carenuity

adafruit

Connecting DHT11 to Adafruit IO

By Cynthia Kipruto

What is Adafruit IO?

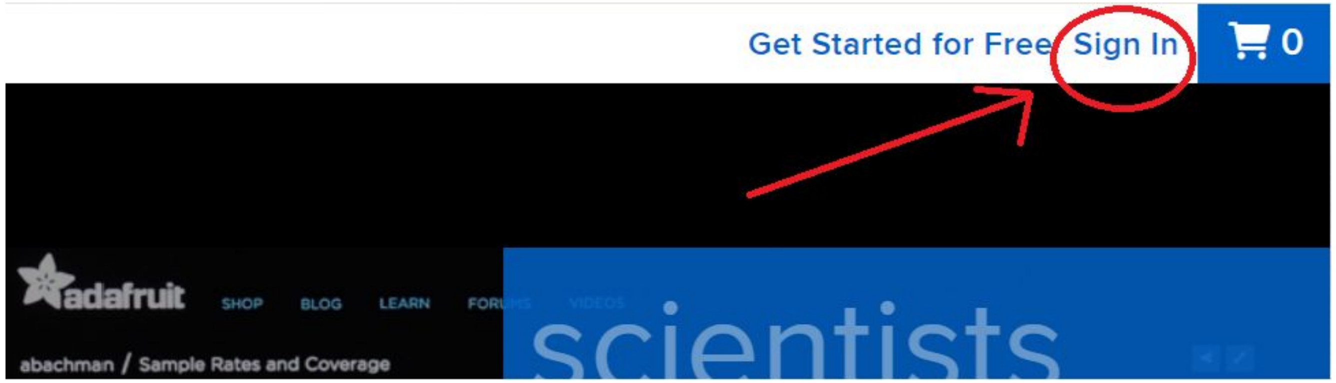
carenuity

- A cloud platform designed specifically for IoT Applications.

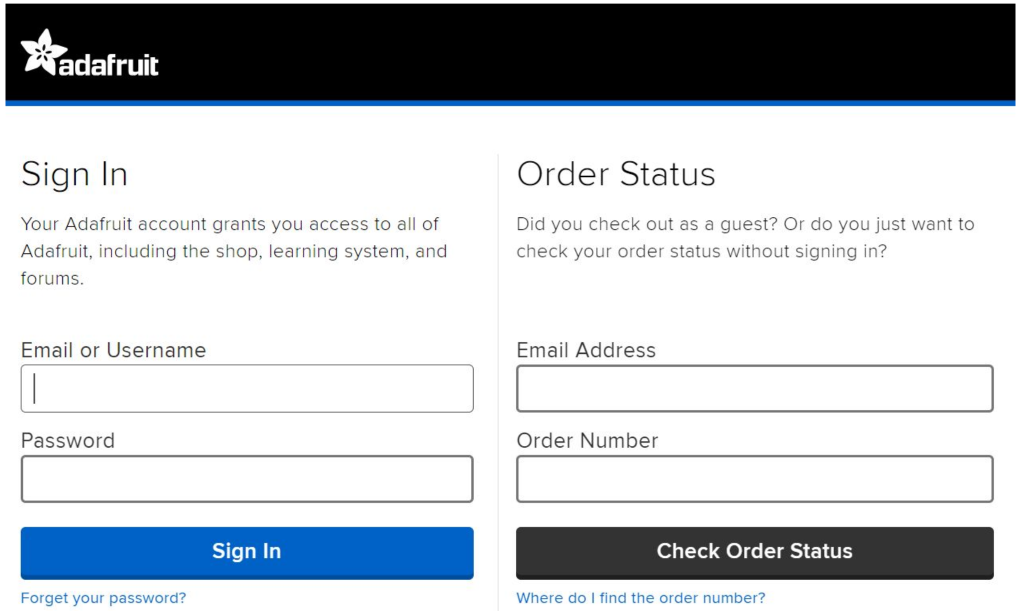
- Allows users to connect and manage their IoT devices, collect data and create visual dashboards to monitor and analyze this data in real time.

Creating an Account on Adafruit IO

- Go to the Adafruit IO Website: https://io.adafruit.com/
- Click on "Sign Up"



- Enter your details and create an account



Sign In

Your Adafruit account grants you access to all of Adafruit, including the shop, learning system, and forums.

Email or Username

Password

Sign In

Forget your password?

Order Status

Did you check out as a guest? Or do you just want to check your order status without signing in?

Email Address

Order Number

Check Order Status

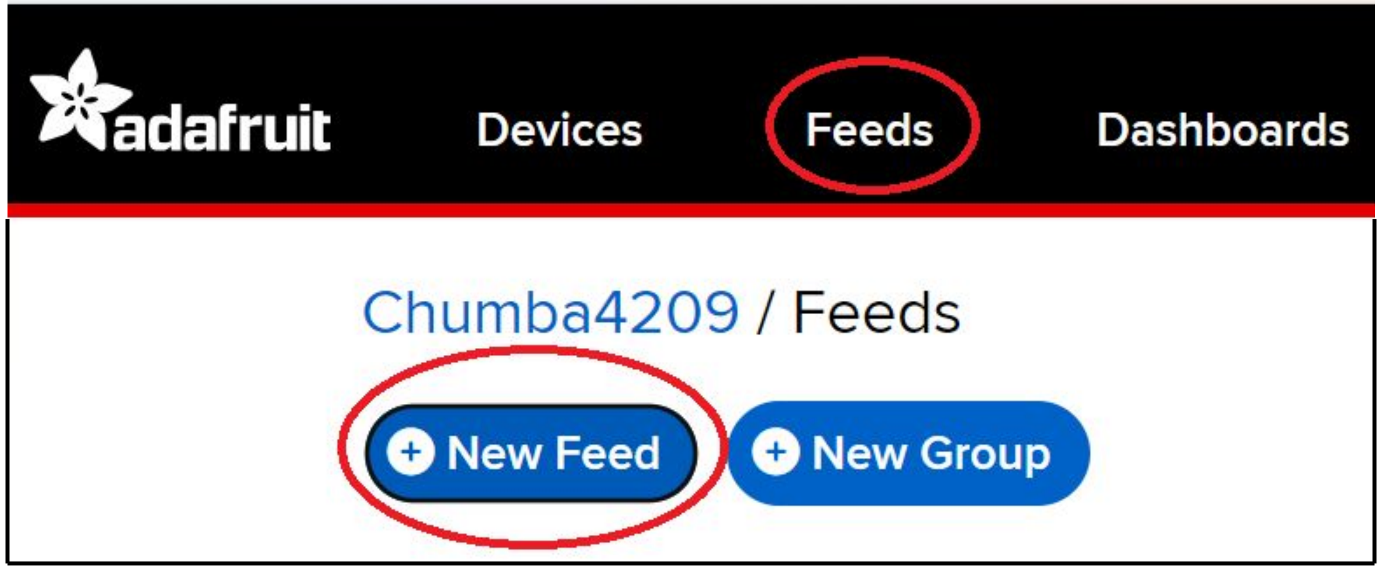
Where do I find the order number?

