**UNIVERSIDAD TECNOLÓGICA DE PANAMA**

**CENTRO REGIONAL DE CHIRIQUÍ**

**FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES**

Semestral Ingeniería de Software II 0753

Fecha**. 07/29/2019**

Nombre: Danniel Justavino / Erasmo Navarro Cédula: 4-792-2438/4-795-276

Puntos obtenidos:\_\_\_\_\_\_\_\_\_ Profesor: Juan Saldaña. Total: 100 puntos

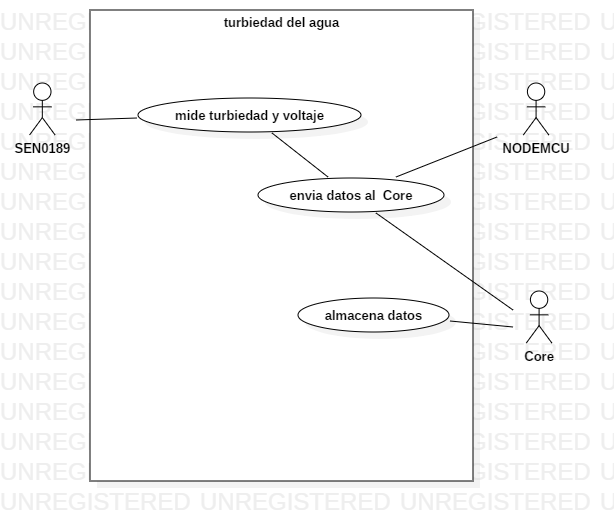
**PROYECTO DE INGENIERÍA DE SOFTWARE**

I. Requerimientos

1. Definición general del sistema

Un sistema que permite realizar una medición del nivel de turbiedad que presenta el agua, de manera que nos indique la cantidad de impurezas en la misma. Esta impureza se calcula mediante la lectura analógica del voltaje por el sensor al ser introducido en agua, si el voltaje es más alto indica menos nivel de turbiedad, pero si el voltaje es bajo indica que hay muchas impurezas en el agua; todo esto es dado por la relación que existe entre le voltaje leído y el NTU (unidad de turbiedad), la cual es una ecuación cuadrática dada por donde “y” representa el valor de NTU y “x” es el voltaje leído.

1. Caso de Uso



II. Diseño

1. Diagrama de Clase



1. Diagrama de Secuencia



1. Diseño del circuito e instrucciones de conexión

A circuit board

Description automatically generated

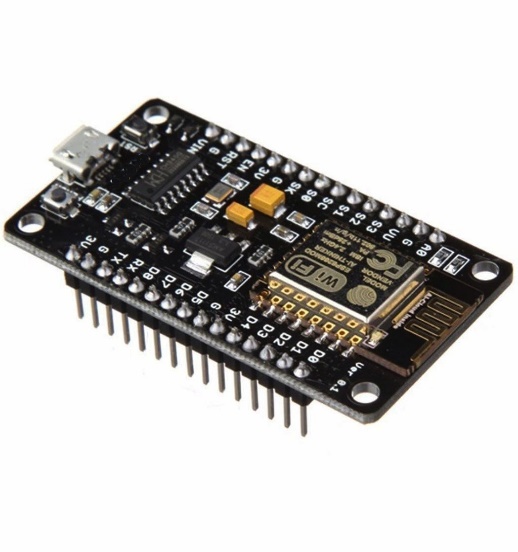
Representa la conexión a tierra, la cual se conecta a uno de los pines de tierra (Ground) del NodeMCU, en este caso en el que está justo al lado del pin VIN

Representa la conexión de voltaje del sensor, al ser un sensor que funciona con entra de 5V, y se conecta en el pin VIN del NodeMCU

Representa la conexión de transferencia de datos, este pin se conecta a una de la entrada analógica del NodeMCU, dado que solo posee uno, se conecta en el pin A0

1. ****Definición de sensores y su funcionalidad

**SEN0189**: El sensor de turbiedad (SEN0189) detecta la calidad del agua midiendo los niveles de turbiedad. Usa una luz para detectar partículas suspendidas en el agua midiendo la transmisión de luz y tasa de dispersión, el cual cambia con la cantidad total de sólidos suspendidos en el agua (TSS). Cuando el TTS incrementa, el líquido de nivel de turbiedad aumenta.

