Actividad 3 IoT

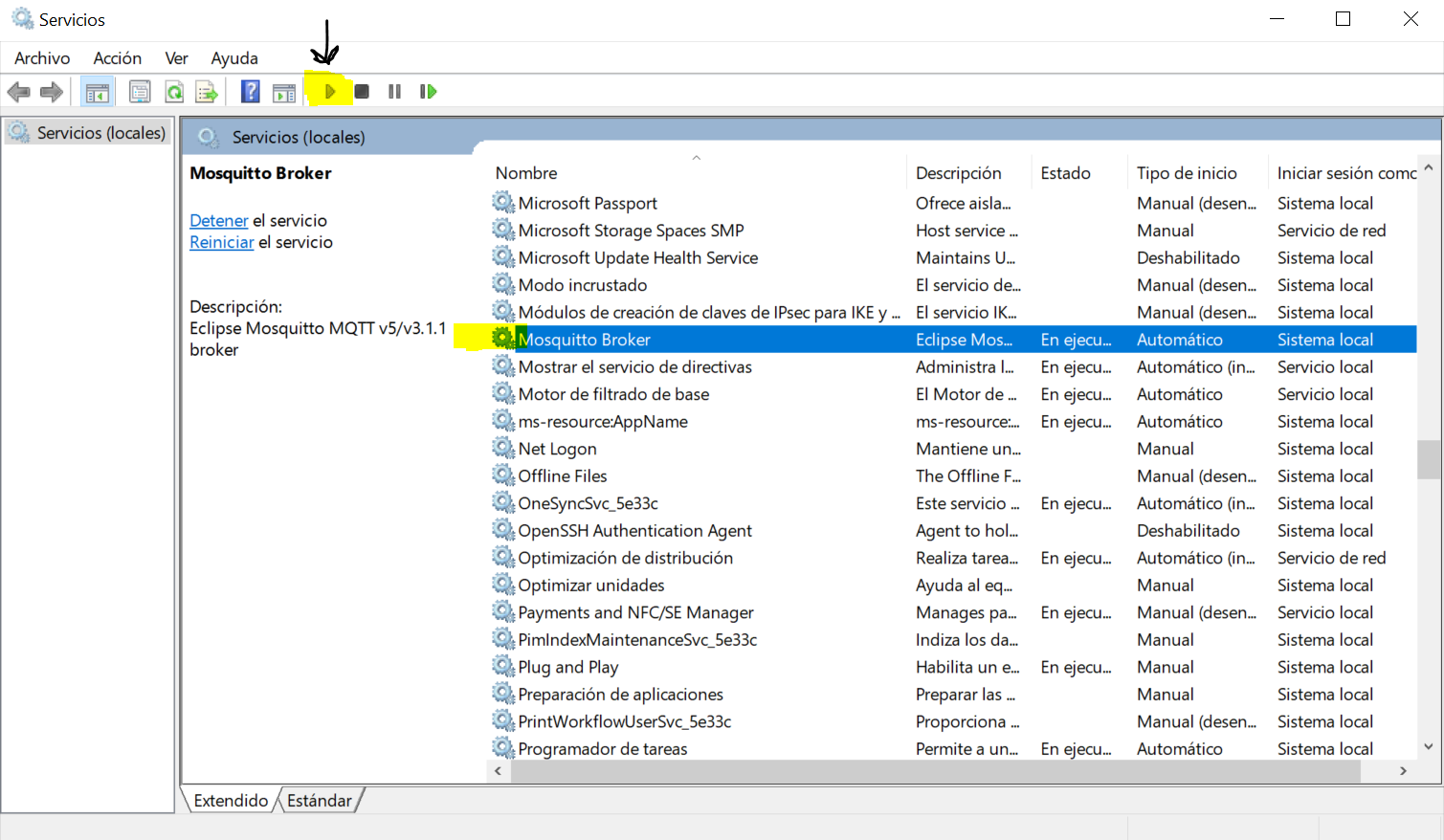
Diego Iván Perea Montealegre (2238513) [diego.perea@uao.edu.co](mailto:diego.perea@uao.edu.co)

Facultad de Ingeniería, Universidad Autónoma de Occidente

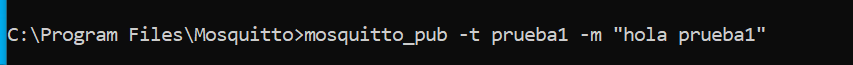
Cali, Valle del Cauca

Instalación y ejecución de Mosquito : <https://mosquitto.org/files/binary/>

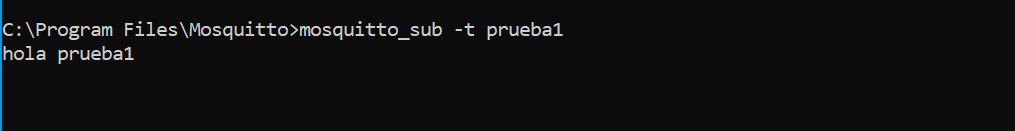
Es aconsejable tener la última versión



En la cmd se va a la ruta de mosquito (donde se instaló mosquito) y se realiza la prueba de creación de la suscripción y de la publicación