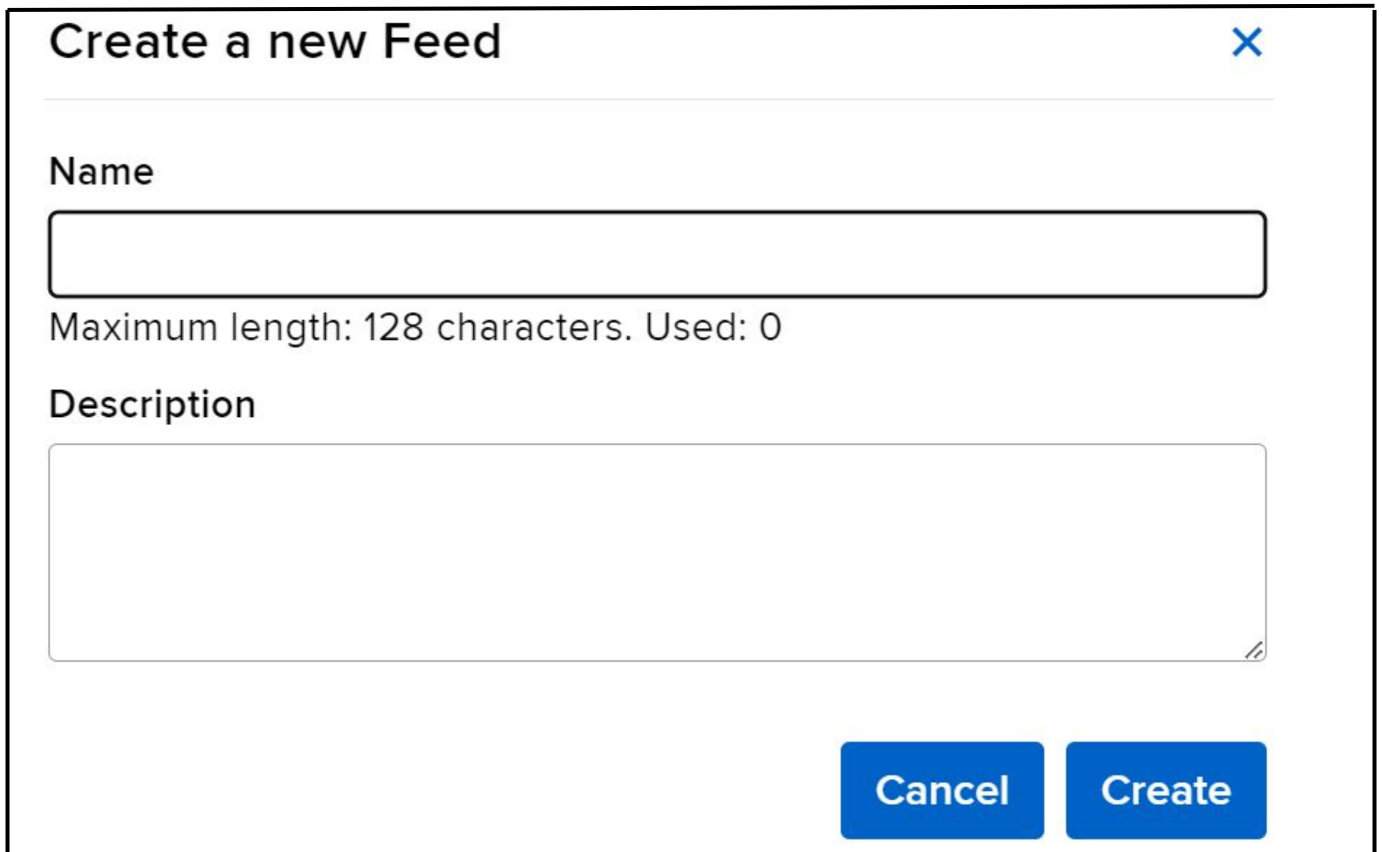
Setting up a new feed

What is a Feed? A feed is a data stream that stores sensor data

Steps:

- Log in to Adafruit IO
- Navigate to the "Feeds" section
- Click "Create New Feed“
- Name your feed (e.g., "Temperature")




Create a new Feed

Name

Maximum length: 128 characters. Used: 0

Description

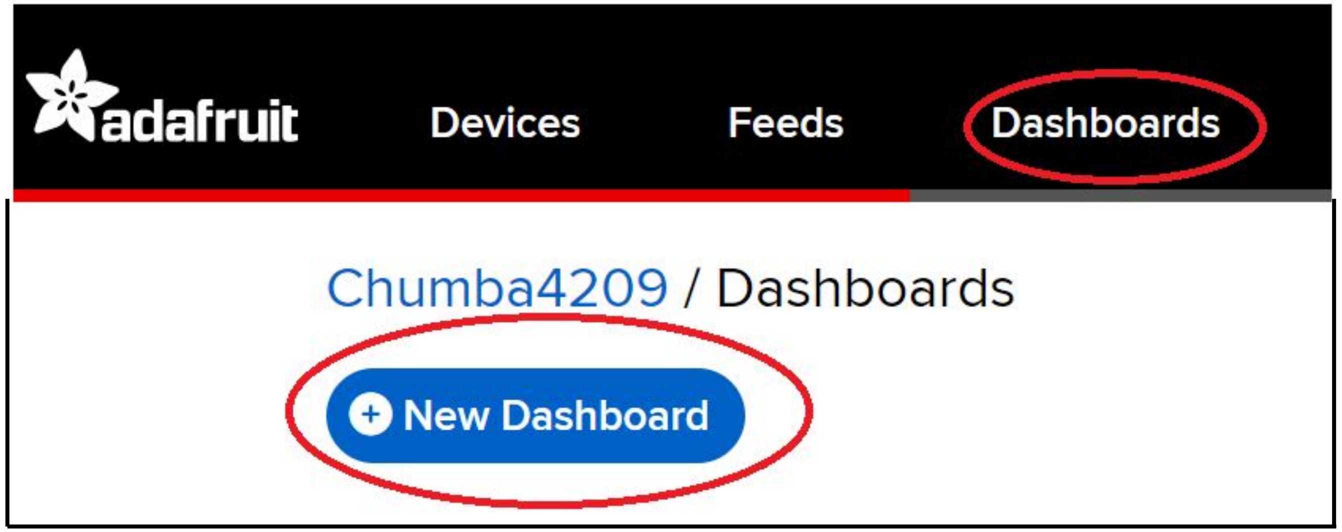
Setting Up the Dashboard

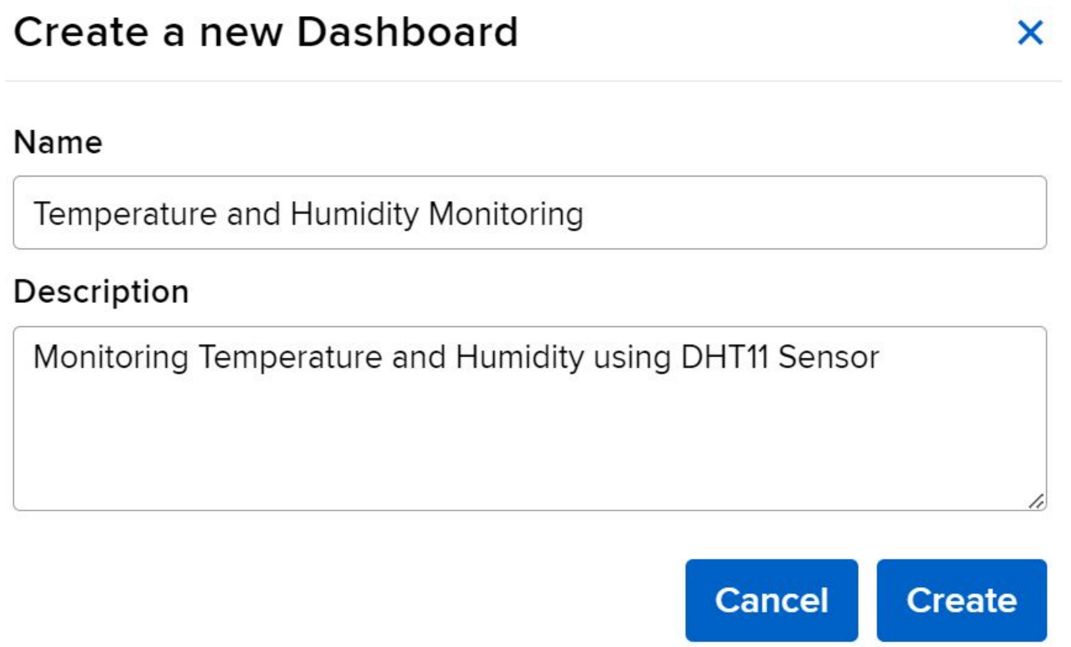
What is a Dashboard?