**NodeMCU V1.0: E**s una placa de desarrollo totalmente abierta, a nivel de software y de hardware. Al igual que ocurre con Arduino, en NodeMCU todo está dispuesto para facilitar la programación de un microcontrolador o MCU (del inglés Microcontroller Unit).

El objetivo es programar la MCU o microcontrolador a través del kit o placa de desarrollo. Todo lo demás nos sirve de apoyo para que crear nuestros propios proyectos sea lo más sencillo posible.

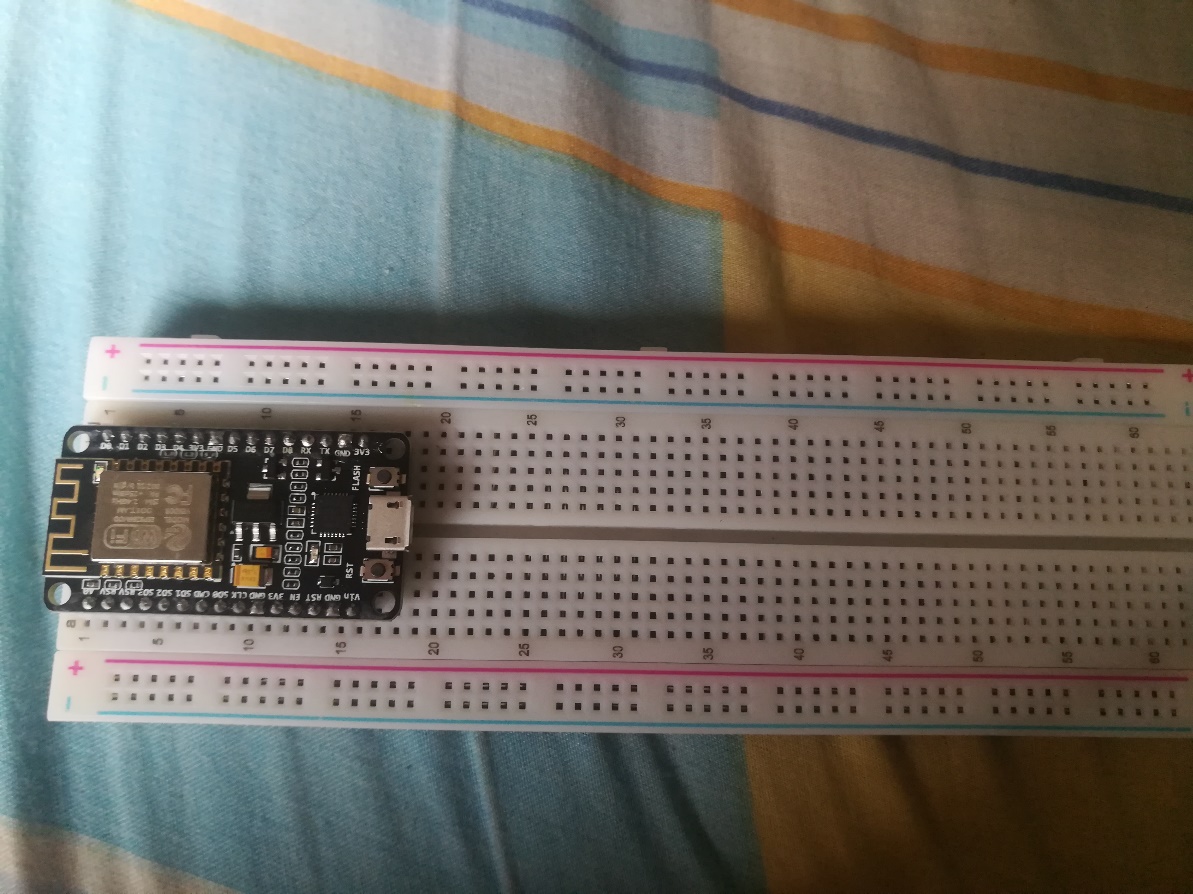
Esto no solo sucede con NodeMCU, las placas como Arduino UNO o Arduino MKR1000 utilizan la misma arquitectura eso si, cada una con sus características particulares.

