Welkom in de handleiding van de code

in deze handleiding zal ik de belangrijkste stukken uitleggen:

De lijndetectie:

Als deze sensor een donker oppervlakte tegenkwam gaf deze een 1 signaal en ij een wit oppervlakte gaf deze een 0 signaal. adhv deze info heb ik de code samengesteld, wanneer deze normaal op de lijn zit zal deze, deze mooi volgen aan een normale snelheid, gaat deze van de lijn af dan gaat deze of naar links sturen of naar rechts sturen dmv zijn snelheden te veranderen of de motor achteruit te laten draaien, dit gebeurt wanneer er een “**-**” voor staat bij de SPEED_LVx, bv **-**SPEED_LV1

```
if (distance[0] > OBSTACLE_DISTANCE && woord != "Stop")  //There is non
obstacle ahead
  {
    for (int i = 0; i < LEDS_COUNT; i++) {
      strip.setLedColorData(i, 0, 255, 0);  //Set the color of the WS2812
    }
    strip.show();  //Call WS2812 to display colors
    Track_Read();
    switch (sensorValue[3]) {
      case 5:                                                    //101
        Motor_Move(SPEED_LV1, SPEED_LV1, SPEED_LV1, SPEED_LV1);  //Move
Forward
        if (client.publish(auto_topic, "De auto rijdt!")) {
          Serial.println("Data Send!");
        }
        break;
      case 7:                    //111
        Motor_Move(0, 0, 0, 0);  //Stop
        if (client.publish(auto_topic, "De auto Stopt!")) {
          Serial.println("Data Send!");
        }
        break;
      case 0:                    //000
        Motor_Move(0, 0, 0, 0);  //stop bij dwarslijn
        if (client.publish(auto_topic, "De auto staat bij een dwarslijn!")) {
          Serial.println("Data Send!");
        }
```

```
        delay(5000);
        Motor_Move(SPEED_LV1, SPEED_LV1, SPEED_LV1, SPEED_LV1);
        if (client.publish(auto_topic, "De auto rijdt weer verder!")) {
          Serial.println("Data Send!");
        }

        break;
      case 4:                                                    //100
        Motor_Move(SPEED_LV4, SPEED_LV4, -SPEED_LV3, -SPEED_LV3);  //Turn
Right
        break;
      case 6:                                                    //110
        Motor_Move(SPEED_LV3, SPEED_LV3, -SPEED_LV2, -SPEED_LV2);  //Turn
Right
        break;
      case 1:                                                    //001
        Motor_Move(-SPEED_LV3, -SPEED_LV3, SPEED_LV4, SPEED_LV4);  //Turn Left
        break;
      case 3:                                                    //011
        Motor_Move(-SPEED_LV2, -SPEED_LV2, SPEED_LV3, SPEED_LV3);  //Turn Left
        break;
      default:
        Motor_Move(0, 0, 0, 0);  //Stop the car
        for (int i = 0; i < LEDS_COUNT; i++) {
          strip.setLedColorData(i, 255, 0, 0);  //Set the color of the WS2812
        }
        strip.show();  //Call WS2812 to display colors
        break;
    }

  } else {
    if (woord == "Stop") {
      Motor_Move(0, 0, 0, 0);  //Stop the car
      if (client.publish(auto_topic, "auto gestopt door de controlekamer!")) {
        Serial.println("Data Send!");
      }
      Motor_Move(0, 0, 0, 0);  //Stop the car
      for (int i = 0; i < LEDS_COUNT; i++) {
        strip.setLedColorData(i, 255, 153, 51);  //Set the color of the WS2812
      }
      strip.show();  //Call WS2812 to display colors
```

```
      if (client.publish(auto_topic, "Object voor de auto, hierdoor staat deze
stil!")) {
        Serial.println("Data Send!");
      }
    }
```

De collision detection:

```
//Get distance values
  void get_distance(int car_mode) {
    distance[0] = Get_Sonar();
  }
```

Via de library kan ik met deze eenvoudige code de ultrasone sensor laten detecteren. Het werkt als volgt, de sensor zendt een signaal terug als deze op iets bots keert dit terug naar de ontvanger en zo weten wij wat de afstand is van het object, echter gebeurt er in de code op de achter dan deze afstand gedeeld door twee gaat omdat hij het parkoer 2 keer aflegt heen en terug. Maar met deze gemeente afstand zal het voertuig of gaan rijden of gaan stilstaan.Zoals je in de code van de lijndetectie ziet staan staat deze in een grote if statement.

