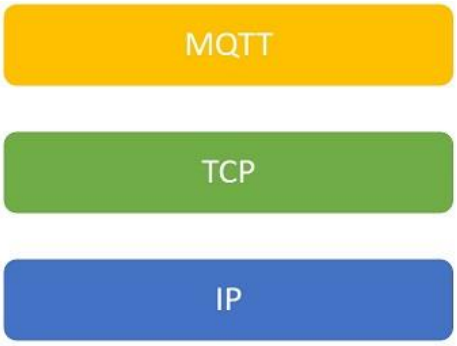
Labolatorium Sisk – MQTT

Wykonali: Radosław Feiglewicz, Maksymilian Mruszczak

1. Opis protokołu

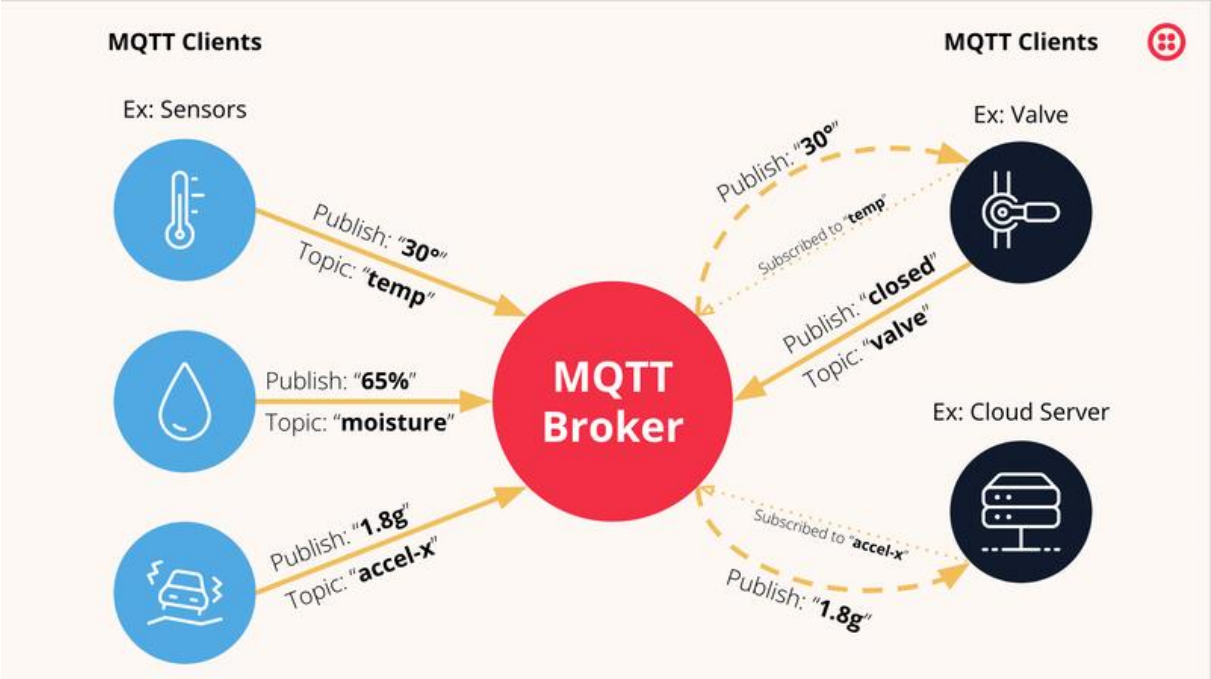
MQTT (**MQ Telemetry Transport) -** lekki protokół transmisji danych oparty o wzorzec publikacja/subskrybcja pracujący na szczycie warstwy TCP/IP. Wykorzystywany przede wszystkim w aplikacjach IoT oraz M2M (Machine to Machine).



Cechy protokołu:

1. Szybkość transmisji do 1000 mbps
2. Ilość urządzeń podłączonych do jednej sieci zależy przede wszystkich od brokera i może ich być ponad 1000
3. niskie zapotrzebowanie względem zasobów sprzętowych
4. wymiana informacji, odbywająca się w niemal czasie rzeczywistym
5. niskie obciążenie łącza komunikacyjnego
6. bezpieczeństwo komunikacji w MQTT realizowane jest na poziomie szyfrowania protokołu w oparciu o SSL

2. Zasada działania



Protokół opiera się na wzorcu publikacja/subskrypcja czyli, urządzenie typu broker stanowi pośrednika w wymianie informacji między klientami. Każdy klient może publikować informacje o danym temacie oraz payloadzie ale tą informację otrzymają tylko ci klienci którzy wcześniej zasubskrybowali dany temat. Jeśli nie klienci nie mają zasubskrybowanego tematu to broker nie wysyła do nich tej informacji przez co nie „zaśmieca” medium transmisyjnego niepotrzebnymi wiadomościami.

Dla publikacji dużą rolę odgrywa parametr QoS (Quality of Service). W uproszczeniu określa on, jak bardzo dbamy o dostarczenie paczki.