1. Diagrama y manual de Implementación

Implementación del sensor

1. **Colocar el módulo ESP-8266 en un Protoboard, de manera que en ambos lados tenga pines para conectar**

Figura1. ESP8266, conectado al protoboard



1. **Sacamos el módulo que trae el sensor de turbiedad**



Este módulo dispone de un switch que nos permite cambiar el tipo de lectura ente Analógico y Digital.

Figura 2. Módulo del sensor de turbiedad

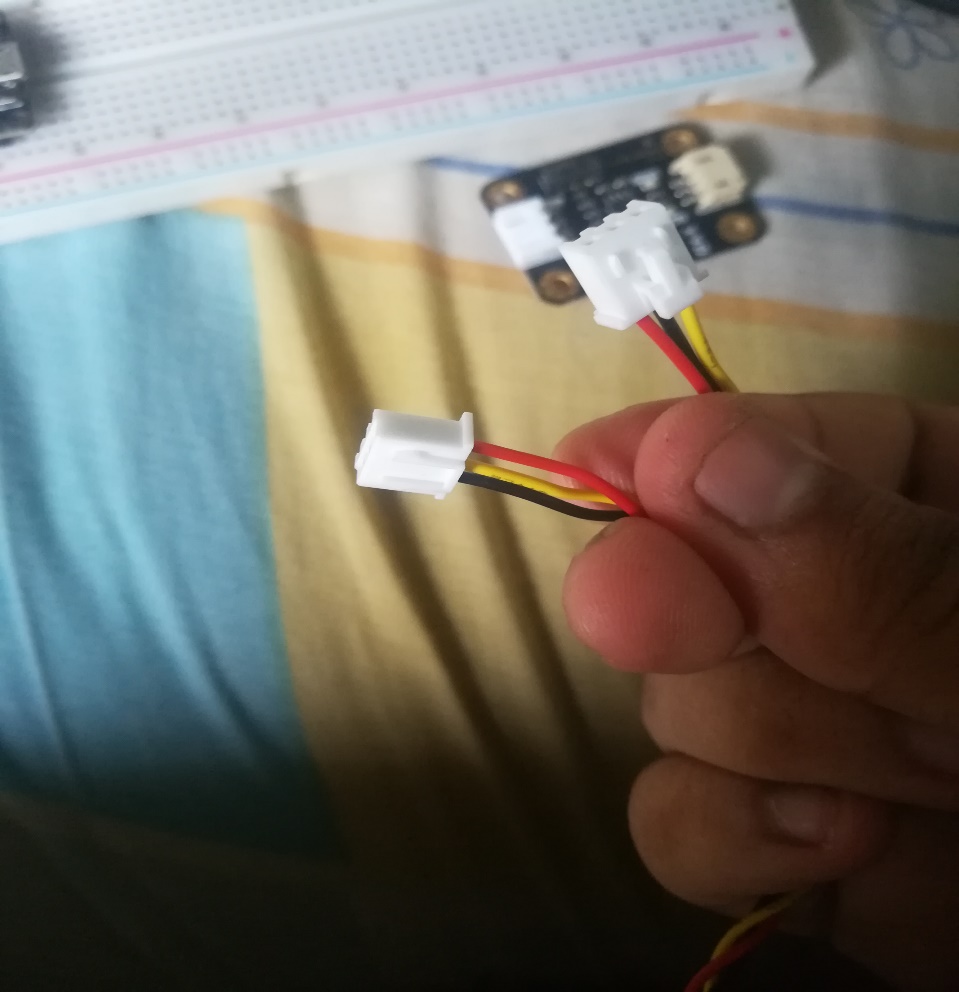
1. **Comprobamos la forma de los conectores para no usar el lado incorrecto**

Figura 3. Conector del módulo al sensor

1. **Unimos el conector al módulo y posteriormente al mismo sensor de turbiedad**

Figura 4. Conexión del conector al sensor

****

1. **Hacemos uso del otro conector para conectarlo en el módulo, el cual será el que dará la conexión al sensor con el Arduino/NodeMCU**

