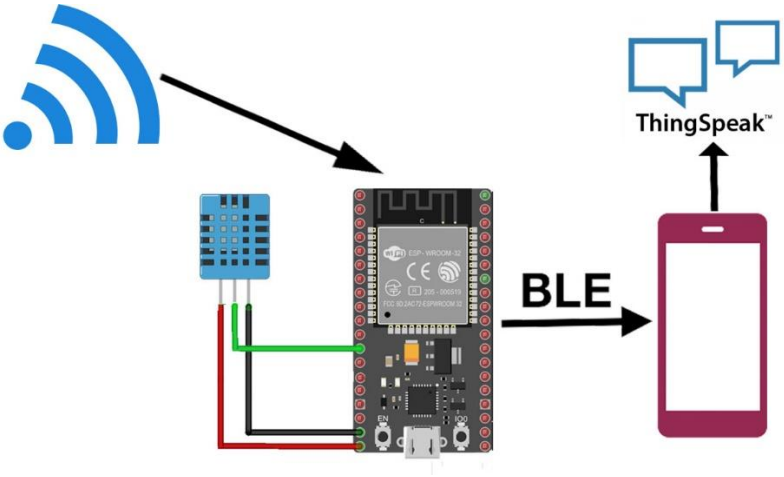
A. System Architecture

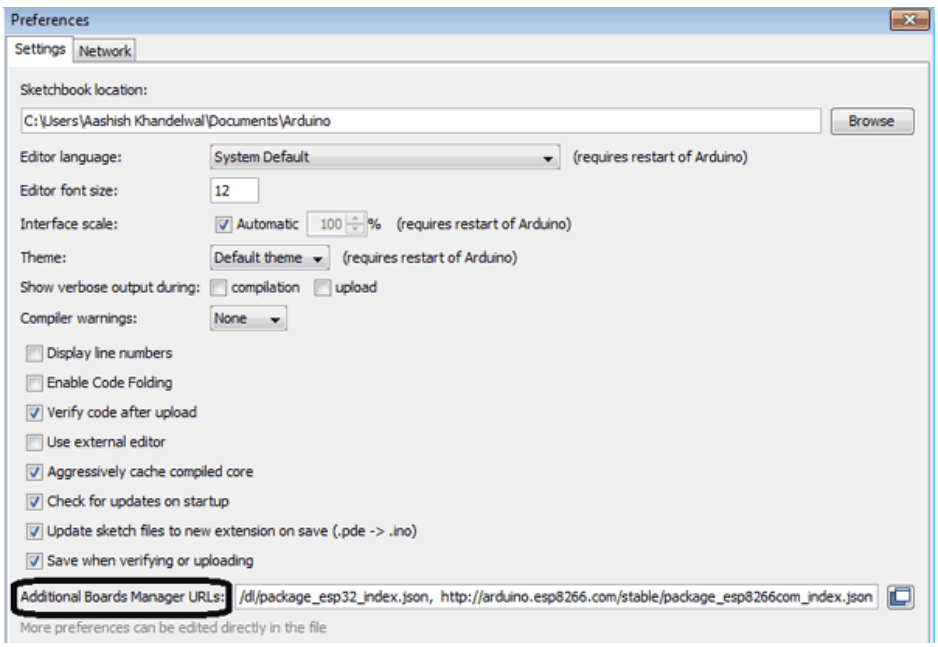


Penjelasan Code, Mempersiapkan ESP32 di Arduino IDE, Flowchart & Psuedocode.

