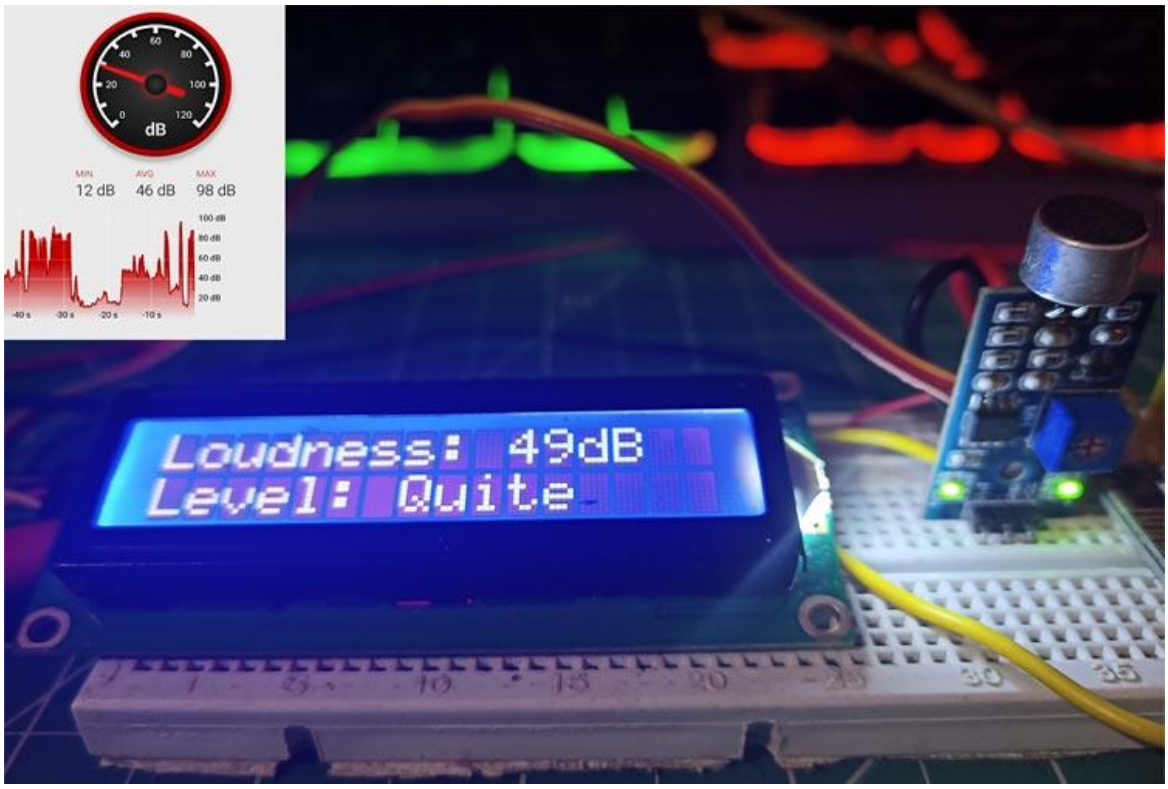
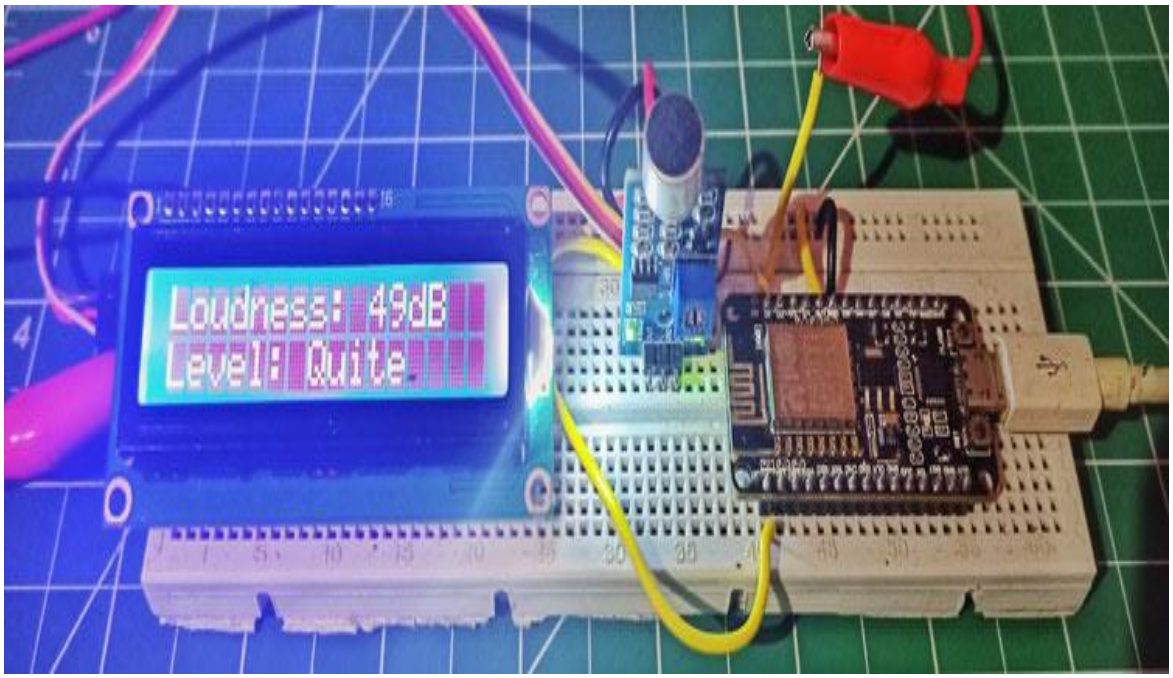