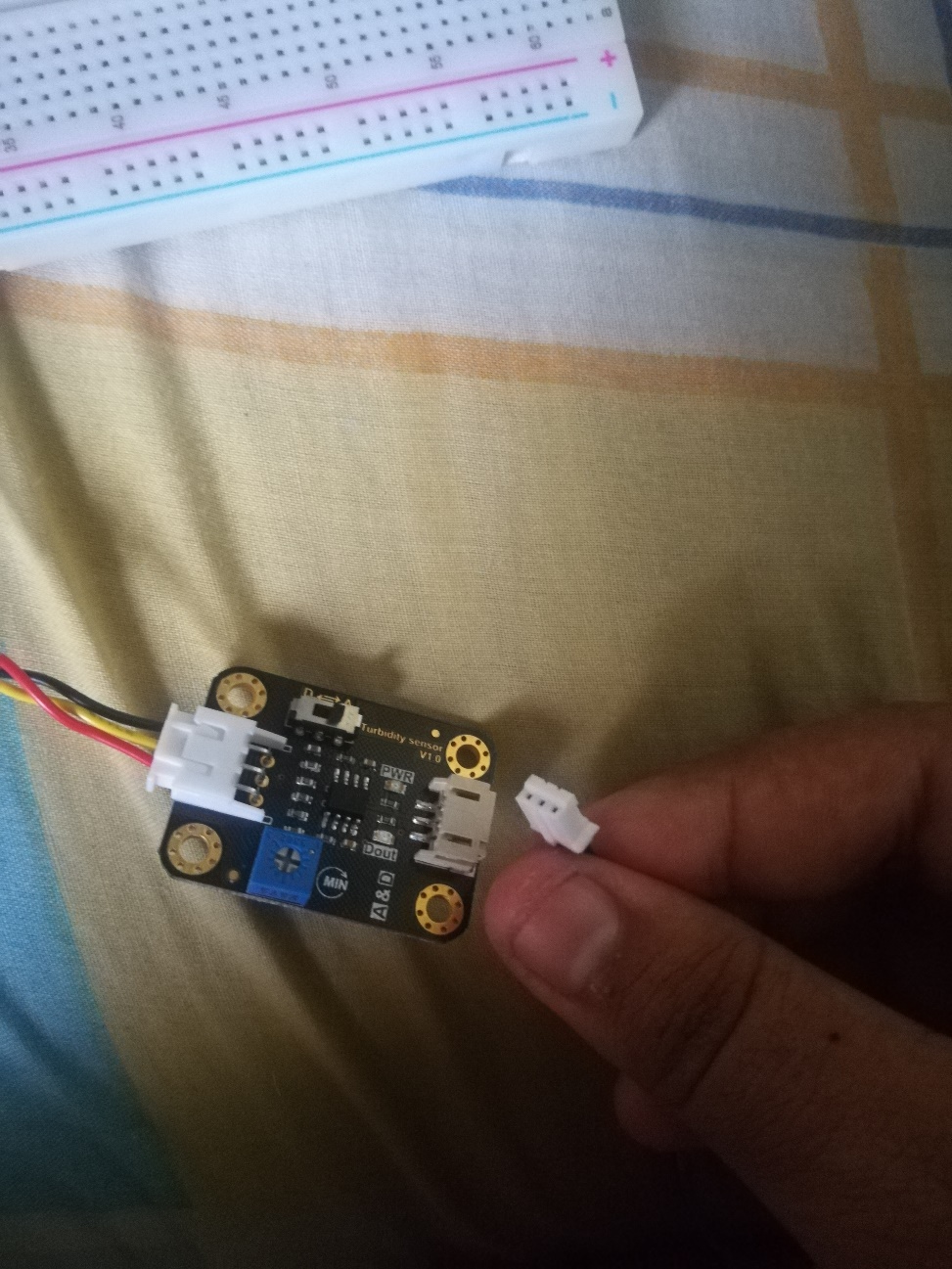
****

Figura 5. Conector del módulo al NodeMCU

1. **Finalmente conectamos el conector usando conectores (jumper) tipo Macho-Macho al NodeMCU**

Se uso el mismo tipo de colores en los conectores para que se pueda representar de manera más adecuada.

En este caso la conexión sería:

-El conector rojo se conecta al pin VIN del NodeMCU

-El conector negro se conecta al pin GROUND ubicado al lado del pin VIN.

