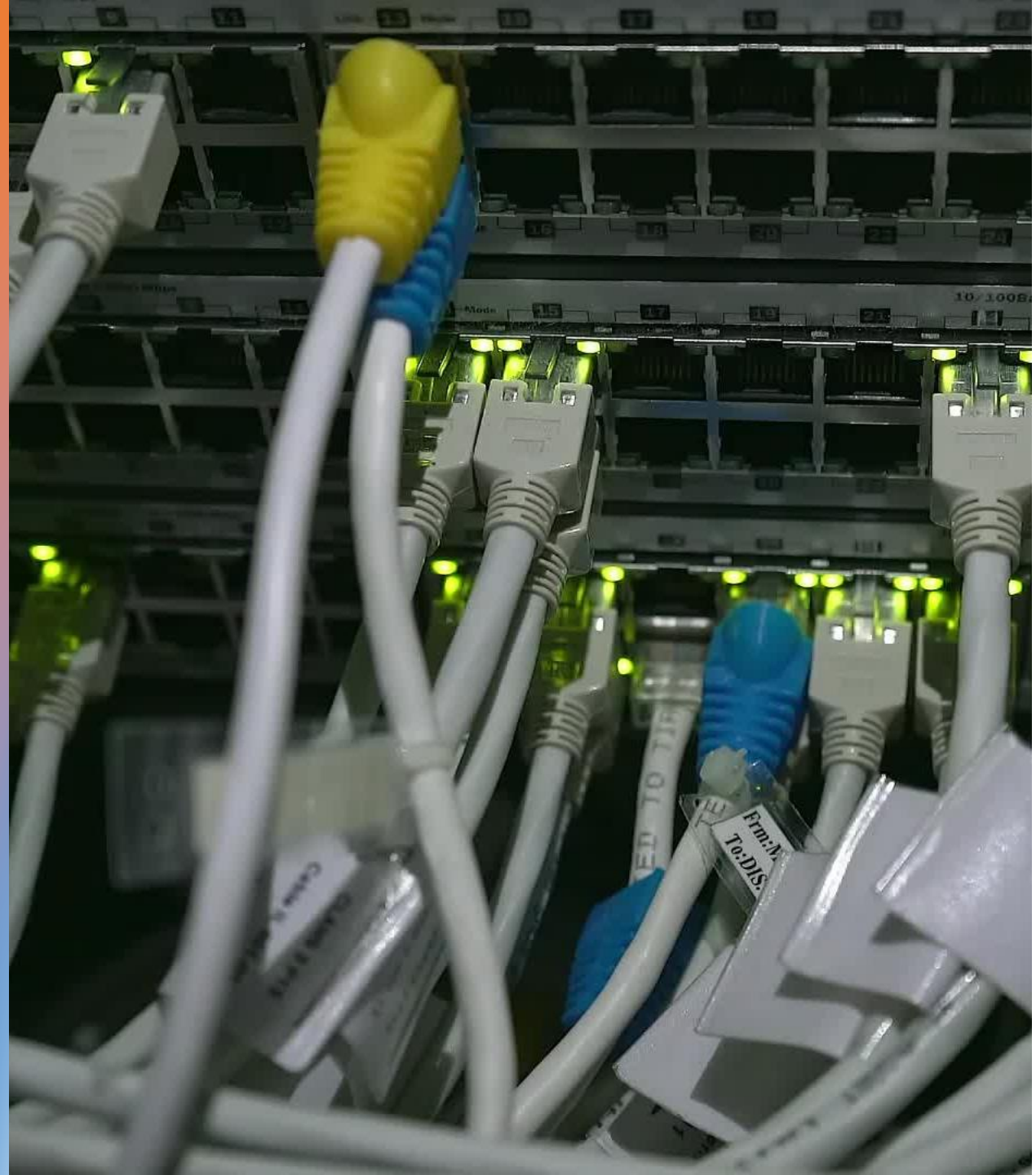
Internet of Things (IoT)

IFA 2025

CoAP

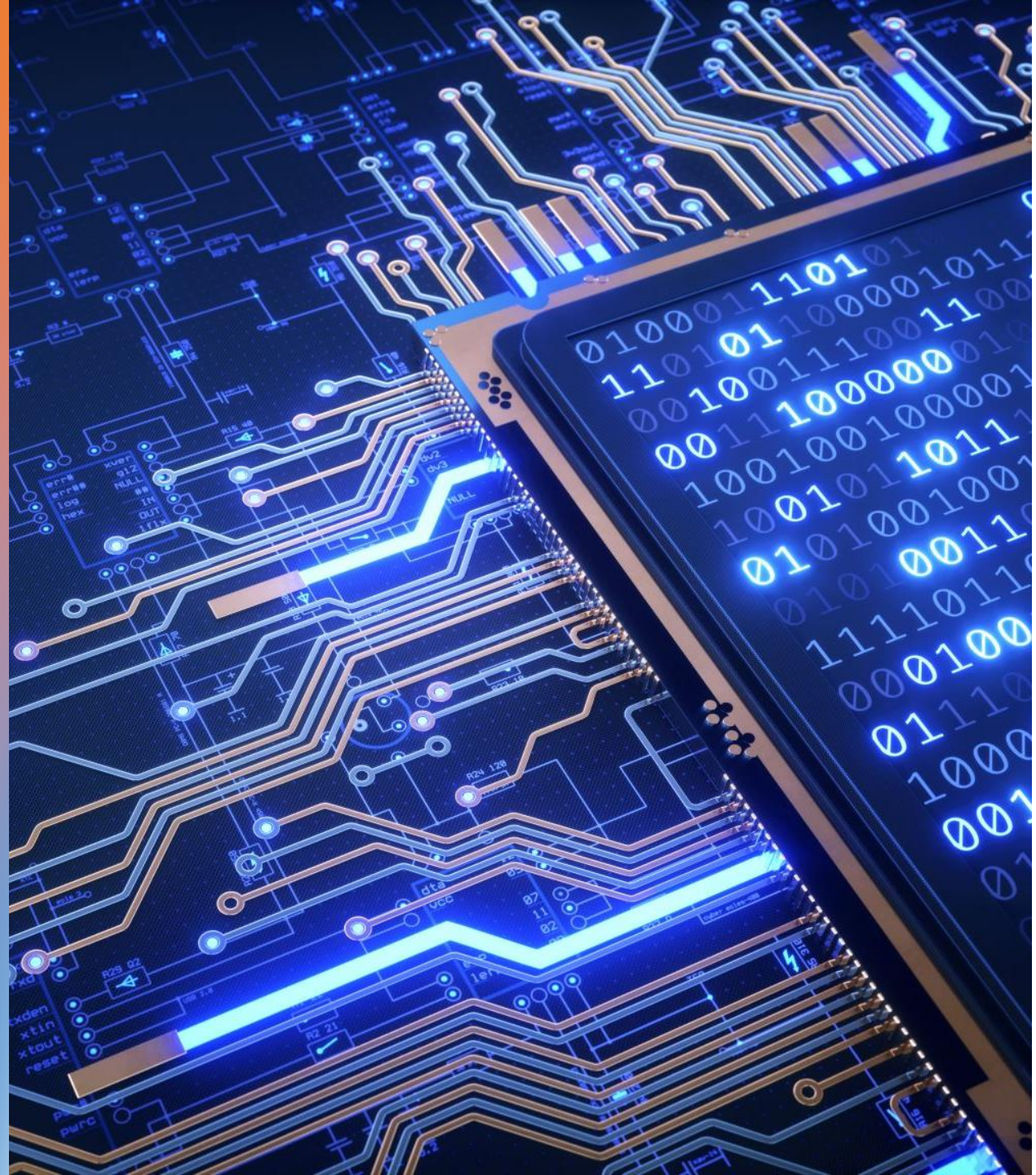
outline

- IoT
- Cloud
- Blynk
- Firebase
- BoT Telegram
- MQTT Protocol



WHAT IS IOT ?

- Internet of Things (IoT) adalah jaringan objek fisik — perangkat, kendaraan, bangunan, dan barang-barang lainnya yang disisipi dengan elektronik, perangkat lunak, sensor, dan konektivitas jaringan — yang memungkinkan objek-objek ini mengumpulkan dan bertukar data.



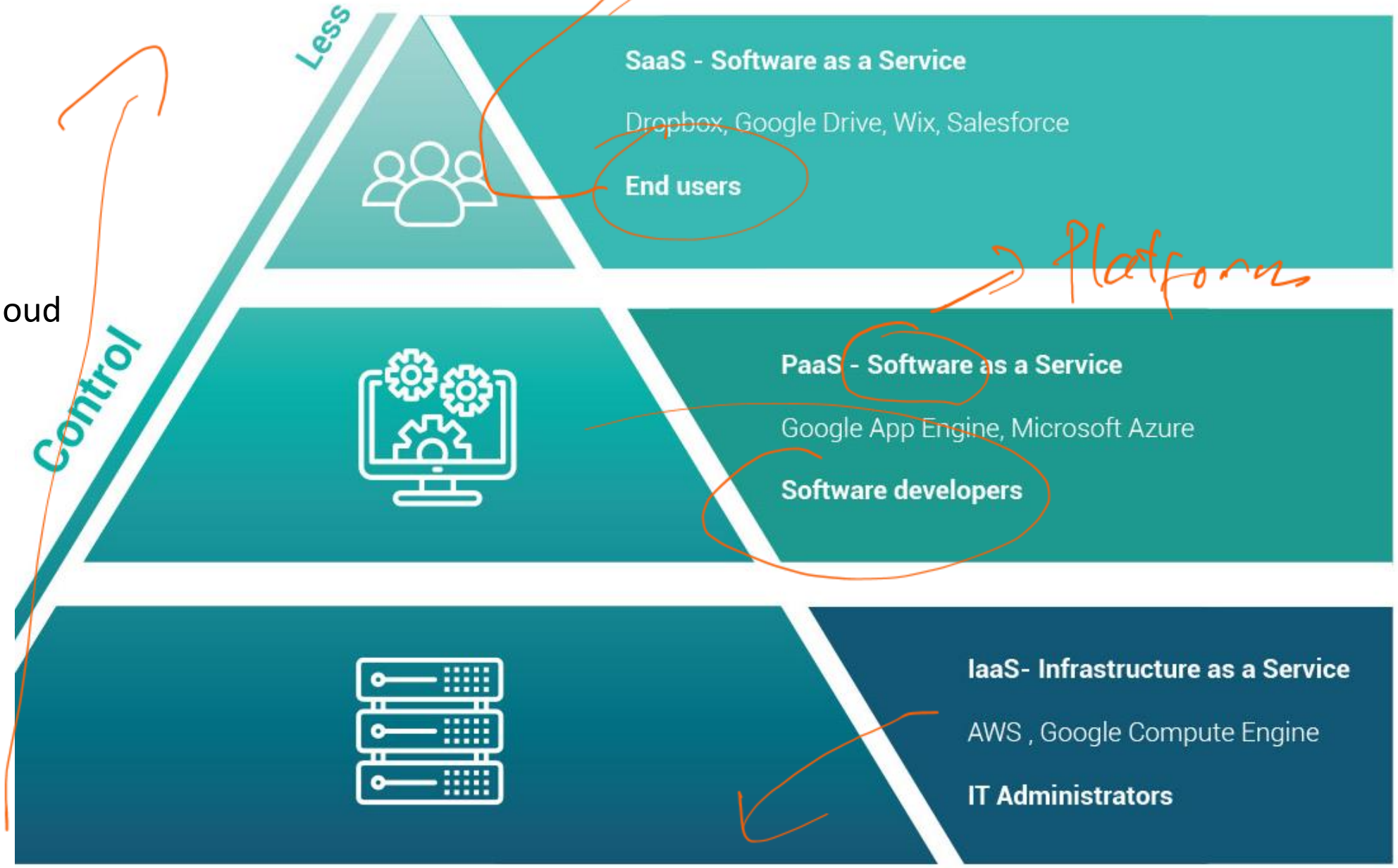
NAMA LAIN (ALIAS) DARI IOT

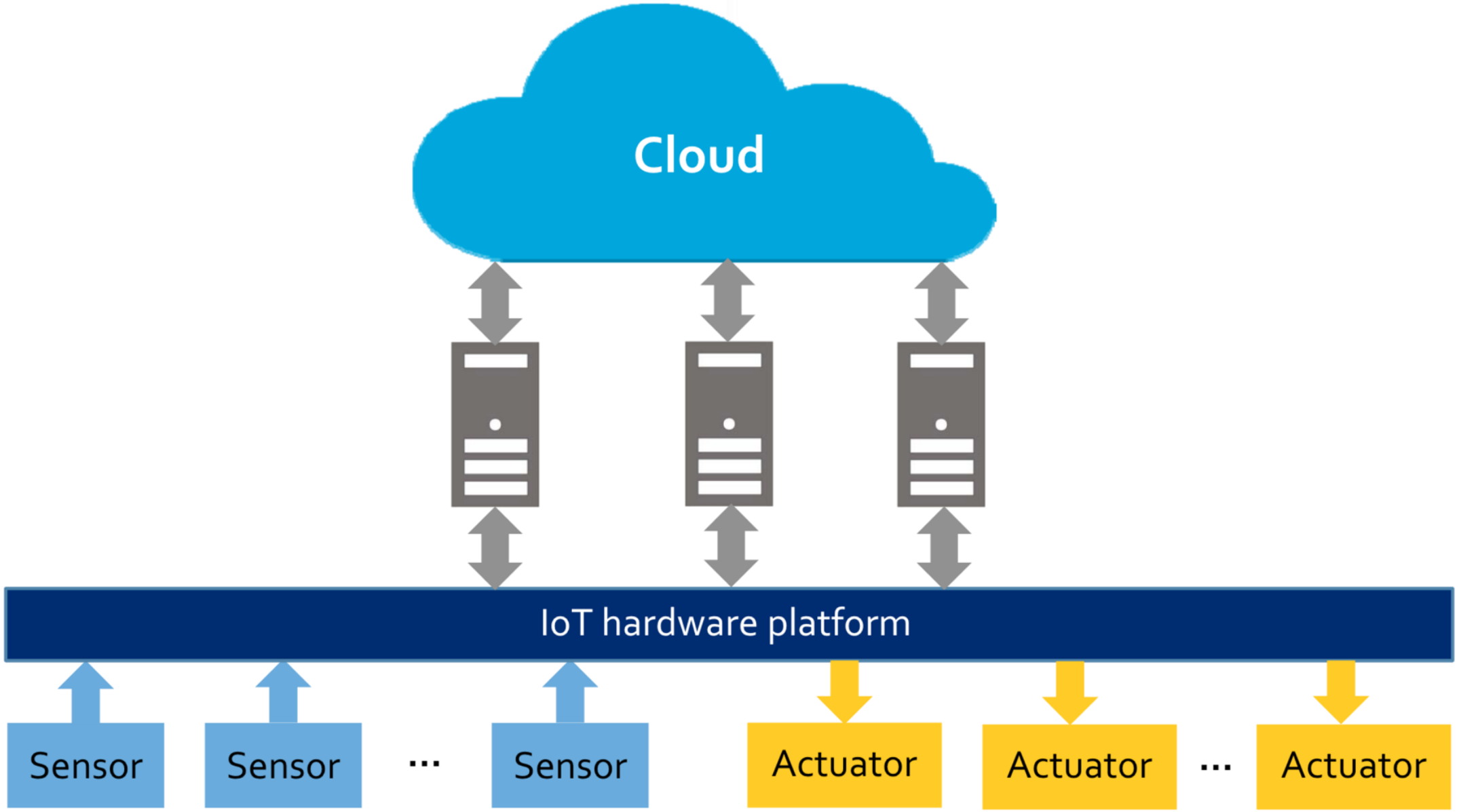
- M2M (Machine to Machine)
- “Internet of Everything” (Cisco Systems) manusia, data, proses
- “World Size Web” (Bruce Schneier)
- “Skynet” (Terminator movie)

CLOUD COMPUTING

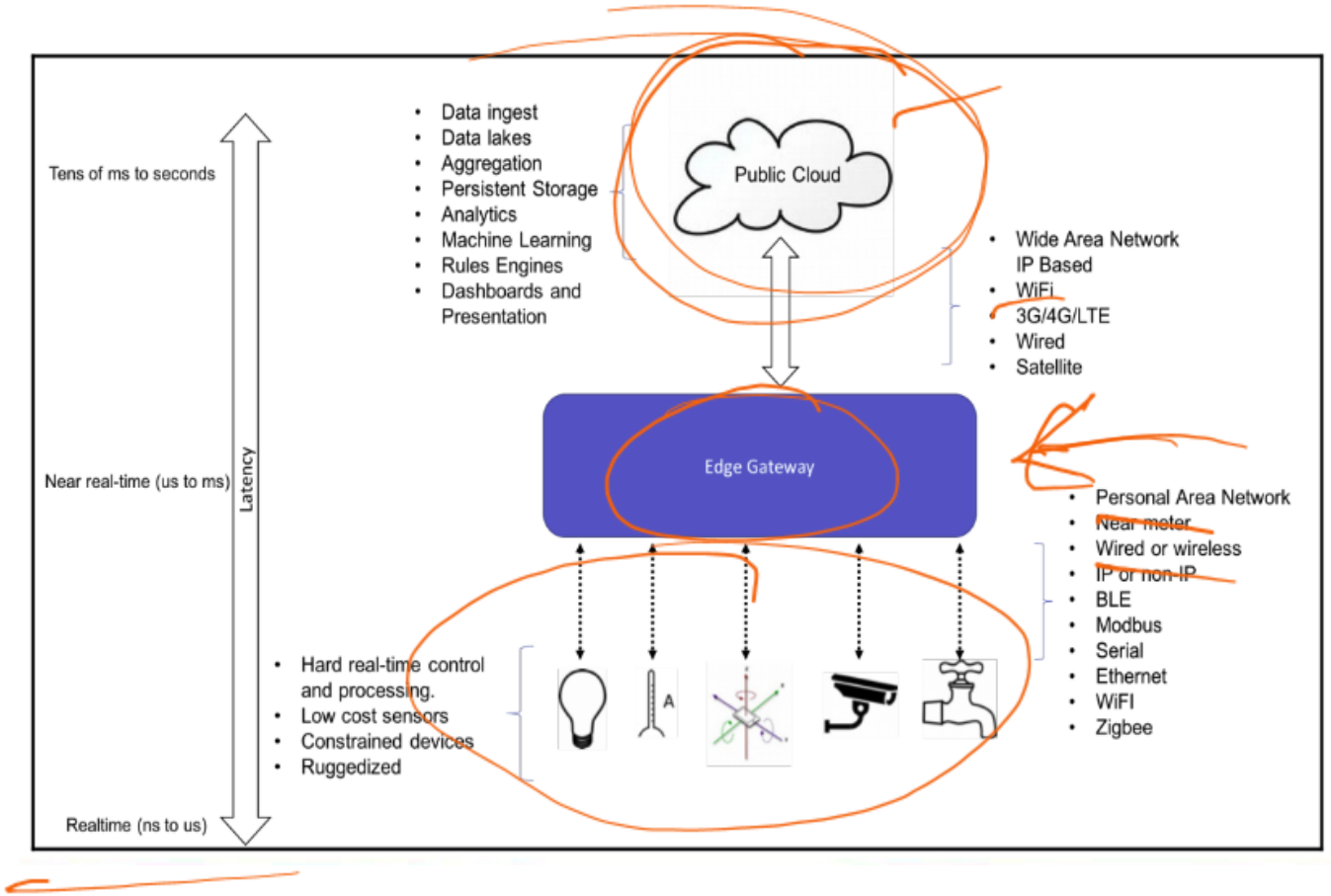
- Cloud Computing adalah istilah umum yang digunakan untuk menggambarkan komputasi berbasis jaringan yang terjadi melalui Internet
  - Menggunakan Internet untuk komunikasi dan transportasi serta penyediaan perangkat keras, perangkat lunak, dan layanan jaringan kepada klien

- 3 Layanan Cloud




Cloud
IoT hardware platform
Sensor
Sensor
...
Sensor
Actuator
Actuator
...
Actuator

- Batasan Arsitektur cloud



↳ real-time.

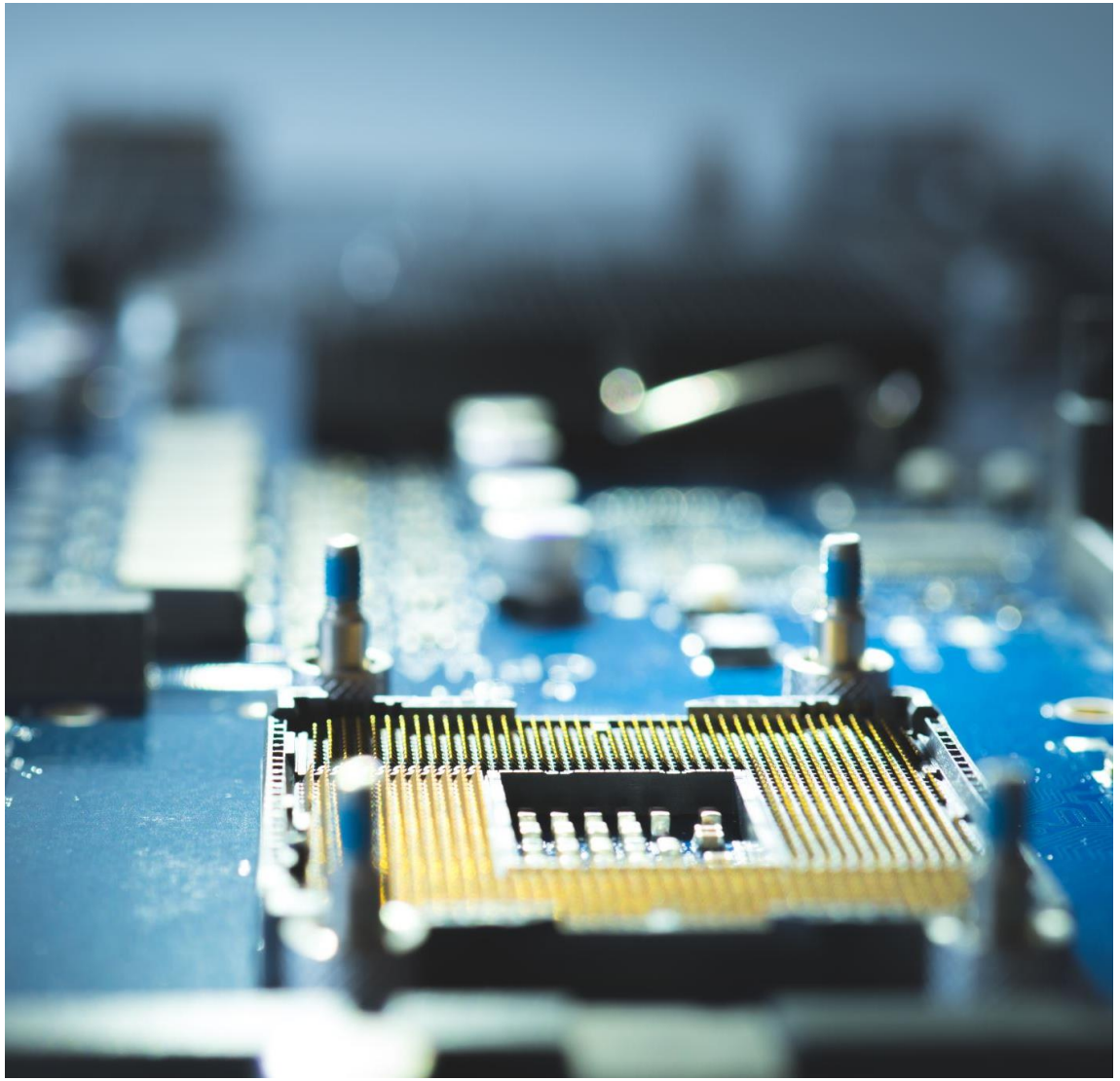
MQTT – THE PROTOCOL OF IOT

MQTT

- MQTT adalah singkatan dari Message Queuing Telemetry Transport. Ini adalah protokol komunikasi yang digunakan dalam komunikasi mesin ke mesin (M2M) dan Internet of Things (IoT).

Karakteristik MQTT

1. **Ringan dan Efisien**
2. **Model Pub/Sub**
3. **QoS (Quality of Service) yang Dapat Disesuaikan**
4. **Koneksi yang Ringan**
5. **Keamanan**
6. **Skalabilitas**
7. **Dukungan berbagai platform**



Broker

MQTT Broker memiliki suatu alamat yang dapat diakses oleh Publisher dan Subscriber.

EMQX.

Broker = Server WA.

Publisher = Orang yg mengirimkan pesan

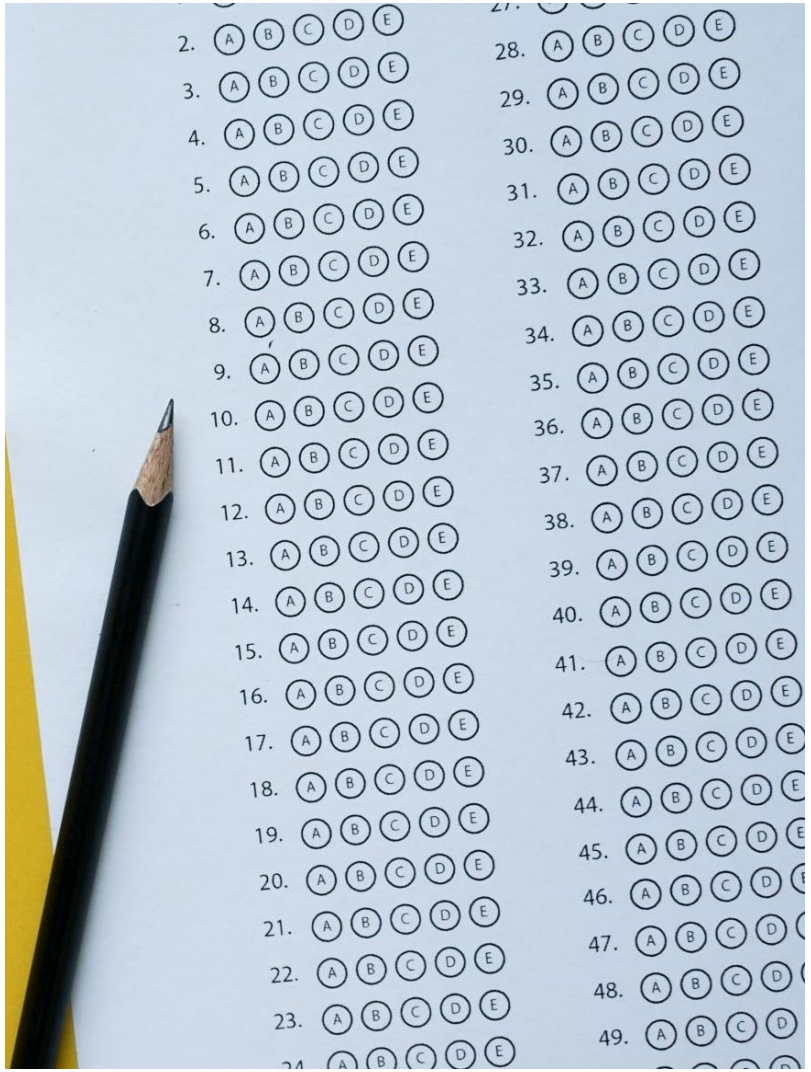
Subscriber = anggota WA G.

sensor / temperatur / kamartidur
sensor / temperatur / ruang tamu.