A dashboard is a customizable interface for visualizing data

Steps:

- Go to the "Dashboards" section
- Click "Create a New Dashboard“
- Name your dashboard (e.g., “Temperature monitoring")




Create a new Dashboard

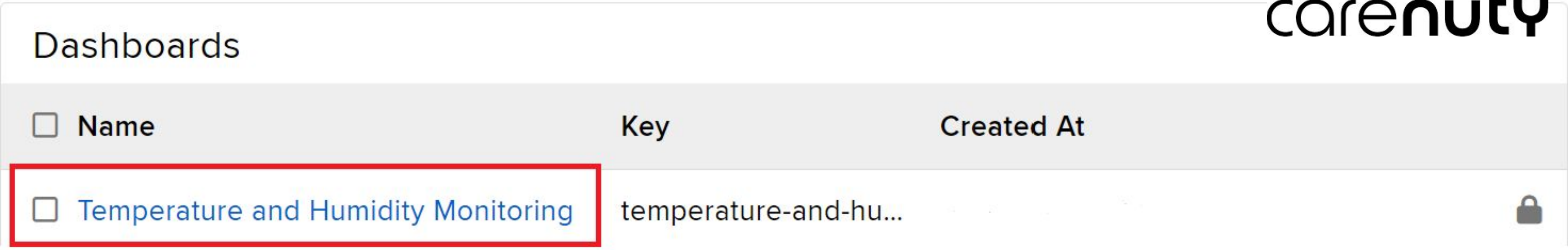
Name

Temperature and Humidity Monitoring

Description

Monitoring Temperature and Humidity using DHT11 Sensor

- Click on the dashboard created

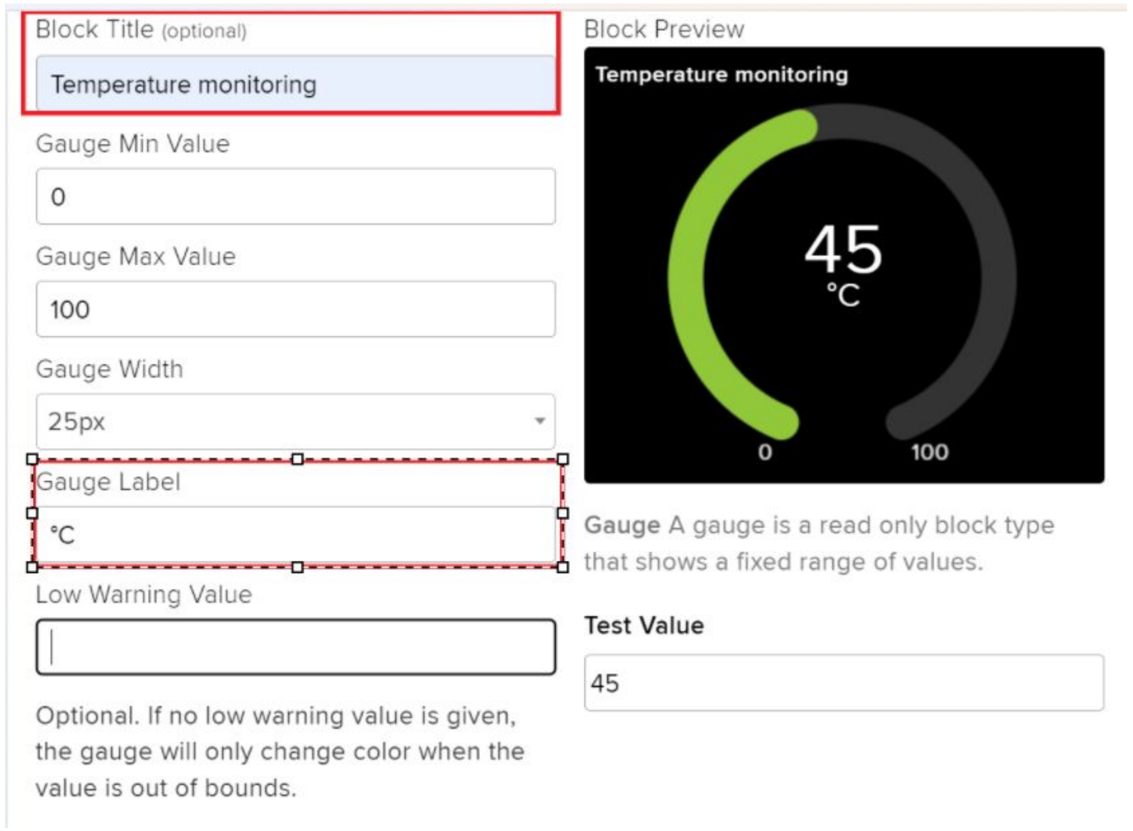


Dashboards

	Name	Key	Created At
☐	Temperature and Humidity Monitoring	temperature-and-hu...	

- Click on the Gear I con and select create new block
- Select the Gauge block

- Select the feed you had created earlier i.e. temperature and click next.
- Input the Block Title and Gauge Level then click create block



Block Title (optional)

Temperature monitoring

Gauge Min Value

0

Gauge Max Value

100

Gauge Width

25px

Gauge Label

°C

Low Warning Value

Optional. If no low warning value is given, the gauge will only change color when the value is out of bounds.

Block Preview

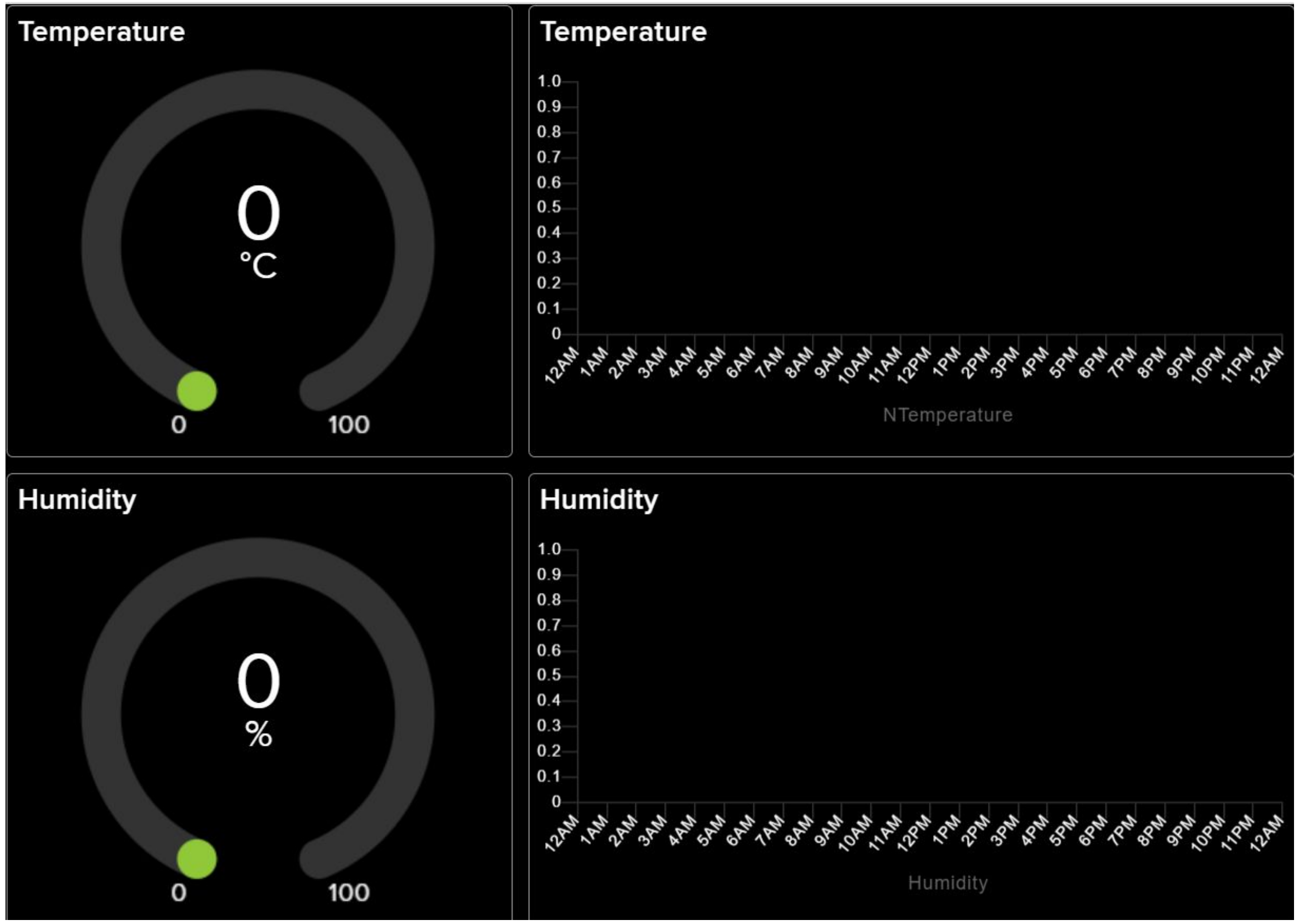
Gauge A gauge is a read only block type that shows a fixed range of values.