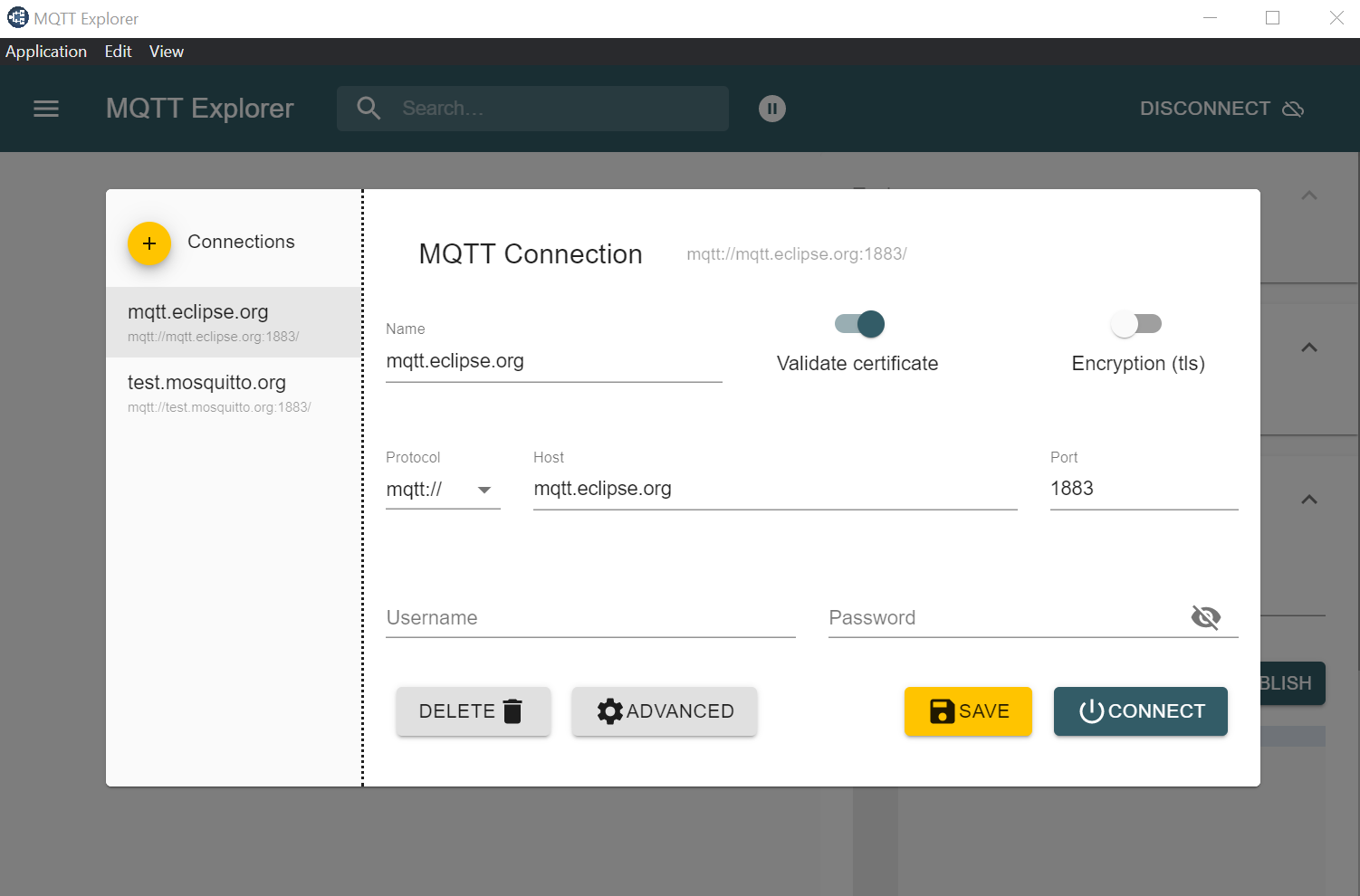


Comando de publicación a prueba1

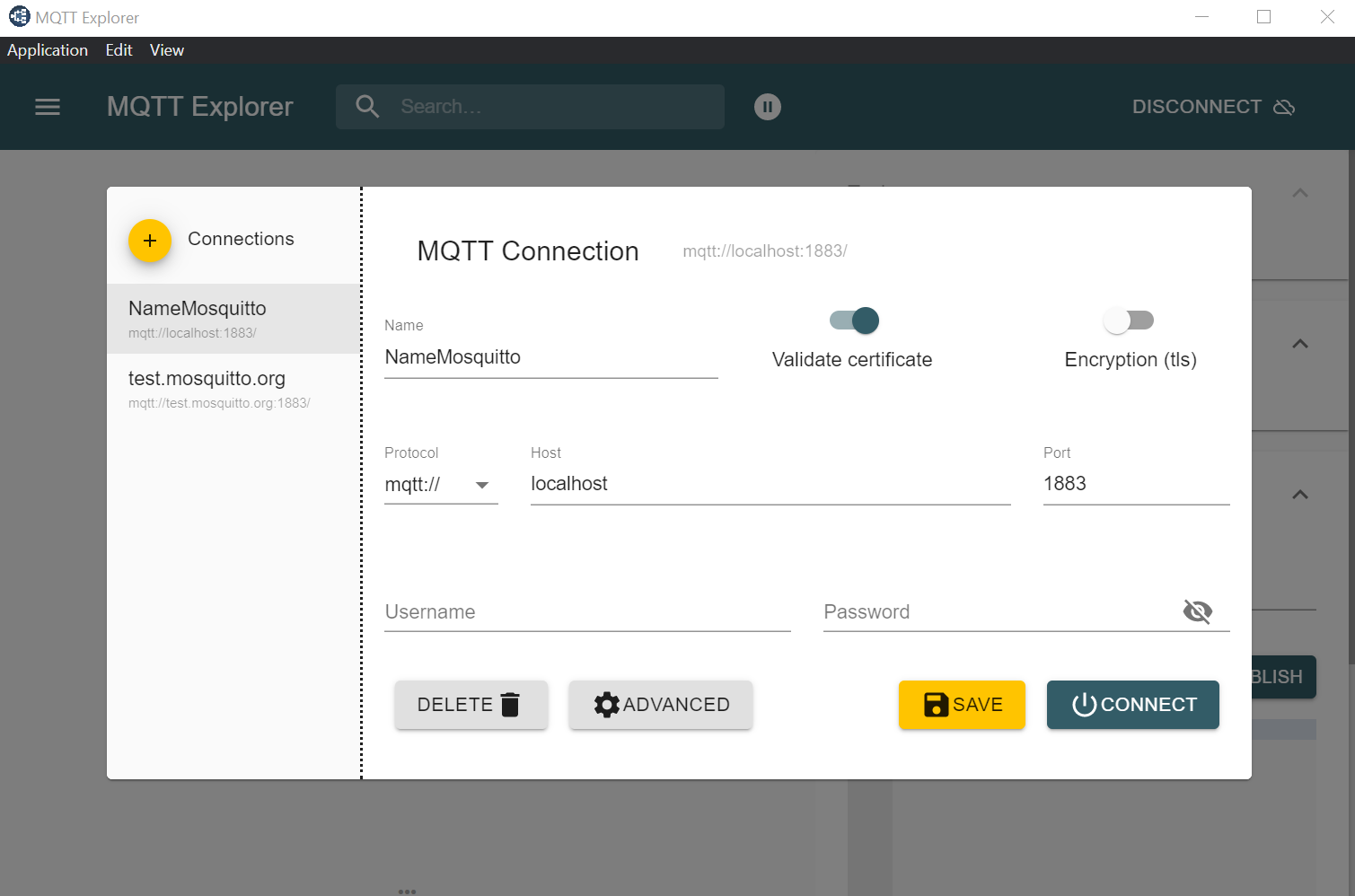