-El conector Azul que representa la señal es conectado al pin A0

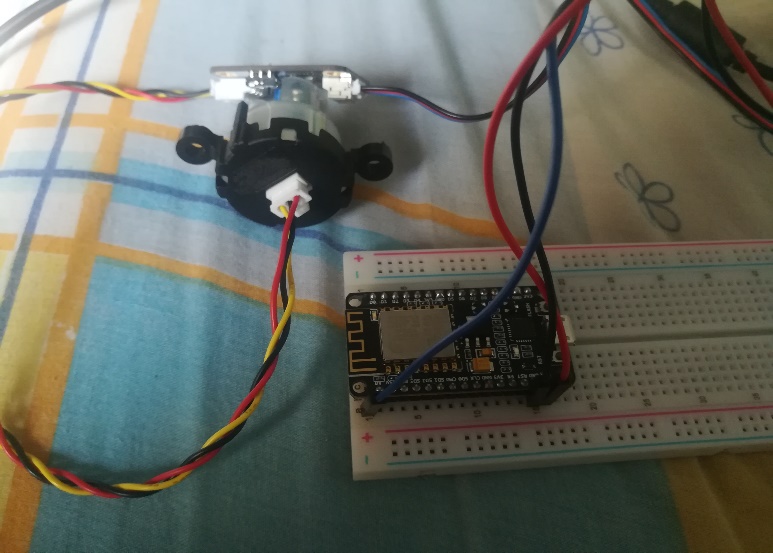
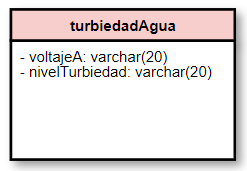
****

Figura 6. Conexión completa

1. Diagrama de base de datos



III. Desarrollo

1. Instrucciones de instalación de herramientas

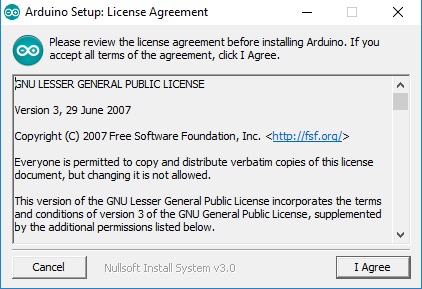
**IDE ARDUINO**

Para poder instalar el IDE de Arduino se debe seguir los siguientes pasos:

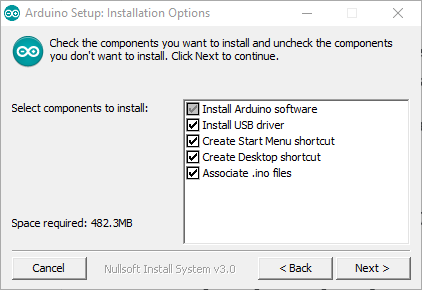
* Ejecuta el instalador del programa y sigue los pasos de instalación.

[](http://arduino.cl/wp-content/uploads/2014/10/2019-02-28-17_00_57-Descargas.png)

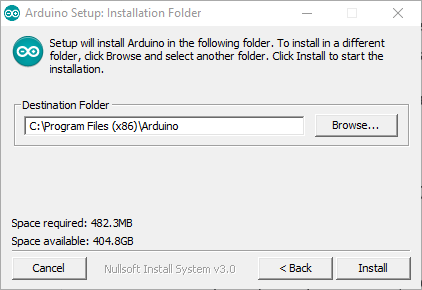
* Acepta los términos y condiciones de la licencia.

[](http://arduino.cl/wp-content/uploads/2014/10/Terminos-licencia.jpg)

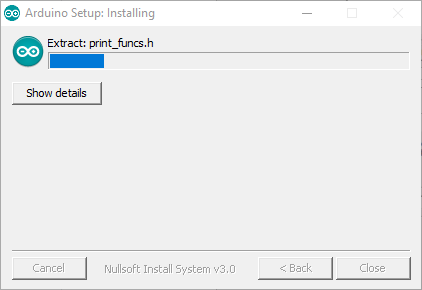
* Selecciona todas las opciones para que instale todos los complementos y drivers necesarios.

[](http://arduino.cl/wp-content/uploads/2014/10/2019-02-28-17_21_46-Arduino-Setup_-Installation-Options.png)

* Selecciona la ruta de instalación y presiona «install».

[](http://arduino.cl/wp-content/uploads/2014/10/2019-02-28-17_23_49-Arduino-Setup_-Installation-Folder.png)

* Espera un par de minutos que termine el proceso de instalación.

