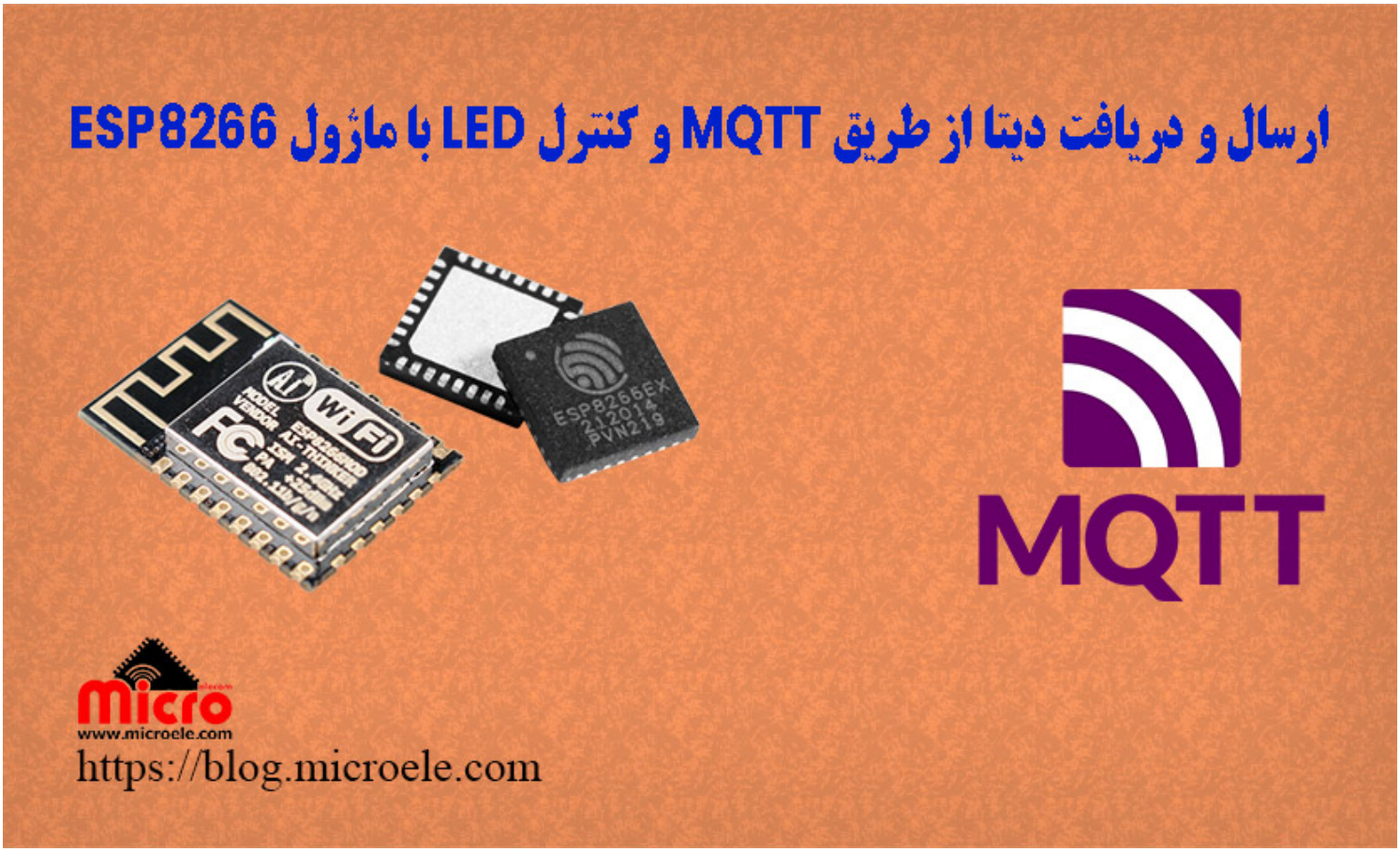
ارسال و دریافت دیتا از طریق MQTT و کنترل LED با ماژول ESP8266



تاریخ انتشار۱۴ تیر, ۱۴۰۲ توسط سید حسین سلطانی

سلام و درود خدمت همراهان همیشگی مایکروالکام. در مطالب قبلی از سری مطالب مربوط به سری آموزش های ESP8266 به معرفی سریال مانیتور تحت وب پرداخته شد. در این مطلب به بررسی کامل ارسال و دریافت دیتا/پیام با استفاده از پروتکل MQTT و بروکر رایگان HiveMQ و همچنین کنترل GPIO ماژول ESP8266 پرداخته خواهد شد. پس با من تا انتهای مطلب همراه باشید. همچنین شما میتونید سایر مطالب من رو از این قسمت دنبال کنید.

