Smart Dustbin with IoT Notifications

As we get busy with our day to day lives, we tend to forget to empty the garbage bin. What if you can check the garbage level from your phone without having to open the bin and check it? This Smart Dustbin does that for you. The level can be viewed via the Blynk application in your phone. This dustbin also opens automatically when it receives the signal and closes its lid.

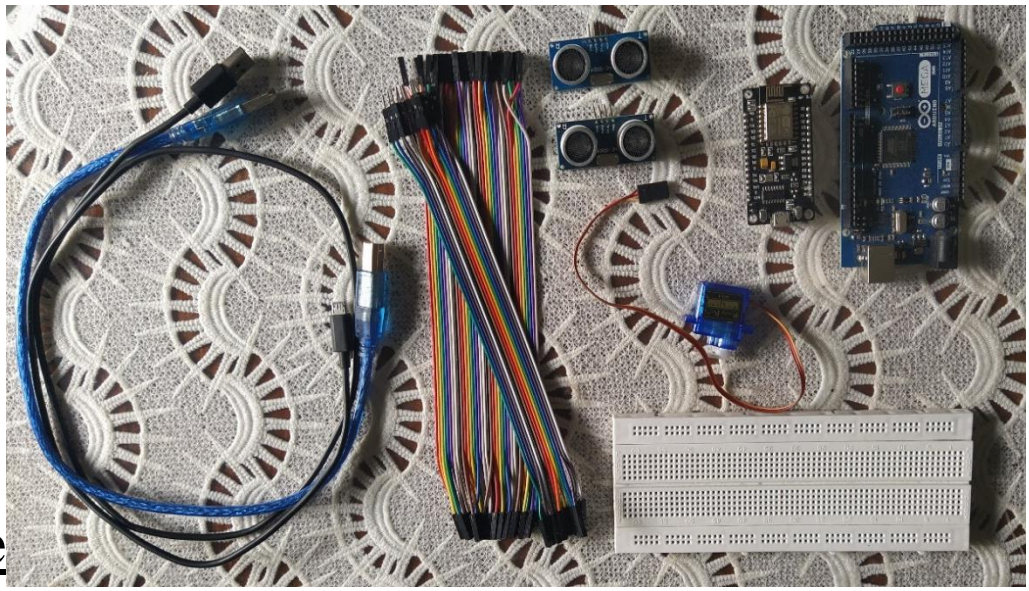
Hardware Specifications

- Arduino Mega 2560
- Ultrasonic Sensors -2
- NodeMCU 0.9
- Cables and Connectors
- Breadboard
- Bin Frame
- Servo Motor SG90

Software Specifications

- Arduino Compiler
- NodeRED / FRED
- Blynk

Conne



Sensor A – Distance

VCC – VV (NodeMCU)

Trig – D1 (NodeMCU)

Echo – D2 (NodeMCU)

GND -GND (NodeMCU)

Sensor B – Garbage Level (lid)

VCC – VV (NodeMCU)

Trig – D5 (NodeMCU)

Echo – D6 (NodeMCU)

GND – GND (NodeMCU)