Comando de suscripción llamado prueba 1

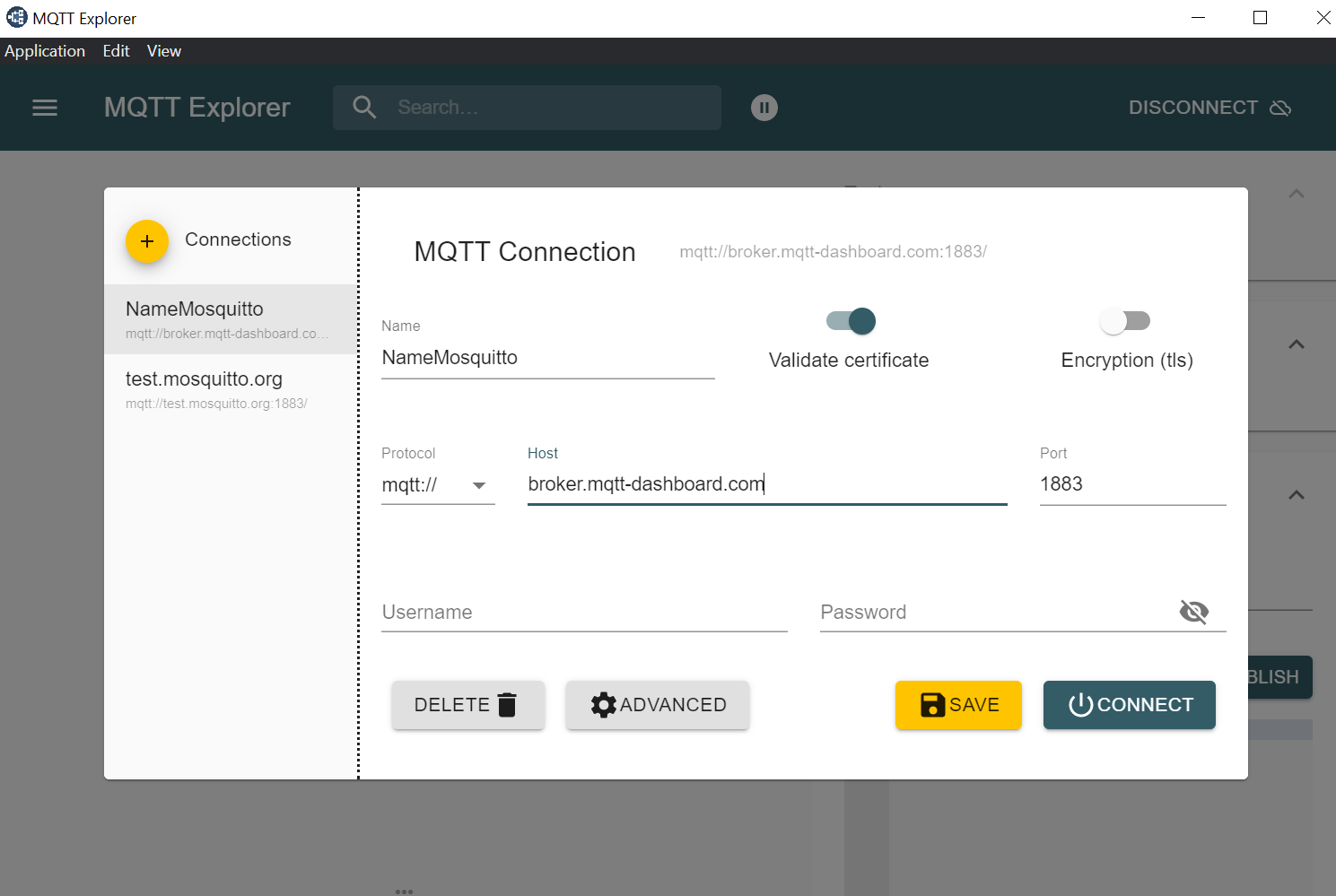
Instalación de mqtt explorer : <https://mqtt-explorer.com/>