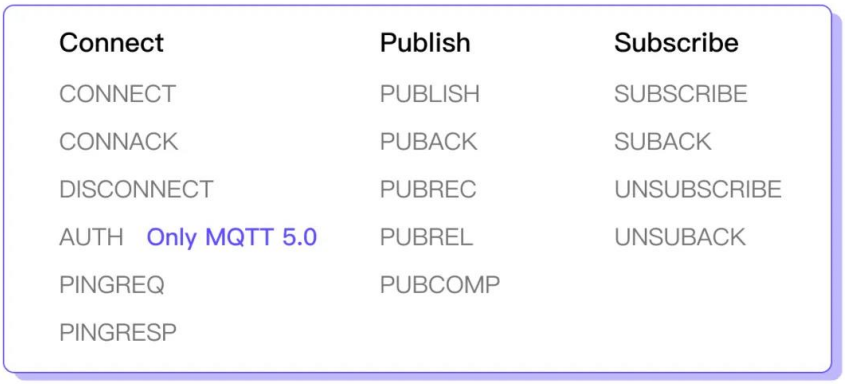
– QoS = 0 – publikujemy i nie obchodzi nas, co dalej dzieje się z paczką.

– Qos = 1 – po nadaniu klient publikujący przechowuje wiadomość aż do ACK, dając sobie tym samym możliwość do powtórnego wysłania w razie braku potwierdzenia.

– QoS = 2 – mechanizm oparty o dwupoziomowy handshake, kiedy chcemy być pewni dostarczenia naszej paczki oraz uniknięcia wysłania jej więcej niż raz.

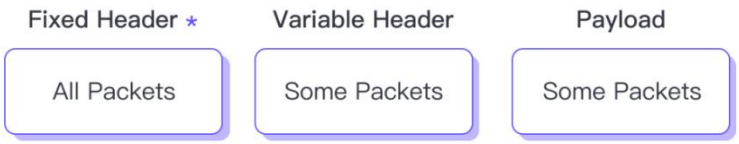
Struktura pakietu MQTT

MQTT definiuje 15 typów pakietów kontrolnych.



- CONNECT – Używany przez klienta do rozpoczęcia połączenia z brokerem jeśli połączenie przebiegnie prawidłowo wysyłane je CONNACK
- DISCONNECT – klient oraz broker mogą wysłać to jeśli zakończą połączenie
- PINREQ oraz PINGRRESP – pakiety które cyklicznie muszą być przesyłane aby broker wiedział czy dany klient jeszcze żyje
- PUBLISH – pakiet używany do przesyłania wiadomości
- SUBSCRIBE – pakiet wysyła klient jeśli chce dany temat zasubskrybować. W odpowiedzi od brokera otrzymuje SUBACK
- SUBSCRIBE – pakiet wysyła klient jeśli chce dany przestać subskrybować. W odpowiedzi od brokera otrzymuje UNSUBACK

W zależności od rodzaju pakietu struktura pakietu przedstawia się następująco:

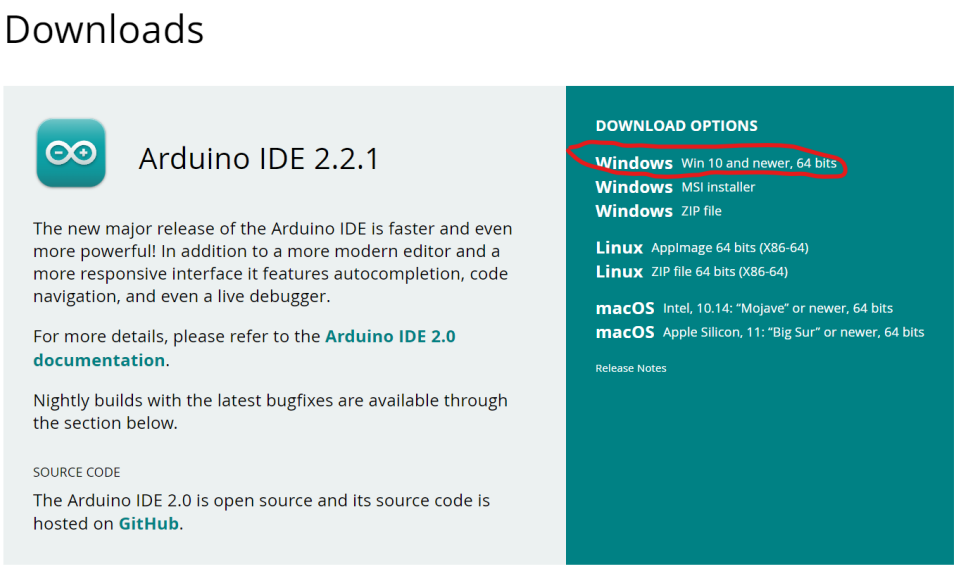


Fixed Header znajduje się w każdym pakiecie natomiast Variable Header oraz Payload występują w zależności od typu pakietu.