Mempersiapkan ESP32 di Arduino IDE

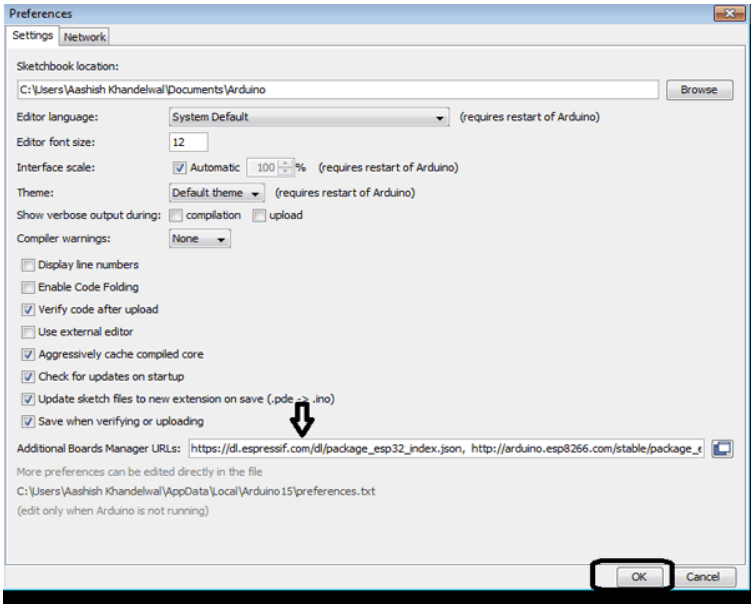
Langkah 1: Pertama Anda harus mengunduh dan menginstal perangkat lunak Arduino IDE yang dapat Anda unduh dari **https://www.arduino.cc/en/Main/Software** secara gratis. Jika Anda telah menginstal di PC Anda, pastikan itu adalah versi IDE terbaru karena versi yang lebih lama tidak menyertakan papan ESP32.

Langkah 2: Setelah menginstal, buka IDE dan buka File -> Preferensi dan buka jendela preferensi dan lihat "URL Boards Manager Tambahan" sebagai:

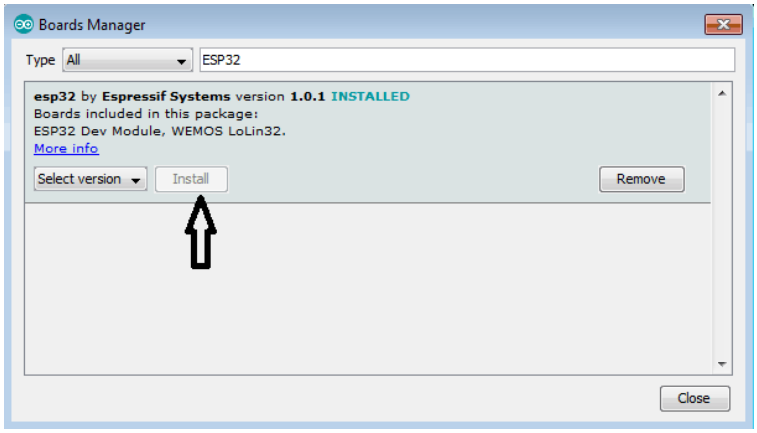


Langkah 3: Kotak ini mungkin kosong atau berisi beberapa URL lain jika Anda telah menggunakannya sebelumnya untuk ESP8266. Anda hanya perlu menempelkan URL di bawah ini ke dalam kotak ini jika kotak tersebut sudah berisi URL lain, lalu tempel dengan memisahkan URL lain menggunakan koma (,).

Langkah 4: Setelah menempelkan URL yang diberikan, jendela saya terlihat seperti ini karena saya sudah menggunakan ESP82666, Sekarang tekan OK dan jendela akan menghilang.



