FIRE DETECTION AND ALERTING SYSTEM

DANDA SRI SHREYA

B.Tech. ELECTRONICS AND COMMUNICATIONENGINEERING

ABSTRACT

Internet of Things (IoT) is basically a network which consists of physical systems that are embedded with sensors connected to a cloud where data is exchanged with the help of a gateway, which facilitates to and fro communication of data connected over the internet.

Fire detection systems are designed to discover fires early in the development when time will still be available for the safe evacuation of occupants. Early detection also plays a significant role in protecting of safety of emergency response personnel. Property loss can be reduced and downtime for the operation minimized through early detection because control efforts are started while the fire is still small.

At present the application of IoT based systems is extend to real time detection and warning system. However, cost has been a major factor for development and implementation of IoT systems. Considering the cost, ease of implementation, the proposed system proposes a low cost yet efficient IoT system for warning and alerting fire incidents.

The proposed system has sensors which gathers data from the physical world and is continuously sent to the cloud platform. When the temperature increases the preset threshold, the http webhook will be triggered and the notification is sent to the fire department. The amount of equipment to be taken by the fire departments can be moderated with the help of the displayed number of people in the affected area which has been added in the proposed system.

INTRODUCTION

OBJECTIVES AND GOALS

- Our main aim is to control the fire as soon as possible.
- The sensors (Infra-red and Temperature) will gather the data from physical word.
- The data gathered is sent to the cloud platform (ThingSpeak) continuously.
- When the temperature increases more than the preset threshold value, the http webhook will be triggered.
- A notification will be sent to the fire department and the number of people in the separate areas(rooms) in the affected area(building) will be displayed in the LCD.

APPLICATIONS

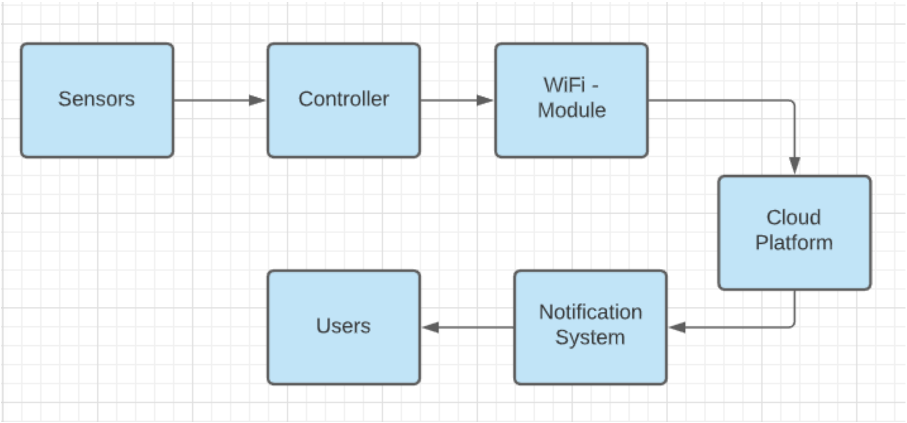
- Can be installed in every building **// ADD MOREAPPLICATIONS//**

FEATURES

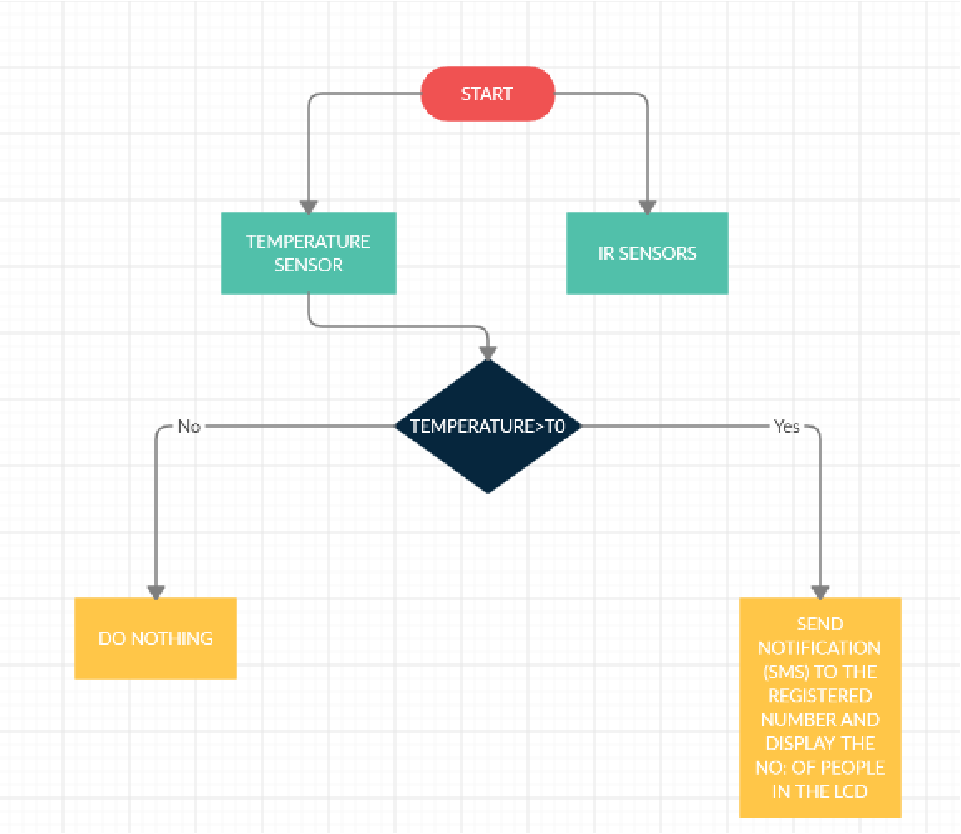
- Temperature sensor to sense the temperature of the surroundings.
- Arduino as the controller.
- ThingSpeak is used for the cloud storage.
- For triggering the messaging service, ifttt(If This Then That) is used.
- The efficient coding is done in the software tool called TinkerCad and Arduino IDE is used.
- WiFi – Module for gateway connection.
- The number of people is displayed on the LCD.