Servo Motor

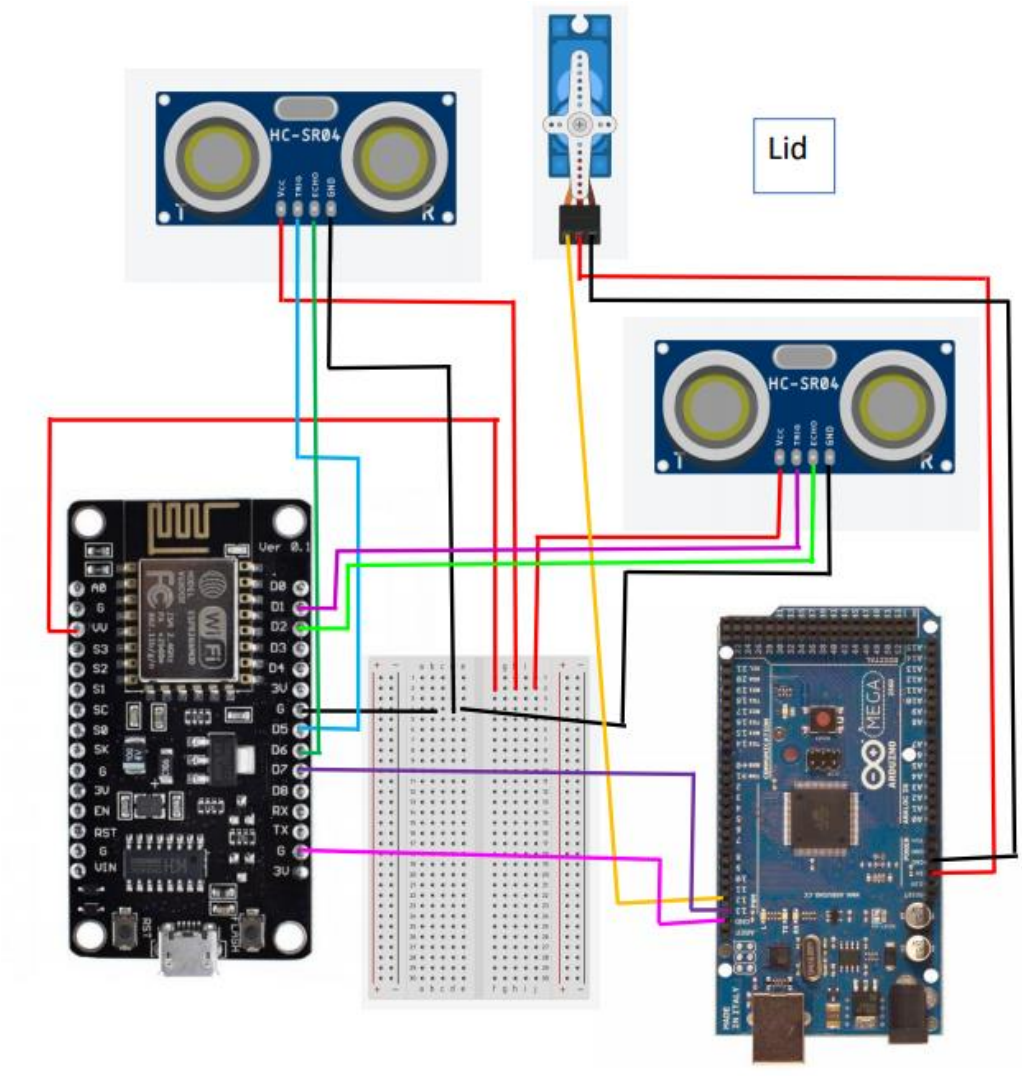
Red- 5V (Arduino)

GND – GND (Arduino)

Yellow -12 (Arduino)

