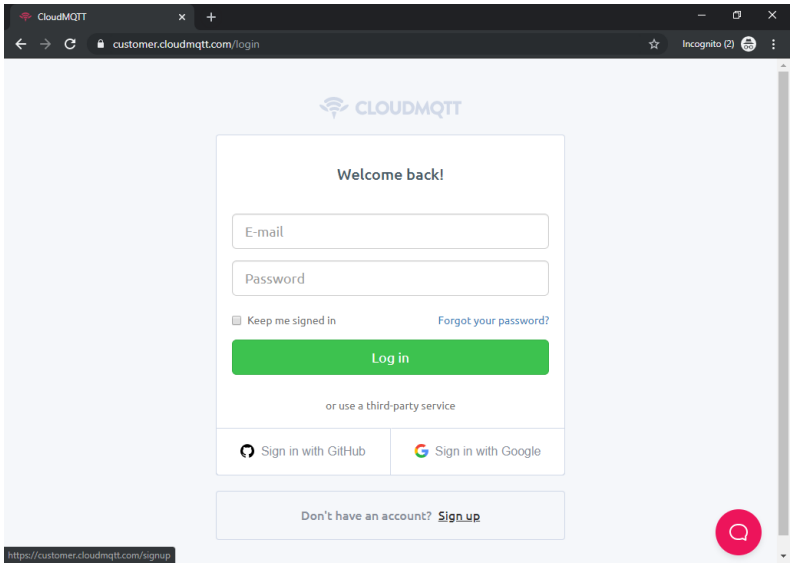
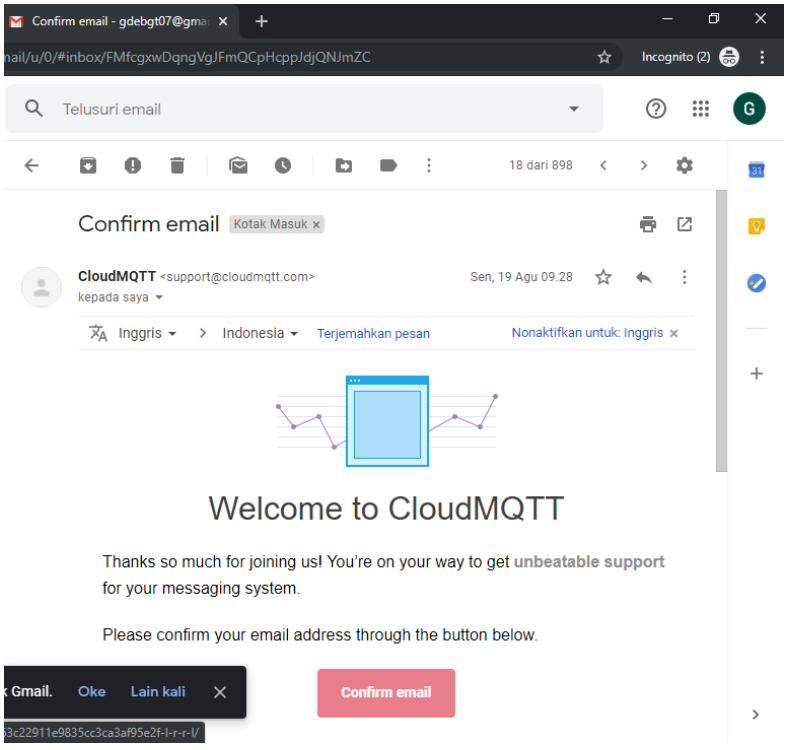
Lagkah-langkah implementasi penggunaan Publisher (esp32) dan subscriber (laptop atau hp) menggunakan CloudMQTT sebagai Broker

Setting Broker

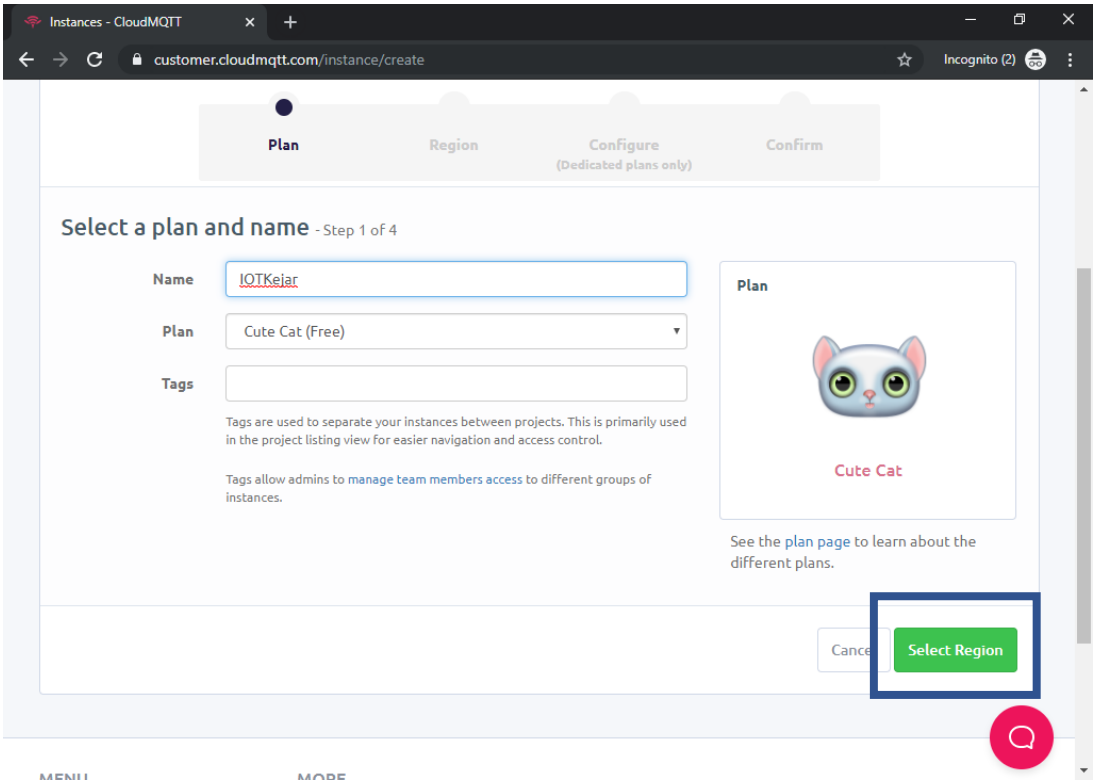
1. Buka website CloudMQTT di https://www.cloudmqtt.com/
2. Daftar dengan menggunakan email yang anda miliki di menu Login, lalu sign up