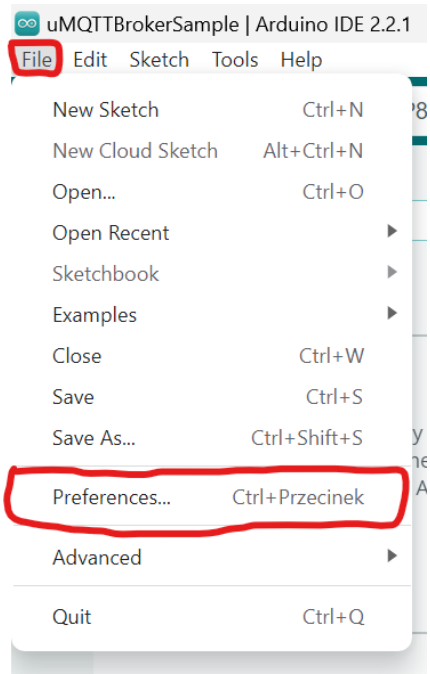
3. Przygotowanie środowiska

Na początku należy pobrać i zainstalować Arduino IDE :

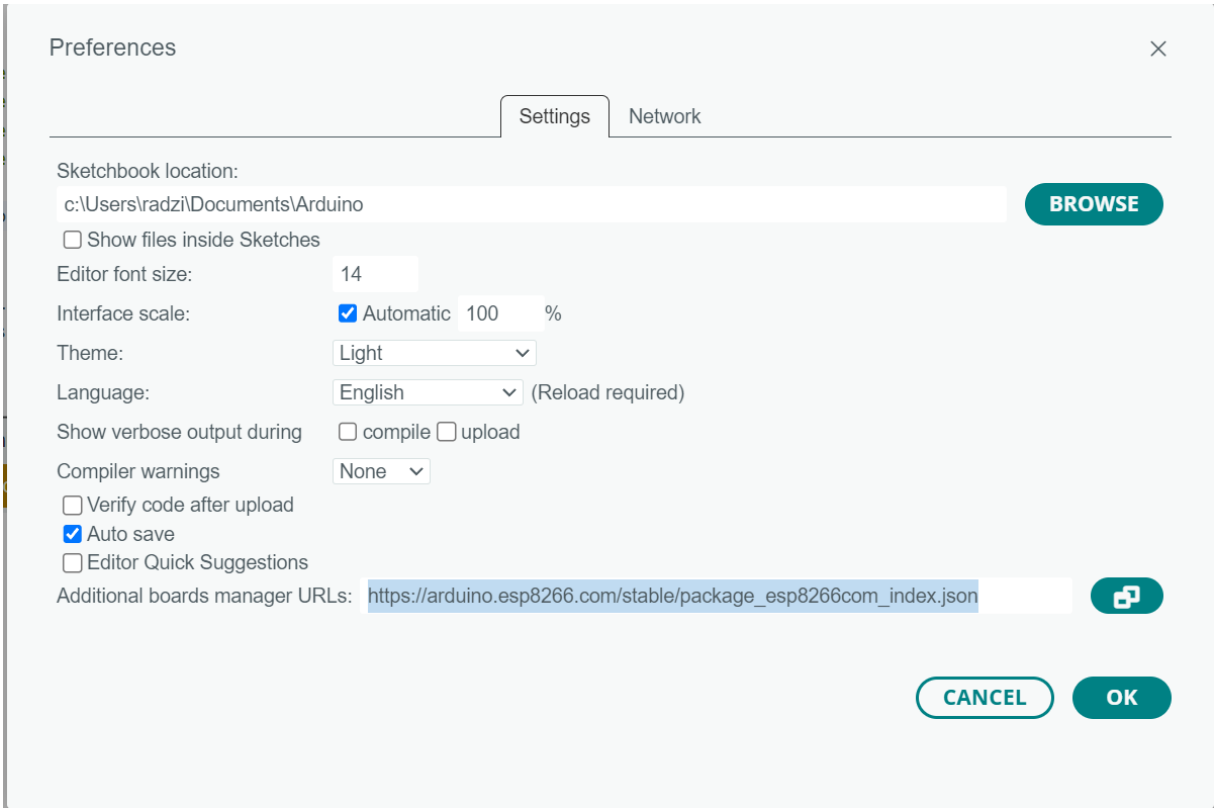
https://www.arduino.cc/en/software



Następnie należy zainstalować wsparcie dla płytek esp8266. W tym celu należy otworzyć Arduino IDE i wejść w zakładkę *File->Preferences..*



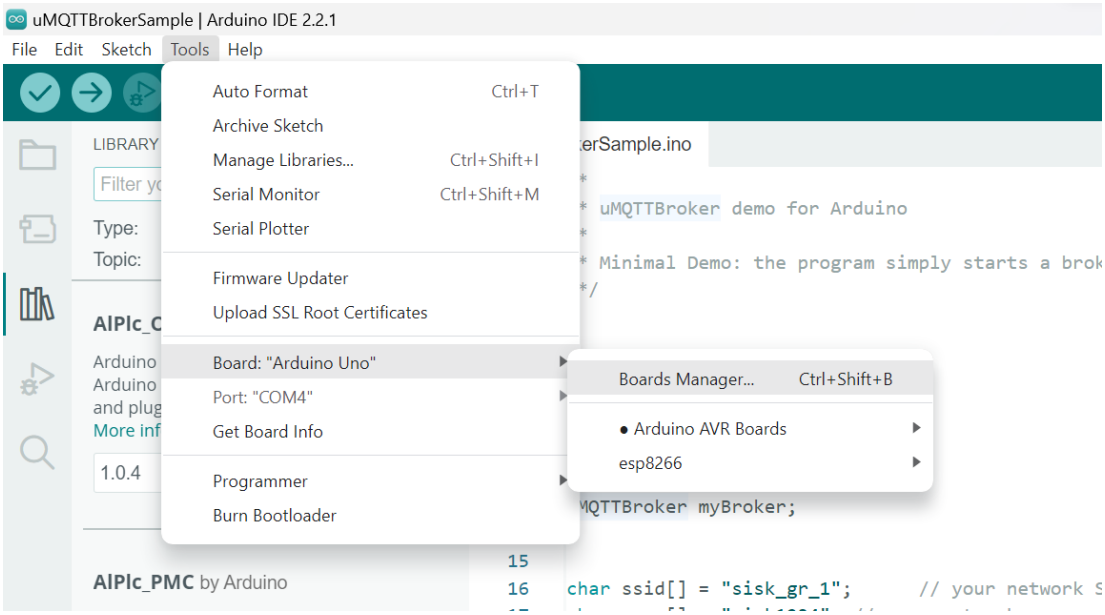
a następnie w zakładce *Settings* należy podać w polu *Additional Boards Manager URLs*: następujący link:
https://arduino.esp8266.com/stable/package_esp8266com_index.json



