

VOICE RECOGNITION



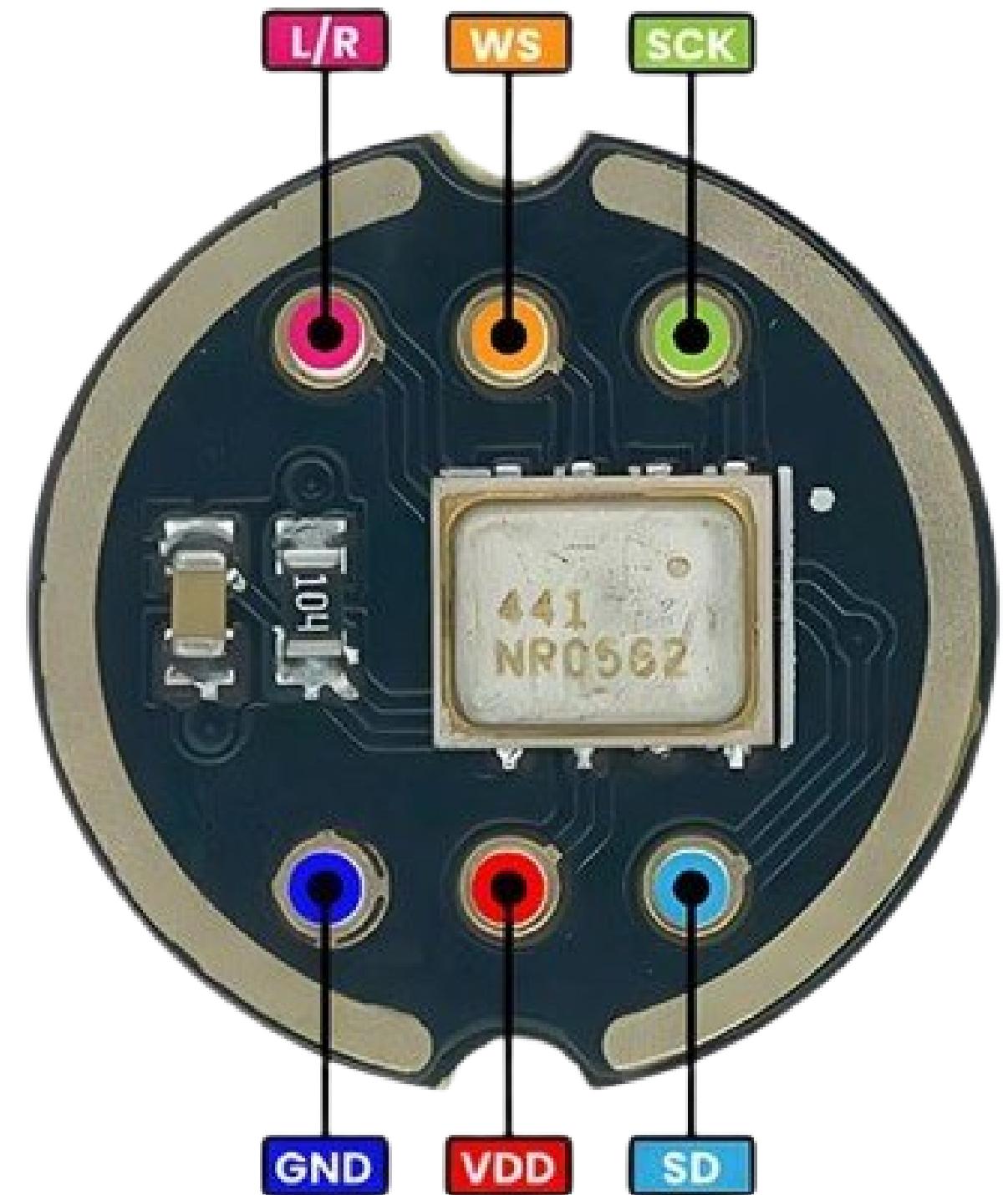
REAL WORLD APPLICATIONS

- **Virtual Assistants:** Voice recognition powers assistants like Alexa, Siri, and Google Assistant, enabling hands-free control of smart devices and access to information.
- **Security:** Voice biometrics are used for authentication in banking and secure systems, allowing user verification through unique vocal traits.
- **Customer Service:** Call centers use voice recognition to route calls and automate responses, enhancing customer support.
- **Translation:** Real-time voice translation helps overcome language barriers in travel, business, and education.
- **Accessibility:** Voice recognition aids individuals with disabilities by enabling hands-free device operation and real-time captioning.