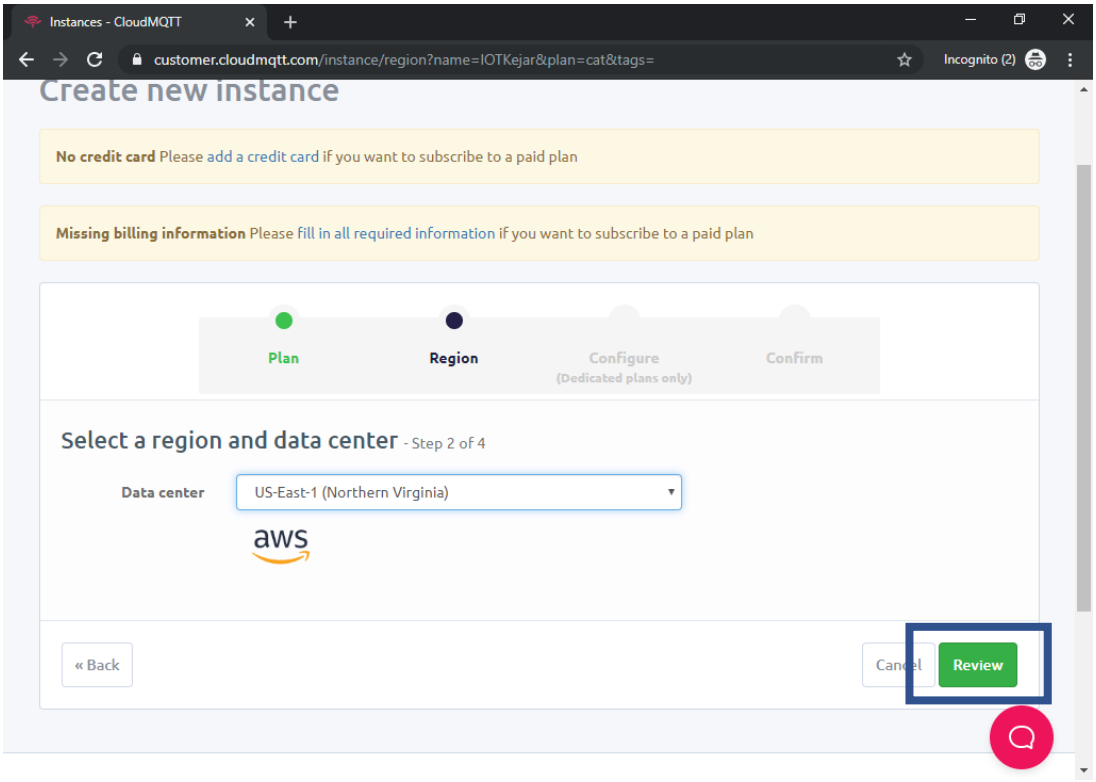
3. Buka email anda, lalu lakukan konfirmasi email



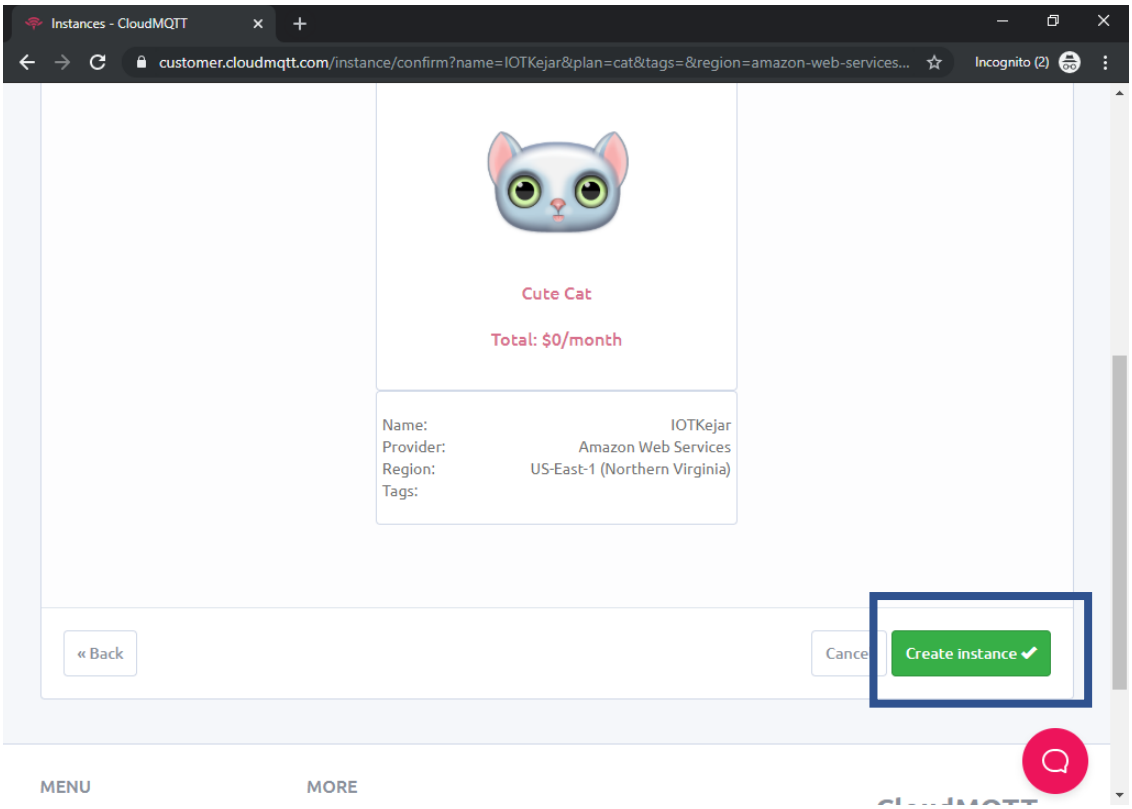
4. Login dengan menggunakan account yang telah anda buat
5. Klik Create New Instance
6. Masukkan nama instance yang diinginkan, lalu klik select regional



