**LOAD CELL CODE:**

#include "HX711.h"

#define DOUT 3

#define CLK 2

HX711 scale(DOUT, CLK);

//float calibration\_factor = -96650; //==============================SETUP===================================

void setup() {

Serial.begin(9600);

Serial.println("Press T to tare");

scale.set\_scale(-96650); //Calibration Factor obtained from sketch

scale.tare(); //Reset the scale to 0

}

//==============================LOOP===================================

void loop() {

Serial.print("Weight: ");

Serial.print(scale.get\_units(), 3); //Up to 3 decimal points

Serial.println(" kg");

if(Serial.available())

{

char temp = Serial.read();

if(temp == 't' || temp == 'T')

scale.tare(); //Reset the scale to zero

}

}

**SERVER CODE:**

//-- Libraries Included --------------------------------------------------------------

#include <ESP8266WiFi.h>

//------------------------------------------------------------------------------------

// Define I/O Pins

#define LED0 2 // WIFI Module LED

#define LED1 D5 // Connectivity With Client #1

#define LED2 D8 // Connectivity With Client #2

#define BUTTON D2 // Connectivity ReInitiate Button

//------------------------------------------------------------------------------------

// Authentication Variables

char\* TKDssid; // SERVER WIFI NAME

char\* TKDpassword; // SERVER PASSWORD

//------------------------------------------------------------------------------------

#define MAXSC 4 // MAXIMUM NUMBER OF CLIENTS

WiFiServer TKDServer(9001); // THE SERVER AND THE PORT NUMBER

WiFiClient TKDClient[MAXSC]; // THE SERVER CLIENTS

// defines pins numbers

#define trigPin D6 // WIFI Module LED sensor 1

#define echoPin D7

#define trigPin1 D0 // WIFI Module LED sensor 2

#define echoPin1 D1

// defines variables

long duration;

int distance;

long duration1;

int dist1;

void setup()

{

// Setting The Serial Port

Serial.begin(9600); // Computer Communication

SetWifi("TAKEONE", "");

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

pinMode(trigPin1, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin1, INPUT); // Sets the echoPin as an Input

// Setting The Mode Of Pins

pinMode(LED0, OUTPUT); // WIFI OnBoard LED Light

pinMode(LED1, OUTPUT); // Indicator For Client #1 Connectivity

pinMode(LED2, OUTPUT); // Indicator For Client #2 Connectivity

pinMode(BUTTON, INPUT\_PULLUP); // Initiate Connectivity

// Print Message Of I/O Setting Progress

Serial.println();

Serial.println("I/O Pins Modes Set .... Done");

}

void loop()

{

//Loop for Sensor 1

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance= duration\*0.034/2;

//Loop for Sensor 2

// Clears the trigPin

digitalWrite(trigPin1, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin1, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin1, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration1 = pulseIn(echoPin1, HIGH);

// Calculating the distance

dist1= duration1\*0.034/2;

IsClients(distance,dist1);

}

//Setting WiFI Access Point

void SetWifi(char\* Name, char\* Password)

{

// Stop Any Previous WIFI

WiFi.disconnect();

// Setting The Wifi Mode

WiFi.mode(WIFI\_AP\_STA);

Serial.println("WIFI Mode : AccessPoint Station");

// Setting The Access Point

TKDssid = Name;

TKDpassword = Password;

// Starting The Access Point

WiFi.softAP(TKDssid, TKDpassword);

Serial.println("WIFI < " + String(TKDssid) + " > ... Started");

// Wait For Few Seconds

delay(1000);

// Getting Server IP

IPAddress IP = WiFi.softAPIP();

// Printing The Server IP Address

Serial.print("AccessPoint IP : ");

Serial.println(IP);

// Starting Server

TKDServer.begin();

Serial.println("Server Started");

}

**FIRST BOT CODE:**

//------------------------------------------------------------------------------------

#include <ESP8266WiFi.h>

//------------------------------------------------------------------------------------

// Defining I/O Pins

//------------------------------------------------------------------------------------

#define LED0 2 // WIFI Module LED

#define LED1 D5 // Connectivity With Client #1

#define LED2 D8 // Connectivity With Client #2

#define BUTTON D2 // Connectivity Re-Initiate Button

#define MOT1 D0

#define MOT12 D1

#define MOT2 D3

#define MOT21 D5

//------------------------------------------------------------------------------------

// Authentication Variables

//------------------------------------------------------------------------------------

char\* TKDssid;

char\* TKDpassword;

IPAddress TKDServer(192, 168, 4, 1);

WiFiClient TKDClient;

// defines pins numbers

#define trigPin D6 // WIFI Module LED

#define echoPin D7

// defines variables

long duration;

int distance;

int k=1;

int j=1;

int w=1;

void setup()

{

// Setting The Serial Port -----------------------------------------

Serial.begin(9600); // Computer Communication

digitalWrite(MOT1, LOW);

digitalWrite(MOT12, LOW);

digitalWrite(MOT2, LOW);

digitalWrite(MOT21, LOW);

pinMode(MOT1, OUTPUT);

pinMode(MOT12, OUTPUT);

pinMode(MOT2, OUTPUT);

pinMode(MOT21, OUTPUT);