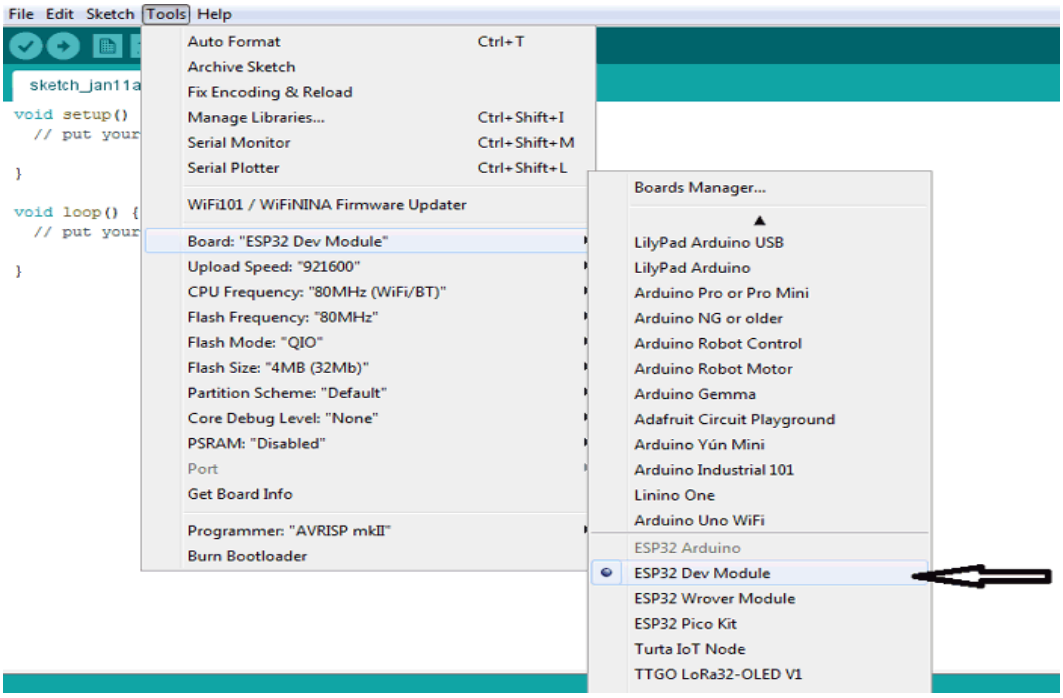
Langkah 5 : Sekarang pergi ke Tools-> Board-> Board Manager dan cari ESP32 dan tekan install , akan memakan waktu untuk menginstal pastikan Anda memiliki koneksi internet saat menginstal setelah menginstal jendela Anda terlihat seperti ini:



Pemrograman ESP32 menggunakan Arduino IDE

untuk mengunggah program di ESP32 kita harus mengikuti beberapa langkah sebagai berikut:

1. Buka Arduino IDE Anda dan buat file baru dan simpan di tempat yang Anda inginkan.
2. Salin kode yang diberikan.
3. Sekarang masuk ke Tools -> Board -> ESP32 Dev Module.



project_jaringan_nirkabel_uas

```
// Library Wifi
#include "WiFi.h"
WiFiClient client;

// Library DHT11 Sensor
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT11

// Deklarasi variabel DHT11
DHT dht(DHTPIN, DHTTYPE);

// Deklarasi variabel thingspeak
String thingSpeakAddress = "api.thingspeak.com";
String writeAPIKey;
String tsfield1Name;
String request_string;
// Library BLE
#include <BLEDevice.h>
#include <BLEUtils.h>
#include <BLEServer.h>
// define uuid BLE
#define SERVICE_UUID        "4fafc201-1fb5-459e-8fcc-c5c9c331914b"
#define CHARACTERISTIC_UUID "beb5483e-36e1-4688-b7f5-ea07361b26a8"
```

Dalam kesempatan kali ini di mohon untuk menyalahkan Wifi laptop terlebih dahulu. Karena ini menggunakan akses ke internet