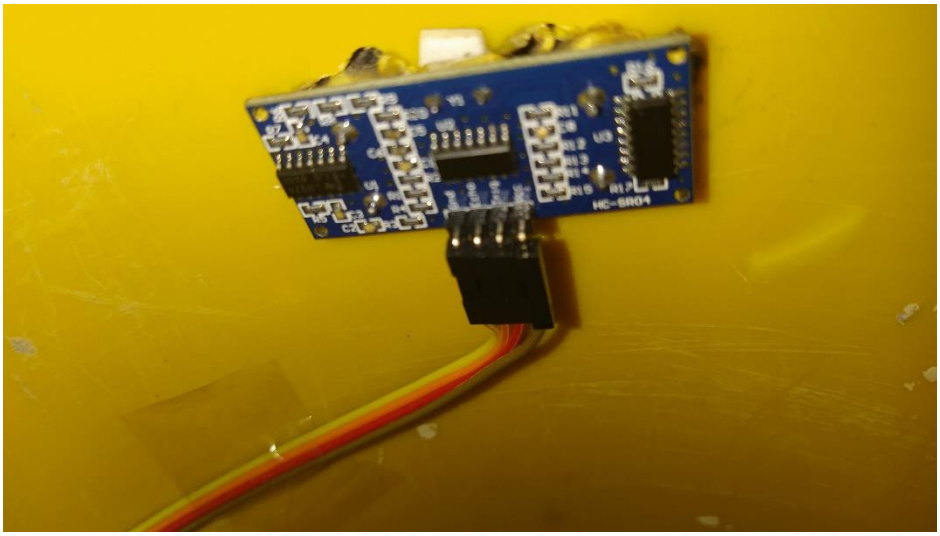
Arduino 13 – D7 NodeMCU

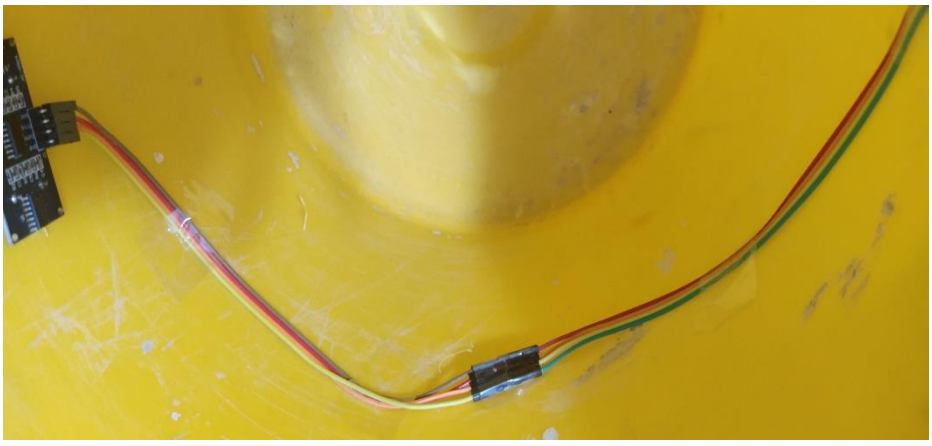
Arduino GND- NodeMCU GND



Step 01

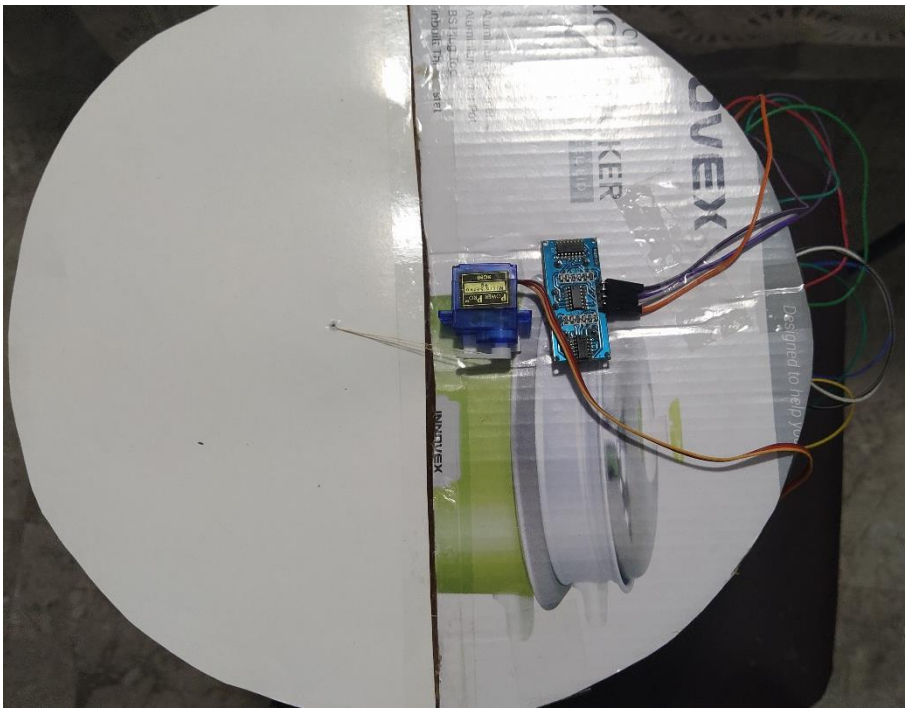
Attach the ultrasonic sensor A to the dustbin.

