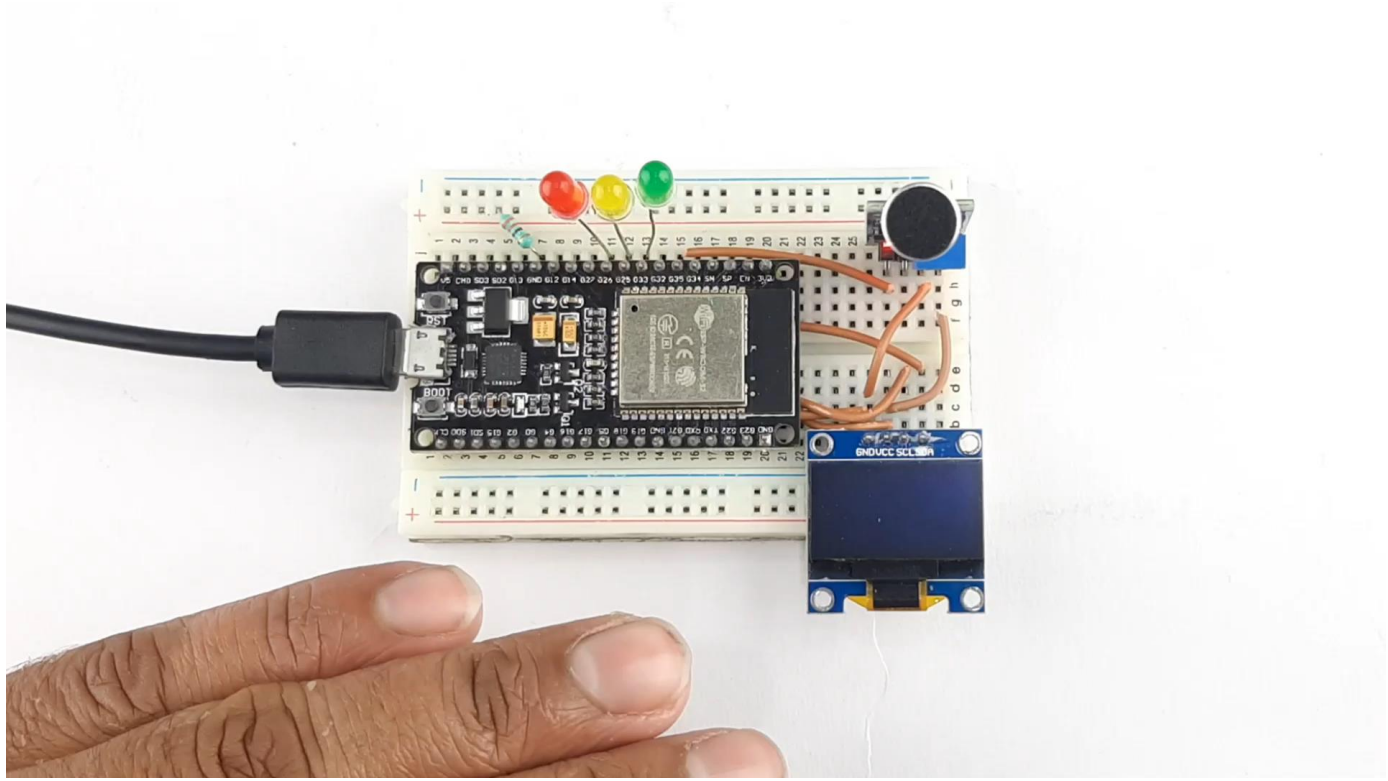
```
else {
   digitalWrite(PIN_QUIET, LOW);
   digitalWrite(PIN_MODERATE, LOW);
   digitalWrite(PIN_LOUD, LOW);
 }
 delay(200);
}
```