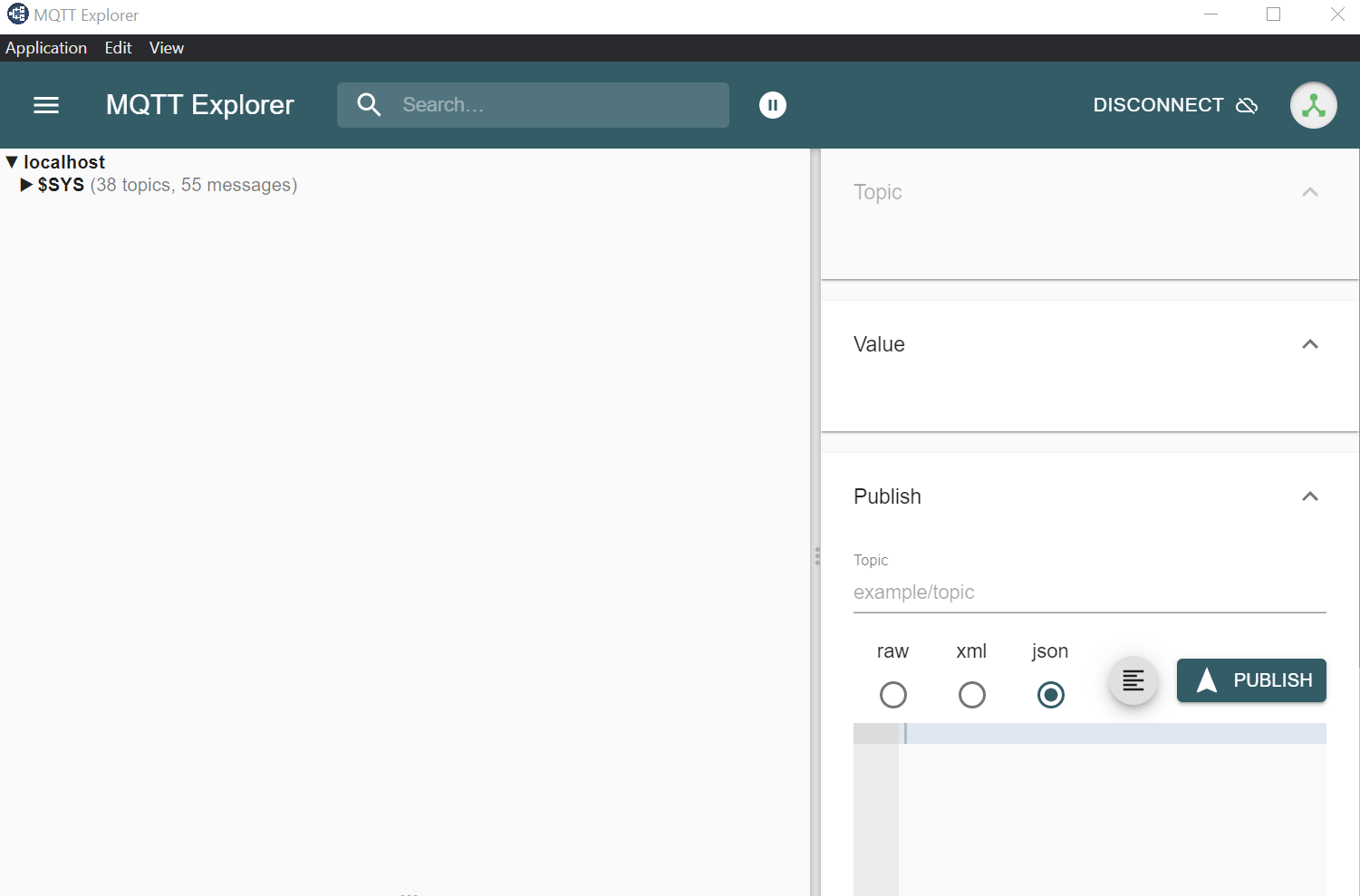
Para conectarse con el bróker mosquitto de forma local seria :