[](http://arduino.cl/wp-content/uploads/2014/10/2019-02-28-17_25_07-Arduino-Setup_-Installing.png)

Si estás utilizando la placa [Arduino Picaro](https://www.mcielectronics.cl/shop/product/picaro-arduino-uno-compatible-11289), necesitarás instalar los drivers del chip FTDI. Sigue [estos pasos](http://olimex.cl/website_MCI/static/documents/PICAROxDriverInstall.rar) para completar la instalación.

Si está usando una placa Arduino Uno original, los driver se instalan automáticamente con el IDE.

1. Documentación del código

#include <Wire.h> //se incluye la libreria Wire

#include <ESP8266WiFi.h> //se incluye la librería para el ESP8266

#include "Adafruit\_MQTT.h" //se incluyen la librerias para el uso del Adafruit MQTT

#include "Adafruit\_MQTT\_Client.h"

ADC\_MODE(ADC\_VCC); //Fija el ADC para leer VCC

#define WLAN\_SSID "Skynet" //Se definen las SSID y la Contraseña utilizadas para conectarse a dicha red

#define WLAN\_PASS "2!,y^5#t5<;T$eC\*zBUb4BM)'[N9Em>G"

#define HOST "172.20.160.108" //se define la dirección IP del Host (router)

#define PORT 1883 //se define el puerto de enlace

#define USERNAME "username" //se deja la configuración del usuario y contraseña fijadas

#define PASSWORD "12345678"

WiFiClient client;

Adafruit\_MQTT\_Client mqtt(&client, HOST, PORT, USERNAME, PASSWORD); //se define el cliente del MQTT, con su host, puerto, usuario y contraseña

Adafruit\_MQTT\_Publish voltaje = Adafruit\_MQTT\_Publish(&mqtt, "voltaje"); //Topic utilizado para vincular los datos que se publican en la base de datos

Adafruit\_MQTT\_Publish turbiedad = Adafruit\_MQTT\_Publish(&mqtt, "turbiedad"); //Topic utilizado para vincular los datos que se publican en la base de datos

void MQTT\_connect(); //se declara una función que representa la conexión con el protocolo MQTT

//Declaración de variables

float volt; //se declara una variable flotante para representar el voltaje

float ntu; //se declara una variable flotante para representar la turbiedad medida en NTU

int sensorTurb = 2; //se declara una variable que guarda el pin analógico utilizado A0 (pin 2)

**void setup()** {

Serial.begin(115200); //se inicia la comunicación serial en 115200 baudios

WiFi.forceSleepWake();

WiFi.mode(WIFI\_STA);

Serial.print("Connecting to "); //impresión indicando a que red se está conectando

Serial.println(WLAN\_SSID); //se imprime el nombre de la red

WiFi.begin(WLAN\_SSID, WLAN\_PASS); //se inicializa la conexión con el SSID y la Contraseña puesta

while (WiFi.status() != WL\_CONNECTED) { //Condición usada para definir mientras el enlace con el Wifi este aun sin conexión

delay(500); //retraso de medio segundo

Serial.print("."); //mientras esto pase, imprime un punto

}

Serial.println();

Serial.println("WiFi connected"); //Cuando la conexión se realiza correctamente, se muestra que está conectado

Serial.println("IP address: "); Serial.println(WiFi.localIP()); //Se imprime la dirección IP local

MQTT\_connect(); //se ejecuta la función de conexión del MQTT

}

**void loop()** {

volt = 0; //Se inicializa el voltaje en 0

for(int i=0; i<800; i++){ //El voltage del sensor es ruidoso así que se toma 800 muestras y luego se promedian

volt += analogRead(sensorTurb) \* (3.3/1023.0); //Convierte la lectura analógica (0~1023) a voltaje (0 ~ 3.3V) y se va sumando

}

volt = volt/800; //Se divide el voltaje entre la cantidad de muestras (800) para encontrar el voltaje individual

volt = round\_to\_dp(volt,1); //Se redondea el número de voltios a 1 decimal

if (volt>=3.3){ //Condición fijada a que si el voltaje medido este mayor a 3.3V

ntu=0; //si se cumple la condición, el valor de NTU es 0

}else{ //Sino se cumple

if(volt<2.5){ //Se pregunta si el voltaje es menor a 2.5V,

ntu=3000; //así se fuja cada lectura por debajo de los 2.5V con 3000 NTU

}else{ //Sino no es uno ni otro entonces

ntu= -1120.4\*sq(volt)+5742.3\*volt-4353.8; // Calcula el actual nivel de NTU que viene dada por la función