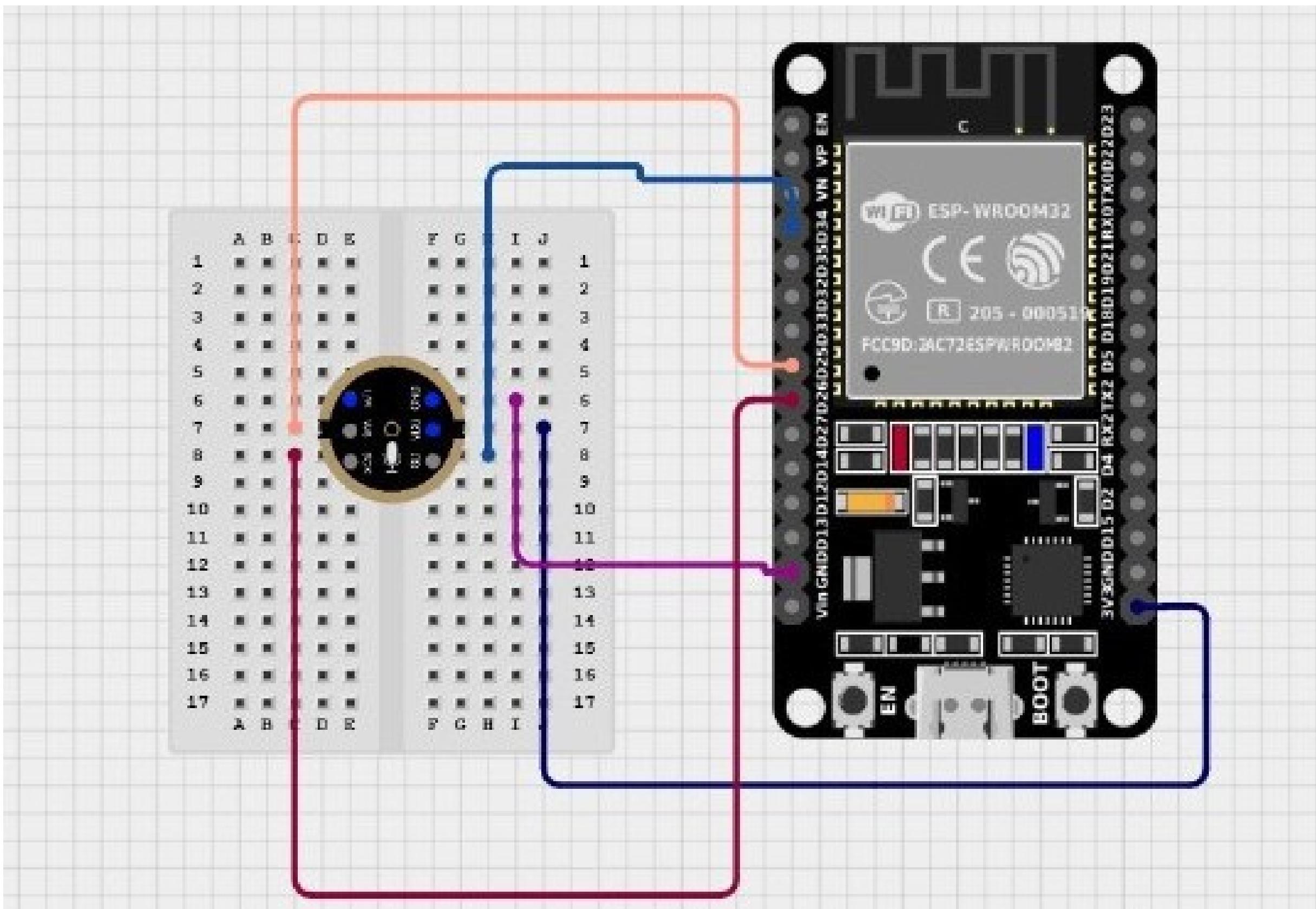
WORKING

The voice recognition project uses the INMP441 digital MEMS microphone to capture sound, converting it into digital data via I2S output. Edge Impulse processes this data with machine learning models to identify voice patterns or commands in real time. This setup enables quick, accurate voice recognition on edge devices with low power use, making it ideal for smart home controls, voice assistants, and other hands-free applications.