Test Value

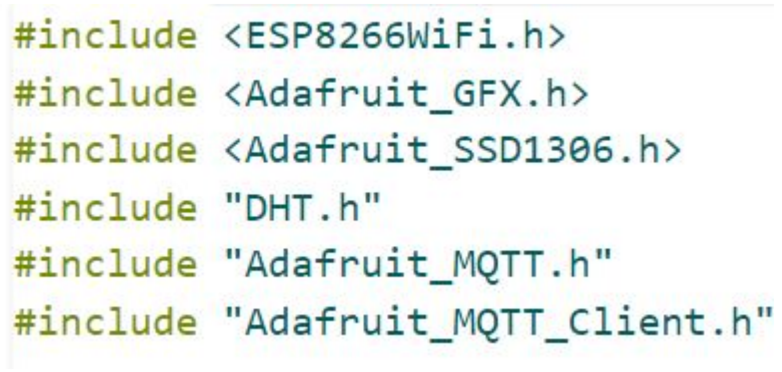
45

- Repeat the same procedure to add the Line chart block for Temperature

- Repeat the same procedure to create a gauge block and Line chart block for Humidity



• Installing the required libraries in Arduino IDE

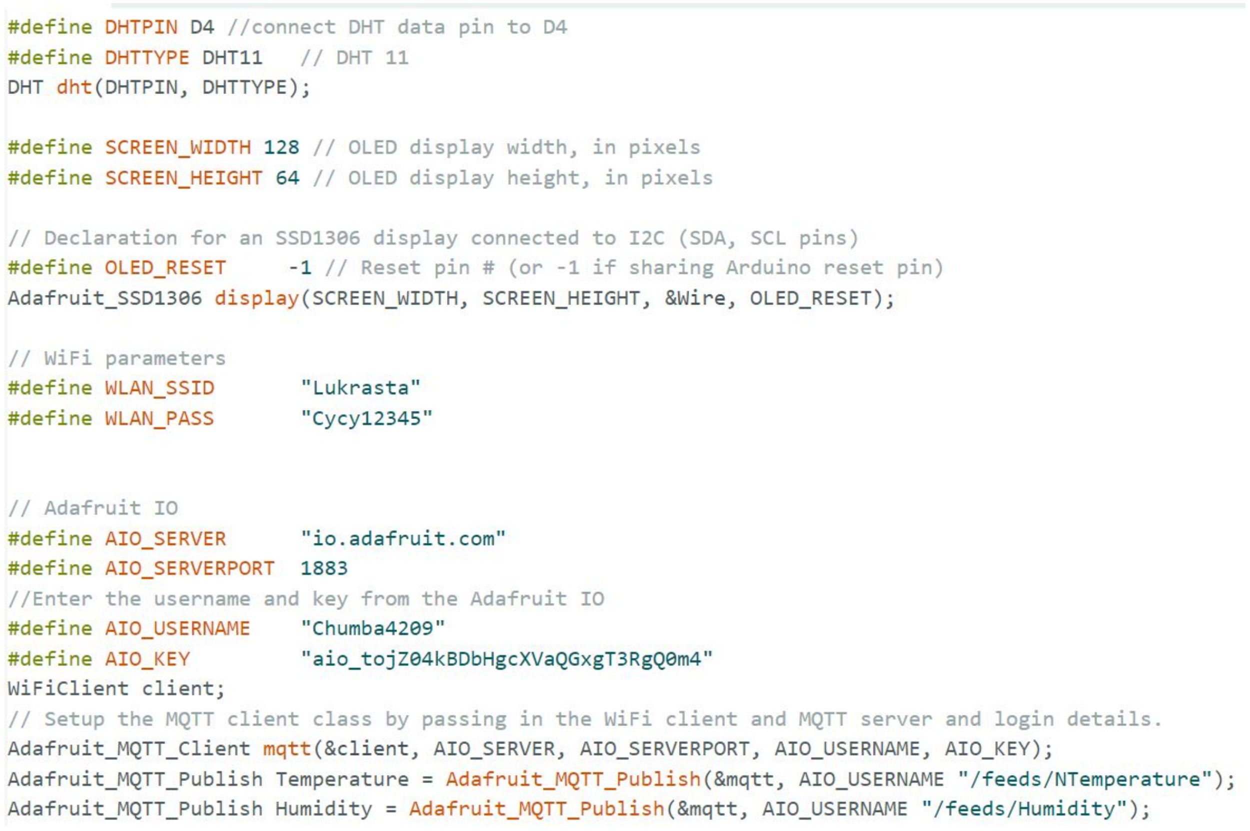


```
#include <ESP8266WiFi.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "DHT.h"
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
```

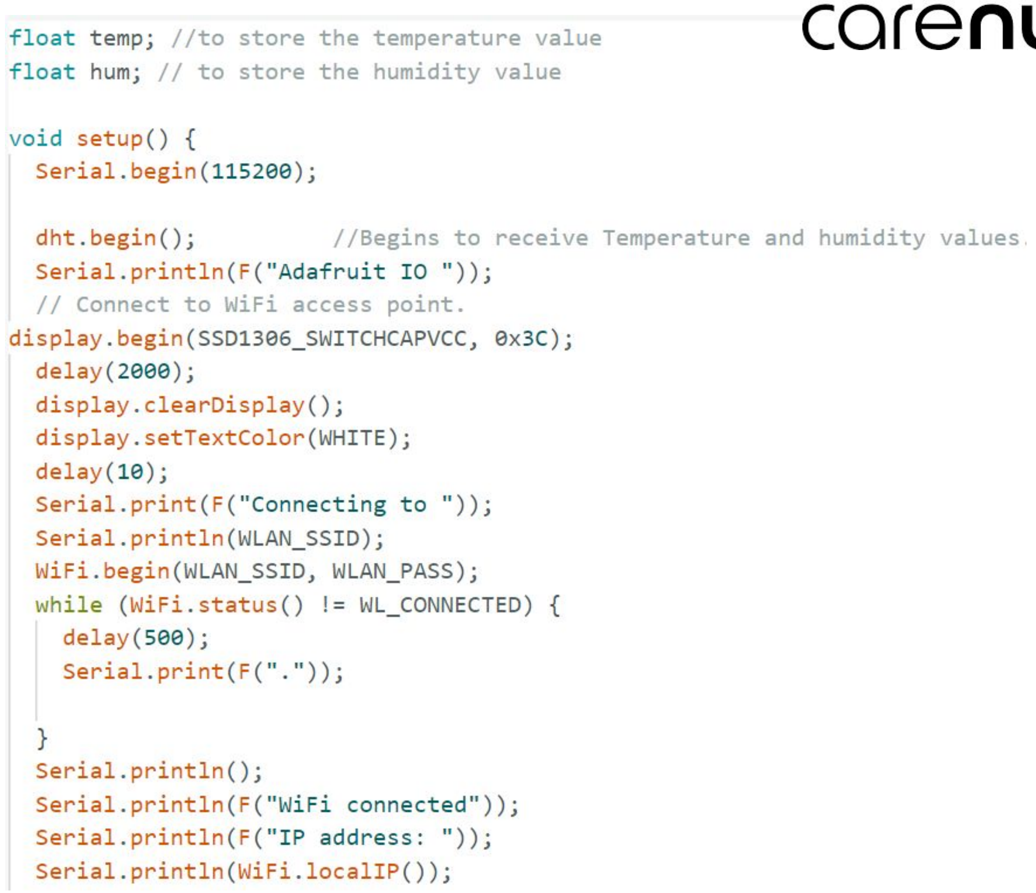
Steps:

- Open Arduino IDE
- Go to "Sketch" -> "Include Library" -> "Manage Libraries"
- Search for and install the required libraries

- A snippet of the code:



```
#define DHTPIN D4 //connect DHT data pin to D4
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHTPIN, DHTTYPE);

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
#define OLED_RESET     -1 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

// WiFi parameters
#define WLAN_SSID       "Lukrasta"
#define WLAN_PASS       "Cycy12345"


// Adafruit IO
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
//Enter the username and key from the Adafruit IO
#define AIO_USERNAME    "Chumba4209"
#define AIO_KEY         "aio_tojZ04kBDbHgcXVaQGxgT3RgQ0m4"
WiFiClient client;
// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
Adafruit_MQTT_Publish Temperature = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/NTemperature");
Adafruit_MQTT_Publish Humidity = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/Humidity");
```



