ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

BÁO CÁO TIỂU LUẬN

Môn học: **ĐO LƯỜNG VÀ ĐIỀU KHIỂN BẰNG MÁY TÍNH**

Đề tài: Kết nối wifi điện thoại với ESP32 sử dụng Hybrid Mobile App thu thập và điều khiển

GVHD: Nguyễn Trọng Tài

Danh sách thành viên:

Phạm Gia Huy	1812414
Trần Đức Huy	1812422
Hồ Nghĩa Gia Bảo	1810034

TÓM TẮT

Hiện nay xã hội phát triển, kỹ thuật ngày càng hiện đại nên nhu cầu về trao đổi thông tin giải trí, nhu cầu về điều khiển các thiết bị từ xa,…ngày càng cao. Và những hệ thống dây cáp phức tạp lại không thể đáp ứng nhu cầu này, nhất là ở những khu vực chật hẹp, những nơi xa xôi, trên các phương tiện vận chuyển,… Vì vậy công nghệ không dây đã ra đời và phát triển mạnh mẽ, tạo rất nhiều thuận lợi cho con ngọời trong đời sống hằng ngày. Trong những năm gần đây công nghệ truyền nhận dữ liệu không dây đang có những bước phát triển mạnh mẽ, góp công lớn trong việc phát triển các hệ thống điều khiển, giám sát từ xa, đặc biệt là các hệ thống thông minh. Hiện nay, có khá nhiều công nghệ không truyền nhận dữ liệu không dây như RF, Wifi, Bluetooth, NFC,…Trong đó, Wifi là một trong những công nghệ được phát triển từ lâu và luôn được cải tiến để nâng cao tốc độ cũng như khả năng bảo mật. Trên thị trường Việt Nam hiện nay có nhiều sản phẩm điều khiển thiết bị không dây, nhưng đa số những sản phẩm hiện có đều là nhập khẩu từ nước ngoài với giá thành cao. Việc nghiên cứu và thiết kế một bộ sản phẩm thu thập dữ liệu và điều khiển thiết bị không dây có một ý nghĩa lớn, giúp tăng thêm sự lựa chọn cho người sử dụng, sản phẩm được sản xuất trong nước nên giá thành rẻ và góp phần phát triển các hệ thống điều khiển thông minh. Do đó, chúng em quyết định thực hiện đề tài: “ Bộ điều khiển thiết bị sử dụng ESP32 thông qua Hybrid Mobile App để thu thập dữ liệu và điều khiển ”. Đề tài ứng dụng công nghệ Wifi phổ biến trên nhiều thiết bị, Bộ điều khiển có thể điều khiển, đặc biệt điểm mới của đề tài so với các sản phẩm hiện có là điều khiển thông qua hệ điều hành Android giúp tận dụng những thiết bị sử dụng hệ điều hành Android có sẵn của ngọời dùng giúp giảm giá thành sản phẩm. Tuy nhiên, đây chỉ là sản phẩm nghiên cứu, còn có nhiều mặt hạn chế, hy vọng sẽ phát triển hoàn thiện nhất trong tương lai.

MỤC LỤC

1. GIỚI THIỆU

1.1 Mục đích

- Chúng ta sẽ xây dựng ứng dụng dùng cảm biến DHT11 (cảm biến nhiệt độ và độ ẩm) để thu thập thông qua module ESP32. Thông tin về nhiệt độ và độ ẩm sẽ được hiển thị trên app.

1.2 Nhiệm vụ đề tài

- Khi thiết kế ứng dụng, người dùng cần một giao diện giám sát và điều khiển thân thiện, đồng thời có thể phát triển thêm các tính năng như hiển thị kết quả dưới dạng đồ thị (chart), lưu trữ dữ liệu theo thời gian chỉ định hay điều khiển trạng thái các thiết bị chỉ với 1 click chuột trên máy tính. Các dự án với mô hình phức tạp sẽ cần quản lí các kết nối cũng như dữ liệu của các thiết bị…
- Chúng ta sẽ giải quyết những vấn đề trên thông qua ứng dụng đọc giá trị nhiệt độ và độ ẩm hiện tại và gửi về server. Đây là một ứng dụng khá đơn giản, hữu ích và dễ làm. Thông qua phần này chúng ta có thể xây dựng được một ứng dụng IoT thực tế, nắm bắt được các kiến thức cơ bản về thu thập dữ liệu, xây dựng thiết bị và server.

2. LÝ THUYẾT

2.1 Mobile App

2.1.1 Tổng quan

Trong kỷ nguyên thông tin, hiện tượng này là ngành công nghiệp kỹ thuật số tạo ra một xã hội dựa trên tri thức bao quanh bởi một nền kinh tế toàn cầu công nghệ cao, bao trùm ảnh hưởng của nó đến phương pháp sản xuất trong toàn ngành và lĩnh vực dịch vụ được vận hành một cách hiệu quả và thuận tiện. Thời đại thông tin được hình thành bằng cách tận dụng sự tiến bộ của kỹ thuật thu nhỏ kích cỡ máy tính. Sự tiến triển của công nghệ trong đời sống hàng ngày và tổ chức xã hội đã dẫn đến sự

hiện đại hóa tiến trình thông tin và truyền thông đã trở thành động lực của sự tiến hóa về mặt xã hội.

Dưới tác động của đại dịch COVID-19 đã gây ra rất nhiều xáo trộn cho nền kinh tế và hành vi của người tiêu dùng cũng có sự thay đổi mạnh mẽ khi tập người dùng số tại Việt Nam gia tăng, khiến cho dư địa phát triển cho các ngành công nghiệp xoay quanh hệ sinh thái di động còn rất lớn.

a/ <u>Phân loại</u>

- Thời điểm hiện tại có 2 ứng dụng mà chúng ta thường sử dụng đó là Native Mobile App và Hybrid Mobile App.

  Native Mobile App : Native Mobile App hay còn được gọi là ứng dụng gốc - là ứng dụng được viết riêng cho một loại hệ điều hành như iOS, Android, Windows bằng ngôn ngữ lập trình tương ứng. Ứng dụng này có đầy đủ công cụ hỗ trợ, giúp các tính năng vận hành trên các hệ điều hành có tốc độ mượt mà tốt , mọi quá trình không cần phải thông qua Engine hay một ứng dụng trung gian nào khác. Đối với mỗi hệ điều hành Native sẽ có một một ngôn ngữ tương thích với chương trình tạo ra. Đối với các ứng dụng phục vụ trên Android, sẽ sử dụng chủ yếu bằng hai ngôn ngữ là Kotlin và Java. Còn đối với iOS, thì các chương trình sẽ được tạo ra nhờ sự hỗ trợ của Swift (hay Objective C).