Diagrams, Schematics And Screenshots

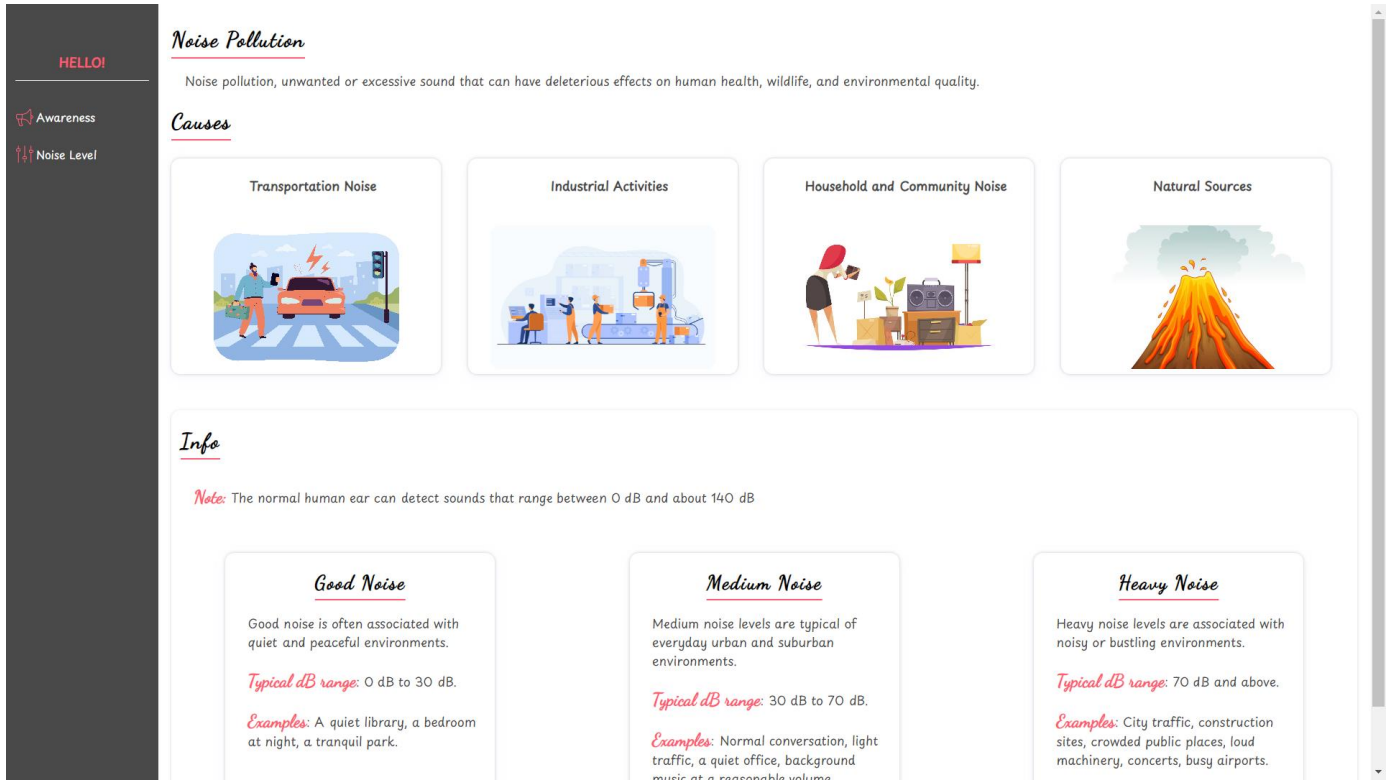




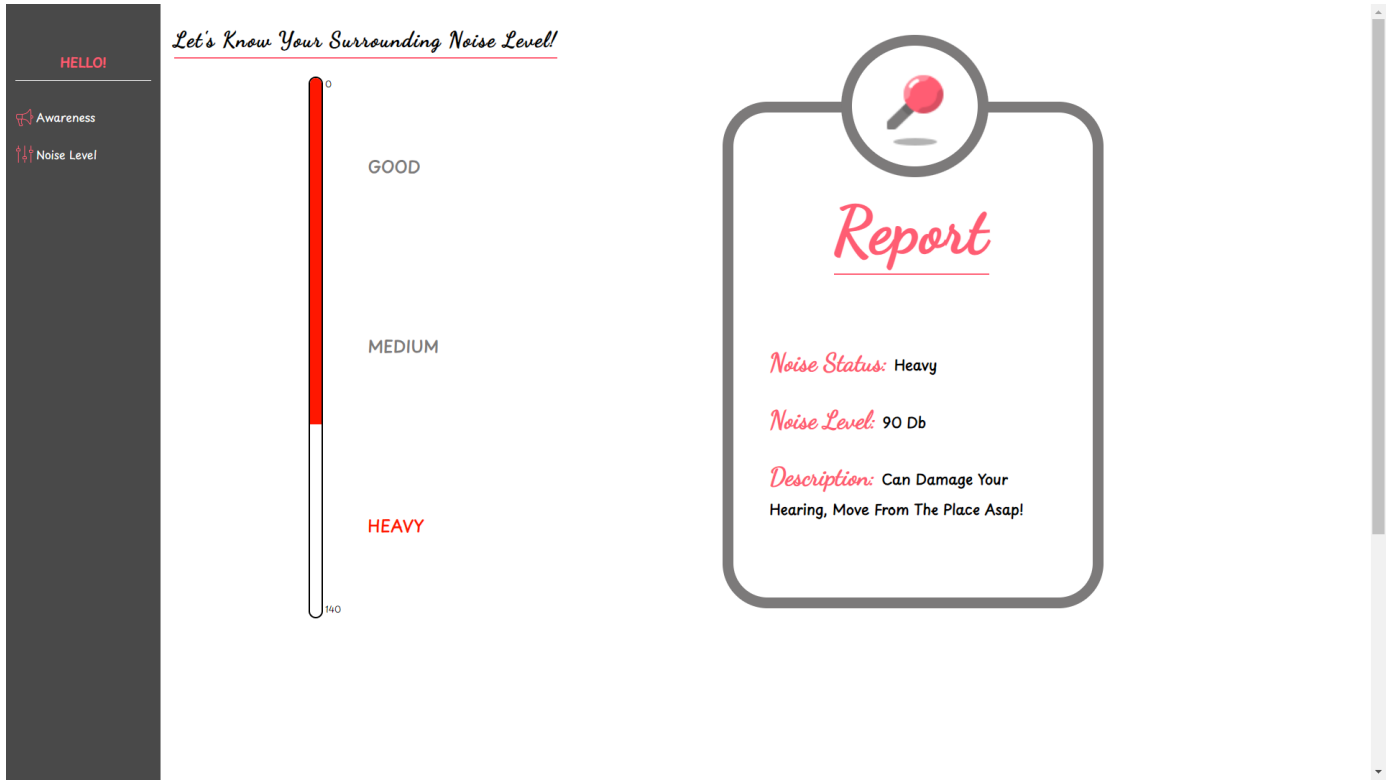
Diagrams, Schematics And Screenshots



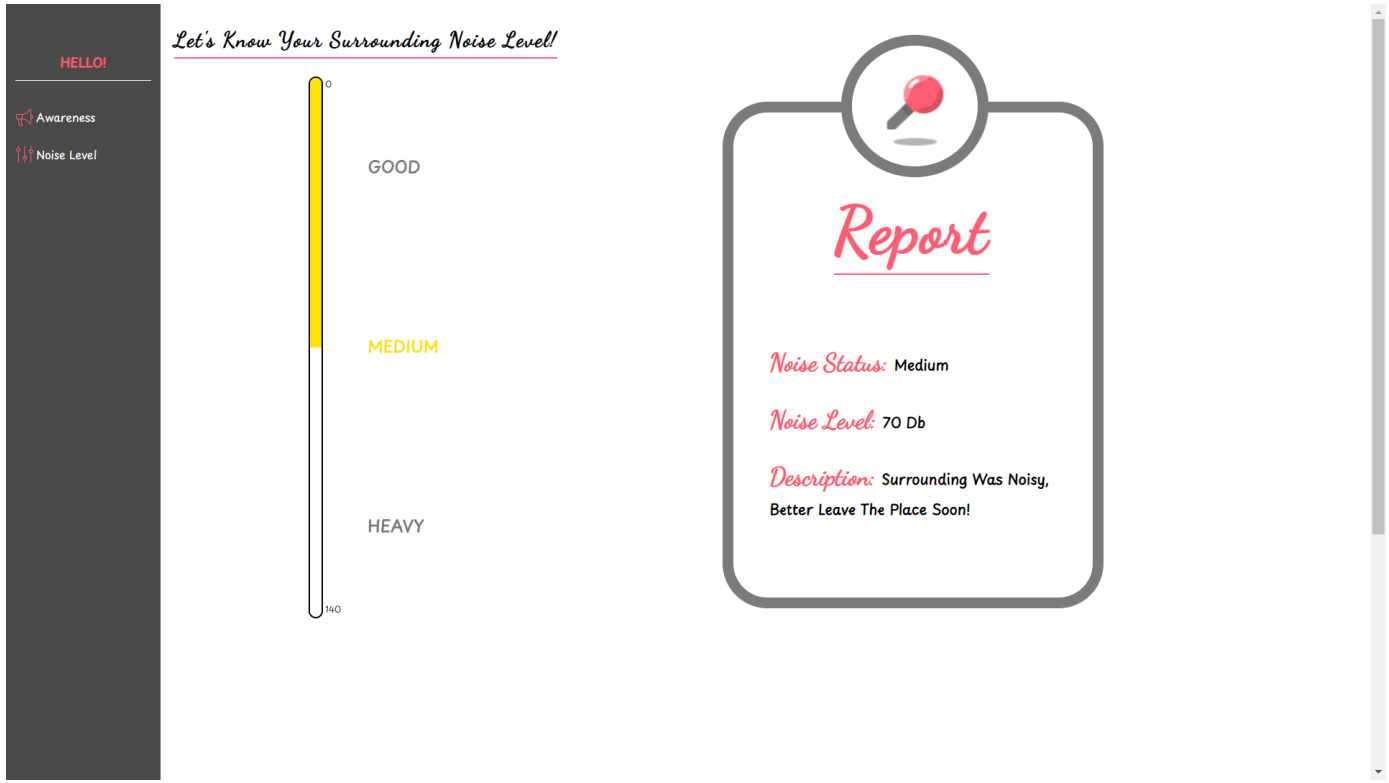