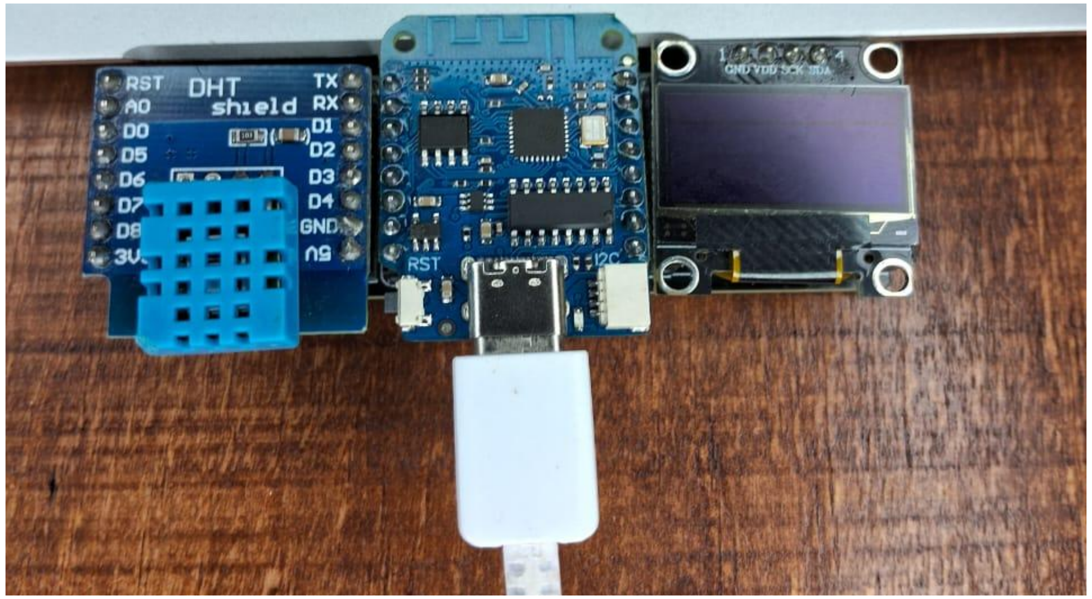
```
float temp; //to store the temperature value
float hum; // to store the humidity value

void setup() {
  Serial.begin(115200);

  dht.begin();           //Begins to receive Temperature and humidity values
  Serial.println(F("Adafruit IO "));
  // Connect to WiFi access point.
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  delay(2000);
  display.clearDisplay();
  display.setTextColor(WHITE);
  delay(10);
  Serial.print(F("Connecting to "));
  Serial.println(WLAN_SSID);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(F("."));

  }
  Serial.println();
  Serial.println(F("WiFi connected"));
  Serial.println(F("IP address: "));
  Serial.println(WiFi.localIP());
```

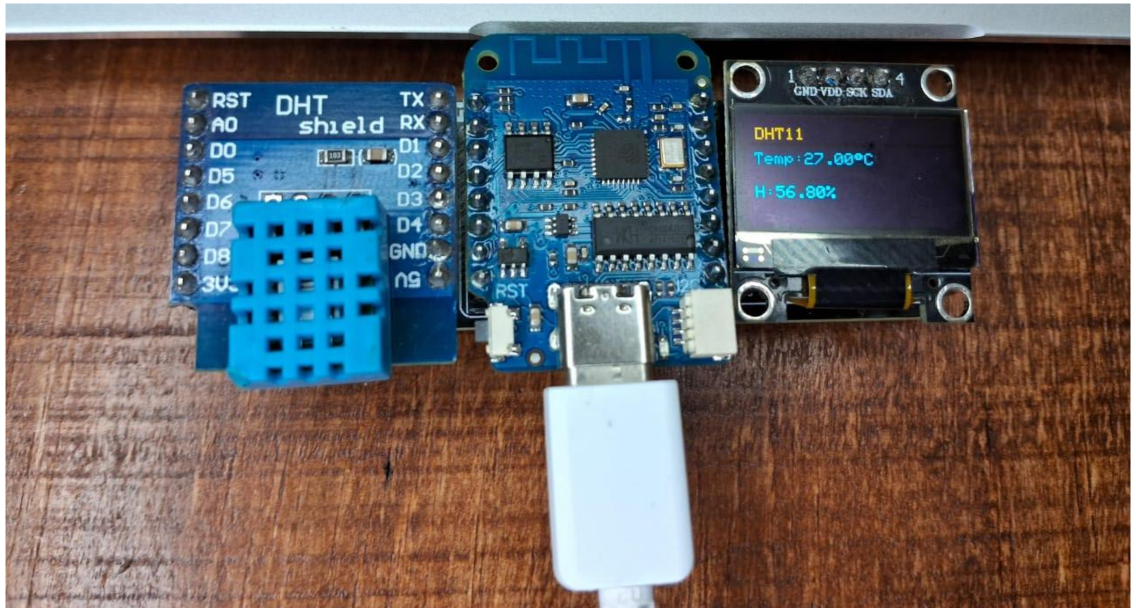
Uploading Code to the D1 Mini

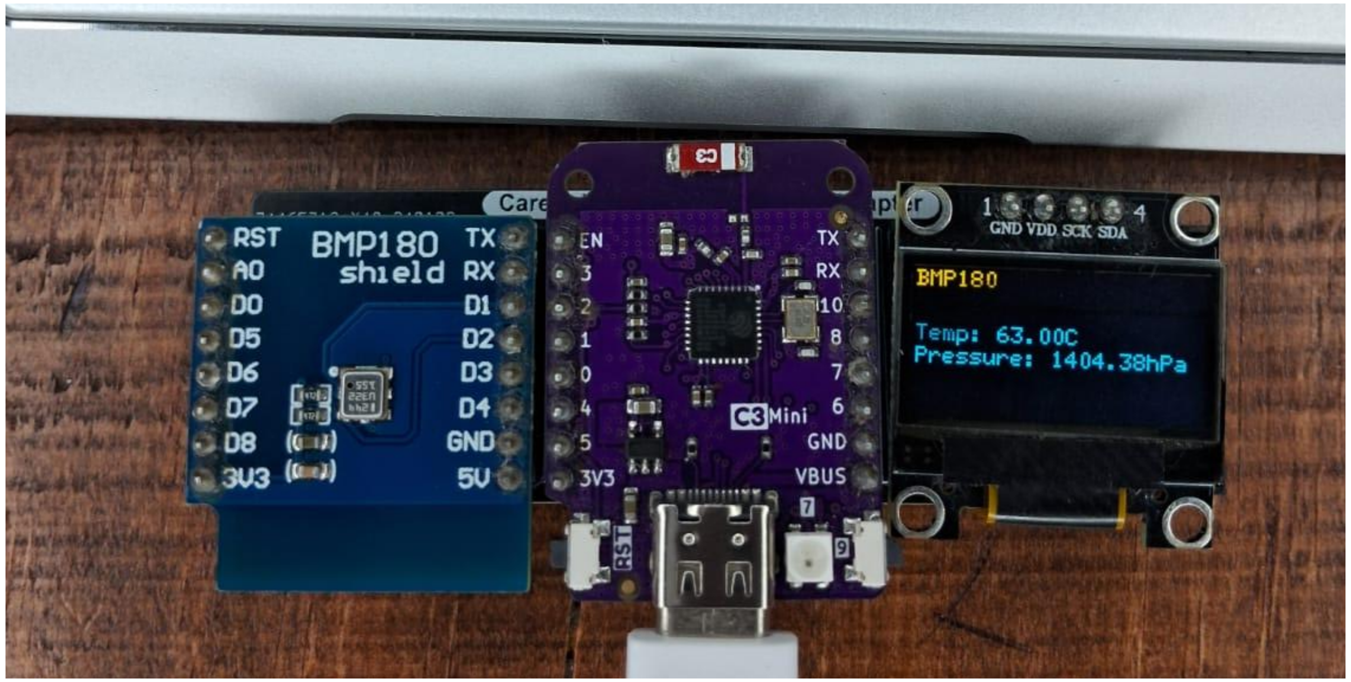
Steps:

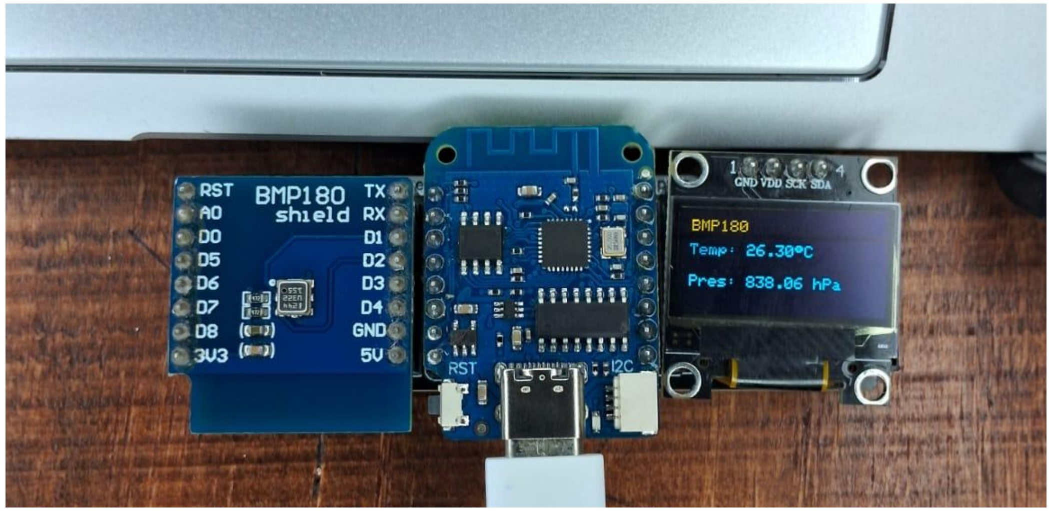
- Connect the D1 Mini to your computer via USB
- Select the appropriate board and port in Arduino IDE
- Click the upload button to flash the code onto the D1 Mini

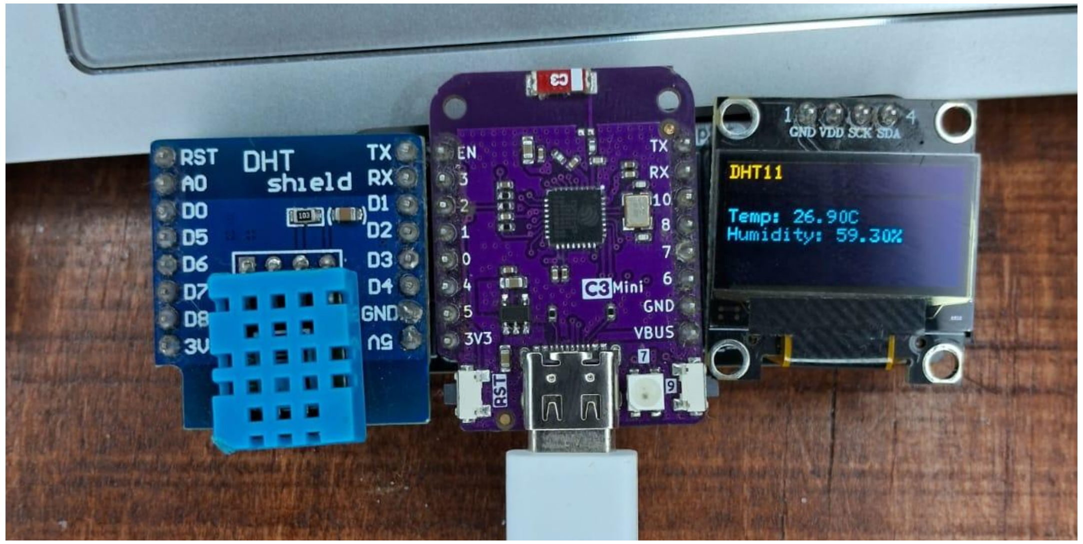
Note:
DS18B20
DHT11
BMP180
D1 Mini, C3 Mini
Add images of C3 Mini (6)



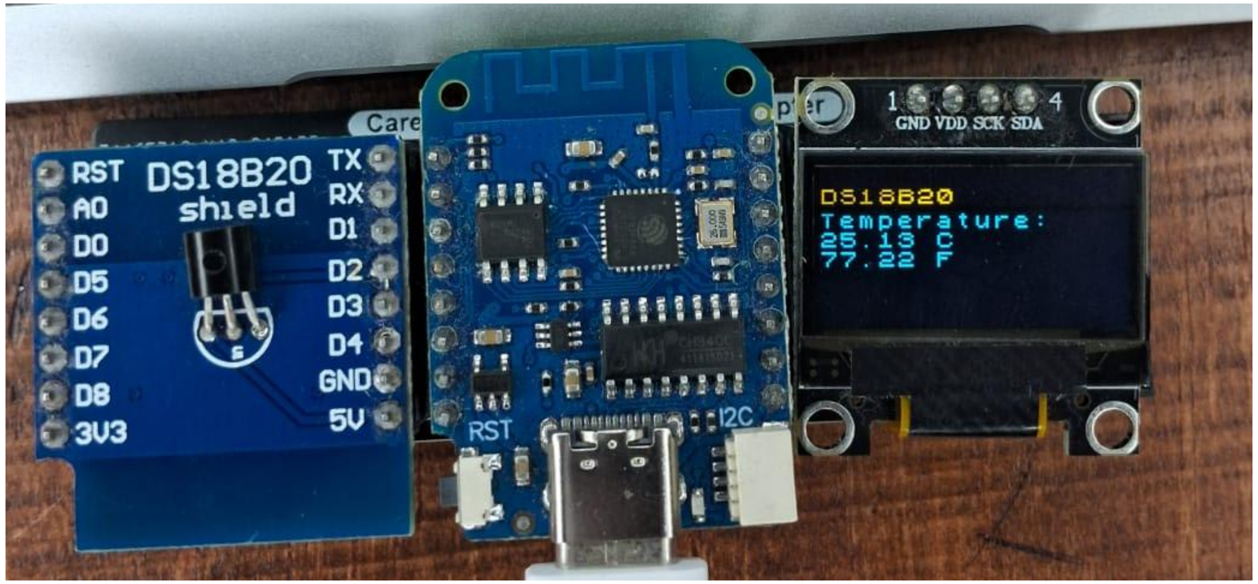



RST
BMP180
shield
A0
D0
D5
D6
D7
D8
3V3
TX
RX
D1
D2
D3
D4
GND
5V
C3
EN
C3 Mini
VBUS
RST
GND VDD SCK SDA
BMP180
Temp: 63.00C
Pressure: 1404.38hPa