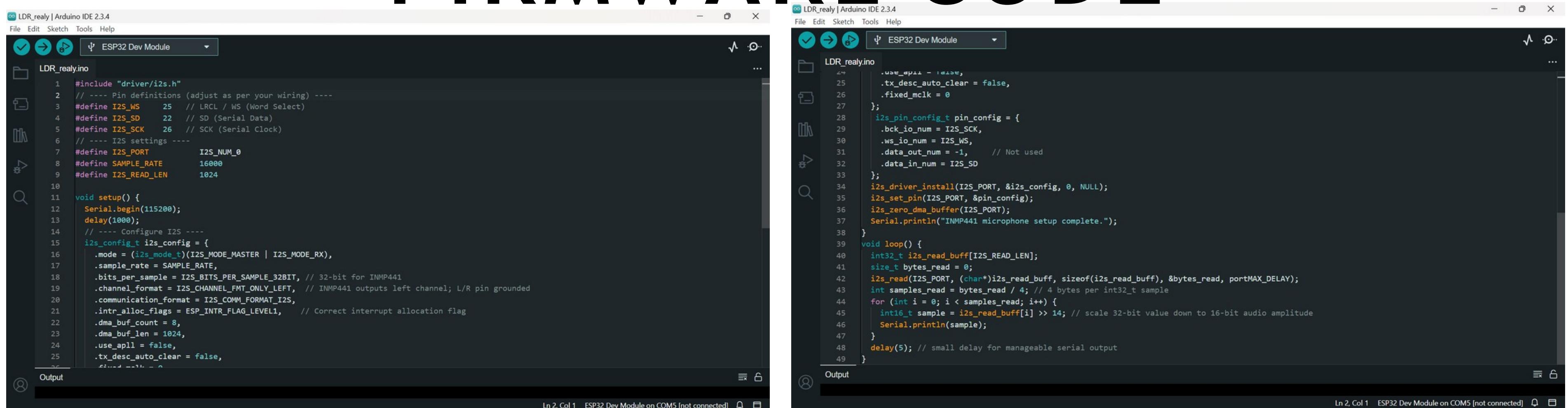
PINOUT



EK NAZAR IDHAR BHI



FIRMWARE CODE



The image shows two instances of the Arduino IDE running side-by-side. Both windows have the title 'LDR_realy | Arduino IDE 2.3.4' and show the same code file 'LDR_realy.ino'. The code is written in C++ and defines an I2S driver for an ESP32 Dev Module. It includes pin definitions for I2S_WS (25), I2S_SD (22), and I2S_SCK (26). It sets up the I2S port at 16000 samples per second and reads data from the INMP441 microphone using I2S_NUM_0. The code then prints the received audio samples to the serial port.

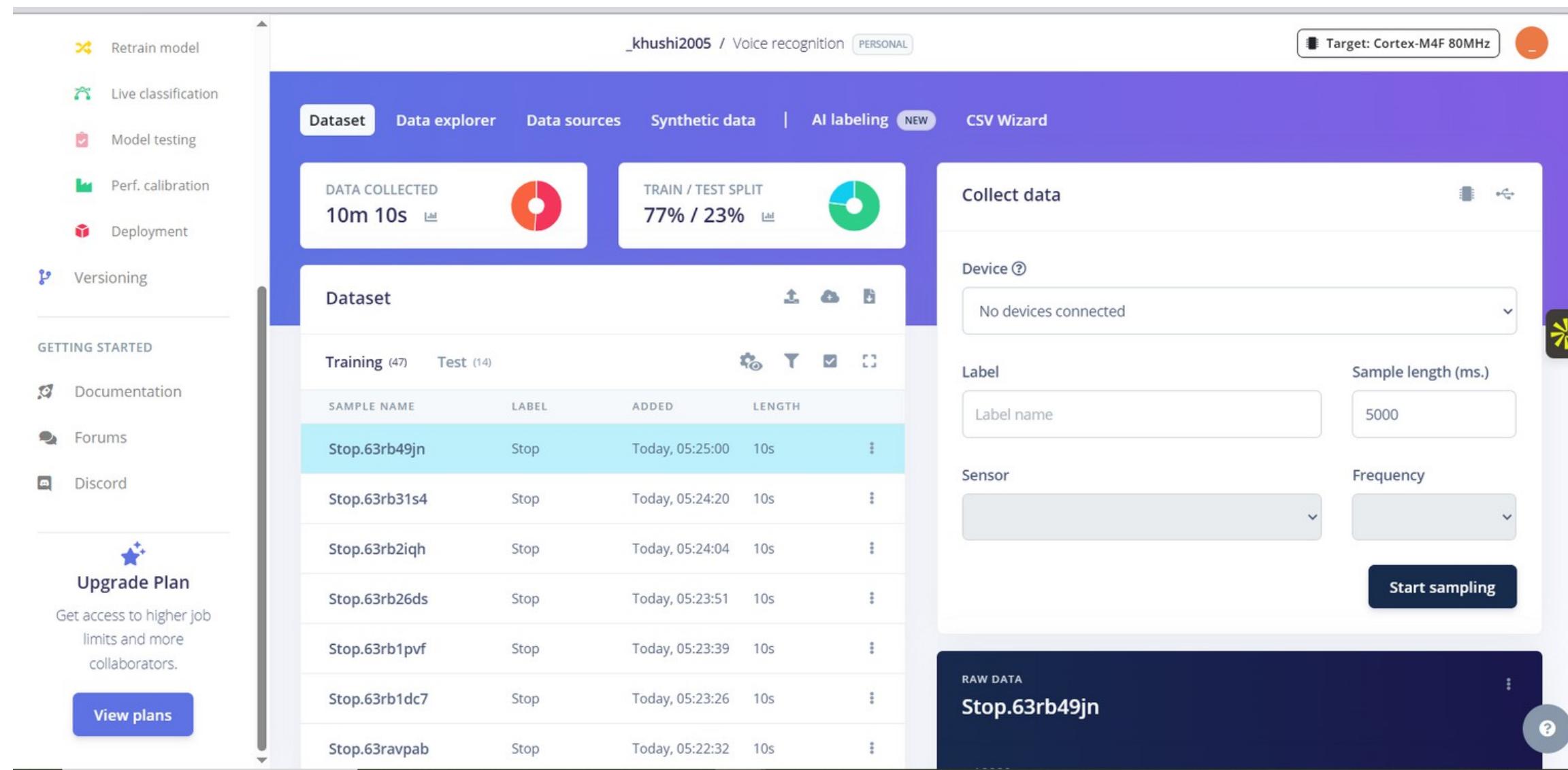
```
#include "driver/i2s.h"
// ---- Pin definitions (adjust as per your wiring) ----
#define I2S_WS 25 // LRCL / WS (Word Select)
#define I2S_SD 22 // SD (Serial Data)
#define I2S_SCK 26 // SCK (Serial Clock)
// ---- I2S settings ----
#define I2S_PORT I2S_NUM_0
#define SAMPLE_RATE 16000
#define I2S_READ_LEN 1024

void setup() {
    Serial.begin(115200);
    delay(1000);
    // ---- Configure I2S ----
    i2s_config_t i2s_config = {
        .mode = (i2s_mode_t)(I2S_MODE_MASTER | I2S_MODE_RX),
        .sample_rate = SAMPLE_RATE,
        .bits_per_sample = I2S_BITS_PER_SAMPLE_32BIT, // 32-bit for INMP441
        .channel_format = I2S_CHANNEL_FMT_ONLY_LEFT, // INMP441 outputs left channel; L/R pin grounded
        .communication_format = I2S_COMM_FORMAT_I2S,
        .intr_alloc_flags = ESP_INTR_FLAG_LEVEL1, // Correct interrupt allocation flag
        .dma_buf_count = 8,
        .dma_buf_len = 1024,
        .use_apll = false,
        .tx_desc_auto_clear = false,
    };
    i2s_driver_install(I2S_PORT, &i2s_config, 0, NULL);
    i2s_set_pin(I2S_PORT, &pin_config);
    i2s_zero_dma_buffer(I2S_PORT);
    Serial.println("INMP441 microphone setup complete.");
}

void loop() {
    int32_t i2s_read_buff[I2S_READ_LEN];
    size_t bytes_read = 0;
    i2s_read(I2S_PORT, (char*)i2s_read_buff, sizeof(i2s_read_buff), &bytes_read, portMAX_DELAY);
    int samples_read = bytes_read / 4; // 4 bytes per int32_t sample
    for (int i = 0; i < samples_read; i++) {
        int16_t sample = i2s_read_buff[i] >> 14; // scale 32-bit value down to 16-bit audio amplitude
        Serial.println(sample);
    }
    delay(5); // small delay for manageable serial output
}
```