  Hybrid Mobile App : Trước khi tìm hiểu về Hybrid Mobile App, ta cần biết về Web App, nó là các ứng dụng được viết trên nền tảng browser để người dùng có thể sử dụng và tương tác ngay trên đó.

<u>Ví dụ</u>: Các web game (slither.io) hoặc những mini game trên Facebook thỉnh thoảng ta được bạn bè mời chơi.

Đây là ứng dụng lai và từ Hybrid ở đây có nghĩa là hỗn hợp. Ứng dụng này là sự kết hợp giữa những ưu điểm của Native App và Web App. Ứng dụng này cũng giống với hầu hết những ứng dụng khác dành cho điện thoại di động , chúng được tạo dưới dạng một ứng dụng duy nhất để sử dụng trên nhiều nền tảng như Android, IOS, Windows. Các Hybrid Mobile App này được viết bằng ngôn ngữ lập trình web (như HTML5, Javascript hay CSS3) và sau đó được "bao bọc" bằng một lớp vỏ (container) bên ngoài để trở thành giống như Native Mobile App. Trên thực tế, nó hiển thị các trang web từ một trang web máy tính phù hợp với màn hình WebView, nội dung web có thể được hiển thị ngay khi ứng dụng được mở hoặc chỉ dành cho một số phần nhất định của ứng dụng.

1.2 Tổng kết

Mỗi loại ứng dụng mang lại một trải nghiệm khác nhau và tùy theo những trường hợp cụ thể để một lập trình viên đánh giá và xem xét các ưu và nhược điểm của từng loại ứng dụng từ đó đưa ra phương án xây dựng cụ thể để tối ưu hóa về chi phí, nhân lực, cũng như mang lại những giá trị về mặt hiệu suất, trải nghiệm người dùng.

Tuy nhiên ta phải khẳng định tầm quan trọng của Mobile App đối với các doanh nghiệp nói riêng và cuộc sống của chúng ta nói chung. Những ứng dụng này mang lại những lợi ích nhất định làm tăng hiệu quả của việc kinh doanh và những hoạt động trong cuộc sống thường ngày.

2.2 Microcontrollers ESP32 series

1.1 Tổng quan

ESP32 là một series các vi điều khiển trên một vi mạch giá rẻ, năng lượng thấp có hỗ trợ WiFi và dual-mode Bluetooth (Bluetooth chế độ kép) ở cả hai biến thể lõi

kép và lõi đơn, và bao gồm các công tắc antenna tích hợp, RF balun, bộ khuếch đại công suất, bộ khuếch đại thu nhiễu thấp, bộ lọc và module quản lý năng lượng.

ESP32 được chế tạo và phát triển bởi Espressif Systems, một công ty Trung Quốc có trụ sở tại Thượng Hải, và được sản xuất bởi TSMC bằng cách sử dụng công nghệ 40 nm. ESP32 là sản phẩm kế thừa từ vi điều khiển ESP8266.

Có thể nói ESP32 là sự nâng cấp hoàn hảo của ESP8266, với ESP8266 phù hợp với các dự án nhỏ và tiết kiệm chi phí. ESP32 lại phù hợp với các dự án phức tạp hơn, tốc độ xử lý cao hơn và tích hợp nhiều ngoại vi mạnh mẽ hơn.

Về bộ nhớ ESP32 có thêm 4MB External Flash và 520KB SRAM (static random access memory) trong đó 8 KB RAM RTC tốc độ cao – 8 KB RAM RTC tốc độ thấp (dùng ở chế độ DeepSleep).

ESP32 hỗ trợ Bluetooth 4.2 và BLE (Bluetooth Low Energy). Việc hỗ trợ cả bluetooth khiến ESP32 có thể tương tác với các thiết bị như là bàn phím, chuột, điện thoại khi mà không có Wifi.

Ultra Low Power giải quyết vấn đề năng lượng cho ESP bởi vì sử dụng Wi-Fi sẽ rất ngốn điện đặc biệt khi chúng ta sử dụng pin phải tính toán rất kĩ.
Ngoài ra ESP32 đang được rất nhiều các công ty trong và ngoài nước ưa chuộng

3. YÊU CẦU HỆ THỐNG

- Từ mục đích sử dụng và lý thuyết nêu trên ta có thể đưa ra một số yêu cầu về mô hình như sau:
    - Có khả năng thu thập dữ liệu (nhiệt độ và độ ẩm) từ cảm biến.
    - Hiển thị và vẽ biểu đồ của 2 thông số trên theo thời gian.
    - Có chế độ điều khiển bằng tay (manual) và chế độ tự động (auto).
    - Cảnh báo người dùng khi một thông số (nhiệt độ hoặc độ ẩm) trên hoặc dưới mức cho phép.

- Sử dụng Hybrid App và mô hình nhỏ gọn.

4. ĐẶC TẢ HỆ THỐNG

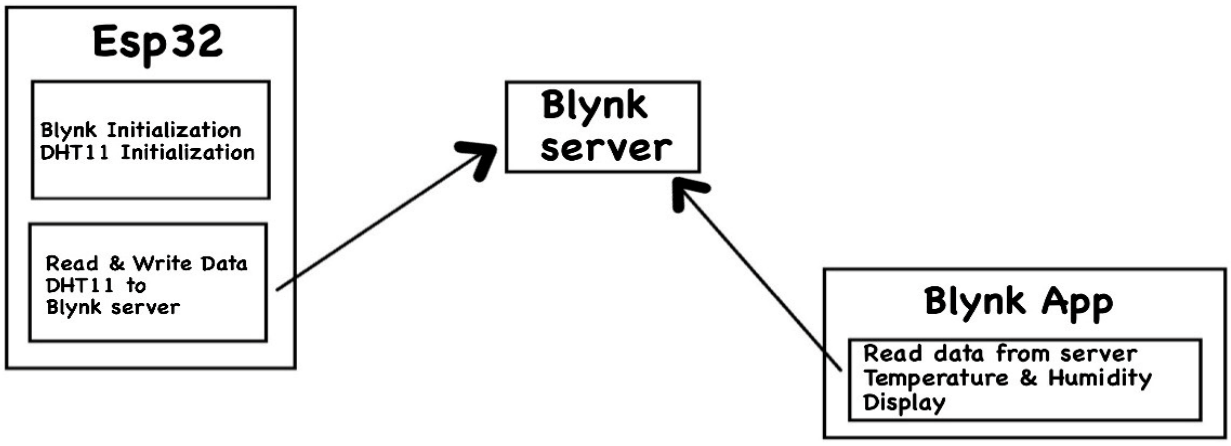
Từ yêu cầu và nhiệm vụ trên ta chia công việc thành 2 phần chính:

4.1 Thiết kế phần mềm

- Do ta sử dụng App Blynk đã được cung cấp sẵn, nên ta thực hiện việc kết nối đến module ESP32 và thiết kế giao diện thân thiện dễ sử dụng cho trên App cho người dùng.
- Thực hiện lập trình cho module ESP32 kết nối được với ứng dụng Blynk, thu thập và truyền dữ liệu từ cảm biến, cuối cùng là thực hiện việc điều khiển led, motor, máy bơm.

