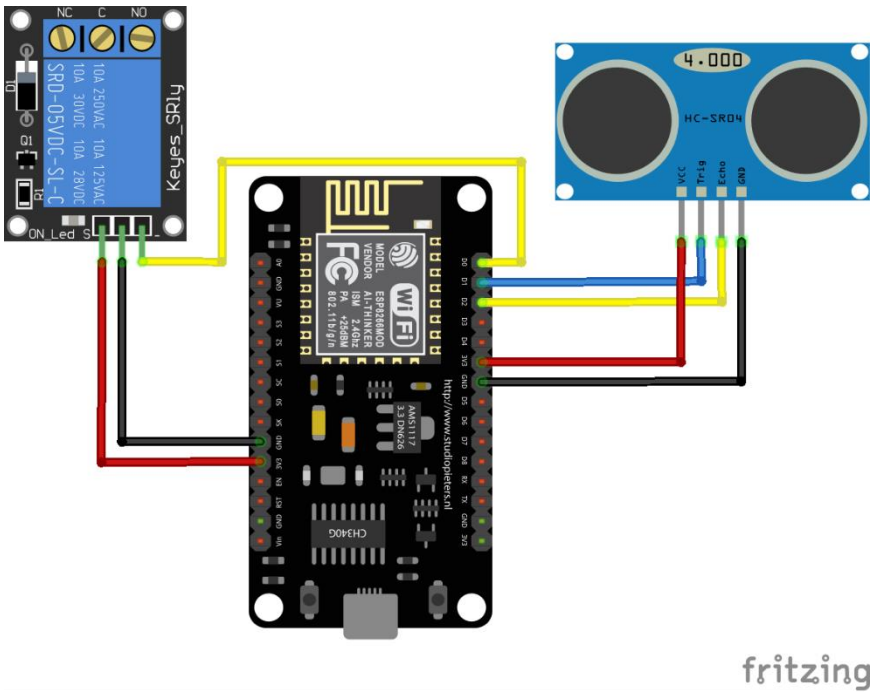
LAPORAN PROJECT PKL BULAN SEPTEMBER

Keterangan Project yang dibuat :

1. Membuat rangkaian ESP8266 dengan sensor HCSR-04 dan Relay 1 Channel.
2. Membuat tampilan dashboard pada Node Red untuk mengendalikan atau memasukan maximal jarak pada sensor HCSR-04.
3. Membuat nilai maximal jarak tetap tersimpan didalam controller meskipun controller sempat di matikan atau di reset.

1. Rangkaian Schematic pada Fritzing



Gambar 1.1

Perangkat keras yang digunakan :

1. ESP8266
2. Sensor HCSR-04
3. Relay 1 Channel 5V
4. Breadboard
5. Kabel Jumper

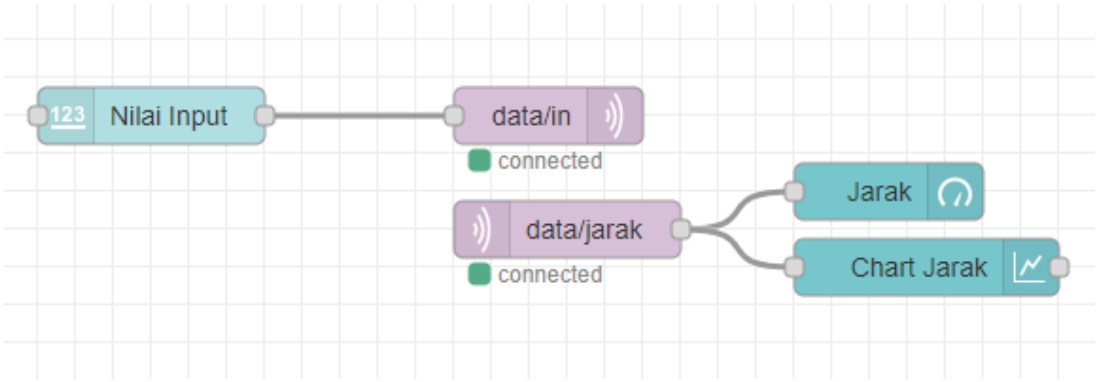
Keterangan Pin yang digunakan :

