



**BỘ MÔN KỸ THUẬT MÁY TÍNH – VIỆN THÔNG  
CƠ SỞ VÀ ỨNG DỤNG IOTS**

**MMH: ITFA436064**

**Thời gian thực hiện: 2 buổi**

**Trần Nguyễn Quang Lâm - 20139040**

**Tô Gia Huy - 20139003**

**Trần Đức Hiếu - 20139074**

**Phạm Thanh Hà - 20139073**

**Nguyễn Trí Ban - 20139062**

**1. So sánh server Thingspeak và Google Firebase**

	<b>Thingspeak</b>	<b>Google Firebase</b>
<b>Tính chất</b>	<ul style="list-style-type: none"><li>• Một platform với mã nguồn dữ liệu mở của kết nối vạn vật [1]</li><li>• Thường được sử dụng cho các sản phẩm thử nghiệm hoặc các mô hình hệ thống IoT cần phải phân tích dữ liệu [1]</li></ul>	<ul style="list-style-type: none"><li>• Một nền tảng di động giúp người dùng nhanh chóng phát triển các ứng dụng chất lượng cao với quy mô lớn [2]</li><li>• Firebase là một dịch vụ kết nối Backend được liên kết với Google, hỗ trợ cả 2 nền tảng iOS và Android [2]</li><li>• Firebase không được thiết kế cho nhu cầu của các nhà phát triển IoT [1]</li></ul>
<b>Tính năng, ưu điểm</b>	<ul style="list-style-type: none"><li>• Cho phép truy cập và tải dữ liệu, đăng nhập vào hệ thống với API của phần cứng và các mạng xã hội [3]</li><li>• Biểu thị dữ liệu của cảm biến ở thời gian thực [3]</li><li>• Phát triển thiết bị IoT mà không cần làm server hoặc phần mềm [1]</li><li>• Có hỗ trợ MATLAB, cho phép phân tích, trực quan hóa dữ liệu online với biểu đồ spline [3]</li><li>• Trang web và ứng dụng server hỗ trợ Python, Ruby và Node.js, đây là các ngôn ngữ mạnh và phổ biến nên sẽ được hỗ trợ rất nhiều tính năng [3]</li><li>• Cho phép bên thứ ba thu thập dữ liệu theo nhu cầu của người dùng (dữ liệu từ các cảm biến) [3]</li><li>• Có thể lên lịch các tính toán cho kênh giao tiếp [1]</li><li>• Hành động dựa trên dữ liệu thu thập được (vd: tắt bật thiết bị nếu dữ liệu vượt ngưỡng...) [1]</li></ul>	<ul style="list-style-type: none"><li>• Firebase có thể làm việc với bất kỳ phần cứng nào với giao tiếp chuẩn REST API [1]</li><li>• Thiếu hỗ trợ cho cập nhật firmware OTA, xử lý dữ liệu online [1]</li><li>• Phân tích với Google, do Firebase liên kết trực tiếp với Google Ads,... nên quản lý dữ liệu rất dễ dàng và chính xác [1]</li><li>• Cung cấp bảng điều khiển để quan sát dữ liệu [2]</li><li>• Firebase cung cấp dữ liệu thời gian thực chính xác [2]</li><li>• Hỗ trợ các phương thức offline, các thao tác với dữ liệu Firebase offline sẽ được đồng bộ khi online [2]</li><li>• Firebase cung cấp sự bảo mật (Authentication), khách hàng có thể truy cập dùng tài khoản các platform như Google, Twitter, Facebook, Github [2]</li></ul>
<b>Nhược điểm</b>	<ul style="list-style-type: none"><li>• Giới hạn lượng dữ liệu tải lên qua API mỗi 15 giây [3]</li><li>• Không đa dạng các loại biểu đồ để trực quan hóa dữ liệu (không có pie chart, bar chart và histogram) [3]</li></ul>	<ul style="list-style-type: none"><li>• Database được tổ chức theo kiểu cây (trees) với cấu trúc parent-children khó làm quen với người dùng SQL [2]</li></ul>

## 2. So sánh Amazon AWS IoT và Microsoft Azure IoT hub

	<b>Amazon AWS IoT</b>	<b>Microsoft Azure IoT hub</b>
Là gì?	Amazon Web Services (AWS) là nền tảng đám mây toàn diện và được sử dụng rộng rãi nhất, cung cấp trên 200 dịch vụ đầy đủ tính năng từ các trung tâm dữ liệu trên toàn thế giới	Azure IoT Hub là dịch vụ Internet of Things (IoT) được quản lý được lưu trữ trên đám mây. Được cung cấp bởi đám mây của Microsoft, dịch vụ này cho phép một cuộc trò chuyện hai phía giữa các ứng dụng IoT và các thiết bị mà chúng đã kết nối.
Các thể loại cung cấp	<ul style="list-style-type: none"> <li>• Compute, lưu trữ (Storage)</li> <li>• Phân phối nội dung (Content Delivery)</li> <li>• Cơ sở dữ liệu (Database)</li> <li>• Mạng (Networking)</li> </ul>	<ul style="list-style-type: none"> <li>• Compute , Quản lý dữ liệu (bao gồm cơ sở dữ liệu) (data manage)</li> <li>• Hiệu suất (Performance)</li> <li>• Mạng (Networking)</li> </ul>
Tính năng	Theo dõi việc sử dụng tài nguyên cơ sở hạ tầng thông qua các công cụ quản lý như Amazon CloudWatch, AWS Cloudtrail để theo dõi hoạt động của người dùng và sử dụng API và AWS Config để theo dõi việc kiểm kê và thay đổi tài nguyên.	Azure cũng có một số dịch vụ và tích hợp để giám sát và cảnh báo sâu về các chỉ số hiệu năng và logs.
Ưu điểm	<ul style="list-style-type: none"> <li>• Người tiên phong trong điện toán đám mây</li> <li>• Amazon hiện cung cấp một số giải pháp lai như Storage Gateway, DynamoDB Local, và OpsWorks, nhưng cho tới bây giờ, Microsoft vẫn có lợi thế lớn.</li> <li>• AWS sử dụng linux ngay từ đầu nên hỗ trợ open source tốt hơn</li> </ul>	<ul style="list-style-type: none"> <li>• Người đi đầu trong việc đưa hybrid cloud đến với khách hàng. =&gt; Azure cho phép tạo ra các ứng dụng lai khi vừa có thể sử dụng dữ liệu trong server riêng, đồng thời các ứng dụng đó cũng có thể kết hợp với sức mạnh điện toán của đám mây để đẩy mạnh khả năng tính toán, tiết kiệm thời gian hoàn thành công việc.</li> <li>• Azure đã hỗ trợ cho các phiên bản Ubuntu, CentOS, Oracle, SUSE Linux Enterprise, openSUSE, Red Hat Enterprise Linux</li> </ul>
Chi phí	Cả 2 đưa ra các mức giá tương đương nhau để cạnh tranh, nhưng vẫn có sự chênh lệch giữa các dịch vụ riêng biệt. Và có các công cụ riêng biệt.	
Nhược điểm	<ul style="list-style-type: none"> <li>• AWS khó tiếp cận hơn với người mới.</li> </ul>	<ul style="list-style-type: none"> <li>• Dễ tiếp cận do Azure tích hợp hệ sinh thái Microsoft, tích hợp hệ điều hành window,visual studio....</li> </ul>

### 3. Nguyên lý chuyển đổi và thông số của cảm biến MQ2 (MQ3).

Tên thông số	Giá trị	Đơn vị
Kí hiệu	MQ-3	
Chất phản ứng	Cồn (ethanol)	
Dải đo	0,04-0,4	mg/l
Điện áp làm việc	<24	V
Điện áp sấy	$5 \pm 0,2$	V(AC hoặc DC)
Tải đầu ra	Điều chỉnh được	$\Omega$
Điện trở sấy	$31 \pm 3$	$\Omega$
Công suất sấy	$\leq 900$	mW
Điện trở cảm biến	2÷20	K $\Omega$ tại nồng độ cồn 0,4 mg/l
Độ nhạy	$\geq 5$	Tỉ lệ điện trở cảm biến khi nồng độ cồn bằng 0 và 0,4 mg/l

#### Nguyên lí hoạt động:

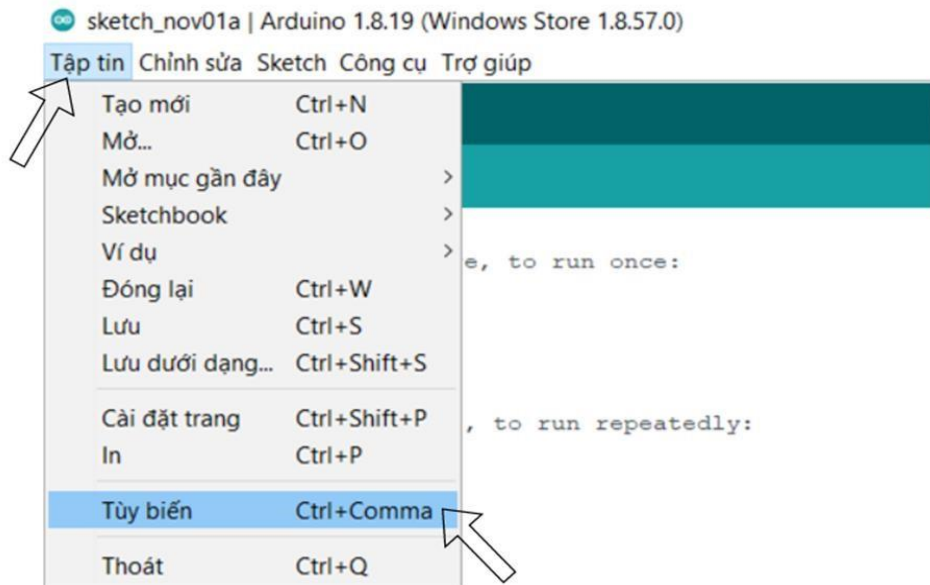
Khi một lớp bán dẫn SnO<sub>2</sub> được nung nóng đến nhiệt độ cao, oxy được hấp phụ trên bề mặt. Khi không khí sạch, các electron từ dải dẫn của thiếc điôxit bị các phân tử ôxy thu hút. Điều này tạo ra một lớp suy giảm điện tử ngay bên dưới bề mặt của các hạt SnO<sub>2</sub>, tạo thành một rào cản tiềm năng. Kết quả là màng SnO<sub>2</sub> trở nên có điện trở cao và ngăn dòng điện chạy qua.