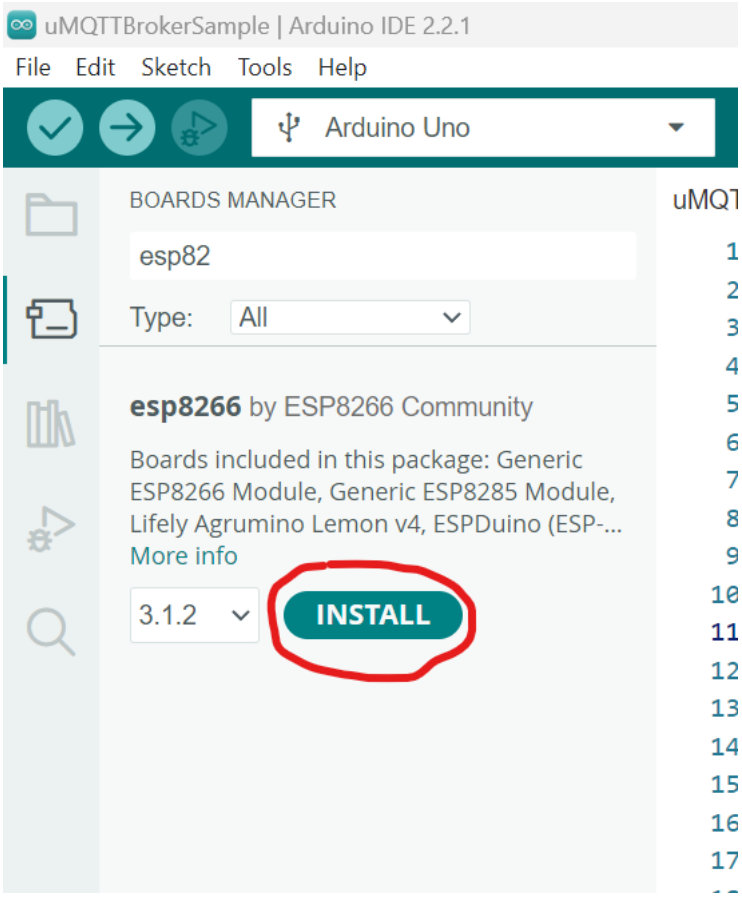
Jeśli powyższy link już był wpisany to nic nie trzeba robić. W przypadku gdy znajduje się jakiś inny link to wklejamy go oddzielając link przecinkiem. Następnie wciskamy przycisk OK.

Teraz trzeba zainstalować pakiet do esp8266. W tym celu należy wejść w zakładkę:

Tools->Board->Board Manager…



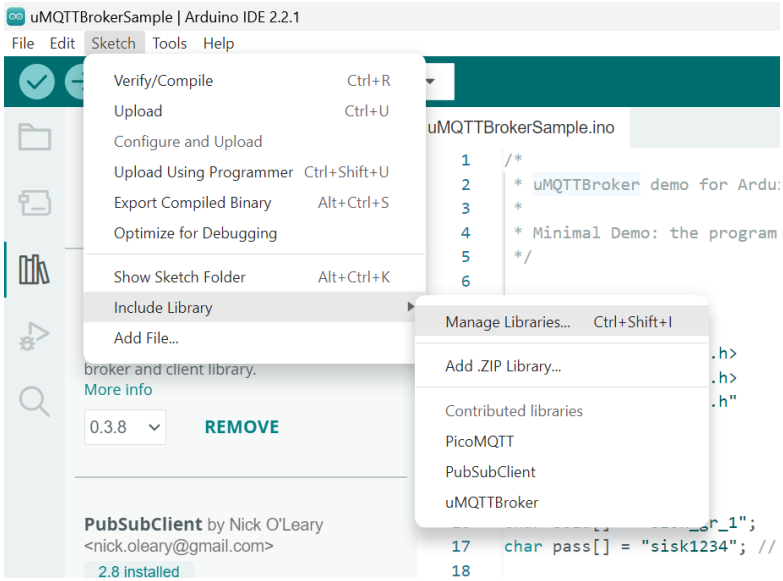
Następnie w boards manager wyszukujemy esp82 i instalujemy rozszerzenie **esp8266 by ESP8266 Community**