Pin D0 ➔ Pin Relay

Pin D1 ➔ untuk Trigger Pin

Pin D2 ➔ untuk Echo Pin

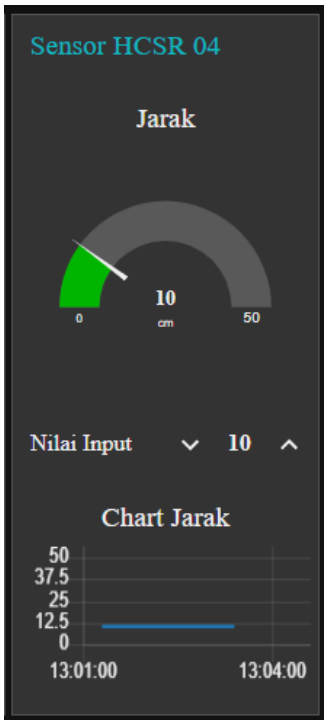
2. Rangkaian Dashboard pada Node Red



Gambar 2.1

Keterangan Diagram diatas :

1. Dashboard Numeric ➔ Untuk nilai maximal yang diinputkan melalui node red.
2. MQTT Out dengan Topic data/in.
3. MQTT In dengan Topic data/jarak.
4. Dashboard Gauge & Chart jarak untuk indikator jarak yang terdeteksi.



Gambar disamping adalah hasil dari rangkaian diagaram
Nilai yang tampil pada dashboard tersebut didapat dari Sensor HCSR-04

Gambar 2.2

3. **Script Program**

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <HCSR04.h>
#include <EEPROM.h>

//INISIALISASI PIN
#define triggerPin  D1
#define echoPin     D2
#define Relay D0

//MQTT TOPIC
#define MQTT_JARAK "data/jarak"
#define MQTT_DURASI "data/subscribe"
#define MQTT_RELAY "data/in"

long durasi, jarak1;
String payloadTemp;
const char* ssid = "PKL ATMDUINO";
const char* password = "gedebacot";
const char* mqtt_server = "broker.mqtt-dashboard.com";
unsigned long startMillis=0; //variabel menyimpan milidetik terakhir dari loop

int target;

WiFiClient espClient;
PubSubClient client(espClient);

void setup_wifi() {
  delay(10);
  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 JARAK_config(){
    unsigned long currentMillis = millis();// variabel mengambil waktu yang berjalan dan
                                                    menyimpan di variabel current millis
    if (currentMillis - startMillis >= 2000) { //setiap milis_sekarang - hitungan_milis yang
                                        mencapai nilai lebih besar atau sama dengan 2000
    startMillis = currentMillis;
    digitalWrite(triggerPin, LOW);
    delayMicroseconds(2);
    digitalWrite(triggerPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerPin, LOW);

    durasi = pulseIn(echoPin, HIGH);
    jarak1 = (durasi/2) / 29.1;

    Serial.print("Jarak :");
    Serial.print(jarak1);
    Serial.println(" cm");

      if (jarak1 <= (EEPROM.read(0))) {
        digitalWrite(Relay, HIGH); //data target diambil dari topic mqtt relay
        Serial.println("Relay Mati");
      }

      else {
        digitalWrite(Relay, LOW);
        Serial.println("Relay Hidup");
      }
  }

  client.publish(MQTT_JARAK, String(jarak1).c_str());
  client.publish(MQTT_DURASI, String(durasi).c_str());
}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()){
    Serial.print("Attempting MQTT connection...");
    //String clientId = "belajarmqttt";
    //clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect("espClient"))
```

```
    {
      Serial.println("connected");
      client.subscribe(MQTT_RELAY);
      //client.subscribe(MQTT_JARAK);
      //client.publish("data/jarak", "Online");

