

NOISE POLLUTION MONITORING

PHASE - 4

About Thing Speak:

Thingspeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyse live data streams in the cloud. The platform's real-time data management tools facilitate swift decision-making based on live information streams. Its popularity in the IoT landscape stems from the simplicity it offers in channel creation and data manipulation. Whether tracking environmental variables, controlling smart devices, or conducting research, Thingspeak provides a robust framework. The collaborative nature of Thingspeak encourages a community-driven approach, fostering the sharing of ideas and code snippets. Overall, Thingspeak stands as a dynamic hub for IoT enthusiasts, amplifying the potential for innovation in the interconnected world of devices and data.

Code to display Real time Transit Information on Thingspeak:

Noise_pollution.ino

```
1 #include <ESP8266WiFi.h>
2 #include <SPI.h>
3 #include <Wire.h>
4 #include <Adafruit_GFX.h>
5 #include <Adafruit_SSD1306.h>
6
7 #define SCREEN_WIDTH 128 // OLED display width, in pixels
8 #define SCREEN_HEIGHT 64 // OLED display height, in pixels
9 #define OLED_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin)
10 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
11
12 String apiKey = "N2WXZ15RHOJED7VC"; // Enter your Write API key from ThingSpeak
13 const char *ssid = "Alexahome"; // replace with your wifi ssid and wpa2 key
14 const char *pass = "12345678";
15 const char* server = "api.thingspeak.com";
16
17 const int sampleWindow = 50; // Sample window width in ms (50 ms = 20Hz)
18 unsigned int sample;
19
20 WiFiClient client;
21
22 void setup()
23 {
24     Serial.begin(115200); //Serial comms for debugging
25     display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //OLED display start
26     display.display(); //show buffer
27     display.clearDisplay(); //clear buffer
28     display.setTextSize(1); //Set text size to 1 (1-6)
29     display.setTextColor(WHITE); //Set text color to WHITE (no choice lol)
30     display.setCursor(0,0); //cursor to upper left corner
31     display.println("Decibelmeter"); //write title
32     display.display(); //show title
33     delay(2000); //wait 2 seconds
34
35     WiFi.begin(ssid, pass);
--
```

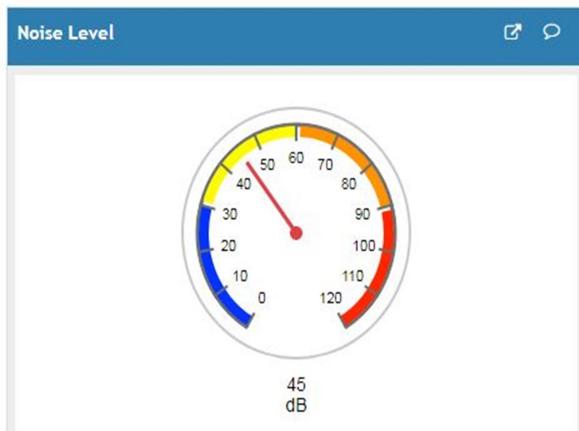
```
36
37     while (WiFi.status() != WL_CONNECTED)
38     {
39         delay(500);
40         Serial.print(".");
41     }
42     Serial.println("");
43     Serial.println("WiFi connected");
44
45     display.clearDisplay();
46     display.setCursor(0,0);
47     display.setTextSize(1);
48     display.setTextColor(WHITE);
49     display.print("WiFi connected");
50     display.display();
51     delay(4000);
52     display.clearDisplay();
53 }
54
55 void loop()
56 {
57     unsigned long startMillis= millis(); // Start of sample window
58     float peakToPeak = 0; // peak-to-peak level
59
60     unsigned int signalMax = 0; //minimum value
61     unsigned int signalMin = 1024; //maximum value
62
63     ||||| // collect data for 50 mS
64     while (millis() - startMillis < sampleWindow)
65     {
66         sample = analogRead(0); //get reading from microphone
67         if (sample < 1024) // toss out spurious readings
68         {
69             || if (sample > signalMax)
```

