**INTELLIGENT CARGO MANAGEMENT SYSTEM**

P.Rashmitha

M.Samhitha

D.Snehitha

P.Spandana

**ABSTRACT**

Internet of Things (IOT) conceptualizes the idea of remotely connecting and monitoring real world objects (things) through the Internet.  When it comes to cargo management, this concept can be aptly incorporated to make it smarter, safer and automated.The evolution of multipurpose sensors over the last decades has been investigated with the aim of developing innovative devices with applications in several fields of technology. The integration of such sensors in cargo has paved the way for intelligent transportation. These integrated systems are capable of providing reliable information about the quality of the goods during their storage period. To accomplish this goal, we use a variety of sensors suited for monitoring the quality and safety of food products and goods by recording the evolution of parameters like temperature, humidity and air quality, fire detection. This information is transmitted wirelessly to the IBM cloud providing an interface where the authorized person can observe the product quality over time and also receives alert messages in case of bad air quality and abnormal conditions.

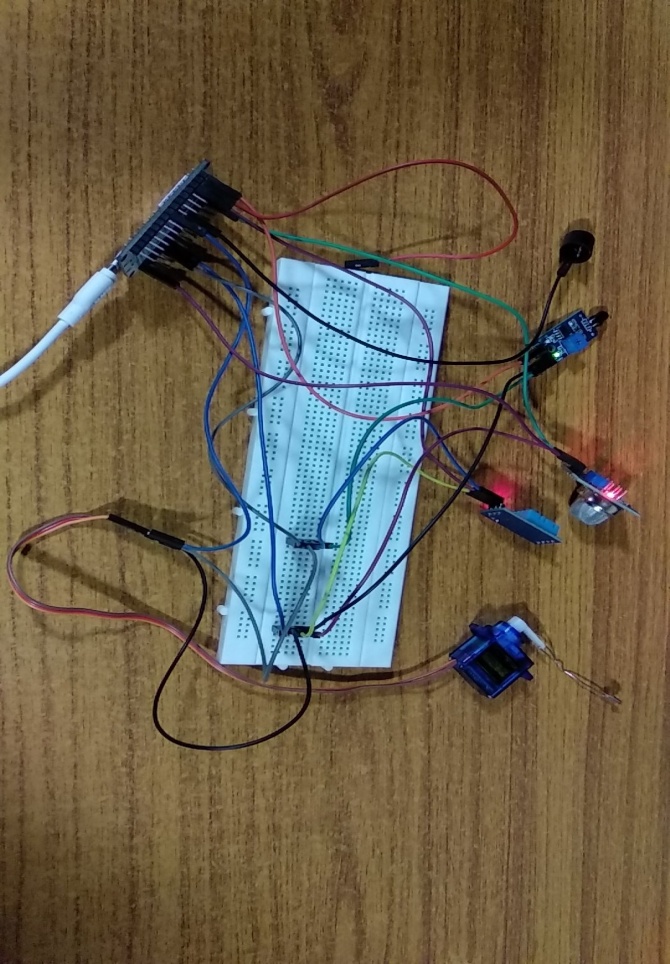
**HARDWARE SOFTWARE**

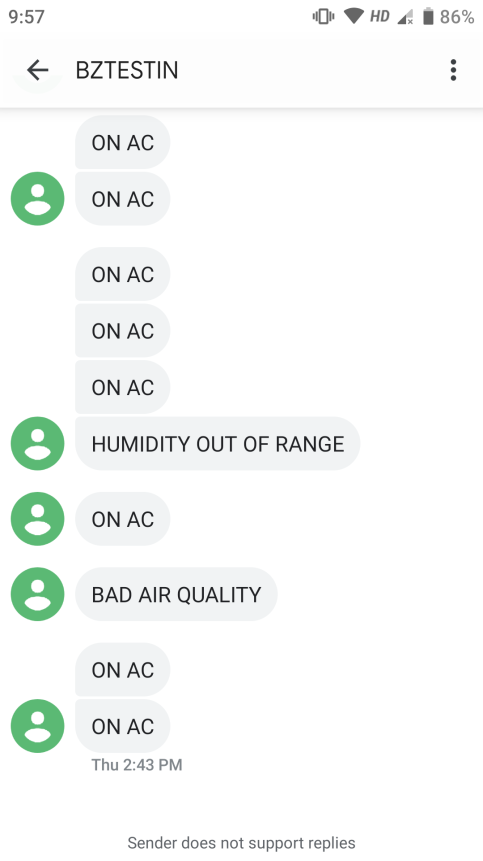
* NODEMCU 1**.** ARDUINO IDE
* FLAME SENSOR 2.ANDROID(MIT APP INVENTOR)
* MQ135 3.IBM CLOUD
* DHT11
* SERVO MOTOR
* BUZZER