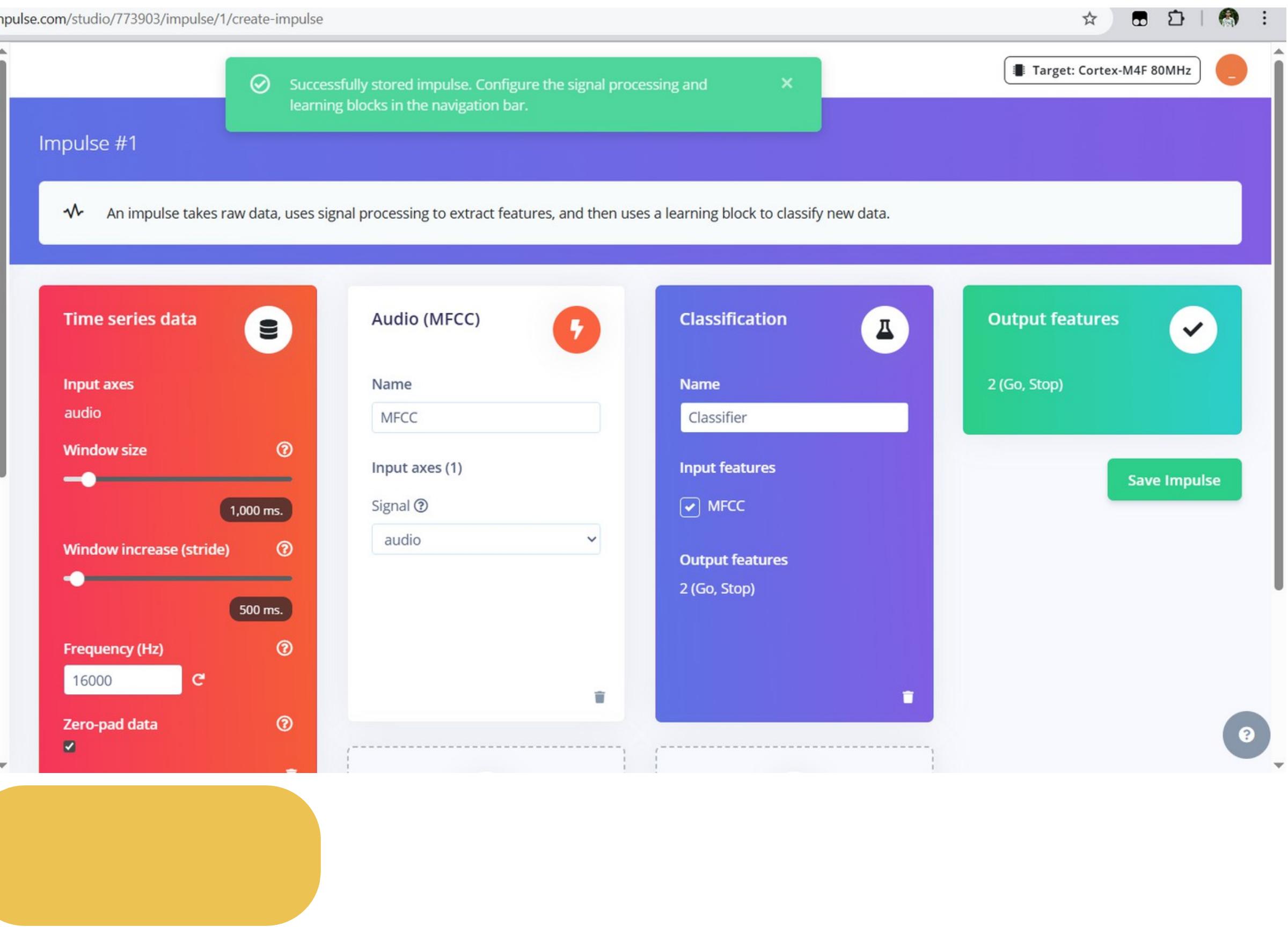
This code is used a firmware to your esp which enable it to send your data through A bridge called Edge impulse CLI. This is already uploaded in your esp..