مقدمه

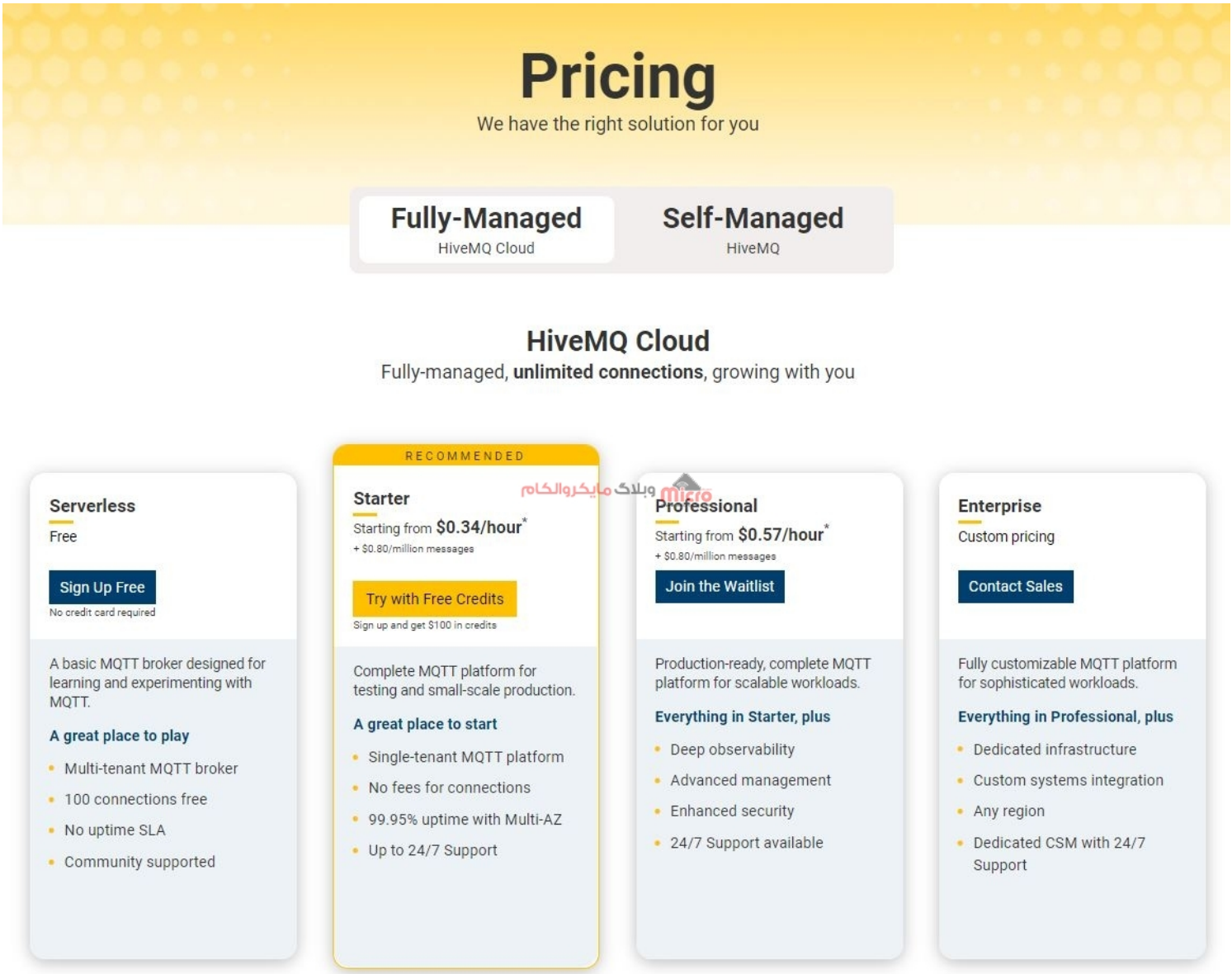
در مطالب قبلی به معرفی و موارد استفاده پروتکل MQTT پرداخته شد. در این مطلب بصورت کامل نحوه استفاده از آن جهت تبادل دیتا (ارسال و دریافت) بررسی شده است. می‌توان با ارتقا و ویرایش سورس کد این آموزش آن را شخصی سازی کرده و متناسب با پروژه خود از آن استفاده کنید. در این مطلب علاوه بر بحث ارسال و دریافت دیتا، یک بروکر (MQTT Brocker) رایگان که استفاده از آن نیز راحت می‌باشد معرفی گردیده است. همچنین علاوه بر بررسی تبادل دیتا در ESP8266، از یک نرم افزار کلاینت با قابلیت نصب روی ویندوز استفاده شده است که مضاف بر رصد آنلاین دیتا در سایت اصلی، می‌توان از آن جهت ارسال و دریافت نیز استفاده کرد.

بروکر MQTT Brocker

سایت و ارائه کنندگانی در سراسر اینترنت وجود دارند که خدماتی که شامل پروتکل MQTT است را ارئه می‌دهند. لذا طریقه کلی استفاده از آنها تقریبا مشابه و دارای روش مشابهی می‌باشند. برای این منظور از بروکر HiveMQ استفاده شده است. اولین قدم برای استفاده از آن ایجاد یک اکانت بوده تا بتوانیم از Cloud آن استفاده کنیم.

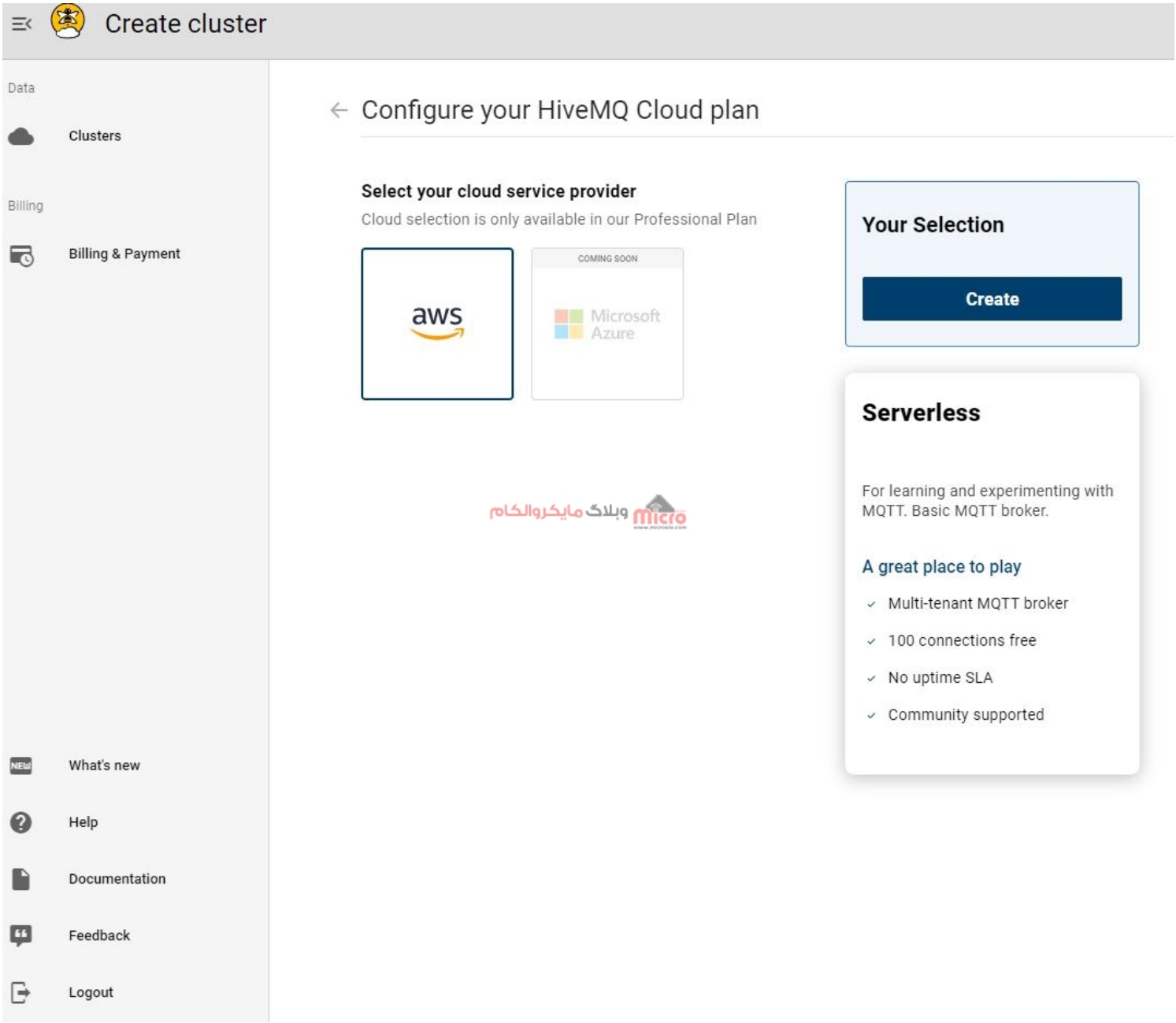
ثبت اکانت در بروکر HiveMQ

در ابتدا وارد سایت HiveMQ شده و از بالا سمت راست گزینه Get Started را بزنید. در صفحه جدید تعرفه و قیمت های استفاده از آن را نمایش می‌دهد که گزینه Serverless که رایگان است را انتخاب کنید. در این قسمت گزینه Sign up free را انتخاب و اکانت خود را ایجاد نمایید.