4.2 Thiết kế phần cứng

- Ta thực hiện mạch nguồn đơn giản cho module ESP32.
- Thiết kế vỏ mica cho mô hình.
- Cách thức hoạt động của mô hình:

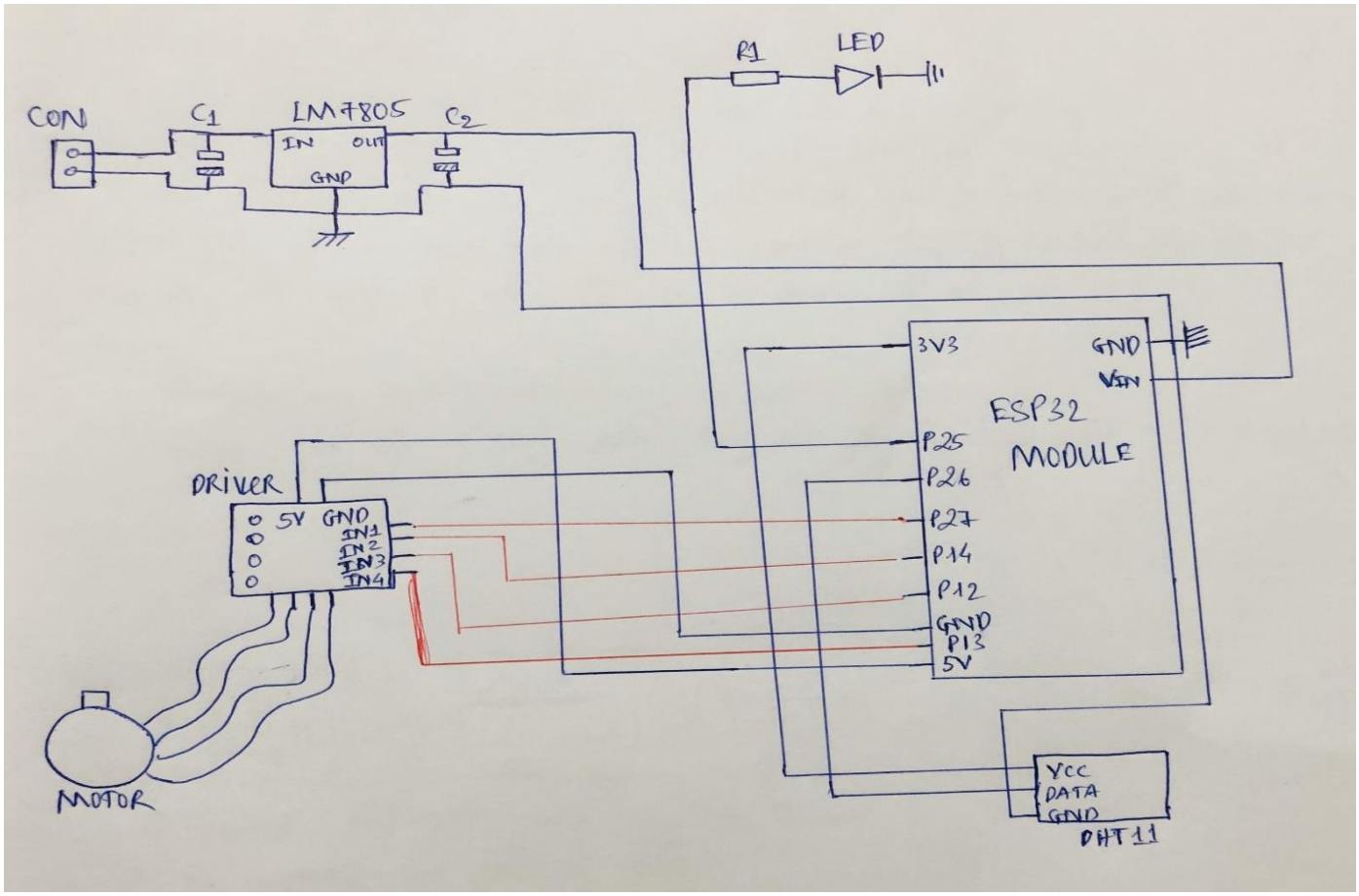


5. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

a) Mạch cấp nguồn

- Ta sử dụng các linh kiện thành phần sau:
  - Pin 9V và hộp đựng pin
  - Module ESP32
  - Cảm biến DHT11 , Step Motor và Driver
  - IC LM7805
  - Connector
  - Dây nối
  - Dụng cụ làm mạch (mỏ hàn, phíp đồng, khoan mạch,…)

b) Sơ đồ nguyên lý



6. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM

a) Lập trình cho module ESP32

- Ta thêm các thư viện cần thiết như: WiFi (có sẵn), BlynkSimpleEsp32, thư viện DHT và thư viện Stepper của động cơ bước

```
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include "DHT.h"
#include <Stepper.h>
```

- Khi ta thực hiện kết nối từ Blynk tới module ta sẽ nhận được một token ứng với mỗi project, ta thực hiện khai báo token và WiFi cho module

```
char auth[] = "t6Ybrt5il-Ib93nccl8-hmt0WKBJCmYT";
char ssid[] = "Tien Dat 3";
char pass[] = "06082000";
```

- Sau đó ta khai báo các biến sử dụng trong phần code của chúng ta

```
int led = 25;
int ac = 32;
int ac1 = 33;
int val = 0;
int cungchieu = 0;
#define DHTTYPE DHT11
#define DHTPIN 26
DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;
```

Trong đó:

- led: đèn led báo hiệu
- ac: bật tắt chế độ auto
- ac1: bật tắt motor
- val: biến đọc giá trị ac từ app
- cungchieu: biến đọc giá trị ac1 từ app
- chân đọc tín hiệu của cảm biến là chân 26