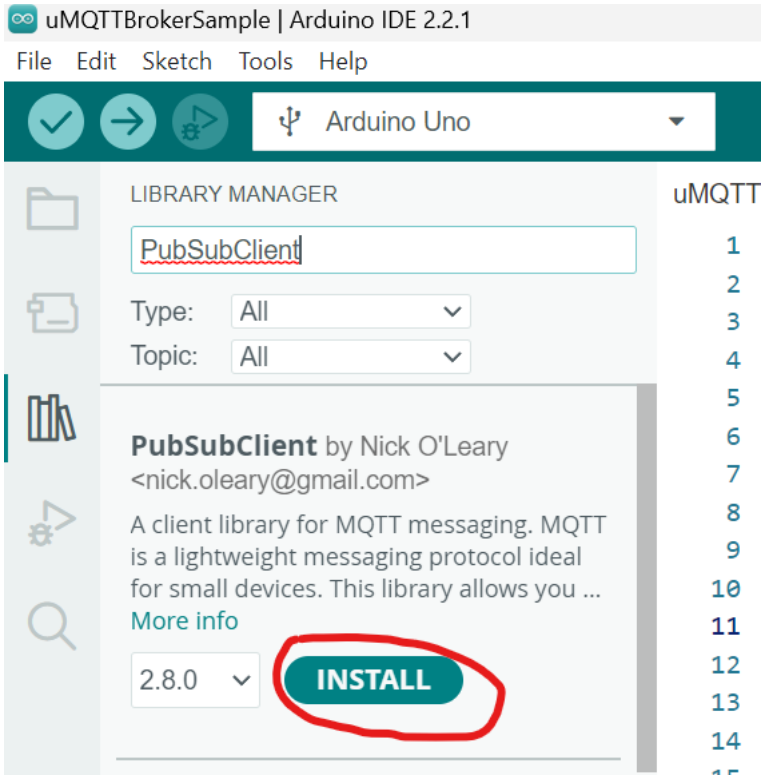
Teraz musimy dołączyć biblioteki które będziemy używać:

1) PubSubClient – bliblioteka wykorzystywana przez clienta to komunikacji z brokerem

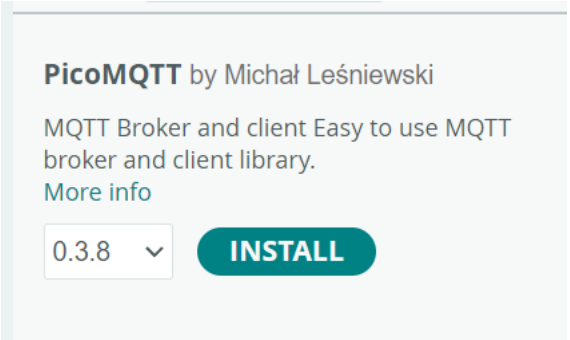
Przechodzimy do *Sketch -> Include Library -> Manage Libraries…*



A następnie w okienku wyszukiwania bibliotek wpisujemy *PubSubClient* i instalujemy rozszerzenie które jest przedstawione na rysunku poniżej.



2) PicoMQTT – biblioteka do obsługi brokera MQTT. Tak samo instalujemy jak poprzednią bibliotekę.