1. DESIGN

BLOCK DIAGRAM

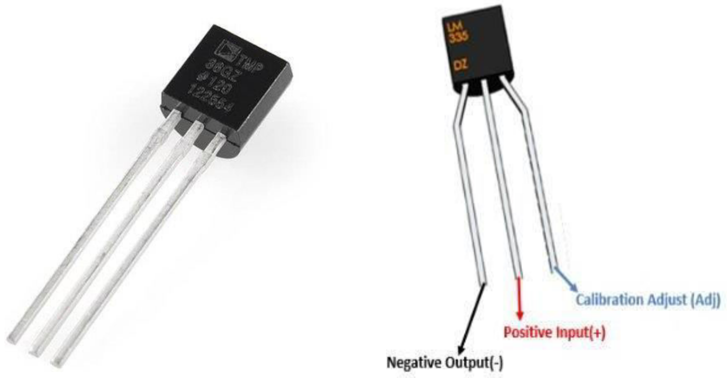


FLOW CHART


START
TEMPERATURE SENSOR
IR SENSORS
TEMPERATURE>T0
No
Yes
DO NOTHING
SEND NOTIFICATION (SMS) TO THE REGISTERED NUMBER AND DISPLAY THE NO: OF PEOPLE IN THE LCD

SPECIFICATIONS OF THE EQUIPMENT

TEMPERATURE SENSOR (LM335)

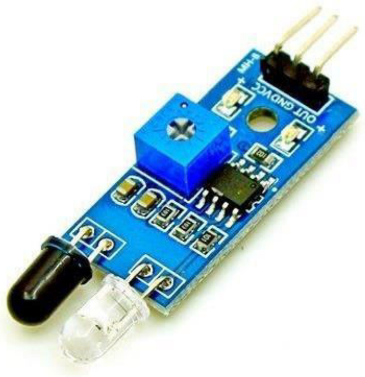


A temperature sensor is an electronic device that measures the temperature of its environment and converts the input data into electronic data to record, monitor, or signal temperature changes.

- Reverse Current: 15mA.

- Forward Current: 10mA.
- Operating Output Voltage: 2.95V to 3.01V.
- Temperature Error (at 25°C): 2°C (max).
- Thermal Resistance: 202°C/W.
- Specified Temperature Range: -40 to 100°C.
- Storage Temperature Range: -60 to 150°C.

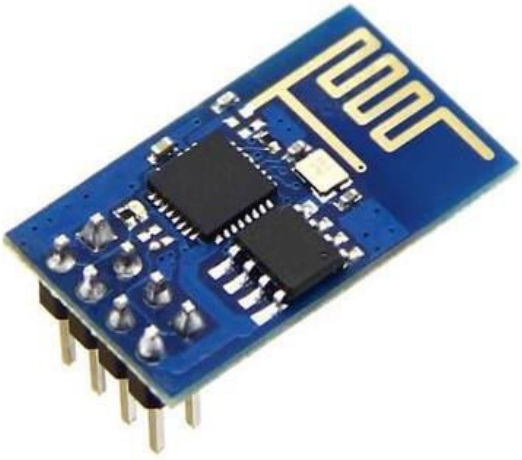
INFRA-RED (IR) SENSOR



The emitter is an IR LED and the detector is an IR photodiode. The IR photodiode is sensitive to the IR light emitted by an IR LED. The photo-diode's resistance and output voltage change in proportion to the IR light received.