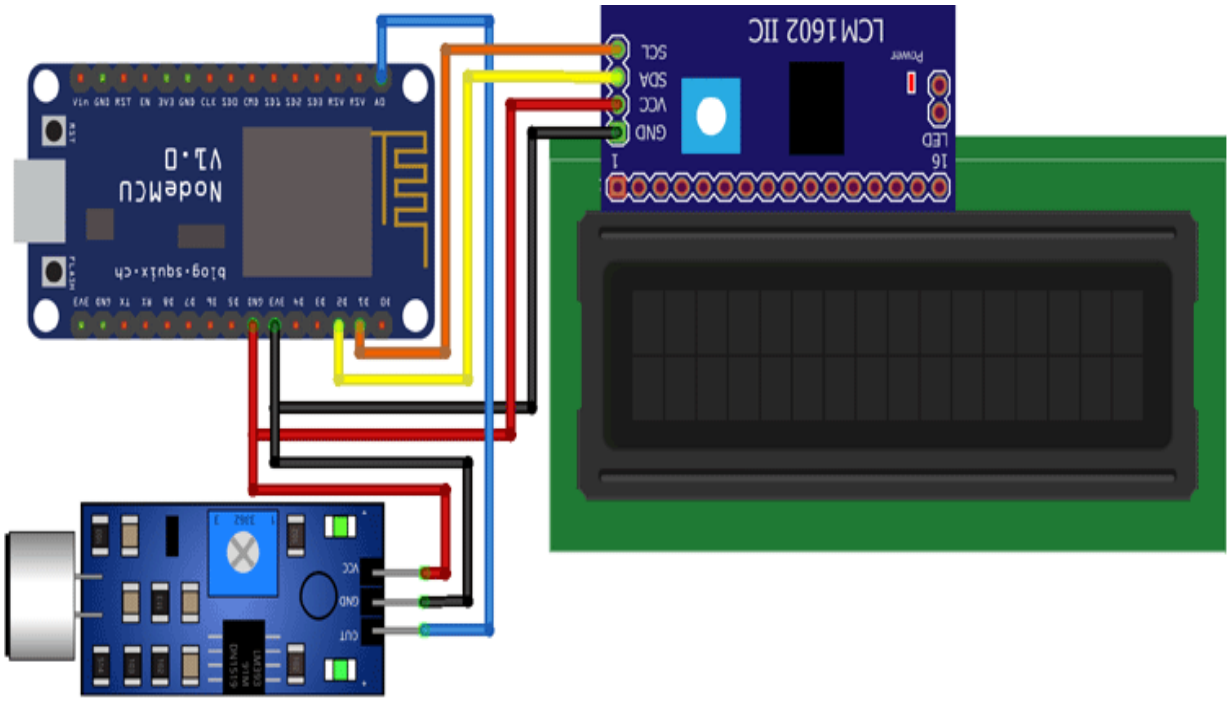