    }
    else
    {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println("try again in 5 Seconds");
      delay(5000);
    }
  }
}

void callback(char *topic, byte *payload, unsigned int length){
  Serial.print("Message arrived in topic: ");
  Serial.println(topic);
  Serial.print("Message : ");
  String payloadTemp;

  for (int i = 0; i < length; i++) // fungsi for disini sebagai mengukur lenght data nya
  {
    Serial.print((char)payload[i]);
    Serial.println();
    if((char)payload[i] !="");
      payloadTemp += (char)payload[i];

      if (String(topic) == MQTT_RELAY){
        target = payloadTemp.toInt();
        Serial.print("Data: ");
        Serial.println(target);
        EEPROM.write(0, target);
        EEPROM.commit();
      }
  }
}

void setup() {
  Serial.begin(115200);
  EEPROM.begin(512);
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(Relay, OUTPUT);
```

```
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);

  EEPROM.read(0); //berfungsi untuk membaca status pada eeprom di memori 0
  Serial.print("Data yang disimpan : ");
  Serial.println(EEPROM.read(0));
  delay(1000);
}

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

4. Cara Kerja Program & Controller

1. MQTTBox dengan Node-Red.
   Cara kerja MQTTBox disini hanya sebagai tool checker, apakah data inputan dari Node-RED berhasil masuk melalui topic yang digunakan. Data tersebut dikirim melalui topic “data/in” dan pada MQTTBox bagian subscribe diisi topic yang sama.

2. Rangkaian Controller
   Pada bagian controller terdapat Node MCU ESP8266, Sensor Jarak HCSR-04, dan Relay 1 Channel. Disaat ESP8266 menerima nilai inputan dari Node Red, ESP8266 sebagai control utama akan langsung membaca data tersebut dan diteruskan ke dalam sensor dan Relay, jika jarak yang sudah ditentukan maka Relay akan mati jika kurang dari jarak benda dan akan menyala jika lebih dari jarak benda.

3. Memori EEPROM
   Pada dasarnya, sebuah EEPROM adalah sebuah “lemari” bertingkat yang memiliki alamat yang didalamnya ada sebuah data.
   EEPROM.write (addr, val) ➔ untuk menulis data ke EEPROM.
   EEPROM.read(address) ➔ untuk membaca/mengambil data dari EEPROM.
   Setelah data berhasil disimpan kedalam memori EEPROM maka data tersebut tidak akan hilang meskipun catu daya dihilangkan, direset, atau dimatikan kecuali kita menulis ulang program pada memori EEPROM tersebut.

5. Uji Coba pada Rangkaian Controller

- Uji coba pertama kali dilakukan untuk tester apakah nilai inputan maximal jarak masuk kedalam aplikasi MQTTBox