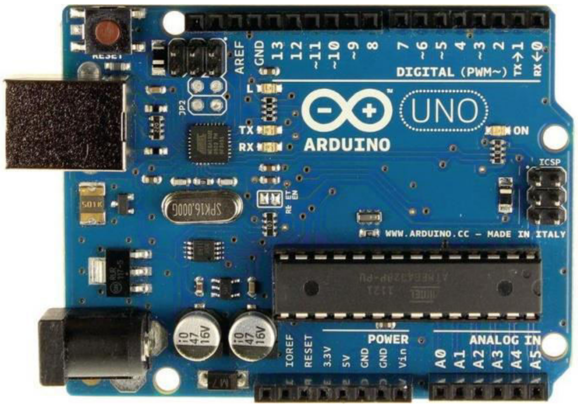
- 5VDC Operating voltage.
- I/O pins are 5V and 3.3V compliant.
- Range: Up to 20cm.
- Adjustable Sensing range.
- Built-in Ambient Light Sensor.
- 20mA supply current.
- Mounting hole.

WiFi Module (ESP8266)



The ESP8266 WiFi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your WiFi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Finetworking functions from another application processor.

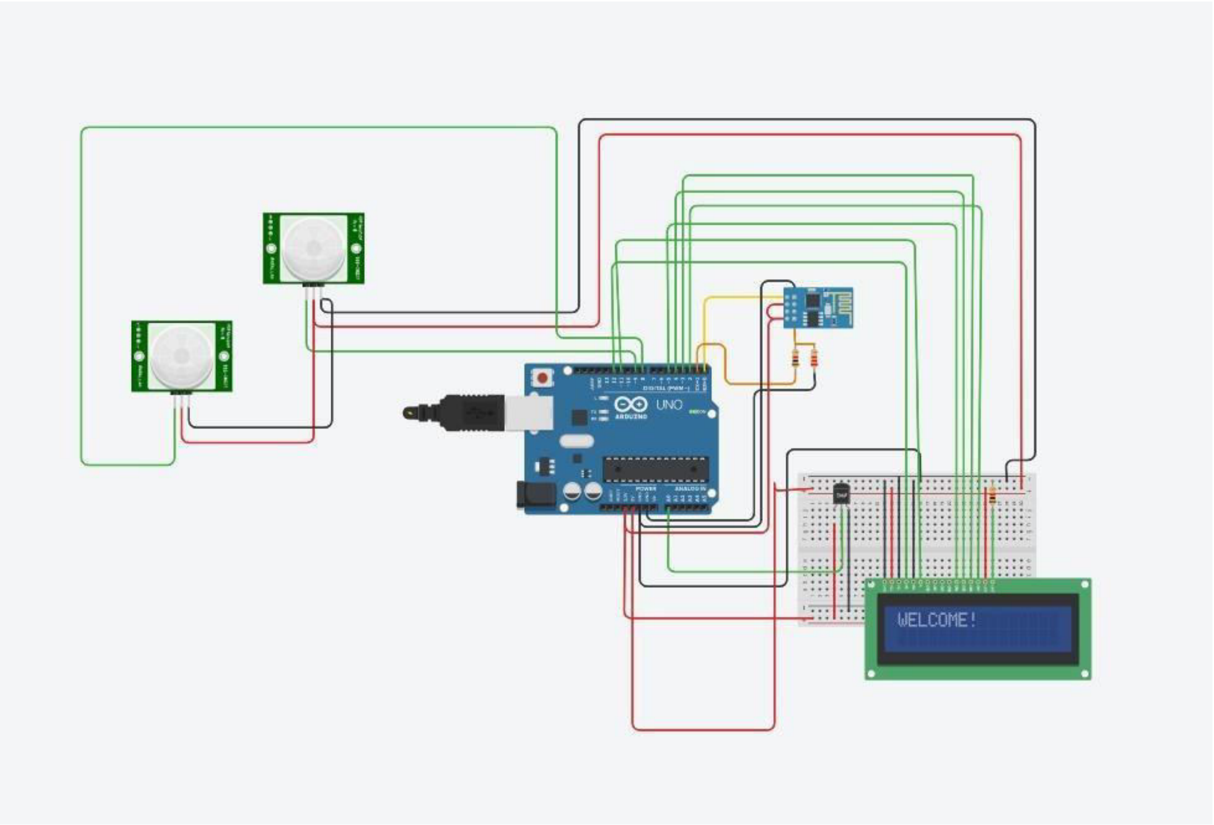
Arduino



Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning onan LED, publishing something online.