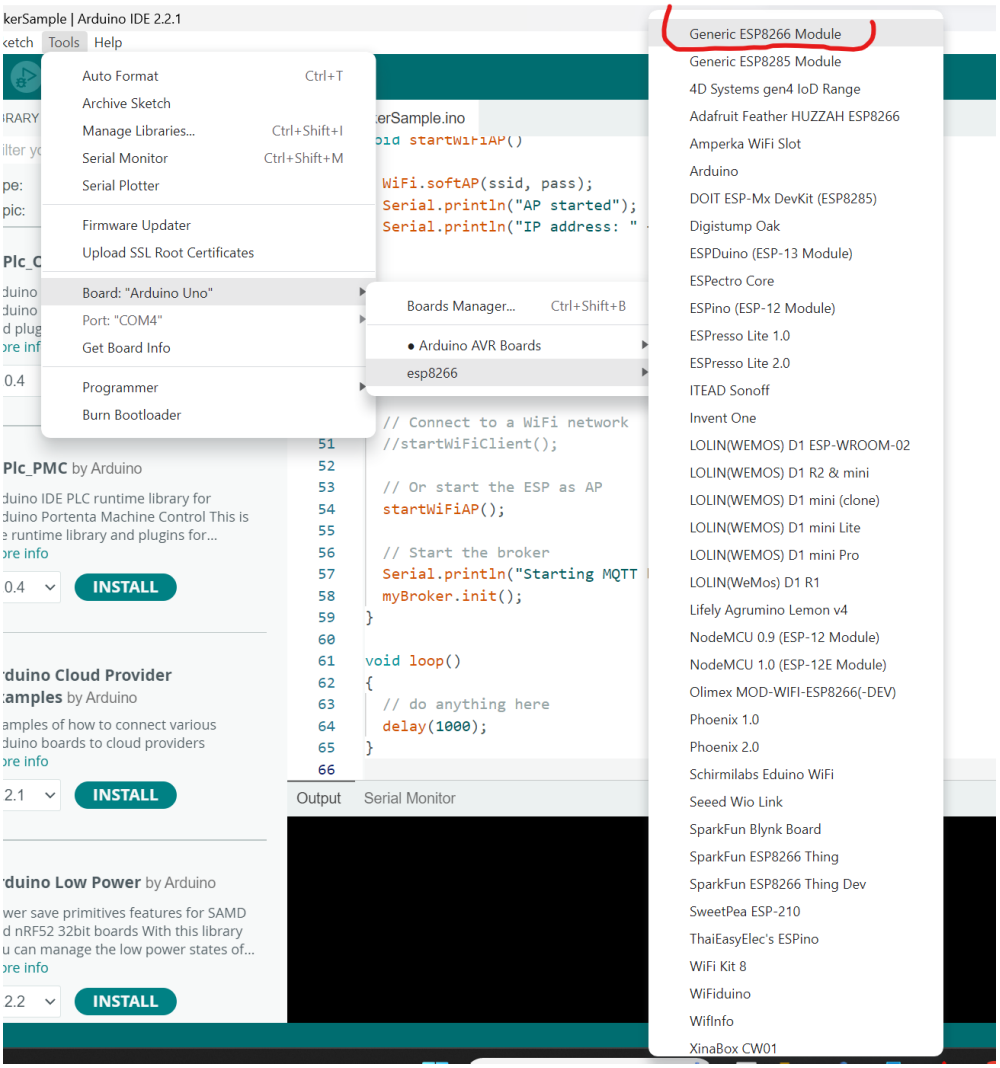


4. Uruchomienie brokera

Należy wkleić podany kod do Arduino IDE:

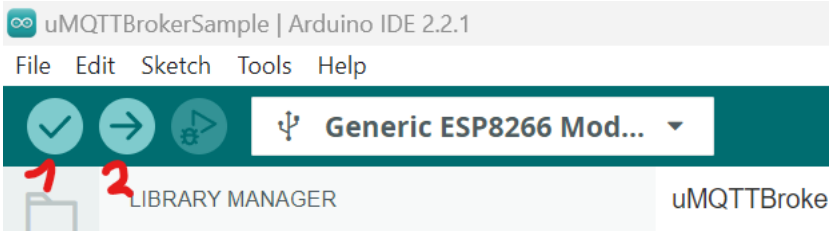
```
#include <Arduino.h>
#include <PicoMQTT.h>
#include <ESP8266WiFi.h>

PicoMQTT::Server mqtt;

char ssid[] = "sisk_gr_1";      // your network SSID (name)
char pass[] = "sisk1234"; // your network password

void setup() {
    // Usual setup
    Serial.begin(115200);
    WiFi.softAP(ssid, pass);
    Serial.println("AP started");
    Serial.println("IP address: " + WiFi.softAPIP().toString());

    // Subscribe to a topic pattern and attach a callback
    mqtt.subscribe("#", [](const char * topic, const char * payload) {
        Serial.printf("Received message in topic '%s': %s\n", topic,
payload);
    });

    // Start the broker
    mqtt.begin();
}

void loop() {
    // This will automatically handle client connections.  By default,
all clients are accepted.
    mqtt.loop();

    if (random(1000) == 0)
        mqtt.publish("picomqtt/welcome", "Hello from PicoMQTT!");
}
```

A następnie zmienić ssid i pass na wybrane przez siebie nazwy.



Następnie ustawiamy odpowiednią płytkę w zakładce *Tools->Board: -> esp8266 -> Generic ESP8266 Module* jak również wybieramy port szeregowy za pomocą którego jesteśmy podłączeni z płytką w zakładce Tools->Port

Następnie klikamy na Verify (1) i po udanej weryfikacji klikamy upload (2)



Po prawidłowym wgraniu naszego programu przechodzimy do zakładki Serial Monitor i wybieramy baud 115200. Teraz należy zresetować płytkę za pomocą przycisku reset i na ekranie powinien pojawić się następujący komunikat o prawidłowym uruchomieniu brokera. Ponadto należy zapisać sobie ip address ponieważ będzie on potrzebny w dalszych krokach.

5. Uruchomienie klienta

Należy otworzyć nowy sketch *File-> New Sketch* i wkleić poniższy kod:

```
/*
 Basic ESP8266 MQTT client

*/

#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

// Update these with values suitable for your network.

const char* ssid = "sisk_gr_1";
const char* password = "sisk1234";
const char* mqtt_server = "192.168.4.1";

WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE	(50)
char msg[MSG_BUFFER_SIZE];
int value = 0;
float sin_time = 0; // Zmienna reprezentująca czas
float temperature = 25.0;

void setup_wifi() {

 delay(10);
 // We start by connecting to a WiFi network
 Serial.println();
 Serial.print("Connecting 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("");
 Serial.println("WiFi connected");
 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());
```

```
}

void callback(char* topic, byte* payload, unsigned int length) {
 Serial.print("Message arrived [");
 Serial.print(topic);
 Serial.print("] ");
 for (int i = 0; i < length; i++) {
  Serial.print((char)payload[i]);
 }
 Serial.println();
 if(strcmp(topic,"LED0") == 0){
 // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
   digitalWrite(BUILTIN_LED, LOW);   // Turn the LED on (Note that LOW is the voltage level
   // but actually the LED is on; this is because
   // it is active low on the ESP-01)
  } else {
   digitalWrite(BUILTIN_LED, HIGH);  // Turn the LED off by making the voltage HIGH
  }

 }

}

void reconnect() {
 // Loop until we're reconnected
 while (!client.connected()) {
  Serial.print("Attempting MQTT connection...");
  // Create a random client ID
  String clientId = "ESP8266Client-";
  clientId += String(random(0xffff), HEX);
  // Attempt to connect
  if (client.connect(clientId.c_str())) {
   Serial.println("connected");
   // Once connected, publish an announcement...
   client.publish("outTopic", "hello world");
   // ... and resubscribe
   client.subscribe("LED0");
  } else {
   Serial.print("failed, rc=");
   Serial.print(client.state());
   Serial.println(" try again in 5 seconds");
   // Wait 5 seconds before retrying
   delay(5000);
  }
 }
}

void setup() {
 pinMode(BUILTIN_LED, OUTPUT);     // Initialize the BUILTIN_LED pin as an output
 Serial.begin(115200);
 setup_wifi();
 client.setServer(mqtt_server, 1883);
 client.setCallback(callback);
}

void loop() {

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

 unsigned long now = millis();
 if (now - lastMsg > 2000) {
  lastMsg = now;
  temperature = random(200, 280) / 10.0; // Losowa temperatura w zakresie od 20 do 30 stopni Celsiusza
  sin_time += 0.1;
  snprintf (msg, MSG_BUFFER_SIZE, "%.2f\n", temperature);
  client.publish("temp_0",msg);

 }
}
```