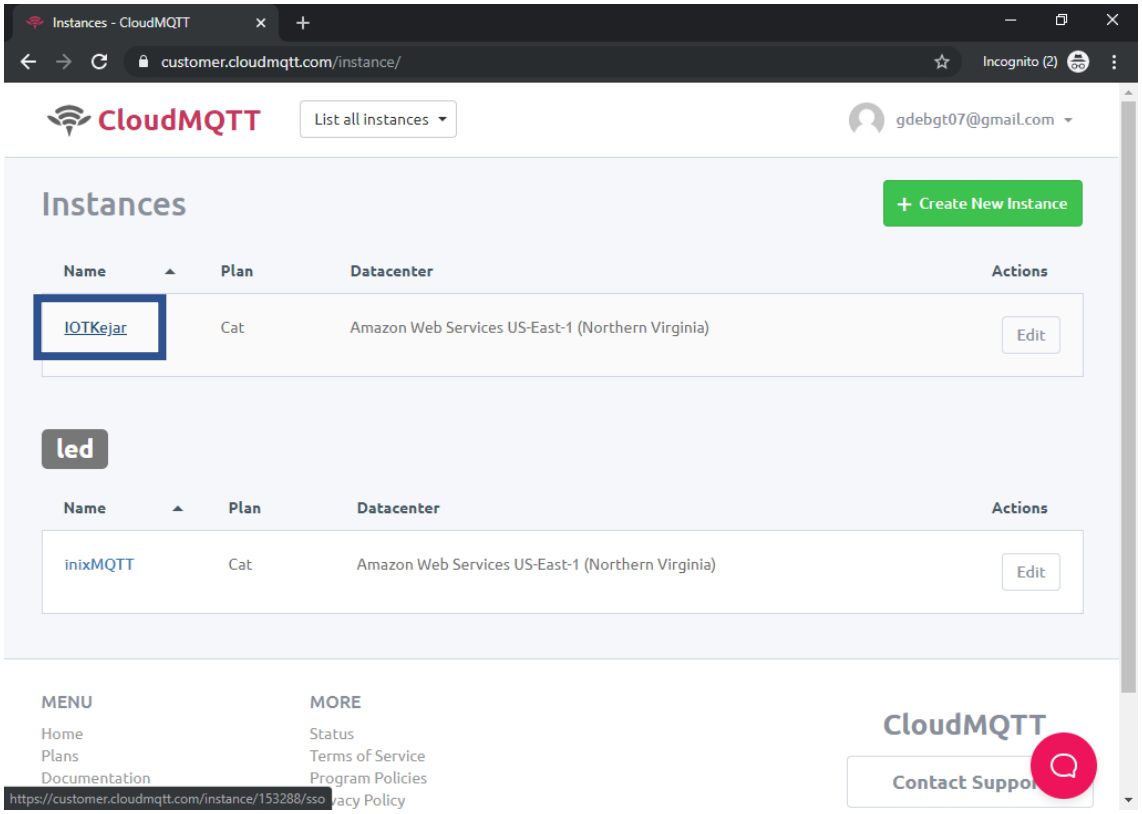
7. Lalu Klik Review



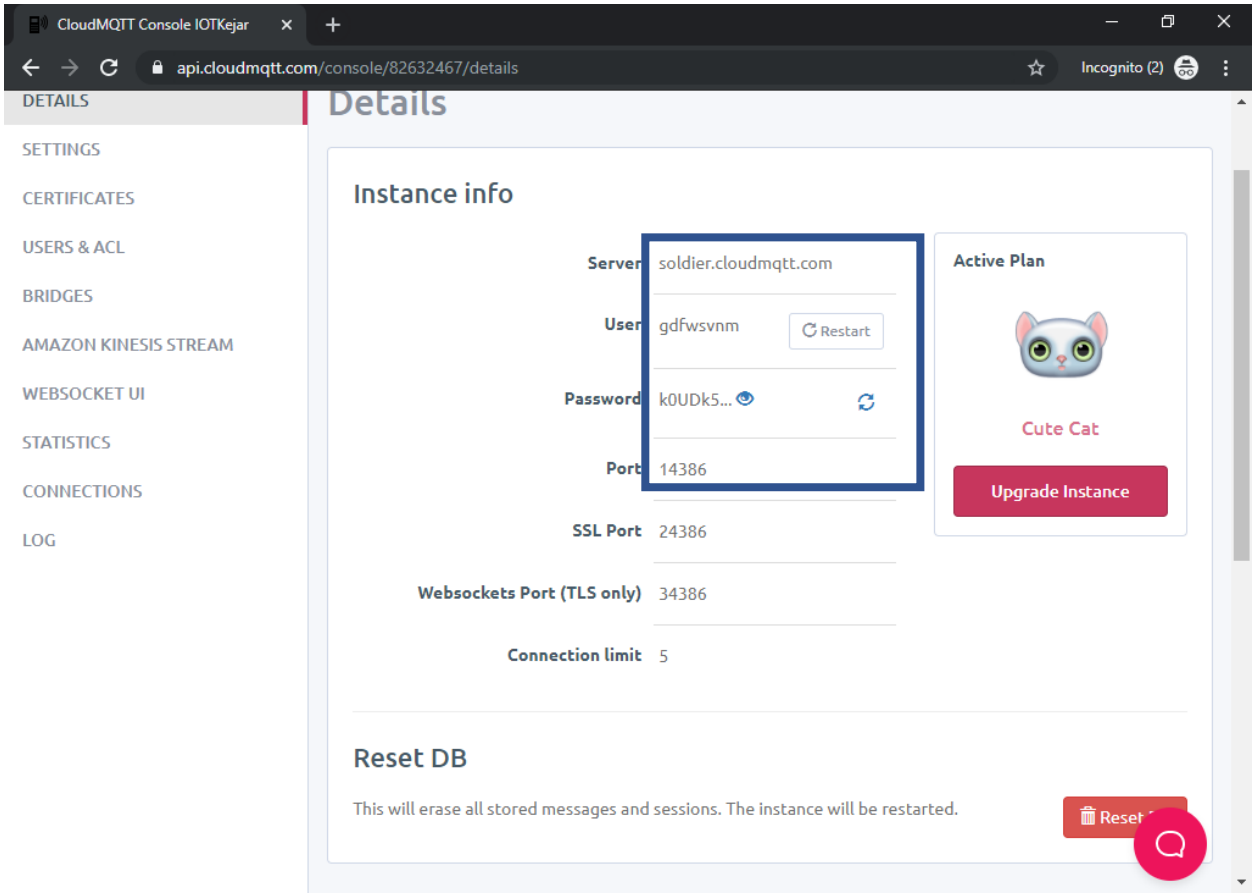
8. Lalu klik Create Instance



9. Klik instance yang baru saja di buat

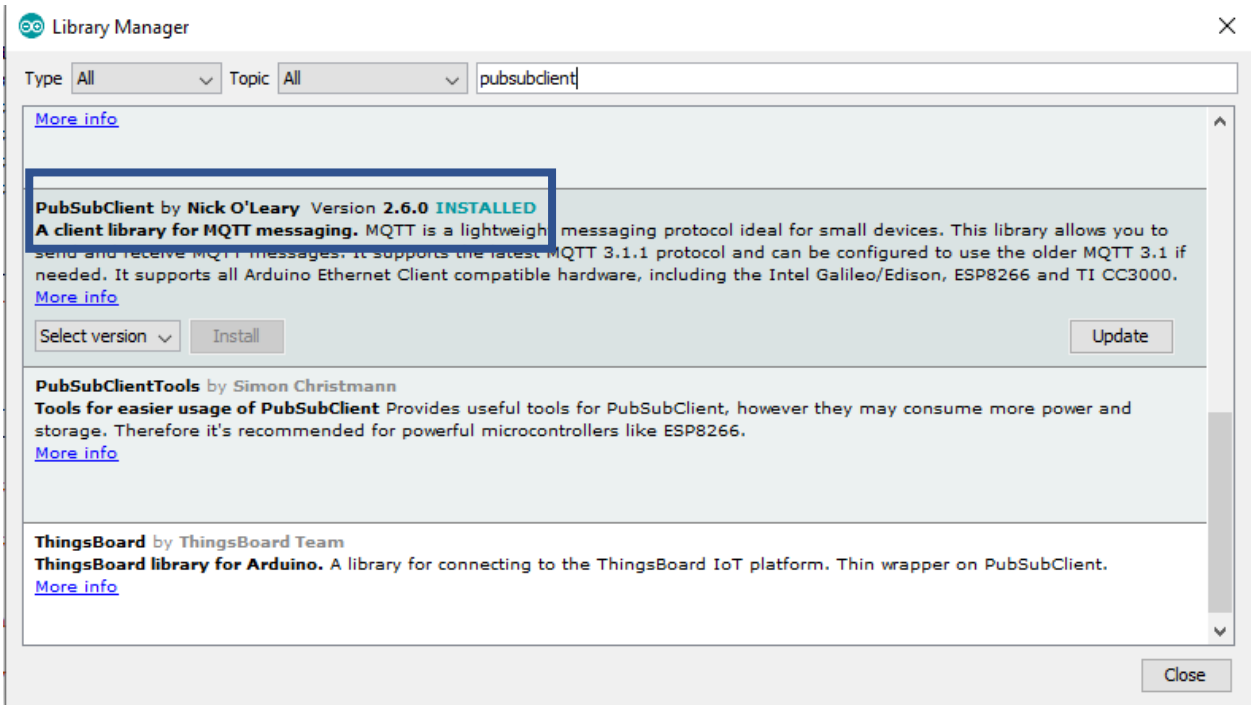


10. Catat user, password, port dan alamat mqtt broker yang akan digunakan



Setting Publisher

1. Buatlah rangkaian IoT Device dengan salah satu sensor (misalkan dalam percobaan ini digunakan sensor LDR)
2. Install library berikut