- biến timer để quy đổi giá trị thời gian thực

- Thực hiện viết function gửi dữ liệu cảm biến (sendSensor)

```
void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println("không đọc được dữ liệu từ DHT11");
    return;
  }

  Blynk.virtualWrite(V5, t);
  Blynk.virtualWrite(V6, h);
}
```

Trong đó:

- h là biến đọc dữ liệu độ ẩm
- t là biến đọc dữ liệu nhiệt độ

Khi không được được dữ liệu sẽ in ra dòng "không đọc được dữ liệu từ DHT11" trên serial monitor của IDE. Ta sẽ gán t cho chân V5 và h cho chân V6 trong app Blynk

- Trong phần setup() ta gán cho led là tín hiệu OUTPUT và ac, ac1 là tín hiệu INPUT nhận từ App Blynk. Bắt đầu cho DHT11 thu thập sau mỗi giây(1000L) và kết nối Blynk với WiFi và module sau đó đặt tốc độ cho động cơ là 10

```
void setup()
{

  pinMode(led,OUTPUT);
  pinMode(ac1,INPUT);
  pinMode(ac ,INPUT);

  Serial.begin(9600);
  dht.begin();
  timer.setInterval(1000L, sendSensor);
  Blynk.begin(auth, ssid, pass);

  myStepper.setSpeed(10);

}
```

- Cuối cùng trong loop() ta thực hiện như sau:

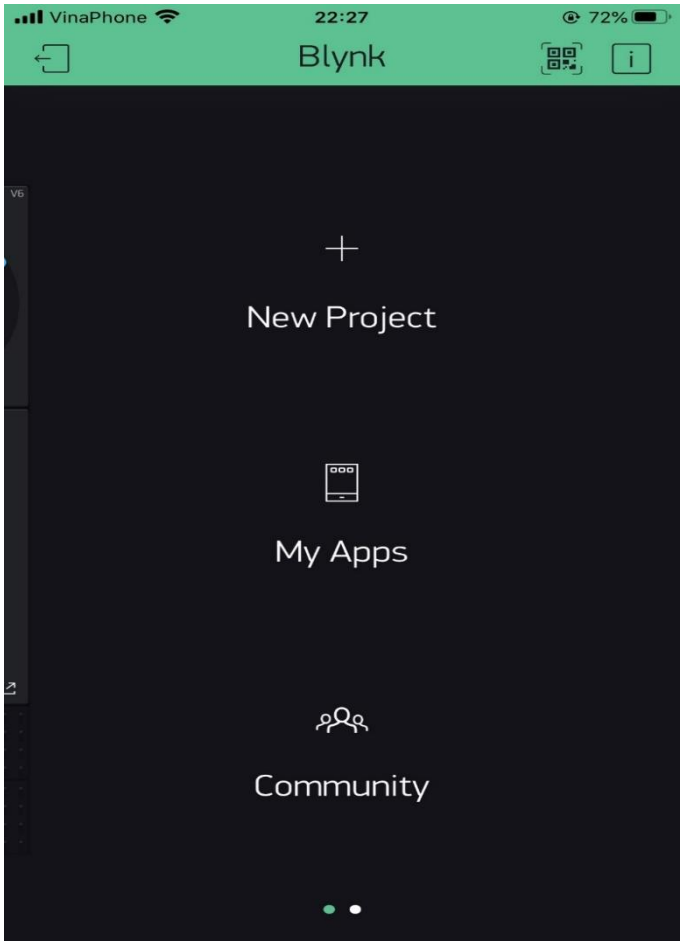
```
void loop(){
  Blynk.run();
  timer.run();
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  val = digitalRead(ac);
  cungchieu = digitalRead(ac1);
  Serial.print("val = ");
  Serial.println(val);
  Serial.print("cungchieu = ");
  Serial.println(cungchieu);
  Serial.print(F("Độ ẩm là: "));
  Serial.print(h);
  Serial.print(F("% || Nhiệt độ là: "));
  Serial.print(t);
  Serial.println(F("^C"));

  if (cungchieu == 1 && val == 0)
  {
    Serial.println("CUNG CHIEU KIM DONG HO");
    myStepper.step(-stepsPerRevolution);
  }

  if (val == 1){
    if (t >= 30 && t <= 35 )
    {
      digitalWrite(led,HIGH);
      Serial.println("NGUOC CHIEU KIM DONG HO");
      myStepper.step(stepsPerRevolution);}
    else
    {
      digitalWrite(led,LOW);
    }}
```

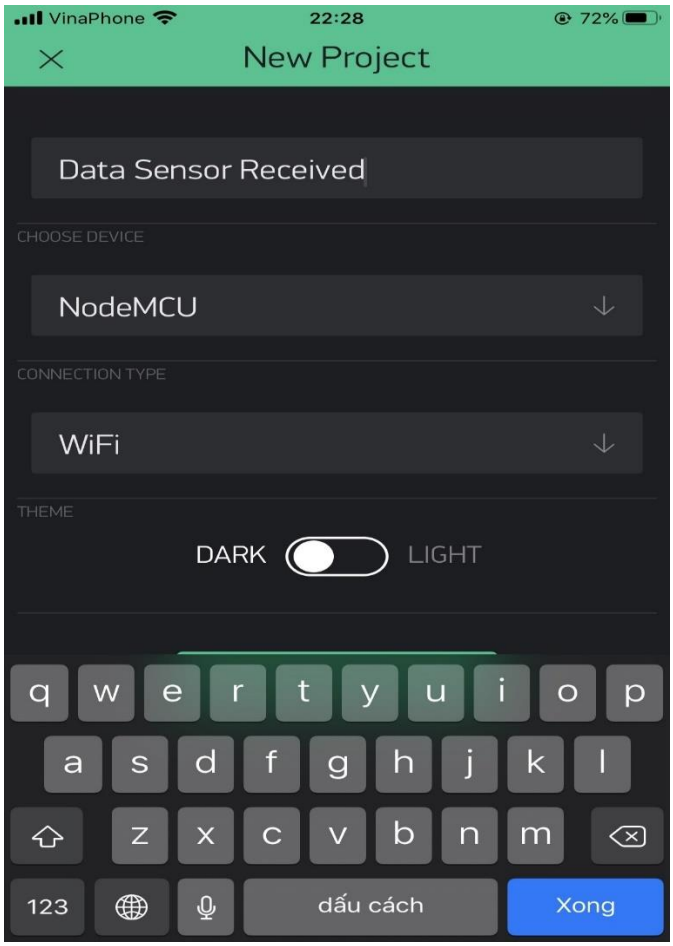
- Chạy Blynk và timer để ghi thời gian.
- Ta cho val và cungchieu đọc tín hiệu từ ac và ac1 (nút nhấn từ Blynk)
- Sau đó đọc giá trị của độ ẩm và nhiệt độ hiển thị lên cả Blynk và Serial monitor
- Chế độ bật tay (manual) nếu ac1 đang ON (cungchieu = 1) và ac đang OFF (val = 0) thì động cơ sẽ chạy cùng chiều kim đồng hồ
- Chế độ tự động (auto) nếu ac đang ON (val = 1) lúc này ta đặt điều kiện để thực hiện điều khiển ở đây ta chọn nhiệt độ và điều kiện là (nhiệt độ t >= 30