DATA ACQUITION



We will use premade dataset for
‘Data Acquition’.

DESIGNING AN IMPULSE



Go to the Create impulse tab, add a Time series data, an Audio (MFCC) and a Classification (Keras) block. Leave the window size to 1second (as that's the length of our audio samples in the dataset) and click Save Impulse.

CONFIGURE THE MFCC BLOCK

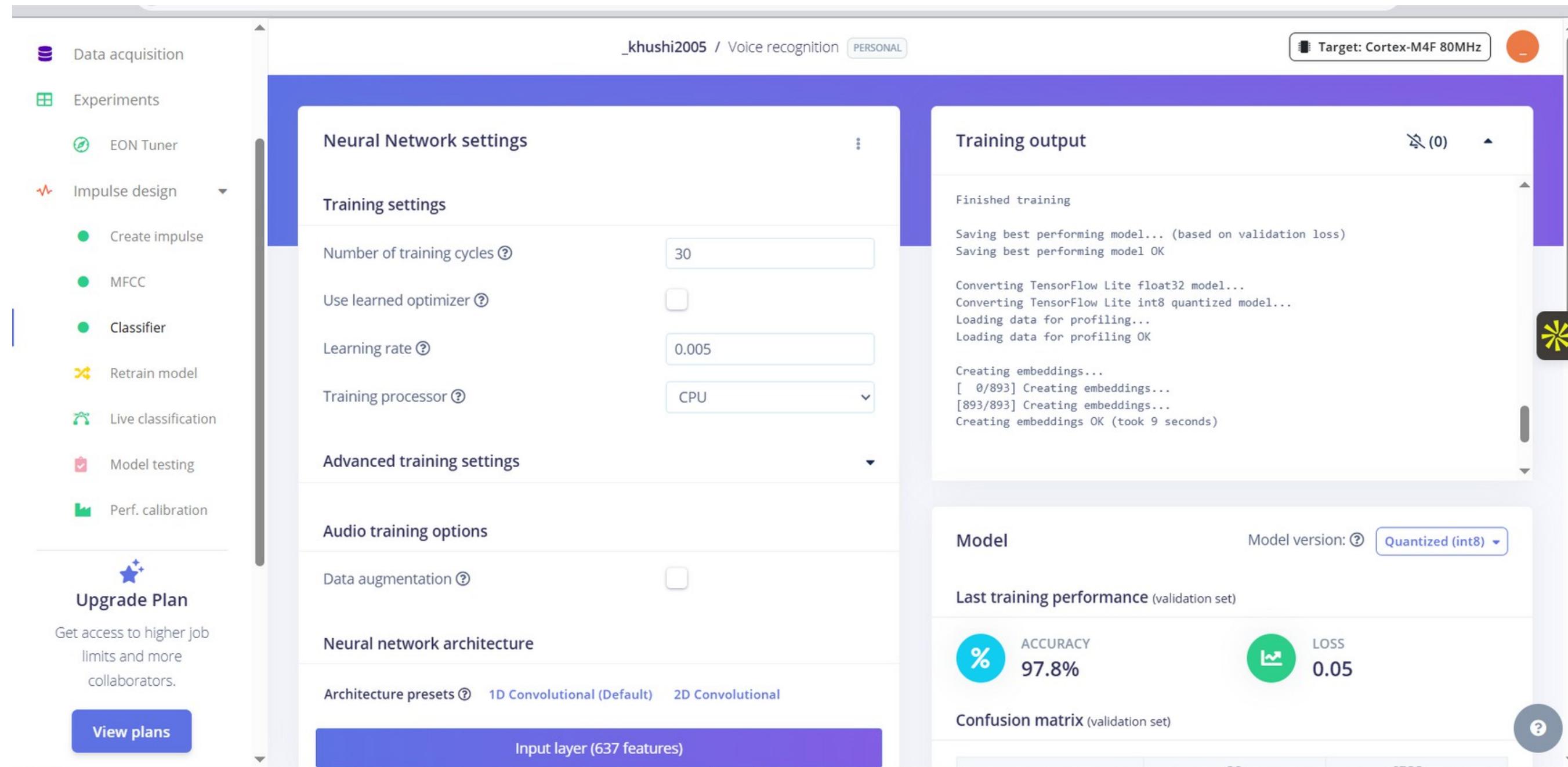
Now click on the MFCC tab in the left hand navigation menu. You just click on the autotune parameters and then save parameters.