Tuy nhiên, với sự có mặt của rượu, mật độ bề mặt của oxy bị hấp phụ giảm khi nó phản ứng với rượu, làm giảm hàng rào tiềm năng. Kết quả là, các electron được giải phóng vào thiếc điôxit, cho phép dòng điện chạy qua cảm biến một cách tự do.

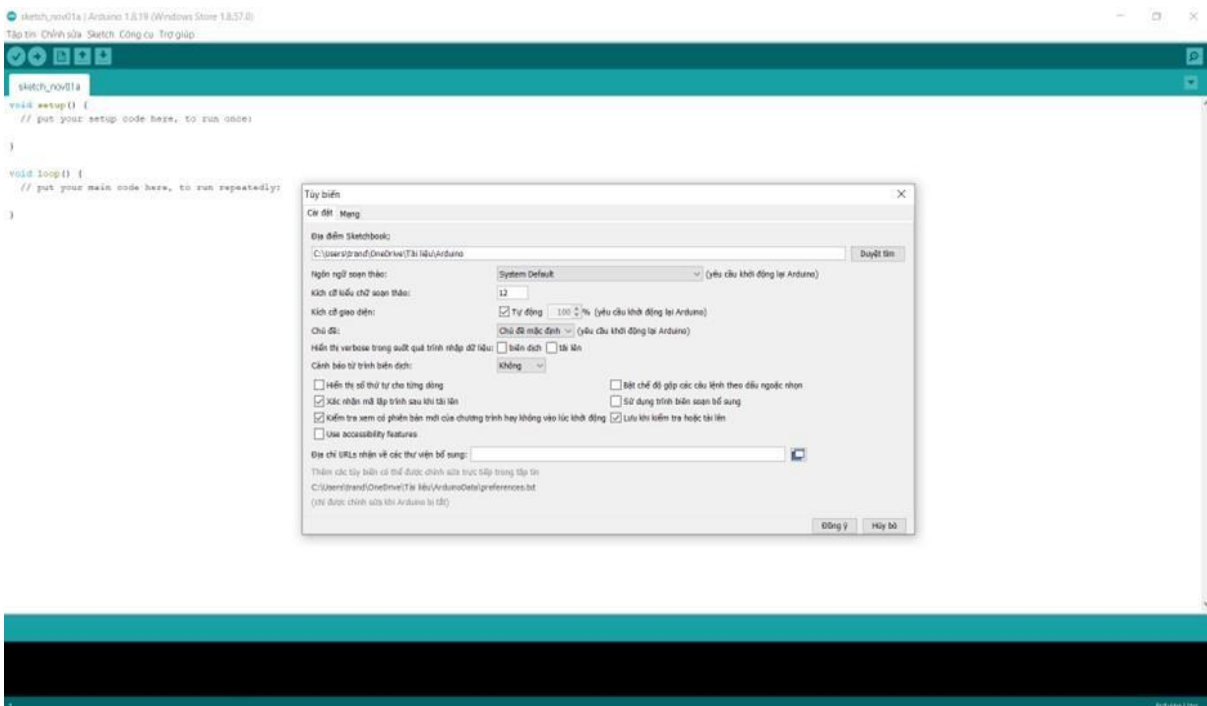
4. Các bước giao tiếp cảm biến MQ2 (MQ3) và ESP32 (kèm theo hình ảnh các bước thực hiện và sơ đồ kết nối).

a. Cài đặt thư viện

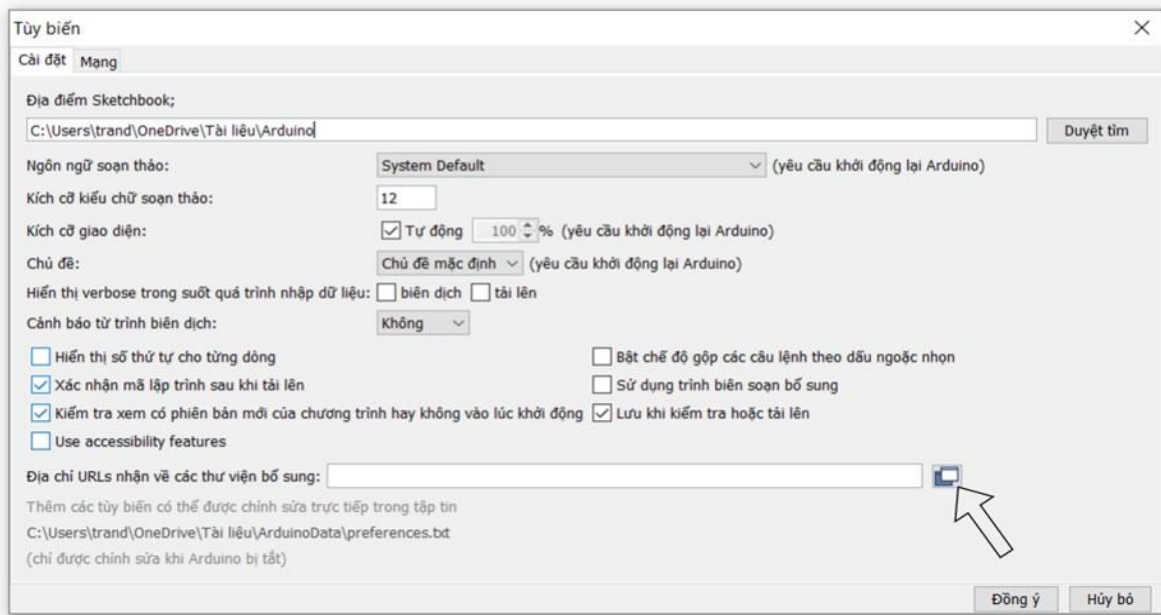
+ Để cài thư viện cho ESP32 các bạn vào File>Tùy biến hoặc bấm tổ hợp phím Ctr+Comma



+ Cửa sổ Tùy biến mới hiện ra



+ Tại cửa sổ tùy biến ta cần thay đổi địa chỉ URLs nhận về các thư viện bổ sung:

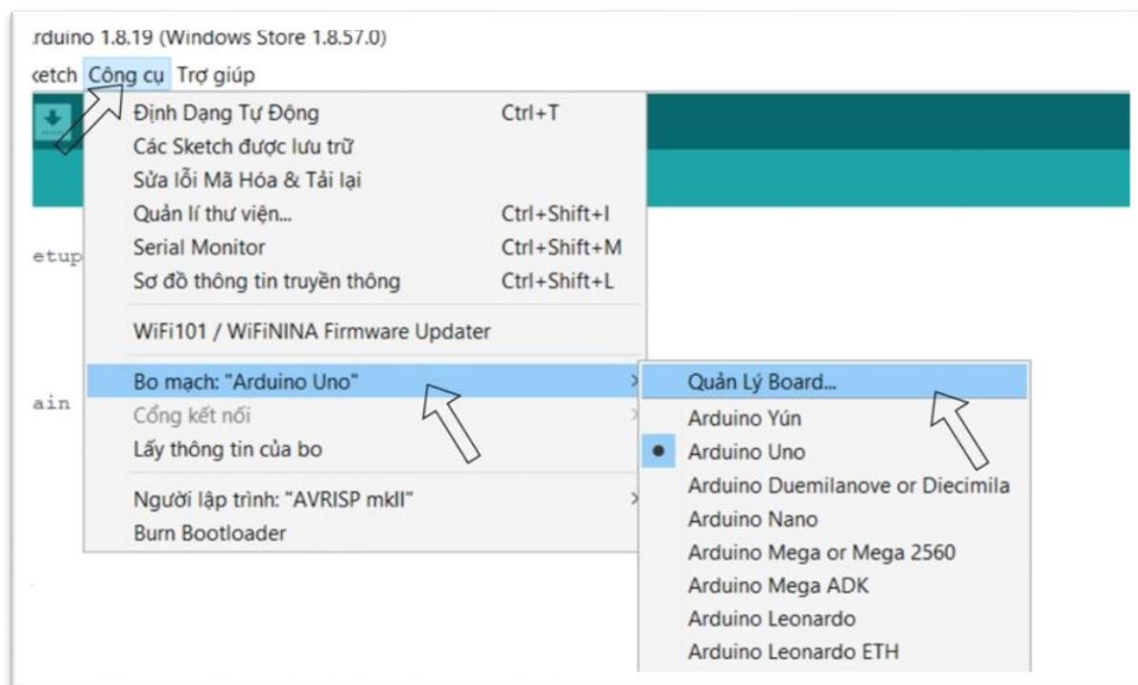


+ Dán đường dẫn sau đây để cài đặt thư viện cho ESP32:

[https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json)

+ Link thư viện đã nhúng vào IDE bây giờ sẽ tải và cài đặt thư viện ESP32:

Công cụ > Bo mạch: “Arduino Uno” > Quản lý Board

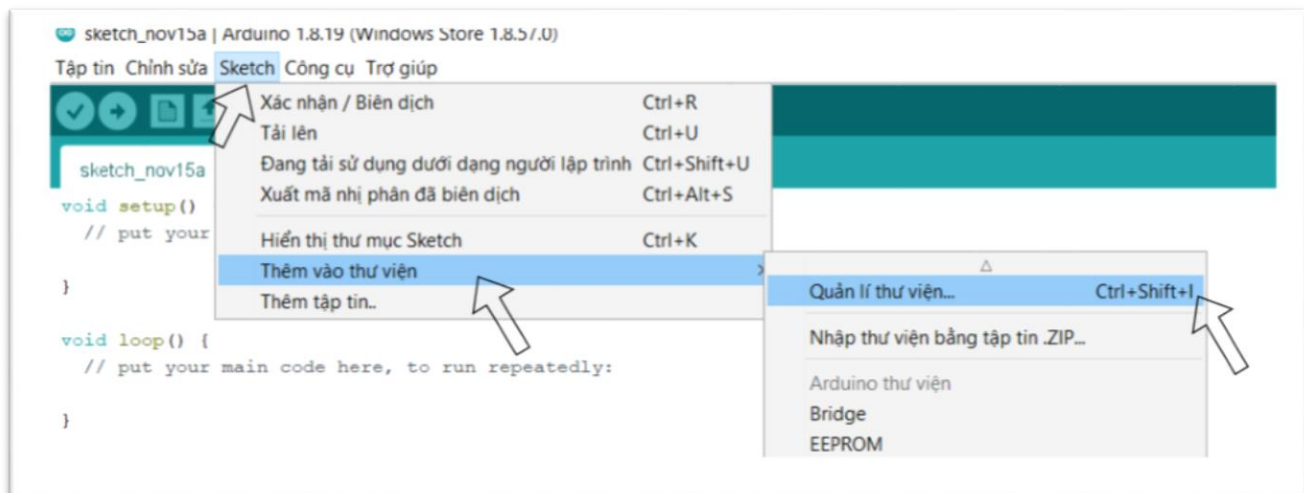


+ Sau đó ta tìm kiếm từ khóa esp và cài đặt esp32

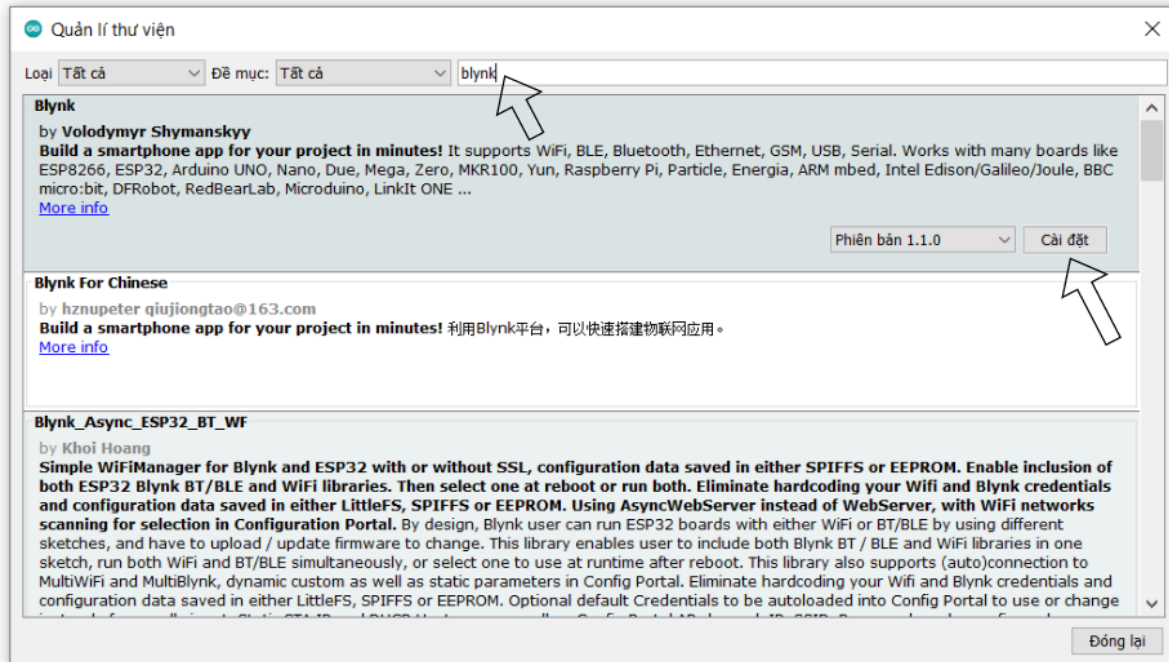


+ Đợi cho thư viện esp32 cài đặt thành công, cài tiếp thư viện Blynk:

Click Sketch trên thanh công cụ chọn Thêm vào thư viện sau đó chọn quản lý thư viện hoặc có thể bấm tổ hợp phím Ctrl+Shift+I để mở Quản lý thư viện.



## + Cài thư viện Blynk



Bây giờ chỉ cần đợi tải xong thư viện là thành công nhé!

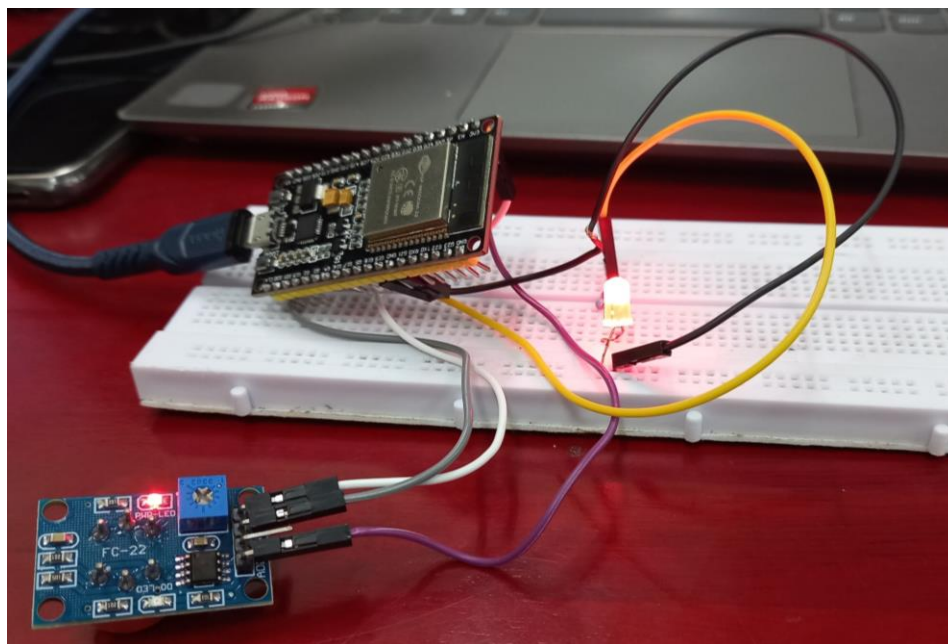
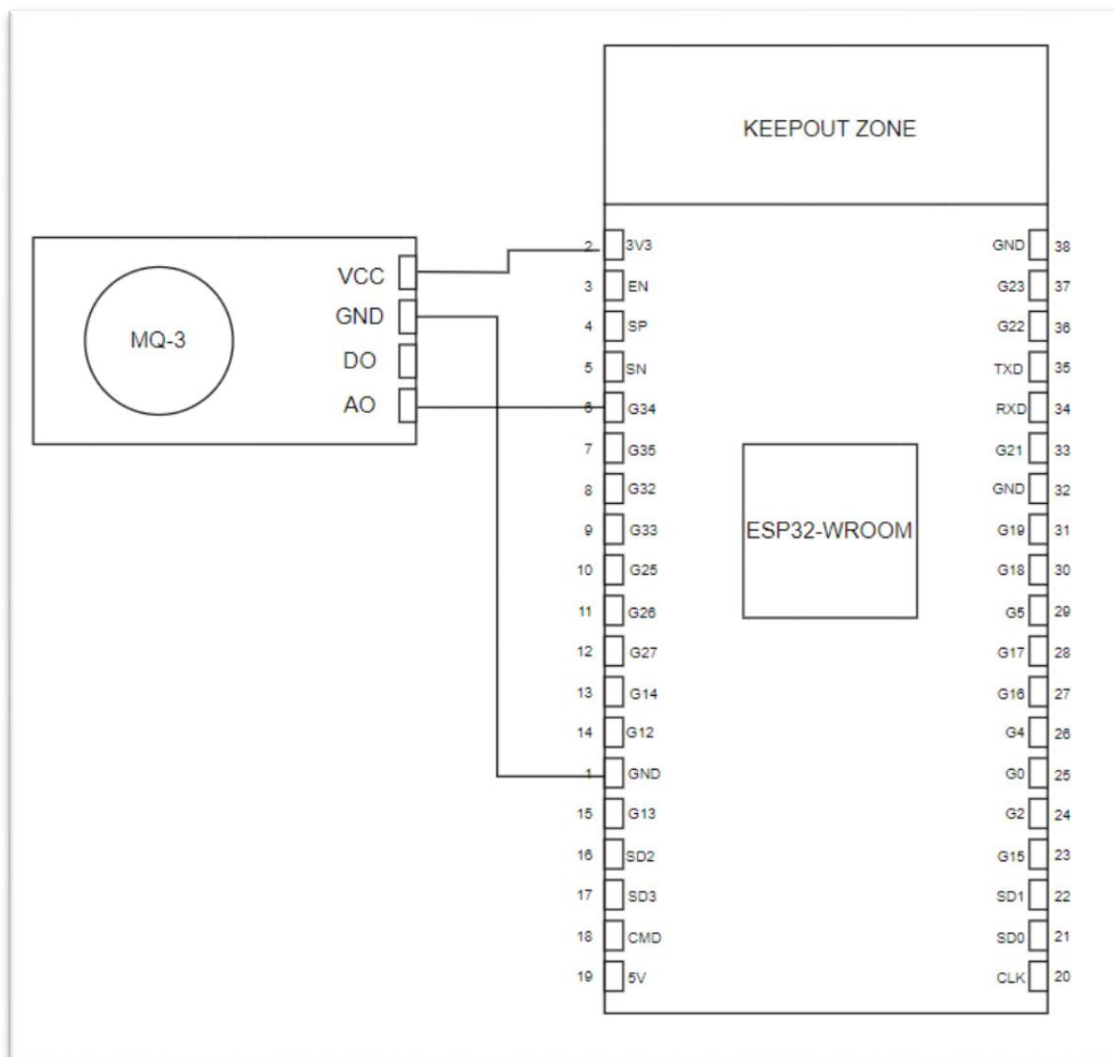
b. Ảnh code giao tiếp cảm biến MQ-3 và ESP32:

```
LAB3$

// Create Variables
float sensorValue;
#define MQ 18
#define SENSOR 34
void setup()
{
  Serial.begin(115200); // Khởi tạo kết nối Serial để truyền dữ liệu đến máy tính
  // Gõ địa chỉ IP đến máy tính
  pinMode(SENSOR, INPUT);
  Serial.println("sensor start");
  pinMode(MQ, OUTPUT);
}
void loop()
{
  sensorValue= analogRead(34);
  Serial.println("VALUE:");
  Serial.println(sensorValue);
  if( sensorValue>2000){
    digitalWrite(18,1);
    delay(50);
    digitalWrite(18,0);
    delay(30);
  }
}
```