Diagrams, Schematics And Screenshots



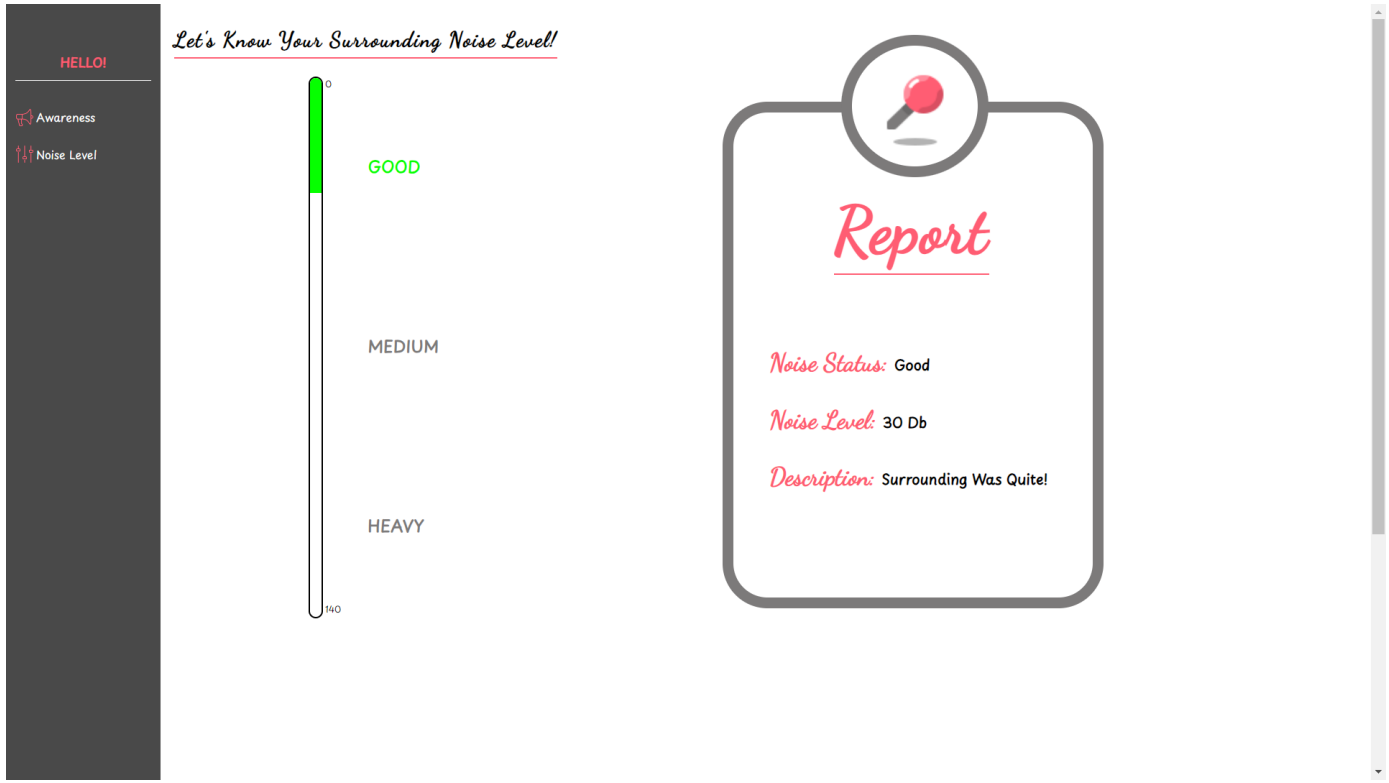
Diagrams, Schematics And Screenshots



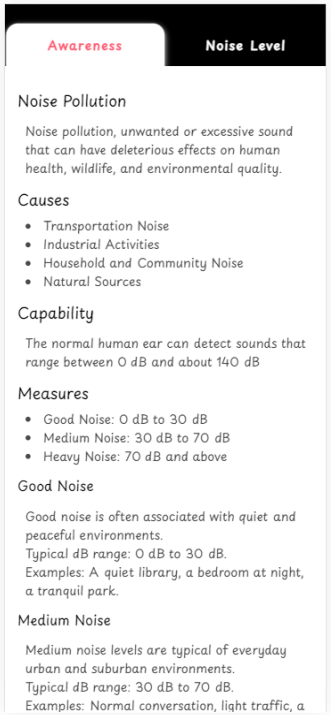
Diagrams, Schematics And Screenshots