Elemen Data yang dikirimkan oleh MQTT

1. **Topik (Topic)**
2. **Payload** teks →
3. **Pesan ID (Message ID)** → level QoS,
4. **Kualitas Layanan (Quality of Service/QoS)**:
5. **Informasi Tambahan**

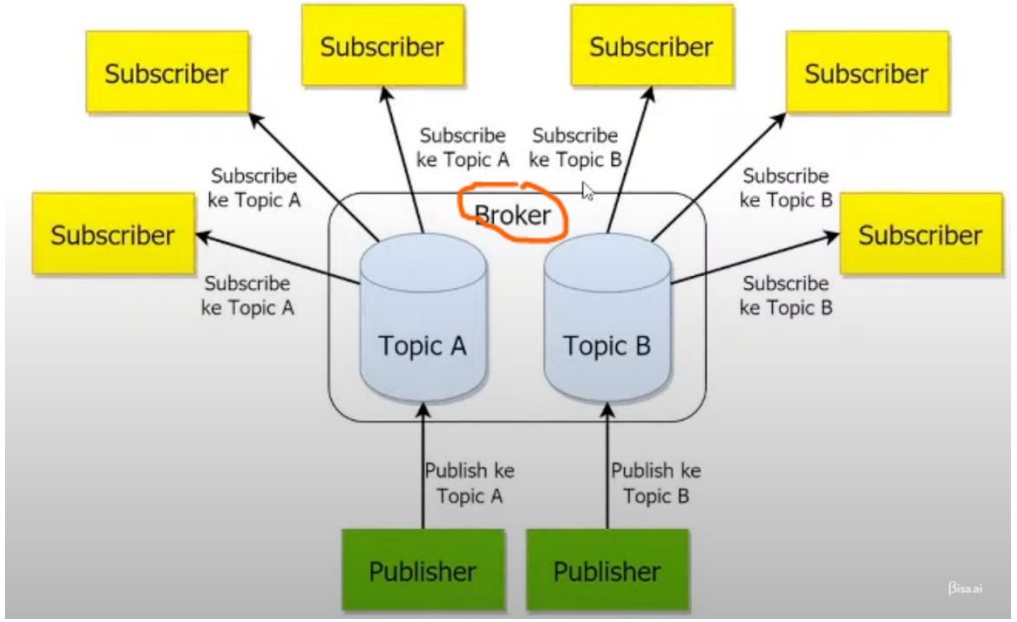
temperatur = 30
JSON { "temp": 24 }



Komponen Utama MQTT

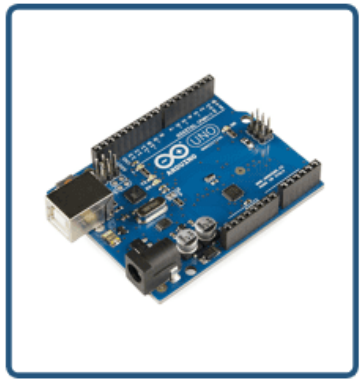
- MQTT terdiri dari tiga komponen utama yang bekerja bersama untuk memungkinkan komunikasi antara perangkat:

1. **Publisher (Penerbit)**: Publisher adalah perangkat atau entitas yang mengirimkan pesan ke broker MQTT.
2. **Broker**: Broker adalah perantara pusat dalam arsitektur MQTT. Semua pesan yang diterima oleh broker dari publisher akan disalurkan ke subscriber yang tertarik pada topik yang sama dengan pesan yang diterbitkan.
3. **Subscriber (Pelanggan)**: Subscriber adalah perangkat atau entitas yang berlangganan (subscribe) ke broker untuk menerima pesan-pesan dari topik tertentu yang menarik minatnya.



Sensor Echo

Publisher



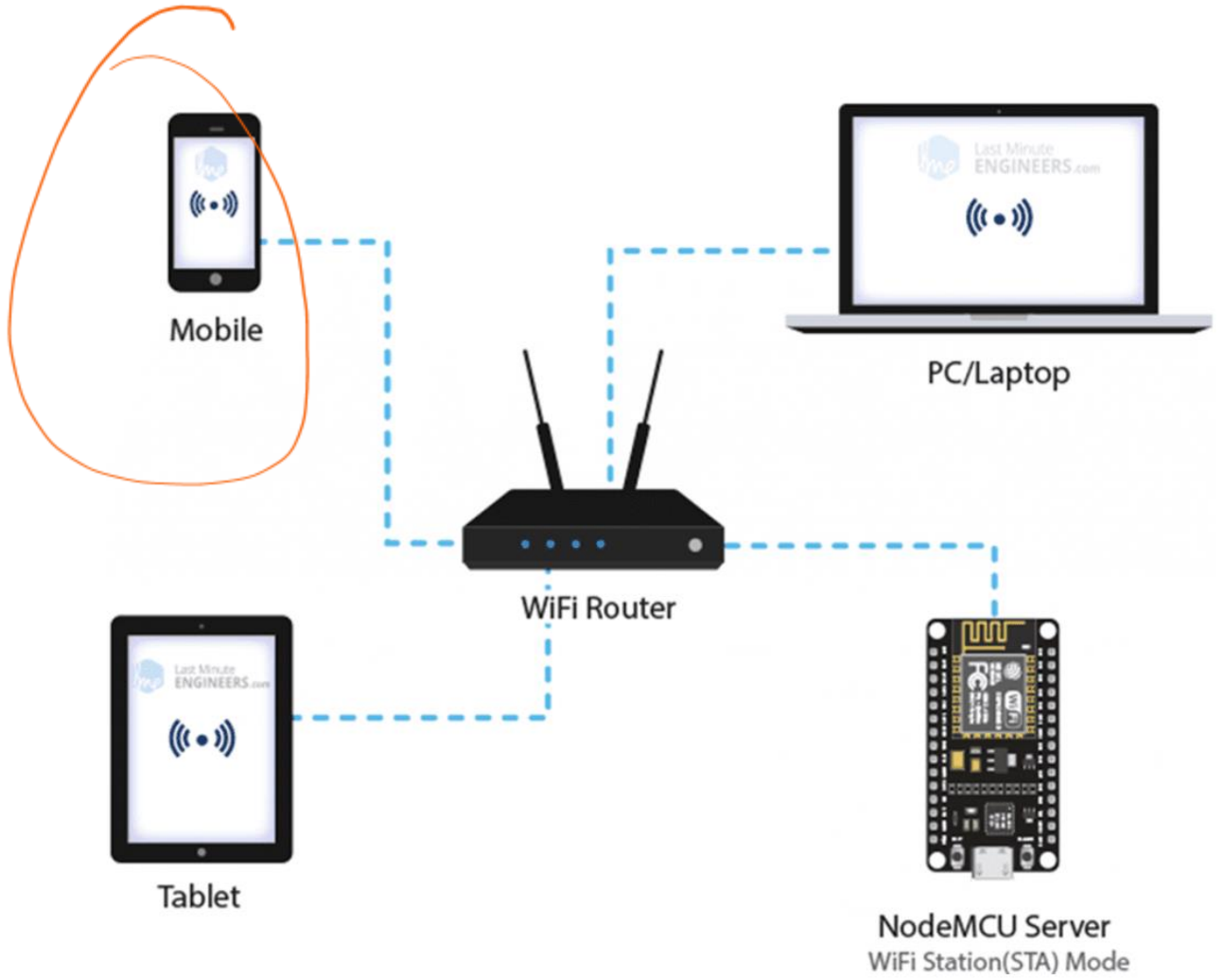
Arduino Uno

ESP32

Raspberry Pi

Android

Subscriber

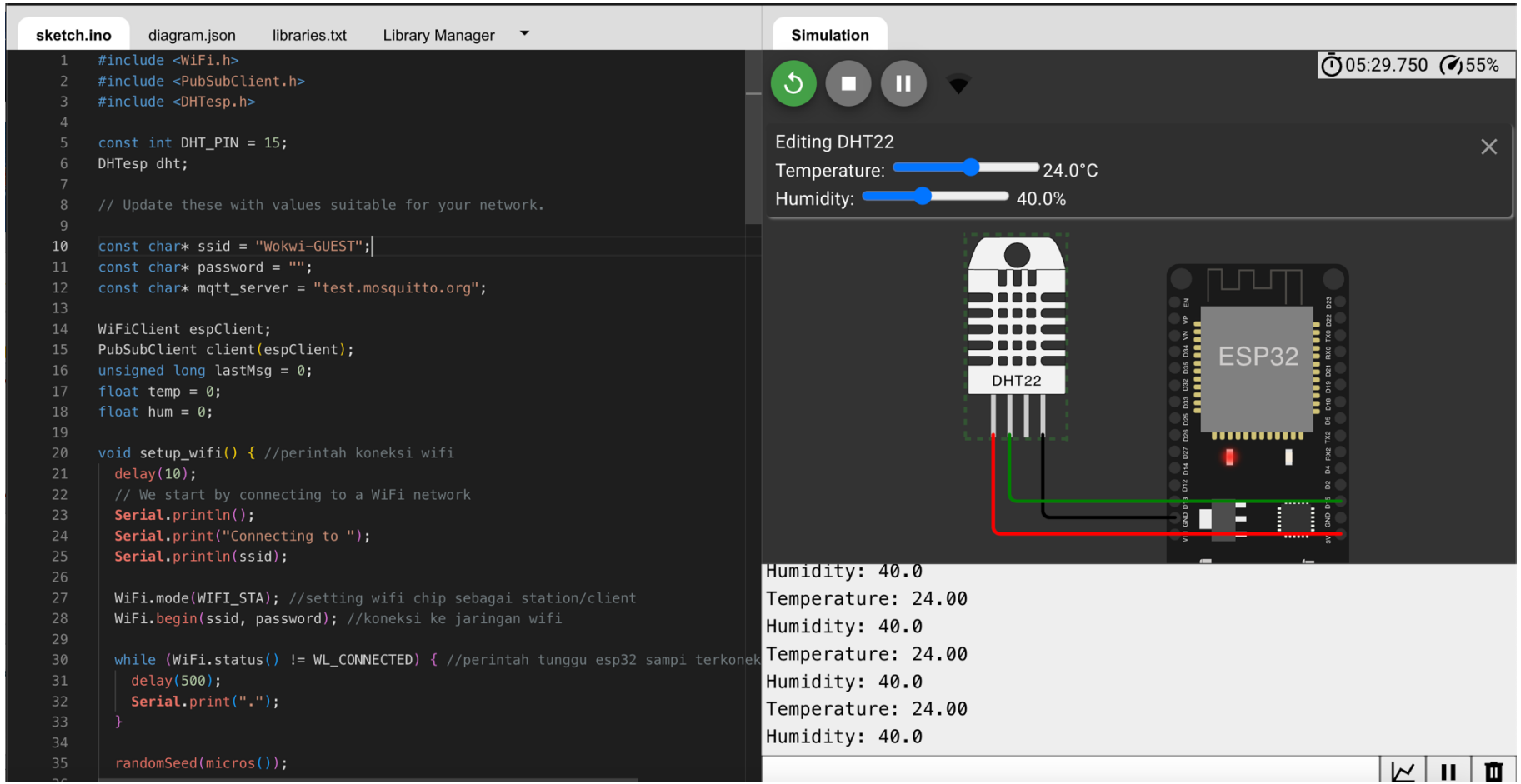

Mobile
PC/Laptop
WiFi Router
Tablet
NodeMCU Server
WiFi Station(STA) Mode

3 Level QoS (Quality of Services) Pada MQTT

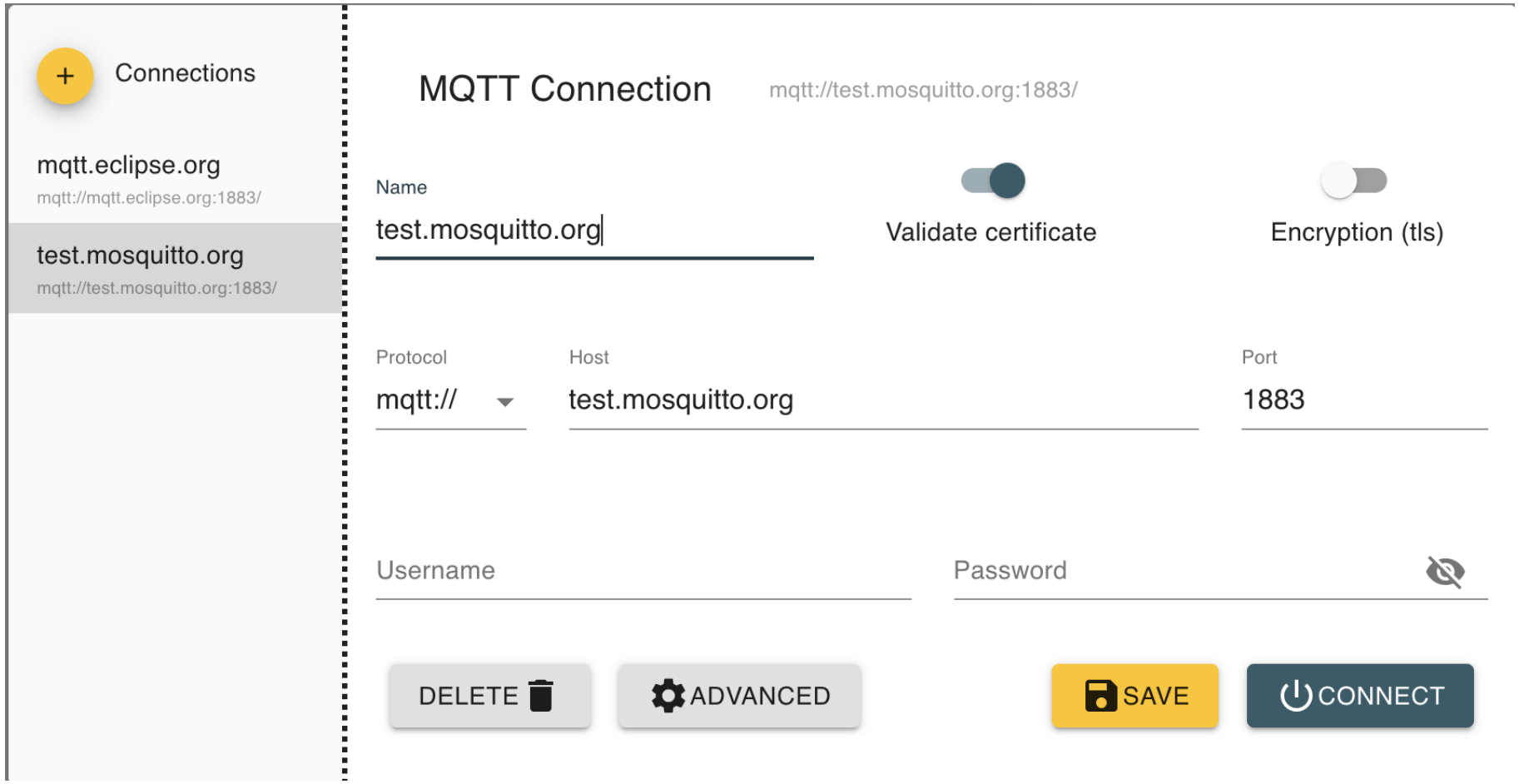
1. **QoS Level 0 (At most once)**:
    1. Disebut juga "fire and forget".
    2. Pesan dikirimkan sekali tanpa konfirmasi atau jaminan pengiriman.
    3. Broker akan meneruskan pesan hanya sekali, tanpa memperhatikan apakah pesan tersebut sampai ke subscriber atau tidak.
2. **QoS Level 1 (At least once)**:
    1. Memastikan setidaknya satu pengiriman pesan ke subscriber.
    2. Pesan dikirimkan oleh publisher dan broker akan mengonfirmasi penerimaan pesan.
    3. Jika pesan tidak terkonfirmasi oleh subscriber, broker akan mengirimkan kembali pesan tersebut.
3. **QoS Level 2 (Exactly once)**:
    1. Memberikan jaminan bahwa setiap pesan akan diterima oleh subscriber tepat satu kali.
    2. Prosesnya lebih kompleks dan memakan lebih banyak sumber daya karena melibatkan tiga langkah pertukaran pesan antara publisher, broker, dan subscriber.

Project DHT 22

https://wokwi.com/projects/425058095440888833

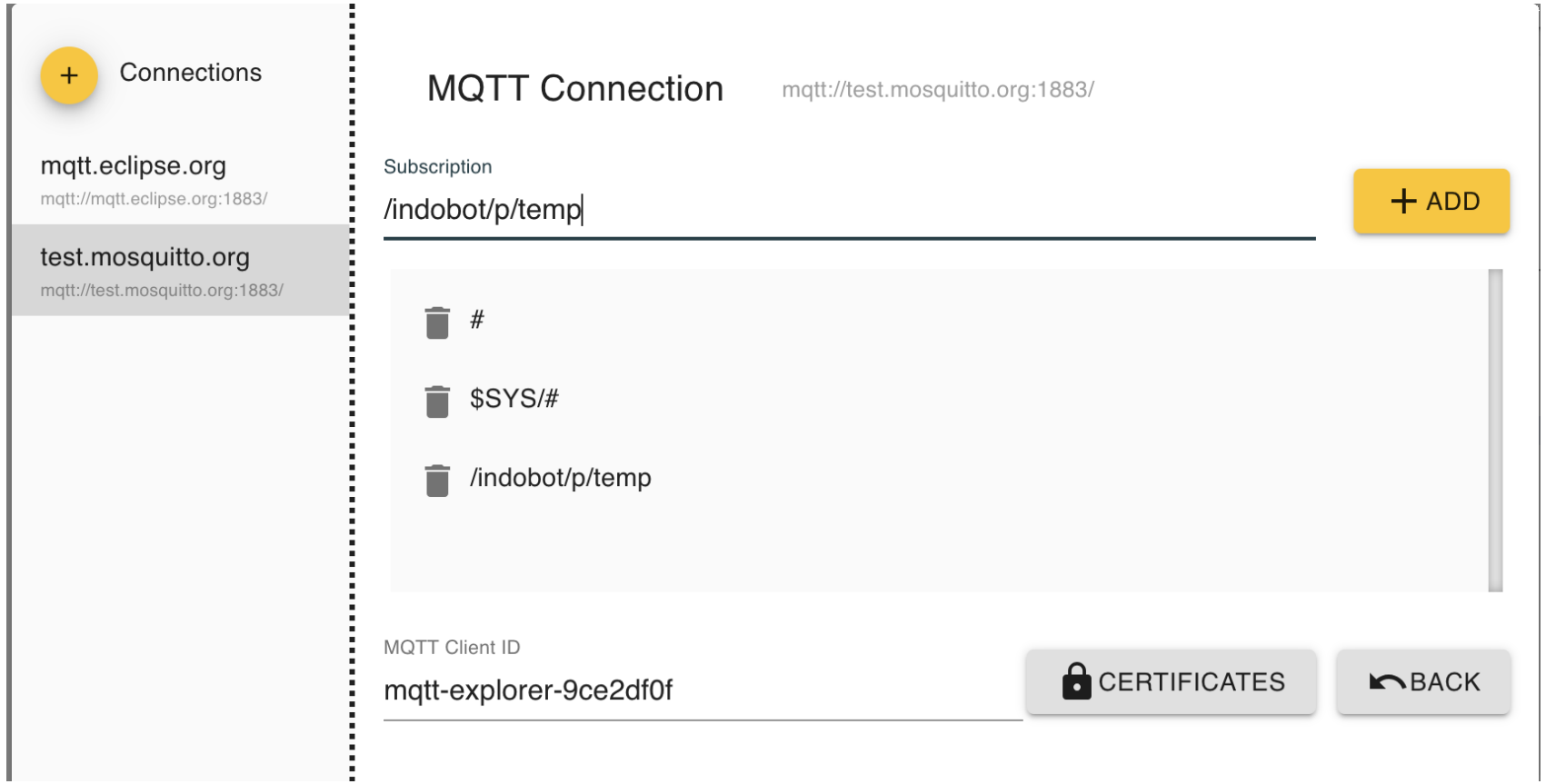


MQTT Explorer



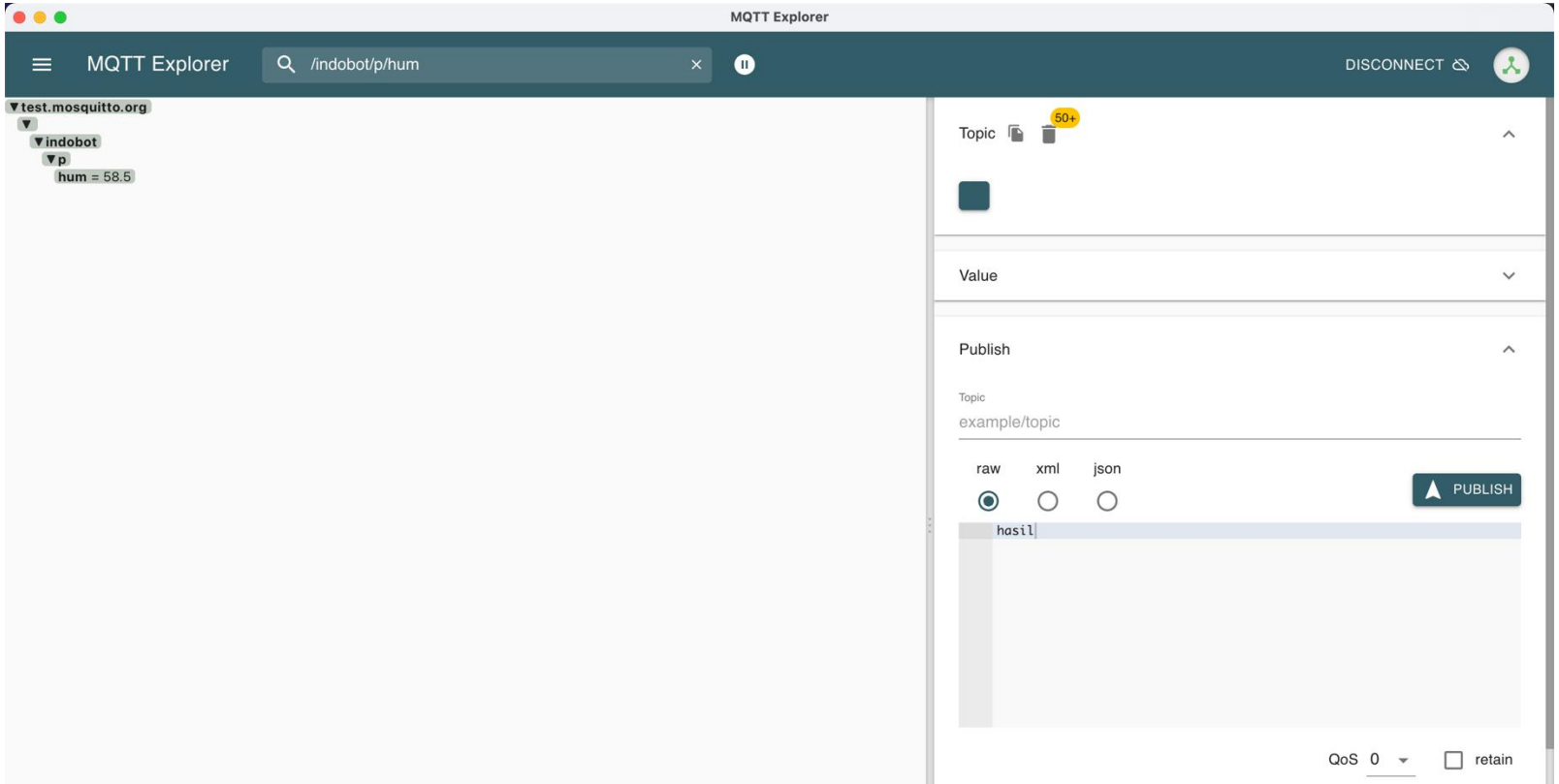
Pilih : advanced untuk subscription

Pilih : connect



Tampilan advanced dari menu sebelumnya

Subscription ke
/indobot/p/temp atau /indobot/p/hum, selanjutnya klik **button add**