بروکر *MQTT HiveMQ*

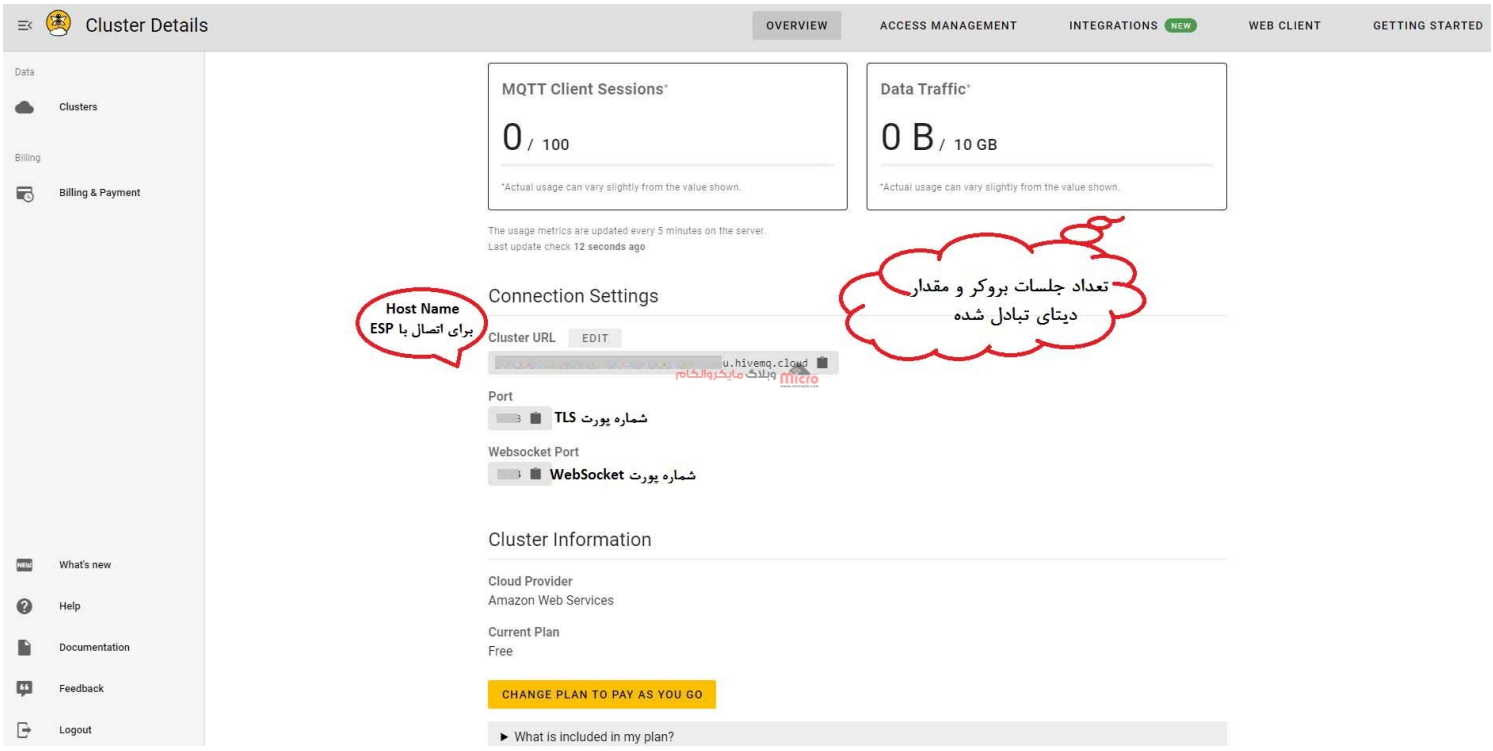
پس از ورود به اکانت خود از بخش Cluster با کلیک روی New Cluster یک خوشه جدید ایجاد کنید. در بخش جدید باز شده مجددا از بخش Serverless، گزینه Get Start را انتخاب کنید. مشابه تصویر زیر AWS را انتخاب و گزینه Creat را کلیک می‌کنیم.



ایجاد *Cluster* در بروکر *MQTT*

حال به منوی Cluster هدایت شده یا می‌توانید با انتخاب آن خوشه هایی که ایجاد کرده اید را مشاهده نمایید. با انتخاب Manage Cluster وارد جزئیات آن شده، با انتخاب گزینه Access Management یک username و password را تعریف کرده تا

بتوانیم توسط ESP8266 یا میکروکنترلر خود به آن متصل شده و تبادل دیتا نماییم. با انتخاب Web Client می‌توانیم به اکانت دسترسی داشته و با یک topic جلسه یا Session برقرار کرده و دیتا ارسال یا دیتای دریافتی را مشاهده نماییم.



جزئیات *Cluster* در بروکر *MQTT copy*

برنامه نویسی جهت اتصال به بروکر MQTT با ESP8266

سورس کد این بخش جهت استفاده و برنامه نویسی ماژول ESP8266 می‌باشد. برای ماژول ESP32 نیز مشابه بوده و نیاز به ویرایش خیلی جزئی چند قسمت کد ها دارد. در ابتدا نیاز است که کتابخانه PubSubClient برای اتصال و تبادل دیتا از طریق پروتکل MQTT نصب شود. برای دانلود و نصب آن از این لینک استفاده کنید.

نکته: برای ارتباط با سرور به چند مورد اصلی نیاز است که در ادامه به آن اشاره شده است (این موارد در بخش Overview اکانت خود قابل مشاهده است). همچنین دقت شود که نوع برقراری ارتباط با سرور بر بستر TLS بوده و باید ارتباط برقرار شده ماژول نیز ایمن باشد. برای این منظور از کتابخانه WiFiClientSecure استفاده شده و Certificate مورد نیاز سرور نیز در سورس کد آمده است.