MQTT:

Om natuurlijk het voertuig te kunnen laten communiceren via MQTT via node red dat draait op de raspberry pi heb je deze code nodig:

eerst stellen we al onze basisinstellingen in:

```
const char* ssid = "KamerGlenn2";  //voor nu test ik het nog op mijn wifi maar
kan dit altijd aanpassen
const char* password = "GlennIsCool";
const char* auto_topic = "home/toonzaal/auto";
const char* auto_batt = "home/toonzaal/autobatt";
const char* controlekamer_topic = "home/toonzaal/controlekamer";
const char* mqttServer = "192.168.137.184";
const int mqttPort = 1883;
const char* mqttUser = "glenn";
const char* mqttPassword = "glenn";
const char* clientID = "client_auto";  // MQTT client ID
```

Dan gaan we verbinding maken met de wifi, dit gebeurt in de void setup():

```
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
  }
  Serial.println("Connected to the WiFi network");
```

Dan gaan we verbinden met de MQTT

```
client.setServer(mqttServer, mqttPort);
```

```
client.loop();
    while (!client.connected()) {
      Serial.println("Connecting to MQTT...");
      if (client.connect("ESP32Client", mqttUser, mqttPassword)) {
        Serial.println("connected");
      } else {
        Serial.print("failed with state ");
        Serial.print(client.state());
        delay(2000);
      }
```

alsook subscriben op een topic voor de info die wij binnenkrijgen van de raspberry pi:

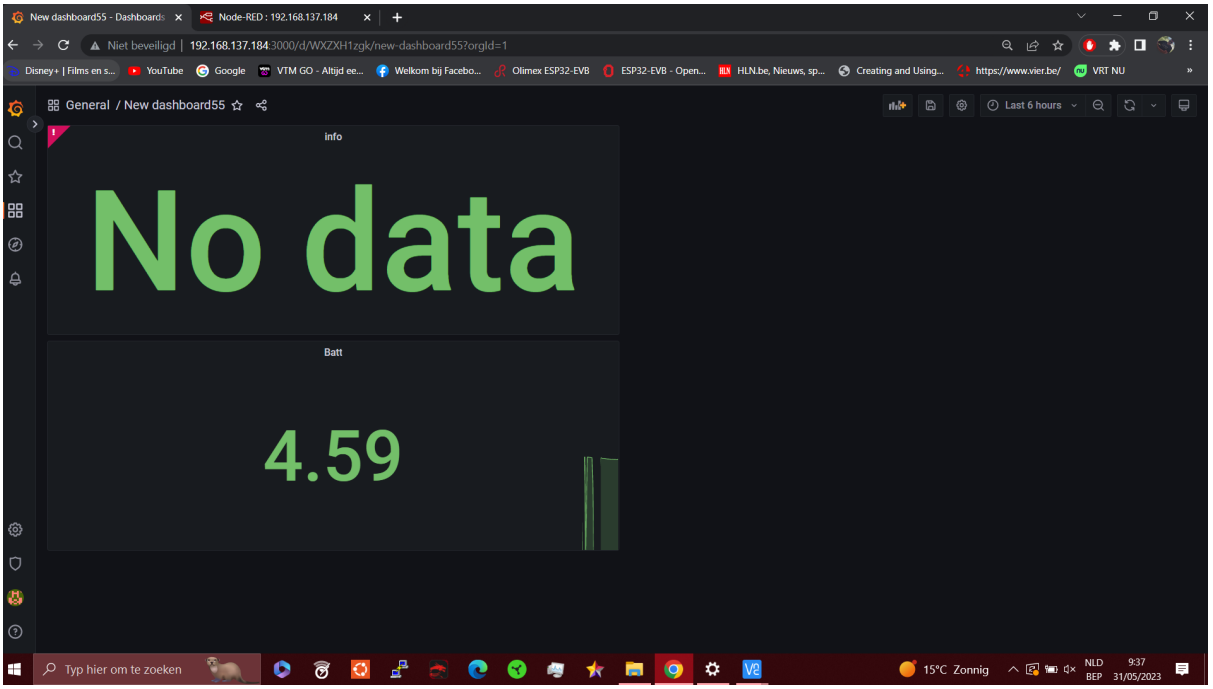
```
client.subscribe(controlekamer_topic);
```