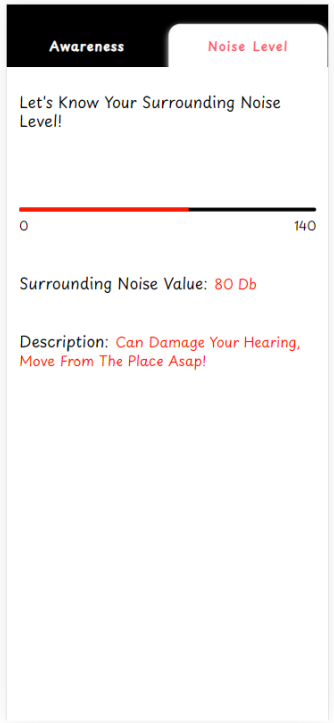


Diagrams, Schematics And Screenshots

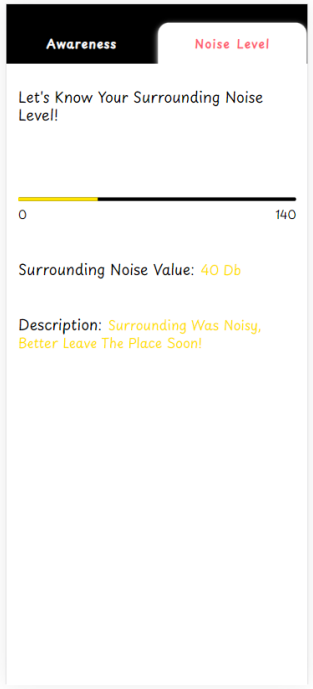


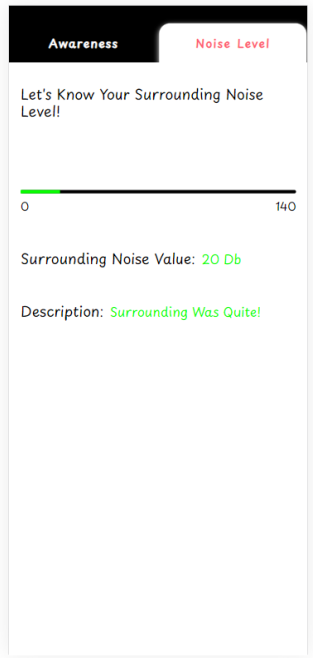
Diagrams, Schematics And Screenshots