- Server: همان آدرس یکتایی که سرور به شما اختصاص داده است.
- username: نام کاربری معرفی شده در Access Management
- password: رمز عبور معرفی شده در Access Management
- شماره پورت: شماره پورت TLS

معرفی کردن کتابخانه و متغییر های لازم

در ابتدای کد ها کتابخانه های مورد نیاز را معرفی کرده و از آنجا که اتصال به بروکر MQTT مبتنی بر پروتکل TLS می‌باشد، بایستی Certificate آن نیز در کد ها درج شده باشد. برای همین از متغیر PROGMEM root_ca* استفاده شده است. همچنین از کتابخانه WiFiClientSecure نیز به همین منظور استفاده شده است.

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <WiFiClientSecure.h>

const int led = 2;
const char* ssid = "SSID";
const char* password = "Password";

const char* mqtt_server = "********.eu.es.cloud";
const char* mqtt_username = "username";
const char* mqtt_password = "password";
const int mqtt_port = 8883;

WiFiClientSecure espClient;
PubSubClient client(espClient);

unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];
```

```
//root certificate
static const char *root_ca PROGMEM = R"EOF(
                                                -----BEGIN CERTIFICATE-----
MIIFazCCA1OgAwIBAgIRAIIQz7DSQONZRGPgu2OCiwAwDQYJKoZIhvcNAQELBQAw
TzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2Vh
cmNoIEdyb3VwMRUwEwYDVQQDEwxJU1JHIFJvb3QgWDEwHhcNMTUwNjA0MTEwNDM4
WhcNMzUwNjA0MTEwNDM4WjBPMQswCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJu
ZXQgU2VjdXJpdHkgUmVzZWFyY2ggR3JvdXAxFTATBgNVBAMTDElTUkcgUm9vdCBY
MTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBAK3oJHP0FDfzm54rVygc
h77ct984kIxuPOZXoHj3dcKi/vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+
0TM8ukj13Xnfs7j/EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U
A5/TR5d8mUgjU+g4rk8Kb4Mu0UlXjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW
T8KOEUt+zwvo/7V3LvSye0rgTBIlDHCNAymg4VMk7BPZ7hm/ELNKjD+Jo2FR3qyH
B5T0Y3HsLuJvW5iB4YlcNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC
B5iPNgiV5+I3lg02dZ77DnKxHZu8A/lJBdiB3QW0KtZB6awBdpUKD9jf1b0SHzUv
KBds0pjBqAlkd25HN7rOrFleaJ1/ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn
OlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEvzG6uBQE3xDk3SzynTn
jh8BCNAw1FtxNrQHusEwMFxIt4I7mKZ9YIqioymCzLq9gwQbooMDQaHWBfEbwrbw
qHyGO0aoSCqI3Haadr8faqU9GY/rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI
rU7m2Ys6xt0nUW7/vGT1M0NPAgMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV
HRMBAf8EBTADAQH/MB0GA1UdDgQWBBR5tFnme7bl5AFzgAiIyBpY9umbbjANBgkq
hkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkgJykIrGF1XIpu+ILlaS/V9lZL
ubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfFQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ
3BebYhtF8GaV0nxvwuo77x/Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/KK
NFtY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5
ORAzI4JMPJ+GslWYHb4phowim57iaztXOoJwTdwJx4nLCgdNbOhdjsnvzqvHu7Ur
TkXWStAmzOVyyghqpZXjFaH3pO3JLF+l+/+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC
jNPElpzVmbUq4JUagEiuTDkHzsxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc
oyi3B43njTOQ5yOf+1CceWxG1bQVs5ZufpsMljq4Ui0/1lvh+wjChP4kqKOJ2qxq
4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/9yi0GE44Za4rF2LN9d11TPA
mRGunUHBcnWEvgJBQl9nJEiU0Zsnvgc/ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d
emyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/+jh5Xos1AnX5iItreGCc=
```

```
-----END CERTIFICATE-----
)EOF";
```

اتصال ESP8266 به مودم

برای برقراری ارتباط با بروکر MQTT نیاز به اتصال به مودم و اینترنت داریم. در بالا در متغیر های ssid و password نام و پسورد مودم یا روتر خود را جایگزین کنید. از تابع زیر جهت اتصال به مودم استفاده شده است.

```
void setup_wifi() {
                    //delay(10);
                    Serial.print("\nConnecting to ");
                    Serial.println(ssid);
                    WiFi.mode(WIFI_STA);
                    WiFi.begin(ssid, password);
                    while (WiFi.status() != WL_CONNECTED) {
                                                            delay(500);
Serial.print(".");
                                                          }
                    //randomSeed(micros());
                    Serial.println("\nWiFi connected\nIP address: ");
                    Serial.println(WiFi.localIP());
                  }
```

تابع اتصال ESP8266 به بروکر MQTT

پس از اتصال به اینترنت، باید به سرور یا بروکر MQTT خود متصل شویم. برای این کار نیاز به چند پارامتر اصلی داریم که در ابتدای کد ها معرفی شده است. این موارد شامل آدرس سرور، پورت، یوزرنیم و پسورد می‌باشد. در کد زیر بجای Client_ID می‌توانیم یک نام دلخواه وارد نماییم.

```
void reconnect() {
                    while (!client.connected()) {
```

```
                                                    Serial.print("Attempting
MQTT connection...");
                                                    // Attempt to connect
                                                    if
(client.connect("Client_ID", mqtt_username, mqtt_password)) {
Serial.println("connected");
client.subscribe("test"); //subscribe to topic
}
else {
Serial.print("failed, rc=");
Serial.print(client.state()); //MQTT connection status
Serial.println(" try again in 3 seconds"); //Wait 3 sec. before retrying
delay(5000);
}
                                                }
                }
```

دریافت دیتا از بروکر MQTT

علاوه بر ارسال دیتا به سرور از طریق پروتکل MQTT باید پیام های دریافتی که ارسال می‌شود را نیز بتوانیم دریافت کنیم. به همین منظور پس از اصطلاحا شرکت کردن در یک topic یا برقراری یک Session، پیام های آن را بوسیله تابع زیر دریافت خواهیم کرد. می‌توانیم با نوشتن یک شرط ساده، یک LED را روشن و خاموش کنیم. در شرط if کد زیر، از تابع strcmp استفاده شده است. چنانچه topic معرفی شده در آرگومان تابع هنگام فراخوانی با topic مورد نظر خود که در اینجا test نام دارد برابر بود، شرط اجرا خواهد شد.