**PROBLEM STATEMENT**

Supply Chain Management consists of a number of phases in which the transportation plays an important role. The perishable products are to be delivered at the desirable quality in the right time. The lack of constant monitoring of the product during the freight contributes towards the food wastage during the transportation phase. The various factors that influence the freshness scale of the product which includes the natural and physical conditions are to be considered before the movement of the goods from the source to the destination.

**PROJECT WORKING PROCESS**

****The sensors are deployed inside the truck to monitor the freshness of the product. Various parameters like temperature, humidity and the air quality inside the truck are monitored. An app is designed for selecting the type of goods to be stored in the truck, for door automation and retrieving the values from the sensor. Initially the authorized person of the truck has to select the type of goods stored in the truck through the app. Based on the selection, the temperature and humidity values at which the goods are supposed to be stored will be known. Using the DHT11 sensor, the actual temperature and humidity inside the truck can be known. The temperature and humidity values can be obtained from the app .By comparing these values, a message will be sent to the authorized person asking him/her to turn on the AC or the heater in any abnormal conditions. The air quality inside the truck is constantly monitored using MQ135 sensor. If the quality of the air inside the truck is bad, a message is sent to the authorized person regarding the same .Through this goods can be maintained in constant conditions and their quality is maintained for longer time. Fire sensor is also deployed in the truck to detect fire. If fire is detected, the buzzer rings alerting the driver of the fire. The door of the truck is automated using a servo motor. The authorized person of the truck can open or close the door of the truck using the app.

****

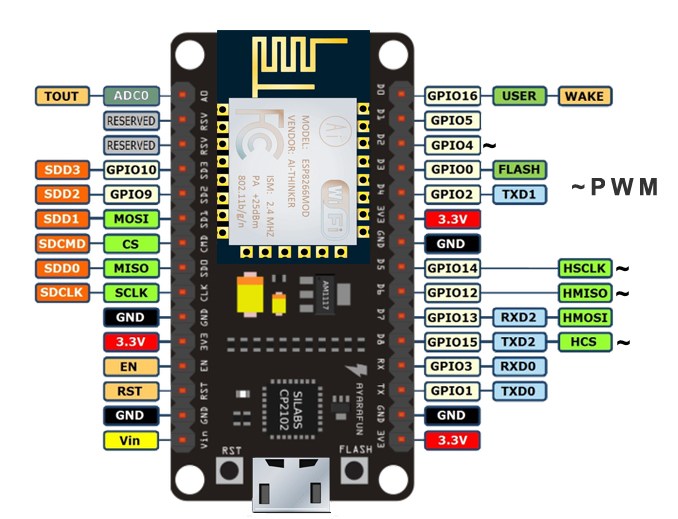
**COMPONENTS USED**

**NODEMCU**

Node MCU development board is an open-source IOT development kit .It includes firmware which runs on the ESP8266Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12E module. The term "NodeMCU" by default refers to the firmware rather than the dev kits. The firmware uses the Lua scripting language .It is a low cost hardware platform available for development of IOT applications.

**SPECIFICATIONS**

* ESP8266 is high integration wireless SOC.
* ESP-12E Wi-Fi module is developed by Ai-thinker Team.
* Integrated low power 32-bit MCU -TensilicaL106, with 4MB Flash memory, 128KB SRAM.
* Clock speed support 80 MHz and over clock 160 MHz, Integrated 10-bit ADC and 13 GPIO’s.
* Integrated TCP/IP protocol stack.
* Supports IEEE 802.11 b/g/n, Wi-Fi 2.4GHz, WPA/WPA2,+20dBm output power.
* Support STA/AP/STA+AP operation modes.
* Integrated UART, I2C, I2S, IRDA, PWM, GPIO, SDIO 2.0, (H) SPI interface.
* Deep sleep power <10uA, Standby power consumption less than 1.0mW Supports the RTOS (Real-Time Operating System)

**SERVO MOTOR**

Servo implies an error sensing feedback control which is utilized to correct the performance of a system. A **servo motor** is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. A servo motor consists of three wires- one connected to ground, one to control unit and the other to power supply.