MQTT Explorer

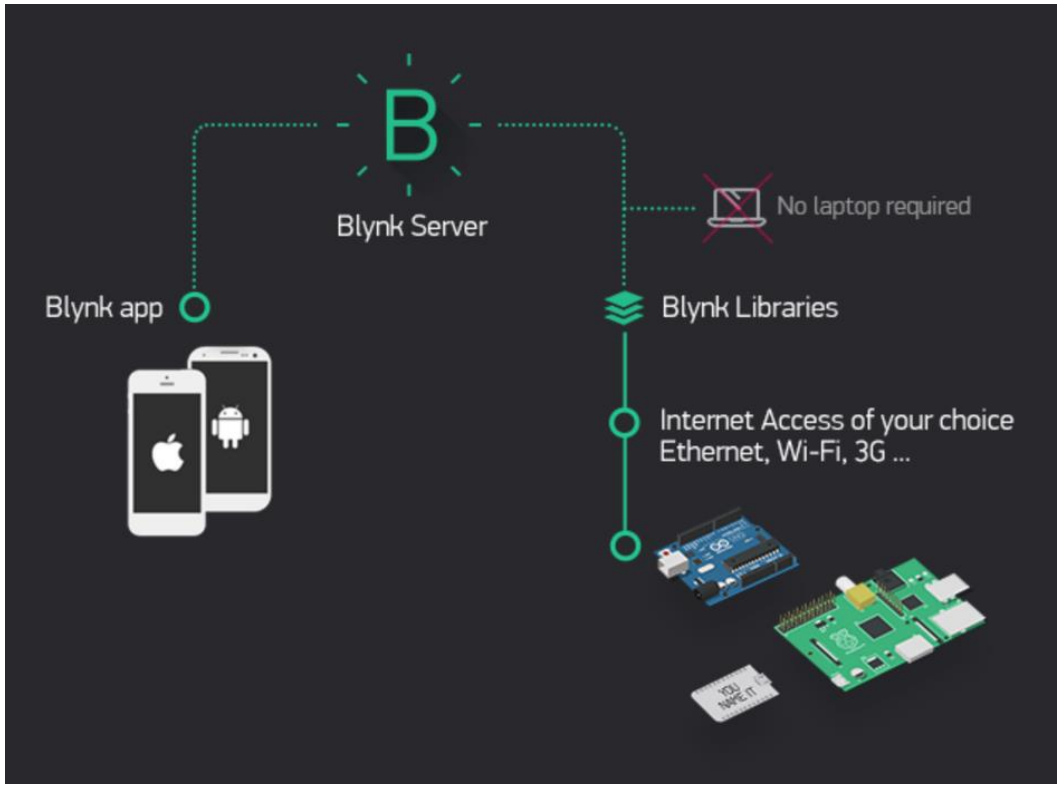
Masukkan /indobot/p/temp untuk melihat suhu

Masukkan /indobot/p/hum untuk melihat kelembaban

Masukkan /indobot/p/mqtt untuk melihat pesan lain dari ESP32

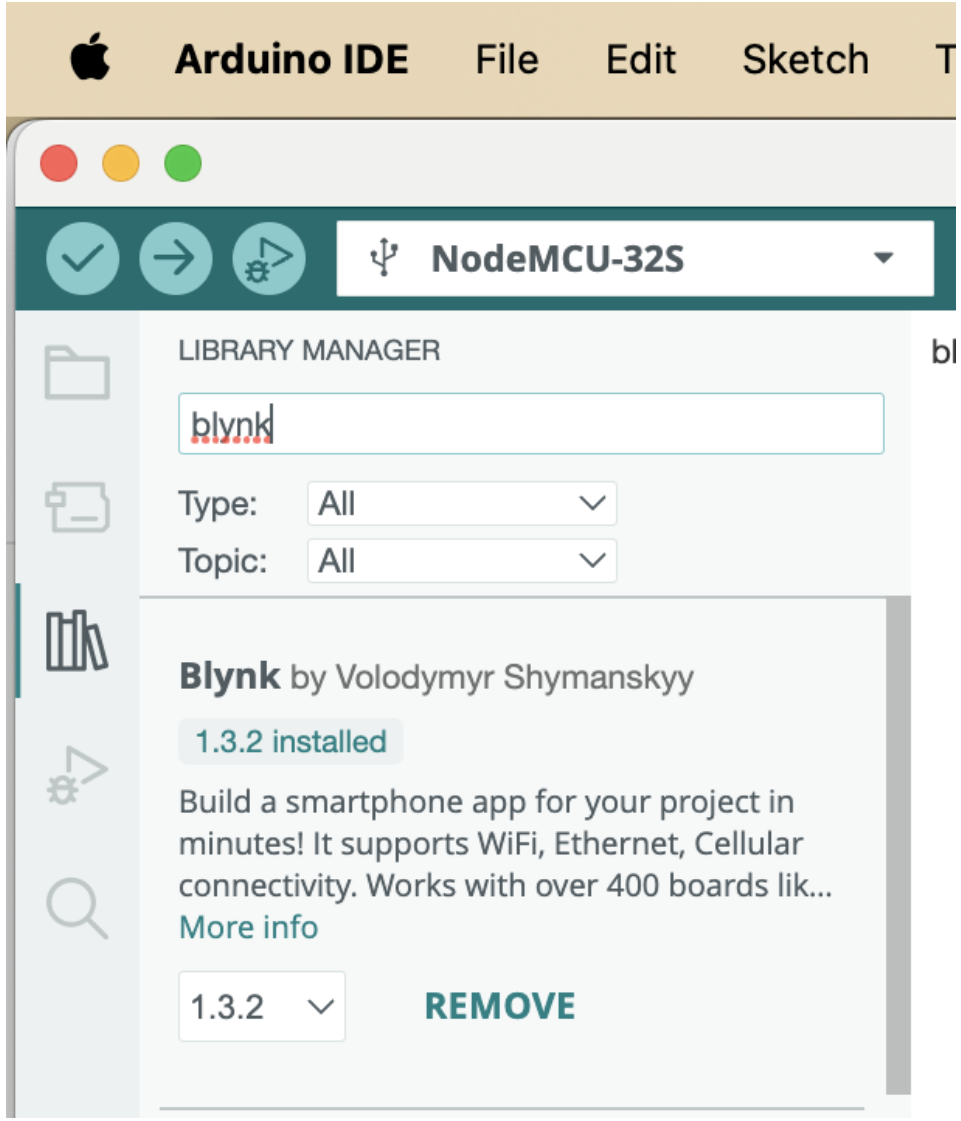
Konsep

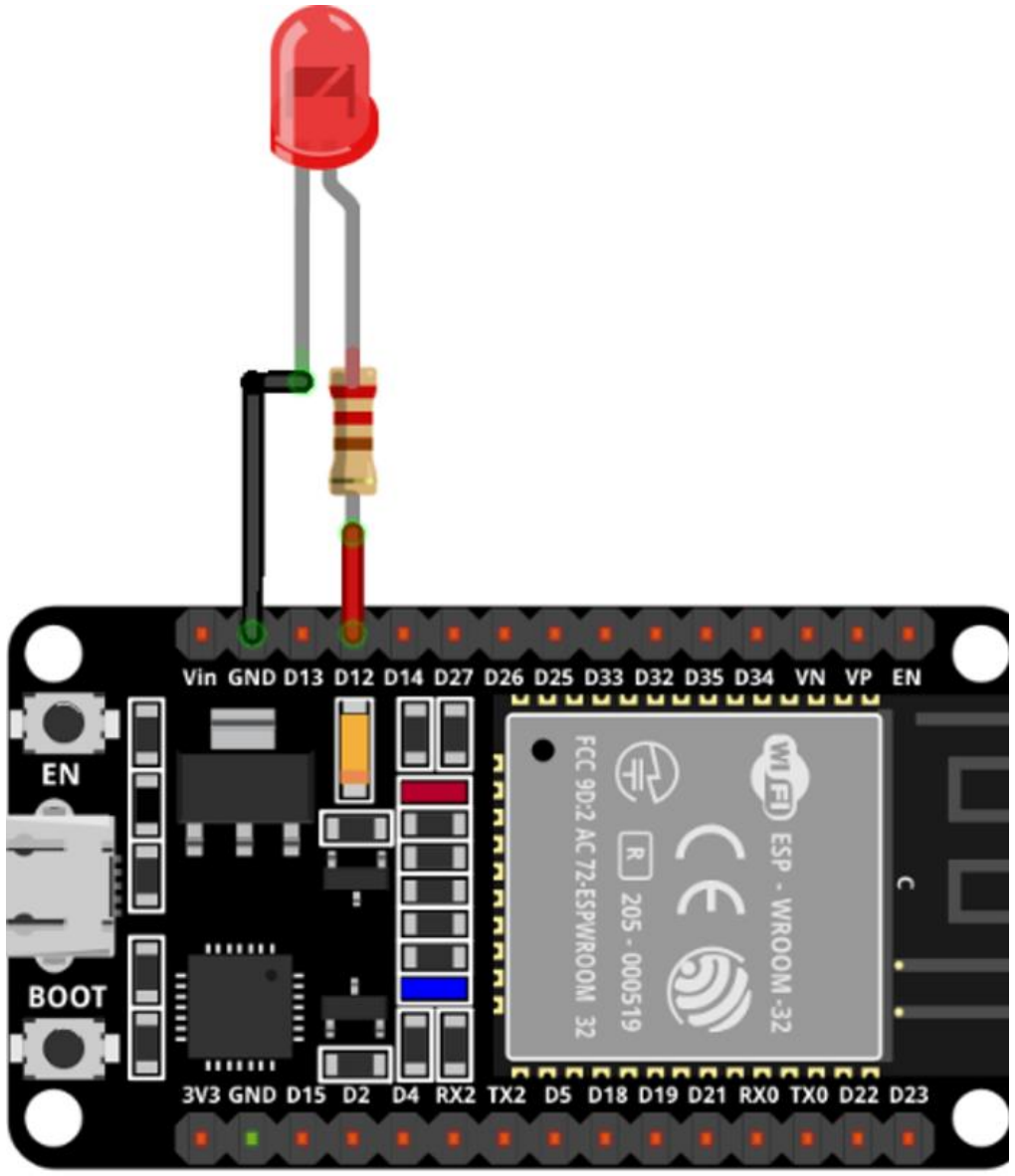
- Blynk adalah platform yang menyediakan alat dan infrastruktur untuk membuat aplikasi mobile yang mengontrol dan memantau perangkat IoT.
- Blynk juga mendukung berbagai perangkat IoT populer, mempermudah pengguna untuk mengembangkan solusi IoT yang fleksibel, intuitif, dan terhubung.



Blynk

- Buka Arduino IDE dan lakukan instalasi library blink **dari Volodymr**





Skema Rangkaian LED dan ESP32

- Lakukan konektivitas pada rangkaian yang telah ditetapkan

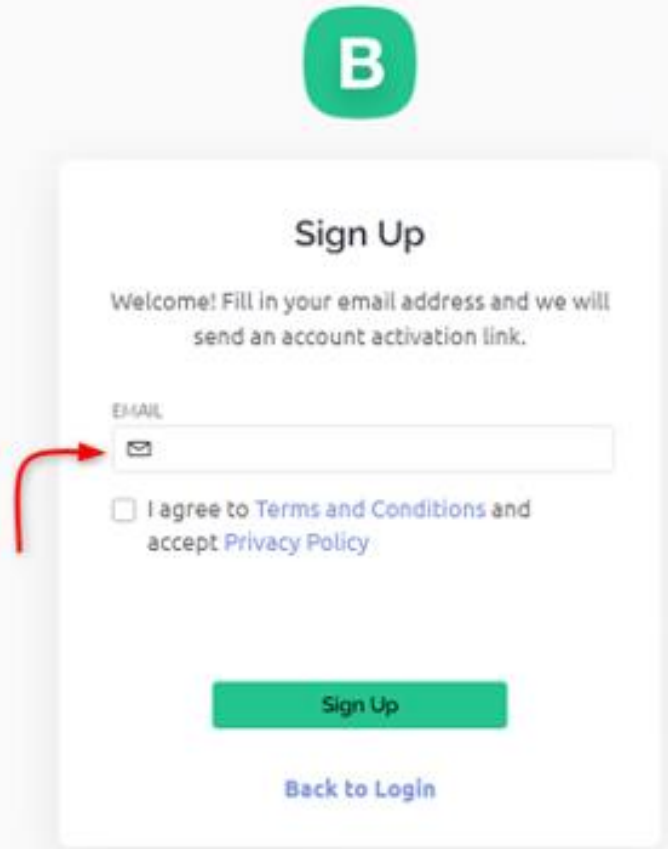
Kode Program

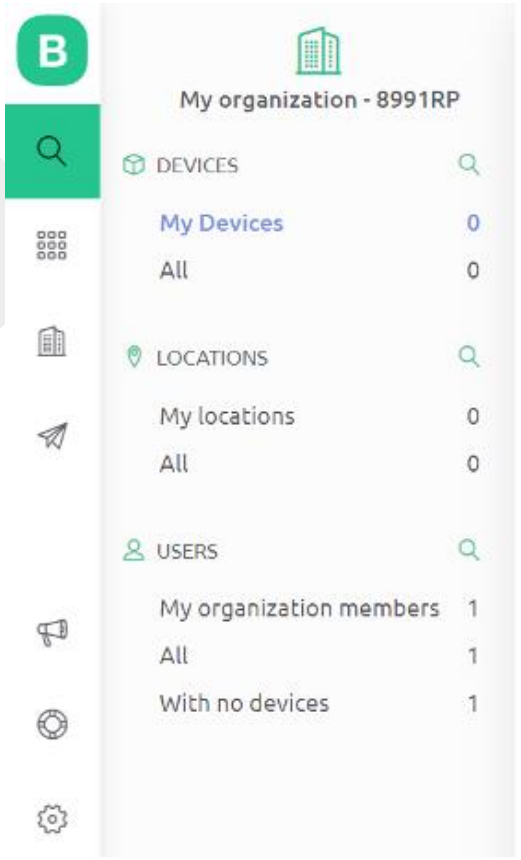
- Setelah selesai terhubung, maka upload kode program berikut ke board ESP32
- Blynk template ID, name, auth token
- Library
- Koneksi WIFI

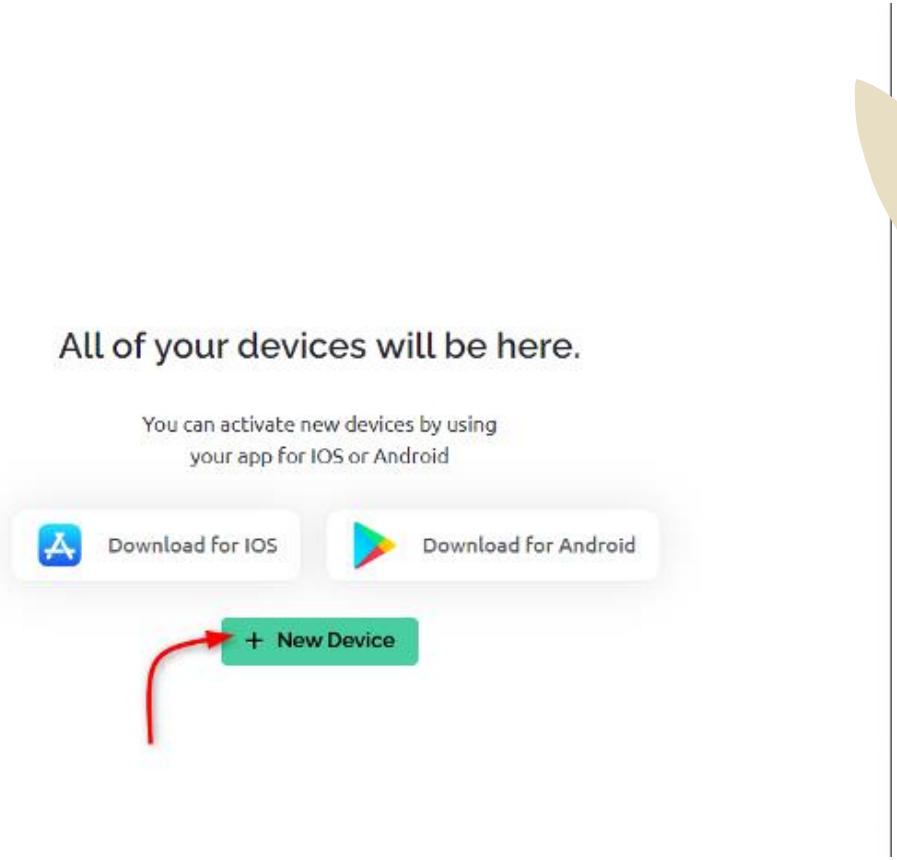
```
#define BLYNK_TEMPLATE_ID "TMPL6W-NhriSs"
#define BLYNK_TEMPLATE_NAME "LED Blink"
#define BLYNK_AUTH_TOKEN "

#define BLYNK_PRINT Serial  /*  include Blynk Serial
#include <WiFi.h>    /*ESP32 WiFi Library*/
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char auth[] = BLYNK_AUTH_TOKEN;
//Enter your WIFI SSID and password
char ssid[] = "          ";
char pass[] = "          ";

void setup(){
  Serial.begin(9600);  /*Baud rate for serial commun
  Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
}

void loop(){
  Blynk.run();
}
```

Tahap 1

- Buku link blynk.cloud. Lakukan signup atau login dengan akun yang sudah didadftarkan.



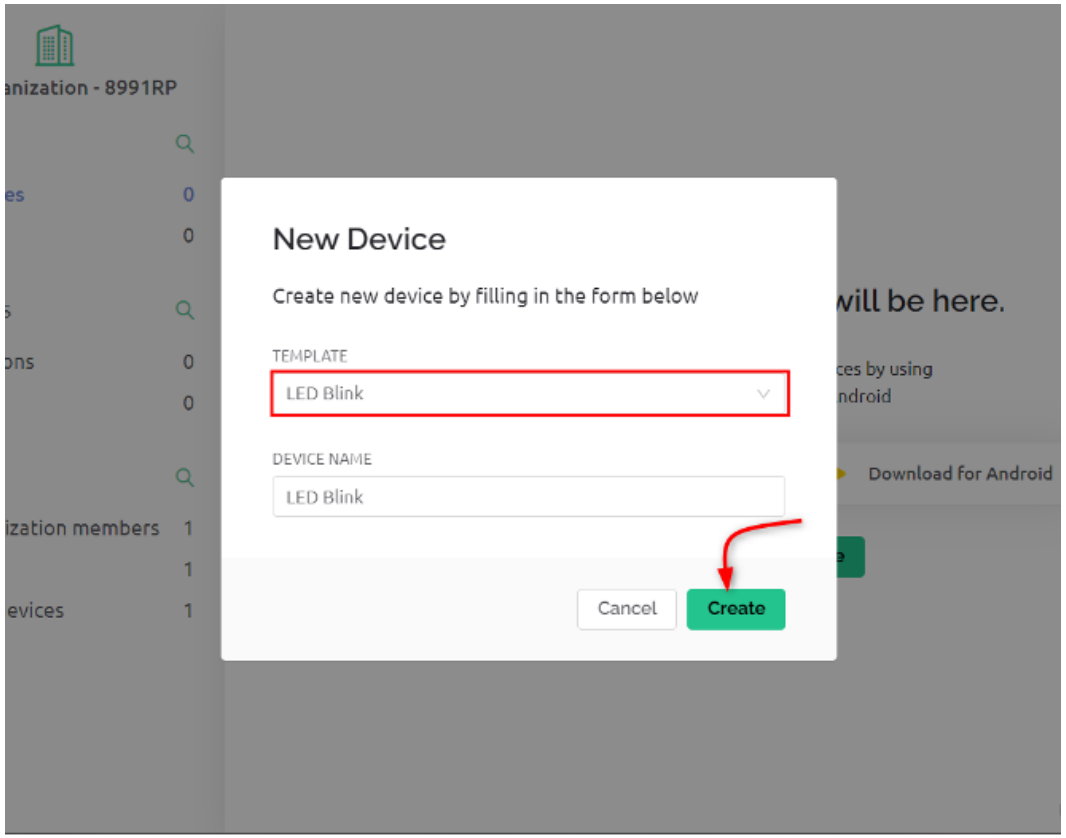




Tahap 2:

setelah login ke Blynk, buka perangkat baru (New Device) .

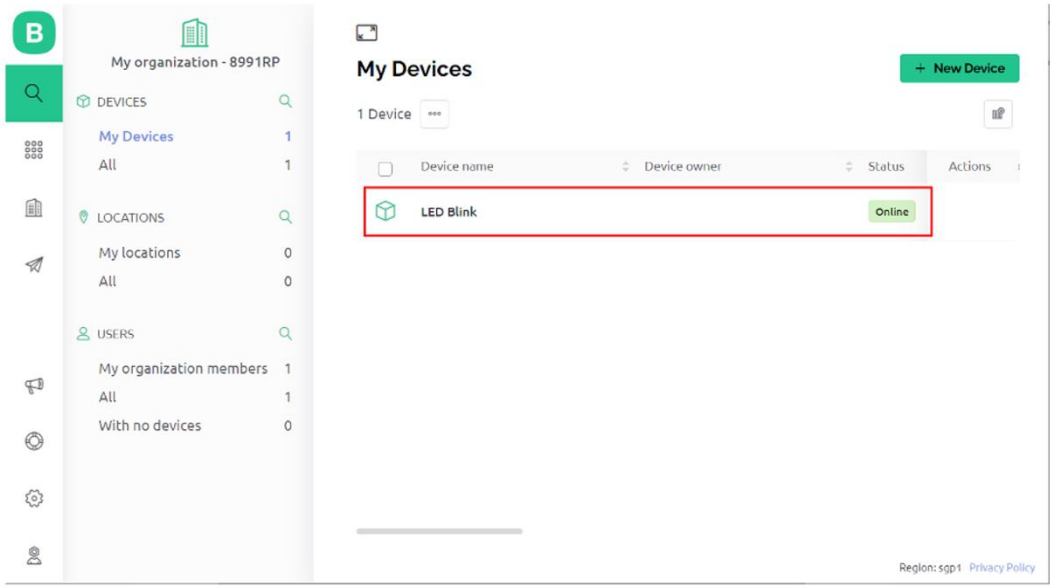
Tahap 3:



- **Tahap 3:** Namakan perangkat yang nanti nya akan dibuatkan GUI nya untuk control LED. Disini dinamakan LED Blink.

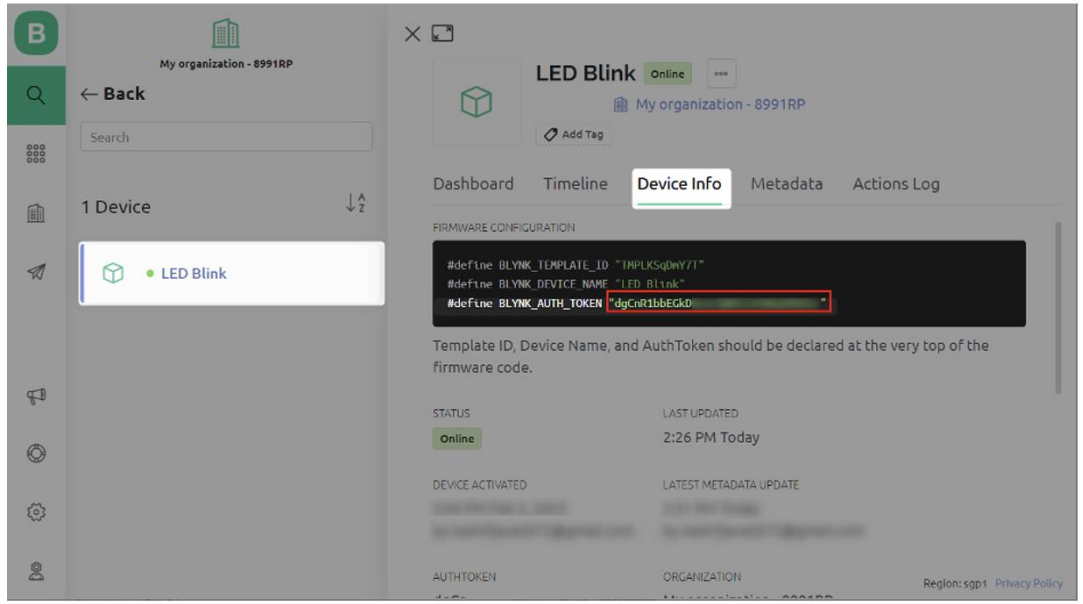
Tahap 4:

- Perangkat baru dengan nama LED Blynk sudah berhasil dibuat.



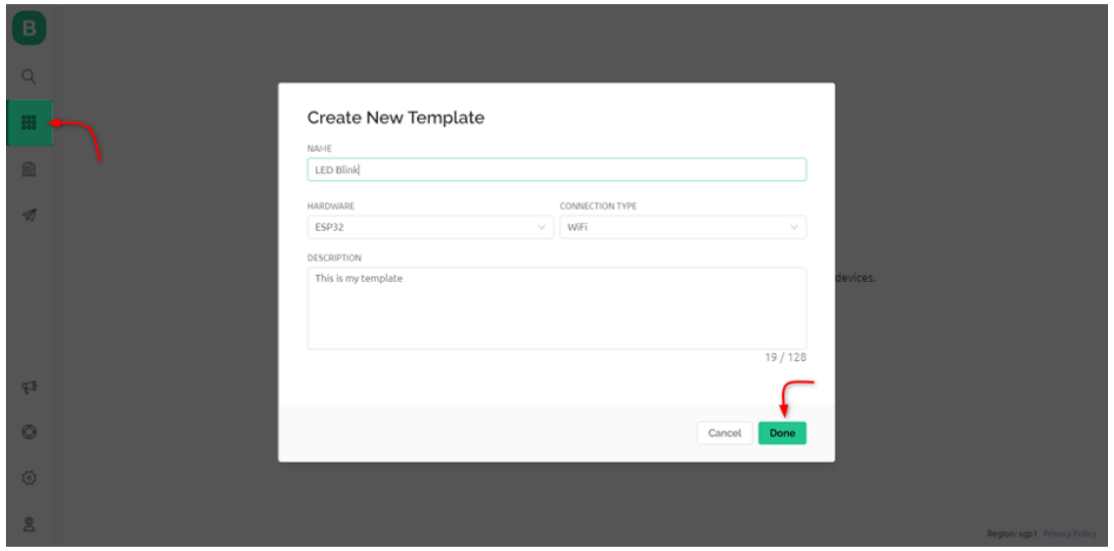
Tahap 5

- Pada bagian device info, maka didapatkan informasi terkait authentication token yang akan digunakan pada kode program di Arduino IDE



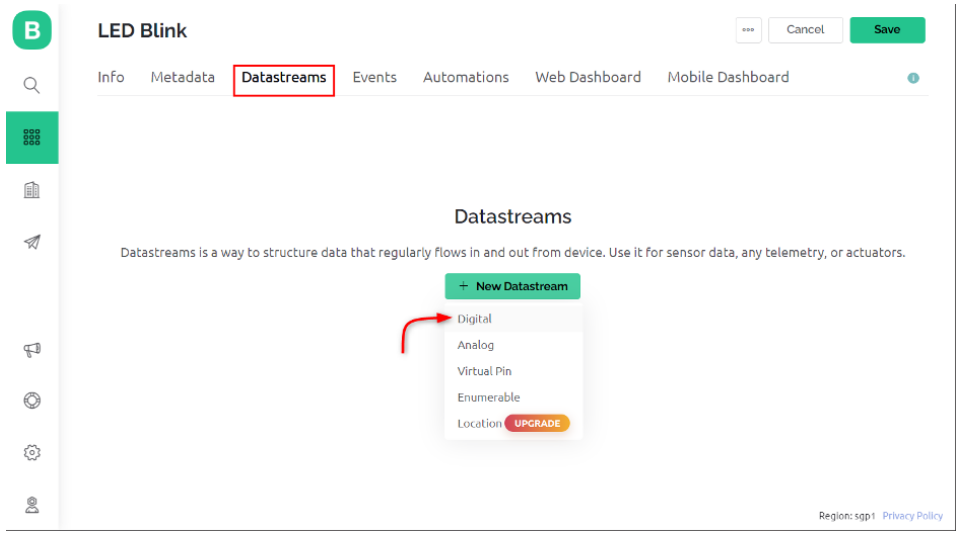
Tahap 6

- Selanjutnya buka template baru. Pada tahap ini dapat dipilih jenis hardware dan tipe koneksi. Jika sudah selesai diisikan maka klik **Done** untuk menyimpan konfigurasi.



