#include <SoftwareSerial.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include <Wire.h>

SoftwareSerial esp8266(2, 3); // RX is pin 2, TX Arduino line is pin 3.

#define DEBUG true

#define ONE\_WIRE\_BUS 8 // \*\*\*changed to 8 from 12 for rev2 stepper

OneWire oneWire(ONE\_WIRE\_BUS);

DallasTemperature sensors(&oneWire);

int ventType = 2; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*change this for vent type 1 for servo, 2 for stepper motor\*\*\*\*\*\*\*\*\*\*

String networkName = "traitors";//"STuPiDiNTeRNeT"; //"traitors";

String networkPass = "12345678";//"DELAPPPW"; // "12345678";

int vent2Position = 9; //assume that vent is closed (9)

//stepper motor funct info

int Speed = 0; // This controls how long we wait between pulses (In seconds)

int Distance = 0; // Creating an integer variable named "Distance" that counts the number of steps we've made

int FullRotation = 4096; // How many steps are in a full revolution of the motor shaft?

int Rotation = 1024;//FullRotation/4;

int lookup[8] = {B01000, B01100, B00100, B00110, B00010, B00011, B00001, B01001};

// This array contains bimary values that will be pushed to the outpur pins to properly pulse the stepper

//outputs for the stepper motor

//inverse for v2 stepper

int IN1 = 7;

int IN2 = 6;

int IN3 = 5;

int IN4 = 4;

void setup()

{

Serial.begin(9600);

esp8266.begin(9600);

sensors.begin();

delay(1500);

// reset wireless

sendData("AT+RST\r\n", 5000, DEBUG); // rst

delay(3000);

//sendData("AT+CWMODE=1\r\n", 1000, DEBUG); // set wifi in station mode

//sendData("AT+CIPMUX=0\r\n", 1000, DEBUG); // configure for single connections

//open connection

sendData("AT+CWJAP=\"" + networkName + "\",\"" + networkPass + "\"\r\n", 5000, DEBUG); // join the access point specified above

delay(3000);

if (ventType == 2) {

// set pin mode for stepper motor

pinMode(IN1, OUTPUT);

pinMode(IN2, OUTPUT);

pinMode(IN3, OUTPUT);

pinMode(IN4, OUTPUT);

//park vent to closed position, so that it can always start at the same position

parkVent();

vent2Position = 9;

}

}

void loop()

{

//get temperature

sensors.requestTemperatures();

float temp = sensors.getTempCByIndex(0);

temp = temp \* 9 / 5 + 32; // convert temp to farenheit

delay(300);

//int connectionId = esp8266.read() - 48; // subtract 48 because the read() function returns

// the ASCII decimal value and 0 (the first decimal number) starts at 48

// '0' - 48 = 0

// '1' - 48 = 1

//concatenate server request string

String webpage = "GET /vent2.php?vent2=";

webpage += temp;

webpage += " HTTP:/1.1\r\nHost: 52.41.237.167\r\n\r\n";

//starts connection as client

sendData("AT+CIPSTART=\"TCP\",\"ec2-52-41-237-167.us-west-2.compute.amazonaws.com\",80\r\n", 1000, DEBUG);

Serial.print("temp: \n");

Serial.print(temp);

Serial.print("\n");

//concatenate cipsend string (length of data to be sent) over wifi

String cipSend = "AT+CIPSEND=";

cipSend += webpage.length();

cipSend += "\r\n";

//command to specify length of data to be sent via wifi

sendData(cipSend, 1000, DEBUG);

delay(500);

//send info to server page, then set reply as webpageResponse

String webpageResponse = sendData(webpage, 1000, DEBUG);

Serial.print(webpageResponse);

//parse webpageResponse to get the requested vent status

int eqIndex = webpageResponse.indexOf('=');

int eqIndex2 = webpageResponse.indexOf('=', eqIndex + 1);

String ventCommand = webpageResponse.substring(eqIndex2 + 1, eqIndex2 + 2);

// Serial.print(ventCommand + "/n"); //prints the requested vent command

delay(500);

//convert parsed string number to an integer

char cmnd[1];

strcpy(cmnd, ventCommand.c\_str());

int ventCommandI = atoi(cmnd);

//if vent position is more closed than requested, open vent

while (vent2Position > ventCommandI) {

if (ventCommandI < 0) {

// this if is for error checking to avoid inf loop if webdata returns wrong

Serial.print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Error. command is neg!\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

} else {

OPEN();

vent2Position--;

}

delay(150);

Serial.print("\n ventPos");

Serial.print(vent2Position);

Serial.print(" ");

Serial.print(ventCommandI);

}

//if vent position is more open than requested, close vent

while (vent2Position < ventCommandI) {

if (ventCommandI > 9) {

// this if is for error checking to avoid inf loop if webdata returns wrong

Serial.print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Error. command is high!\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

break;

}else{

CLOSE();

vent2Position++;

}

delay(150);

Serial.print("\n ventPos");

Serial.print(vent2Position);

Serial.print(" ");

Serial.print(ventCommandI);

}

delay(2000);

}

//function to park stepper vent

void parkVent() {

for (int j = 128; j > 0; j--) {

for (int stepper = 0; stepper < 8; stepper++)

{

MoveStep(stepper);

delay(Speed);

}

}

Serial.print("parked");

}

//function to open stepper vent 1/9th

void OPEN() {

for (int i = 0; i < 16; i++) {

for (int stepper = 7; stepper >= 0; stepper--) {

MoveStep(stepper);

delay(Speed);

}

}

}

//function to close stepper vent 1/9th

void CLOSE() {

for (int j = 16; j > 0; j--) {

for (int stepper = 0; stepper < 8; stepper++)

{

MoveStep(stepper);

delay(Speed);

}

}

}

// This function will increment the stepper motor one step in either direction (depending on which function called this function)

void MoveStep(int eighth)

{

digitalWrite(IN1, bitRead(lookup[eighth], 0));

digitalWrite(IN2, bitRead(lookup[eighth], 1));

digitalWrite(IN3, bitRead(lookup[eighth], 2));

digitalWrite(IN4, bitRead(lookup[eighth], 3));

Serial.print(bitRead(lookup[eighth], 0));

Serial.print(bitRead(lookup[eighth], 1));

Serial.print(bitRead(lookup[eighth], 2));

Serial.print(bitRead(lookup[eighth], 3));

Serial.print("\t");

}

//function for send and receive data

String sendData(String command, const int timeout, boolean debug)

{

//send data, then recieve response and concatenate characters into a string,

//then output to the serial monitor

String response = "";

esp8266.print(command); // send the read character to the esp8266

long int time = millis();

while ( (time + timeout) > millis())

{

while (esp8266.available())

{

// The esp has data so display its output to the serial window

char c = esp8266.read(); // read the next character.

response += c;

}

}

if (debug)

{

Serial.print(response);

}

return response;

}