và t <=35) thì led sẽ bật lên cảnh báo và động cơ sẽ chạy ngược chiều kim đồng hồ

b) Thiết kế giao diện App Blynk

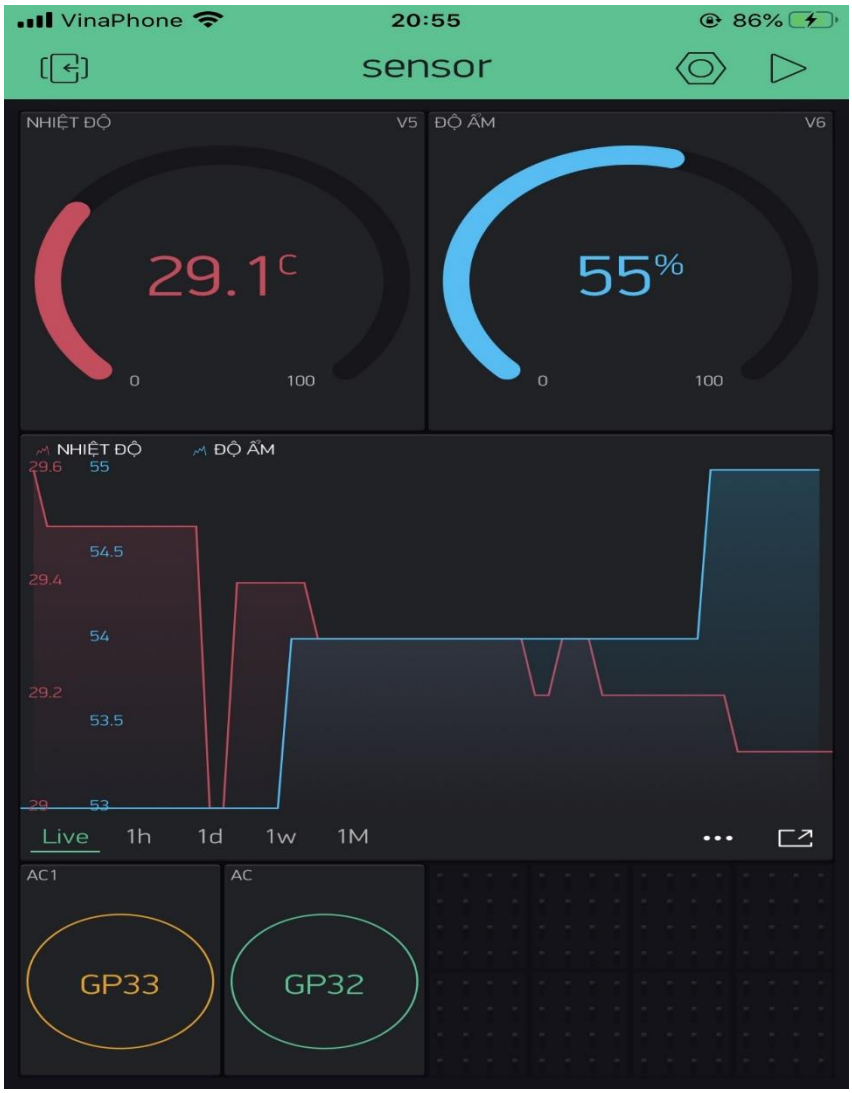
- Sau khi tải Blynk App từ nền tảng Google Play Store (Android) hoặc App Store (iOS) ta bắt đầu tạo dự án mới (New Project)





- Sau đó đặt tên cho Project và chọn NodeMCU (CHOOSE DEVICE) chọn kết nối WiFi (CONNECTION TYPE)