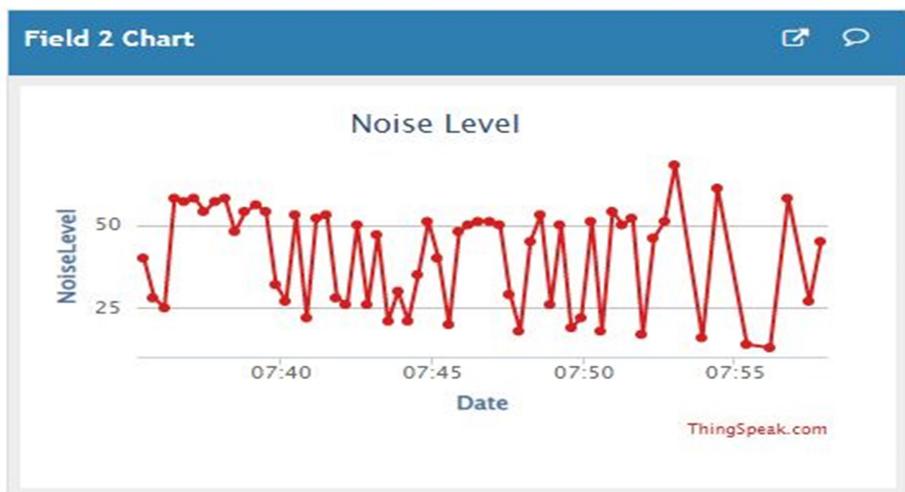
```

70     | {
71     | | signalMax = sample;           // save just the max levels
72     | |
73     | | else if (sample < signalMin)
74     | | {
75     | | | signalMin = sample;        // save just the min levels
76     | | }
77   }
78 }
79 peakToPeak = signalMax - signalMin;
80 float db = map(peakToPeak,20,900,49.5,90);
81 display.setCursor(0,0);
82 display.setTextSize(2);
83 display.print(db);
84 display.print(" dB");
85
86
87 for(int x =5;x<114;x=x+6){           //draw scale
88   display.drawLine(x, 32, x, 27, WHITE);
89 }
90 display.drawRoundRect(0, 32, 120, 20, 6, WHITE);    //draw outline of bar graph
91 int r = map(db,0,120,1,120);                     //set bar graph for width of screen
92 display.fillRoundRect(1, 33, r, 18, 6, WHITE);      //draw bar graph with a width of r
93 display.display();                                //show all that we just wrote & drew
94 display.clearDisplay();                           //clear the display
95
96 if (client.connect(server, 80)) // "184.106.153.149" or api.thingspeak.com
97 {
98   String postStr = apiKey;
99   postStr += "&field1=";
100  postStr += String(db);
101  postStr += "r\n";
102
103  client.print("POST /update HTTP/1.1\n");
104  client.print("Host: api.thingspeak.com\n");
105
106  client.print("Connection: close\n");
107  client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
108  client.print("Content-Type: application/x-www-form-urlencoded\n");
109  client.print("Content-Length: ");
110  client.print(postStr.length());
111  client.print("\n\n");
112  client.print(postStr);
113 }
114 client.stop();
115 delay(150);
116 }

```

OUTPUT:





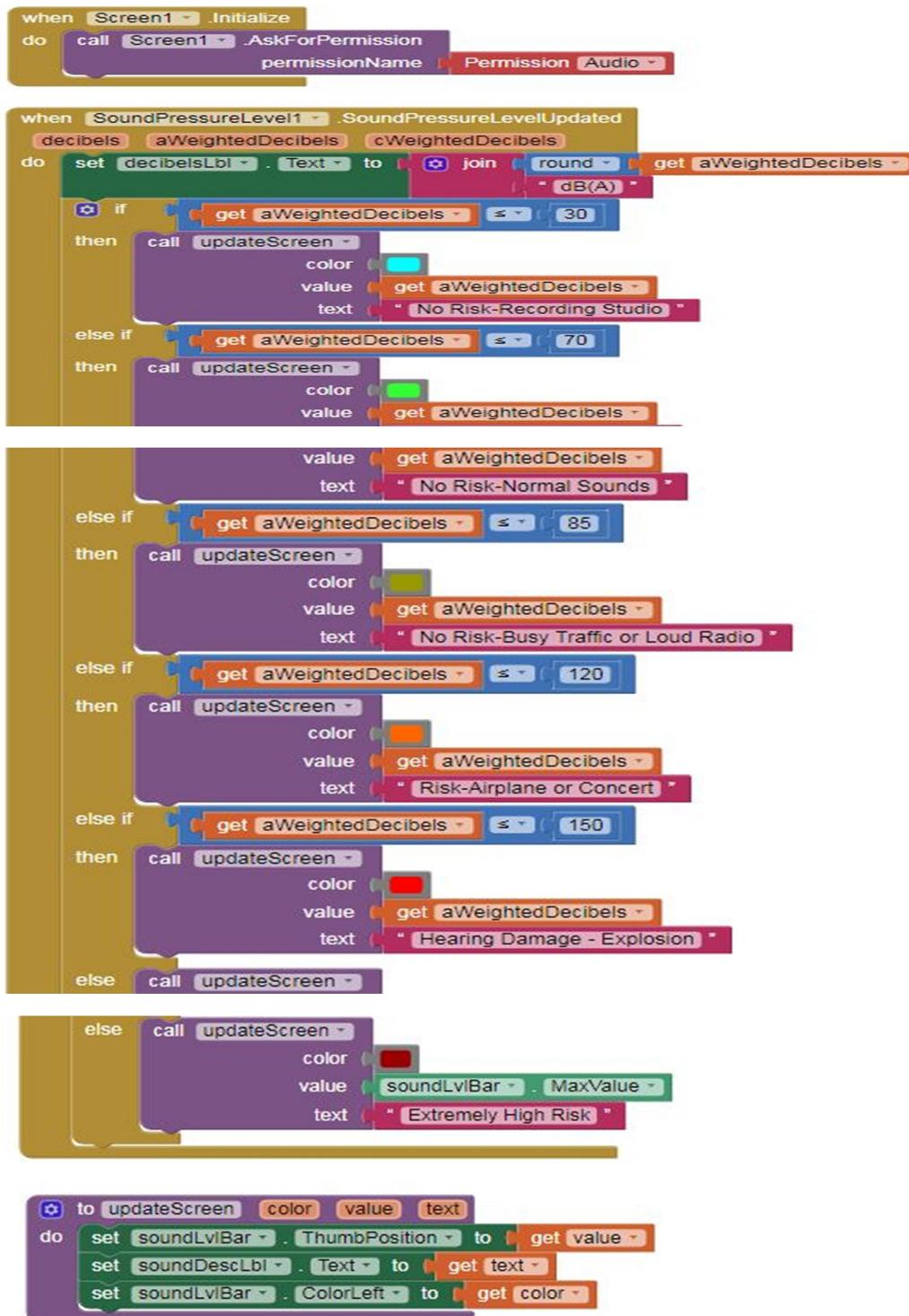
MOBILE APPLICATION:

Additionally, MIT App Inventor provides a simple and intuitive interface for testing and debugging apps in real-time. It emphasizes accessibility by enabling users to create functional apps with minimal effort. The platform supports a variety of features, including sensors, media playback, and connectivity options, allowing users to explore diverse app functionalities. MIT App Inventor serves as a valuable resource for individuals looking to kickstart their journey into mobile app development, fostering creativity and hands-on learning.

For collecting and storing real-time transit data, ThingSpeak channels were employed, capturing information like route details, location, and passenger count for each data point. In the MIT App Inventor, a visual interface facilitated app development, utilizing components like labels, text boxes, and web

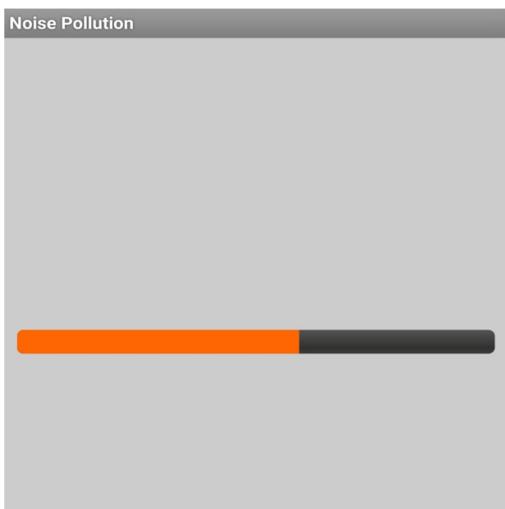
elements to shape the user interface. To retrieve data, the app was configured to interact with ThingSpeak channels through the Web component, making API calls for the latest entries. The app dynamically displayed the retrieved data, presenting route information, location, and passenger count in real-time on its interface.

MIT APP INVENTOR:

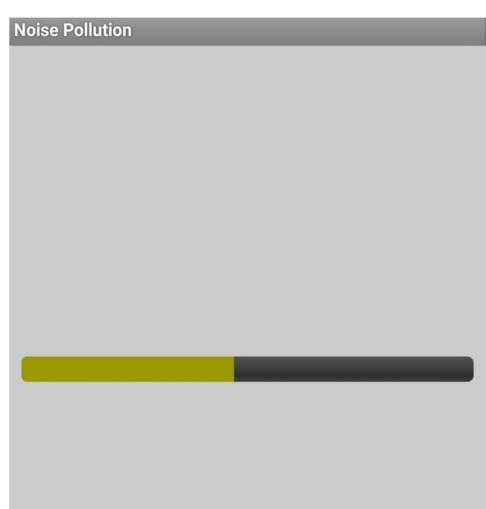


OUTPUT :

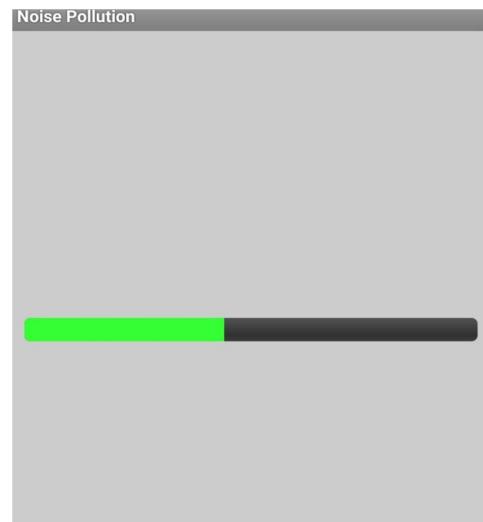
When noise level <= 120 dB



When noise level <= 85dB



When noise level <= 70dB



TEAM MEMBERS:

- | | | |
|----|---------------|----------------|
| 1. | Jayanth N | - (2021504523) |
| 2. | Lavanya V | - (2021504525) |
| 3. | Mathana K | - (2021504528) |
| 4. | Sudhanthira P | - (2021504548) |
| 5. | Vidhya SS | - (2021504557) |