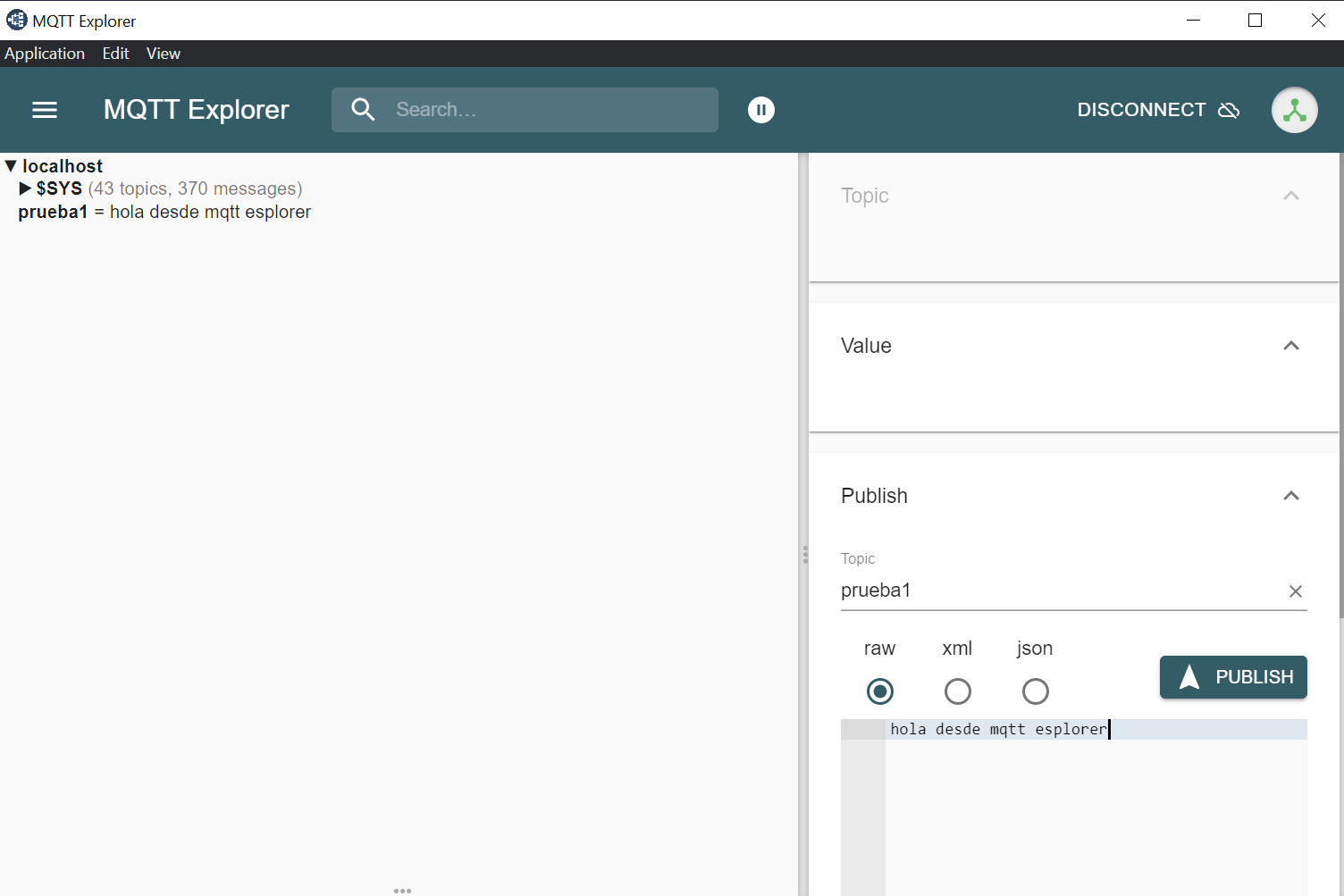
Pero si no se tiene mosquitto instalado de forma local , se puede usar uno en la nube que seria el HOST: broker.mqtt-dashboard.com



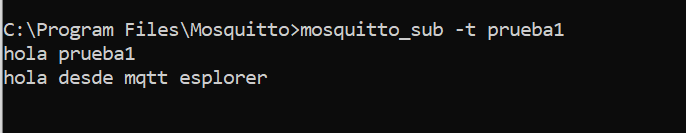
Y no se pone nada en username y password y darle en connect :



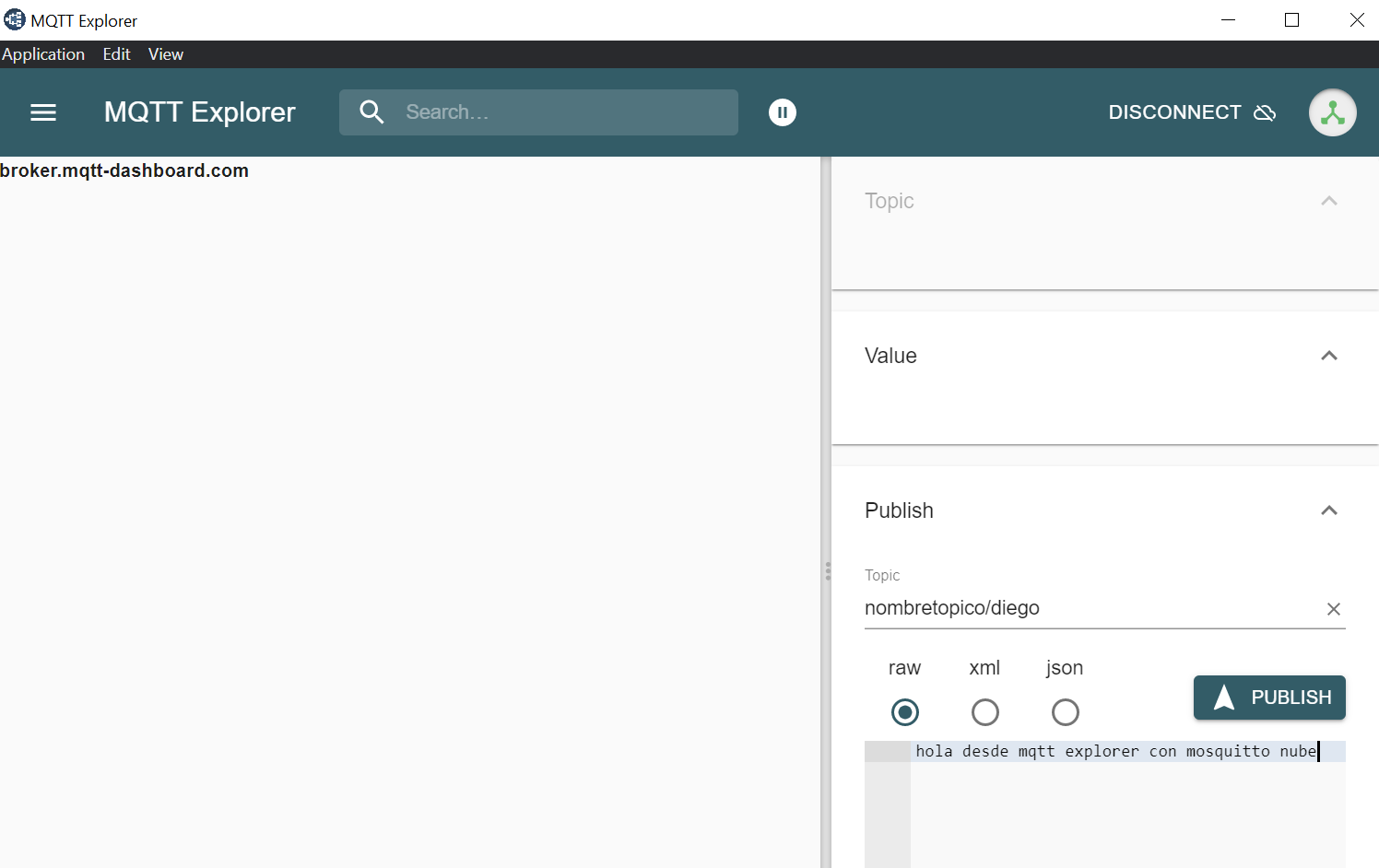
Para publicarlo seria de esta forma :