memasukkan kunci Write API dari server ThingSpeak

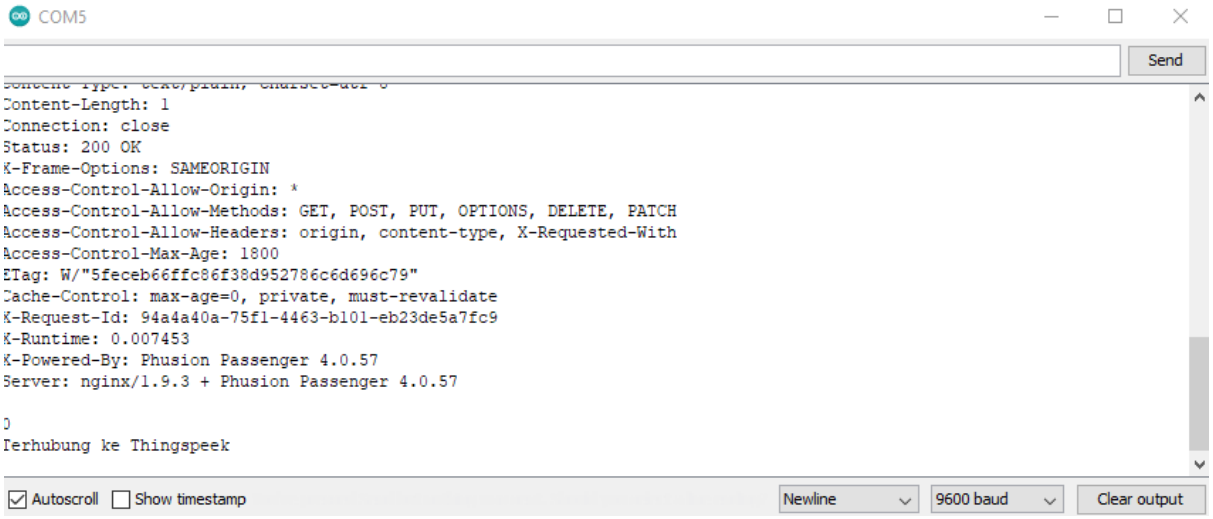

```
// Library Wifi
#include "WiFi.h"
WiFiClient client;

// Library DHT11 Sensor
```