- Trong Widget Box ta lấy các mục Button, Gauge và SuperChart như hình

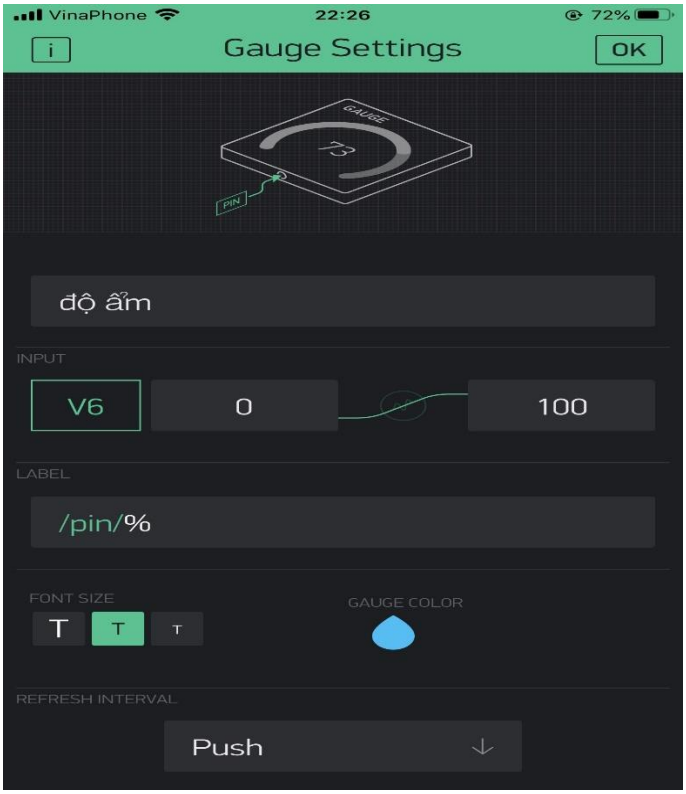


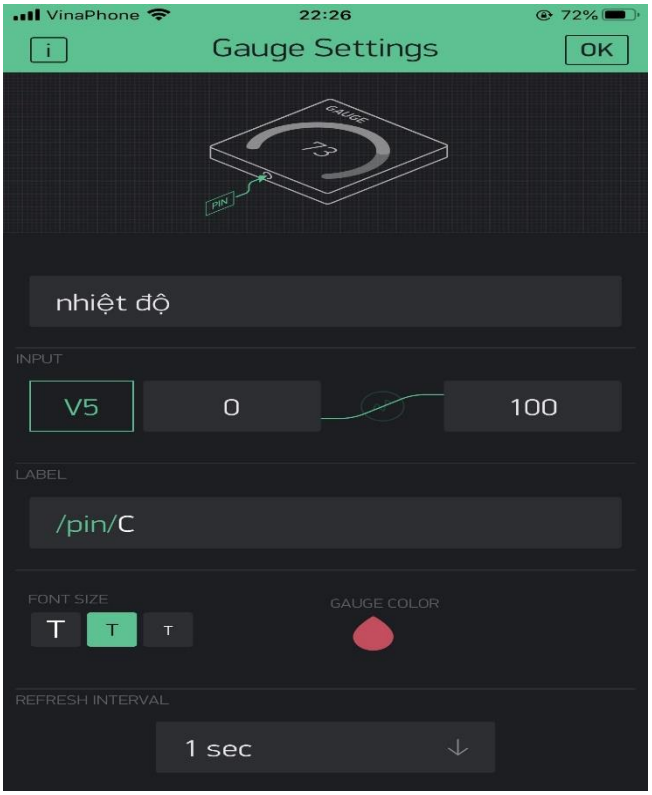
Gauge là các thang đo ứng với màu đỏ là thang đo cho nhiệt độ (C) và màu xanh là thang đo cho độ ẩm (%)

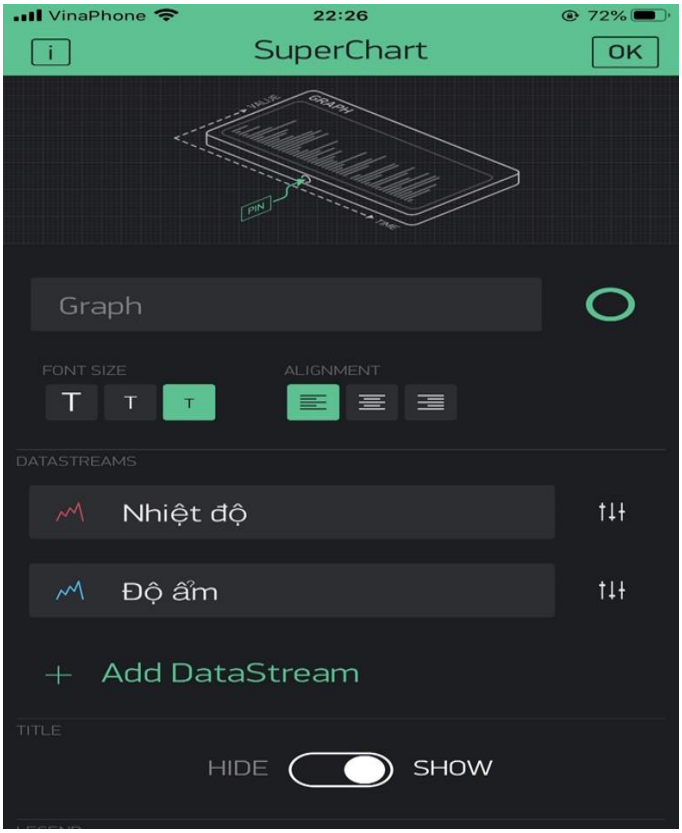
SuperChart ta sẽ vẽ đồ thị của nhiệt độ và độ ẩm theo thời gian (với các mốc 1h, 1d, 1w, 1M) và live.

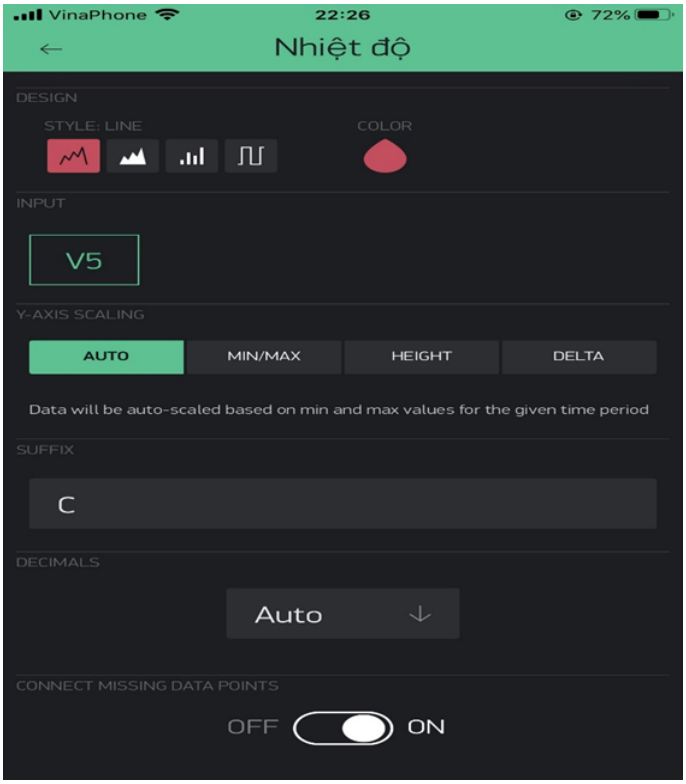
2 Button ứng với GP33 là bật tắt motor bằng tay (manual) và GP32 là bật tắt chế độ auto (ứng với mức 0-1 của biến ac).