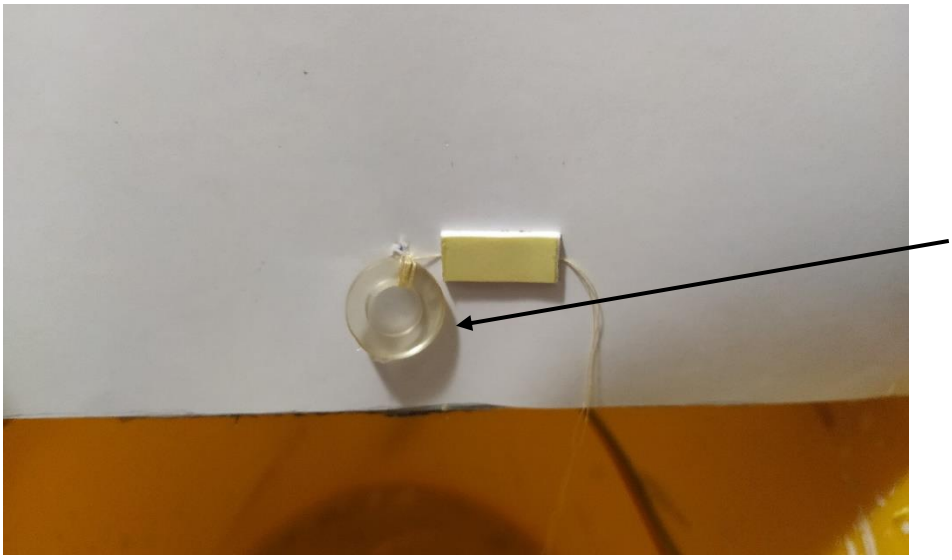




Step 02

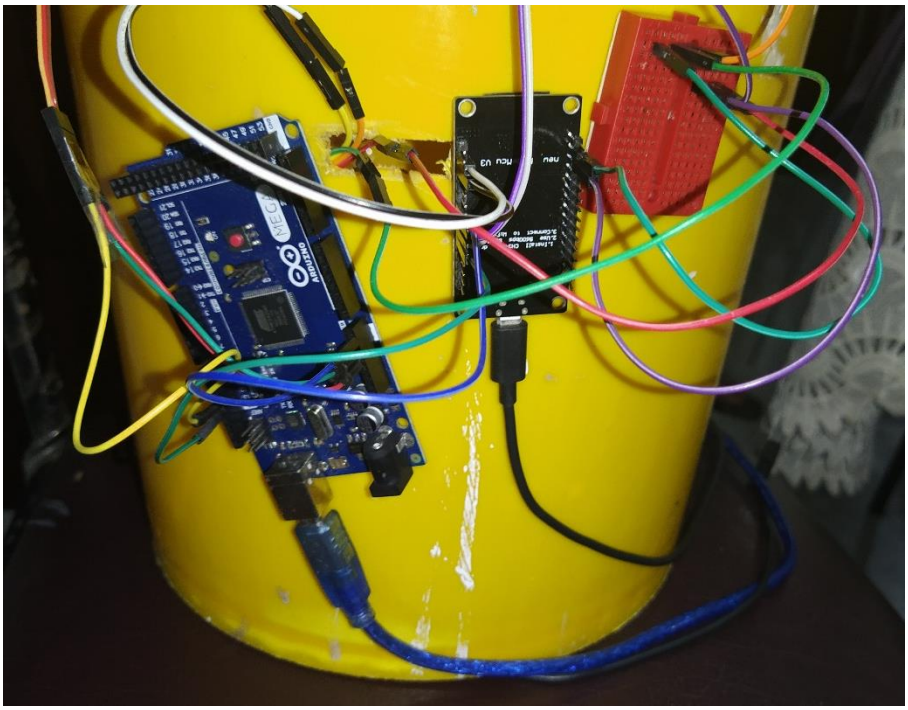
Attach the ultrasonic sensor B and the servo motor on the lid.





Step 03

Connect sensors and the servo motor to NodeMCU and Arduino Mega 2560.



The codes are below. Upload it separately to the Arduino Mega2560 and the NodeMCU. Do not remove the cables from the PC/ Laptop as I have taken power from it.

Step 04

Open the Blynk app and make a new project. Add 2 Labeled Value Displays and 1 LCD display.