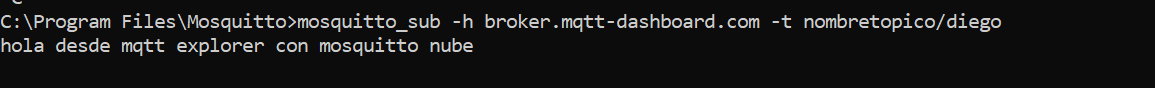


Y se visualiza que se tomo correctamente y se evidencia en el cmd :



Ahora probando con mosquitto en nube :



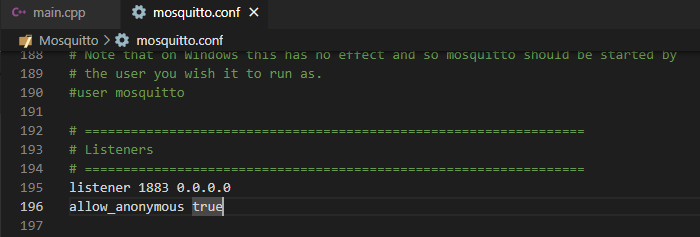


Para ESP32:

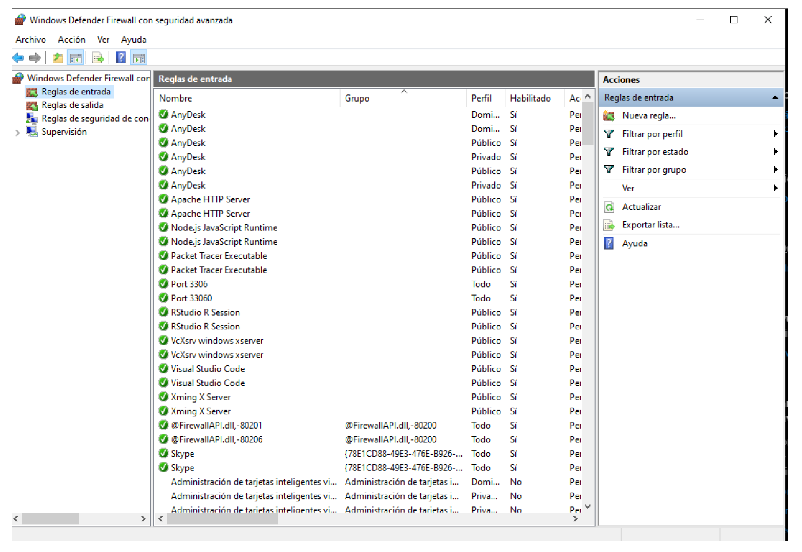
Si la conexión de mosquitto no se da con el esp32 hacer:

Desde la carpeta mosquito , modificar archivo mosquitto.conf y en parte Listeners agregar : listener 1883 0.0.0.0