```
void callback(char* topic, byte* payload, unsigned int length) {
                                                                    String
incommingMessage = "";
                                                                    for (int i
= 0; i < length; i++){
incommingMessage += (char)payload[i];
```

```
}
                                                                String
message = incommingMessage;
Serial.println("Incomming Message: " + message);
if(strcmp(topic, "test") == 0){
if(message == "on"){
Serial.println("LED is ON");
digitalWrite(led, LOW);
}
else if(message == "off"){
Serial.println("LED is OFF");
digitalWrite(led, HIGH);
}
}
                                                            }
```

ارسال دیتا به بروکر MQTT با ESP8266

برای ارسال دیتا ابتدا باید در topic مورد نظر عضو شده و پس از آن دیتای خود را ارسال نماییم.

```
void publishMessage(const char* topic, String payload , boolean retained){
if (client.publish(topic, payload.c_str(), true))
Serial.println("Message publised [" + String(topic) + "]: " + payload);
else
Serial.println("Unable to publishing!");
                                                                    }
```

با استفاده از کد زیر می‌توانیم در صورت نیاز خود در برنامه، پیام مد نظر خود را ارسال کنیم. به یاد داشته باشید بجای Your_topic، همان topic که قصد ارسال پیام به آن را دارید عینا وارد کنید. و بجای mqtt_message نیز پیام/دیتا دلخواه خود را وارد نمایید.

```
publishMessage("Your_topic", "mqtt_message", true);
```

تابع setup

در این تابع تنظیمات اولیه جهت راه اندازی، تنظیمات سریال، اتصال به مودم، فراهم کردن اتصال به بروکر MQTT و مواردی از این قبیل را انجام خواهیم داد.

```
void setup() {
                pinMode(led, OUTPUT);
                digitalWrite(2, HIGH);
                Serial.begin(115200);
                setup_wifi();
                #ifdef ESP8266
                  espClient.setInsecure();
                #else
                  espClient.setCACert(root_ca);      // enable this line and
the "certificate" code for secure connection
                #endif
                client.setServer(mqtt_server, mqtt_port); //setting server
and port
                client.setCallback(callback);
             }
```

تابع loop

در این تابع یک شرط قرار داده شده است که در صورت عدم اتصال یا قطع ارتباط با بروکر، مجددا تلاش به برقراری ارتباط می‌کند. می‌توانیم با افزودن delay مدت زمان آن را محدود نماییم.

```
void loop() {
              if (!client.connected()) reconnect(); // check if client is not
```

```
connected, try again
              client.loop();
              //delay(1000);
            }
```

سورس کد کامل

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <WiFiClientSecure.h>

const int led = 2;
const char* ssid = "SSID";
const char* password = "Password";

const char* mqtt_server = "********.eu.es.cloud";
const char* mqtt_username = "username";
const char* mqtt_password = "password";
const int mqtt_port = 8883;

WiFiClientSecure espClient;
PubSubClient client(espClient);

unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50) //تعیین مقدار باقر برای پیام
char msg[MSG_BUFFER_SIZE];

//root certificate
static const char *root_ca PROGMEM = R"EOF(
                                              -----BEGIN CERTIFICATE-----
MIIFazCCA1OgAwIBAgIRAIIQz7DSQONZRGPgu2OCiwAwDQYJKoZIhvcNAQELBQAw
```

```
TzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2Vh
cmNoIEdyb3VwMRUwEwYDVQQDEwxJU1JHIFJvb3QgWDEwHhcNMTUwNjA0MTEwNDM4
WhcNMzUwNjA0MTEwNDM4WjBPMQswCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJu
ZXQgU2VjdXJpdHkgUmVzZWFyY2ggR3JvdXAxFTATBgNVBAMTDElTUkcgUm9vdCBY
MTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBAK3oJHP0FDfzm54rVygc
h77ct984kIxuPOZXoHj3dcKi/vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+
0TM8ukj13Xnfs7j/EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U
A5/TR5d8mUgjU+g4rk8Kb4Mu0UlXjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW
T8KOEUt+zwvo/7V3LvSye0rgTBIlDHCNAymg4VMk7BPZ7hm/ELNKjD+Jo2FR3qyH
B5T0Y3HsLuJvW5iB4YlcNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC
B5iPNgiV5+I3lg02dZ77DnKxHZu8A/lJBdiB3QW0KtZB6awBdpUKD9jf1b0SHzUv
KBds0pjBqAlkd25HN7rOrFleaJ1/ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn
OlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEvzG6uBQE3xDk3SzynTn
jh8BCNAw1FtxNrQHusEwMFxIt4I7mKZ9YIqioymCzLq9gwQbooMDQaHWBfEbwrbw
qHyGO0aoSCqI3Haadr8faqU9GY/rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI
rU7m2Ys6xt0nUW7/vGT1M0NPAgMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV
HRMBAf8EBTADAQH/MB0GA1UdDgQWBBR5tFnme7bl5AFzgAiIyBpY9umbbjANBgkq
hkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkgJykIrGF1XIpu+ILlaS/V9lZL
ubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfFQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ
3BebYhtF8GaV0nxvwuo77x/Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/KK
NFtY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5
ORAzI4JMPJ+GslWYHb4phowim57iaztXOoJwTdwJx4nLCgdNbOhdjsnvzqvHu7Ur
TkXWStAmzOVyyghqpZXjFaH3pO3JLF+l+/+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC
jNPElpzVmbUq4JUagEiuTDkHzsxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc
oyi3B43njTOQ5yOf+1CceWxG1bQVs5ZufpsMljq4Ui0/1lvh+wjChP4kqKOJ2qxq
4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/9yi0GE44Za4rF2LN9d11TPA
mRGunUHBcnWEvgJBQl9nJEiU0Zsnvgc/ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d
emyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/+jh5Xos1AnX5iItreGCc=
                                          -----END CERTIFICATE-----
                                        )EOF";

//Connect to WiFi
```

```
void setup_wifi() {
                    //delay(10);
                    Serial.print("\nConnecting to ");
                    Serial.println(ssid);
                    WiFi.mode(WIFI_STA);
                    WiFi.begin(ssid, password);
                    while (WiFi.status() != WL_CONNECTED) {
                                                            delay(500);
Serial.print(".");
                                                          }
                    //randomSeed(micros());
                    Serial.println("\nWiFi connected\nIP address: ");
                    Serial.println(WiFi.localIP());
                  }

//Connect to MQTT Broker
void reconnect() {
                    while (!client.connected()) {
                                                  Serial.print("Attempting
MQTT connection...");
                                                  // Attempt to connect
                                                  if
(client.connect("Client_ID", mqtt_username, mqtt_password)) {
Serial.println("connected");
client.subscribe("test"); //subscribe to topic
}
else {
Serial.print("failed, rc=");
Serial.print(client.state()); //MQTT connection status
Serial.println(" try again in 3 seconds"); //Wait 3 sec. before retrying
delay(5000);
}
```