The function of the servo motor is to receive a control signal that represents a desired output position of the servo shaft and apply power to its DC motor until its shaft turns to that position. It consists of a suitable motor coupled to a sensor for position feedback.

The servo motors usually have a revolution cut off from 90° to 180°.IWe can get a very high torque servo motor in a small and light weight packages. Due to these features they are being used in many applications like toy car, RC helicopters and planes, Robotics, Machine etc.

**WORKING PRINCIPLE**

Servo motor works on **PWM (Pulse width modulation)** principle, means its angle of rotation is controlled by the duration of applied pulse to its Control PIN. Basically servo motor is made up of **DC motor which is controlled by a variable resistor (potentiometer) and some gears**. High speed force of DC motor is converted into torque by Gears. We know that WORK= FORCE X DISTANCE, in DC motor Force is less and distance (speed) is high and in Servo, force is High and distance is less. Potentiometer is connected to the output shaft of the Servo, to calculate the angle and stop the DC motor on required angle.

**SPECIFICATIONS**

* Operating Voltage is +5V typically
* Torque: 2.5kg/cm
* Operating speed is 0.1s/60°
* Gear Type: Plastic
* Rotation : 0°-180°

**MQ135**

Sensitive material MQ135 gas sensor is used in clean air low conductivity tin oxide (SnO2). When the sensor is kept in their environment, in the polluting gases, the conductivity of the sensor increases with increasing air pollution. Using a simple circuit can change the conductivity.

 MQ135 is an air quality sensor for detecting a wide range of gases, including NH3, NO2, alcohol, benzene, smoke and CO2. It is ideal for use in office or factory, simply drive and monitoring circuit.

**WORKING PRINCIPLE**

The MQ-135 alcohol sensor consists of a tin dioxide (SnO2), a perspective layer inside aluminium oxide micro tubes (measuring electrodes) and a heating element inside a tubular casing. The end face of the sensor is enclosed by a stainless steel net and the back side holds the connection terminals. Ethyl alcohol present in the breath is oxidized into acetic acid passing through the heat element. With the ethyl alcohol cascade on the tin dioxide sensing layer, the resistance decreases. By using the external load resistance the resistance variation is converted into a suitable voltage variation.

**SPECIFICATION**

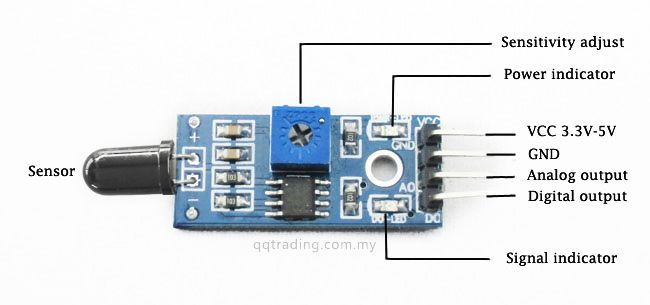
* Has a long life and reliable stability.
* Rapid response and recovery characteristics.
* Wide detecting scope.
* Fast response and High sensitive.
* Simple drive circuit.
* Stable and Long Life. Detection Range: 10 - 300 ppm NH3, 10 - 1000 ppm Benzene, 10 - 300 Alcohol.
* Heater Voltage: 5.0V.
* Dimensions: 18mm Diameter, 17mm High excluding pins, Pins - 6mm High.

**FLAME SENSOR**

A **flame detector** is a [sensor](https://en.wikipedia.org/wiki/Sensor) designed to detect and respond to the presence of a [flame](https://en.wikipedia.org/wiki/Flame) or [fire](https://en.wikipedia.org/wiki/Fire), allowing **flame detection**. Responses to a detected flame depend on the installation, but can include sounding an alarm, deactivating a fuel line (such as a [propane](https://en.wikipedia.org/wiki/Propane) or a [natural gas](https://en.wikipedia.org/wiki/Natural_gas) line), and activating a fire suppression system. When used in applications such as industrial furnaces, their role is to provide confirmation that the furnace is working properly; in these cases they take no direct action beyond notifying the operator or control system .There are different types of flame detection methods. Some of them are: Ultraviolet detector, near IR array detector, infrared (IR) detector, Infrared thermal cameras, UV/IR detector etc.