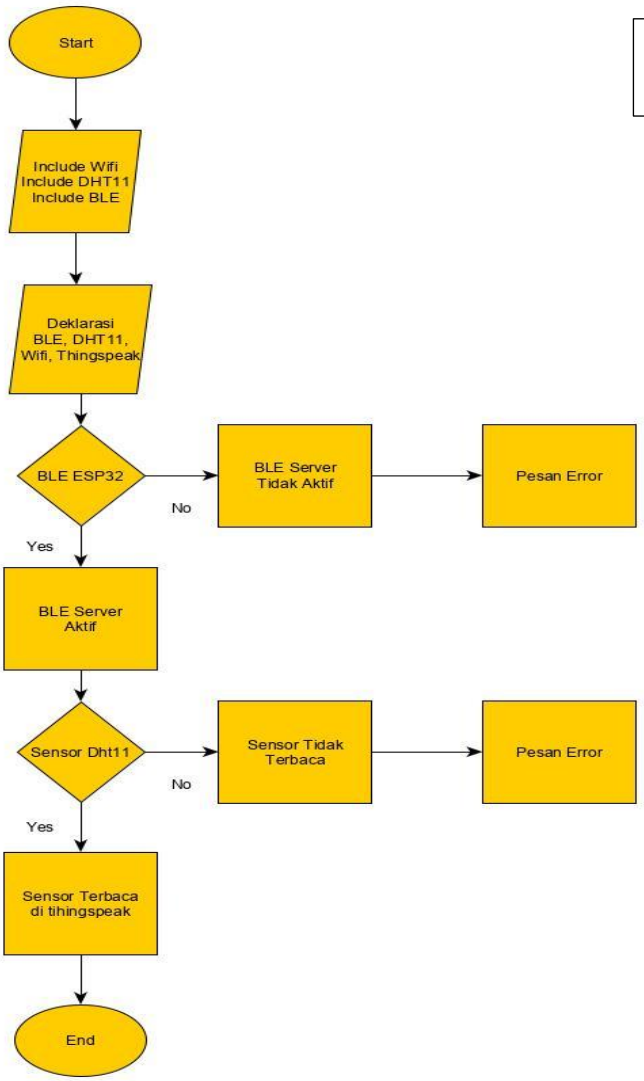
```
#include "DHT.h"
#define DHTPIN 25
#define DHTTYPE DHT11

// Deklarasi variabel DHT11
DHT dht(DHTPIN, DHTTYPE);

// Deklarasi variabel thingspeak
String thingSpeakAddress = "api.thingspeak.com";
String writeAPIKey;
String tsfield1Name;
String request_string;

// Library BLE
#include <BLEDevice.h>
#include <BLEUtils.h>
#include <BLEServer.h>
// define uuid BLE
#define SERVICE_UUID        "4fafc201-1fb5-459e-8fcc-c5c9c331914b"
#define CHARACTERISTIC_UUID "beb5483e-36e1-4688-b7f5-ea07361b26a8"

//Setup setting BLE dan WiFi
void setup() {
  Serial.begin(115200);
  Serial.println("Starting BLE work!");

  BLEDevice::init("ESP32 WROOM BLE");
  BLEServer *pServer = BLEDevice::createServer();
  BLEService *pService = pServer->createService(SERVICE_UUID);
  BLECharacteristic *pCharacteristic = pService->createCharacteristic(
                                         CHARACTERISTIC_UUID,
                                         BLECharacteristic::PROPERTY_READ |
                                         BLECharacteristic::PROPERTY_WRITE
                                       );

  pCharacteristic->setValue("Berhasil Konek ke BLC ESP32");
  pService->start();
  // BLEAdvertising *pAdvertising = pServer-
>getAdvertising();  // this still is working for backward compatibility
  BLEAdvertising *pAdvertising = BLEDevice::getAdvertising();
  pAdvertising->addServiceUUID(SERVICE_UUID);
  pAdvertising->setScanResponse(true);
  pAdvertising->setMinPreferred(0x06);  // functions that help with iPhone connections issue
  pAdvertising->setMinPreferred(0x12);
```

```
  BLEDevice::startAdvertising();
  Serial.println("Smart Phone Woke");

  //WiFi Clinet Connect...

   Serial.begin(9600);
   WiFi.disconnect();
   WiFi.begin("WiFi", "saya2133");
   while ((!(WiFi.status() == WL_CONNECTED))) {
   delay(300);
   Serial.print(".");
   }
   Serial.println("");
   Serial.println("WiFi connected");
   Serial.println("IP address: ");
   Serial.println(WiFi.localIP());
   dht.begin();
}

// Loop proses data sensor dht11
void loop() {
   delay(10000);
   float h = dht.readHumidity();
   float t = dht.readTemperature();
   float f = dht.readTemperature(true);
   // pemangil fungsi void thingspeak.....
   kirim_thingspeak(t, h);
   if (isnan(h) || isnan(t) || isnan(f)) {
   Serial.println("Sensor Tidak Terbaca");
   return;
   }
}

//fungsi thingspeak.. sensor
void kirim_thingspeak(float suhu, float hum) {
   if (client.connect("api.thingspeak.com", 80)) {
   request_string = "/update?";
   request_string += "key=";
   request_string += "Z74CCGG3A8MEAZ8G"; //key thingspeak....
   request_string += "&";
   request_string += "field1";
   request_string += "=";
   request_string += suhu; //variabel void dan pada thingspeak..
   request_string += "&";
```

```
  request_string += "field2";
  request_string += "=";
  request_string += hum; //variabel void dan pada thingspeak..

  // Get manampilkan ke Serial http
  Serial.println(String("GET ") + request_string + " HTTP/1.1\r\n" +
  "Host: " + thingSpeakAddress + "\r\n" +
  "Connection: close\r\n\r\n");

 // Get manampilkan ke HTTP...
  client.print(String("GET ") + request_string + " HTTP/1.1\r\n" +
  "Host: " + thingSpeakAddress + "\r\n" +
  "Connection: close\r\n\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0) {
  if (millis() - timeout > 5000) {
  Serial.println(">>> Client Timeout !");
  client.stop();
   return;
  }
  }
  while (client.available()) {
  String line = client.readStringUntil('\r');
  Serial.print(line);
  }
  Serial.println();
  Serial.println("Terhubung ke Thingspeek");
  }
}
```