1st Labeled Value Display – Name – Garbage Level

Input – V1

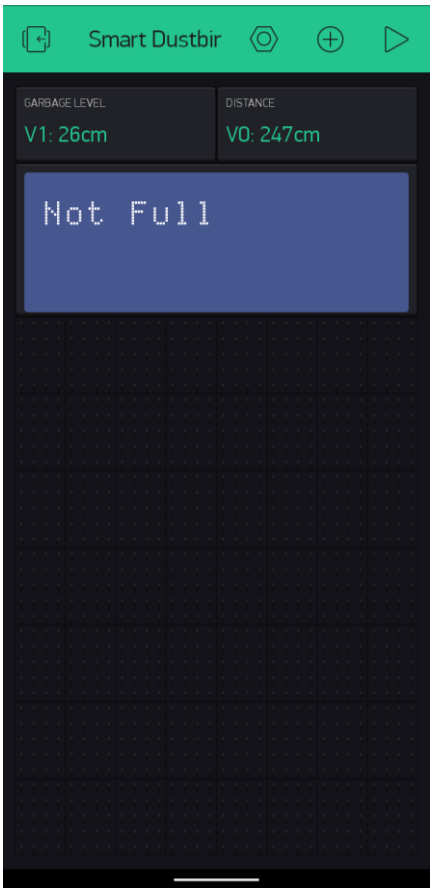
Label – cm

2nd Labeled Value Display – Name – Distance

Input – V0

Label – cm

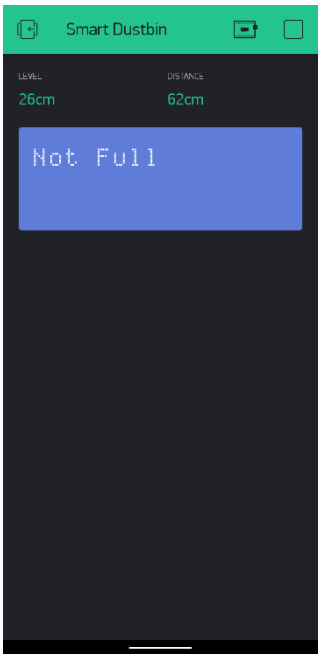
LCD Display – (0) V2



Step 05

Open NodeRED / FRED and create the following nodes. Change the Blynk Auth Token Key with yours.

Done! Run the program and check it through your phone.



Arduino Code for Smart Dustbin

```
#include <Servo.h>

Servo myservo;

int esp8266_D7 = 0;
int opened = 0;
void setup() {
 pinMode(13,INPUT);
 myservo.attach(12);
 myservo.write(0);
 delay(2000);
}

void loop() {

   esp8266_D7 = digitalRead(13);

   if (esp8266_D7 == HIGH)
   {
     myservo.write(180);
     delay(1000);
   }

   if (esp8266_D7 == LOW)
   {
     myservo.write(0);
     delay(1000);
   }
```

NodeMCU Code

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



const char* ssid = "Enter your wifi ssid";
const char* password = "Enter your wifi password";
const char* mqtt_server = "broker.hivemq.com";

WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE  (500)
char msg[MSG_BUFFER_SIZE];
int value = 0;



//#include <Servo.h>
//Servo myservo;

const int trigPinA = D1; // Trigger Pin of Ultrasonic Sensor
const int echoPinA = D2; // Echo Pin of Ultrasonic Sensor

const int trigPinB = D5; // Trigger Pin of Ultrasonic Sensor //lid
const int echoPinB = D6; // Echo Pin of Ultrasonic Sensor



// holds the current count value for our sketch
int count = 0;

long HCSR04_B_Unit()
{
  long duration, cm;
```

```
 pinMode(trigPinB, OUTPUT);
 digitalWrite(trigPinB, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPinB, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPinB, LOW);
 pinMode(echoPinB, INPUT);
 duration = pulseIn(echoPinB, HIGH);
 cm = microsecondsToCentimeters(duration);
 Serial.print("B Unit -  ");
 Serial.print(cm);
 Serial.print("cm");
 Serial.println();
 return cm;
}