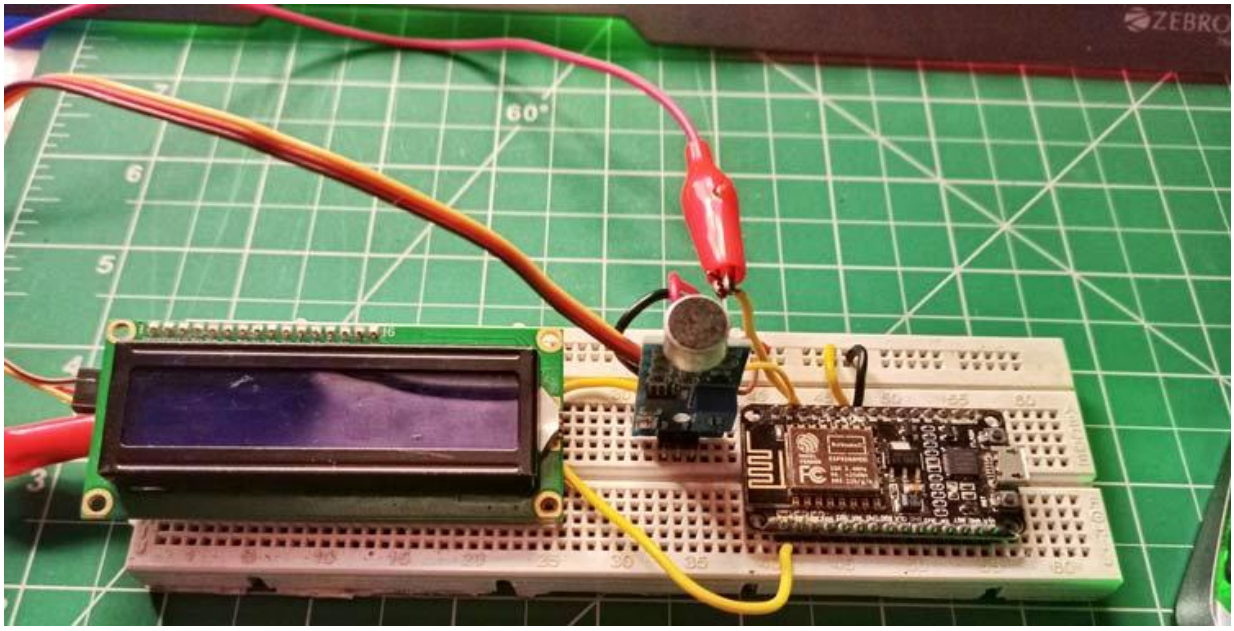
IOT BASED NOISE POLLUTION MONITORING SYSTEM