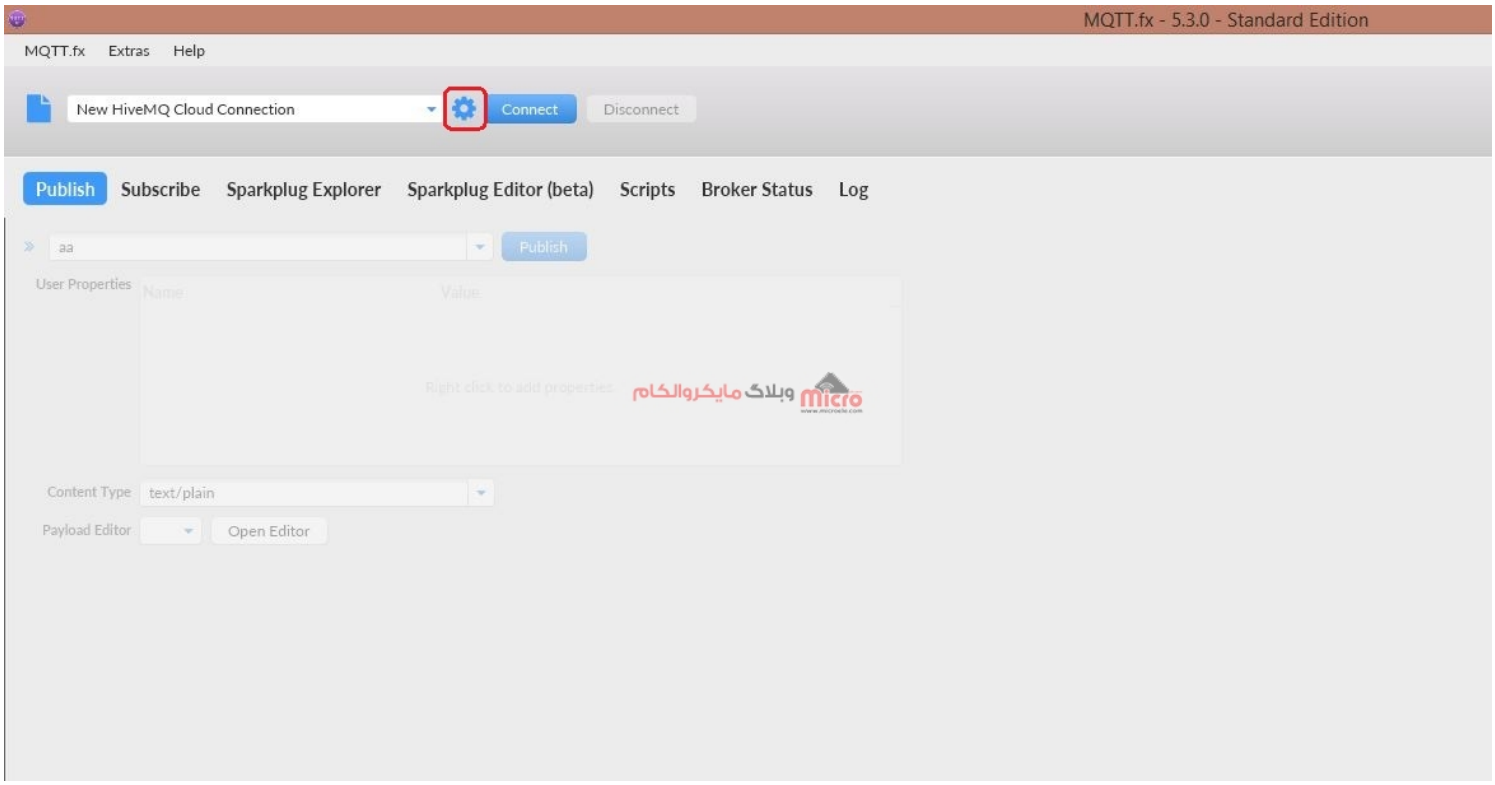
```
                                                    }
                }

//Receiving MQTT messages
void callback(char* topic, byte* payload, unsigned int length) {
                                                                    String
incommingMessage = "";
                                                                    for (int i
= 0; i < length; i++){
incommingMessage += (char)payload[i];
}
                                                                    String
message = incommingMessage;
Serial.println("Incomming Message: " + message);
if(strcmp(topic, "test") == 0){
if(message == "on"){
Serial.println("LED is ON");
digitalWrite(led, LOW);
}
else if(message == "off"){
Serial.println("LED is OFF");
digitalWrite(led, HIGH);
}
}
                                                                }

//Publishing MQTT Message
void publishMessage(const char* topic, String payload , boolean retained){
if (client.publish(topic, payload.c_str(), true))
Serial.println("Message publised [" + String(topic) + "]: " + payload);
else
Serial.println("Unable to publishing!");
```

```
                                                                    }


void setup() {
                pinMode(led, OUTPUT);
                digitalWrite(2, HIGH);
                Serial.begin(115200);
                setup_wifi();
                #ifdef ESP8266
                  espClient.setInsecure();
                #else
                  espClient.setCACert(root_ca);      // enable this line and
the "certificate" code for secure connection
                #endif
                client.setServer(mqtt_server, mqtt_port); //setting server
and port
                client.setCallback(callback);
             }

void loop() {
              if (!client.connected()) reconnect(); // check if client is not
connected, try again
              client.loop();
              //delay(1000);
            }
```