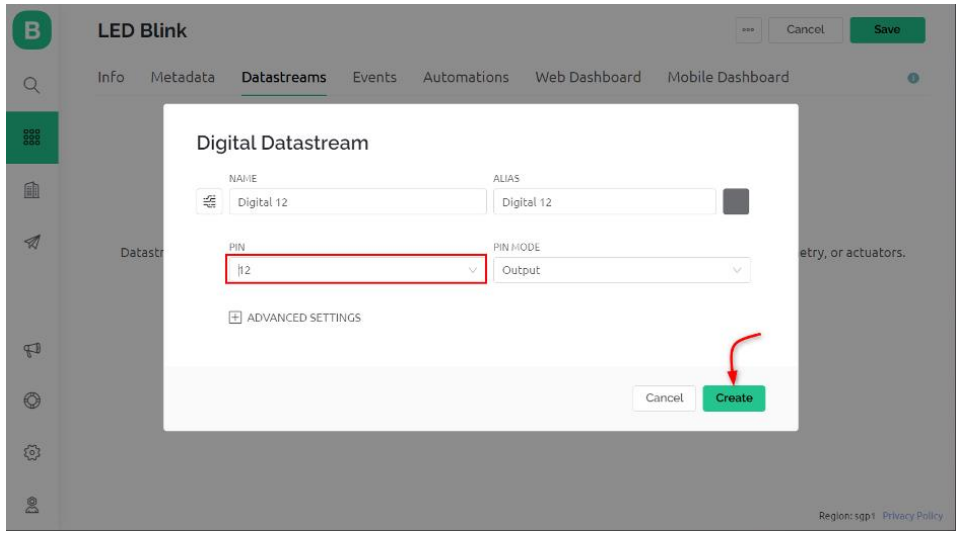
Tahap 7

- Setelah template baru dibuat, kita dapat menambahkan ***datastream*** di proyek kita. Dengan menggunakan ***datastream*** ini, makan dimungkinkan untuk dapat mengontrol pin ESP32 mana pun. Karena kita perlu mengontrol LED maka kita akan menggunakan pin digital untuk aliran data:



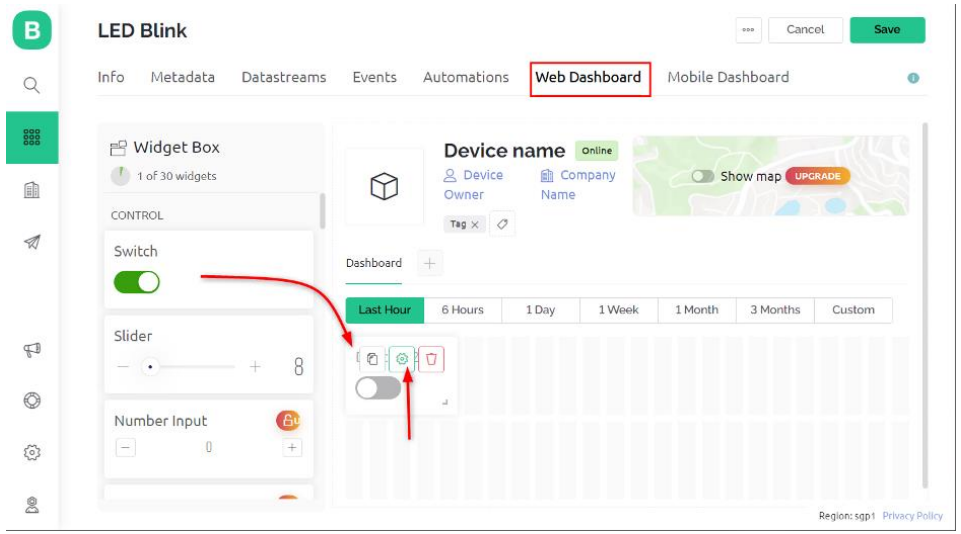
Tahap 8

- Sekarang kita pilih pin dimana LED akan terhubung. Disini kita menggunakan pin D12 pada ESP32 dan kita konfigurasi terkait pin mode nya adalah output.



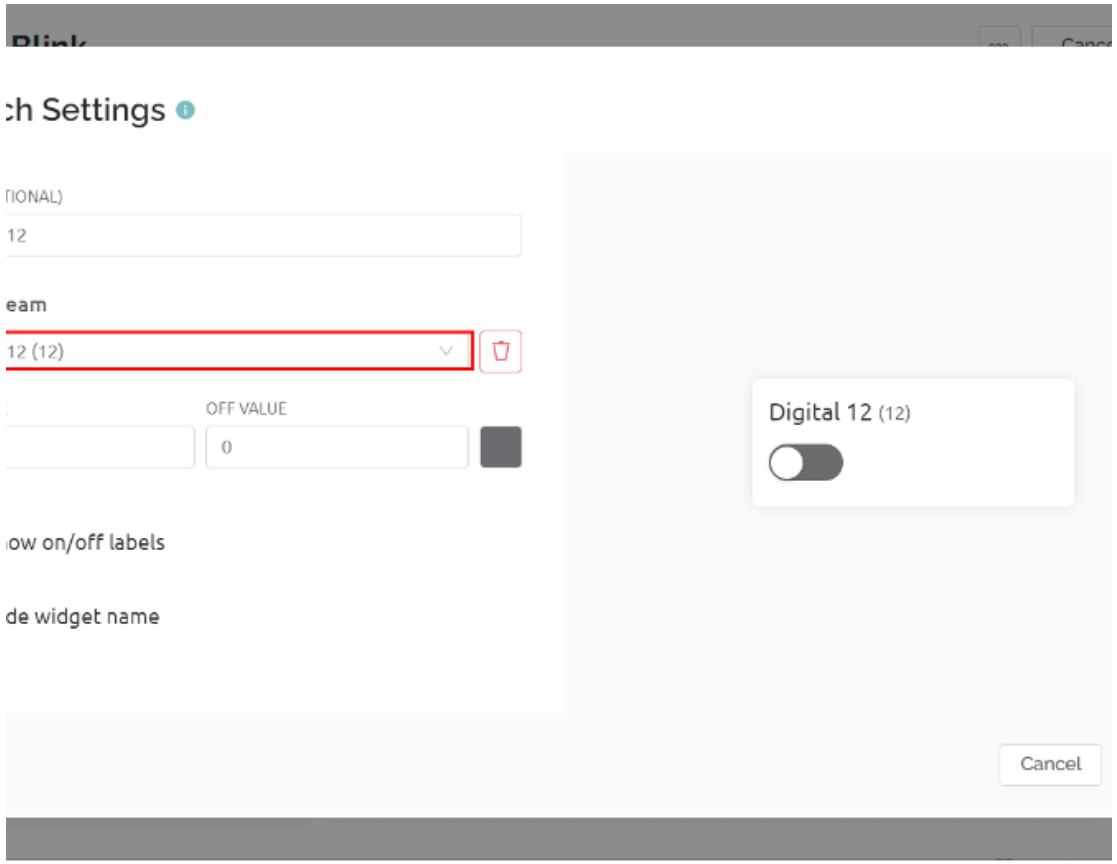
Tahap 9

- Untuk mendesain dasbord, kita dapat membuka menu Dasboard Web. Lakukan ***drag and drop*** switch baru kedalam datastream:



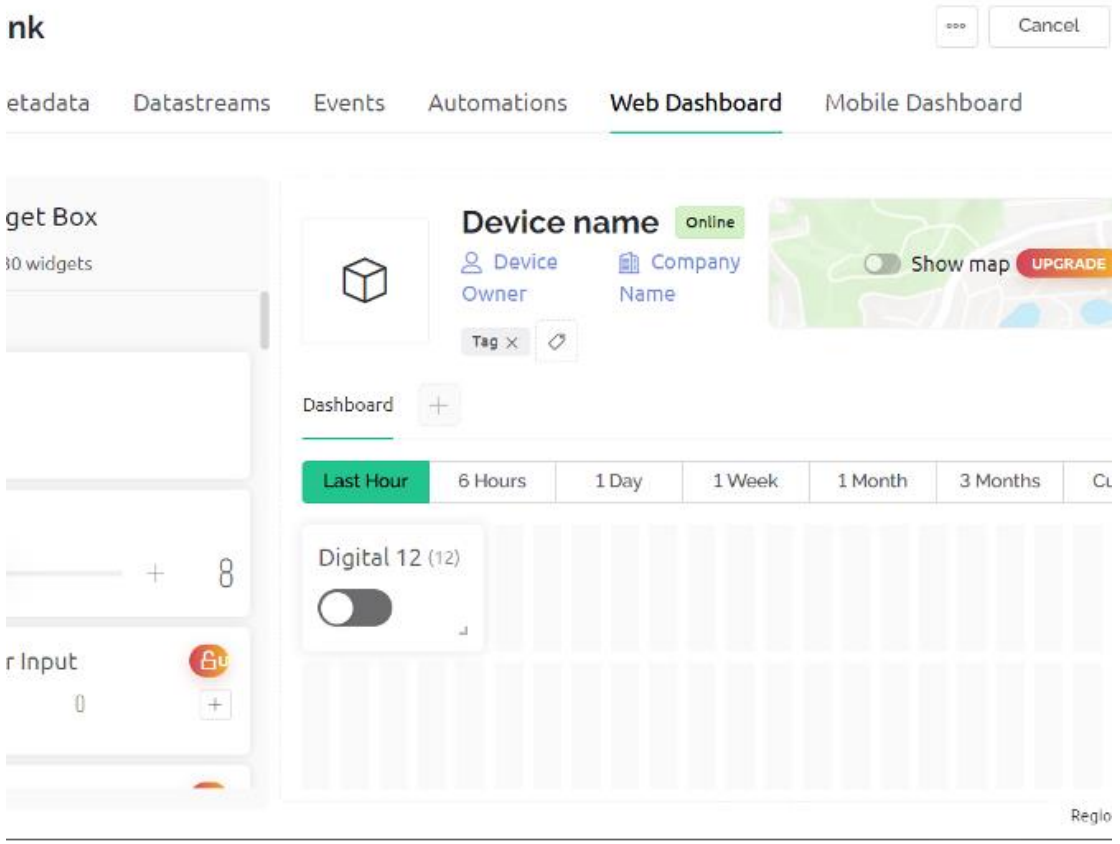
Tahap 10

- Setelah menambahkan tombol baru sekarang pilih opsi pengaturan. Disini lakukan definisi terhadap sumber DataStream sebagai pin Digital 12 dan tetapkan nilai ON ke 1 dan nilai OFF ke 0:



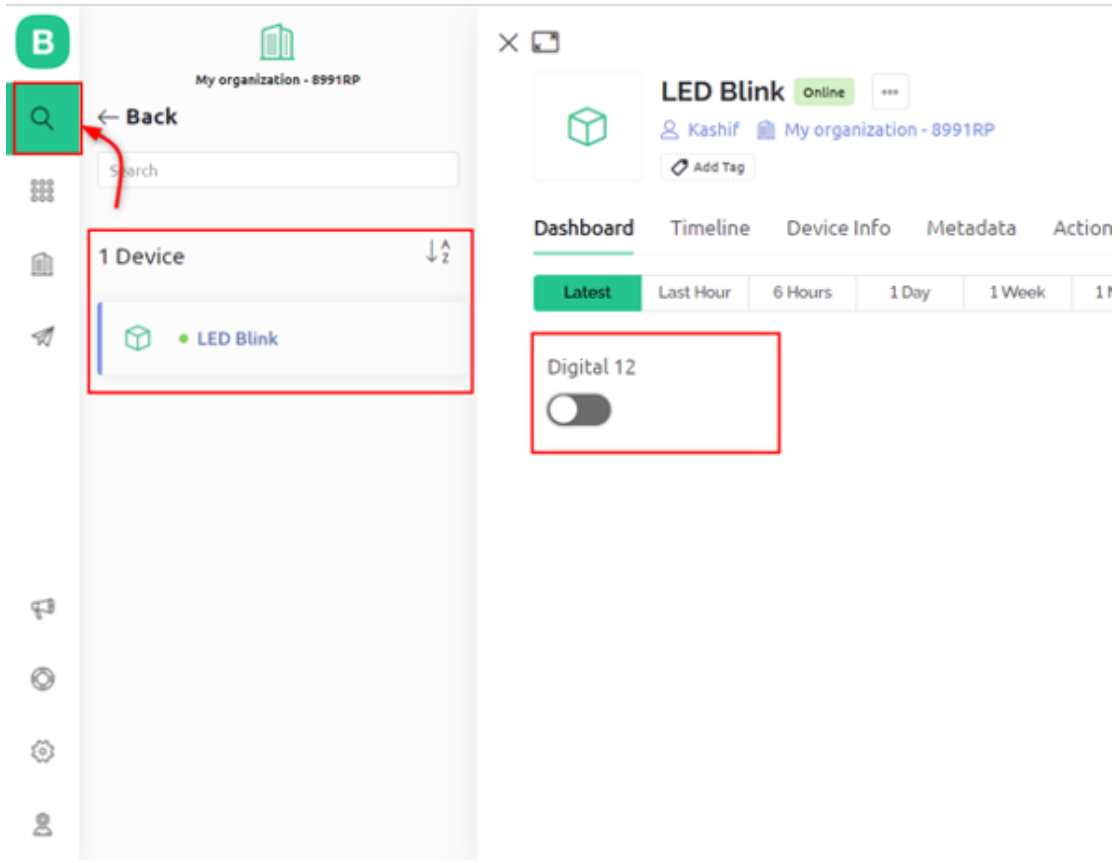
Tahap 11

- setelah menambahkan button baru selanjutnya klik **Save** untuk menyimpan konfigurasi. Dengan menggunakan metode ini, kita dapat menambahkan switch apa pun yang sesuai dengan pin ESP32 tertentu:



Tahap 12

- Sekarang kita dapat control LED menggunakan Blynk, buka dashboard. Disini kita dapat melihat **switch** untuk mengontrol LED yang terhubung pada pin D12.

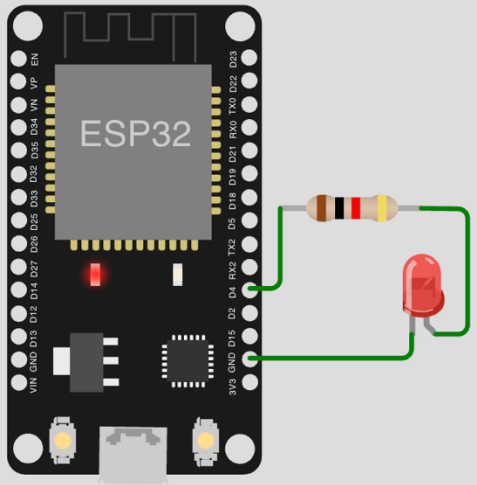


sketch.ino | diagram.json | libraries.txt | Library Manager

```
#define BLYNK_TEMPLATE_ID "TMPL6nT1uVlX5"
#define BLYNK_TEMPLATE_NAME "LCD ESP"
#define BLYNK_AUTH_TOKEN "0xFjhR1BkWEf8wHFrtBKC4s3dx2EM7CI"
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "Wokwi-GUEST"; //nama hotspot yang digunakan
char pass[] = ""; //password hotspot yang digunakan

BLYNK_WRITE(V0)
{
  int pinValue = param.asInt(); // assigning incoming value from pin V1 to a variabl
  if (pinValue == 1)
  {
    digitalWrite(4, HIGH);
  }
  else {
    digitalWrite(4, LOW);
  }
}
void setup()
{
  Serial.begin(115200);
```

Simulation

02:24.956 100%




```
[78524] Connecting to blynk.cloud:80
[88529] Connecting to blynk.cloud:8080
```

Contoh

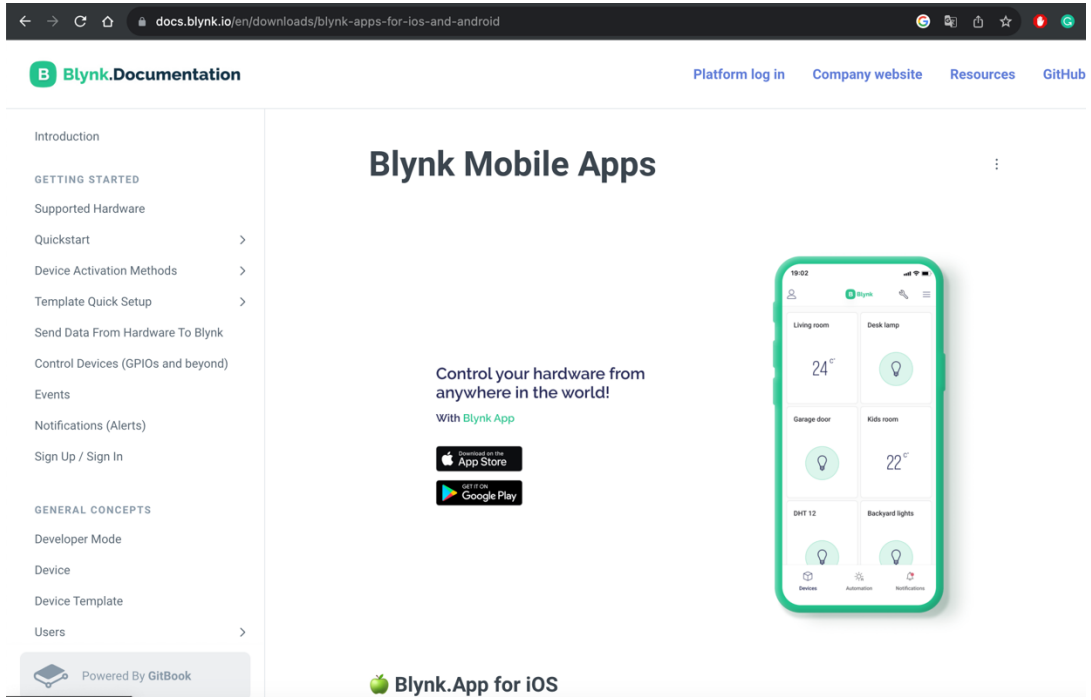
https://wokwi.com/projects/425388840359754753

Success (Serial Monitor)

```
[1383] Connecting to Wokwi-GUEST
[7717] Connected to WiFi
[7717] IP: 10.10.0.2
[7718]

    ___  __           __
   / _ )/ /_ _____  / /__
  / _  / / // / _ \/  '_/
 /____/_/\_, /_//_/_/\_\
        /___/ v1.3.2 on ESP32

 #StandWithUkraine    https://bit.ly/swua


[7728] Connecting to blynk.cloud:80
[8659] Redirecting to sgp1.blynk.cloud:80
[8661] Connecting to sgp1.blynk.cloud:80
[9517] Ready (ping: 196ms).
```

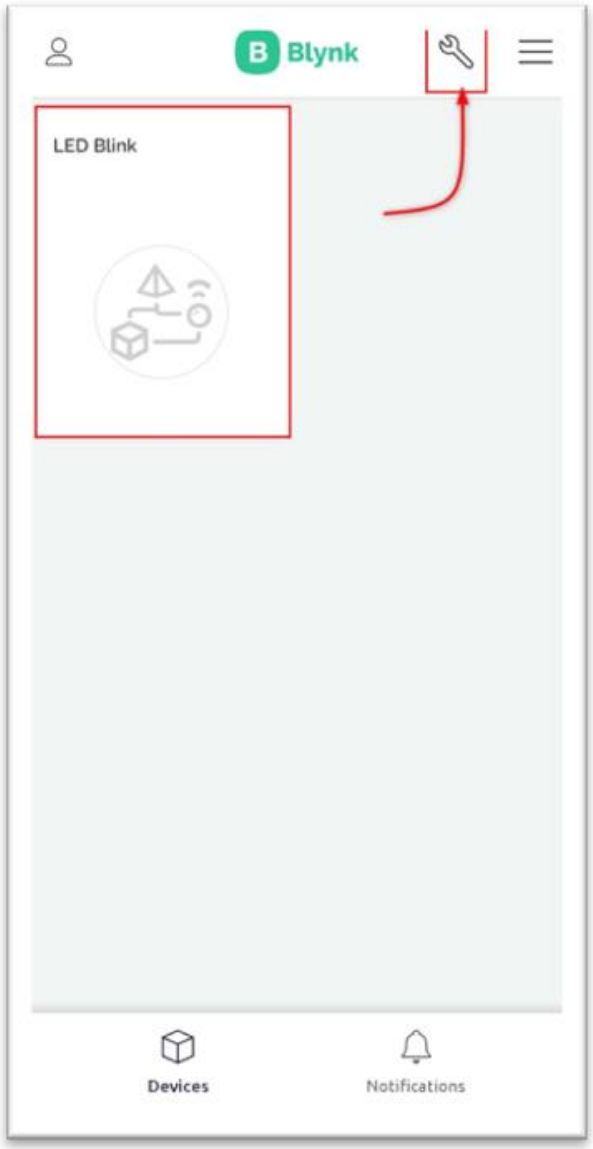
Blynk Mobile Apps

- Blynk menyediakan platform intuitif untuk membuat aplikasi mobile yang mengontrol perangkat IoT.
- Melalui antarmuka pengguna yang disesuaikan, pengguna dapat membuat tombol, slider, dan tampilan data untuk berinteraksi dengan perangkat IoT mereka.