The screenshot shows the Edge Impulse web interface. On the left, a navigation menu includes 'Dashboard', 'Devices', 'Data acquisition', 'Experiments', 'EON Tuner' (which is selected), 'Impulse design', 'Create impulse', 'MFCC' (selected), and 'Classifier'. A 'Upgrade Plan' section is also present. The main area is titled 'DGE IMPULSE' and contains a 'Parameters' section for 'Mel Frequency Cepstral Coefficients' with fields for Number of coefficients (13), Frame length (0.025), Frame stride (0.02), Filter number (32), FFT length (512), Normalization window size (151), Low frequency (80), High frequency (Click to set), and Pre-emphasis (Coefficient 0.98). A 'Save parameters' button is at the bottom right. To the right, a 'DSP result' section shows a heatmap titled 'Cepstral Coefficients' with a color scale from blue to red. Below it is a 'Processed features' list: 1.0122, 0.6761, 0.2268, -0.0601, 0.7023, 0.0163, 0.5544, 0.4388, 0.1600, 0.2859, ...

You will be redirected to feature generation window.
Click on generate features and then you can analyse the graph.

This screenshot shows the 'Feature generation output' section with a progress bar at [829/893] Creating features... and a 'Generate features' button. Below it is an 'On-device performance' section showing a processing time of 154 ms and peak RAM usage of 15 KB. The main area features a 'Feature explorer' scatter plot with orange dots representing 'Stop' and blue dots representing 'Go' audio samples. The left sidebar has sections for 'Training set' (Data in training set: 7m 50s, Classes: 2 (Go, Stop), Training windows: 893) and 'Feature generation output' (status: Job completed (success)). The bottom left shows a 'View plans' button. The bottom right shows a 'View plans' button.

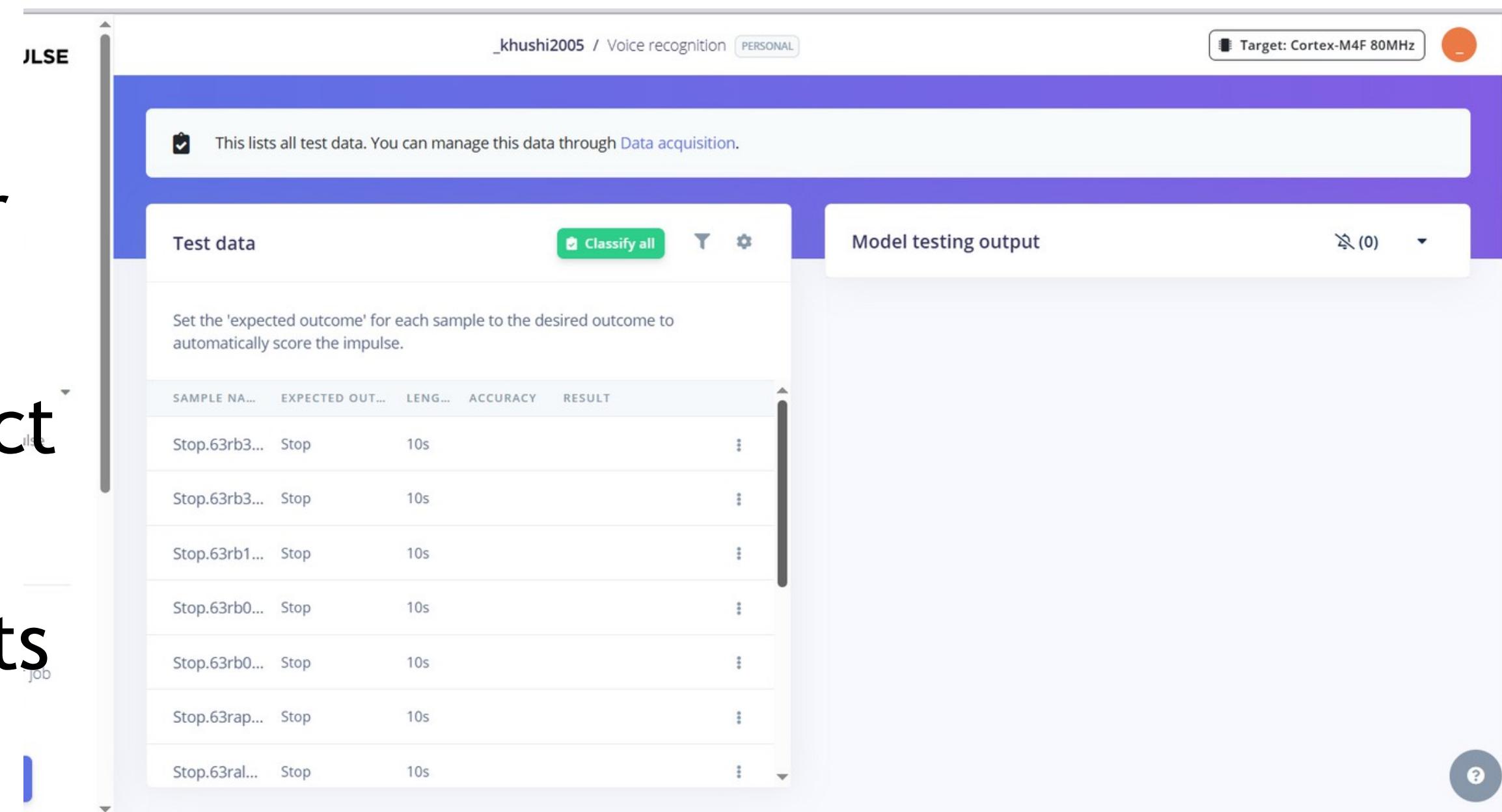
CONFIGURE THE NEURAL NETWORK



Click on **Classifier** in the left hand menu.
Change the number of training cycles to 50 and
train the model.