**WORKING**

When fire burns it emits a small amount of Infra-red light, this light will be received by the Photodiode (IR receiver) on the sensor module. **IR based flame sensor**. It is based on the YG1006 sensor which is a high speed and high sensitive NPN silicon phototransistor. It can detect infrared light with a wavelength ranging from 700nm to 1000nm and its detection angle is about 60°.  Flame sensor module consists of a photodiode (IR receiver), resistor, capacitor, potentiometer, and LM393 comparator in an integrated circuit. The sensitivity can be adjusted by varying the on board potentiometer. Working voltage is between 3.3v and 5v DC, with a digital output. Logic high on the output indicates presence of flame or fire. Logic low on output indicates absence of flame or fire.

**SPECIFICATIONS**

* LM393 comparator chip
* Detection Range: 760 nm to 1100 nm
* Operating Voltage: 3.3 V to 5 V
* Maximum Output Current: 15 mA
* Digital Outputs: 0 and 1
* Detection Angle: about 60 degrees
* Adjustable sensitivity via potentiometer
* LED lights indicators: power (red) and digital switching output (green)
* PCB Size: 32 x 14 mm

**DHT11**

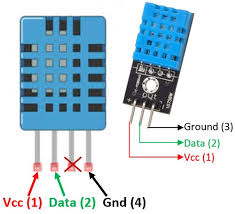
This DHT11 Temperature and Humidity Sensor features a calibrated digital signal output with the temperature and humidity sensor capability. It is integrated  with a high-performance 8-bit microcontroller. Its technology ensures the high reliability and excellent long-term stability.  This sensor includes a resistive element and a sensor for wet NTC temperature measuring devices. It has excellent quality, fast response, anti-interference ability and high performance.

**WORKING**

Each DHT11 sensors features extremely accurate calibration of humidity calibration chamber. The calibration coefficients stored in the OTP program memory, internal sensors detect signals in the process, we should call these calibration coefficients. The single-wire serial interface system is integrated to become quick and easy. Small size, low power, signal transmission distance up to 20 meters, enabling a variety of applications and even the most demanding ones. The product is 4-pin single row pin package. Convenient connection, special packages can be provided according to users need.

**SPECIFICATION**

* Supply Voltage: +5 V
* Temperature range :0-50 °C error of ± 2 °C
* Humidity :20-90% RH ± 5% RH error
* Interface: Digital

****

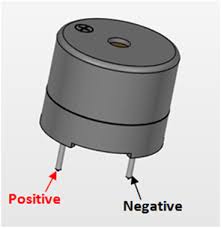
**BUZZER**

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include [alarm devices](https://en.wikipedia.org/wiki/Alarm_devices), [timers](https://en.wikipedia.org/wiki/Timer), and confirmation of user input such as a mouse click or keystroke.

Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones ,timers and other electronic products for sound devices.

**SPECIFICATION**

* Rated Voltage: 6V DC
* Operating Voltage: 4-8V DC
* Rated current: <30mA
* Sound Type: Continuous Beep
* Resonant Frequency: ~2300 Hz
* Small and neat sealed package



**PROGRAM CODE**

#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include <Servo.h>

#include "DHT.h"

const char\* ssid = "honor11";

const char\* password = "spandana";

const char\* host = "api.msg91.com";

#define DHTPIN D2 // what pin we're connected to

#define DHTTYPE DHT11 // define type of sensor DHT 11

#define ORG "668qh1"

#define DEVICE\_TYPE "project1"

#define DEVICE\_ID "1209"

#define TOKEN "123456789"

String command1,x;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char topic[] = "iot-2/cmd/home/fmt/String";

char topic1[] = "iot-2/evt/Data/fmt/json";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

float h,t;

int mintemp=0,maxtemp=100,minhumid=0,maxhumid=100;

int flag1a=2,flag1b=2,flag1c=2,flag1d=2;

int flag2a=2,flag2b=2,flag2c=2,flag2d=2;

int flag3a=2,flag3b=2,flag3c=2,flag3d=2,flag=0;

int gasLevel=0;

void callback(char\* topic, byte\* payload, unsigned int payloadLength);

WiFiClient wifiClient;

WiFiClient client1;

Servo myservo;

DHT dht (DHTPIN, DHTTYPE);

void wifiConnect();

void mqttConnect();

void initManagedDevice();

PubSubClient client(server, 1883, callback, wifiClient);

void PublishData(float temp, float humid);

void setup()

{

Serial.begin(115200);

Serial.println();

dht.begin();

pinMode(A0,INPUT); //mq135

pinMode(D1,OUTPUT); //buzzer positive

pinMode(D0,INPUT); //fire detector A0 connection

myservo.attach(D5); //motor

myservo.write(0);

wifiConnect();

mqttConnect();