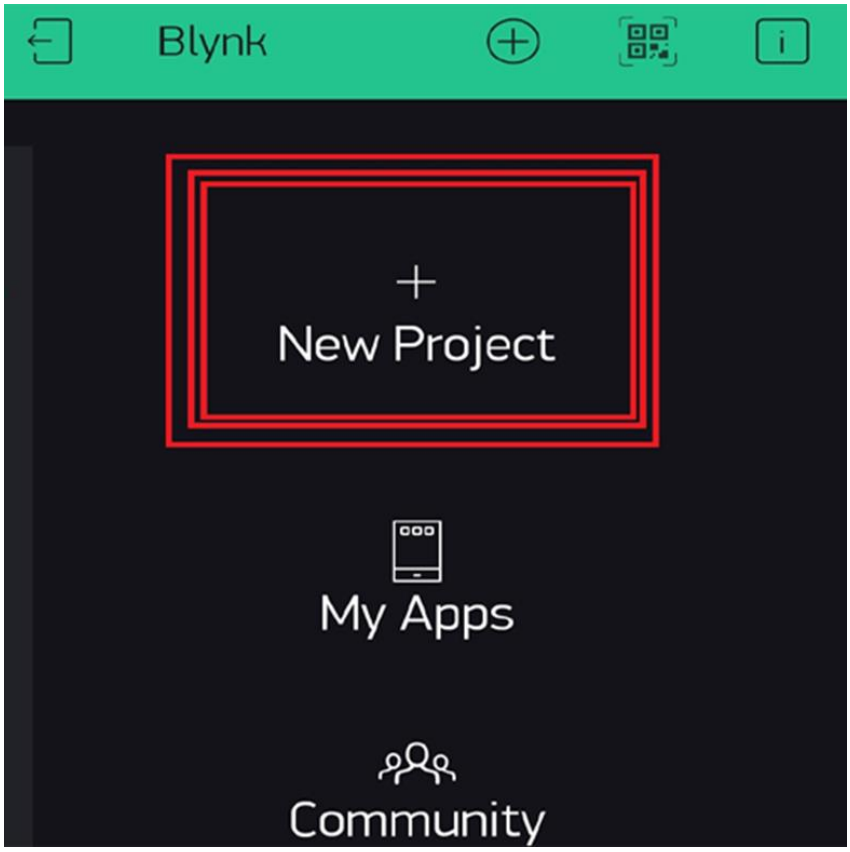


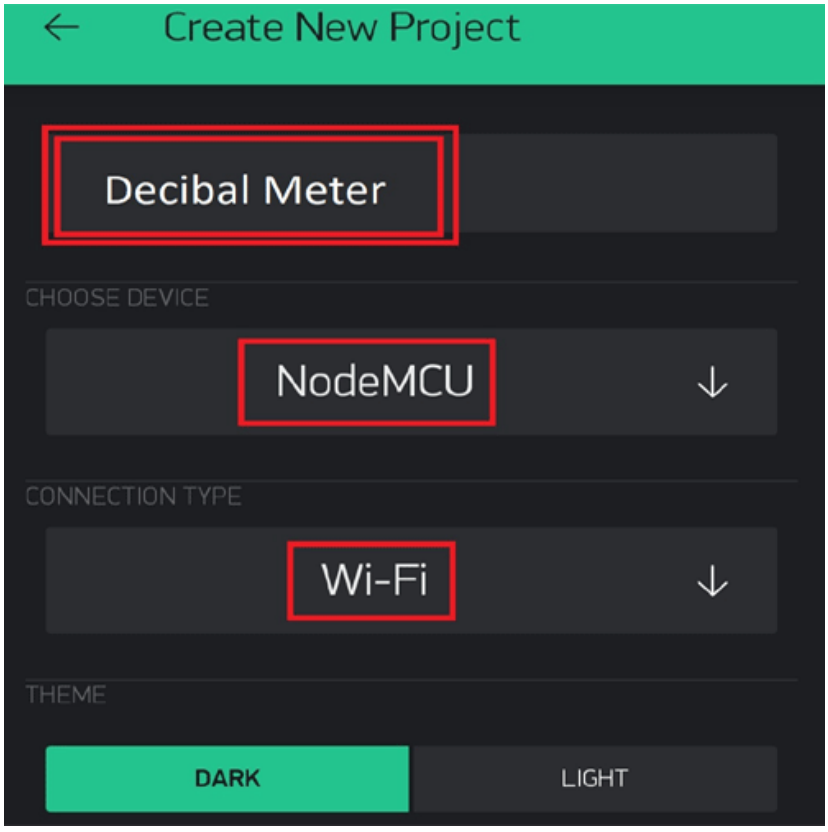


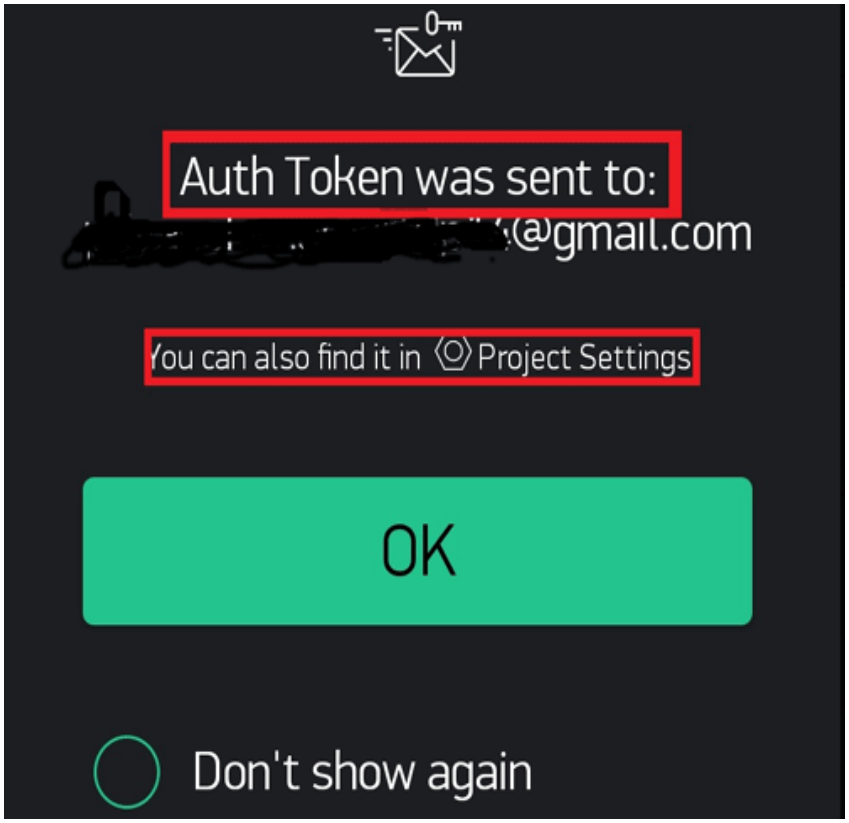
Sound level meters are commonly utilized in sound pollution studies for the quantification of various sorts of noise, especially for industrial, environmental, mining, and aircraft noise. The reading from a sound level meter doesn't correlate well to human-perceived loudness, which is best measured by a loudness meter. Specific loudness may be a compressive nonlinearity and varies at certain levels and certain frequencies. These metrics also can be calculated in several other ways. Here we are going to make an **IoT based decibel meter** that will **measure the sound in decibels(dB)** using a sound sensor and display it to the LCD display along with that, it will also be pushing the readings to the **Blynk IoT platform** making it accessible from across the world.