CLASSIFYING TEST DATA

Add Test Data in Data Acquisition window if not present already. To run your model against the test set, head to **Model testing**, select all items and click **Classify selected**. Click the three dots (:) next to a sample and select Show classification.



DEVELOPMENT

Add the given deployment code in your Arduino IDE and and then start running your model. As you will say something the model will predict it and will display in the serial monitor.

DEPLOYMENT CODE

LDR_realy | Arduino IDE 2.3.4

File Edit Sketch Tools Help

ESP32 Dev Module

```
LDR_realy.ino
1 #include <driver/i2s.h>
2 #include <Arduino.h>
3
4 // I2S Microphone config
5 #define I2S_SAMPLE_RATE 16000
6 #define I2S_SAMPLE_BITS 32
7 #define I2S_READ_LEN 2048
8 #define I2S_MIC_CHANNEL I2S_CHANNEL_FMT_ONLY_LEFT
9 #define I2S_FORMAT I2S_COMM_FORMAT_I2S
10 #define I2S_PORT I2S_NUM_0
11
12 #define PIN_I2S_WS 25
13 #define PIN_I2S_SD 32
14 #define PIN_I2S_SCK 26
15
16 // Buffer for audio samples
17 static int32_t i2s_read_buffer[I2S_READ_LEN];
18
19 // Setup I2S
20 void i2s_init() {
21     i2s_config_t i2s_config = {
22         .mode = (i2s_mode_t)(I2S_MODE_MASTER | I2S_MODE_RX),
23         .sample_rate = I2S_SAMPLE_RATE,
24         .bits_per_sample = I2S_SAMPLE_BITS,
25         .channel_format = I2S_MIC_CHANNEL,
26         .communication_format = I2S_FORMAT
27     };
28
29     i2s_driver_install(I2S_PORT, &i2s_config, 0, NULL);
30     i2s_set_pin(I2S_PORT, &pin_config);
31     i2s_zero_dma_buffer(I2S_PORT);
32
33     i2s_pin_config_t pin_config = {
34         .bck_io_num = PIN_I2S_SCK,
35         .ws_io_num = PIN_I2S_WS,
36         .data_out_num = I2S_PIN_NO_CHANGE,
37         .data_in_num = PIN_I2S_SD
38     };
39
40     i2s_driver_install(I2S_PORT, &i2s_config, 0, NULL);
41     i2s_set_pin(I2S_PORT, &pin_config);
42     i2s_zero_dma_buffer(I2S_PORT);
43
44     // Read samples from I2S and convert to int16_t
45     int i2s_read(int16_t *samples, size_t len) {
46         size_t bytes_read = 0;
47         i2s_read(I2S_PORT, (void*)i2s_read_buffer, len * sizeof(int32_t), &bytes_read, portMAX_DELAY);
48
49         int samples_read = bytes_read / sizeof(int32_t);
50
51         for (int i = 0; i < samples_read; i++) {
52             samples[i] = (int16_t)(i2s_read_buffer[i] >> 14); // shift 32-bit to 16-bit
53         }
54
55         return samples_read;
56     }
57
58     void setup() {
59         Serial.begin(115200);
60         Serial.println("Edge Impulse ESP32 Voice Recognition");
61
62         i2s_init();
63     }
64
65     void loop() {
66         // Buffer for model input
67         static int16_t audio_buffer[EI_CLASSIFIER_SLICE_SIZE];
68
69         // Fill buffer with microphone samples
70         int samples_collected = i2s_read(audio_buffer, EI_CLASSIFIER_SLICE_SIZE);
71
72         if (samples_collected == EI_CLASSIFIER_SLICE_SIZE) {
73             // Wrap into Edge Impulse signal
74             signal_t signal;
75             numpy::signal_from_buffer(audio_buffer, EI_CLASSIFIER_SLICE_SIZE, &signal);
76
77             // Run classifier
78             ei_impulse_result_t result;
79             EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false);
80
81             if (res != EI_IMPULSE_OK) {
82                 Serial.printf("ERROR: Classifier failed (%d)\n", res);
83                 return;
84             }
85
86             // Print predictions
87             Serial.println("Predictions:");
88             for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
89                 Serial.printf(" %s: %.5f\n", result.classification[ix].label, result.classification[ix].value);
90             }
91             Serial.println();
92         }
93     }
94 }
```