long HCSR04_A_Unit()
{
 long duration, cm;
 pinMode(trigPinA, OUTPUT);
 digitalWrite(trigPinA, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPinA, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPinA, LOW);
 pinMode(echoPinA, INPUT);
 duration = pulseIn(echoPinA, HIGH);
 cm = microsecondsToCentimeters(duration);
 Serial.print("A Unit -  ");
 Serial.print(cm);
 Serial.print("cm");
 Serial.println();
 return cm;
 //delay(1000);
```

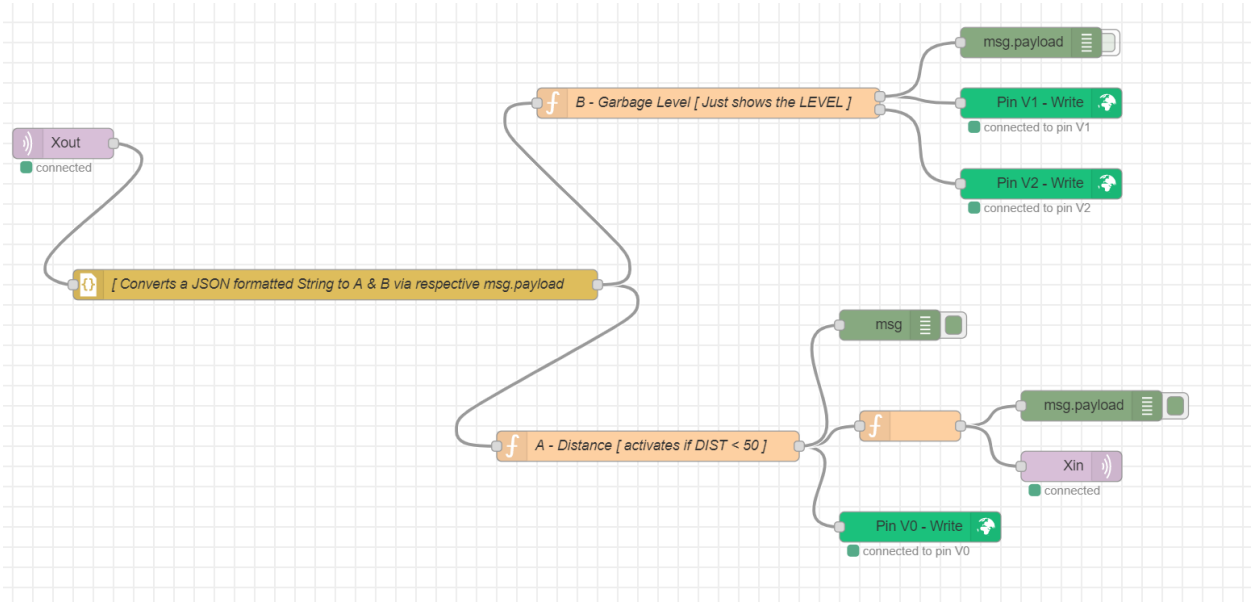
```
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}

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 callback(char* topic, byte* payload, unsigned int length) {
 Serial.print("Message arrived [");
 Serial.print(topic);
 Serial.print("] ");
 for (int i = 0; i < length; i++) {
```

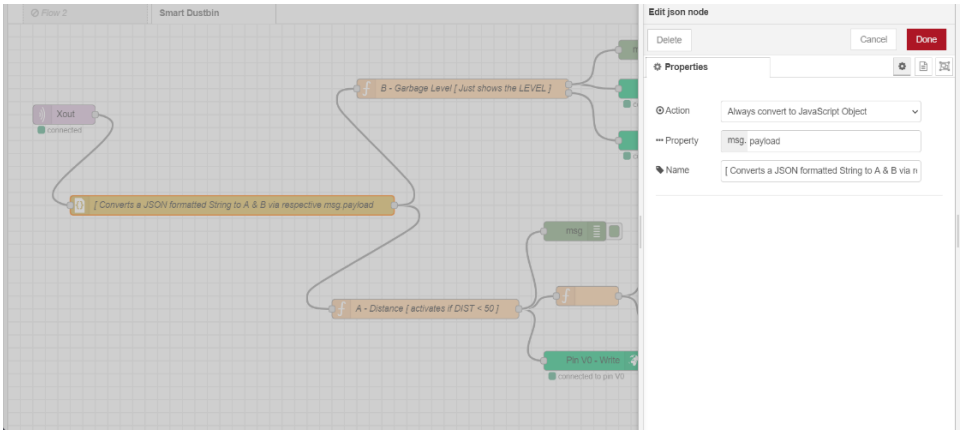
```
  Serial.print((char)payload[i]);
 }
 Serial.println();