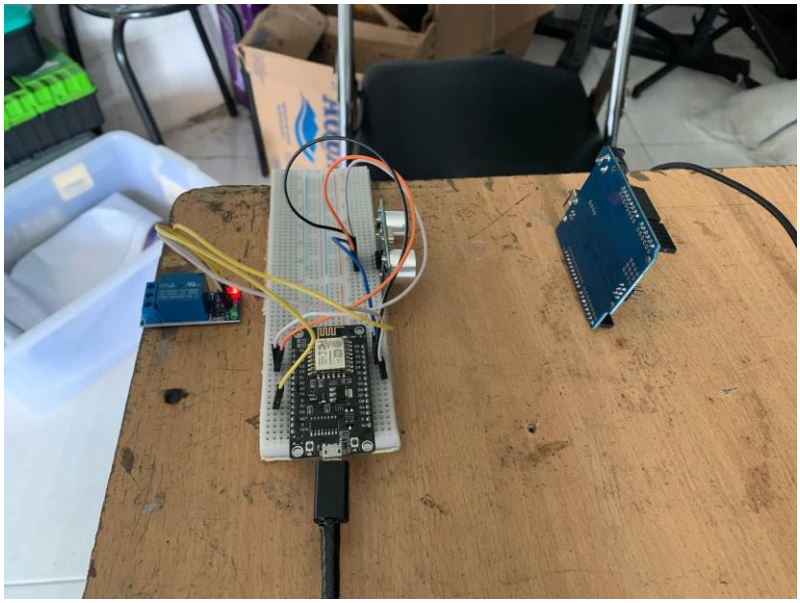
Gambar 5.1

Gambar 5.2

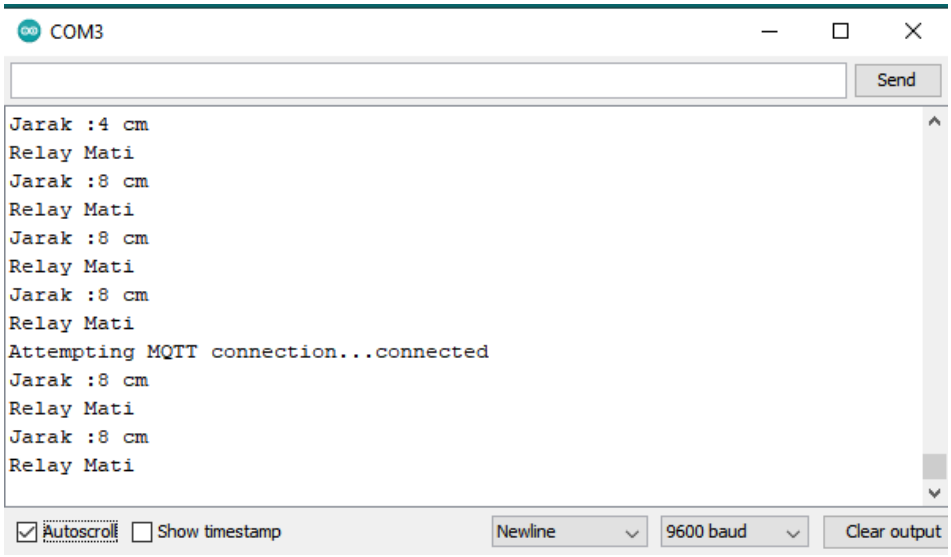
Gambar 5.3

- Pada Gambar 5.1 terdapat nilai input 10 cm yang berarti nilai tersebut adalah nilai maximal yang ditentukan untuk Sensor HCSR-04
- Gambar 5.2 adalah Nilai Input yang tampil di Aplikasi MQTTBox.
- Gambar 5.3, Nilai Input dari node red tampil di serial monitor Arduino IDE

- Uji coba yang kedua yaitu nilai input berfungsi normal atau tidak.