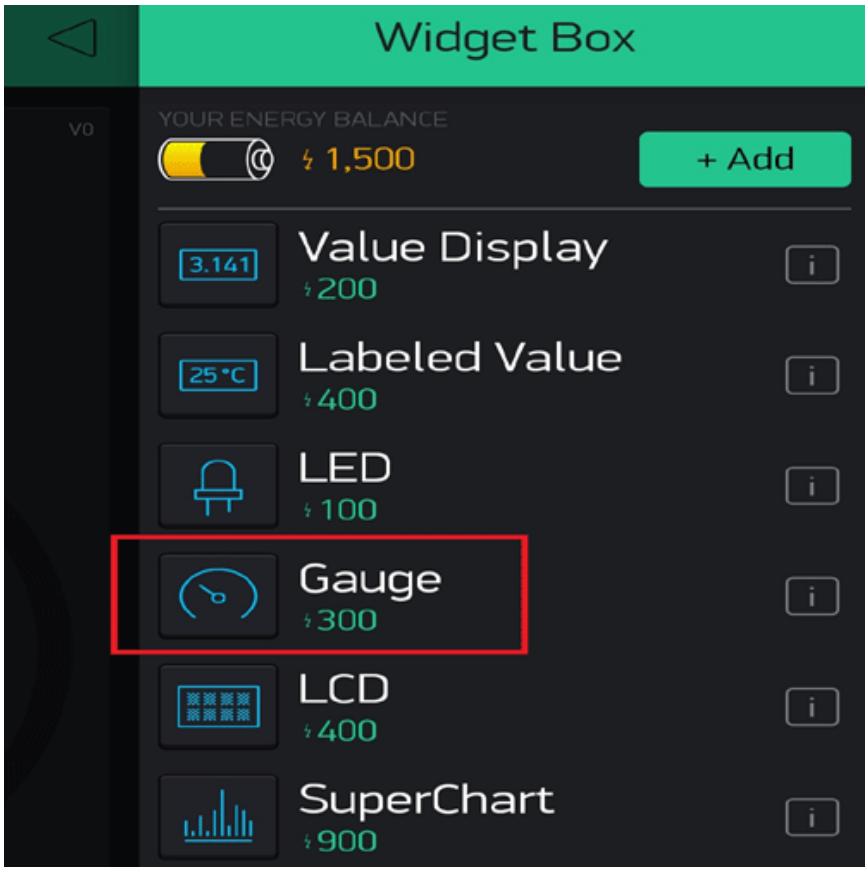


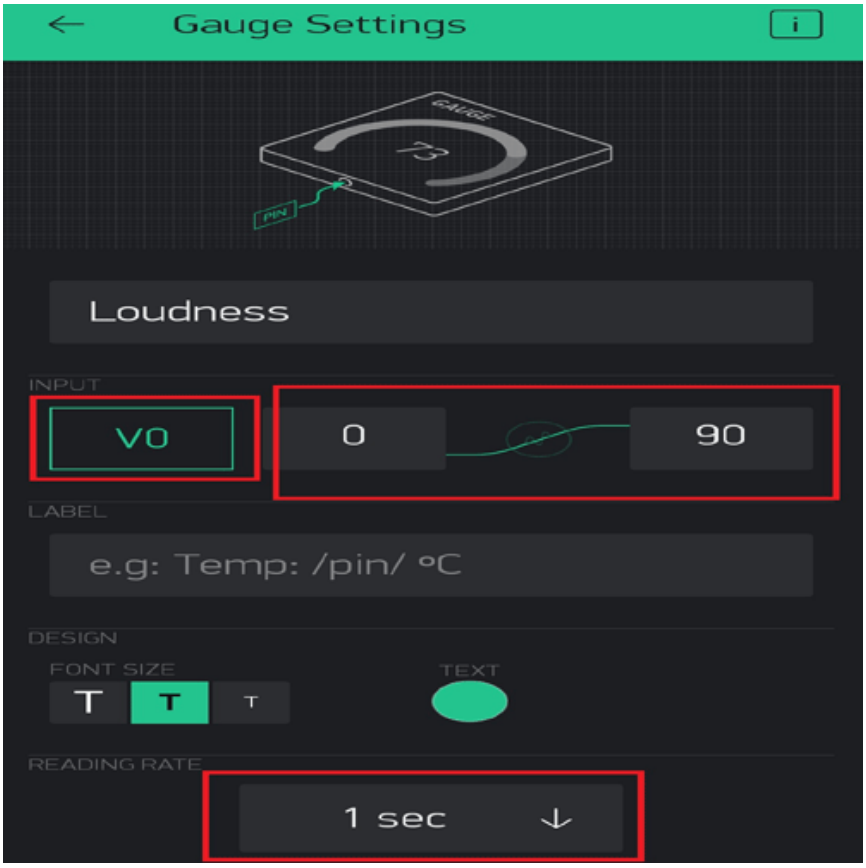



Blynk
New Project
My Apps
Community


Create New Project
Decibal Meter
CHOOSE DEVICE
NodeMCU
CONNECTION TYPE
Wi-Fi
THEME
DARK
LIGHT


Auth Token was sent to:
[redacted]@gmail.com
You can also find it in Project Settings
OK
Don't show again