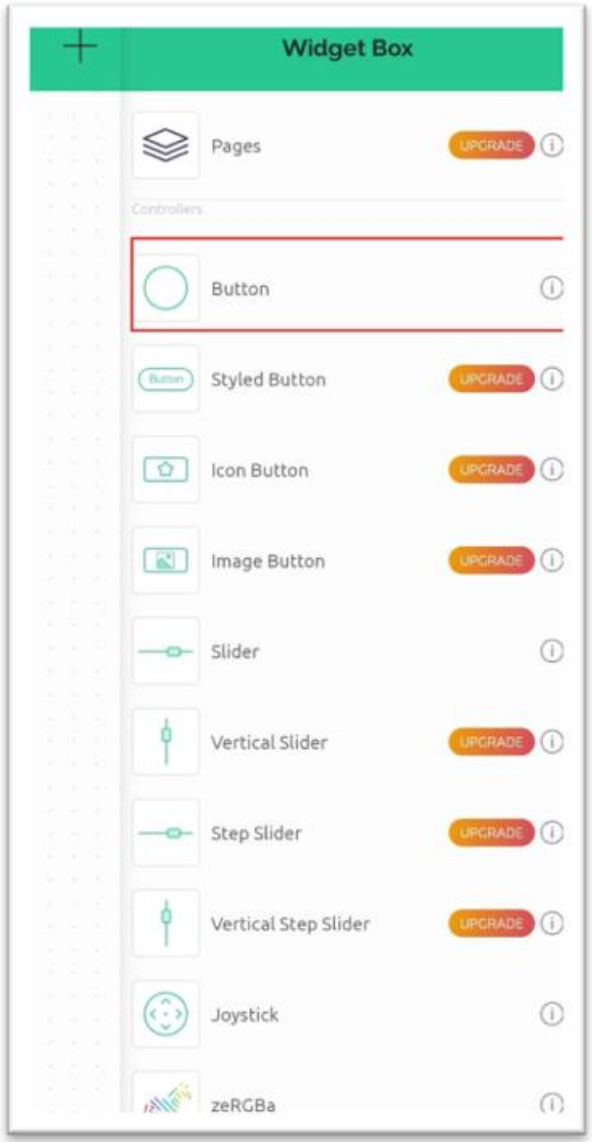
Mobile Development

- Pengembangan via web dashboard dan via mobile application adalah sama, yang perlu diperhatikan bahwasanya Blynk Web dan Mobile Application dibuka dengan menggunakan akun yang sama.
- Jika Anda masuk dengan akun yang sama, Anda akan melihat proyek LED Blink di dalam aplikasi Blynk IoT. Buka mode pengembang menggunakan ikon pengaturan di sudut kanan atas:



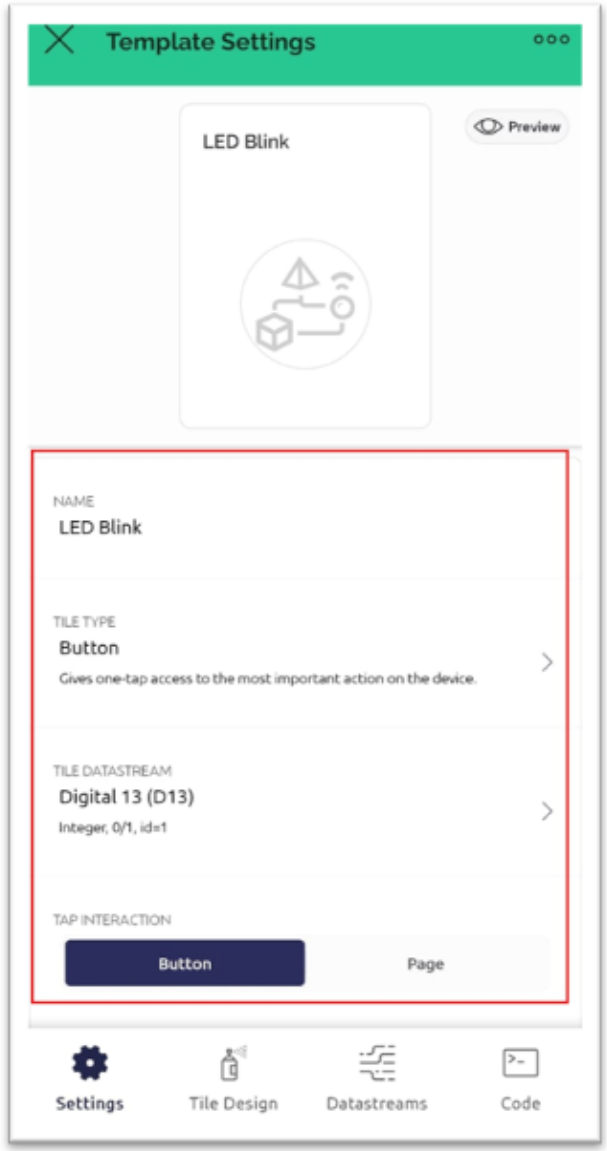
Tahap 1

- Di sini kita dapat membuat tombol baru untuk setiap pin di ESP32 atau menambahkan yang baru:



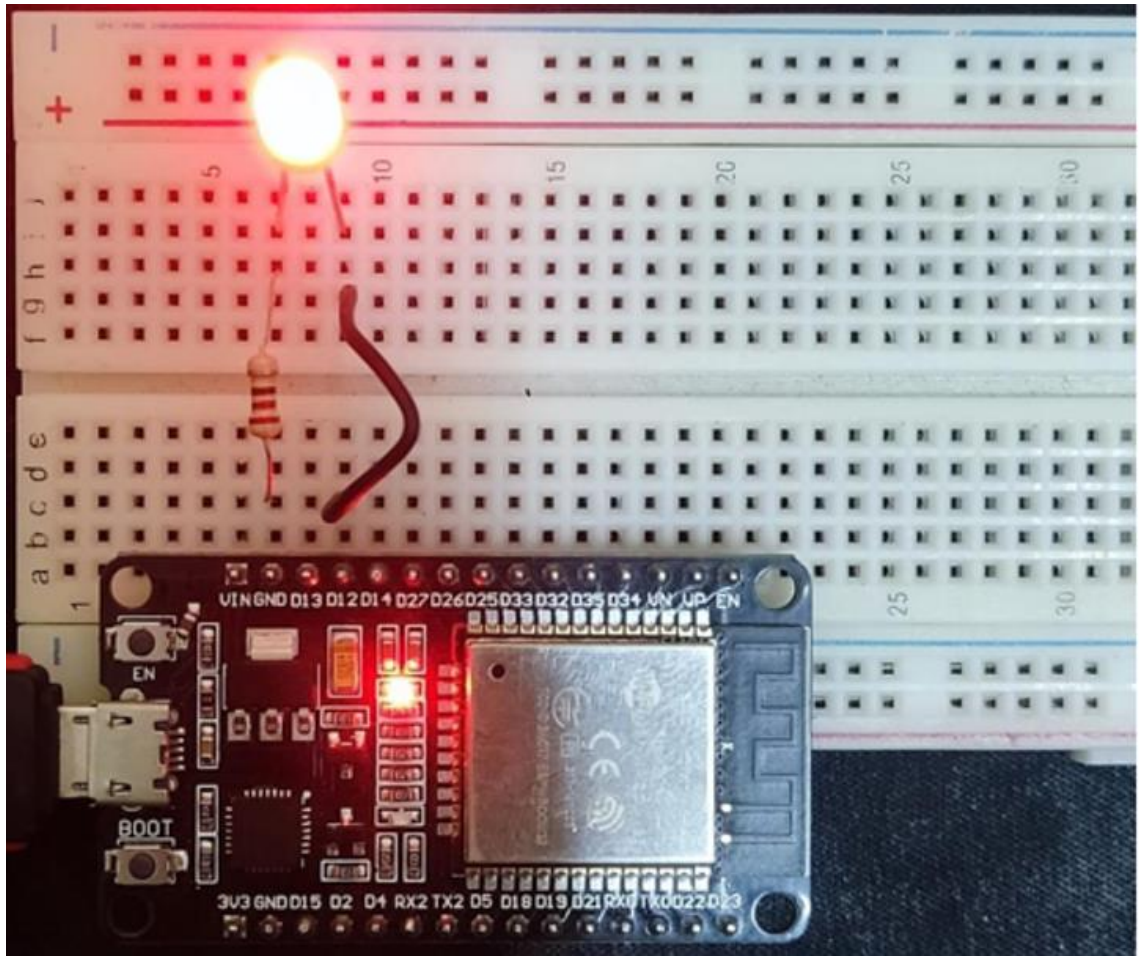
Tahap 2

- Kita juga dapat menyesuaikan pengaturan di dalam template seperti nomor pin atau mengganti mode kerja atau mengatur DataStream baru untuk pin:



Tahap 3

Setelah semua pengaturan selesai, kita dapat melihat LED menyala terhubung ke pin D4.



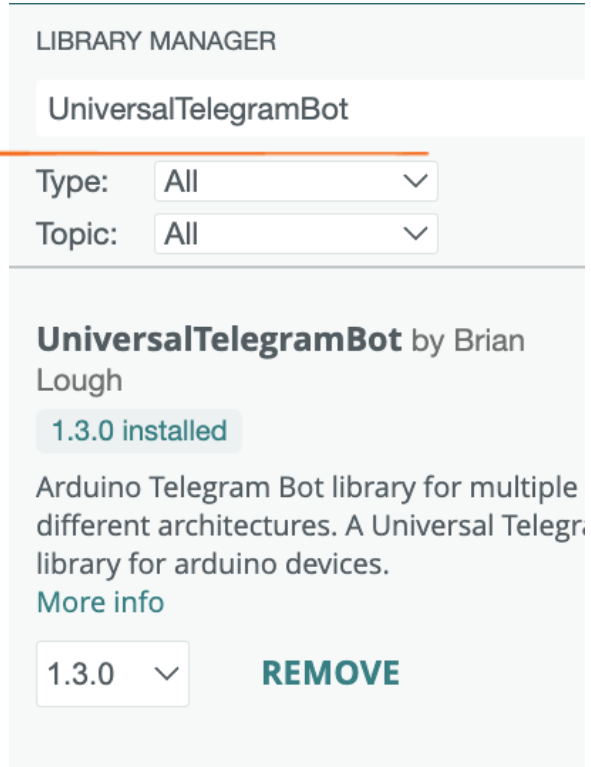
BoT Telegram

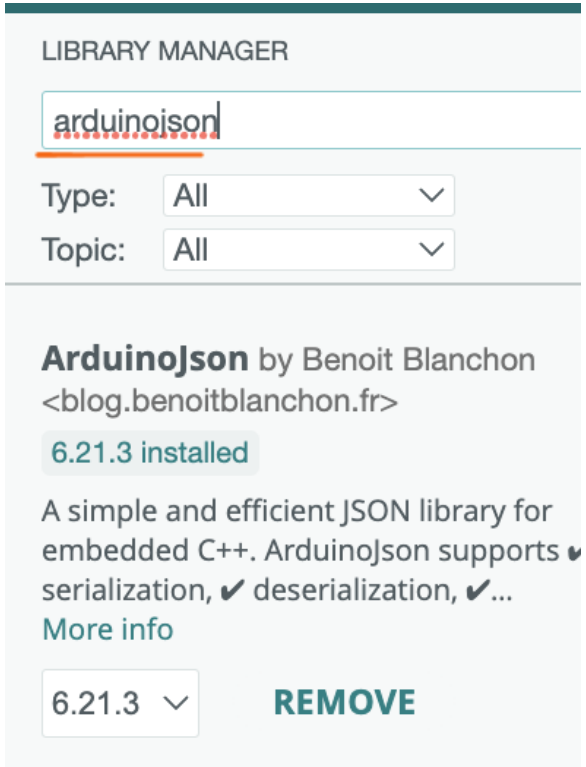
Karakteristik BoT Telegram

1. **Komunikasi dua arah**
2. **Perintah dan Tanggapan**
3. **Notifikasi dan Pemberitahuan**

Library Telegram BoT

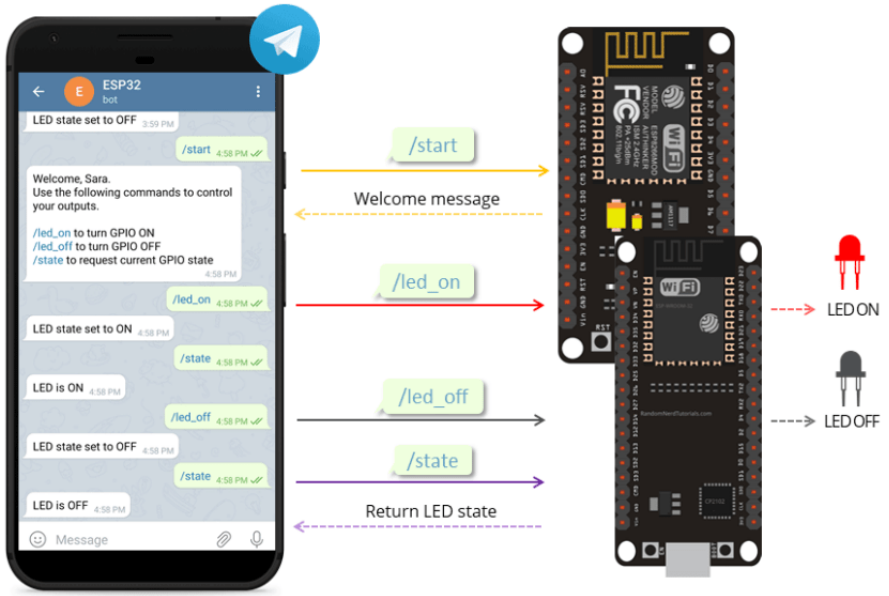
- Persiapan Awal
- 1. Library Universal Telegram Bot
- 2. Library Arduino JSON





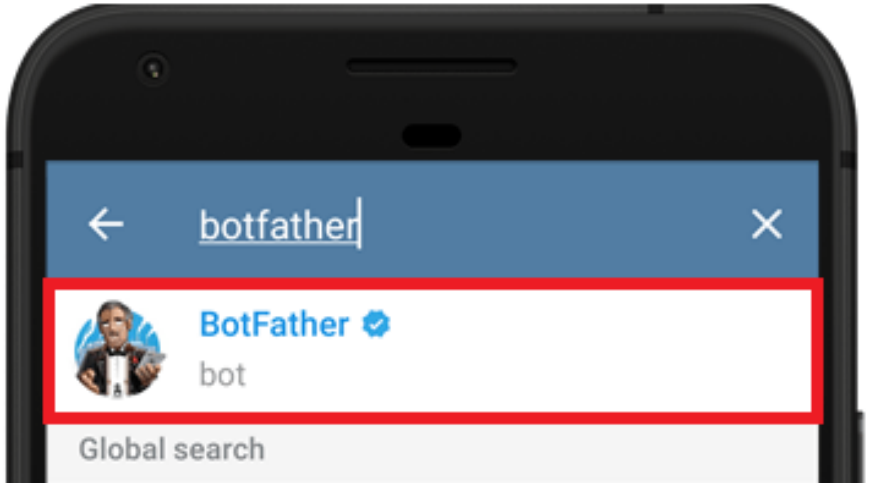
Skema

- Projek ini untuk memberikan command atau instruksi dari bot telegram kepada ESP32 untuk menghidupkan atau menyalakan lampu dan memberikan kondisi/state dari lampu



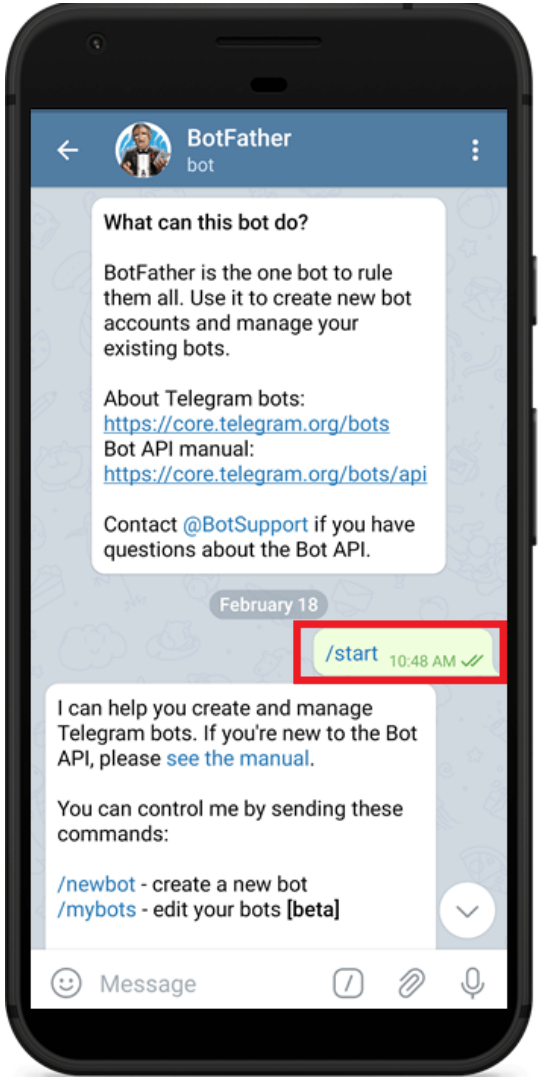
Tahap 1

Buka Telegram dan ikuti langkah-langkah berikut untuk membuat Bot Telegram. Pertama, cari "botfather" dan klik BotFather seperti yang ditunjukkan di bawah ini. Atau buka tautan ini t.me/botfather di smartphone Anda.



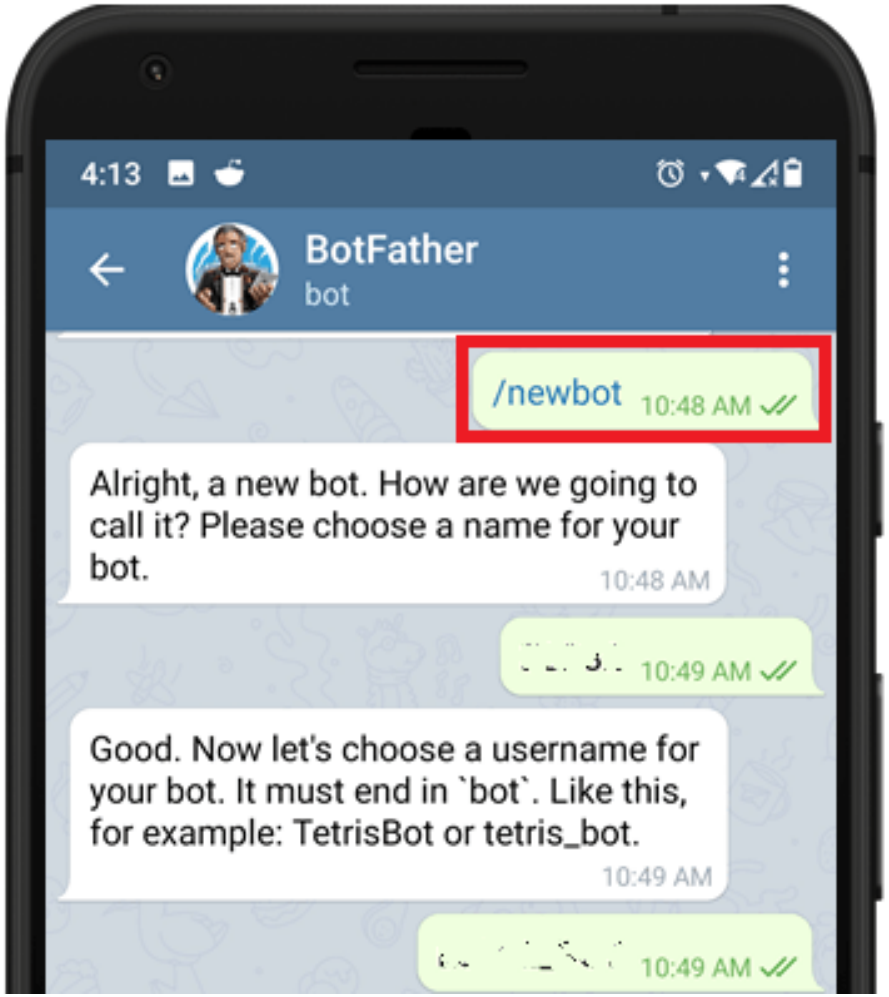
Tahap 2

Setelah diketikaan maka selanjutnya Ketika /start untuk memulai bot.



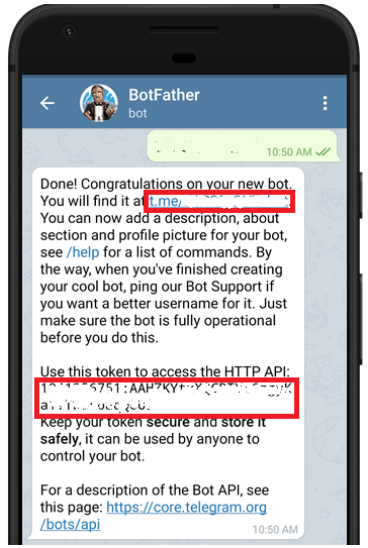
Tahap 3

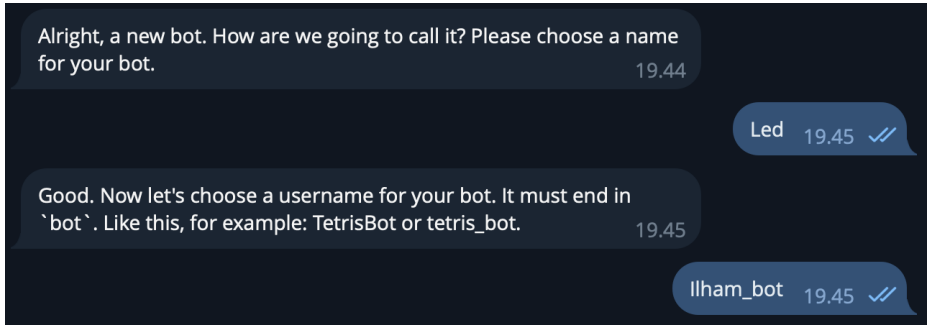
- Ketik /newbot dan ikuti instruksi untuk membuat bot Anda. Beri nama dan nama pengguna. Tuliskan nama dan *username*



Tahap 4

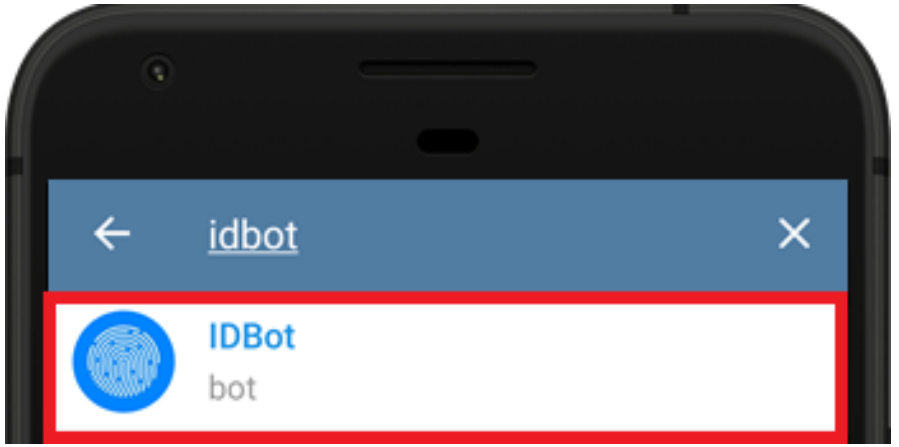
Jika bot telah berhasil dibuat, maka kita akan menerima pesan berisi tautan untuk mengakses bot dan token bot. Simpan token bot karena nantinya akan diperlukan agar ESP32/ESP8266 dapat berinteraksi dengan bot.





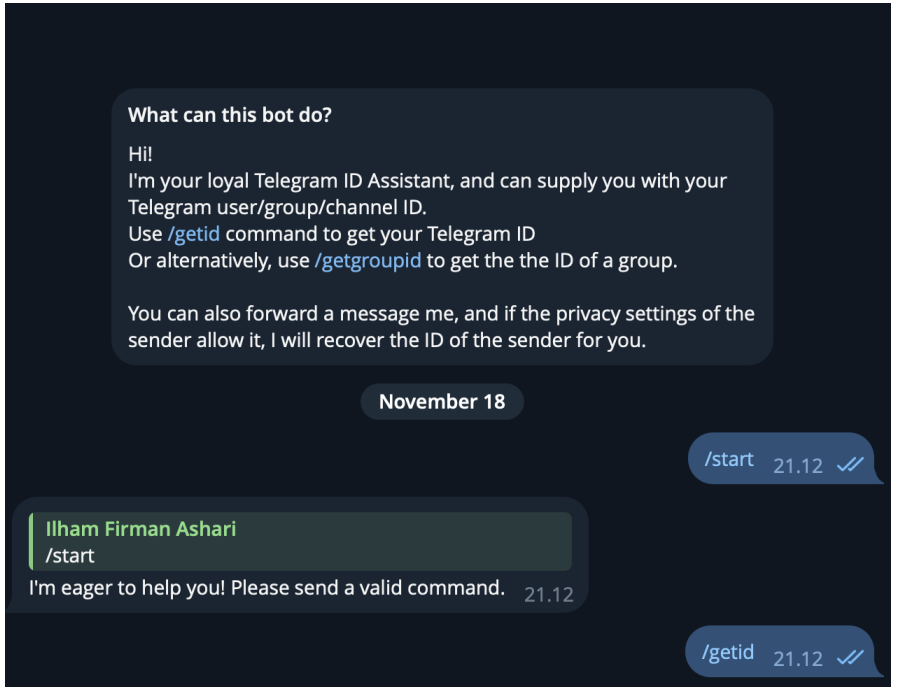
Tahap 5

Di akun Telegram Anda, cari "IDBot" atau buka tautan ini t.me/botfather di smartphone Anda.



ID BOTChat

- Klik /start
- Dan klik /getid untuk mendapatkan id



Kode Program

- Inisialisasi variable dan libraray
- BOT Token dan Chat ID

bot_tele_f.ino

```
#ifdef ESP32
  #include <WiFi.h>
#else
  #include <ESP8266WiFi.h>
#endif
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>
const char* ssid = "IFA-Homes";
const char* password = "berkah143";
// Initialize Telegram BOT
#define BOTtoken "6746225633:AAEjEHtIlS2t1_1RwdotFKqXcvJqyRCcVFg"
#define CHAT_ID "171936607"
#ifdef ESP8266
  X509List cert(TELEGRAM_CERTIFICATE_ROOT);
#endif
WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);
// Checks for new messages every 1 second.
int botRequestDelay = 1000;
unsigned long lastTimeBotRan;
const int ledPin = 2;
bool ledState = LOW;

```

Kode Program 2

- Fungsi: HandleNewMessage (atur informasi di bot)
- atur text bot untuk:
- -led_on
- -led_off
- -state

bot_tele_f.ino

```
25  void handleNewMessages(int numNewMessages) {
26    Serial.println("handleNewMessages");
27    Serial.println(String(numNewMessages));
28
29    for (int i=0; i<numNewMessages; i++) {
30      // Chat id of the requester
31      String chat_id = String(bot.messages[i].chat_id);
32      if (chat_id != CHAT_ID){
33        bot.sendMessage(chat_id, "Unauthorized user", "");
34        continue;
35      }
36      // Print the received message
37      String text = bot.messages[i].text;
38      Serial.println(text);
39      String from_name = bot.messages[i].from_name;
40      if (text == "/start") {
41        String welcome = "Welcome, " + from_name + ".\n";
42        welcome += "Use the following commands to control your outputs.\n\n";
43        welcome += "/led_on to turn GPIO ON \n";
44        welcome += "/led_off to turn GPIO OFF \n";
45        welcome += "/state to request current GPIO state \n";
46        bot.sendMessage(chat_id, welcome, "");
47      }
48
49      if (text == "/led_on") {
50        bot.sendMessage(chat_id, "LED state set to ON", "");
51        ledState = HIGH;
52        digitalWrite(ledPin, ledState);
53      }
54
55      if (text == "/led_off") {
56        bot.sendMessage(chat_id, "LED state set to OFF", "");
57        ledState = LOW;
58        digitalWrite(ledPin, ledState);
59      }
60
61      if (text == "/state") {
62        if (digitalRead(ledPin)){
63          bot.sendMessage(chat_id, "LED is ON", "");
64        }
65        else{
66          bot.sendMessage(chat_id, "LED is OFF", "");
67        }
68      }
69    }
70  }
```

Setup

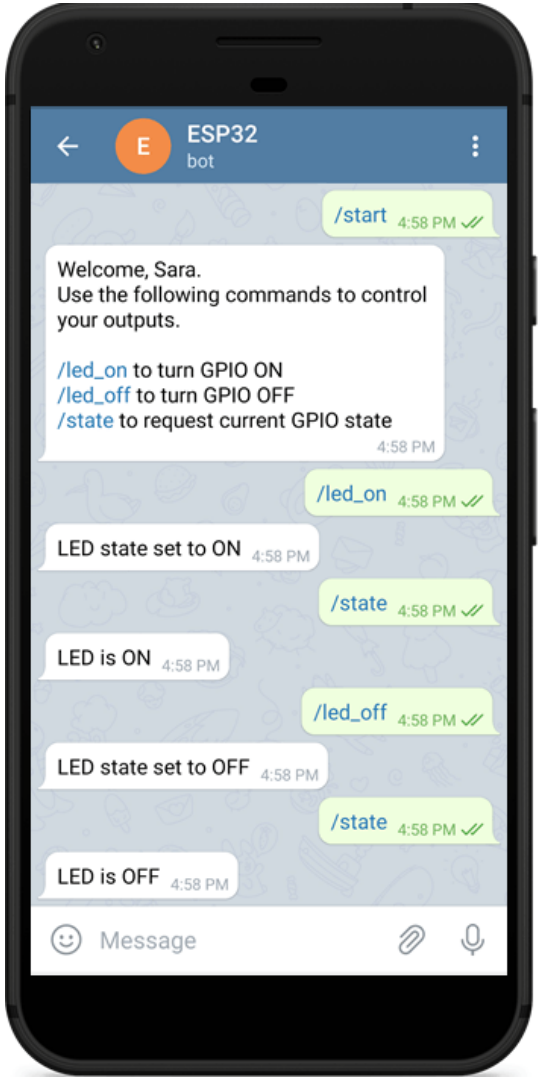
- Koneksi ke WIFI
- Dan void loop

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

  #ifdef ESP8266
    configTime(0, 0, "pool.ntp.org");
    client.setTrustAnchors(&cert);
  #endif