 // Switch on the Sevo Motor if an 1 was received as first character
 if ((char)payload[0] == '1')
 {
  digitalWrite(D7,HIGH);   // Turn the Servo on
  //myservo.write(180);      // tell servo to go to position in variable 'pos'
  //delay(30);
 }
 else
 {
  digitalWrite(D7,LOW);  // Turn the Servo off
  //myservo.write(0);      // tell servo to go to position in variable 'pos'
  //delay(30);
 }

}

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("Xin", "esp8266 is Online");
   // ... and resubscribe
   client.subscribe("Xin");
```

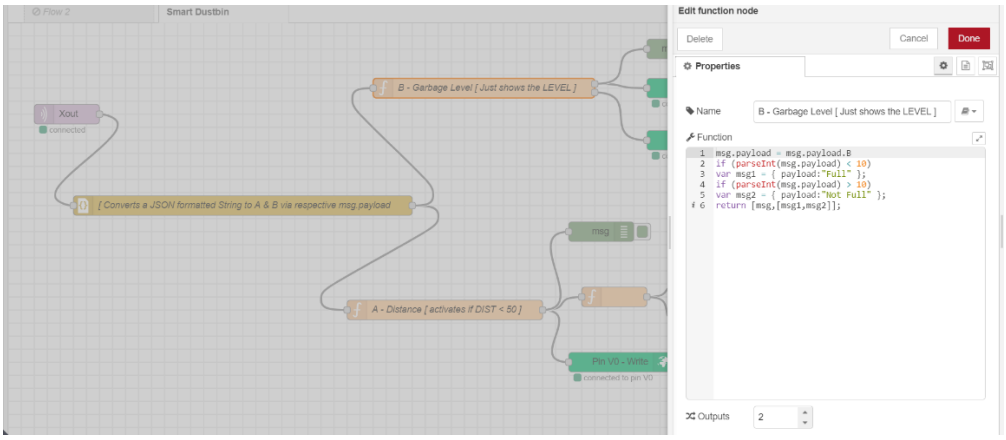
```
    }
    else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup() {

  Serial.begin(9600);
  pinMode(D7,OUTPUT);
  digitalWrite(D7,LOW);
  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 > 5000) {
    lastMsg = now;
    ++value;
    String data = "";
```

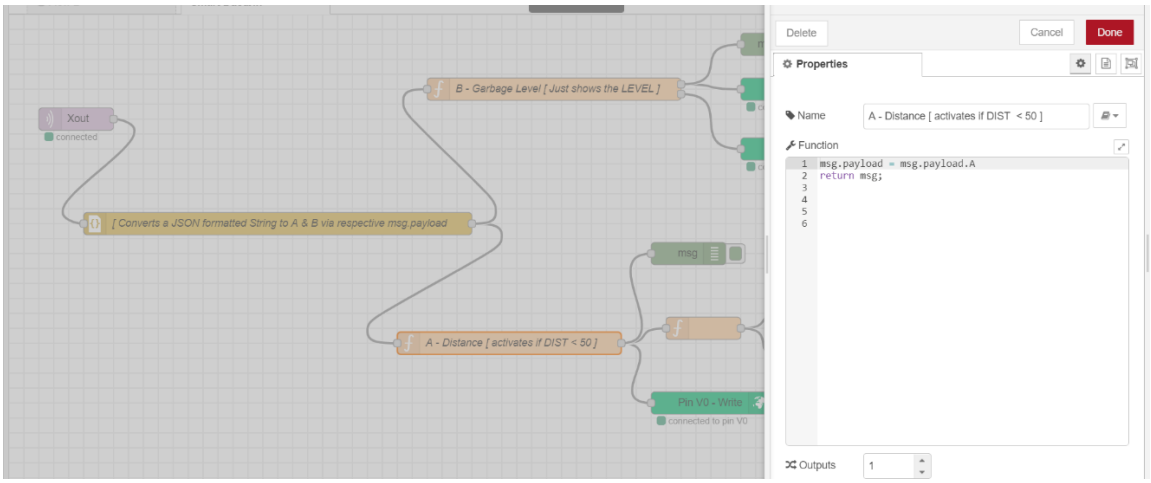
```
  long x = HCSR04_A_Unit();
  long y = HCSR04_B_Unit();