} //y = -1120.4\*x^2 + 5742.3\*x - 4353.8 donde 'x' es el valor del voltaje

}

voltaje.publish(volt); //se envía la información del voltaje para ser publicado por el core

Serial.print(volt); //se imprime el voltaje por serial

Serial.println(" V"); //se imprime la unidad de voltaje

turbiedad.publish(ntu); //se envía la información del turbiedad para ser publicado por el core

Serial.print(ntu); //se imprime el nivel de NTU por serial

Serial.println(" NTU"); //se imprime la unidad de NTU

delay(10000); //retraso de 1 minuto

}

**float round\_to\_dp**(float in\_value, int decimal\_place) //función para redondeo de partes decimales

{

float multiplier = powf(10.0f, decimal\_place); //se define un multiplicador con base 10, potencia que sería el valor de la parte decimal enviada

in\_value = roundf (in\_value\*multiplier)/multiplier; //el valor enviado a redondear se le aplica un redondeo flotante multiplicando el valor por el multiplicador, y posteriormente se divide entre el multiplicador

return in\_value; //se regresa el valor nuevo redondeado

}

**void MQTT\_connect()** { //función para conexión del MQTT

int8\_t ret; //se declara la variable ret como un entero de 8 bits

if (mqtt.connected()) { //si el cliente MQTT está conectado

return;

}

Serial.print("Connecting to MQTT... "); //mensaje indicando la conexión con el protocolo MQTT

uint8\_t retries = 3; //se declara la variable de intentos en 3

while ((ret = mqtt.connect()) != 0) { // la conexión enviará 0 cuando esté conectado

Serial.println(mqtt.connectErrorString(ret)); //se imprime un mensaje de error

Serial.println("Retrying MQTT connection in 5 seconds..."); //se imprime un mensaje de reconexión

mqtt.disconnect(); //se desconecta el MQTT

delay(5000); // retraso de 5 segundos

retries--; // se disminuye el conteo

if (retries == 0) { si el conteo de intentos es 0

// basicamente muere y espera por un reseteo

while (1);

}

}

Serial.println("MQTT Connected!"); //se imprime que la conexión del MQTT está establecida

}

1. Documentación técnica del sensor y valores leídos

**Sensor de Turbiedad del Agua (SEN0189)**

Especificaciones

* Voltaje de Operación: 5V DC
* Corriente de Operación: 40mA (MAX)
* Tiempo de respuesta: <500ms
* Resistencia a Insolación: 100M (Min)
* Métodos de Salida:

Salida Analógica: 0-4.5V

Salida Digital: Señal de nivel Alto/Bajo (puedes ajustar la salida de valores ajustando el potenciómetro)

* Temperatura de Operación: 5°C ~ 90°C
* Temperatura de Almacenaje: -10°C ~ 90°C
* Peso: 30g
* Dimensiones del Módulo: 38mm\*28mm\*10mm

Dentro de los datos leídos se tiene cierto rango de consideraciones, por defecto el sensor mide valores de 0 ~ 4.2V donde al momento en que va bajando el voltaje el valor de NTU va aumentando

Por ejemplo:

* Si el sensor mide 4.2V el valor que arrojaría 0 NTU
* Si el sensor mide un voltaje menor o igual a 2.5V, el valor que arrojaría seria 3000 NTU

En nuestro caso, ya que el módulo NodeMCU en el caso de la entrada analógica A0 que posee, solo permite un voltaje máximo de 3.3V, por lo que la ecuación para calcular el NTU debe variarse un poco.

1. Pruebas
   1. A picture containing player, outdoor

      Description automatically generatedMatriz de avance semanal por grupo
2. Gestión del Proyecto
   1. Creación y Actualización de un repositorio compartido en GitHub
   2. Creación de carpeta de desarrollo por módulo
   3. Subir su información al repositorio
   4. Calendarización (Documento integrado con la información de cada participante)
3. Manual de Usuario
   1. Manual de usuario.

*Mateo 8,5-13 Señor, no soy digno de que entres a mi casa, pero una palabra tuya bastara para sanarme.*