  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, ledState);

  // Connect to Wi-Fi
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  #ifdef ESP32
    client.setCACert(TELEGRAM_CERTIFICATE_ROOT);
  #endif
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }
  // Print ESP32 Local IP Address
  Serial.println(WiFi.localIP());
}

void loop() {
  if (millis() > lastTimeBotRan + botRequestDelay)  {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

    while(numNewMessages) {
      Serial.println("got response");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }
    lastTimeBotRan = millis();
  }
}
```

Running

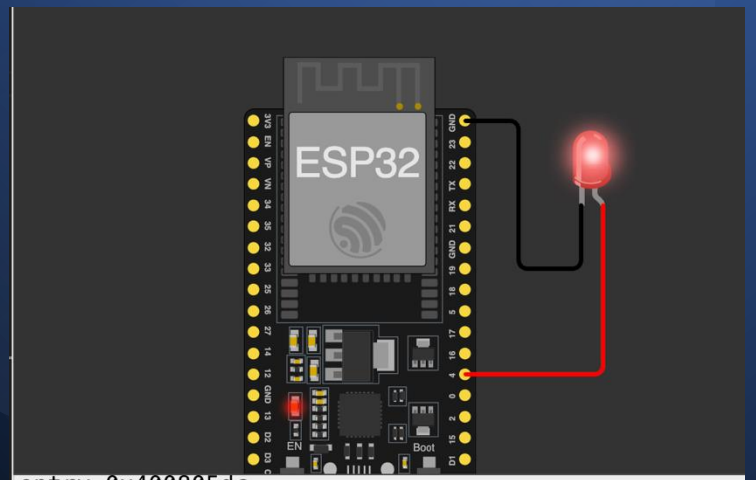
Buka akun Telegram Anda dan buka percakapan dengan bot Anda. Kirim perintah-perintah berikut dan lihat bot merespons:

1. /start menampilkan pesan selamat datang beserta perintah-perintah yang valid.
2. /led_on menyalakan lampu LED.
3. /led_off mematikan lampu LED.
4. /state meminta status terkini dari LED.



https://wokwi.com/projects/425439004471361537

Contoh



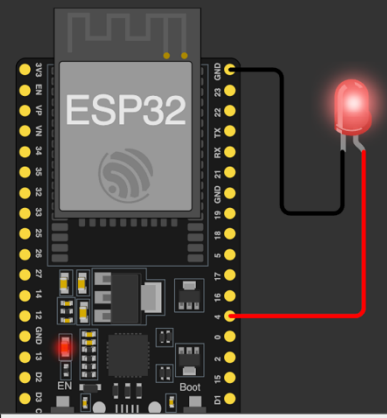

```
sketch.ino  diagram.json  libraries.txt  Library Manager
```

```
50  void ha
51    for (
56      if (chat_id != CHAT_ID) {
59      }
60
61      Serial.println("Pesan diterima: " + text);
62
63      if (text == "/start") {
64        String welcome = "Selamat datang!\n";
65        welcome += "Gunakan perintah berikut:\n";
66        welcome += "/on - Menyalakan LED\n";
67        welcome += "/off - Mematikan LED\n";
68        welcome += "/status - Melihat status LED\n";
69        bot.sendMessage(chat_id, welcome, "");
70      }
71      else if (text == "/on") {
72        digitalWrite(ledPin, HIGH);
73        ledState = HIGH;
74        bot.sendMessage(chat_id, "LED Menyala", "");
75      }
76      else if (text == "/off") {
77        digitalWrite(ledPin, LOW);
78        ledState = LOW;
79        bot.sendMessage(chat_id, "LED Mati", "");
80      }
81      else if (text == "/status") {
82        if (ledState) {
83          bot.sendMessage(chat_id, "LED dalam keadaan ON", "");
84        } else {
85          bot.sendMessage(chat_id, "LED dalam keadaan OFF", "");
86        }
87      }
88    }
89  }
90
```

pesan diter — Aa ab .* 1 of 1

Simulation

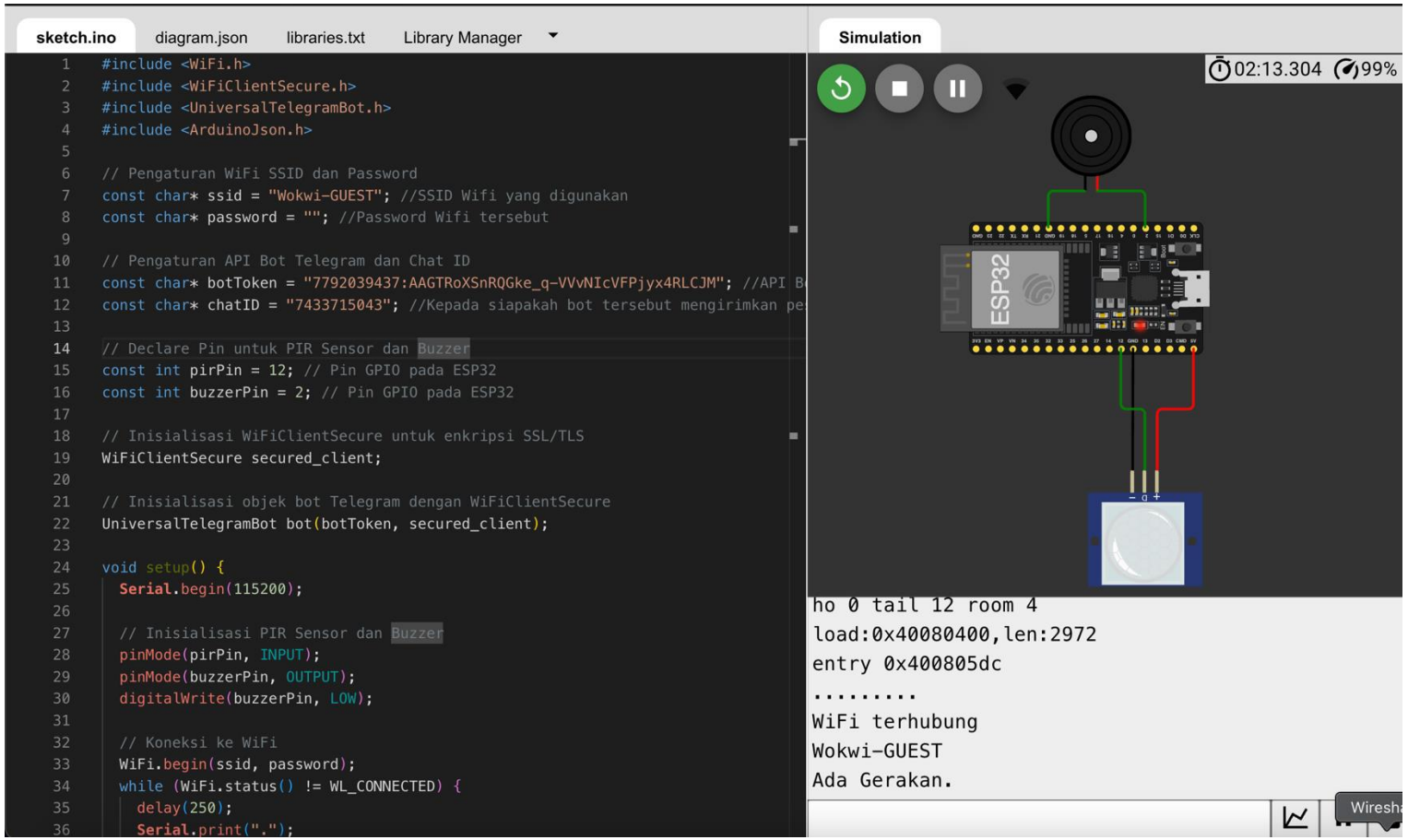
01:48.480 98%