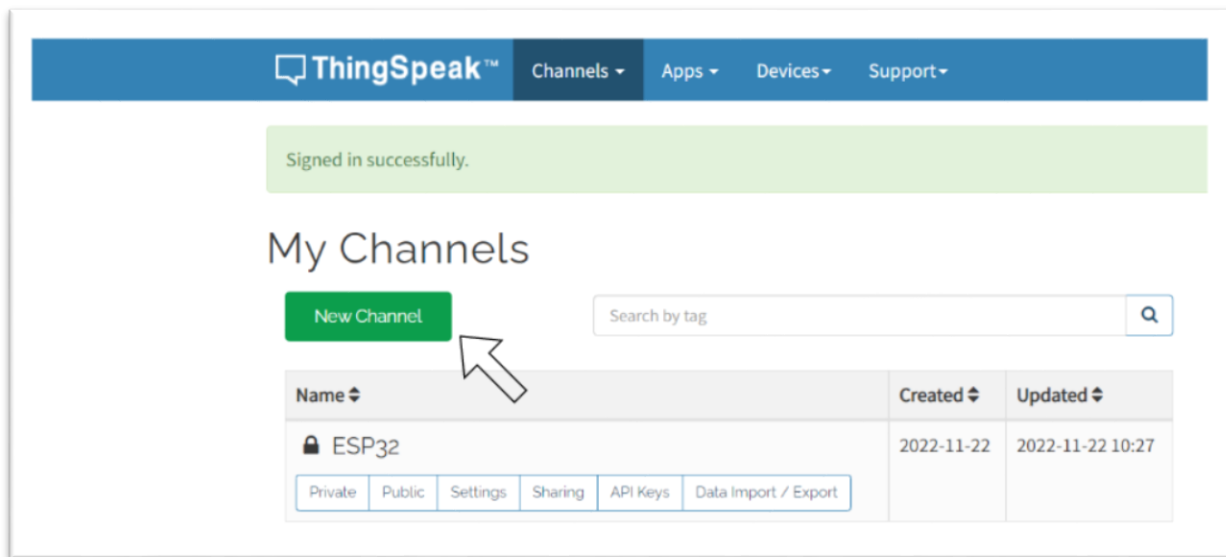
c. Sơ đồ kết nối chân của cảm biến MQ-3 và ESP32:





5. Các bước thực hiện, giải thích code quá trình cập nhật dữ liệu lên server Thingspeak và kết quả thực hiện (video clip demo nếu có).

+ Để vào được ThingSpeak cần đăng kí tài khoản, sau khi đăng kí bạn tiến hành đăng nhập để vào trang chính. Ở trang Channels chọn New Channel.



+ Đặt tên cho Dự án, ở đây mình chỉ thực hiện đo mỗi khí gas nên chỉ tạo 1 Field Label ( Nếu bạn tiến hành đo nhiều hơn có thể chọn thêm các Field 2, Field 3, ...)

New Channel

Name: ESP\_MQ3\_gas

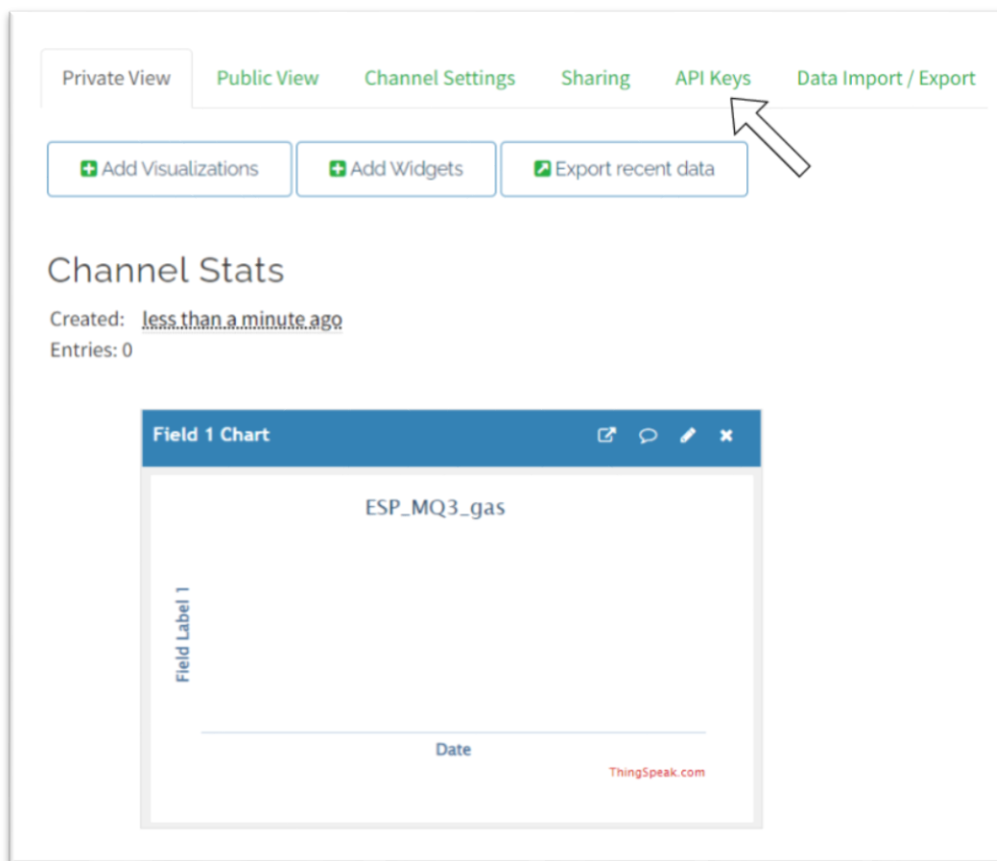
Description: Quan Trắc Khí gas

Field 1: Field Label 1 ☒

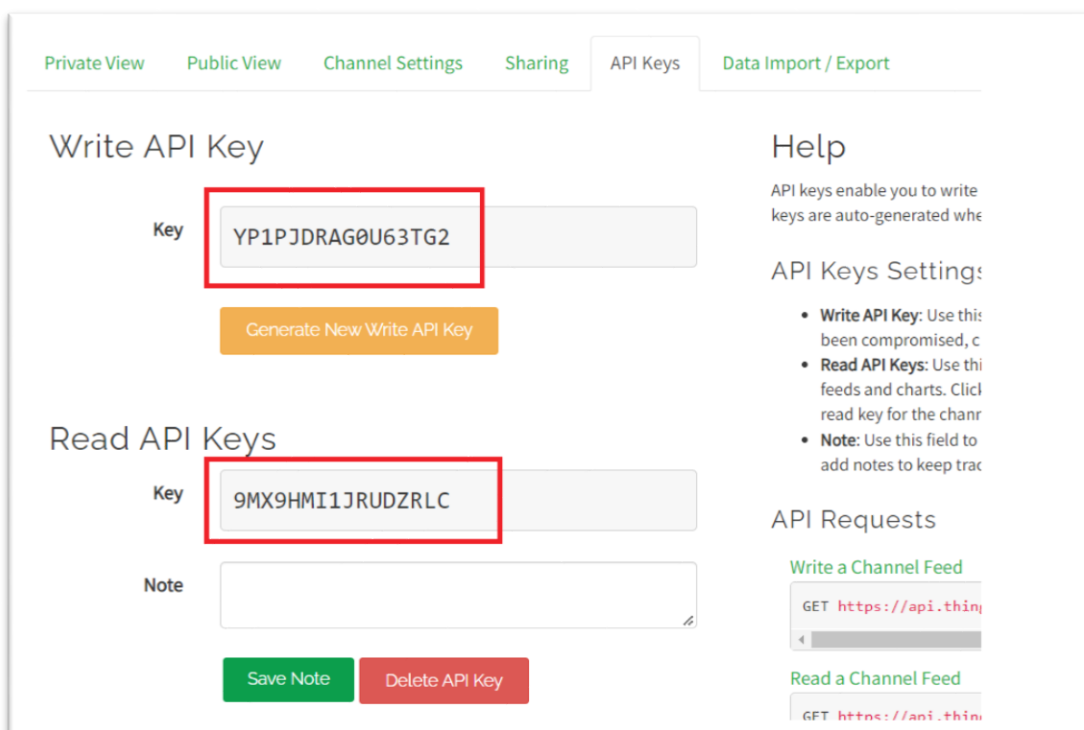
Field 2: ☐

Field 3: ☐

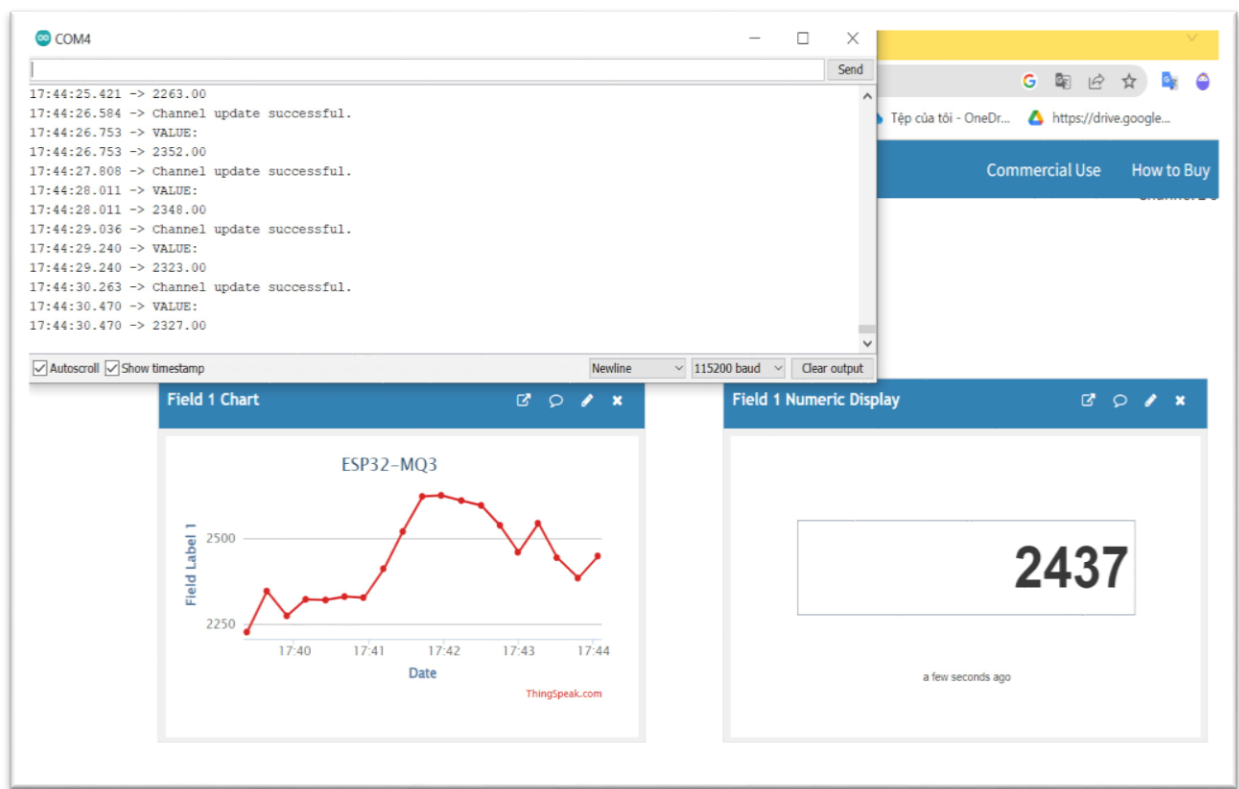
+ Khi Save Channel ThinkSpeak sẽ tạo 1 cửa sổ Chart để trực quan dữ liệu mình đo được nhờ cảm biến MQ3, tiếp đó mình sẽ vào API Keys để lấy API Key.