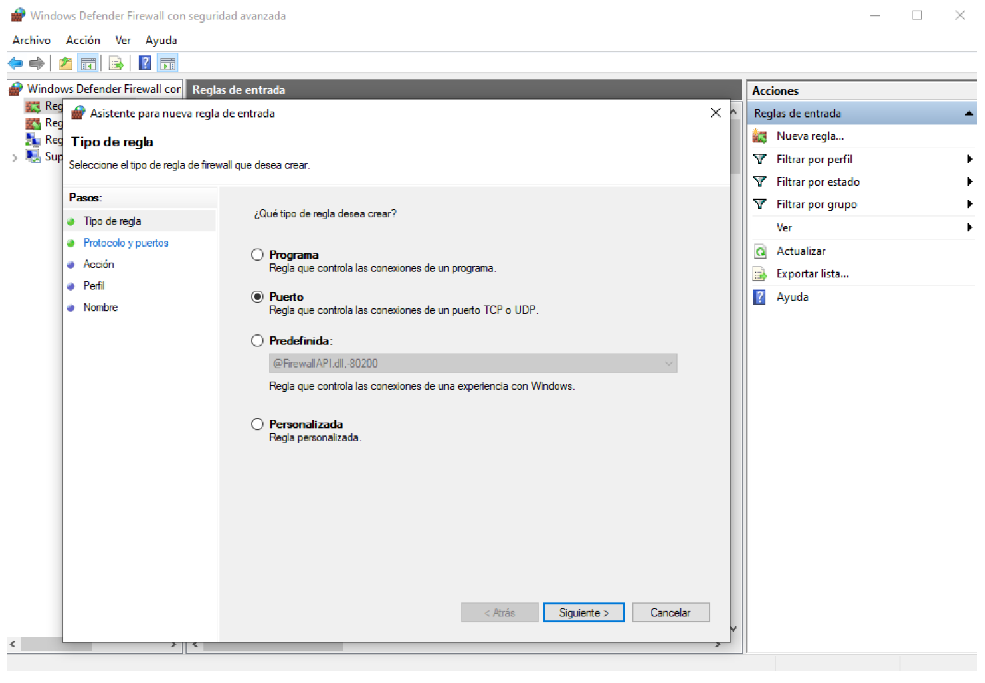
allow\_anonymous true



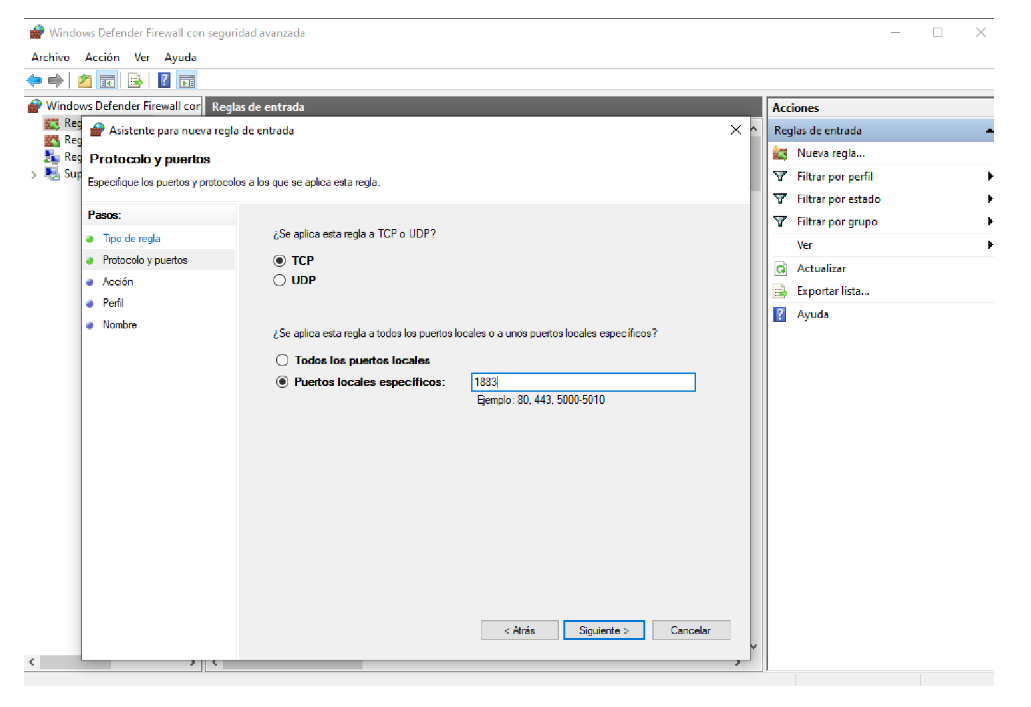
Ir a “windows defender firewall con seguridad avanzada”