Należy ssid oraz password na taki sam jak w przypadku brokera oraz wpisać adres ip który przedtem zanotowaliśmy. Należy tak samo jak w

przypadku brokera ustawić odpowiednią płytkę oraz port, zweryfikować i wgrać program.
Po prawidłowym wgraniu programu, otworzyć Serial Monitor, ustawić baud 115200 i zresetować płytkę za pomocą przycisku reset na płytce. Po resecie powinna pojawić się informacja :

Jeżeli nie może się połączyć z brokerem to najpierw trzeba sprawdzić czy wpisany ssid oraz hasło są takie same w brokerze i w kliencie oraz czy adres ip mqtt_server jest prawidłowy. Jeśli to nie pomoże to można jeszcze spróbować zresetować najpierw brokera a potem kilenta.

5. Sterowanie za pomocą telefonu

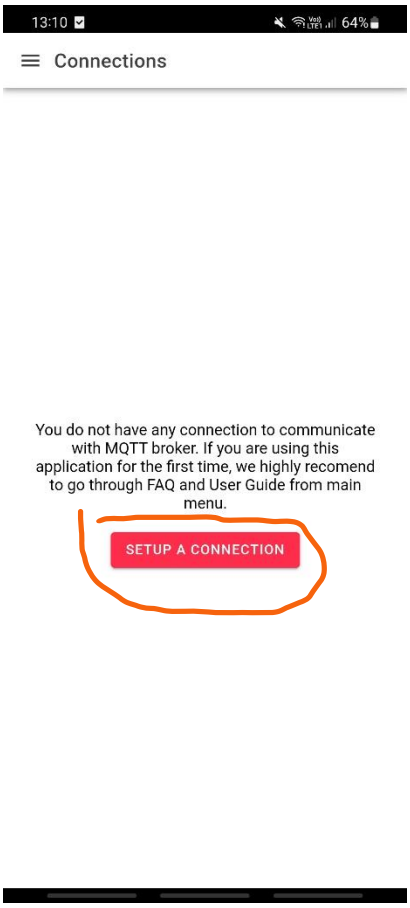
Należy pobrać na telefon aplikację: IoT MQTT Panel:

https://play.google.com/store/apps/details?id=snr.lab.iotmqttpanel.prod&hl=en_US

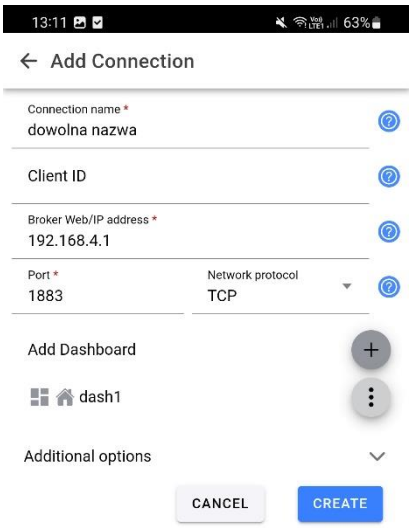
lub ios:

https://apps.apple.com/us/app/iot-mqtt-panel/id6466780124

Następnie połączyć się z siecią wifi brokera, wejść do aplikacji ustawić nowe połoczenie:



W polu connection name wpisujemy dowolną nazwę, następnie w Broker Web/IP address podajemy adres ip brokera, port 1883, oraz protokół TCP. Klikamy Add dashboard i ustawiamy dowolną nazwę naszego panelu. Następnie kilkamy na CREATE.

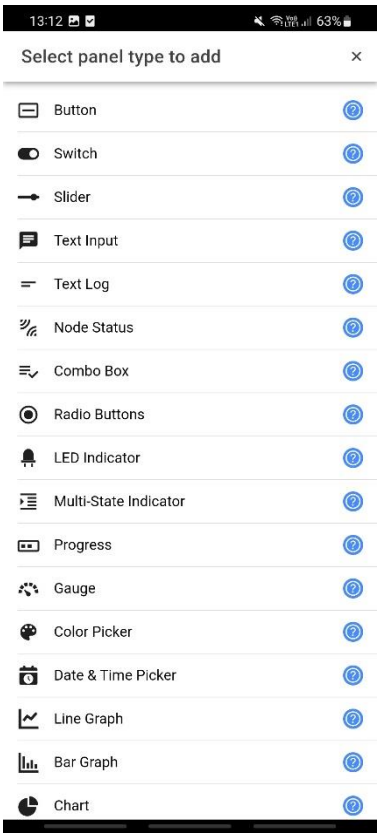


Powinien pojawić się taki panel:

Jeśli nie pojawi się chmurka na pomarańczowym tle należy spróbować kliknąć w tym miejscu żeby się połączyć. Jeśli nadal się nie uda to należy sprawdzić czy nadal jesteśmy połączeni z siecią wifi brokera. Jeśli dalej nie działa to należy zresetować brokera.