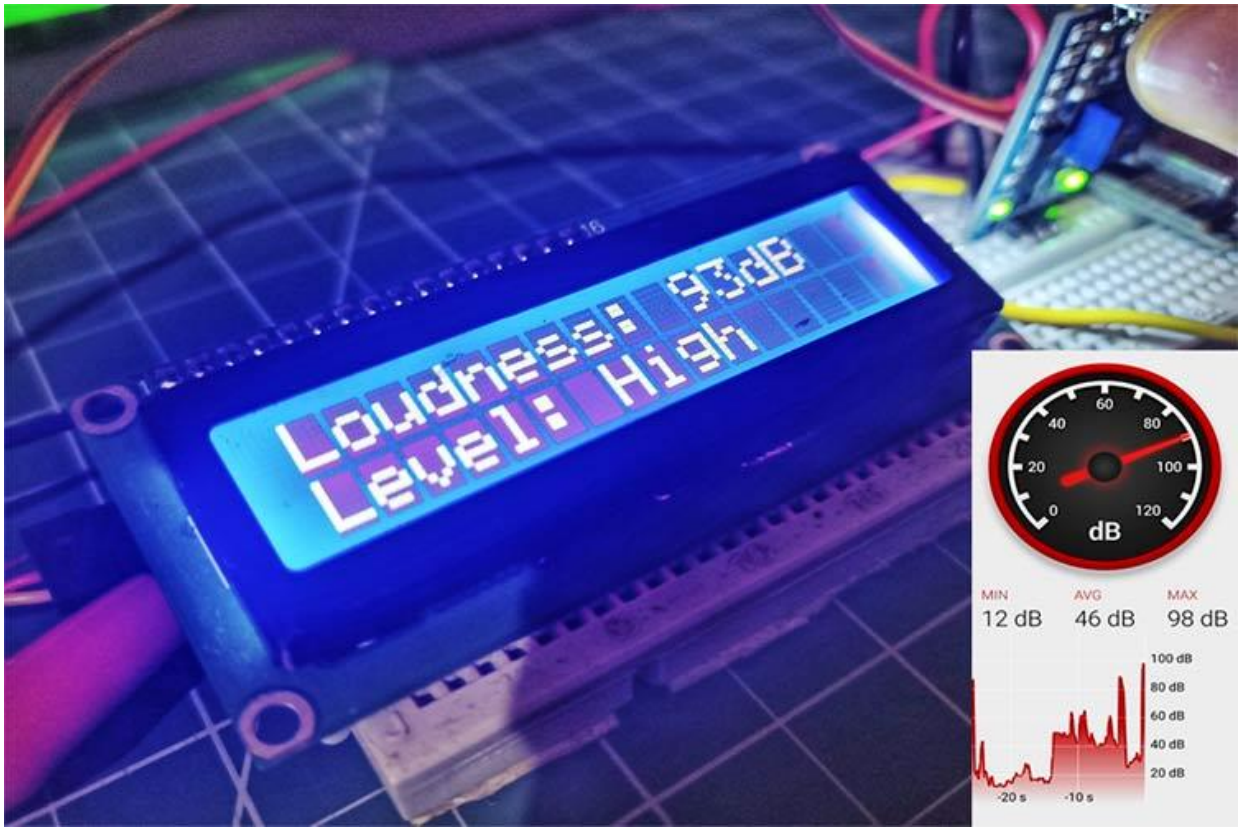

Widget Box
V0
YOUR ENERGY BALANCE
1,500
+ Add
3.141
Value Display
200
25 °C
Labeled Value
400
LED
100
Gauge
300
LCD
400
SuperChart
900



CODE:

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <LiquidCrystal_I2C.h>
#define SENSOR_PIN A0
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
const int sampleWindow = 50;
unsigned int sample;
int db;
char auth[] = "IEu1xT825VDt6hNfrcFgdJ6InJ1QUfsA";
char ssid[] = "realme 6";
char pass[] = "evil@zeb";
BLYNK_READ(V0)
{
 Blynk.virtualWrite(V0, db);
}
void setup() {
 pinMode (SENSOR_PIN, INPUT);
 lcd.begin(16, 2);
 lcd.backlight();
 lcd.clear();
 Blynk.begin(auth, ssid, pass);
}
```

```
void loop() {
 Blynk.run();
 unsigned long startMillis = millis();  // Start of sample window
 float peakToPeak = 0;  // peak-to-peak level
 unsigned int signalMax = 0;  //minimum value
 unsigned int signalMin = 1024;  //maximum value
 // collect data for 50 mS
 while (millis() - startMillis < sampleWindow)
 {
   sample = analogRead(SENSOR_PIN);  //get reading from microphone
   if (sample < 1024)  // toss out spurious readings
   {
     if (sample > signalMax)
     {
       signalMax = sample;  // save just the max levels
     }
     else if (sample < signalMin)
     {
       signalMin = sample;  // save just the min levels
     }
   }
 }
 peakToPeak = signalMax - signalMin;  // max - min = peak-peak amplitude
 Serial.println(peakToPeak);
 db = map(peakToPeak, 20, 900, 49.5, 90);  //calibrate for deciBels
 lcd.setCursor(0, 0);
 lcd.print("Loudness: ");
 lcd.print(db);
 lcd.print("dB");
 if (db <= 50)
 {
   lcd.setCursor(0, 1);
   lcd.print("Level: Quite");
 }
 else if (db > 50 && db < 75)
 {
   lcd.setCursor(0, 1);
   lcd.print("Level: Moderate");
 }
 else if (db >= 75)
 {
   lcd.setCursor(0, 1);
   lcd.print("Level: High");
 }
 delay(600);
 lcd.clear();
}
```


Loudness: 93dB
Level: High
0
20
40
60
80
100
120
dB
MIN
12 dB
AVG
46 dB
MAX
98 dB
100 dB
80 dB
60 dB
40 dB
20 dB
-20 s
-10 s