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

// Setting The Mode Of Pins ------------------------------------------

pinMode(LED0, OUTPUT); // WIFI OnBoard LED Light

pinMode(LED1, OUTPUT); // Indicator For Client #1 Connectivity

pinMode(LED2, OUTPUT); // Indicator For Client #2 Connectivity

pinMode(BUTTON, INPUT\_PULLUP); // Initiate Connectivity

digitalWrite(LED0, !LOW); // Turn WiFi LED Off

// Print Message Of I/O Setting Progress --------------------------------

Serial.println("\nI/O Pins Modes Set .... Done");

// Starting To Connect --------------------------------------------------

WiFi.mode(WIFI\_STA); // To Avoid Broadcasting An SSID

WiFi.begin("TAKEONE"); // The SSID That We Want To Connect

// Printing Message for User That Connetion Is On Process ----------

Serial.println("!--- Connecting To " + WiFi.SSID() + " ---!");

// WiFi Connectivity -----------------------------------------------

// Checking For Connection

CheckConnectivity();

}

void loop()

{

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance= duration\*0.034/2;

if (distance <= 25 && j==1)

{

w=2;

digitalWrite(MOT1, HIGH);

digitalWrite(MOT2, HIGH);

}

if(w==2)

{

sendToServer("on");

}

if(w==1)

{

sendToServer("ok");

}

}

//Function for Checking connectivity

void CheckConnectivity()

{

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

{

for (int i = 0; i < 10; i++)

{

digitalWrite(LED0, !HIGH);

delay(250);

digitalWrite(LED0, !LOW);

delay(250);

Serial.print(".");

}

Serial.println("");

}

// Stop Blinking To Indicate Connected -----------------------------

digitalWrite(LED0, !HIGH);

Serial.println("!—Node.[i] Connected --!");

// Conecting The Device As A Client --------------------------------

void TKDRequest()

{

// First Make Sure You Got Disconnected

TKDClient.stop();

// If Sucessfully Connected Send Connection Message

if (TKDClient.connect(TKDServer, 9001))

{

Serial.println ("<" + nodeIdentifier + "-CONNECTED>");

TKDClient.println ("<" + nodeIdentifier + "-CONNECTED>");

}

}

**OTHER BOT CODE:**

//------------------------------------------------------------------------------------

#include <ESP8266WiFi.h>

//------------------------------------------------------------------------------------

// Defining I/O Pins

//------------------------------------------------------------------------------------

#define LED0 2 // WIFI Module LED

#define LED1 D5 // Connectivity With Client #1

#define LED2 D8 // Connectivity With Client #2

#define BUTTON D2 // Connectivity ReInitiate Button

#define MOT1 D0

#define MOT12 D1

#define MOT2 D3

#define MOT21 D5

//------------------------------------------------------------------------------------

// Authentication Variables

//------------------------------------------------------------------------------------

char\* TKDssid;

char\* TKDpassword;

IPAddress TKDServer(192, 168, 4, 1);

WiFiClient TKDClient;

//====================================================================================

// defines variables

long duration;

int distance;

int j=3;

int k=1;

void setup()

{

// Setting The Serial Port ----------------------------------------------

Serial.begin(9600); // Computer Communication

digitalWrite(MOT1, LOW);

digitalWrite(MOT12, LOW);

digitalWrite(MOT2, LOW);

digitalWrite(MOT21, LOW);

pinMode(MOT1, OUTPUT);

pinMode(MOT12, OUTPUT);

pinMode(MOT2, OUTPUT);

pinMode(MOT21, OUTPUT);

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

// Setting The Mode Of Pins ---------------------------------------------

pinMode(LED0, OUTPUT); // WIFI OnBoard LED Light

pinMode(LED1, OUTPUT); // Indicator For Client #1 Connectivity

pinMode(LED2, OUTPUT); // Indicator For Client #2 Connectivity

pinMode(BUTTON, INPUT\_PULLUP); // Initiate Connectivity

digitalWrite(LED0, !LOW); // Turn WiFi LED Off

// Starting To Connect --------------------------------------------------

WiFi.mode(WIFI\_STA); // To Avoid Broadcasting An SSID

WiFi.begin("TAKEONE"); // The SSID That We Want To Connect

// Printing Message For User That Connetion Is On Process ---------------

Serial.println("!--- Connecting To " + WiFi.SSID() + " ---!");

// WiFi Connectivity ----------------------------------------------------

CheckConnectivity();

//Function for Checking connectivity

void CheckConnectivity()

{

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

{

for (int i = 0; i < 10; i++)

{

digitalWrite(LED0, !HIGH);

delay(250);

digitalWrite(LED0, !LOW);

delay(250);

Serial.print(".");

}

Serial.println("");

}

// Stop Blinking To Indicate Connected -----------------------------

digitalWrite(LED0, !HIGH);

Serial.println("!—Node.[i] Connected --!");

// Conecting The Device As A Client --------------------------------

void TKDRequest()

{

// First Make Sure You Got Disconnected

TKDClient.stop();

// If Sucessfully Connected Send Connection Message

if (TKDClient.connect(TKDServer, 9001))

{

Serial.println ("<" + nodeIdentifier + "-CONNECTED>");

TKDClient.println ("<" + nodeIdentifier + "-CONNECTED>");

}

}