+ Lấy Write Key để đưa dữ liệu từ ESP lên web.



+) Đưa vào code và xem kết quả.



### Giải thích code

<pre>#include &lt;WiFi.h&gt; #include "ThingSpeak.h" #define SECRET_SSID "ARVR Lab" #define SECRET_PASS "ARVR@123456" #define SECRET_CH_ID 1949107 #define SECRET_WRITE_APIKEY "H6KRJSGZCGOBUU0A" char ssid[] = SECRET_SSID; char pass[] = SECRET_PASS;</pre>	<p>Thiết lập cho ESP kết nối với Channel đã tạo trên ThingSpeak</p> <ul style="list-style-type: none"> <li>+ SSID: tên wifi sử dụng</li> <li>+ PASS: nhập mật khẩu wifi</li> <li>+ CH_ID: ID tài khoản</li> <li>+ WRITE_APIKEY: copy trên web</li> <li>+ Gán tên và pass wifi kí tự vào mảng char.</li> </ul>
<pre>WiFiClient client; float sensorValue; #define MQ 18 #define SENSOR 34</pre>	<p>MQ 18: kết nối chân GPIO18 với led SENSOR 34: Analog MQ-3 vs G34.</p>
<pre>void setup() {   Serial.begin(115200);   while (!Serial) {     ;   }   ThingSpeak.begin(client);   pinMode(SENSOR, INPUT);   Serial.println("sensor start");   pinMode(MQ, OUTPUT); }</pre>	<p>Hàm setup, thiết lập các thông số, chuẩn bị cho chương trình chạy</p> <ul style="list-style-type: none"> <li>+Thiết lập giao tiếp nối giữa esp32 và cổng truyền dữ liệu với baud rate 115200.</li> <li>+Chạy Blynk với các tham số đầu vào tương ứng là client để tải lên dữ liệu lấy từ mq-3.</li> <li>+Khi thiết lập giao tiếp thành công lệnh println báo hiệu bắt đầu đo khí gas và gán ngõ ra cho led.</li> </ul>
<pre>void loop() {   if(WiFi.status() != WL_CONNECTED){     Serial.print("Attempting to connect to SSID: ");     Serial.println(SECRET_SSID);</pre>	<p>Chương trình chính tạo vòng lặp liên tục:</p> <ul style="list-style-type: none"> <li>+ Khi wifi không kết nối thì báo in ra màn hình báo hiệu "Attempting to connect to SSID" và tên wifi cần phải kết nối</li> </ul>

```

while(WiFi.status() !=
WL_CONNECTED){
    WiFi.begin(ssid, pass
Serial.print(".");
    delay(5000);
}
Serial.println("\nConnected.");
}
sensorValue= analogRead(34);
Serial.println("VALUE:");
Serial.println(sensorValue);
if( sensorValue>2000){
    digitalWrite(18,1);
    delay(50);
}
}

```

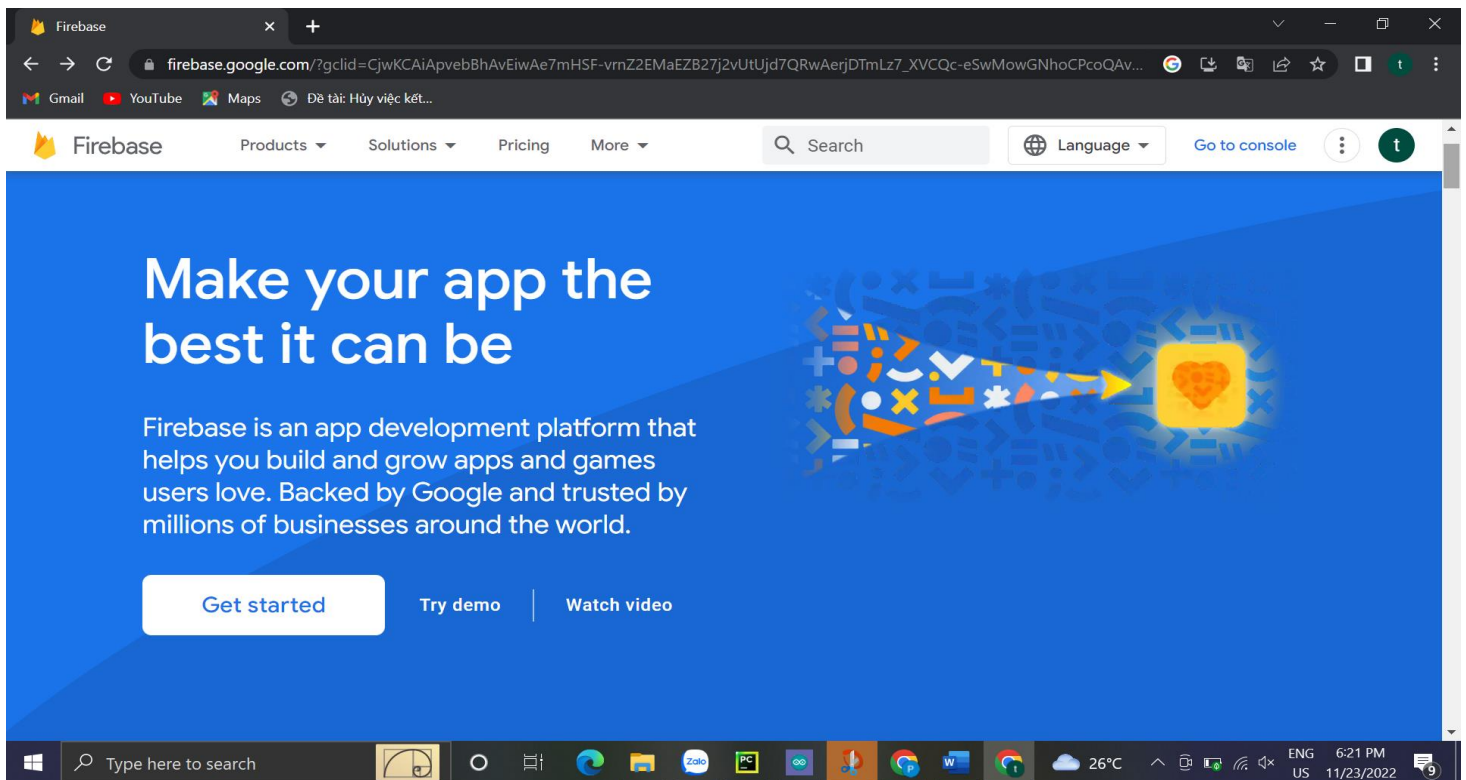
+ Khi wifi được kết nối thành công, báo “Connected”  
 + Gán giá trị sensorValue bằng giá trị khí gas MQ3 đo được.  
 + Nếu sensorValue>2000 thì đèn sáng

6. Các bước thực hiện, giải thích code quá trình cập nhật dữ liệu lên Google Firebase và hình ảnh kết quả thực hiện (video clip demo nếu có).

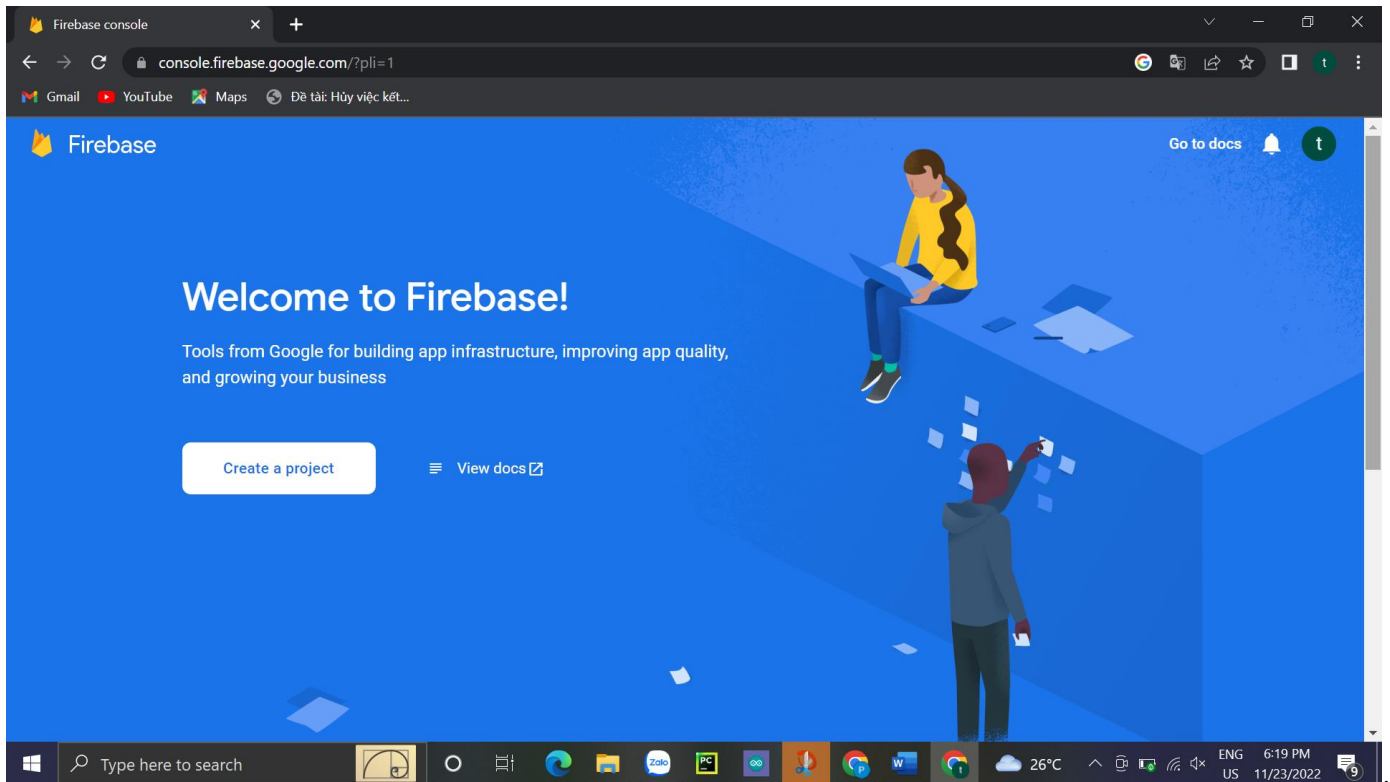
Vào đường link:

[https://console.firebase.google.com/?utm\\_source=firebase.google.com&utm\\_medium=referral&pli=1](https://console.firebase.google.com/?utm_source=firebase.google.com&utm_medium=referral&pli=1)

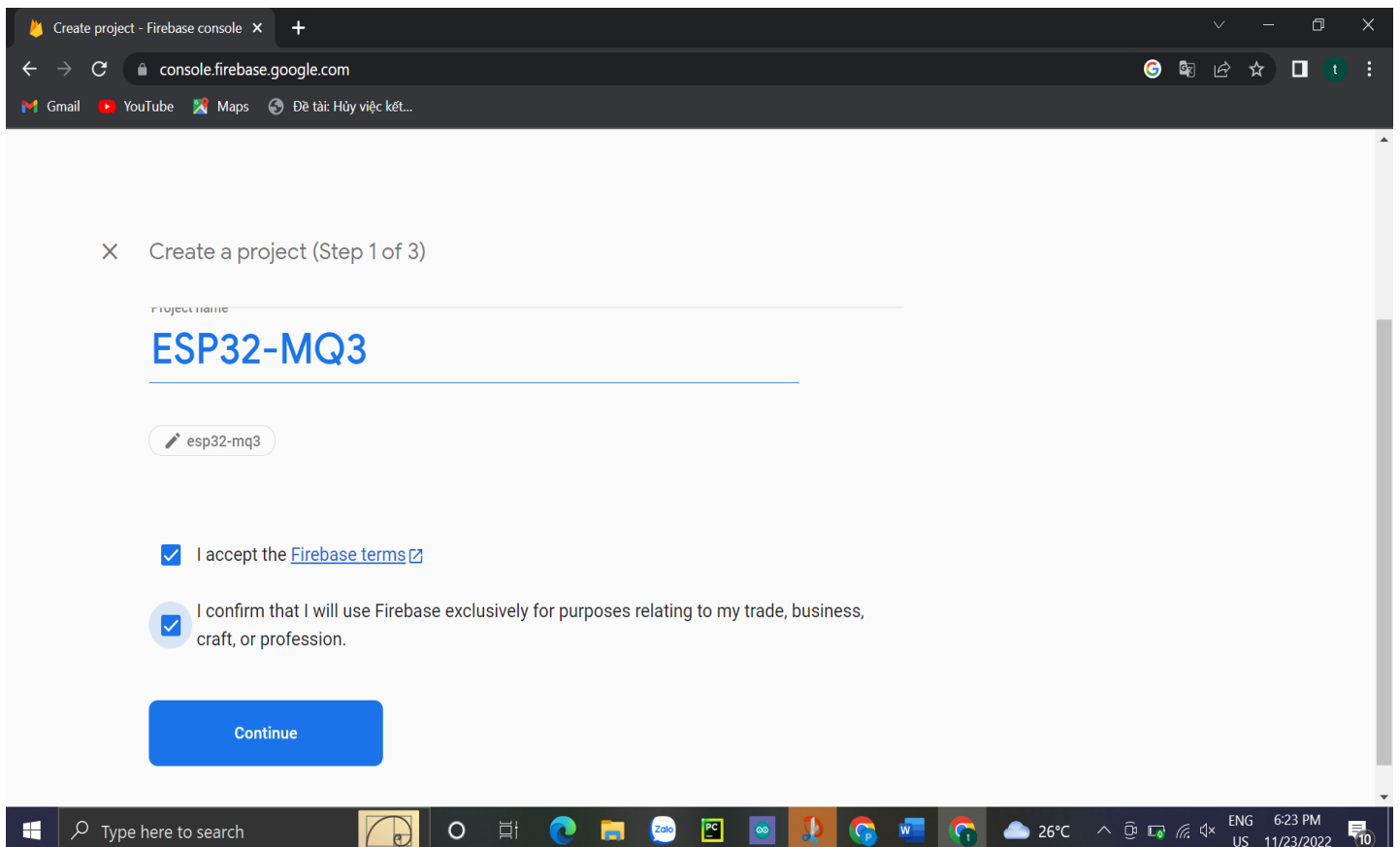
+Chọn “Get Started” để bắt đầu.



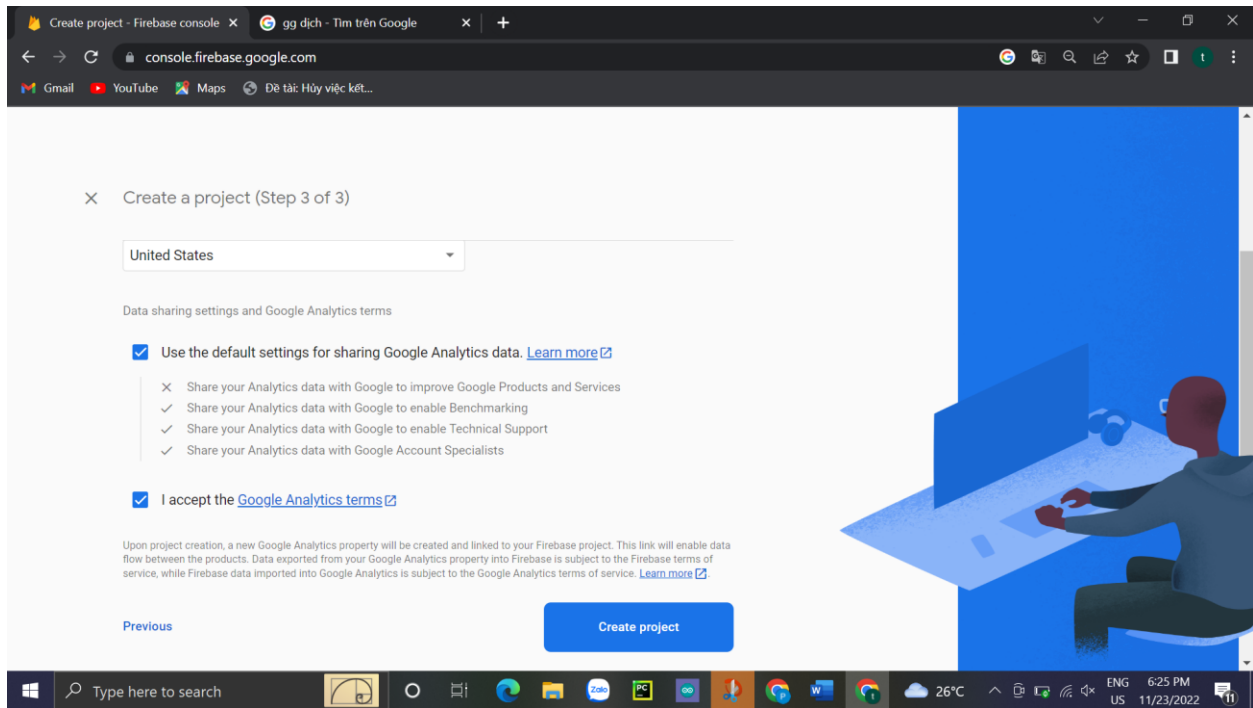
+ Chọn “Create a project” để khởi tạo dự án



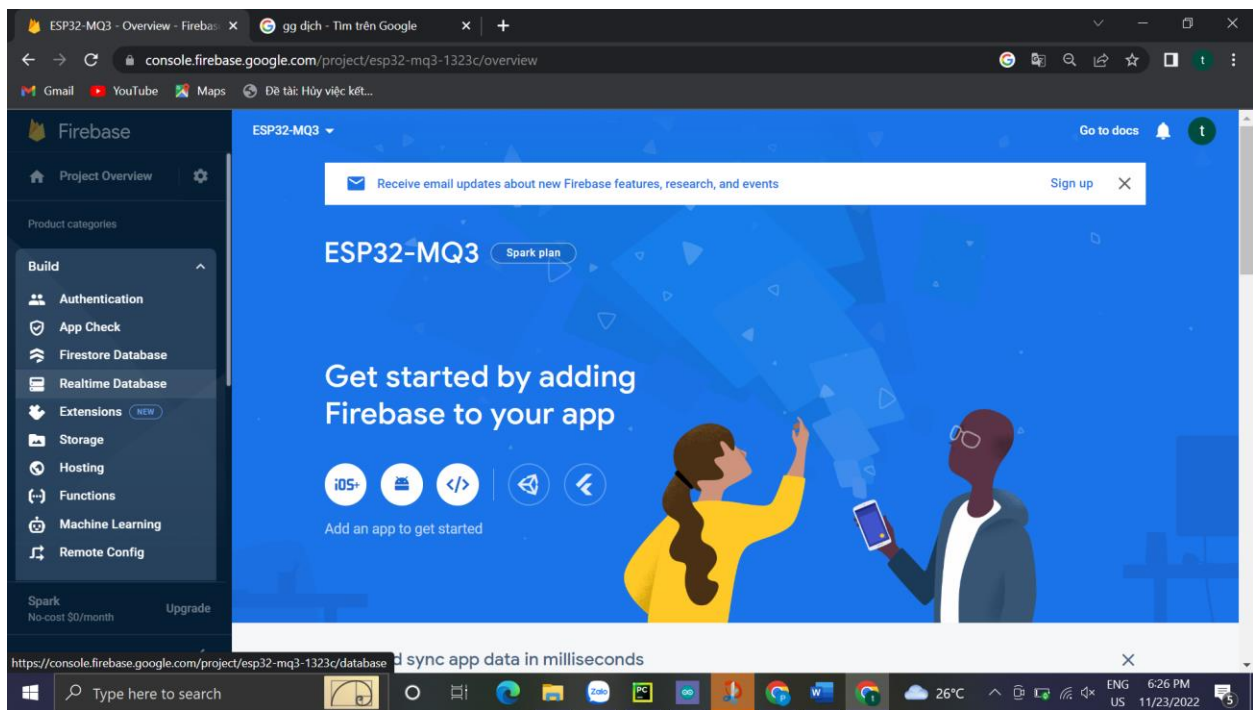
+ Đặt tên cho dự án và đồng ý các điều khoản