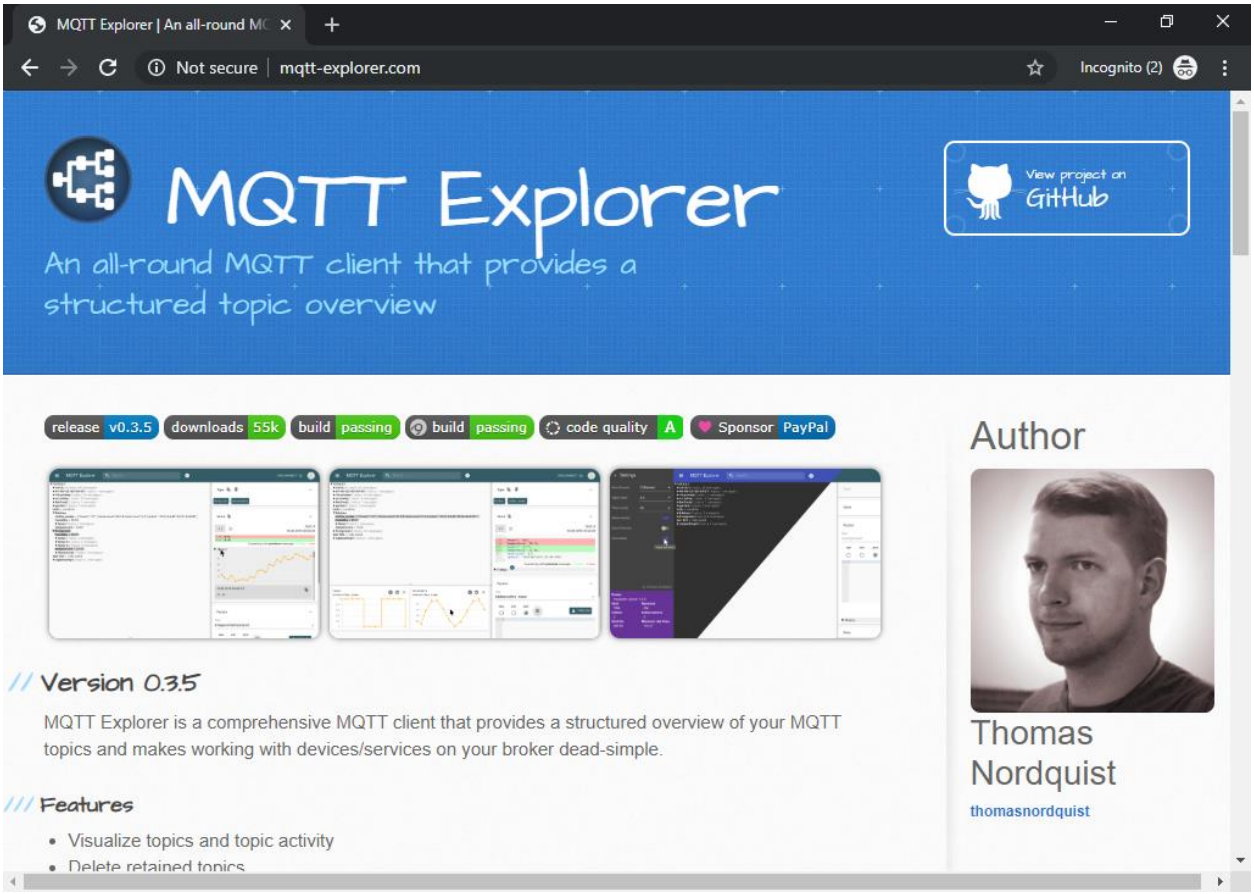
3. Masukkan sketch berikut

```
#include <WiFi.h>
#include <PubSubClient.h>

const char* ssid = "di isi dengan nama wifi";
const char* password =  "di isi dengan password wifi";
const char* mqttServer = "di isi sesuai mqtt broker";
const int   mqttPort = di isi sesuai mqtt broker;
const char* mqttUser = " di isi sesuai mqtt broker ";
const char* mqttPassword = " di isi sesuai mqtt broker ";
int sensor = 33;

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {

 Serial.begin(115200);
 WiFi.begin(ssid, password);
```

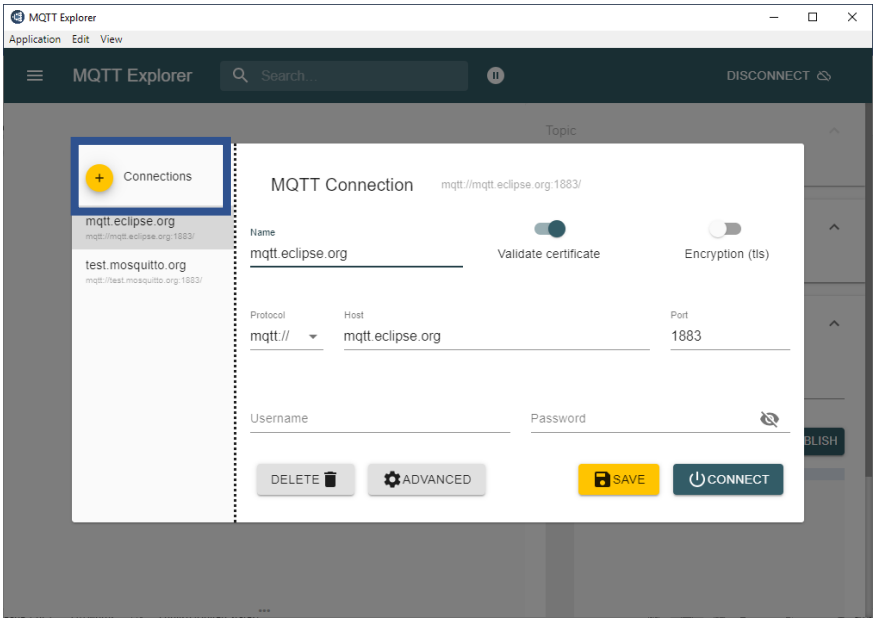
```
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
  }

  Serial.println("Connected to the WiFi network");

  client.setServer(mqttServer, mqttPort);

  while (!client.connected()) {
    Serial.println("Connecting to MQTT...");

    if (client.connect("ESP32Client", mqttUser, mqttPassword )) {

      Serial.println("connected");

    } else {

      Serial.print("failed with state ");
      Serial.print(client.state());
      delay(2000);

    }
  }
  pinMode(sensor,INPUT);

}

void loop() {
  client.loop();
  int SensorLDR=analogRead(sensor);
  char hasil[4];
  dtostrf(SensorLDR, 4, 0, hasil);
  Serial.println(SensorLDR);
  client.publish("cahaya", hasil);
  delay (5000);
}
```