Eenmaal verbonden, kunnen we info doorsturen bv:

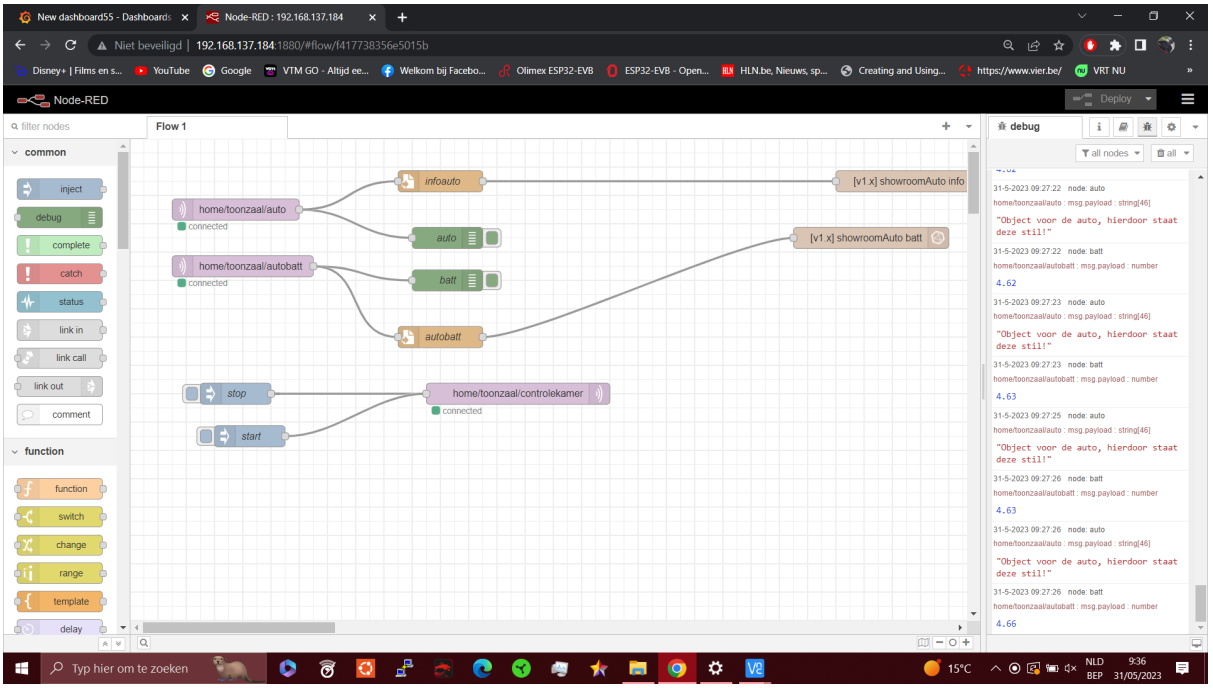
```
if (client.publish(auto_topic, "Object voor de auto, hierdoor staat deze
stil!")) {
        Serial.println("Data Send!");
      }
```

Het gedeelte op de raspberry pi:

daarop hebben wij grafana, nodered en influxdb geïnstalleerd
grafana voor de data die in de influxdb data base staat mooi te kunnen weergeven
nodered voor een webinterface zodat we overal op het netwerk kunnen interacteren



grafana



nodered