Hasil COM Serial pada Arduino IDE





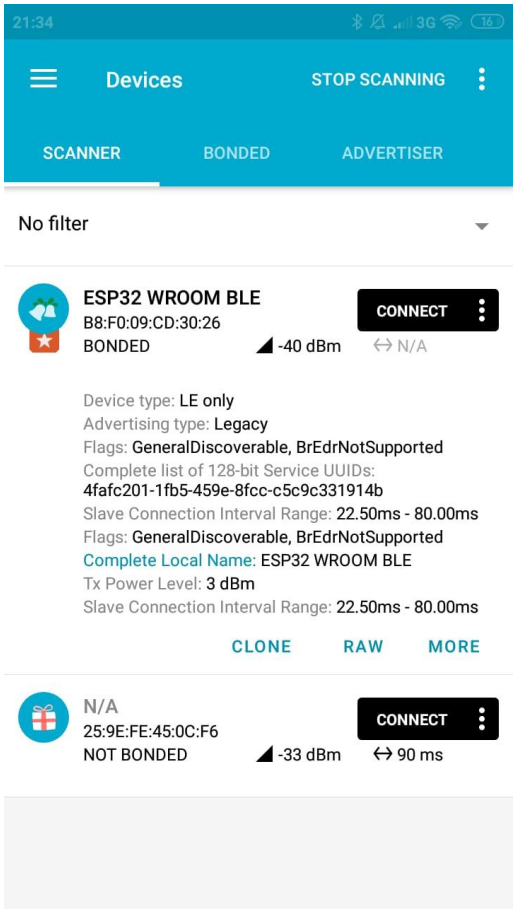
FLOWCHART

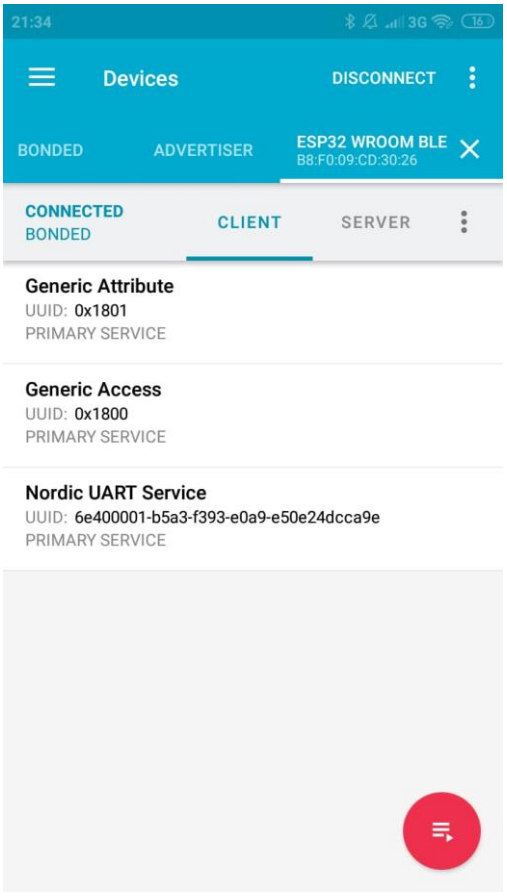
Pseudocode :