Setting Subcsriber di Laptop

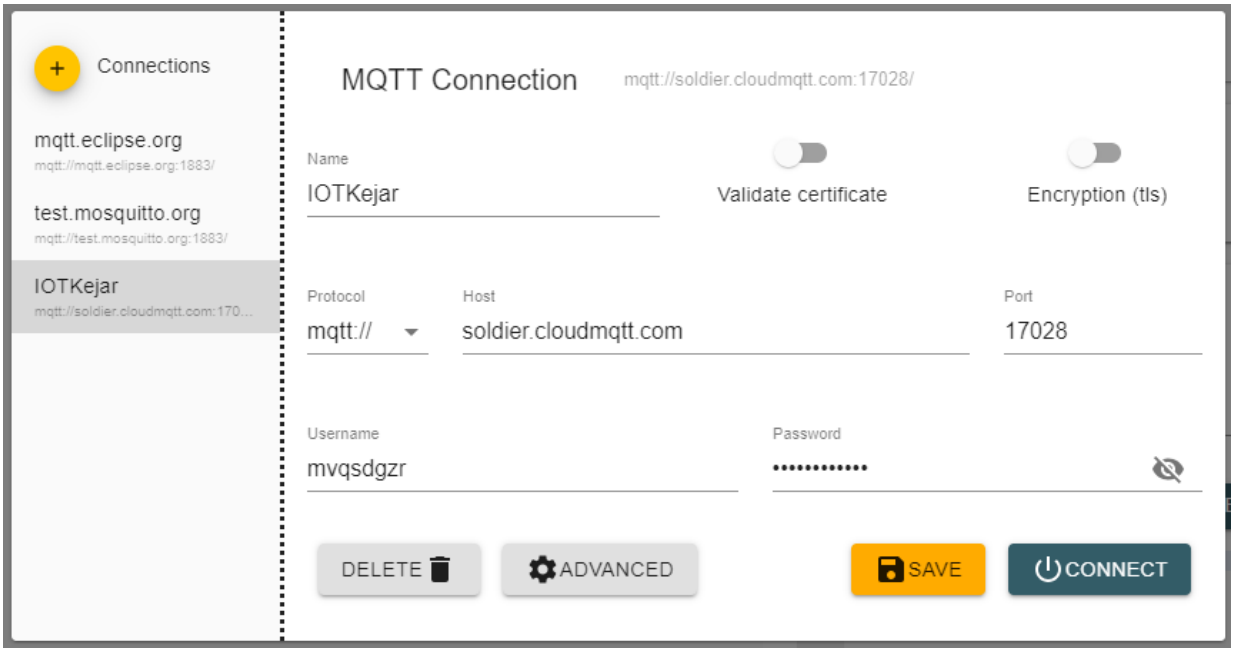
1. Download dan install applikasi MQTTExplorer dari halaman web http://mqtt-explorer.com/