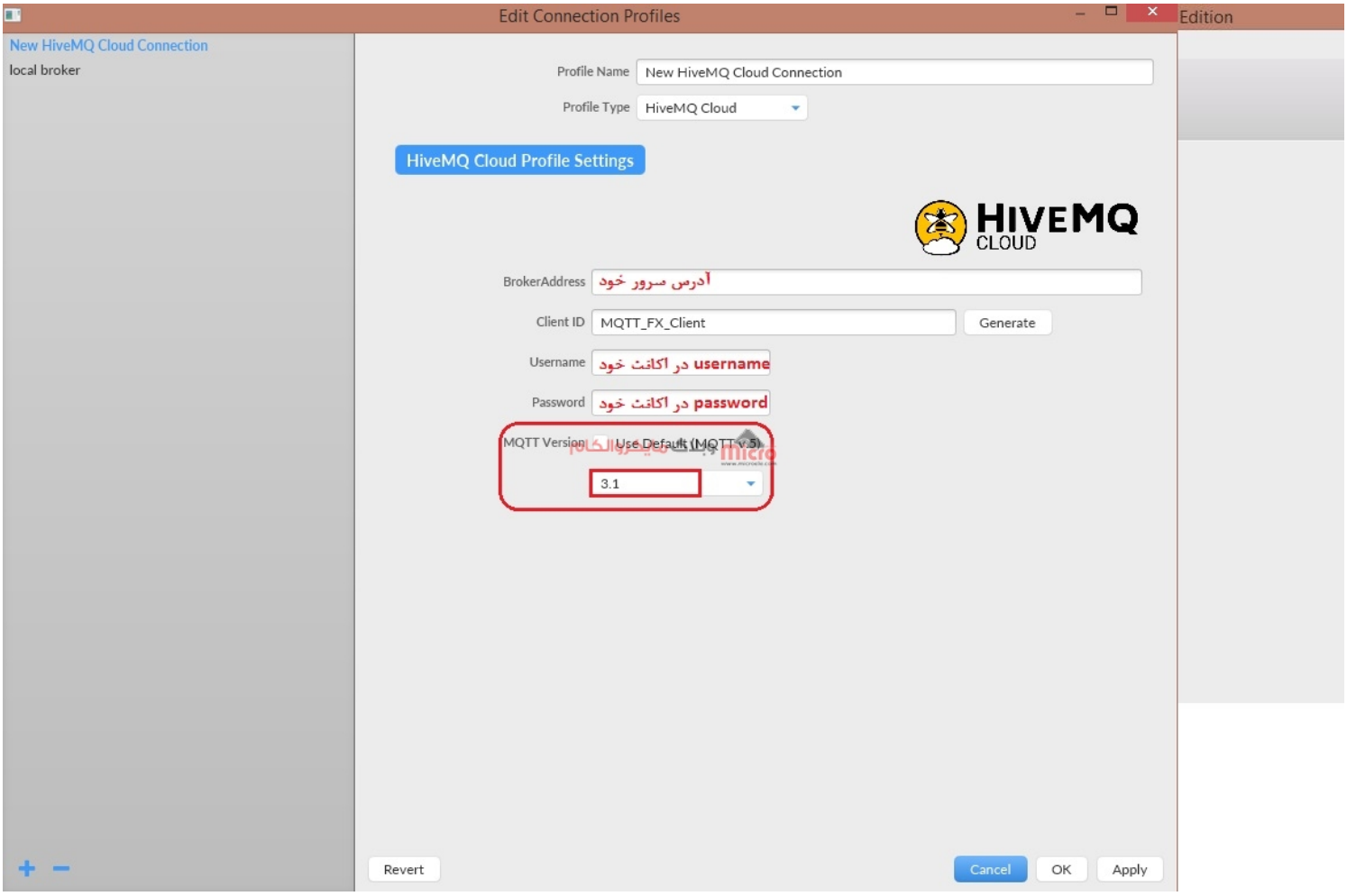
کلاینت MQTT برای ویندوز

می‌توانیم از طریق نرم افزار MQTT.Fx نیز از طریق ویندوز به بروکر متصل شده و تبادل دیتا نماییم. برای دانلود این نرم افزار از این لینک اقدام نمایید. دقت کنید پس از نصب نیاز به لایسنس دارید برای درخواست آن نیز از این لینک

اقدام کنید. پس از چند دقیقه فایل مربوطه ایمیل شده و آن را در notepad باز کرده و کامل کپی کرده و در پنجره باز شده هنگام باز کردن نرم افزار دقیقا paste نمایید. پس از وارد شدن به نرم افزار برروی چرخ دنده مشخص شده در تصویر زیر کلیک کنید. در پنجره باز شده مطابق با تصویر دوم اطلاعات خواسته شده را وارد کنید. نهایتا برروی OK کلیک کرده و به صفحه اصلی باز خواهید گشت. اینبار برروی دکمه Connect که در کنار آیکن چرخ دنده است کلیک کرده و منتظر بمانید تا به بروکر متصل شوید.