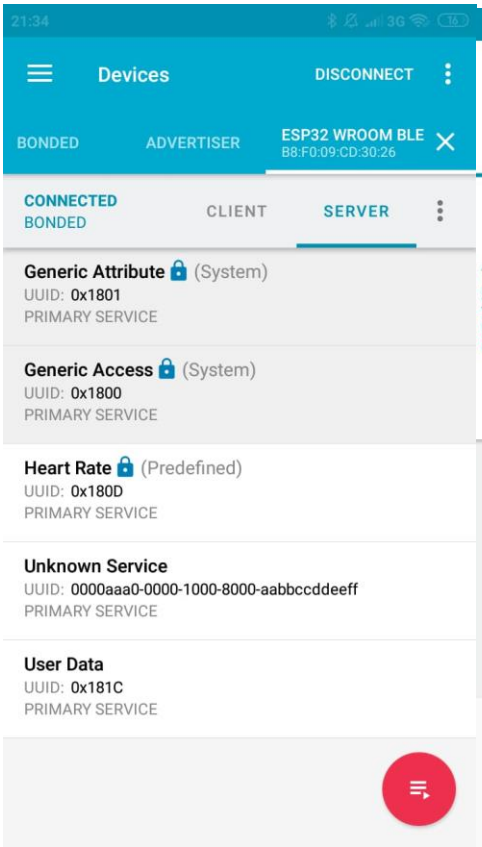
1. Start
2. Input : BLE, Wifi, DHT11

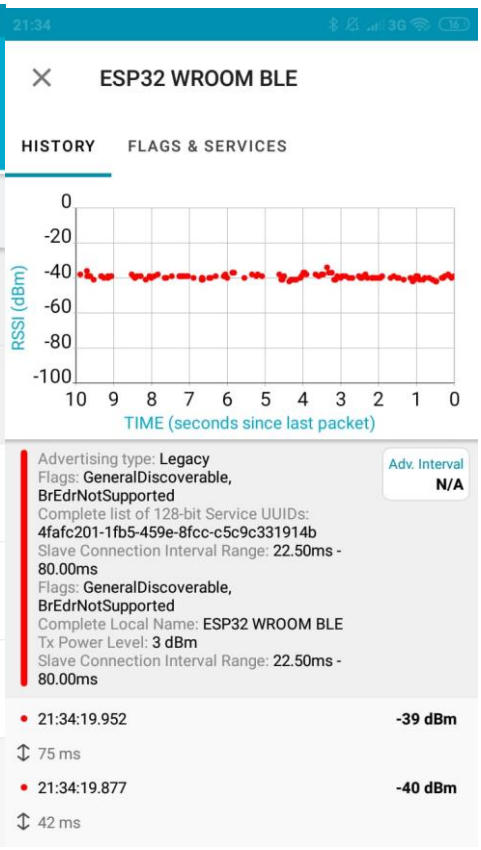
3. Define : pin, UUID,
4. Deklarasi : DHT11, thingspeak
5. Setup : BLE, Wifi
6. If BLE esp32 gagal aktif
   Tampil pesan error
7. If else BLE esp32 berhasil aktif
   Menghubungkan ke DHT11
8. If DHT11 tidak terbaca
   Tampil pesan error
9. If else DHT11 terbaca
   Terbaca di thingspeak
10. End

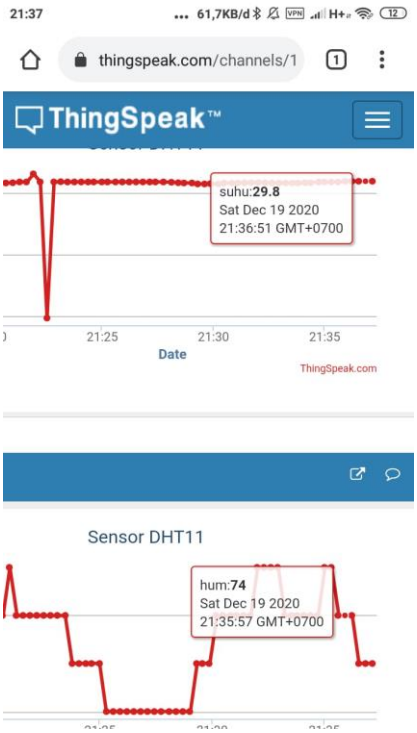
1. Pengujian trigger sensor & Monitoring cloud