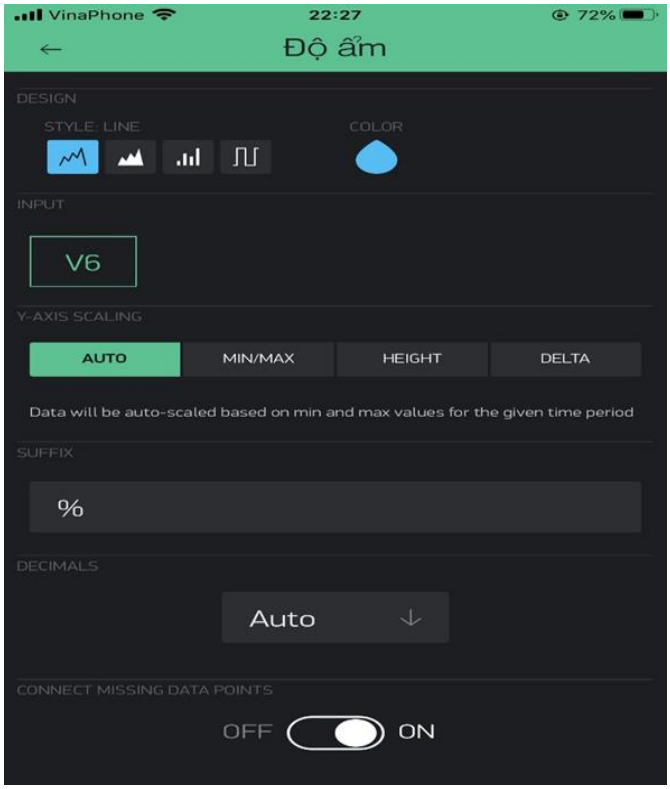
- Sau đó ta cấu hình như sau để phù hợp với phần lập trình cho module

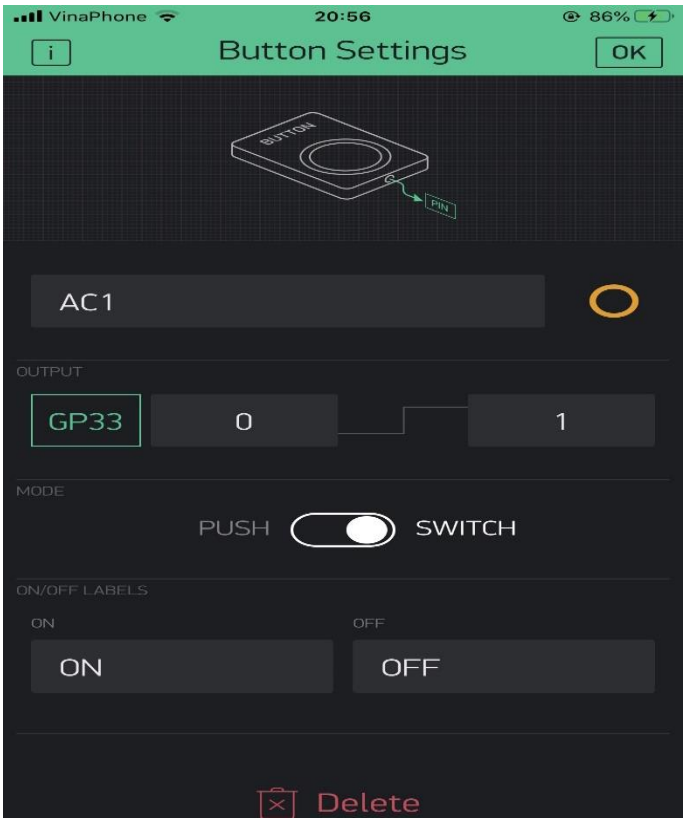


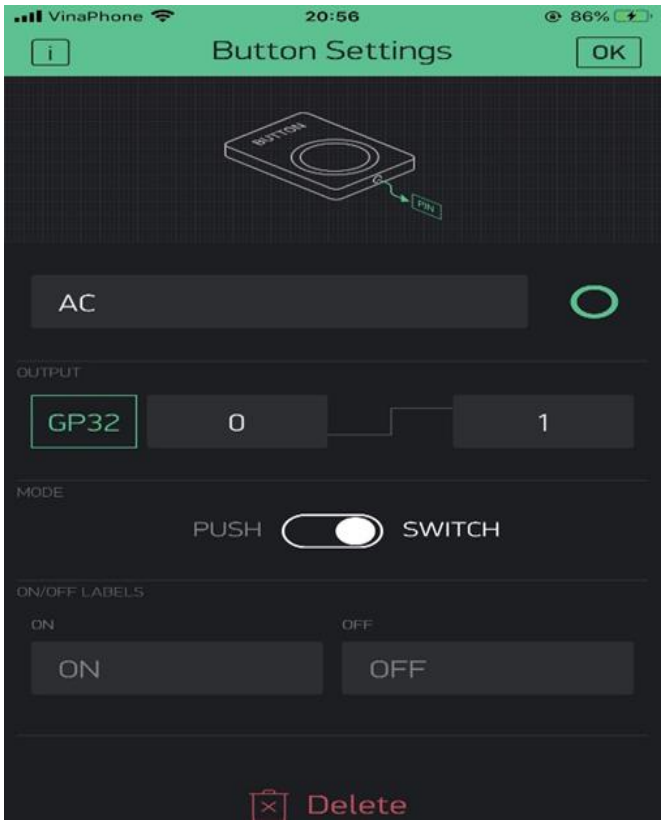












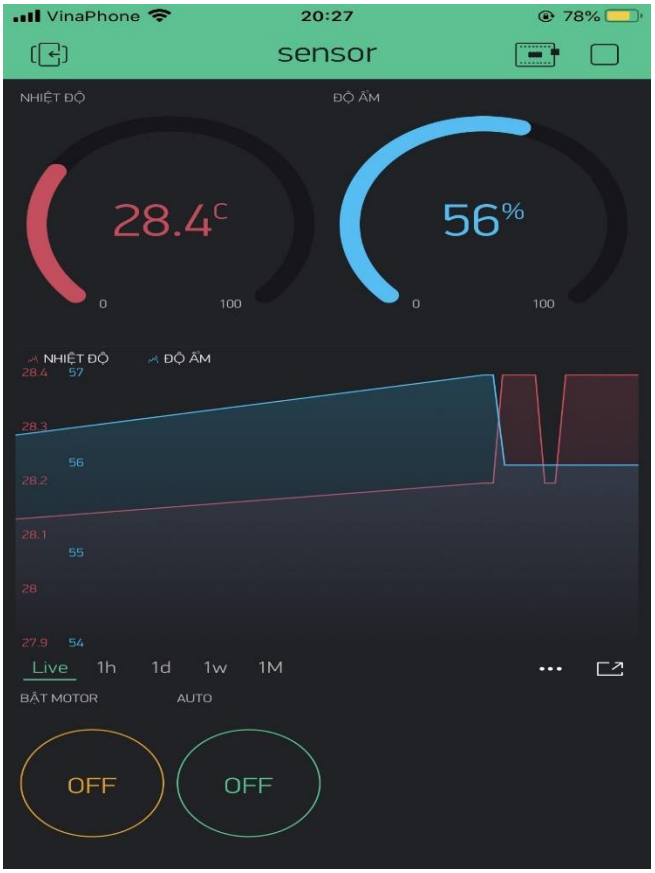
- Sau khi thiết lập xong ta thực hiện kết nối tới module.