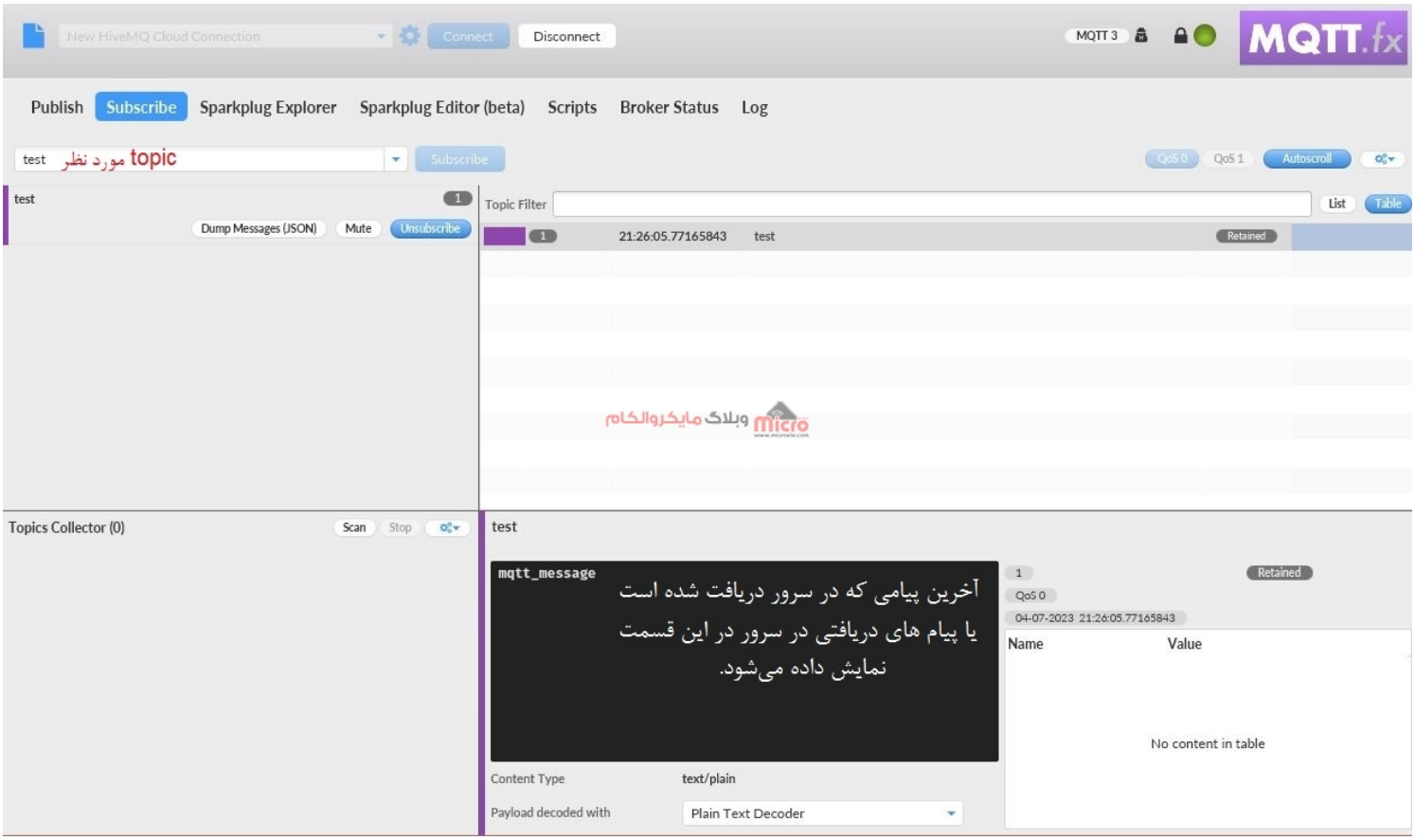


نرم افزار *MQTT.Fx*



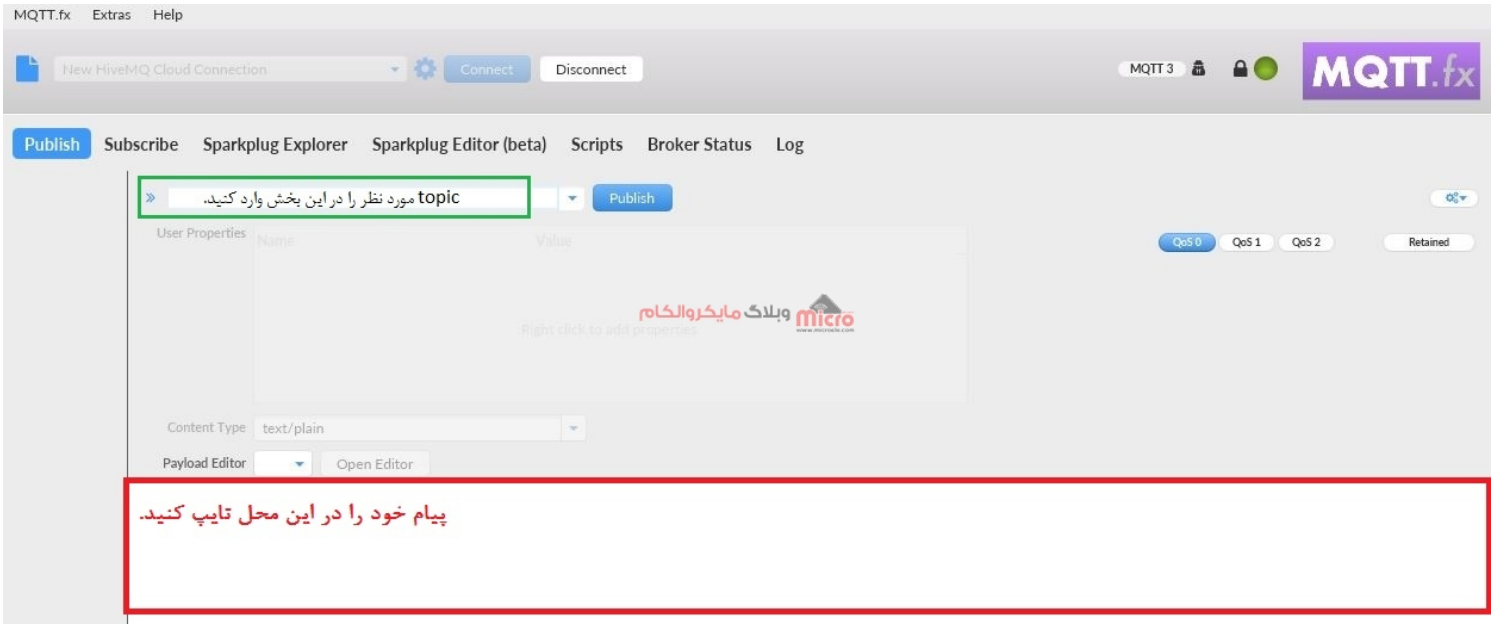
تنظیمات نرم افزار *MQTT.FX*

پس از اتصال موفقیت آمیز، از قسمت subscribe می‌توانیم در topic مورد نظر عضو شده و دیتا های دریافتی را مشاهده نماییم. پس از وارد کردن نام تاپیک بصورت صحیح، برروی subscribe کلیک کنید. پس از موفقیت آمیز بودن همه موارد، تصویر نرم افزار همانند تصویر زیر خواهد شد.



اتصال به یک *topic* در *MQTT.Fx*

در منوی Publish نیز می‌توانیم به topic مورد نظر، دیتا ارسال نماییم. در تصویر زیر در محل مشخص شده، topic مد نظر را وارد و سپس روی گزینه Publish کلیک نمایید. پس از آن می‌توانید در محلی که در تصویر زیر با کادر قرمز مشخص شده ، پیام خود را وارد کرده و نهایتا دکمه publish را زده تا دیتا ارسال گردد. می‌توانیم در بروکر یا سرور خود، پیام های دریافتی را در قسمت Cluster و زیر قسمت Web Client مشاهده نماییم.



ارسال *(Publish)* دیتا به تاپیک مورد نظر توسط *MQTT.Fx*

نتیجه گیری

در این مطلب به معرفی و نحوه ایجاد یک Session در بروکر MQTT پرداخته شد. همچنین روش تبادل دیتا و ارسال و دریافت از طریق پروتکل MQTT جهت کنترل پایه های GPIO ماژول ESP8266 بررسی شد. با استفاده از کد هایی که در این مطلب قرار دارد براحتی می‌توانیم متناسب با پروژه IoT خود آن ها را ویرایش و متناسب با پروژه و کاربرد خود آن را در بستر اینترنت اشیا پیاده سازی کنیم.

امیدوارم از این آموزش کمال بهره را برده باشید. در صورتی که هرگونه نظر یا سوال داشتید درباره این آموزش لطفا اون رو در انتهای همین صفحه در قسمت دیدگاه ها قرار بدید. در کوتاه ترین زمان ممکن به اون ها پاسخ خواهم داد. اگر این مطلب براتون مفید بود، اون رو حتما به اشتراک بگذارید. همینطور میتونید این آموزش را توی اینستاگرام با هشتگ microelecom# به اشتراک بگذارید و پیج مایکروالکام (microelecom@) رو هم منشن کنید.