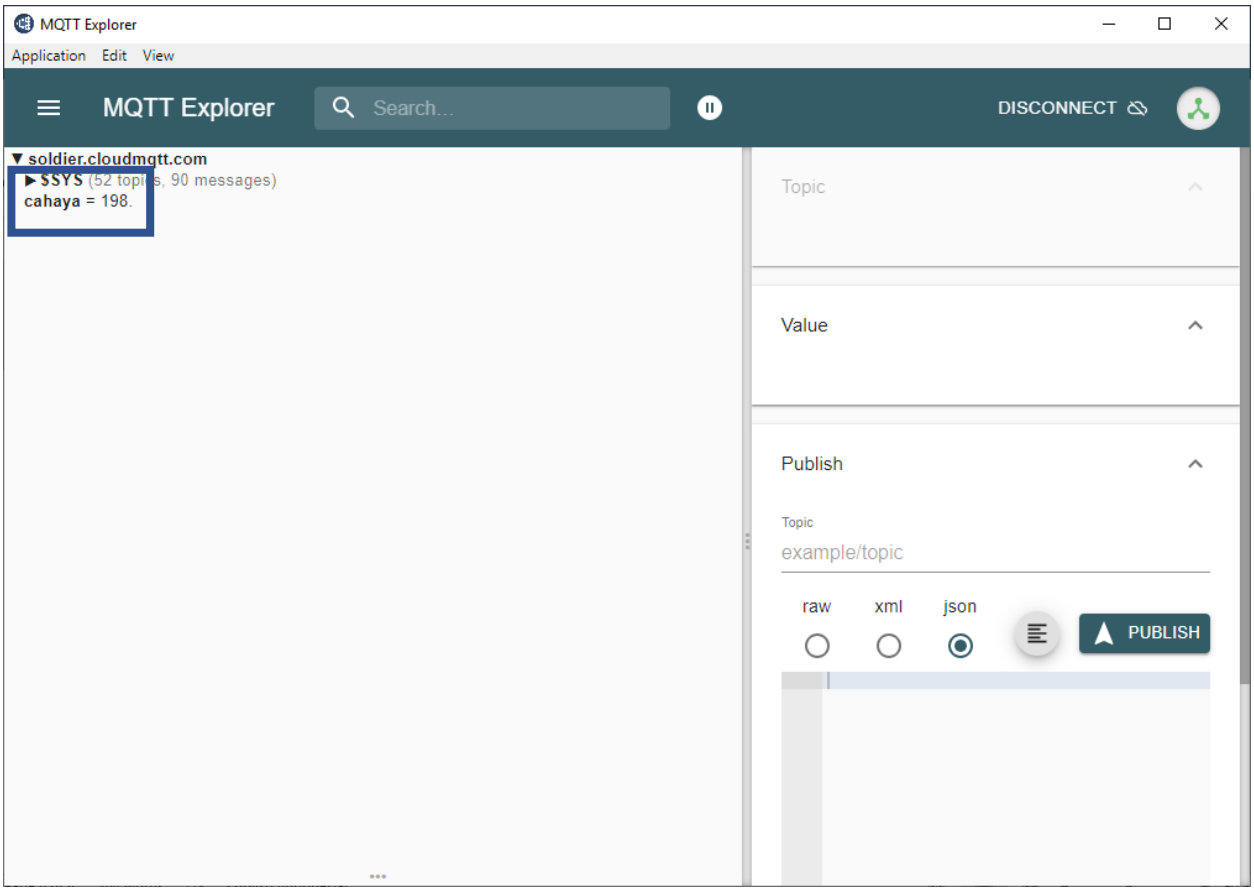
2. Tambahkan koneksi baru



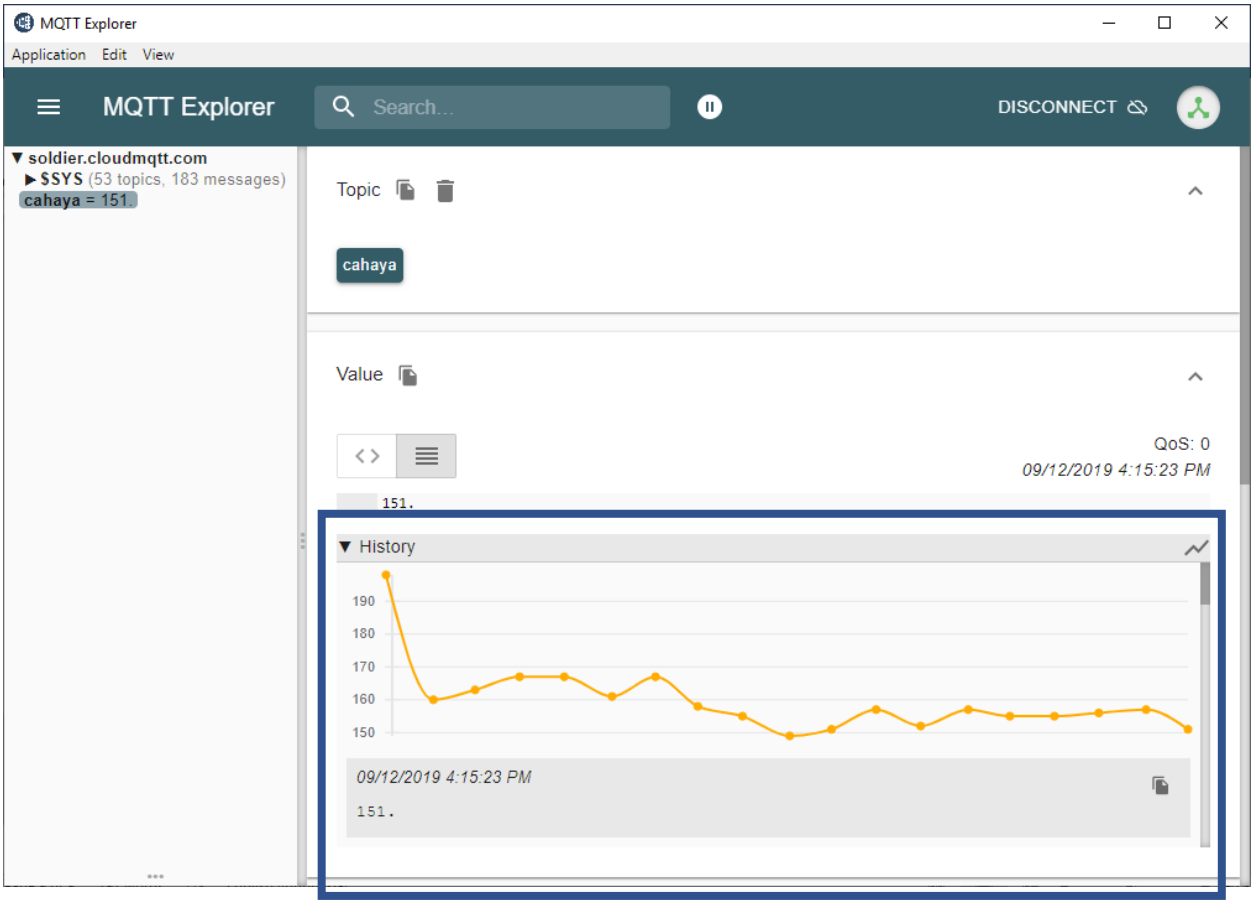
3. Masukkan semua informasi yang di butuhkan, lalu klik connect