## +Chọn vị trí và hoàn tất tạo dự án

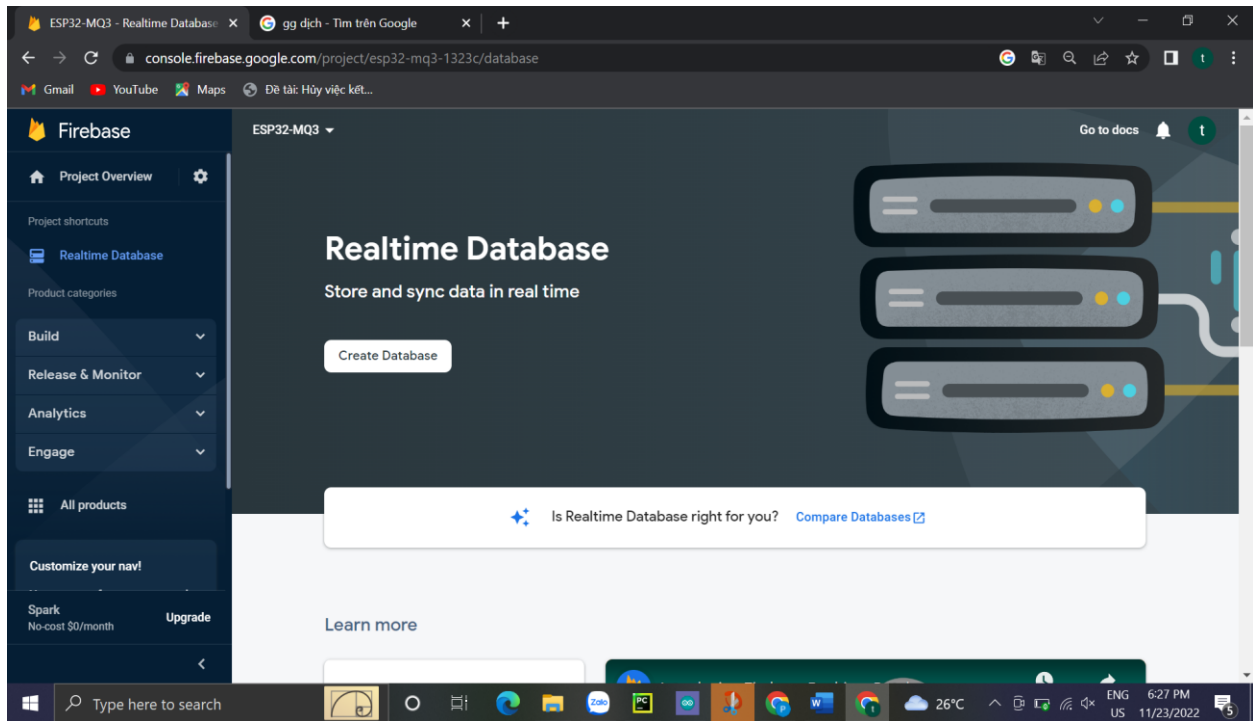


## +Chọn Realtime DataBase

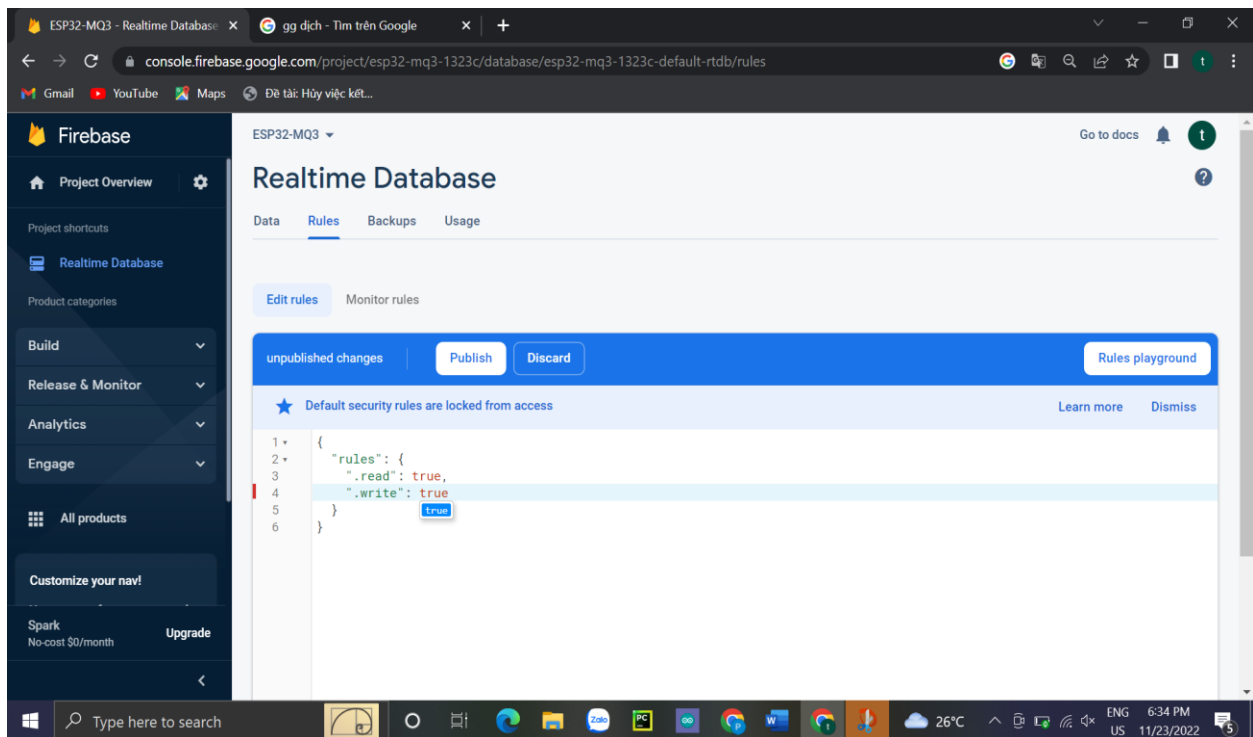




## + Chọn “Create Database”



+ Cài đặt lại dữ liệu cho DataBase: ở Tab Rules, Thay đổi giá trị “false” thành “true” cho 2 trường “read” và “write”, và nhấn Publish để hoàn tất.




ESP32-MQ3

Go to docs

?





# Realtime Database


[Data](#)[Rules](#)[Backups](#)[Usage](#)

 Protect your Realtime Database resources from abuse, such as billing fraud or phishing

[Configure App Check](#)

×

 `https://esp32-mq3-1323c-default-rtdb.firebaseio.com`   

 Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

[Learn more](#)[Dismiss](#)

`https://esp32-mq3-1323c-default-rtdb.firebaseio.com/: null`

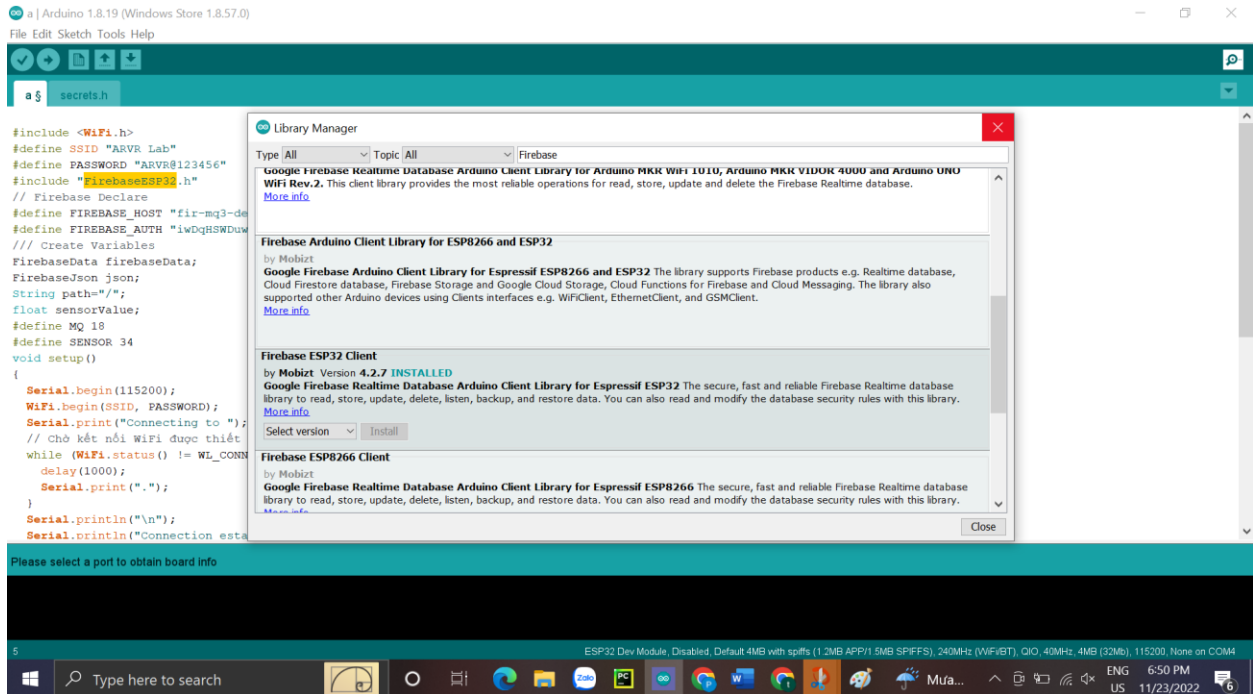
The screenshot shows the Firebase console interface. On the left, the 'Project settings' gear icon is highlighted with a red box and the number 1. A dropdown menu is open, showing 'Project settings' (2), 'Users and permissions', and 'Usage and billing'. The 'Service accounts' tab is selected in the top navigation bar (3). In the left sidebar, the 'Database secrets' option is highlighted with a red box (4). The main content area shows the 'Database secrets' section with a warning message: 'Database secrets are currently deprecated and use a legacy Firebase token generator. Update your source code with the Firebase Admin SDK.' Below this, there is a table with one secret:

Database	Secret
esp32-mq3-1323c-default-rtdb	q2w5Ks918A8z8Aontv1NLU479A6LzhNL2pj18cQs

The 'Secret' value is highlighted with a red box and the number 4. The bottom of the screen shows the Windows taskbar with the time 6:38 PM and date 11/23/2022.