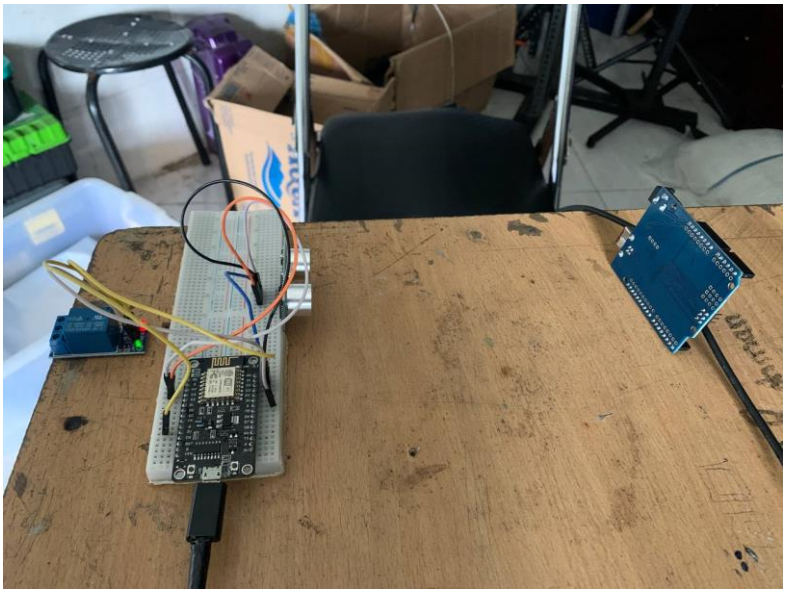


Gambar 5.4

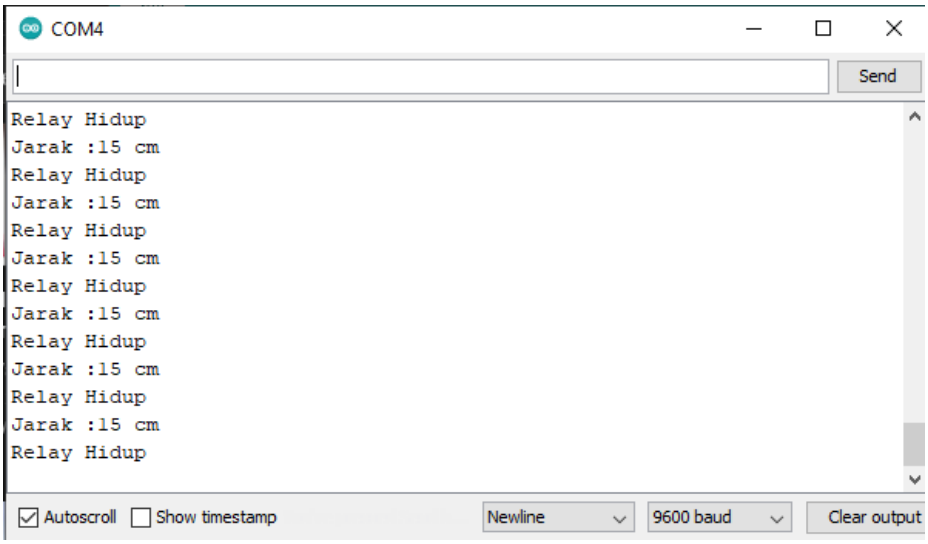


Gambar 5.5

- Gambar 5.4, Jarak benda 8 cm dan nilai input sensor 10 (jika kurang dari 10 mati), maka Relay Mati
- Gambar 5.5, Jarak benda dan status relay tampil pada Serial Monitor Arduino IDE

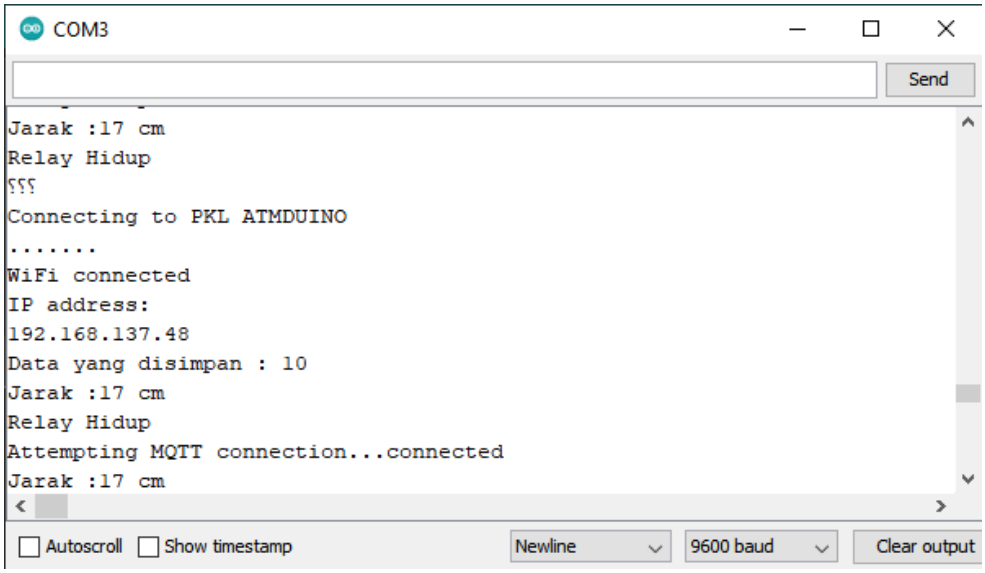


Gambar 5.6



Gambar 5.7

- Gambar 5.6, Jarak benda 15 cm dan nilai input sensor 10, maka Relay Nyala
- Gambar 5.7, Jarak benda dan status relay tampil pada Serial Monitor Arduino IDE



Gambar 5.8

- Uji coba yang terakhir yaitu pengujian memori EEPROM, pada gambar diatas controller setelah di reset akan menghubungkan ulang otomatis dan menampilkan data yang terakhir di inputkan. Data yang tampil tersebut adalah data yang tersimpan didalam memori EEPROM. Jadi tidak perlu mengatur melalui dashboard lagi untuk menentukan nilai maximal jarak controller.