4. Klik topik yang kita telah buat

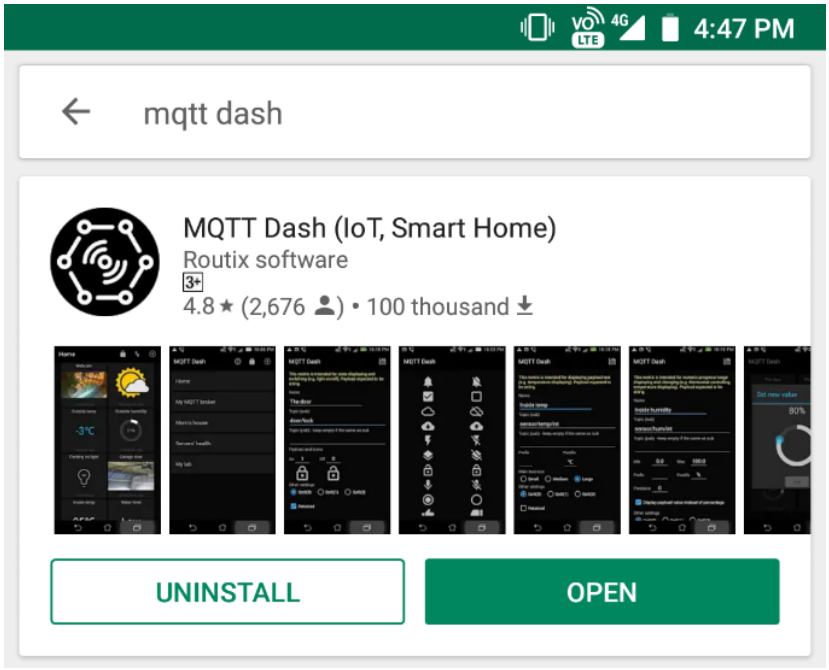


5. Jika ingin melihat hasil dan grafiknya, klik history

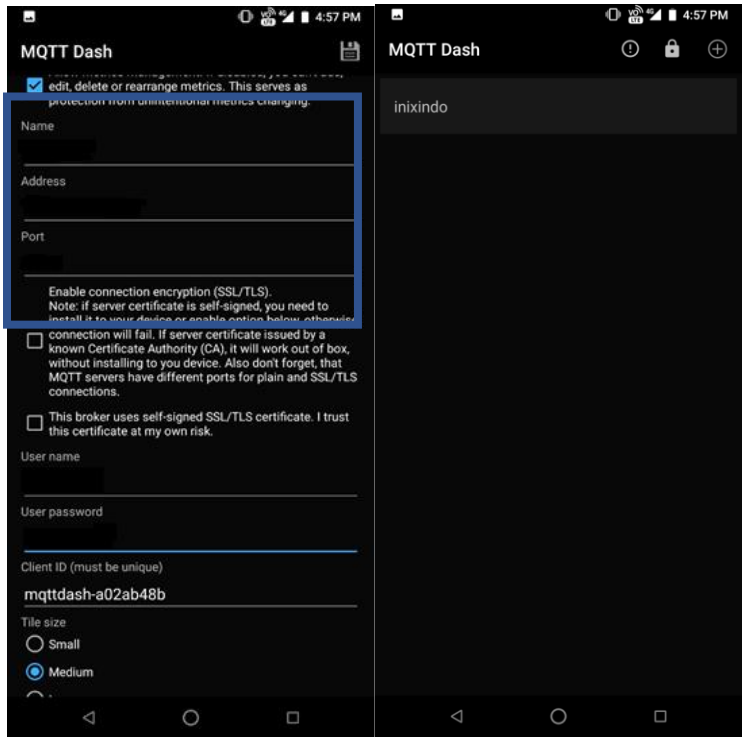


Setting Subcsriber di Handphone

1. Install MQTT Client (dalam lab ini yang akan digunakan adalah MQTTDash)



2. Masukkan informasi MQTTBroker (nama bisa di isi bebas)



3. Tambahkan topik dengan tipe text, tulis nama (bebas) dan topik (sesuai topik yang di buat), lalu save

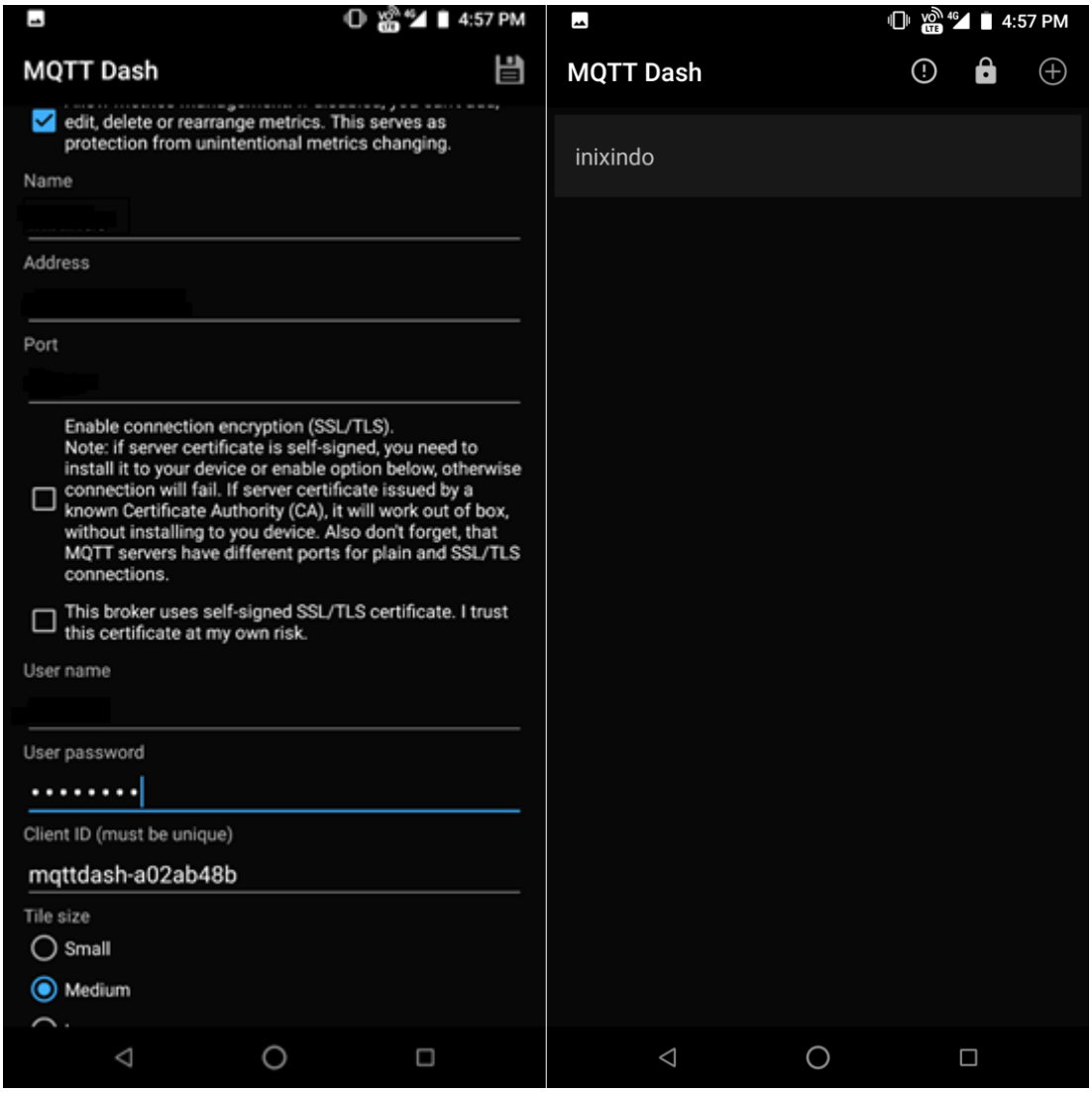
ESP 32 sebagai Publisher

```
#include <WiFi.h>
#include <PubSubClient.h>


const char* ssid = "di isi dengan ssid wifi";
const char* password = "diisi dengan password wifi";
const char* mqtt_server = "diisi dengan alamat broker";
const int   mqtt_port   = <portmqtt>;
const char* mqtt_user   = "username mqtt";
const char* mqtt_pass   = "password mqtt";

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;

void setup_wifi() {

  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
  if (strcmp(topic,"led1")==0){
    for (int i=0;i<length;i++) {
    char receivedChar = (char)payload[i];
    if (receivedChar == '1')
      digitalWrite(BUILTIN_LED, HIGH);
```