```
WiFi connected!
10.10.0.2
Pesan diterima: /led_on
Pesan diterima: /state
Pesan diterima: /led_on
Pesan diterima: /start
Pesan diterima: /on
```

https://wokwi.com/projects/425439061445741569

Contoh 2



Firebase

Sejarah

- Firebase adalah suatu layanan dari Google untuk memberikan kemudahan bahkan mempermudah para developer aplikasi dalam mengembangkan aplikasinya.
- Firebase alias BaaS (*Backend as a Service*)
- Singkat cerita mengenai sejarah dari Firebase didirikan pertama kali pada tahun 2011 oleh Andrew Lee dan James Tamplin

Karakteristik Firebase Realtime Database

1. Database Real-Time
2. Struktur JSON
3. Synchronization
4. API RESTful
5. Security Rules
6. Dukungan untuk Aplikasi Android, iOS, dan Web
7. Skalabilitas

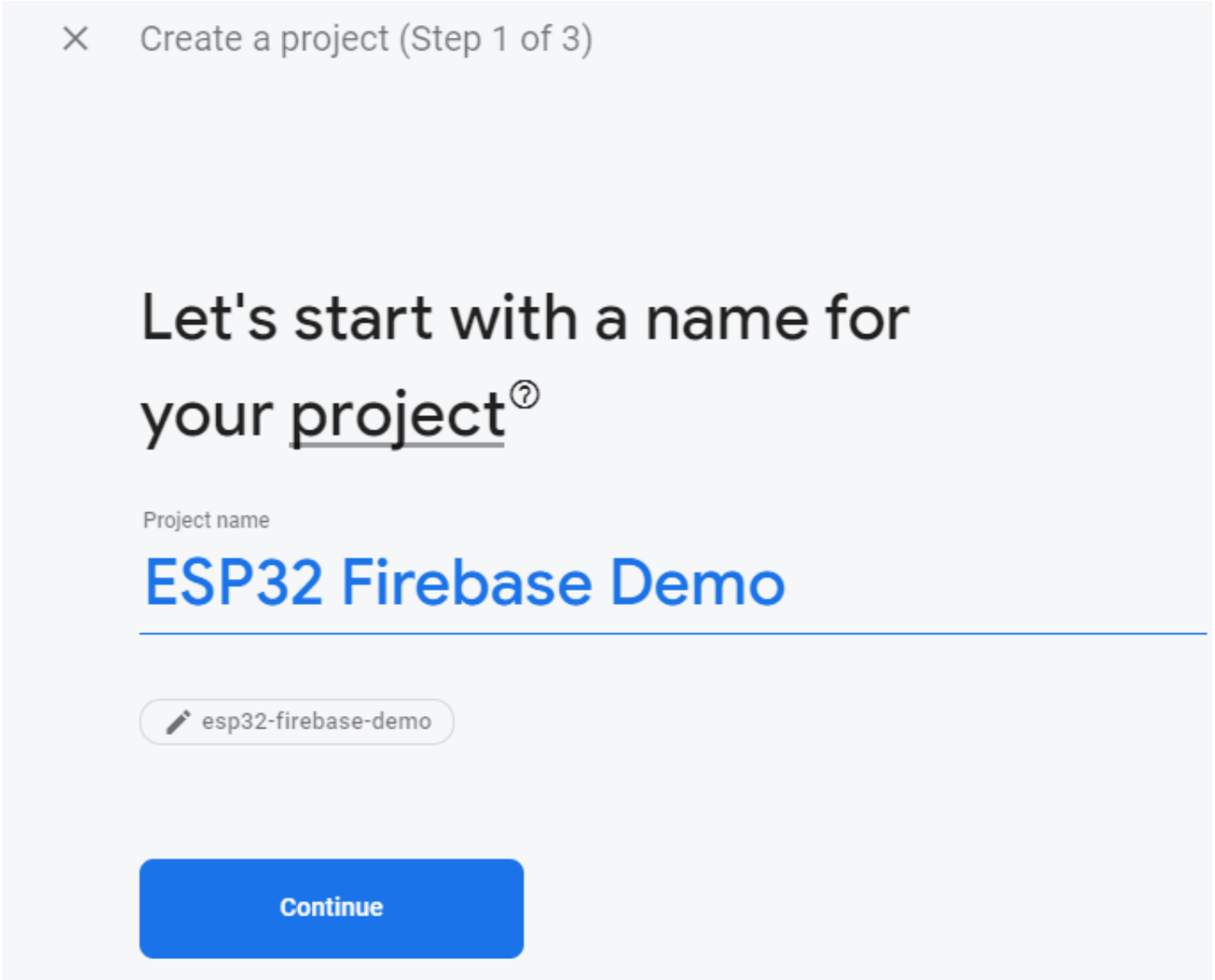
Tahapan Instalasi

- Tahapan yang dilakukan antara lain:

1. Pergi ke website https://firebase.google.com dan silahkan untuk melakukan sign ini menggunakan akun google
2. Selanjutnya click Get Started dan Add Project untuk membuat project Baru
3. Click *Get Started*, and then *Add project* to create a new project.
4. Berikan nama pada project yang diingikan, misalkan : ESP32 Firebase Demo

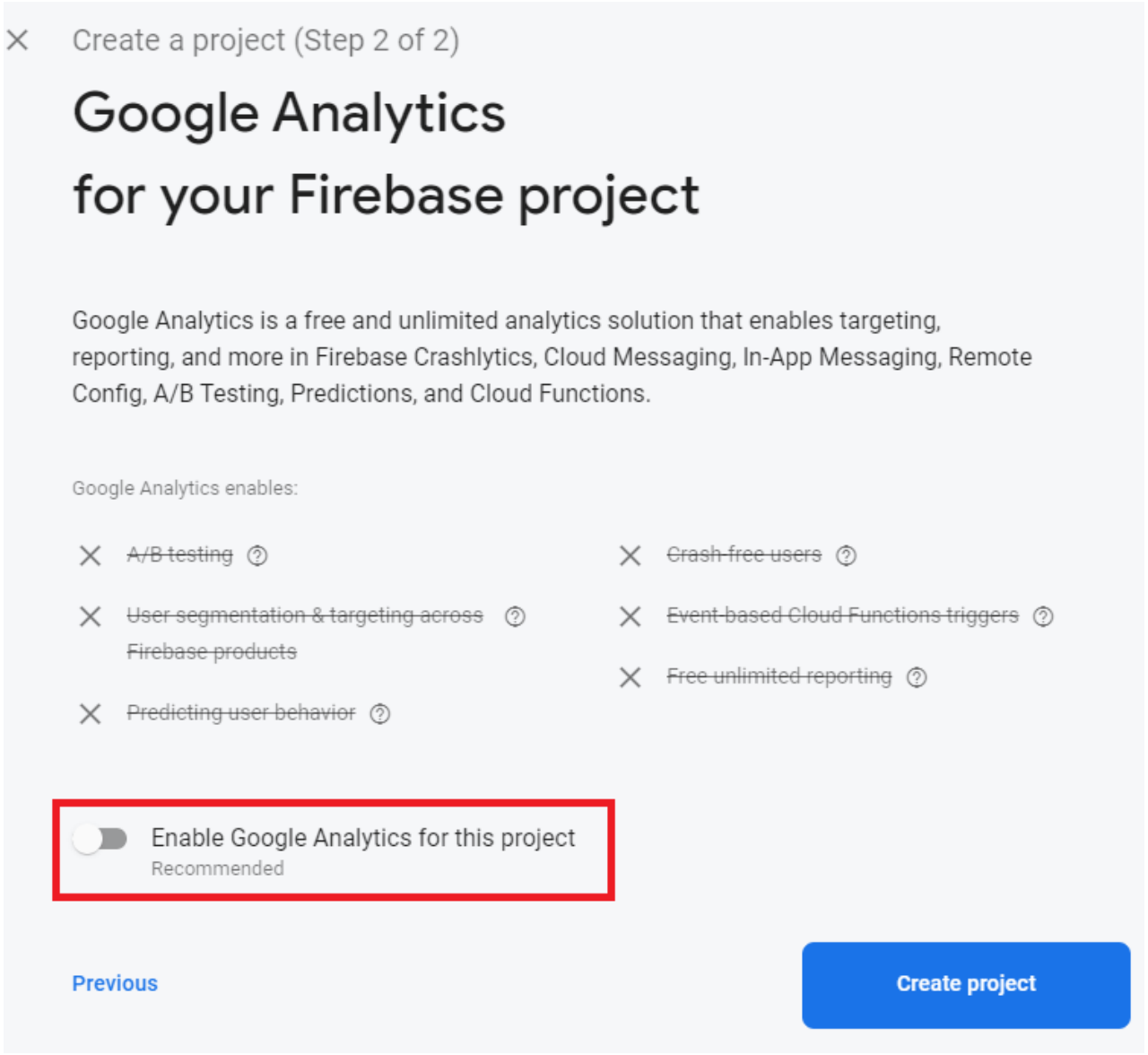
Tahap 1

- Tuliskan nama project



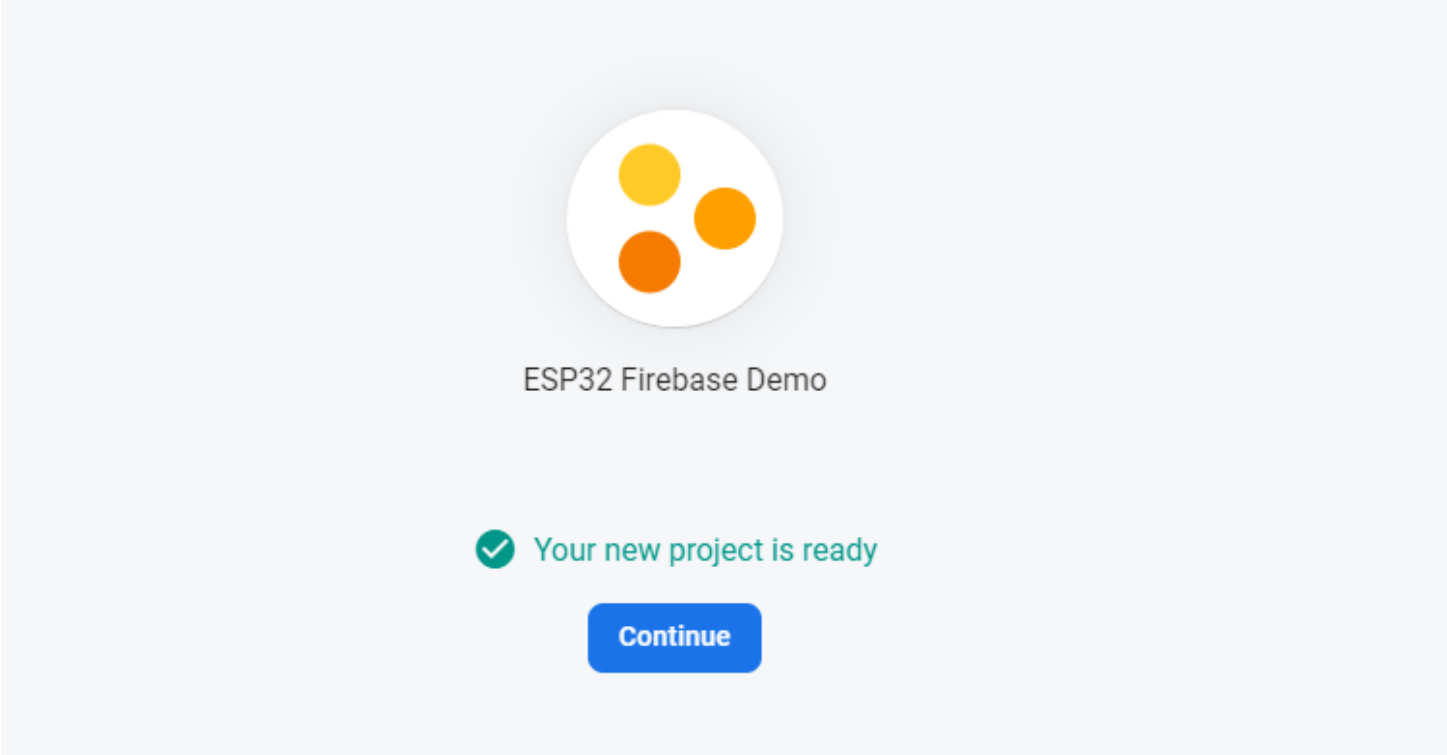
Tahap 2

- Lakukan disable pada pilihan *Enable Google Analytics* dan selanjutnya click Create Project.



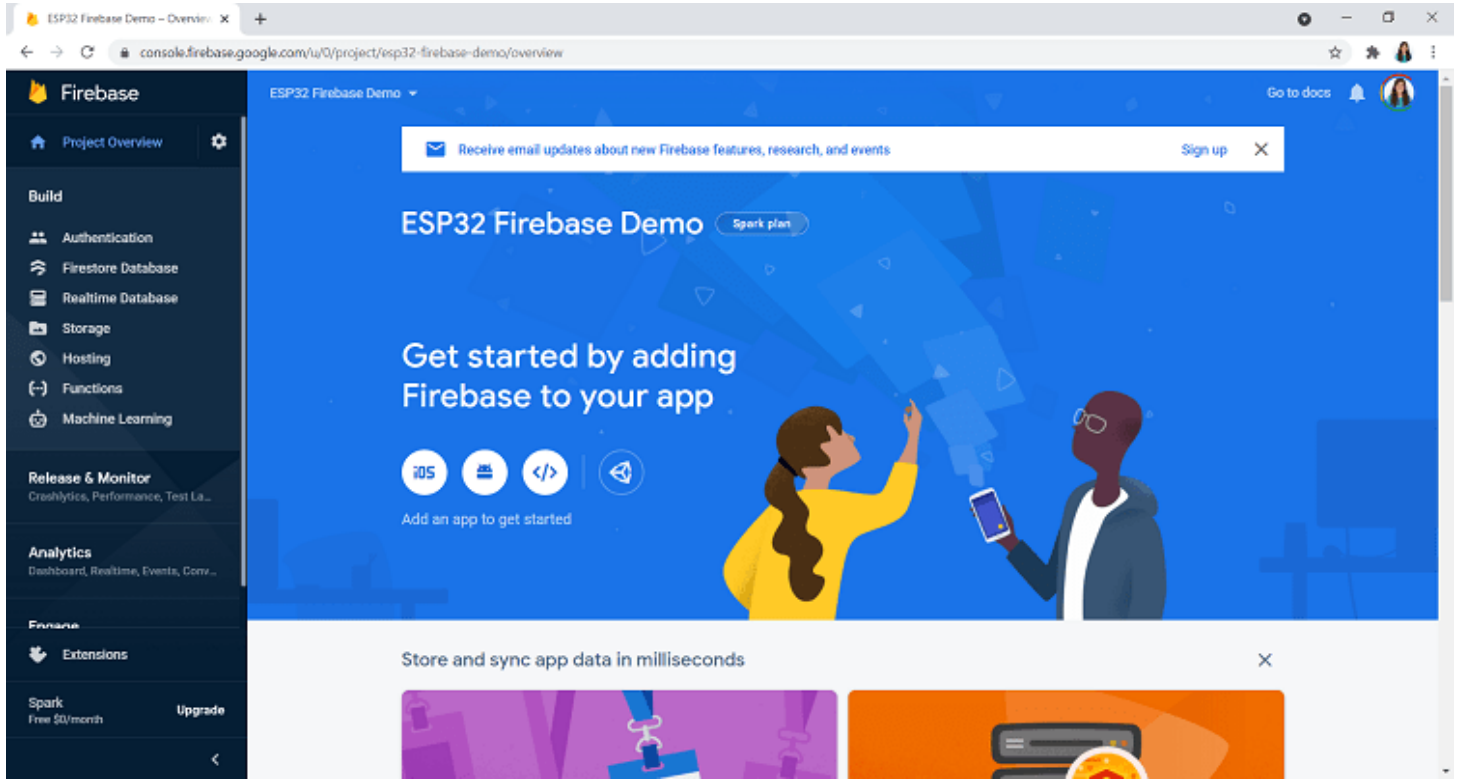
Tahap 3

- Diperlukan beberapa detik untuk menyiapkan proyek Anda. Lalu, klik continue jika sudah siap.



Tahap 4

- Selanjutnya akan diarahkan ke halam console project yang akan dikembangkan

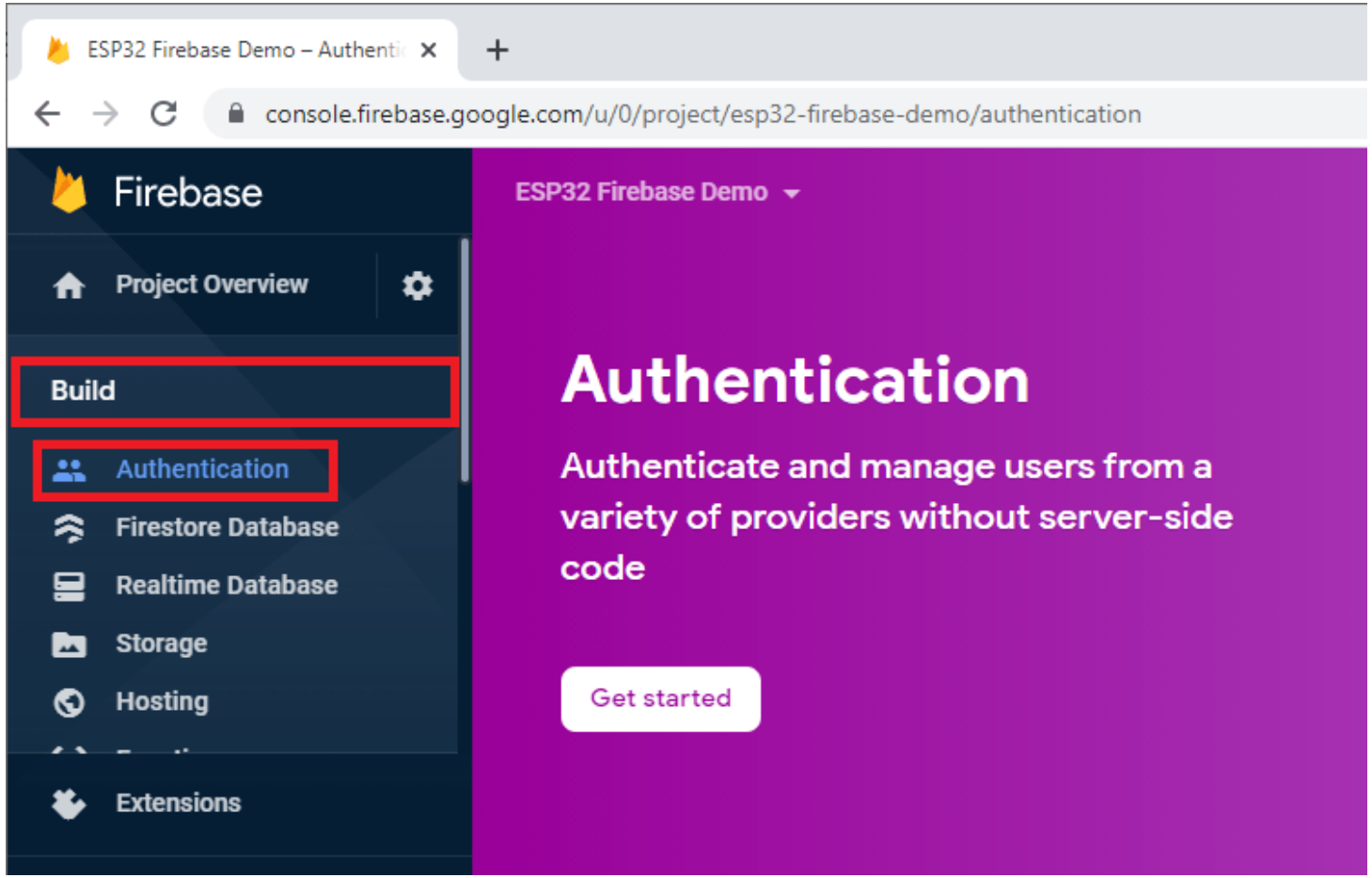


Identitas pengguna menjadi aspek krusial bagi sebagian besar aplikasi, yang melibatkan proses login untuk mengidentifikasi pengguna, seperti pada ESP32. Dengan informasi identitas pengguna yang terkelola, aplikasi dapat menyimpan data pengguna dengan aman di cloud dan menyajikan pengalaman terpersonalisasi yang konsisten di berbagai perangkat pengguna.

Pada menu sidebar sebelah kiri, klik *Authentication* dan *Get Started*

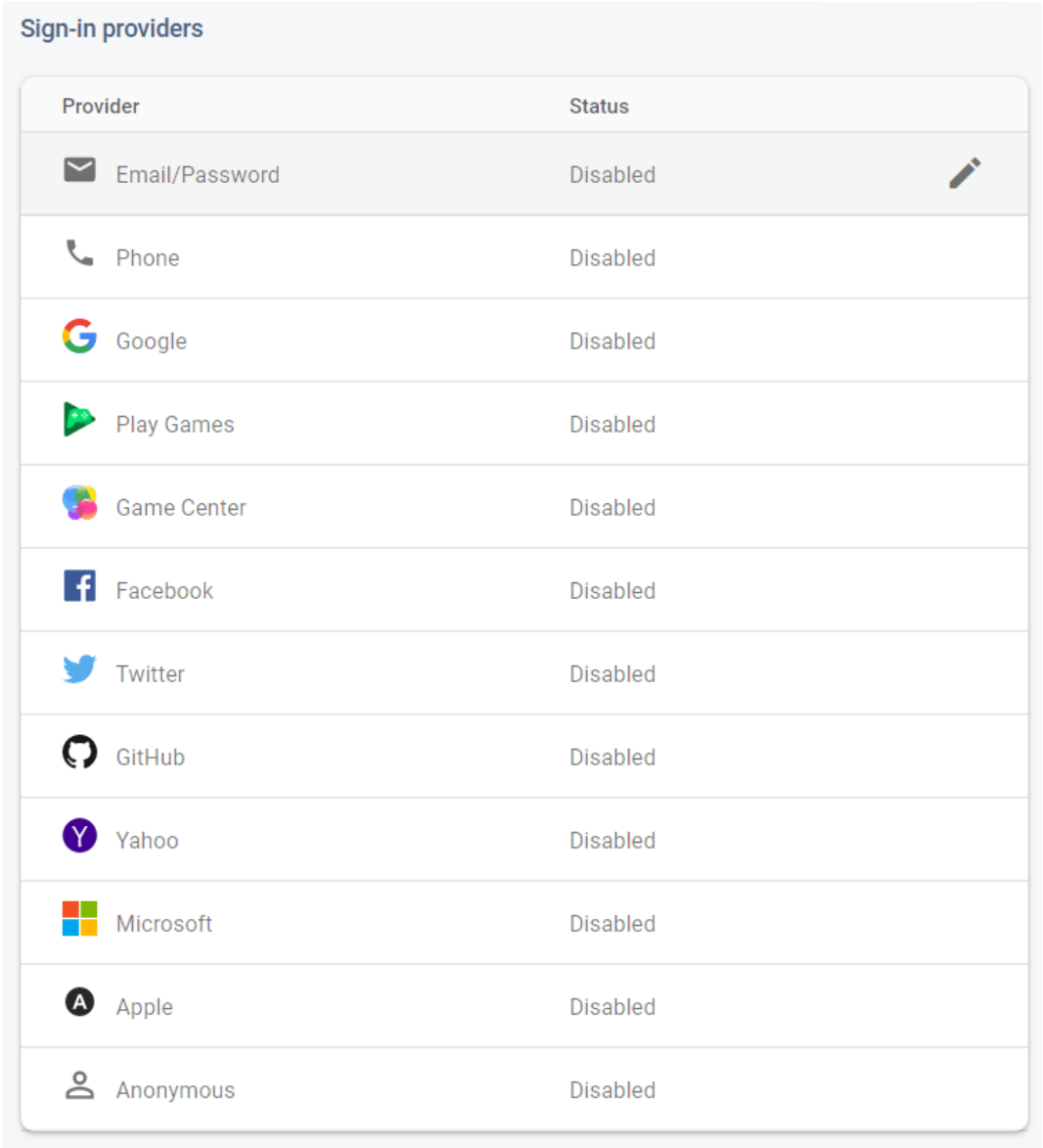
Tahap 1

- Pada menu sidebar sebelah kiri, klik *Authentication* dan *Get Started*



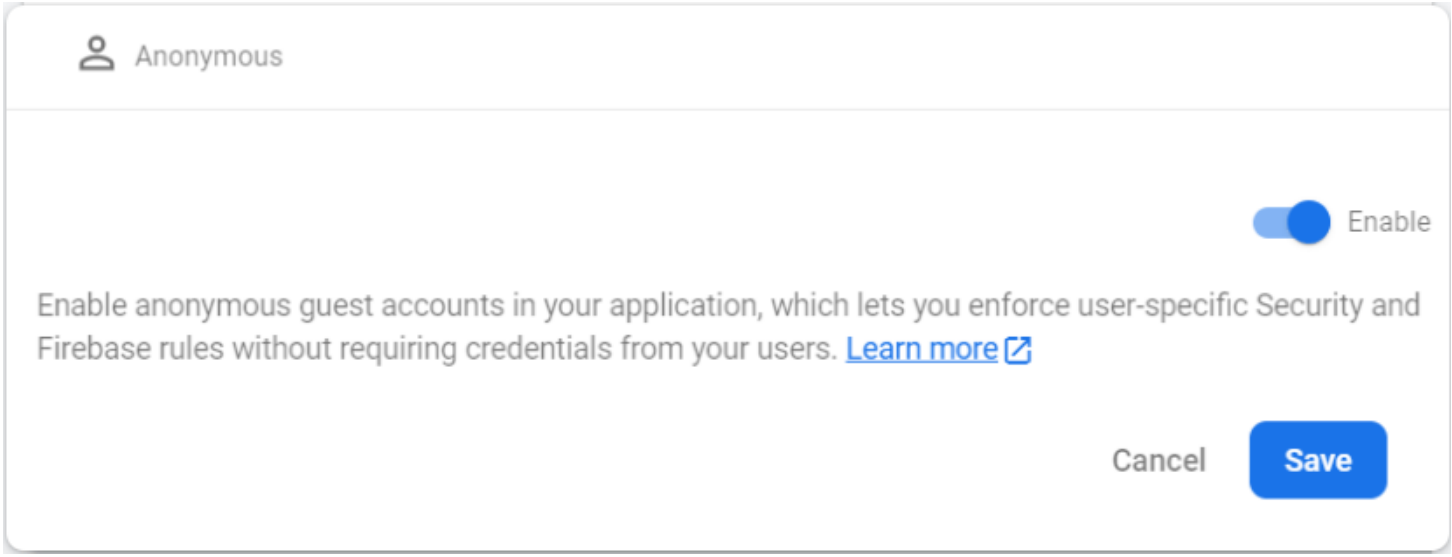
Tahap 2

- Terdapat banyak metode otentikasi yang dapat dipilih misalkan **email atau password, akun facebook, akun google, akun twitter, anonymous, dll.**



Tahap 3

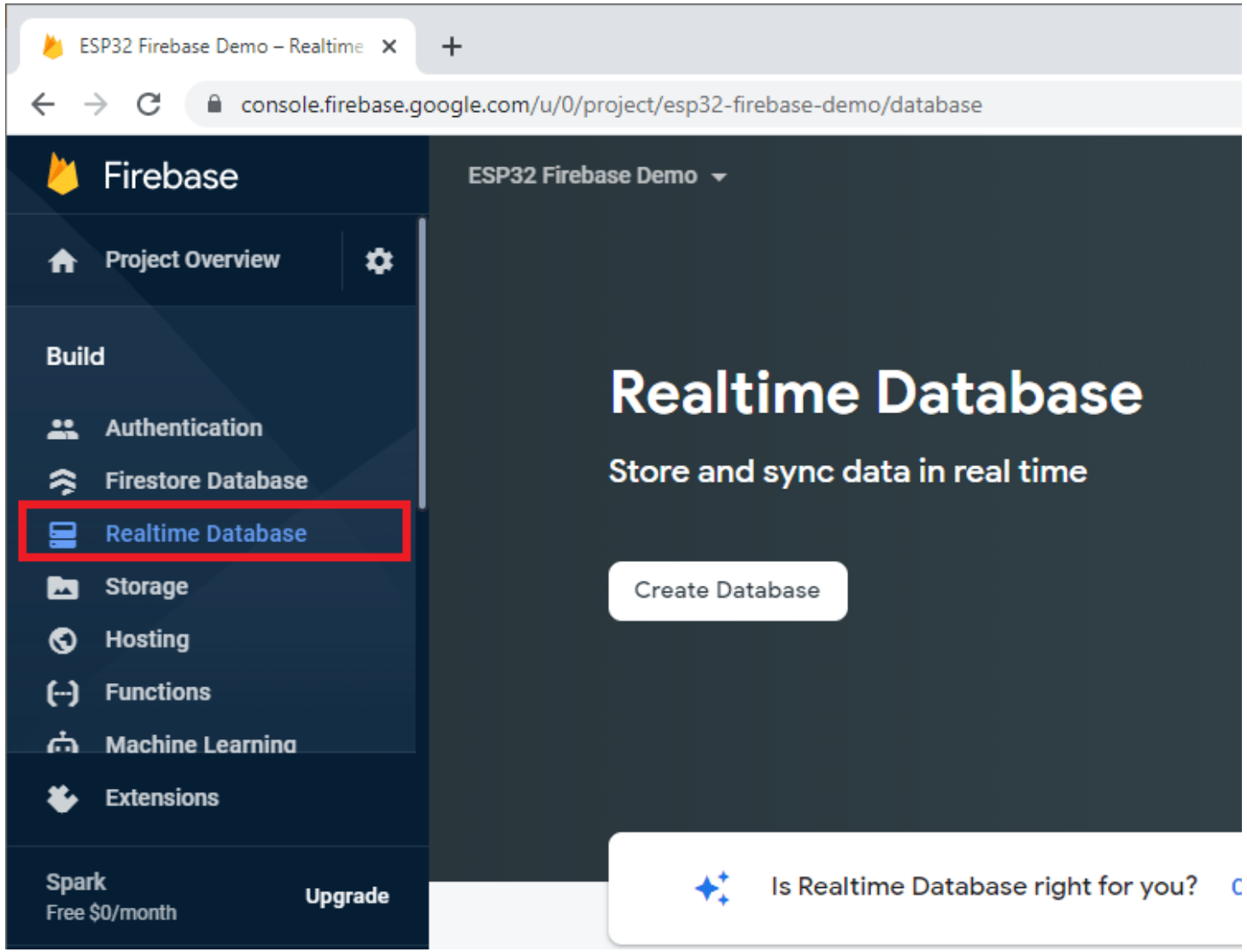
- Untuk tujuan pengujian, kita dapat memilih pengguna Anonim (membutuhkan otentikasi tanpa meminta pengguna untuk masuk terlebih dahulu dengan membuat akun anonim sementara). Aktifkan opsi tersebut dan klik Simpan.



Tahap 4

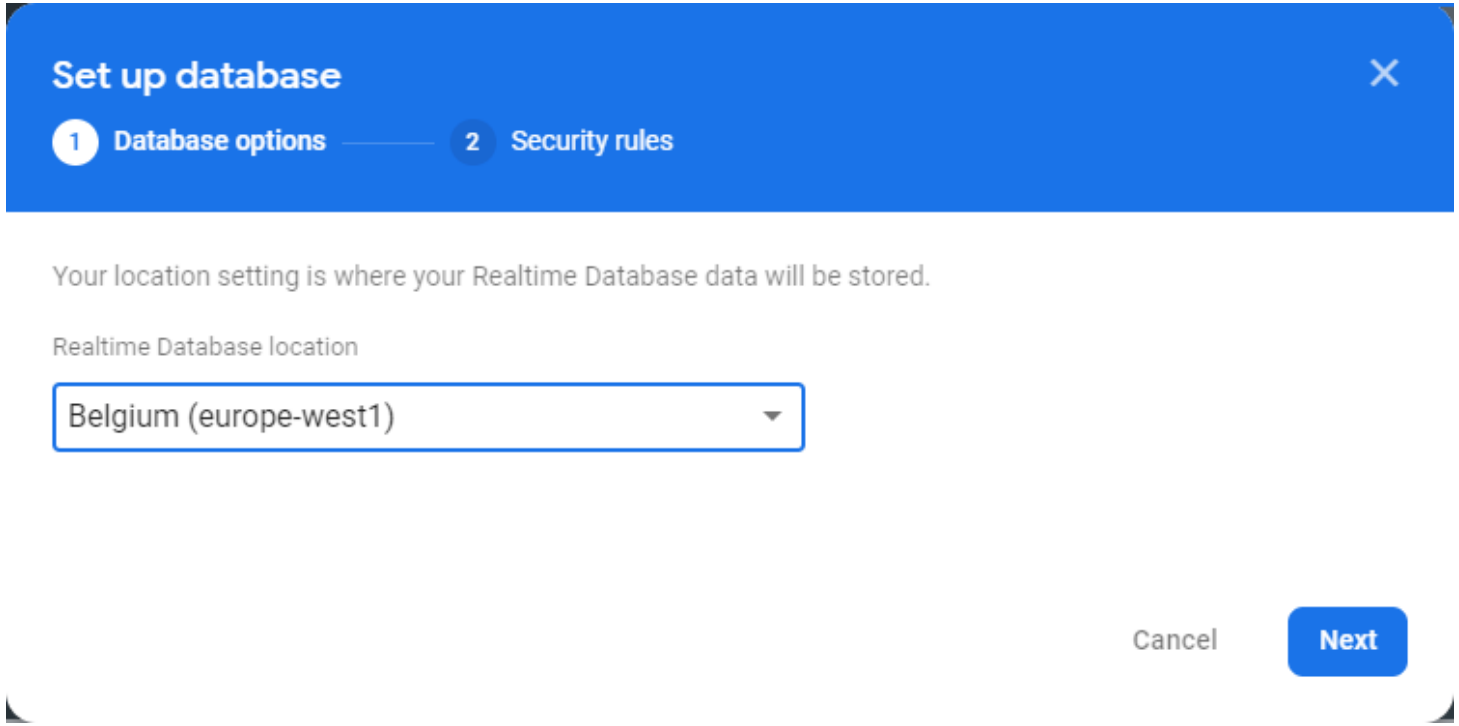
Pada tahap ini akan dibahas cara untuk membuat realtime database. Tahapan yang dilakukan antara lain:

- Pada sidebar sebelah kiri klik Realtime Database lalu klik Create Database.



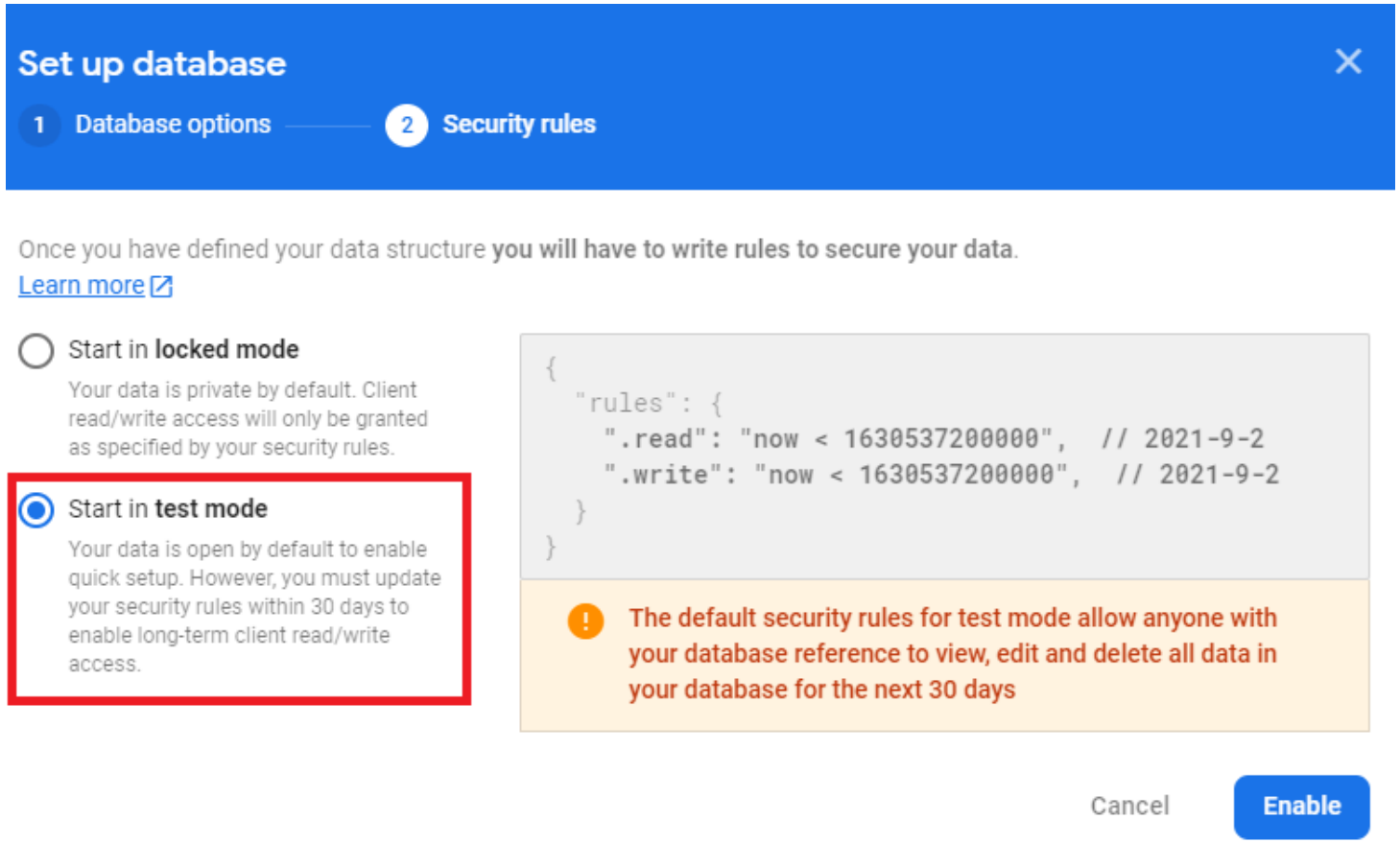
Tahap 5

- Kemudian pilih lokasi realtime database sesuai domisili terdekat.



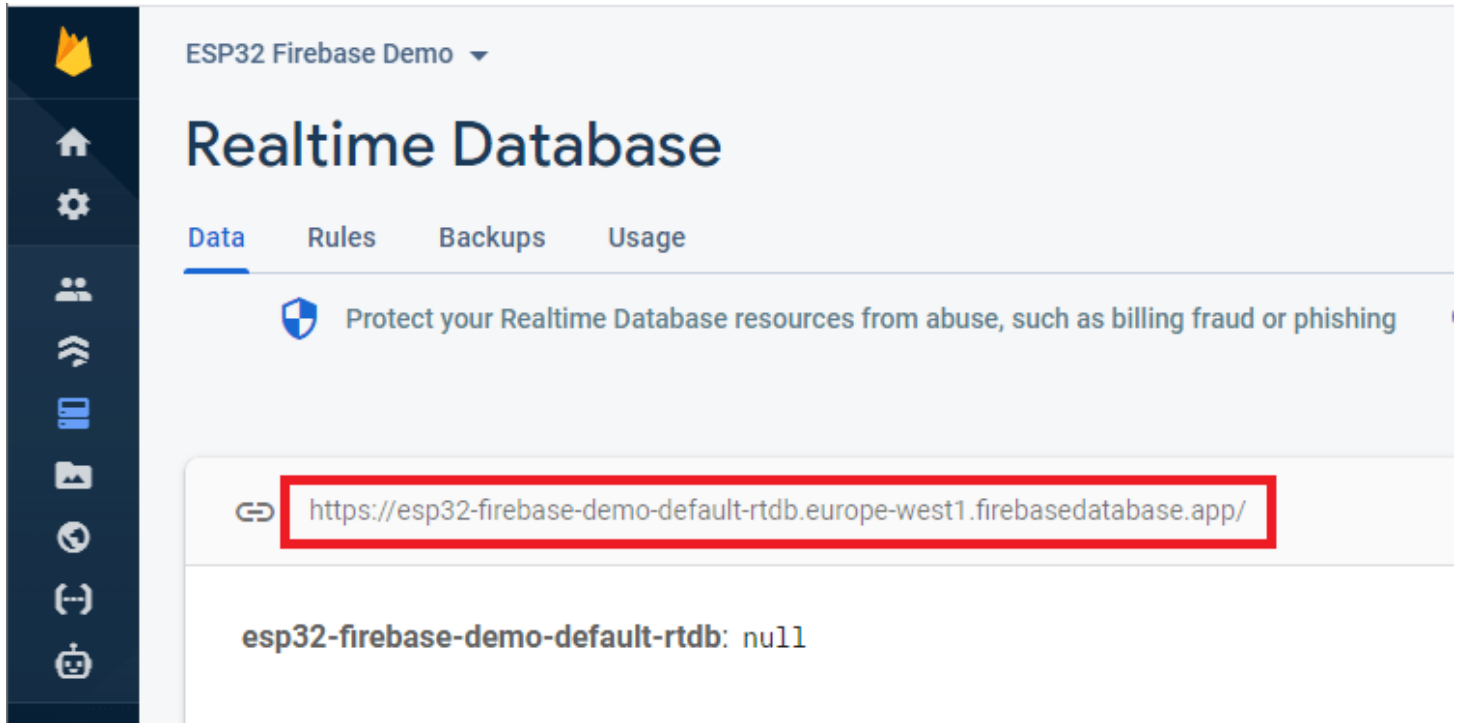
Tahap 6

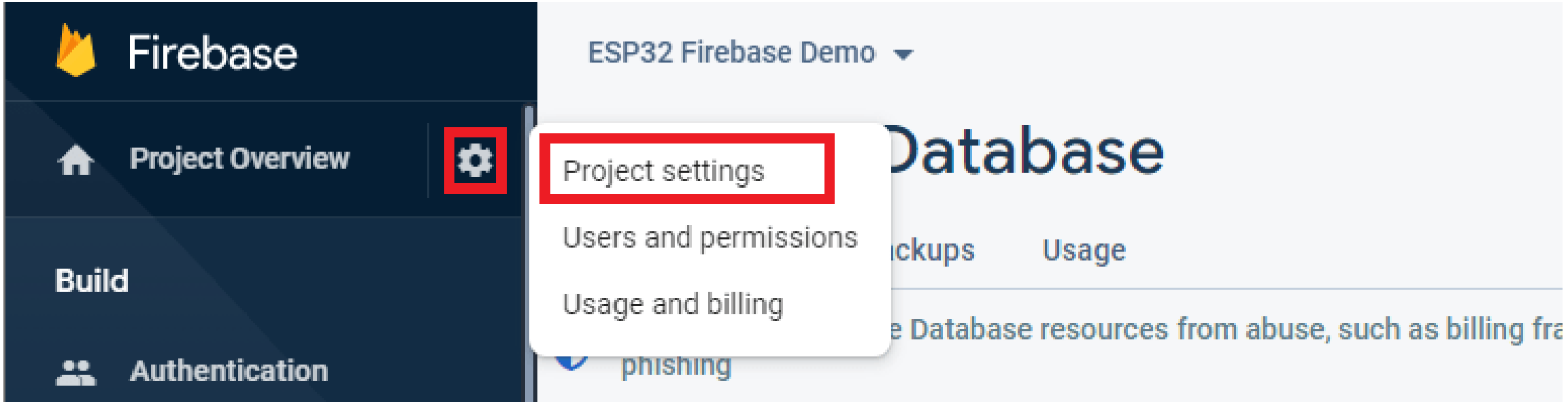
- Atur mode keamanan untuk database. Untuk tujuan pengujian, silahkan pilih *Start in test mode*.

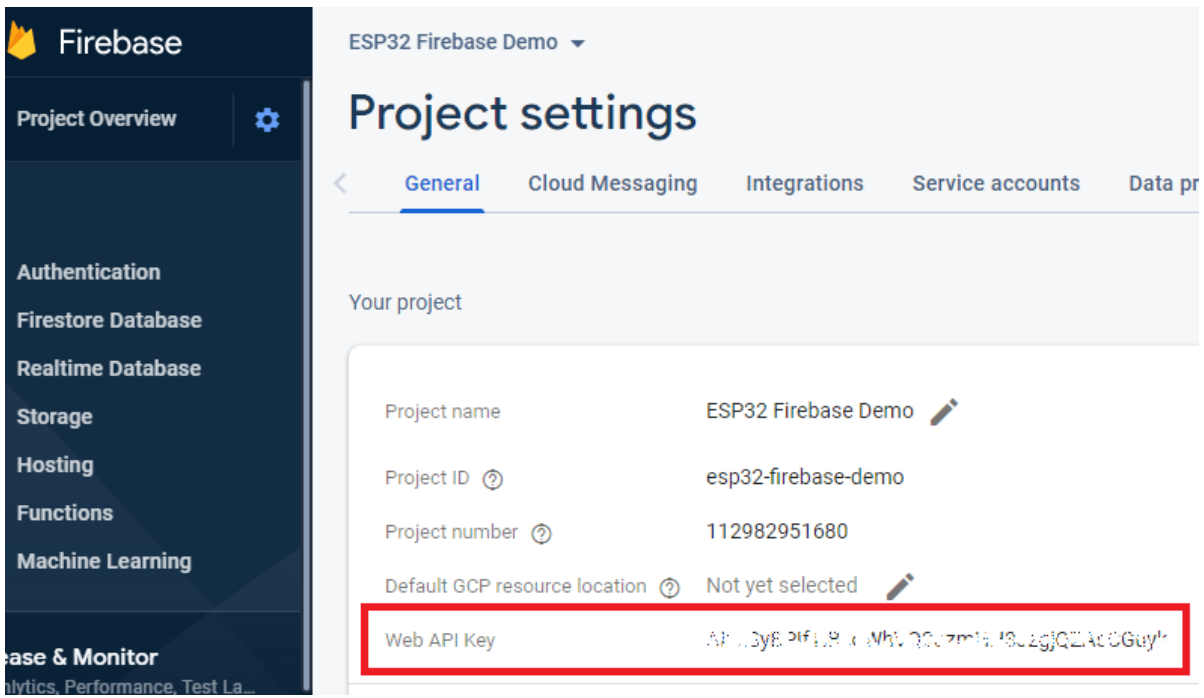


Tahap 7

- Database telah terbuat. Kita membutuhkan untuk copy dan save URL database (nanti akan dibutuhkan untuk kode program).

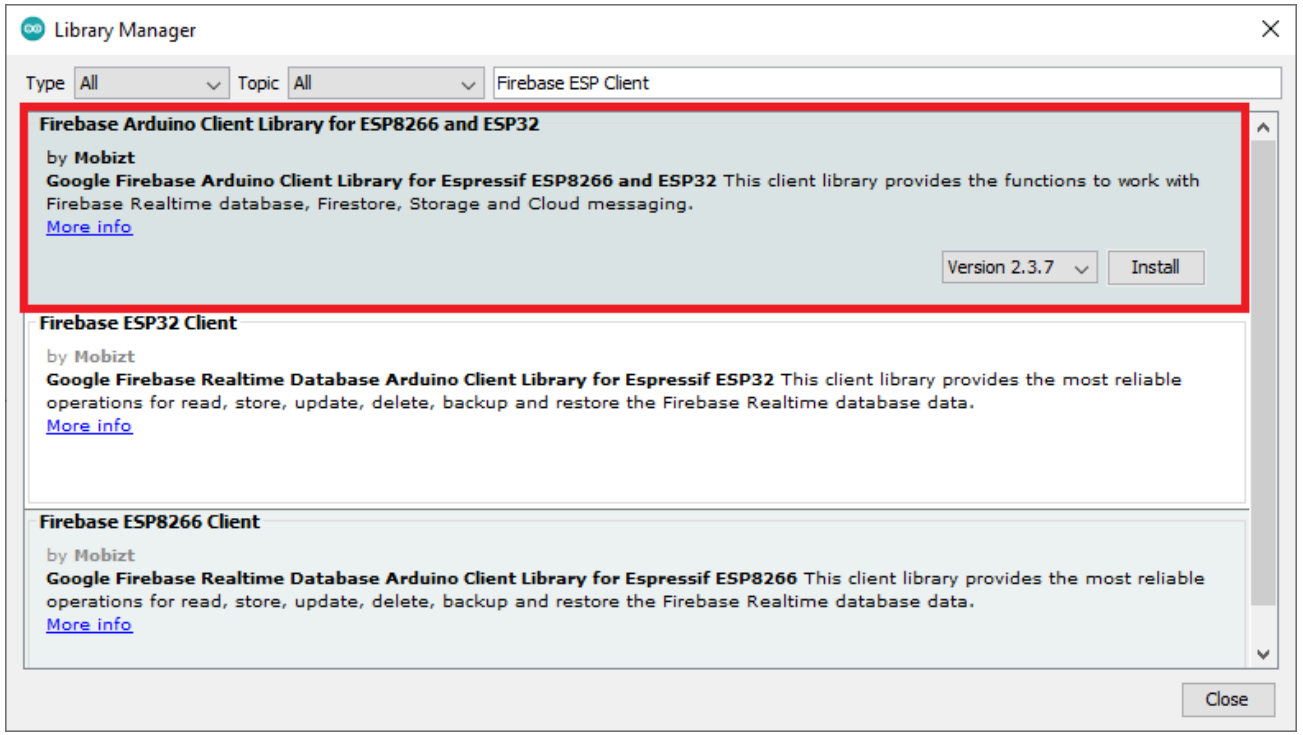



Firebase
ESP32 Firebase Demo
Project Overview
Project settings
Users and permissions
Usage and billing
Build
Authentication
Database
ckups
Usage
e Database resources from abuse, such as billing fra
phishing


Firebase
Project Overview
Authentication
Firestore Database
Realtime Database
Storage
Hosting
Functions
Machine Learning
ase & Monitor
lytics, Performance, Test La...
ESP32 Firebase Demo
Project settings
General
Cloud Messaging
Integrations
Service accounts
Data pr
Your project
Project name
ESP32 Firebase Demo
Project ID
esp32-firebase-demo
Project number
112982951680
Default GCP resource location
Not yet selected
Web API Key

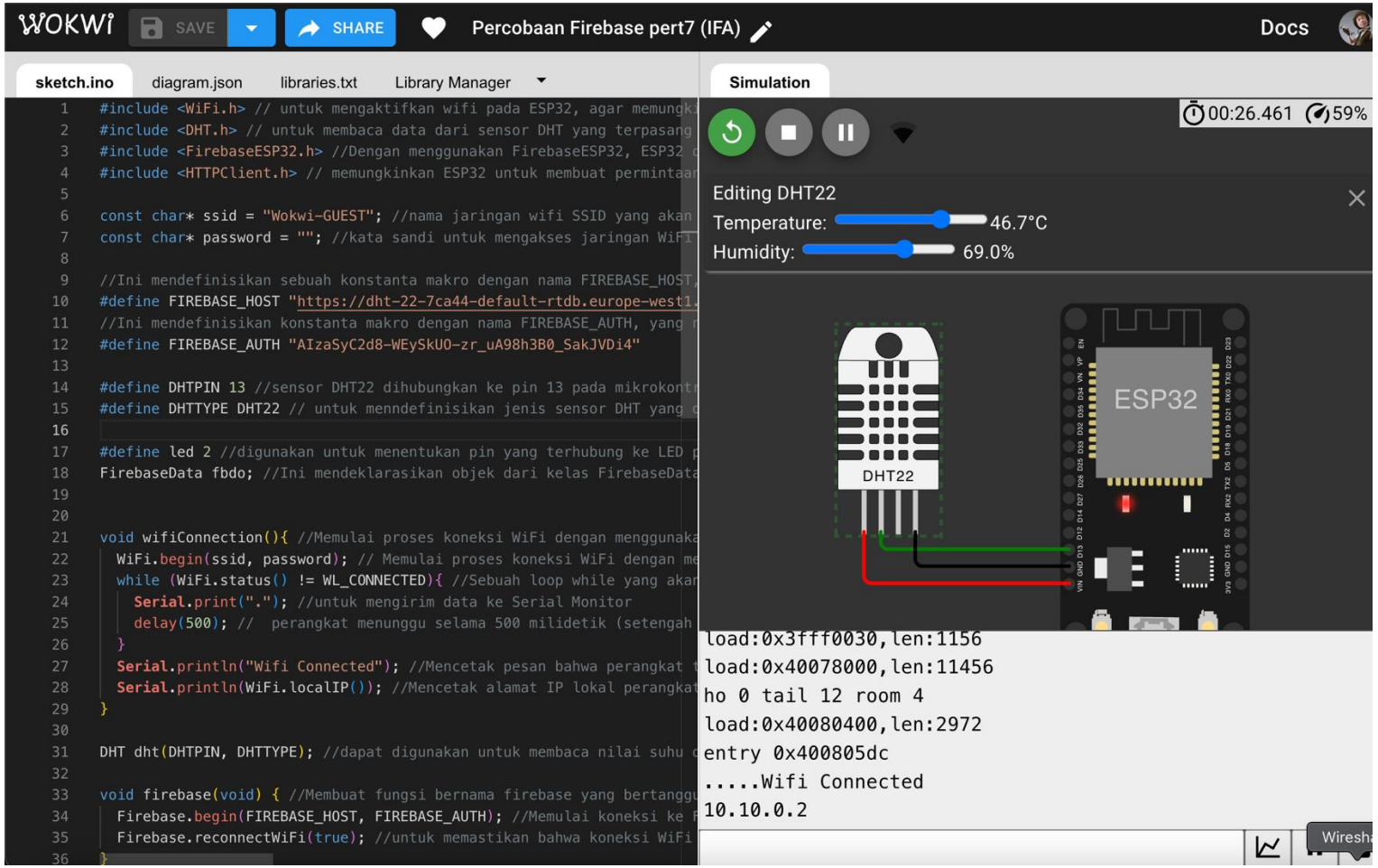
Library (Firebase)

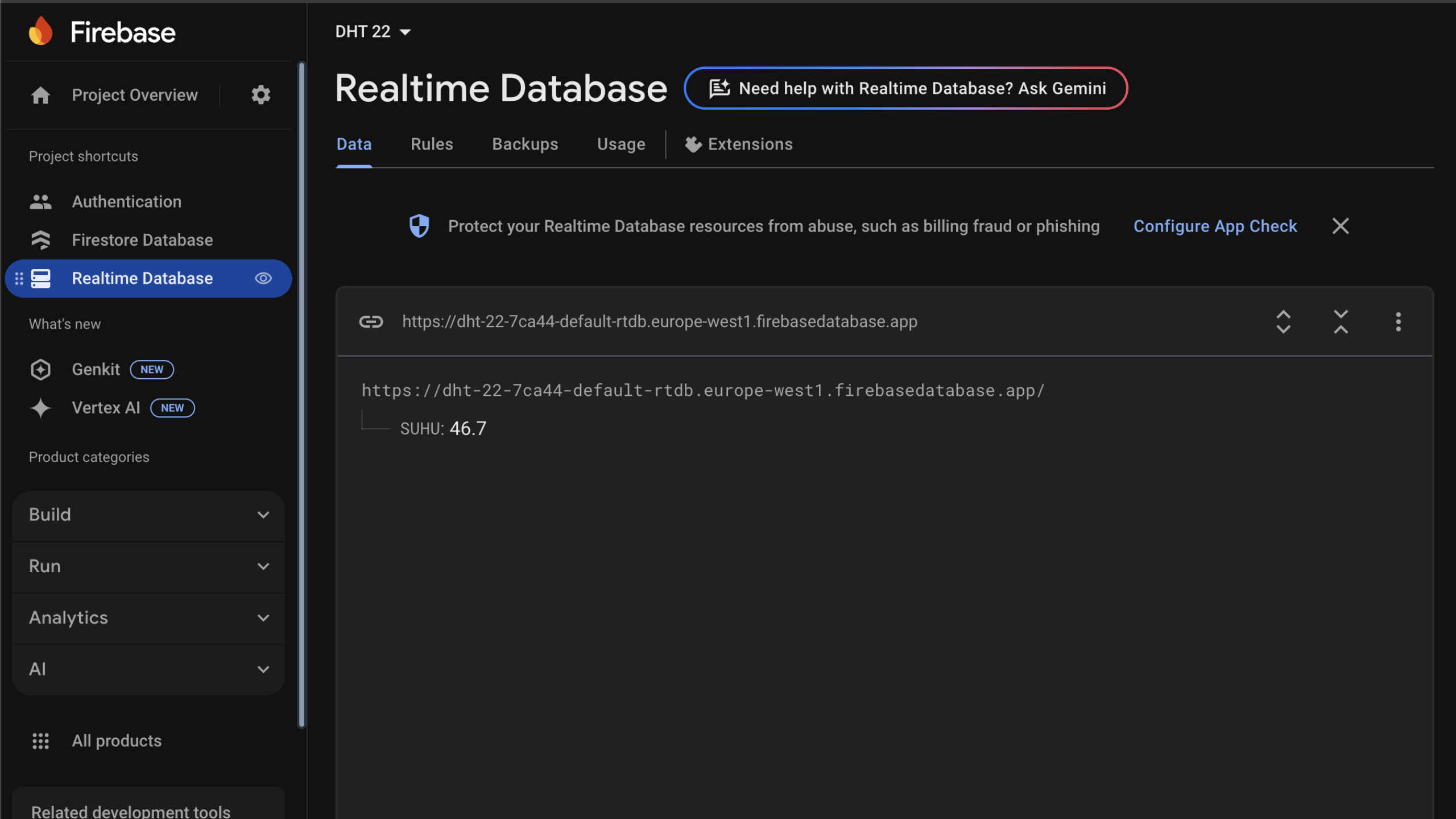
- Jika Anda menggunakan Arduino IDE, ikuti langkah selanjutnya untuk menginstal perpustakaan.
    1. Buka **Sketch > Include Library > Manage Libraries**
    2. Cari Firebase ESP Client dan instal Firebase Arduino Client Library untuk ESP8266 dan ESP32 oleh Mobitz.



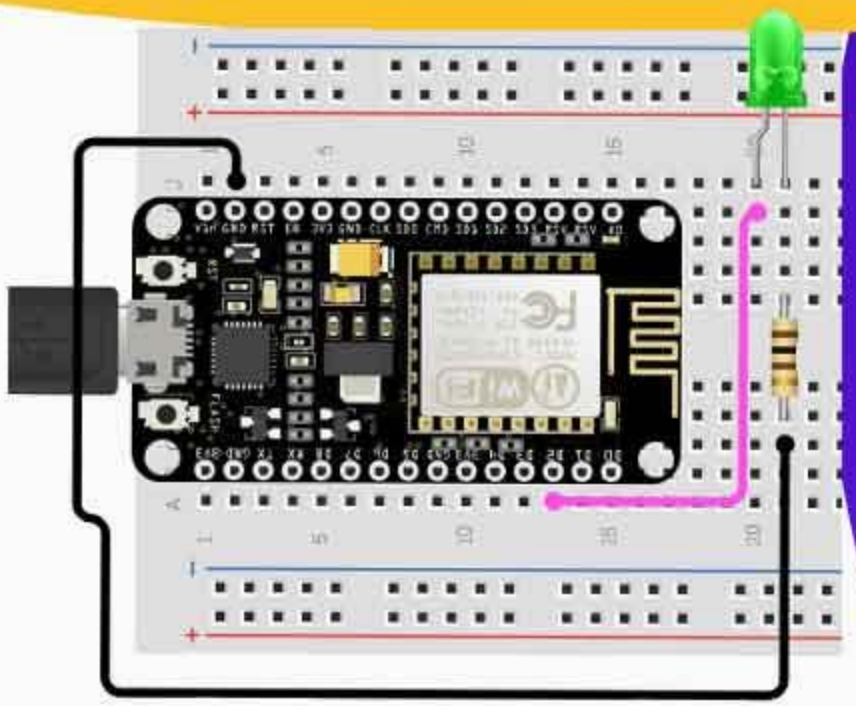
https://wokwi.com/projects/425440393242213377

Contoh




Firebase
DHT 22
Project Overview
Project shortcuts
Authentication
Firestore Database
Realtime Database
What's new
Genkit NEW
Vertex AI NEW
Product categories
Build
Run
Analytics
AI
All products
Related development tools
Realtime Database
Need help with Realtime Database? Ask Gemini
Data
Rules
Backups
Usage
Extensions
Protect your Realtime Database resources from abuse, such as billing fraud or phishing
Configure App Check
https://dht-22-7ca44-default-rtdb.europe-west1.firebasedatabase.app
https://dht-22-7ca44-default-rtdb.europe-west1.firebasedatabase.app/
SUHU: 46.7

IoT Based LED Control using Google Firebase & ESP8266



ESP8266

+

Firebase

Firebase Code

- Deklarasi library dan API Key serta URL Firebase

firebase_led_2.ino