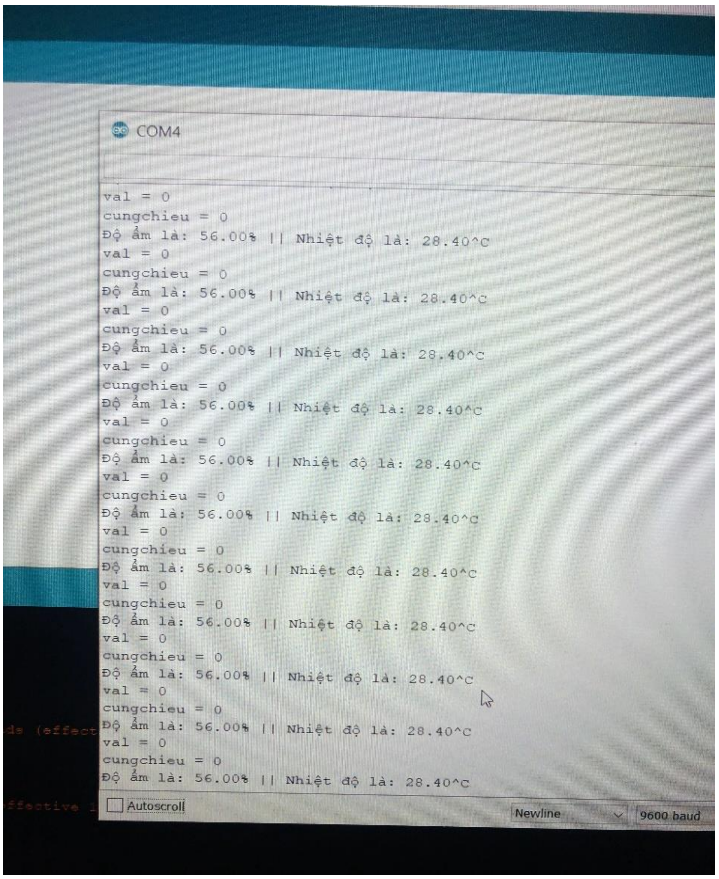
7. KẾT QUẢ THỰC HIỆN

```
[3495] Connected to WiFi
[3495] IP: 192.168.30.108
[3495]

    ___  __          __
   / _ )/ /_ _____  / /__
  / _  / / // / _ \/  '_/
 /____/_/\_, /_//_/_/\_\
        /___/ v1.0.0 on ESP32

[3571] Connecting to blynk-cloud.com:80
[8052] Ready (ping: 3278ms).
```

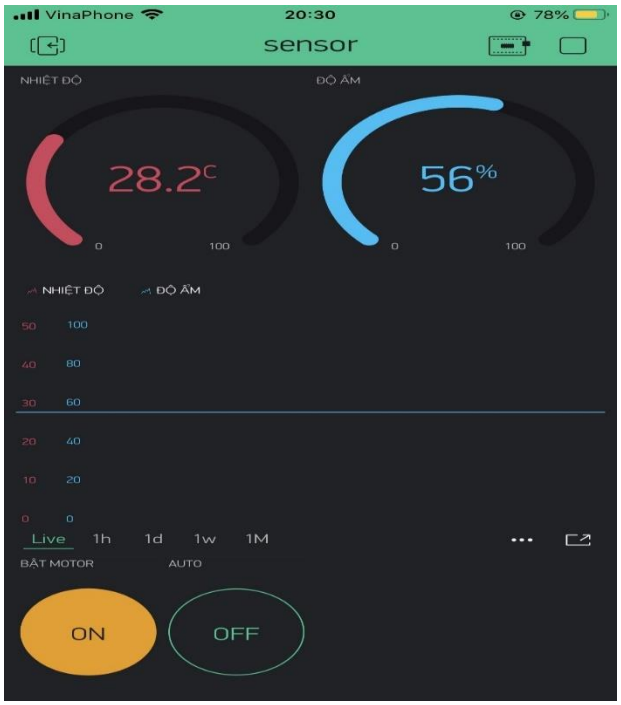
Kết nối thành công tới WiFi và Server của Blynk

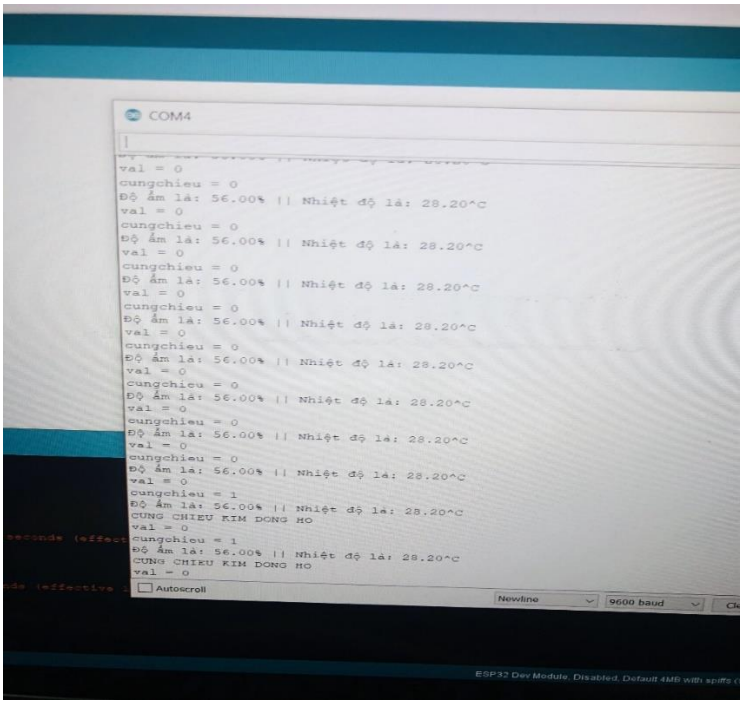
- khi val và cungchieu ở mức 0





- Chế độ manual cungchieu = 1 và val = 0 lúc này động cơ quay cùng chiều kim đồng hồ.





- chế độ auto val = 1 lúc này ta sẽ xét đến điều kiện ta đã đặt khi nhiệt độ t >= 30 và <=35 thì led sẽ bật và động cơ quay ngược chiều.