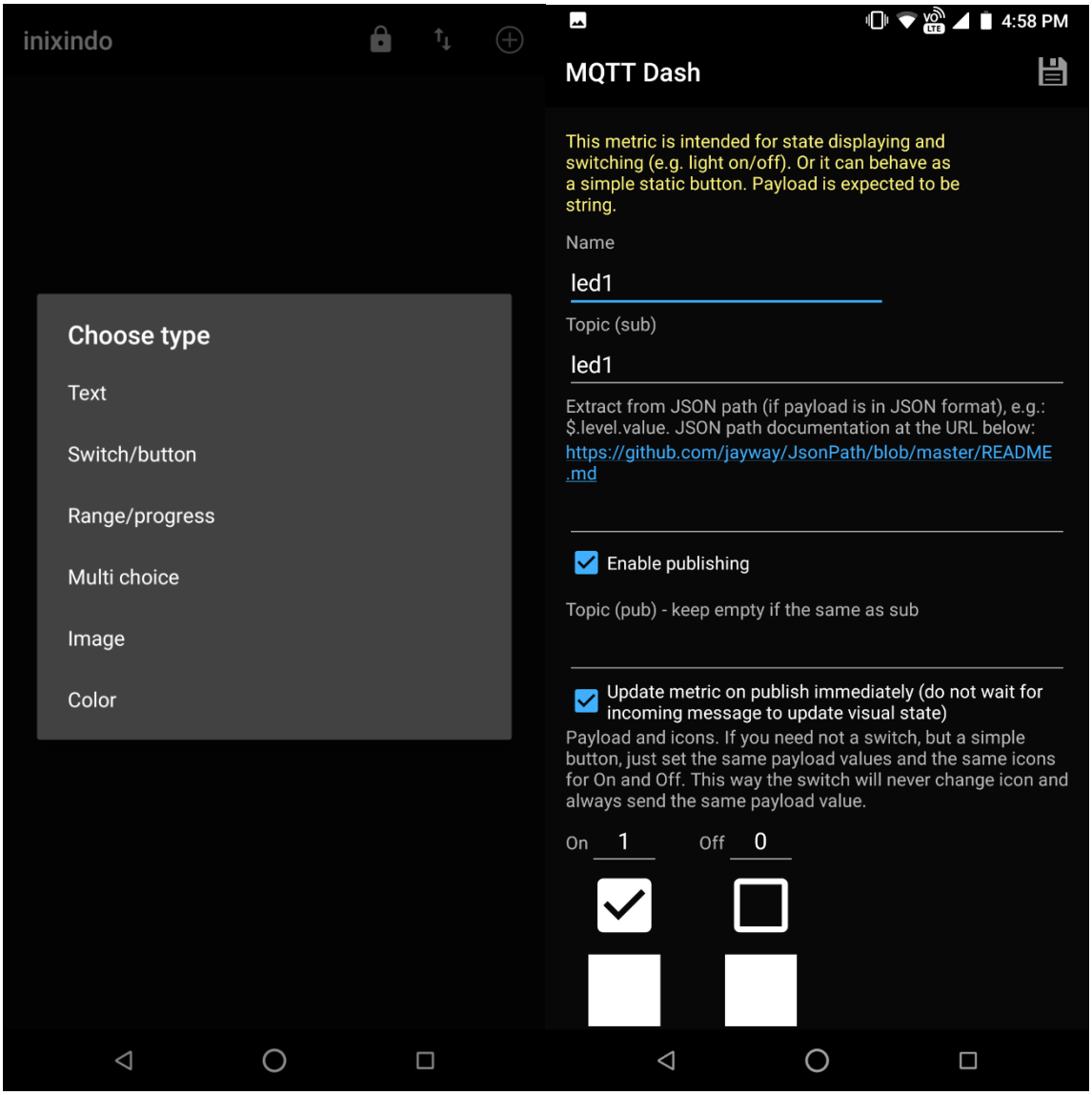
```
    if (receivedChar == '0')
      digitalWrite(BUILTIN_LED, LOW);
    }
  }
}

void reconnect() {

  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");

    String clientId = "ESP32Client-";
    clientId += String(random(0xffff), HEX);

    if (client.connect(clientId.c_str(),mqtt_user,mqtt_pass)) {
      Serial.println("connected");

      client.subscribe("led1");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");

      delay(5000);
    }
  }
}

void setup() {
  pinMode(BUILTIN_LED, OUTPUT);
  Serial.begin(9600);
  setup_wifi();
  client.setServer(mqtt_server, mqtt_port);
  client.setCallback(callback);
}

void loop() {

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

}
```

Setting MQTTDash

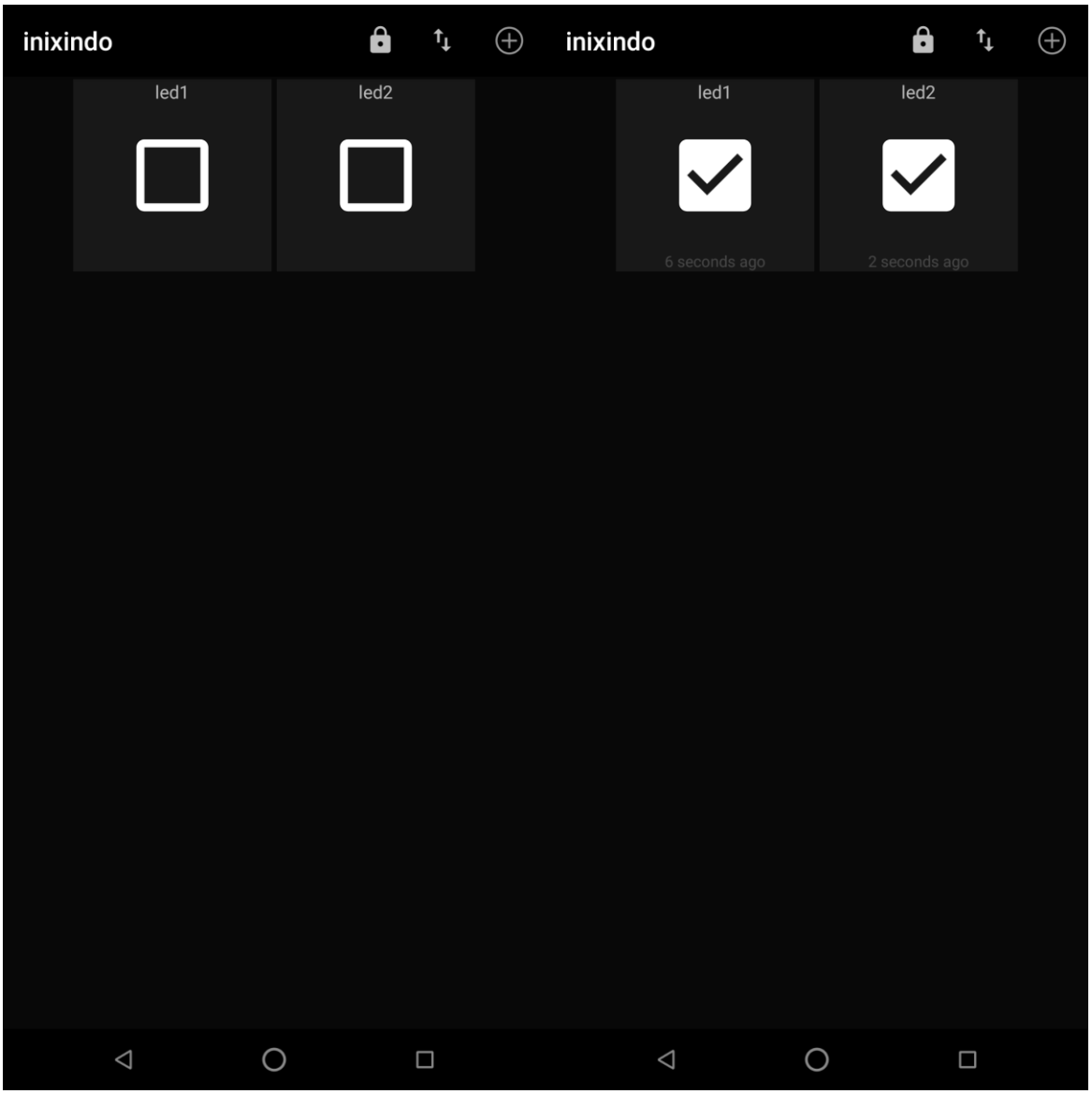
- Masukkan alamat broker, username dan password



- Tambahkan topik sesuai dengan jenis topik yang akan di gunakan


inixindo
Choose type
Text
Switch/button
Range/progress
Multi choice
Image
Color
4:58 PM
MQTT Dash
This metric is intended for state displaying and switching (e.g. light on/off). Or it can behave as a simple static button. Payload is expected to be string.
Name
led1
Topic (sub)
led1
Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md
Enable publishing
Topic (pub) - keep empty if the same as sub
Update metric on publish immediately (do not wait for incoming message to update visual state)
Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.
On 1
Off 0

- Test


inixindo
led1
led2
inixindo
led1
led2
6 seconds ago
2 seconds ago