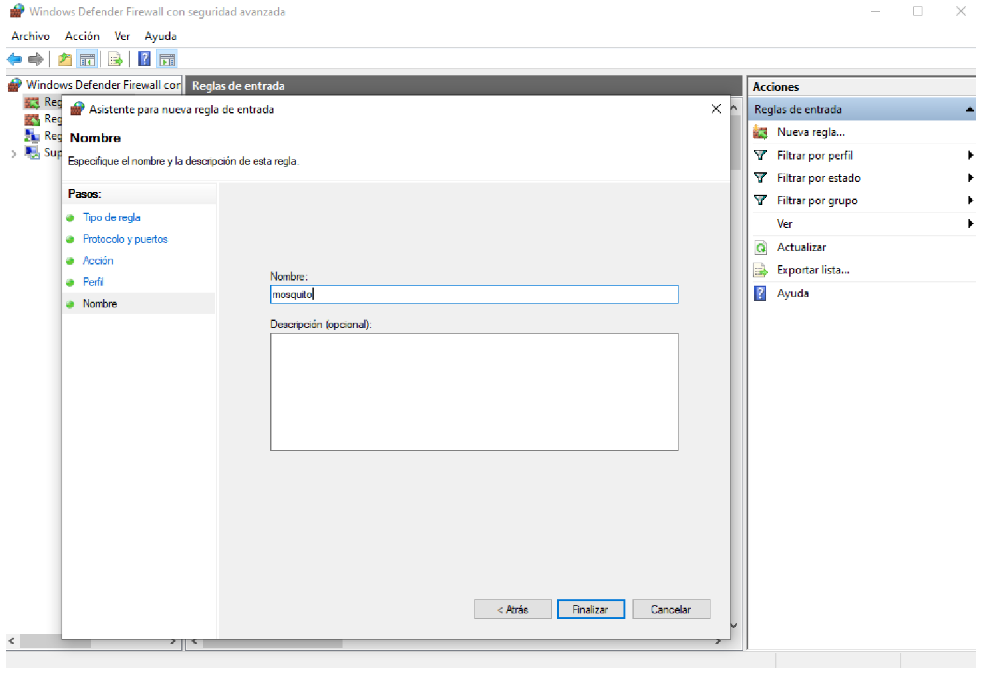
Crear nueva regla , seleccionar puerto



Seleccionar TCP e introducir el pueto “1883” en puertos locales

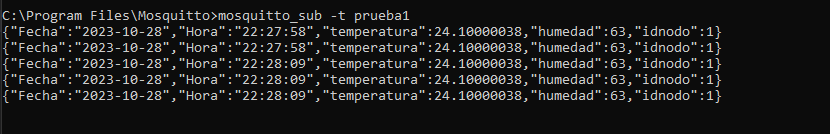


Dar un nombre , y darle en finalizar :

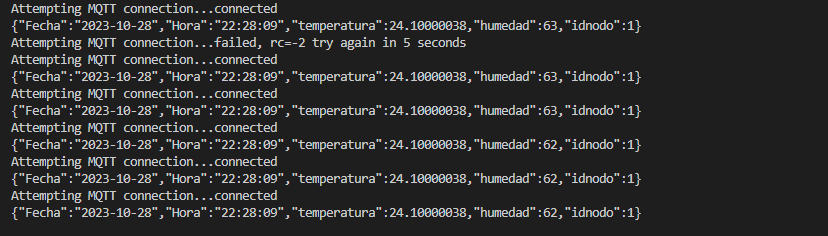


Reiniciar equipo y ejecutar código de ESP32

Cmd con suscripción de mosquito de topic “prueba1”



Envio de datos desde terminal Vscode:



Usando Platformio Codigo de ESP32:

#include <Arduino.h>

#include <ArduinoJson.h>

#include <WiFi.h>

#include <PubSubClient.h>

//LIBRERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)

#include <Adafruit\_Sensor.h>

#include <DHT.h>

//LIBRERIAS PARA FECHA Y HORA

#include <WiFi.h>

#include <NTPClient.h>

#include <WiFiUdp.h>

//DEFINICION DE PINES DHT11

#define DHTPIN 14   // 4 = PIN D4

#define DHTTYPE    DHT11

DHT dht(DHTPIN, DHTTYPE);

// Define NTP Client to get time

WiFiUDP ntpUDP;

NTPClient timeClient(ntpUDP);

// Variables to save date and time

String formattedDate;

String dayStamp;

String timeStamp;

#define mqttUser ""

#define mqttPass ""

#define mqttPort 1883

const char\* ssid = "\*\*NAME\_WIFI\*";//name wifi

const char\* password = "\*PASSWORD\_WIFI\*"; // clave de wifi

char mqttBroker[] = "192.168.\*\*.\*"; //ip del servidor

char mqttClientId[] = "prueba1"; //cualquier nombre

char inTopic[] = "prueba1";//topcico a suscribirse

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();

}

WiFiClient BClient;

PubSubClient client(BClient);

void reconnect() {

// Loop until we're reconnected

  while (!client.connected()) {

  Serial.print("Attempting MQTT connection...");

  // Attempt to connect

  if (client.connect("", mqttUser, mqttPass)) {

  Serial.println("connected");

  // Once connected, publish an announcement...

 // Once connected, publish an announcement...

  float h= dht.readHumidity();

  float t =dht.readTemperature();

  String variable;

  StaticJsonDocument<256> doc;

  doc["Fecha"] = dayStamp;

  doc["Hora"] = timeStamp;

  doc["temperatura"] = t;

  doc["humedad"] = h;

  doc["idnodo"] = 1;

  serializeJson(doc, variable);

  int lon = variable.length()+1;

  Serial.println(variable);

  char datojson[lon];

  variable.toCharArray(datojson, lon);

  client.publish(inTopic,datojson);

  client.disconnect();

  delay(5000);

  // ... and resubscribe

  //client.subscribe("topic2");

  } 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\_wifi() {

  delay(10);

  // We start by connecting to a WiFi network

  Serial.println();

  Serial.print("Connecting to ");

  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL\_CONNECTED) {

  delay(500);

  Serial.print(".");

  }

  Serial.println("");

  Serial.println("WiFi connected");

  Serial.println("IP address: ");

  Serial.println(WiFi.localIP());

  // Initialize a NTPClient to get time

  timeClient.begin();

  // Set offset time in seconds to adjust for your timezone, for example:

  // COLOMBIA -5 , entonces -5\*3600 ->  -18000

  timeClient.setTimeOffset(-18000); //Thailand +7 = 25200

}

void setup()

{

  Serial.begin(9600); //Serial connection

  setup\_wifi(); //WiFi connection

  client.setServer(mqttBroker, mqttPort );

  client.setCallback( callback );

  Serial.println("Setup done");

  delay(1500);

}

void loop(){

    while(!timeClient.update()) {

    timeClient.forceUpdate();

    }

    // The formattedDate comes with the following format:

    // 2018-05-28T16:00:13Z

    // We need to extract date and time

    formattedDate = timeClient.getFormattedDate();

    // Extract date

    int splitT = formattedDate.indexOf("T");

    dayStamp = formattedDate.substring(0, splitT);

    //Serial.print("DATE: ");

    //Serial.println(dayStamp);

    // Extract time

    timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);

    if (!client.connected()) {

    reconnect();

    }

    client.loop();

}