  data += "{";
  data += "\"";
  data += "A";
  data += "\"";
  data += ":";
  data += String(x);
  data += ",";
  data += "\"";
  data += "B";
  data += "\"";
  data += ":";
  data += String(y);
  data += "}";

  data.toCharArray(msg,100);
  Serial.print("Publish message >> ");
  Serial.println(msg);
  client.publish("Xout", msg);
 }
}
```

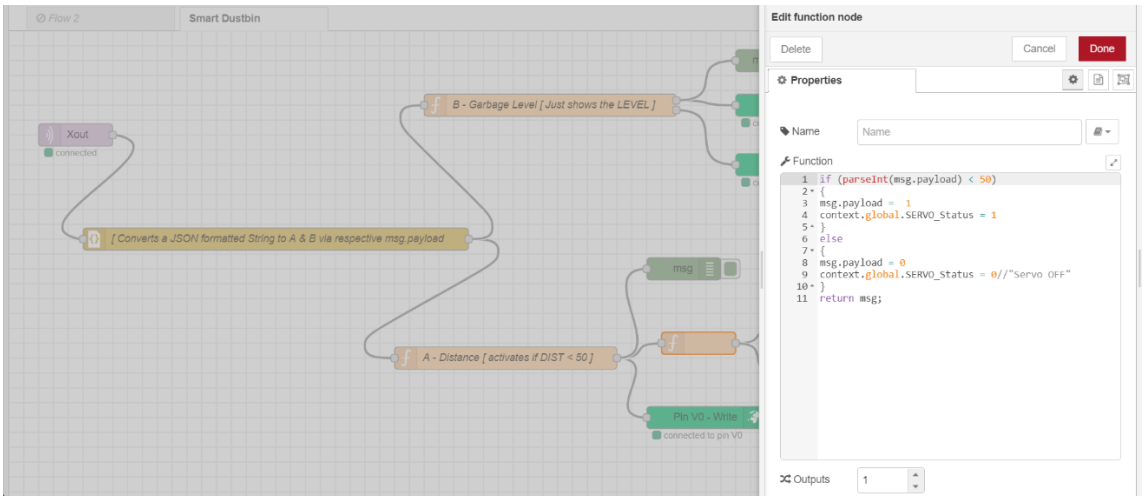
NodeRED nodes

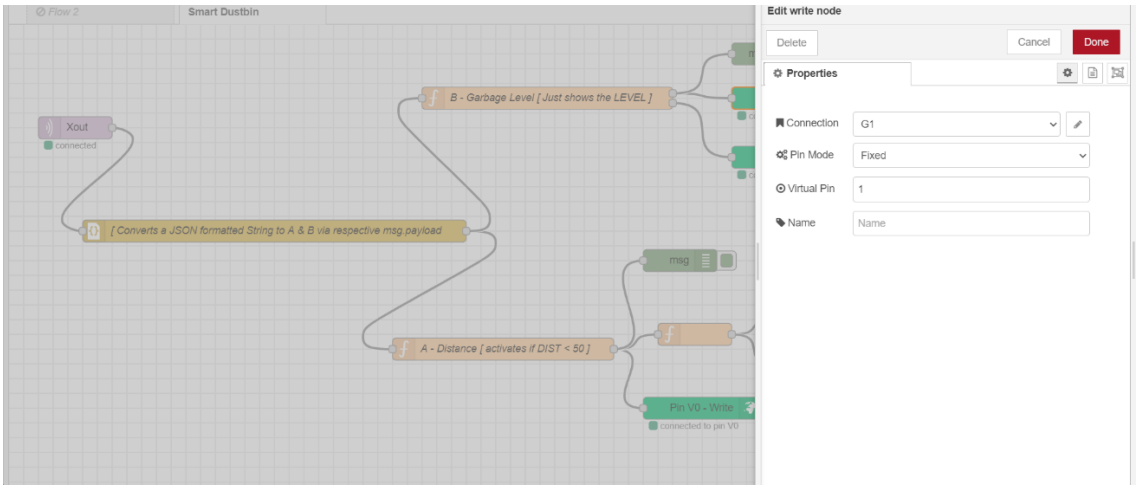












Do the same for other Blynk Pins – Use V0,V1,V2

Upgrades

1. Getting power from a power outlet
2. Adding a LCD to show the garbage level
3. Creating a buzzer system to detect system errors