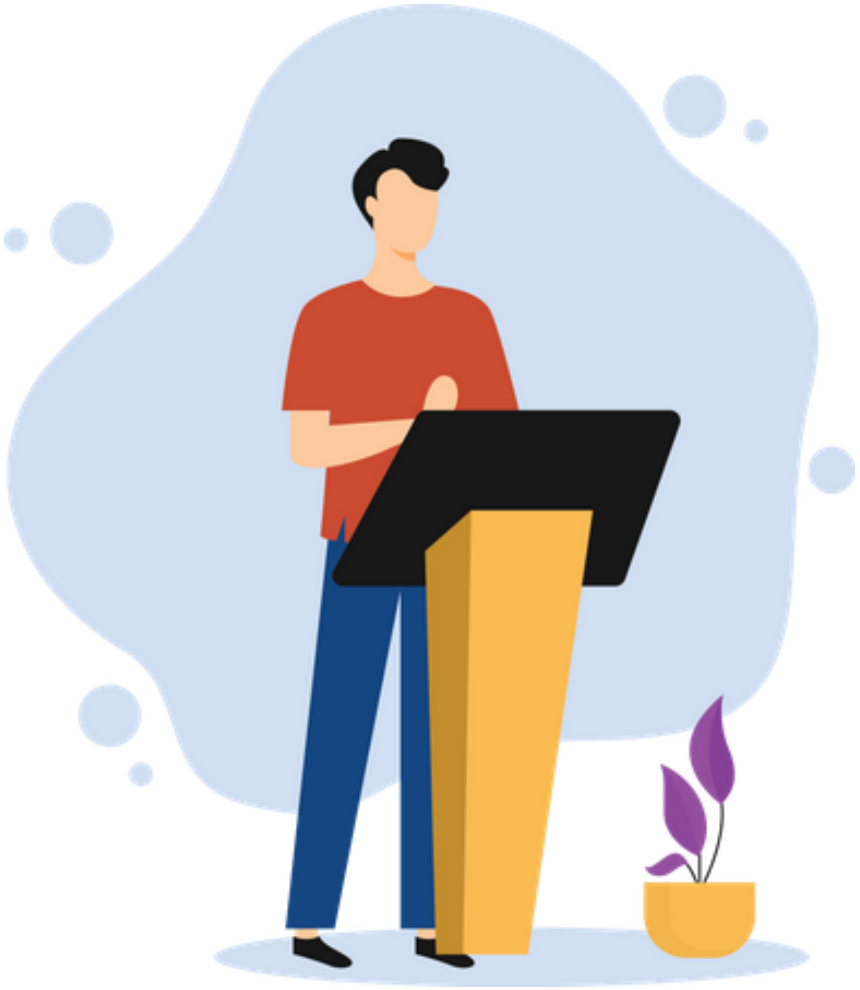




Diagrams, Schematics And Screenshots



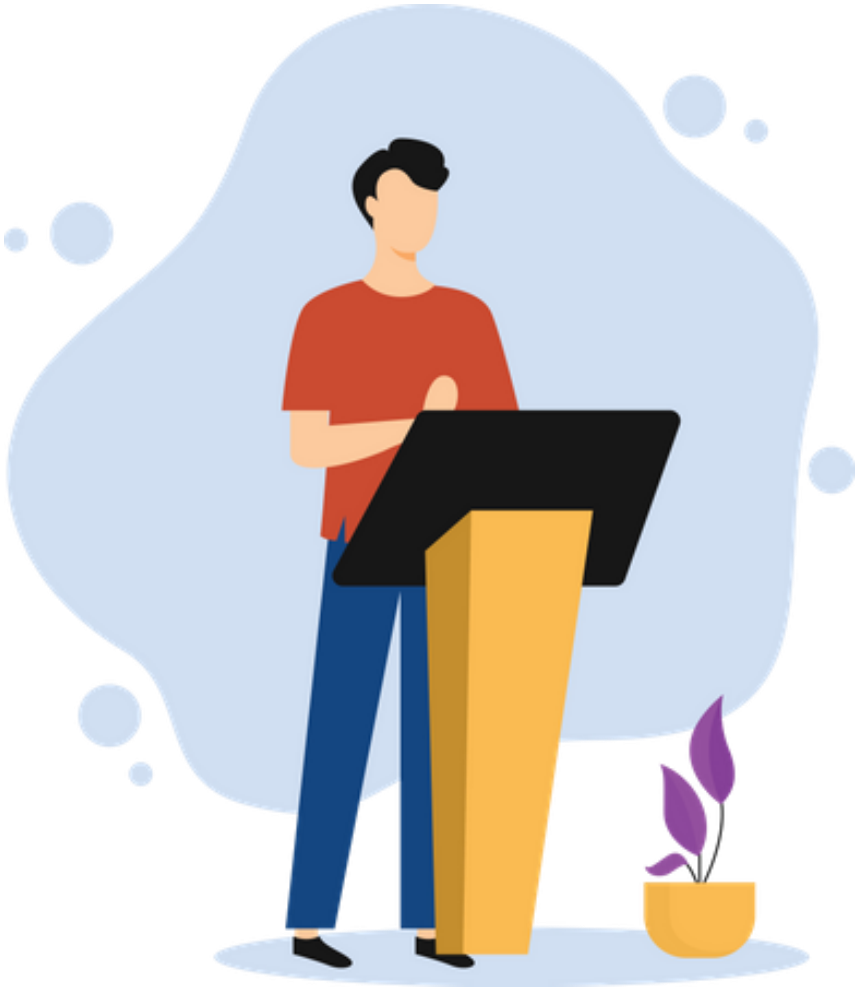




How It Promotes Public Awareness

Both the Platform Provides a Information about Noise Pollution, How Much Db that Human Ear Can Hear, What Are The Levels of Good, Medium and Heavy Noise.

And both of those provides access to real time Decibel Data from the Sensors.



How It Promotes Public Awareness

1. Immediate Feedback
2. Education and Awareness
3. Noise Source Identification
4. Public Health Benefits
5. Behavioural Changes

THANK YOU