Następnie musimy dodać jakieś panele aby móc sterować diodą led oraz odczytywać symulowaną przez klienta temperaturę.

Klikamy add panel, wybieramy typ Switch



A następnie wypełniamy:

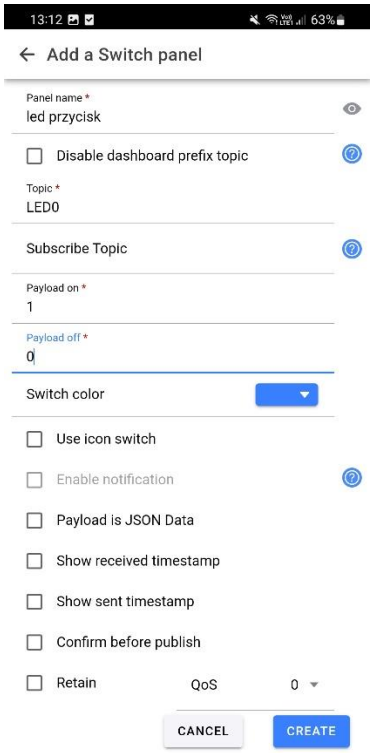
Panel name – dowolna nazwa

Topic - LED0

Payload on - 1

Payload off – 0

Wysyłając wiadomość o temacie LED0 i wiadomości 1 załączymy diodę led na esp który jest klientem, ponieważ ma zasubskrybowany ten temat i jeśli otrzyma wiadomość 1 to załączy diodę a jak 0 to ją wyłączy.



-

Klikamy przycisk CREATE.

Możemy teraz spróbować sterować diodą led za pomocą przycisku.

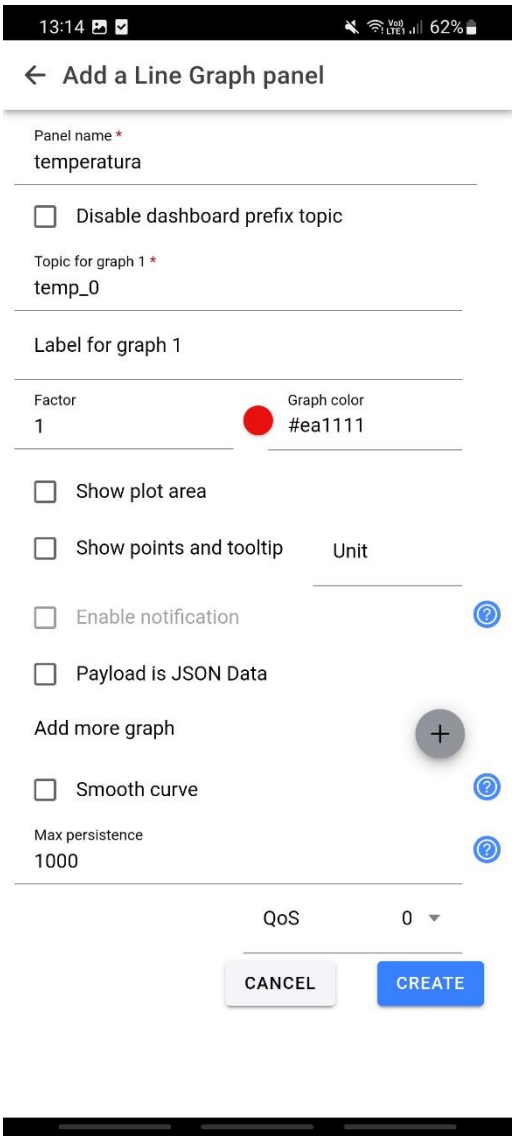
Teraz spróbujemy zobrazować wyniki pomiaru temperatury który przesyłany jest za pomocą tematu temp_0. Klikamy przycisk + na dole po prawej stronie, klikamy Line Graph i wypełniamy następująco:

- Panel name: dowolna nazwa

-Topic for graph 1: temp_0

-Max persistence – 1000 (żeby wyświetlał dłuższą historię danych)

Klikamy CREATE i powinniśmy zaobserwować tworzący się wykres


13:14
62%
Add a Line Graph panel
Panel name *
temperatura
Disable dashboard prefix topic
Topic for graph 1 *
temp_0
Label for graph 1
Factor
1
Graph color
#ea1111
Show plot area
Show points and tooltip
Unit
Enable notification
Payload is JSON Data
Add more graph
Smooth curve
Max persistence
1000
QoS
0
CANCEL
CREATE

Jeśli coś nie działa to: sprawdzić czy u góry po prawej stronie jest chmurka pomarańczowa, jeśli nie to kliknąć tam aby się ponownie połączyć, jeśli dalej nie działa to zresetować najpierw esp brokera a potem esp klienta.