}

void loop()

{

if (!client.loop())

{

mqttConnect();

}

h = dht.readHumidity();

t = dht.readTemperature();

int sensorvalue=digitalRead(D0);

Serial.println(sensorvalue);

delay(100);

if(sensorvalue==1)

{

digitalWrite(D1,LOW);

delay(1000);

}

else

{

digitalWrite(D1,HIGH);

delay(1000);

}

if (isnan(h) || isnan(t))

{

Serial.println("Failed to read from DHT sensor!");

delay(1000);

return;

}

PublishData(t,h);

delay(100);

Serial.print("connecting to ");

Serial.println(host);

const int httpPort = 80;

if (!client1.connect(host, httpPort))

{

Serial.println("connection failed");

return;

}

if(t>maxtemp)

{

if(flag2a==0 || flag2b==0 ||flag2c==0 ||flag2d==0 )

{

String url = "/api/sendhttp.php?mobiles=9502704978&authkey=281702AsMBd6mB5d09bce1&route=4&sender=TESTIN&message=ON%20AC&country=91";

Serial.print("Requesting URL: ");

Serial.println(url);

client1.print(String("GET ") + url + " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"Connection: close\r\n\r\n");

delay(10);

while(client1.available())

{

String line = client1.readStringUntil('\r');

Serial.print(line);

}

if(x=="FRUITS")

{

flag2a=1;

flag1a=0;

flag1b=2;

flag2b=2;

flag1c=2;

flag2c=2;

flag2d=2;

flag1d=2;

}

if(x=="VEGETABLES")

{

flag2b=1;

flag1b=0;

flag1a=2;

flag2a=2;

flag1c=2;

flag2c=2;

flag2d=2;

flag1d=2;

}

if(x=="MEDICINES")

{

flag2c=1;

flag1c=0;

flag1b=2;

flag2b=2;

flag1a=2;

flag2a=2;

flag2d=2;

flag1d=2;

}

if(x=="GOODS")

{

flag2d=1;

flag1d=0;

flag1b=2;

flag2b=2;

flag1c=2;

flag2c=2;

flag2a=2;

flag1a=2;

}

Serial.println();

Serial.println("closing connection");

}

}

if(t<mintemp)

{

if(flag1a==0 || flag1b==0 ||flag1c==0 ||flag1d==0)

{

String url = "/api/sendhttp.php?mobiles=9502704978&authkey=281702AsMBd6mB5d09bce1&route=4&sender=TESTIN&message=ON%20HEATER&country=91";

Serial.print("Requesting URL: ");

Serial.println(url);

client1.print(String("GET ") + url + " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"Connection: close\r\n\r\n");

delay(10);

while(client1.available())

{

String line = client1.readStringUntil('\r');

Serial.print(line);

}

if(x=="FRUITS")

{

flag1a=1;

flag2a=0;

flag1b=2;

flag2b=2;

flag1c=2;

flag2c=2;

flag2d=2;

flag1d=2;

}

if(x=="VEGETABLES")

{

flag1b=1;

flag2b=0;

flag1a=2;

flag2a=2;

flag1c=2;

flag2c=2;

flag2d=2;

flag1d=2;

}

if(x=="MEDICINES")

{

flag1c=1;

flag2c=0;

flag1b=2;

flag2b=2;

flag1a=2;

flag2a=2;

flag2d=2;

flag1d=2;

}

if(x=="GOODS")

{

flag1d=1;

flag2d=0;

flag1b=2;

flag2b=2;

flag1c=2;

flag2c=2;

flag2a=2;

flag1a=2;

}

Serial.println();

Serial.println("closing connection");

}

}

if(h<minhumid || h>maxhumid)

{

if(flag3a==0 || flag3b==0 || flag3c==0 || flag3d==0)

{

String url = "/api/sendhttp.php?mobiles=9502704978&authkey=281702AsMBd6mB5d09bce1&route=4&sender=TESTIN&message=HUMIDITY%20OUT%20OF%20RANGE&country=91";

Serial.print("Requesting URL: ");

Serial.println(url);

client1.print(String("GET ") + url + " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"Connection: close\r\n\r\n");

delay(10);

while(client1.available())

{

String line = client1.readStringUntil('\r');

Serial.print(line);

}

if(x=="FRUITS")

{

flag3a=1;

flag3b=2;

flag3c=2;

flag3d=2;