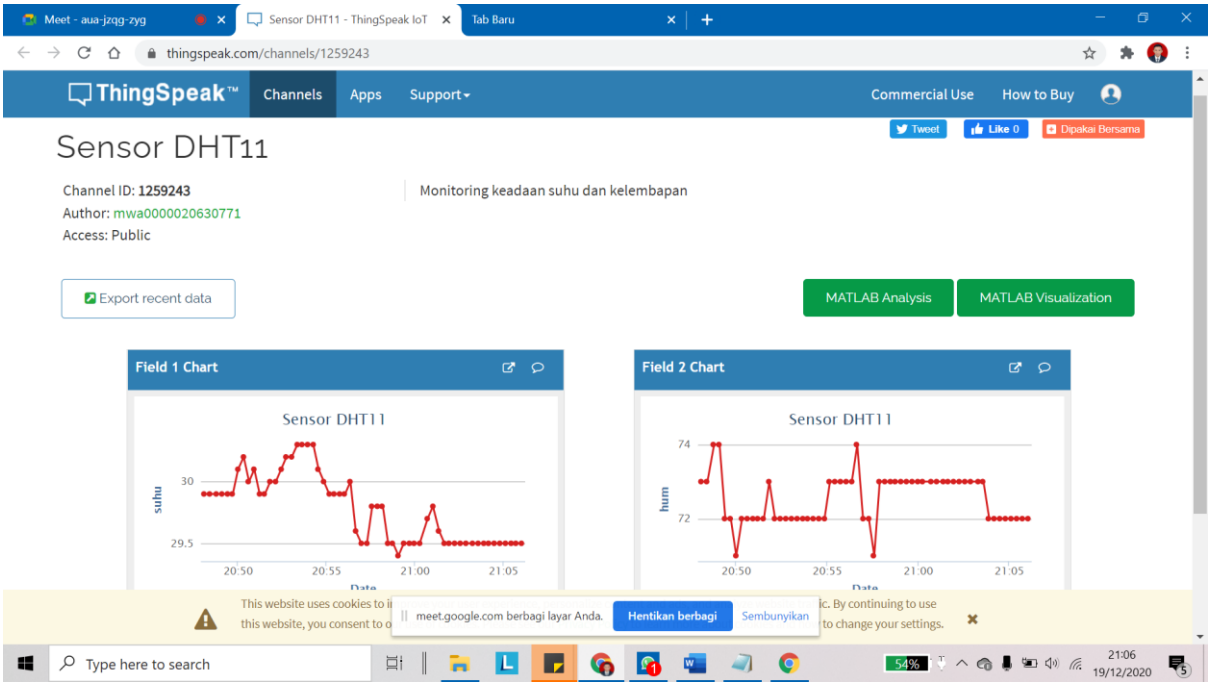





21:34
Devices
DISCONNECT
BONDED
ADVERTISER
ESP32 WROOM BLE
B8:F0:09:CD:30:26
CONNECTED
BONDED
CLIENT
SERVER
Generic Attribute (System)
UUID: 0x1801
PRIMARY SERVICE
Generic Access (System)
UUID: 0x1800
PRIMARY SERVICE
Heart Rate (Predefined)
UUID: 0x180D
PRIMARY SERVICE
Unknown Service
UUID: 0000aaa0-0000-1000-8000-aabbccddeeff
PRIMARY SERVICE
User Data
UUID: 0x181C
PRIMARY SERVICE


21:34
ESP32 WROOM BLE
HISTORY
FLAGS & SERVICES
RSSI (dBm)
0
-20
-40
-60
-80
-100
10 9 8 7 6 5 4 3 2 1 0
TIME (seconds since last packet)
Advertising type: Legacy
Flags: GeneralDiscoverable, BrEdrNotSupported
Complete list of 128-bit Service UUIDs: 4fafc201-1fb5-459e-8fcc-c5c9c331914b
Slave Connection Interval Range: 22.50ms - 80.00ms
Flags: GeneralDiscoverable, BrEdrNotSupported
Complete Local Name: ESP32 WROOM BLE
Tx Power Level: 3 dBm
Slave Connection Interval Range: 22.50ms - 80.00ms
Adv. Interval
N/A
21:34:19.952
-39 dBm
75 ms
21:34:19.877
-40 dBm
42 ms


21:37
61,7KB/d
thingspeak.com/channels/1
ThingSpeak™
suhu:29.8
Sat Dec 19 2020
21:36:51 GMT+0700
21:25
21:30
21:35
Date
ThingSpeak.com
Sensor DHT11
hum:74
Sat Dec 19 2020
21:35:57 GMT+0700



Link : https://thingspeak.com/channels/1259243

2. Kesimpulan

Dari project yang sudah kami kerjakan tersebut ble pada ESP32 bertindak sebagai server sedangkan smartphone ble bertindak sebagai client. Untuk melakukan monitor suhu dan kelembapan menggunakan sensor DHT11 dan menampilkan data dari sensor DHT11 ke cloud thingspeak. Agar dapat terhubung ke cloud thingspeak maka pada esp32 harus sudah terhubung ke internet. Koneksi internet yang kita gunakan yaitu dengan tethering smartphone. Untuk mengetahui ble server sudah berjalan pada client(Smartphone), kami menggunakan tools nRF Connect yang ada pada playstore.