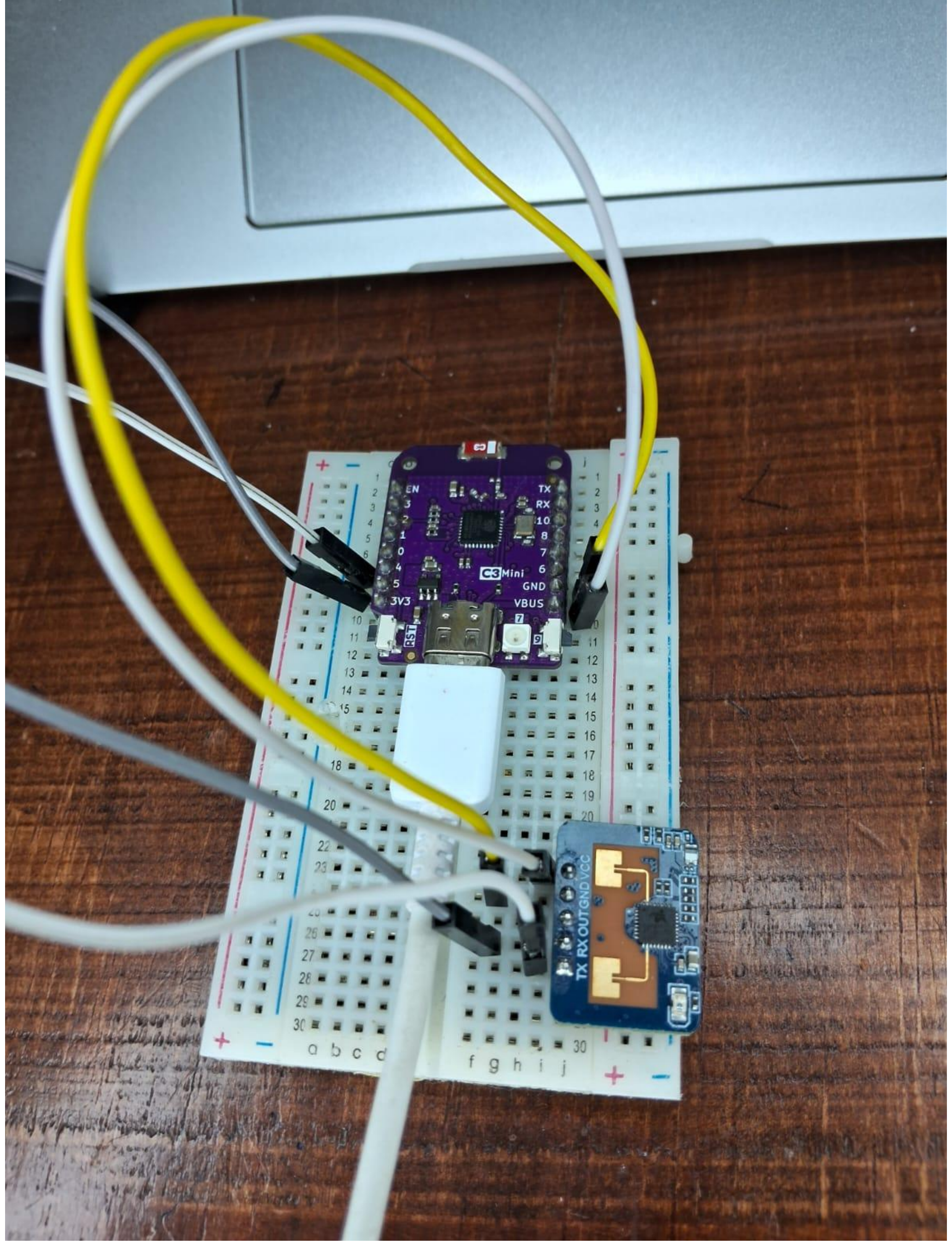

RST
BMP180
shield
A0
D0
D5
D6
D7
D8
3V3
TX
RX
D1
D2
D3
D4
GND
5V
RST
I2C
GND VDD SCK SDA
BMP180
Temp: 26.30°C
Pres: 838.06 hPa



```
DHT11

Temp: 26.90C
Humidity: 59.30%
```

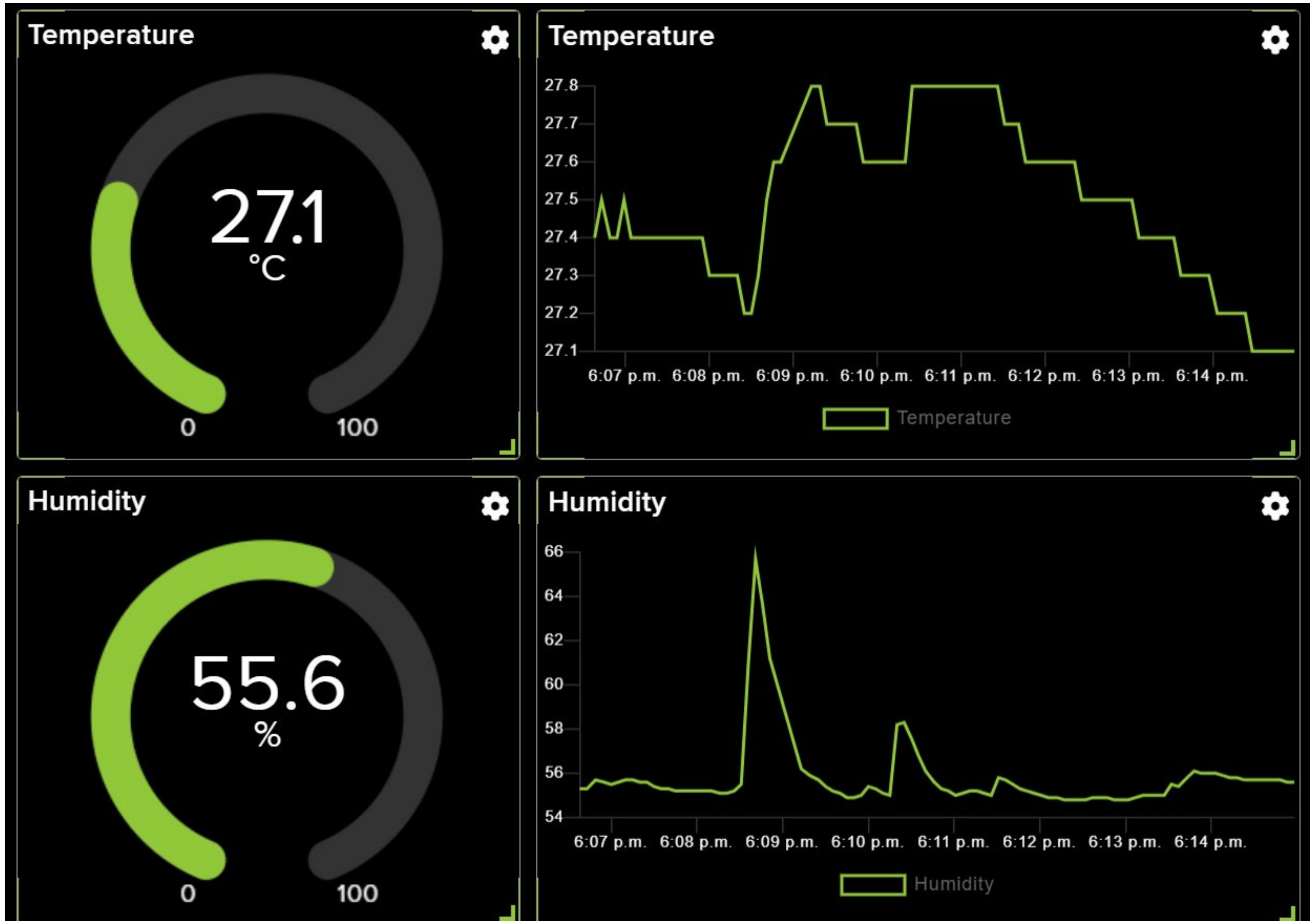


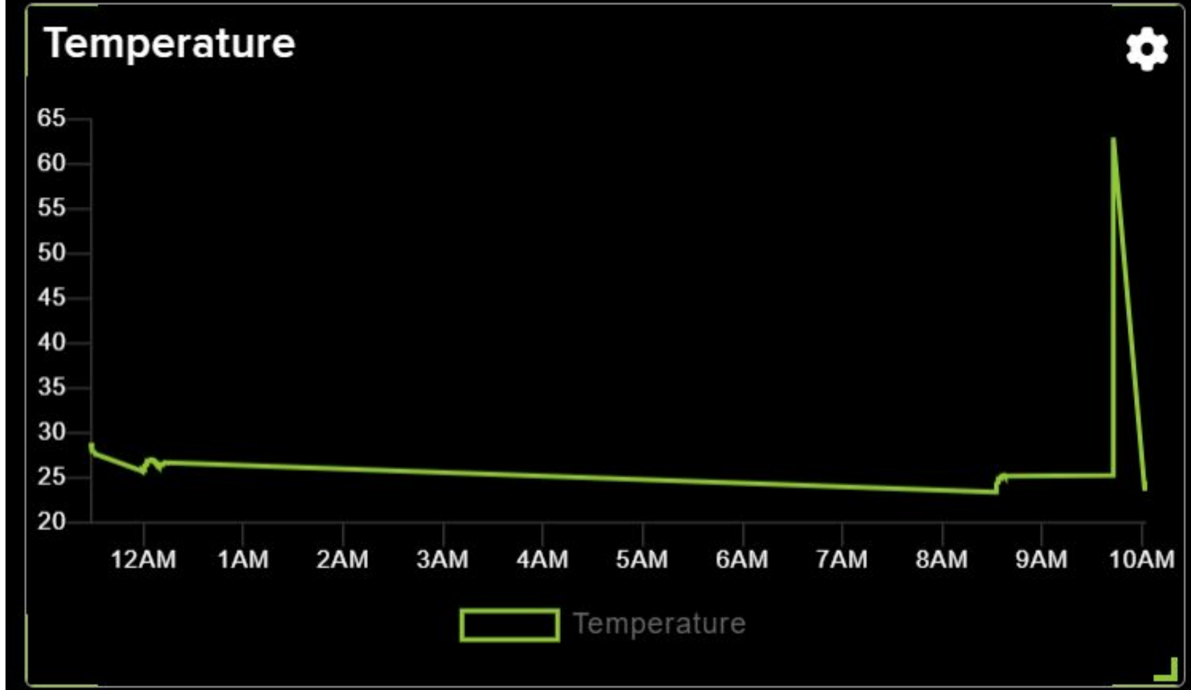
```
DS18B20
Temperature:
25.13 C
77.22 F
```