```
1  #include <Arduino.h>
2  #if defined(ESP32)
3    #include <WiFi.h>
4  #elif defined(ESP8266)
5    #include <ESP8266WiFi.h>
6  #endif
7  #include <Firebase_ESP_Client.h>
8
9  //Provide the token generation process info.
10 #include "addons/TokenHelper.h"
11 //Provide the RTDB payload printing info and other helper functions.
12 #include "addons/RTDBHelper.h"
13
14 // Insert your network credentials
15 #define WIFI_SSID "IFA-Homes"
16 #define WIFI_PASSWORD "berkah143"
17
18 // Insert Firebase project API Key
19 #define API_KEY "[redacted]"
20
21 // Insert RTDB URLefine the RTDB URL */
22 #define DATABASE_URL "https://ledcontrol-7557c-default-rtdb.firebaseio.com/"
```

Firebase Code 2

Fungsi untuk setup

Koneksi dan Konfigurasi Firebase

firebase_led_2.ino

```
24  //Define Firebase Data object
25  FirebaseData fbdo;
26
27  FirebaseAuth auth;
28  FirebaseConfig config;
29
30  //some importent variables
31  String sValue, sValue2;
32  bool signupOK = false;
33  int led1 = 2; //PIN D4 NODEMCU ESP8266 AMICA
34  void setup() {
35    Serial.begin(115200);
36
37    pinMode(led1,OUTPUT);
38    // pinMode(led2,OUTPUT);
39    WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
40    Serial.print("Connecting to Wi-Fi");
41    while (WiFi.status() != WL_CONNECTED) {
42      Serial.print(".");
43      delay(300);
44    }
45    Serial.println();
46    Serial.print("Connected with IP: ");
47    Serial.println(WiFi.localIP());
48    Serial.println();
49
50    /* Assign the api key (required) */
51    config.api_key = API_KEY;
52
53    /* Assign the RTDB URL (required) */
54    config.database_url = DATABASE_URL;
55
56    /* Sign up */
57    if (Firebase.signUp(&config, &auth, "", "")) {
58      Serial.println("ok");
59      signupOK = true;
60    }
61    else {
62      Serial.printf("%s\n", config.signer.signupError.message.c_str());
63    }
64
65    /* Assign the callback function for the long running token generation task */
66    config.token_status_callback = tokenStatusCallback; //see addons/TokenHelper.h
67
68    Firebase.begin(&config, &auth);
69    Firebase.reconnectWiFi(true);
70  }
```

Firebase code 3

Void Loop

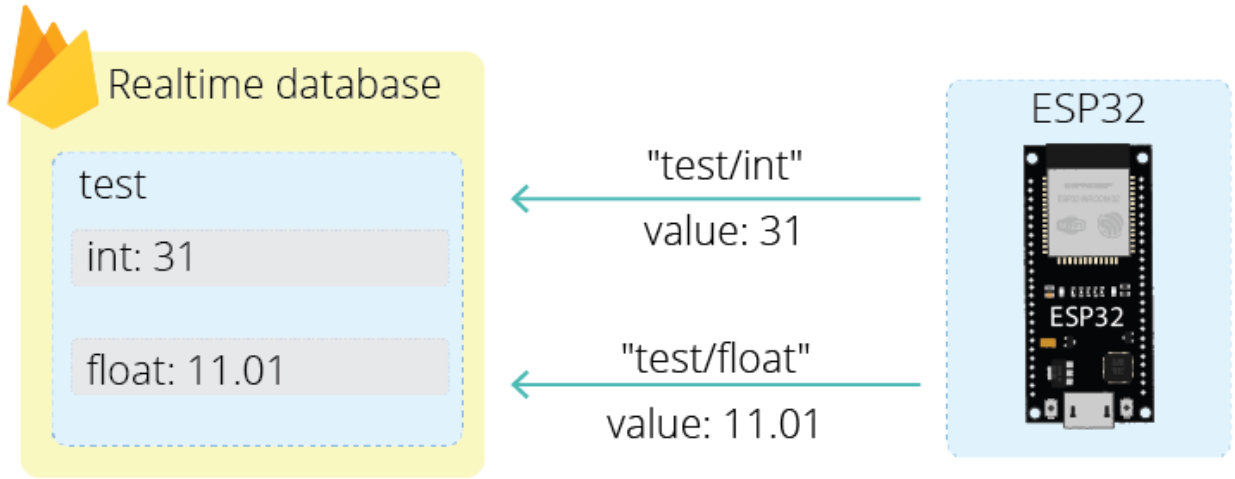
Fungsi untuk menyalakan lampu.

Menerima masukan dalam bentuk string menggunakan **Fungsi GetString**

```
72  void loop() {
73    if (Firebase.ready() && signupOK ) {
74      if (Firebase.RTDB.getString(&fbdo, "/L1")) {
75        if (fbdo.dataType() == "string") {
76          sValue = fbdo.stringData();
77          int a = sValue.toInt();
78          Serial.println(a);
79          if (a == 1){
80            digitalWrite(led1,HIGH);
81          }else{
82            digitalWrite(led1,LOW);
83          }
84        }
85      }
86      else {
87        Serial.println(fbdo.errorReason());
88      }
89
90    }
91  }
```

Simpan Data

- Simpan Data dengan Tipe Data : Int dan Float
- Hasil akan disimpan secara acak setiap 15 detik



Kode Program Write

- Fungsi Loop untuk

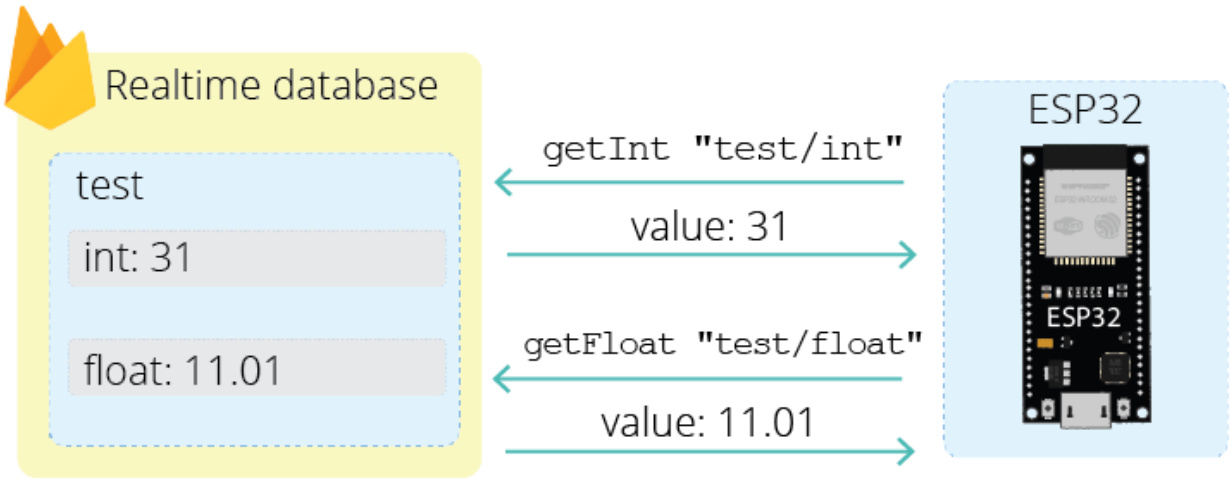
simpan data

SetInt

```
55  void loop(){
56    if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 15000 || sendDataPrevMillis == 0)){
57      sendDataPrevMillis = millis();
58      // Write an Int number on the database path test/int
59      if (Firebase.RTDB.setInt(&fbdo, "test/int", count)){
60        Serial.println("PASSED");
61        Serial.println("PATH: " + fbdo.dataPath());
62        Serial.println("TYPE: " + fbdo.dataType());
63      }
64      else {
65        Serial.println("FAILED");
66        Serial.println("REASON: " + fbdo.errorReason());
67      }
68      count++;
69
70      // Write an Float number on the database path test/float
71      if (Firebase.RTDB.setFloat(&fbdo, "test/float", 0.01 + random(0,100))){
72        Serial.println("PASSED");
73        Serial.println("PATH: " + fbdo.dataPath());
74        Serial.println("TYPE: " + fbdo.dataType());
75      }
76      else {
77        Serial.println("FAILED");
78        Serial.println("REASON: " + fbdo.errorReason());
79      }
80    }
81  }
```

Baca Data

- Baca Data dengan Tipe Data : Int dan Float
- Data akan dibaca dari database setiap 15 detik



Kode Program Read

- Fungsi Loop untuk baca data

GetInt

```
57  void loop() {
58    if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis > 15000 || sendDataPrevMillis == 0)) {
59      sendDataPrevMillis = millis();
60      if (Firebase.RTDB.getInt(&fbdo, "/test/int")) {
61        if (fbdo.dataType() == "int") {
62          intValue = fbdo.intData();
63          Serial.println(intValue);
64        }
65      }
66      else {
67        Serial.println(fbdo.errorReason());
68      }
69
70      if (Firebase.RTDB.getFloat(&fbdo, "/test/float")) {
71        if (fbdo.dataType() == "float") {
72          floatValue = fbdo.floatData();
73          Serial.println(floatValue);
74        }
75      }
76      else {
77        Serial.println(fbdo.errorReason());
78      }
79    }
80  }
```

Terimakasih