}

if(x=="VEGETABLES")

{

flag3b=1;

flag3a=2;

flag3c=2;

flag3d=2;

}

if(x=="MEDICINES")

{

flag3c=1;

flag3a=2;

flag3b=2;

flag3d=2;

}

if(x=="GOODS")

{

flag3d=1;

flag3a=2;

flag3c=2;

flag3b=2;

}

Serial.println();

Serial.println("closing connection");

}

}

gasLevel=analogRead(A0);

Serial.println(gasLevel);

delay(500);

if(gasLevel<300)

{

Serial.println("good");

flag=0;

}

else if(gasLevel>300 && gasLevel<350)

{

Serial.println("Something is wrong");

flag=0;

}

else

{

Serial.println("Bad quality");

if(flag==0)

{

String url = "/api/sendhttp.php?mobiles=9502704978&authkey=281702AsMBd6mB5d09bce1&route=4&sender=TESTIN&message=BAD%20AIR%20QUALITY&country=91";

Serial.print("Requesting URL: ");

Serial.println(url);

client1.print(String("GET ") + url + " HTTP/1.1\r\n" +

"Host: " + host + "\r\n" +

"Connection: close\r\n\r\n");

delay(10);

while(client1.available())

{

String line = client1.readStringUntil('\r');

Serial.print(line);

}

flag=1;

Serial.println();

Serial.println("closing connection");}

}

}

void wifiConnect()

{

Serial.print("Connecting to ");

Serial.print(ssid);

WiFi.begin(ssid, password);

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

{

delay(500);

Serial.print(".");

}

Serial.print("nWiFi connected, IP address: ");

Serial.println(WiFi.localIP());

}

void mqttConnect()

{

if (!client.connected())

{

Serial.print("Reconnecting MQTT client to ");

Serial.println(server);

while (!client.connect(clientId, authMethod, token))

{

Serial.print(".");

delay(500);

}

initManagedDevice();

Serial.println();

}

}

void initManagedDevice()

{

if (client.subscribe(topic))

{

Serial.println("subscribe to cmd OK");

}

else

{

Serial.println("subscribe to cmd FAILED");//if command and format goes wrong

}

}

void callback(char\* topic, byte\* payload, unsigned int payloadLength)

{

Serial.print("callback invoked for topic: ");

Serial.println(topic);

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

{

command1 += (char)payload[i];

}

Serial.println(command1);

x=command1;

if(command1 == "OPEN")

{

myservo.write(90);

Serial.println("DOOR OPENED");

maxtemp=100;

mintemp=0;

minhumid=0;

maxhumid=100;

delay(100);

}

else if(command1 == "CLOSE")

{

myservo.write(0);

maxtemp=100;

mintemp=0;

minhumid=0;

maxhumid=100;

Serial.println("DOOR CLOSED");

delay(100);

}

else if(command1== "FRUITS")

{

Serial.println("fruit");

mintemp=10;

maxtemp=20;

minhumid=60;

maxhumid=100;

flag1a=0;

flag2a=0;

flag3a=0;

delay(100);

}

else if(command1=="VEGETABLES")

{

Serial.println("vegetable");

mintemp=15;

maxtemp=20;

minhumid=60;

maxhumid=80;

flag1b=0;

flag2b=0;

flag3b=0;

delay(100);

}

else if(command1=="MEDICINES")

{

Serial.println("med");

mintemp=1;

maxtemp=5;

minhumid=10;

maxhumid=60;

flag1c=0;

flag2c=0;

flag3c=0;

delay(100);

}

else if(command1=="GOODS")

{

Serial.println("goog");

mintemp=25;

maxtemp=35;

minhumid=60;

maxhumid=80;

flag1d=0;

flag2d=0;

flag3d=0;

delay(100);

}

command1="";

}

void PublishData(float temp, float humid)

{

if (!!!client.connected())

{

Serial.print("Reconnecting client to ");

Serial.println(server);

while (!!!client.connect(clientId, authMethod, token))

{

Serial.print(".");

delay(500);

}

Serial.println();

}

String payload = "{\"d\":{\"temperature\":";

payload += temp;

payload+="," "\"humidity\":";

payload += humid;

payload += "}}";

Serial.print("Sending payload: ");

Serial.println(payload);

if (client.publish(topic1, (char\*) payload.c\_str()))

{

Serial.println("Publish ok");

}

else

{

Serial.println("Publish failed");

}

}