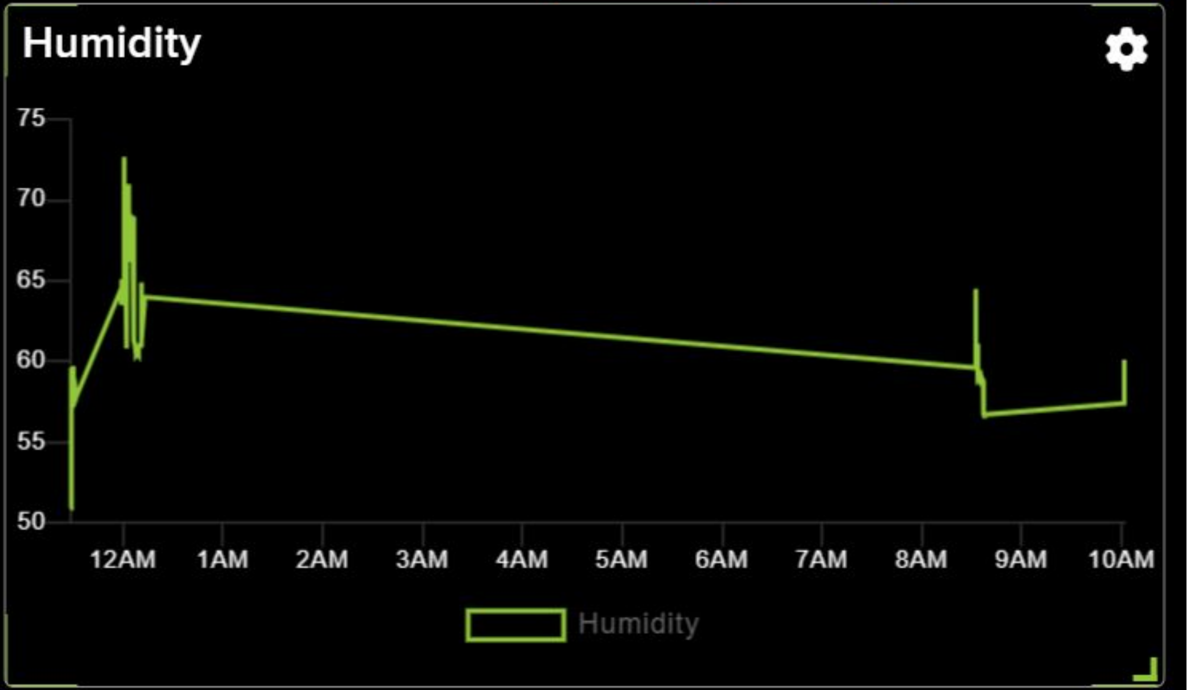


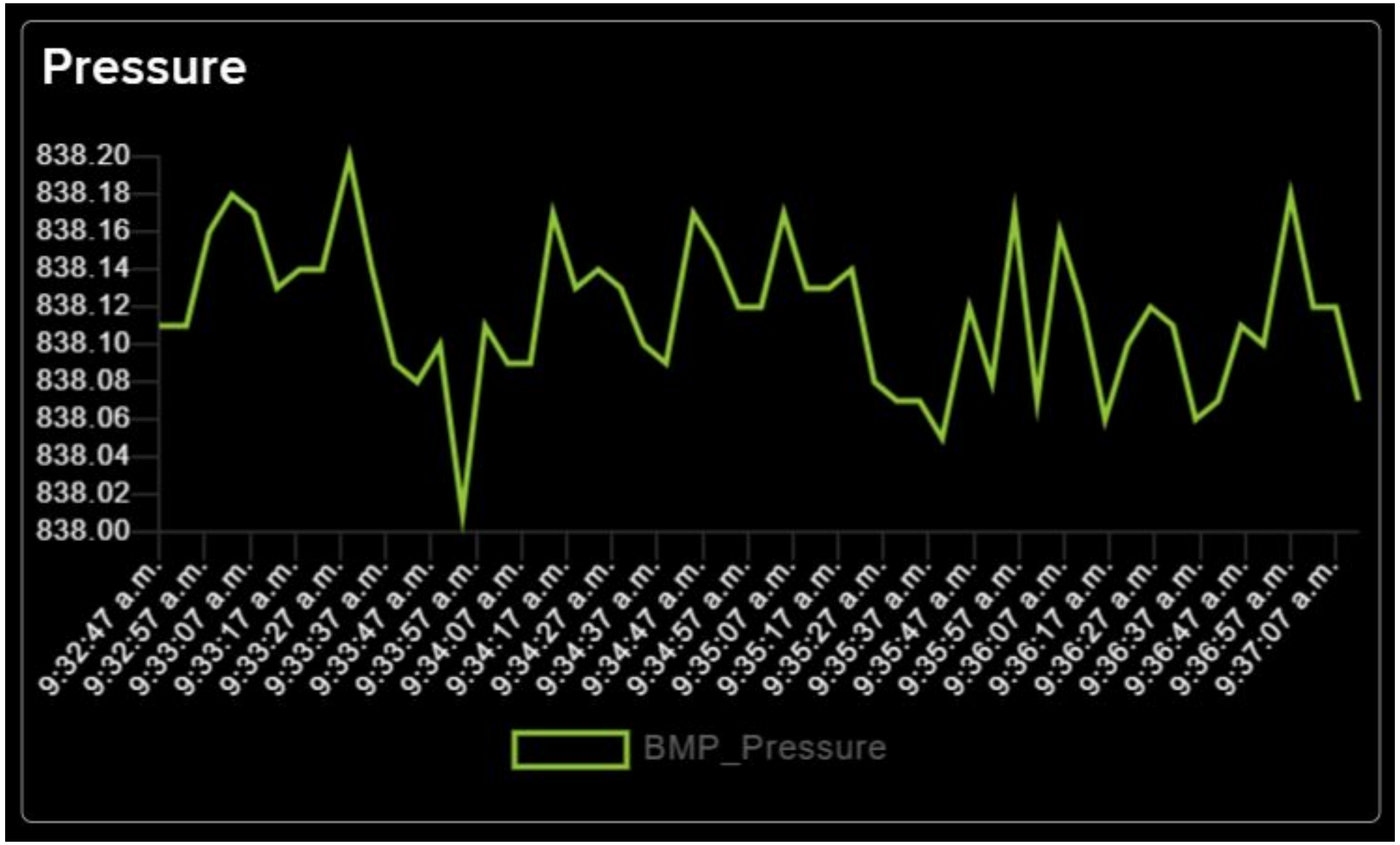
Monitoring Data on Adafruit IO

- Return to your Adafruit IO dashboard
- Watch the real-time data being populated in your widgets




Temperature
65
60
55
50
45
40
35
30
25
20
12AM
1AM
2AM
3AM
4AM
5AM
6AM
7AM
8AM
9AM
10AM
Temperature


Humidity
75
70
65
60
55
50
12AM
1AM
2AM
3AM
4AM
5AM
6AM
7AM
8AM
9AM
10AM
Humidity


Pressure
838.20
838.18
838.16
838.14
838.12
838.10
838.08
838.06
838.04
838.02
838.00
9:32:47 a.m.
9:32:57 a.m.
9:33:07 a.m.
9:33:17 a.m.
9:33:27 a.m.
9:33:37 a.m.
9:33:47 a.m.
9:33:57 a.m.
9:34:07 a.m.
9:34:17 a.m.
9:34:27 a.m.
9:34:37 a.m.
9:34:47 a.m.
9:34:57 a.m.
9:35:07 a.m.
9:35:17 a.m.
9:35:27 a.m.
9:35:37 a.m.
9:35:47 a.m.
9:35:57 a.m.
9:36:07 a.m.
9:36:17 a.m.
9:36:27 a.m.
9:36:37 a.m.
9:36:47 a.m.
9:36:57 a.m.
9:37:07 a.m.
BMP_Pressure

Advantages	**Disadvantages**
User friendly interface for beginners	Data storage and rate limits on the free plan (Limit of 10 feeds,)
Extensive documentation and support	Short data retention in the free plan (Each feed stores data for 30 days)
Wide device compatibility (ESP8266, Arduino, Raspberry Pi, etc.)	Not ideal for very large-scale or enterprise- level projects
Supports MQTT and REST API for communication	Cloud dependent, requiring stable internet connection
Real Time data visualization with customizable widgets	
IFTTT connector enables to move data across the web Zapier connector to automate your work by connecting Adafruit Io to online apps You use	

carenuty

THE END