## • ARDUINO IDE

+Cài đặt thư viện FIREBASE cho ESP32: Vào “Tool”→ chọn “ Manage Libraries”, gõ và chọn Thư viện Firebase ESP32 Client

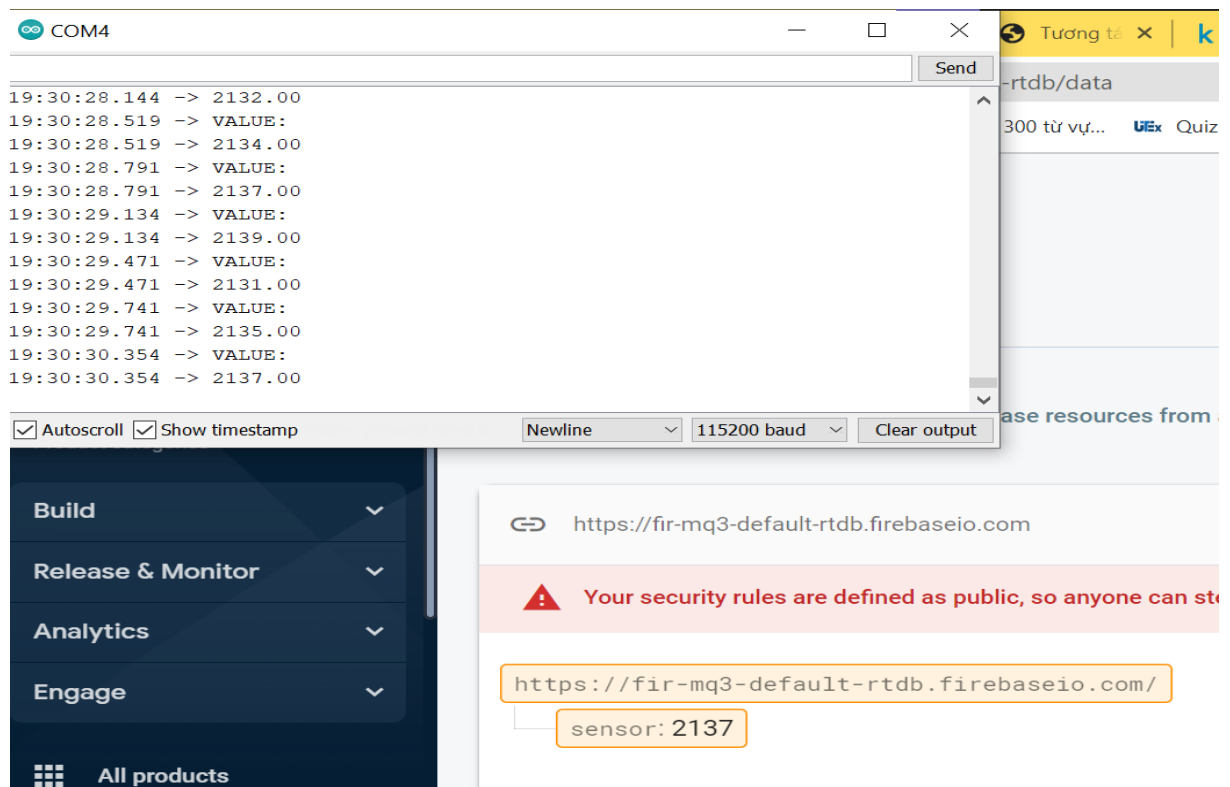


## Giải thích code

<pre>#include &lt;WiFi.h&gt; #define SSID "ARVR Lab" #define PASSWORD "ARVR@123456" #include "FirebaseESP32.h" #define FIREBASE_HOST "fir-mq3-default- rttdb.firebaseio.com" #define FIREBASE_AUTH "iWdQHSWDuw9iIx7x77bpb5EPsckkJQS9Y5sdVk9m" FirebaseData firebaseData; FirebaseJson json; String path="/"; float sensorValue; #define MQ 18 #define SENSOR 34  void setup() {   Serial.begin(115200);   WiFi.begin(SSID, PASSWORD);   Serial.print("Connecting to ");   // Chờ kết nối Wifi được thiết   while (WiFi.status() != WL_CONN     delay(1000);     Serial.print(".");   }   Serial.println("\n");   Serial.println("Connection esta</pre>	<p>Khai báo thư viện WIFI</p> <p>Khai báo tài khoản và mật khẩu Wifi.</p> <p>Khai báo thư viện FireESP32</p> <p>Khai báo Host và Authentication cho module và khai báo biến trung gian để truy xuất đến Firebase</p> <p>Khai báo ki tự ‘\’</p> <p>Khai báo Biến MQ 18 và SENSOR 34</p>
<pre>void setup() {   Serial.begin(115200);   WiFi.begin(SSID, PASSWORD);   Serial.print("Connecting to ");   // Chờ kết nối Wifi được thiết lập   while (WiFi.status() != WL_CONNECTED) {     delay(1000);     Serial.print(".");   }   Serial.println("\n");   Serial.println("Connection established!");</pre>	<p>-Hàm setup, thiết lập các thông số, chuẩn bị cho chương trình chạy</p> <p>-Thiết lập giao tiếp nối giữa esp32 và cổng truyền dữ liệu với baud rate115200. Dùng hàm WiFi.begin để kết nối với đầu vào là Tài khoản và Mật khẩu đã khai báo bên trên.</p> <p>Sử dụng WiFi.status để kiểm tra trạng thái kết nối ở của Wifi bằng vòng lặp While: Khi nào còn chưa kết nối được nó sẽ in.... kết nối xong nó sẽ thông báo thành công và in ra địa chỉ IP sử dụng WiFi.localIP()</p>

<pre>Serial.print("IP address: "); Serial.println(WiFi.localIP());</pre>	
<pre>Firebase.begin(FIREBASE_HOST,FIREBASE_AUTH); Firebase.reconnectWiFi(true); if (!Firebase.beginStream(firebaseData, path))     Serial.println("REASON: " + firebaseData.errorReason());     Serial.println();</pre>	<p>Kết nối Firebase với ESP32 sử dụng Firebase.begin() với thông số đầu vào đã khai báo bên trên.</p> <p>Firebase.reconnectWiFi(true) Kiểm tra xem Firebase có được kết nối với Wifi không, nếu không, sử dụng firebaseData.errorReason() nếu wifi được kết nối và không có phát trực tuyến hoặc bất kỳ thay đổi nào trong firebase, sẽ xuất hiện errorReason.</p>
<pre>pinMode(SENSOR, INPUT); Serial.println("sensor start"); pinMode(MQ, OUTPUT);</pre>	<p>Cấu hình chân GIPO muốn làm INPUT bằng biến SENSOR( để đọc dữ liệu từ cảm biến</p> <p>Cấu hình chân GPIO muốn làm OUTPUT bằng biến MQ( chân để cảnh báo đèn khi đạt ngưỡng)</p>
<pre>void loop() {     sensorValue= analogRead(34);     Serial.println("VALUE:");     Serial.println(sensorValue);     if( sensorValue&gt;2500){         digitalWrite(18,1);         delay(50);     }     Firebase.setInt(firebaseData, path + "/sensor",sensorValue); }</pre>	<p>Khai báo vòng lặp chính:</p> <p>Đọc dữ liệu cảm biến bằng cách gán biến sensorValue sử dụng hàm analogRead(34) với đối số đầu vào chân GPIO mà cảm biến đã cắm vào.</p> <p>Vòng lặp if để kiểm tra nếu sensorValue lớn hơn ngưỡng thiết lập thì sẽ bật sáng đèn dùng digitalWrite() với tham số đầu vào là chân GPIO và mức 1 để bật đèn sáng</p>
<pre>    Firebase.setInt(firebaseData, path + "/sensor",sensorValue); }</pre>	<p>Sử dụng Firebase.setInt() để đưa dữ liệu vào firebaseData</p> <p>Tham số đầu vào là chuỗi string+ trường data( sensor) và giá trị truyền là sensorValue</p>

- Hình ảnh :



# Tài liệu tham khảo

- [1] J. Clark, "back4app," 2022. [Online]. Available: <https://blog.back4app.com/thingspeak-vs-firebase/>. [Accessed 22 11 2022].
- [2] HuuHV, "VIBLO," 18 12 2018. [Online]. Available: <https://viblo.asia/p/uu-diem-va-nhuoc-diem-cua-google-firebase-cac-notification-api-can-thiet-cho-phia-server-E375zwJWKGW>. [Accessed 22 11 2022].
- [3] Marcello A. Gómez Maureira, Daan Oldenhof, Livia Teernstra, "ThingSpeak – an API and Web Service," 2014.
- [4] T. V. Cuong, 27 3 2017. [Online]. Available: <https://viblo.asia/p/so-sanh-aws-va-azure-4dbZN0Qn5YM>.
- [5] A. w. site, "Amazon," [Online]. Available: <https://aws.amazon.com/vi/what-is-aws/>.
- [6] Microsoft, "Azure Iot- internet of things platform," [Online]. Available: <https://azure.microsoft.com/en-us/solutions/iot/>.