LDR_realy | Arduino IDE 2.3.4

File Edit Sketch Tools Help

ESP32 Dev Module

```
LDR_realy.ino
50     int samples_read = bytes_read / sizeof(int32_t);
51     for (int i = 0; i < samples_read; i++) {
52         samples[i] = (int16_t)(i2s_read_buffer[i] >> 14); // shift 32-bit to 16-bit
53     }
54
55     return samples_read;
56 }
57
58 void setup() {
59     Serial.begin(115200);
60     Serial.println("Edge Impulse ESP32 Voice Recognition");
61
62     i2s_init();
63 }
64
65 void loop() {
66     // Buffer for model input
67     static int16_t audio_buffer[EI_CLASSIFIER_SLICE_SIZE];
68
69     // Fill buffer with microphone samples
70     int samples_collected = i2s_read(audio_buffer, EI_CLASSIFIER_SLICE_SIZE);
71
72     if (samples_collected == EI_CLASSIFIER_SLICE_SIZE) {
73         // Wrap into Edge Impulse signal
74         signal_t signal;
75         numpy::signal_from_buffer(audio_buffer, EI_CLASSIFIER_SLICE_SIZE, &signal);
76
77         // Run classifier
78         ei_impulse_result_t result;
79         EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false);
80
81         if (res != EI_IMPULSE_OK) {
82             Serial.printf("ERROR: Classifier failed (%d)\n", res);
83             return;
84         }
85
86         // Print predictions
87         Serial.println("Predictions:");
88         for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
89             Serial.printf(" %s: %.5f\n", result.classification[ix].label, result.classification[ix].value);
90         }
91         Serial.println();
92     }
93 }
```

LDR_realy | Arduino IDE 2.3.4

File Edit Sketch Tools Help

ESP32 Dev Module

```
LDR_realy.ino
25     .channel_format = I2S_MIC_CHANNEL,
26     .communication_format = I2S_FORMAT,
27     .intr_alloc_flags = ESP_INTR_FLAG_LEVEL1,
28     .dma_buf_count = 4,
29     .dma_buf_len = 512,
30     .use_apll = false
31 };
32
33 i2s_pin_config_t pin_config = {
34     .bck_io_num = PIN_I2S_SCK,
35     .ws_io_num = PIN_I2S_WS,
36     .data_out_num = I2S_PIN_NO_CHANGE,
37     .data_in_num = PIN_I2S_SD
38 };
39
40 i2s_driver_install(I2S_PORT, &i2s_config, 0, NULL);
41 i2s_set_pin(I2S_PORT, &pin_config);
42 i2s_zero_dma_buffer(I2S_PORT);
43
44 // Read samples from I2S and convert to int16_t
45 int i2s_read(int16_t *samples, size_t len) {
46     size_t bytes_read = 0;
47     i2s_read(I2S_PORT, (void*)i2s_read_buffer, len * sizeof(int32_t), &bytes_read, portMAX_DELAY);
48
49     int samples_read = bytes_read / sizeof(int32_t);
50
51     for (int i = 0; i < samples_read; i++) {
52         samples[i] = (int16_t)(i2s_read_buffer[i] >> 14); // shift 32-bit to 16-bit
53     }
54
55     return samples_read;
56 }
57
58 void setup() {
59     Serial.begin(115200);
60     Serial.println("Edge Impulse ESP32 Voice Recognition");
61
62     i2s_init();
63 }
64
65 void loop() {
66     // Buffer for model input
67     static int16_t audio_buffer[EI_CLASSIFIER_SLICE_SIZE];
68
69     // Fill buffer with microphone samples
70     int samples_collected = i2s_read(audio_buffer, EI_CLASSIFIER_SLICE_SIZE);
71
72     if (samples_collected == EI_CLASSIFIER_SLICE_SIZE) {
73         // Wrap into Edge Impulse signal
74         signal_t signal;
75         numpy::signal_from_buffer(audio_buffer, EI_CLASSIFIER_SLICE_SIZE, &signal);
76
77         // Run classifier
78         ei_impulse_result_t result;
79         EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false);
80
81         if (res != EI_IMPULSE_OK) {
82             Serial.printf("ERROR: Classifier failed (%d)\n", res);
83             return;
84         }
85
86         // Print predictions
87         Serial.println("Predictions:");
88         for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
89             Serial.printf(" %s: %.5f\n", result.classification[ix].label, result.classification[ix].value);
90         }
91         Serial.println();
92     }
93 }
```

LDR_realy | Arduino IDE 2.3.4

File Edit Sketch Tools Help

ESP32 Dev Module

```
LDR_realy.ino
68     // Fill buffer with microphone samples
69     int samples_collected = i2s_read(audio_buffer, EI_CLASSIFIER_SLICE_SIZE);
70
71     if (samples_collected == EI_CLASSIFIER_SLICE_SIZE) {
72         // Wrap into Edge Impulse signal
73         signal_t signal;
74         numpy::signal_from_buffer(audio_buffer, EI_CLASSIFIER_SLICE_SIZE, &signal);
75
76         // Run classifier
77         ei_impulse_result_t result;
78         EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false);
79
80         if (res != EI_IMPULSE_OK) {
81             Serial.printf("ERROR: Classifier failed (%d)\n", res);
82             return;
83         }
84
85         // Print predictions
86         Serial.println("Predictions:");
87         for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
88             Serial.printf(" %s: %.5f\n", result.classification[ix].label, result.classification[ix].value);
89         }
90         Serial.println();
91     }
92 }
```



THANK YOU