2. IMPLEMENTATION AND ANALYSIS

CIRCUIT

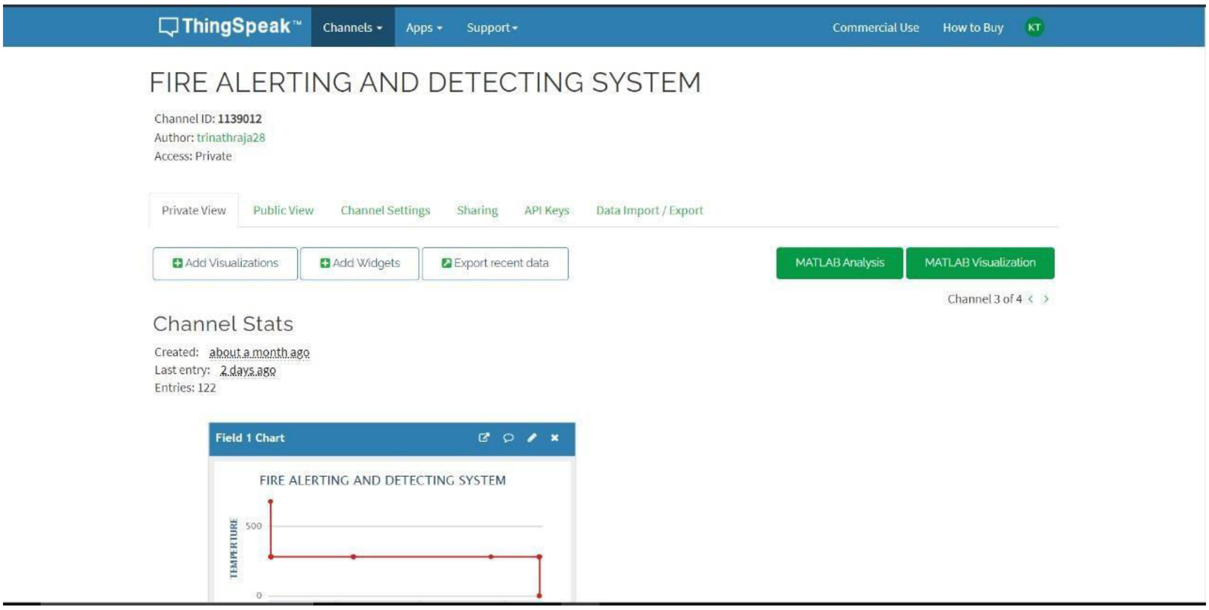


CODING

```
#include<LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);String ssid        = "Simulator Wifi"; String password = "";
String host              = "api.thingspeak.com"; const int httpPort     = 80;
String uri               = "/update?api_key=IAW0VWRDS2AHIB6V&field1="; int sensePin = A0;
int sensorInput; double temp; const int in = 9;
const int out = 8;int c1=0;
int c2=0;


int setupESP8266(void) {
  Serial.begin(115200); Serial.println("AT"); delay(10);
  if (!Serial.find("OK")) return 1;
  Serial.println("AT+CWJAP=\"" + ssid + "\",\"" + password + "\""); delay(10);
  if (!Serial.find("OK")) return 2; Serial.println("AT+CIPSTART=\"TCP\",\"" + host + "\"," +
  httpPort); delay(50);
  if (!Serial.find("OK")) return 3;return 0;
  }
void anydata(void) { sensorInput = analogRead(A0);
  temp = (double)sensorInput / 1024;temp = temp * 5;
  temp = temp - 0.5;temp = temp * 100;
  String httpPacket = "GET " + uri + String(temp) + " HTTP/1.1\r\nHost: "
  + host + "\r\n\r\n";
  int length = httpPacket.length(); Serial.print("AT+CIPSEND="); Serial.println(length); delay(10);
  Serial.print(httpPacket); delay(10);
  if (!Serial.find("SEND OK\r\n")) return;
  }
  void setup() {

  setupESP8266(); lcd.begin(16, 2);

  lcd.setCursor(0,0); lcd.print("WELCOME!"); pinMode(in,INPUT); pinMode(out,INPUT);

  }
void loop() {anydata(); delay(100);
  if(digitalRead(in)==1)
  {
  c1=c1+1;
  lcd.clear(); lcd.setCursor(0,0); lcd.print("No of people ");lcd.setCursor(0,1); lcd.print(c1);
  delay(100);

  }
```

```
if(digitalRead(out)==1)
{
if(c1>0){
c1=c1-1;}
lcd.clear(); lcd.setCursor(0,0); lcd.print("No of people "); lcd.setCursor(0,1); lcd.print(c1);
delay(100);

}

}
```

ThingSpeak Platform Configuration

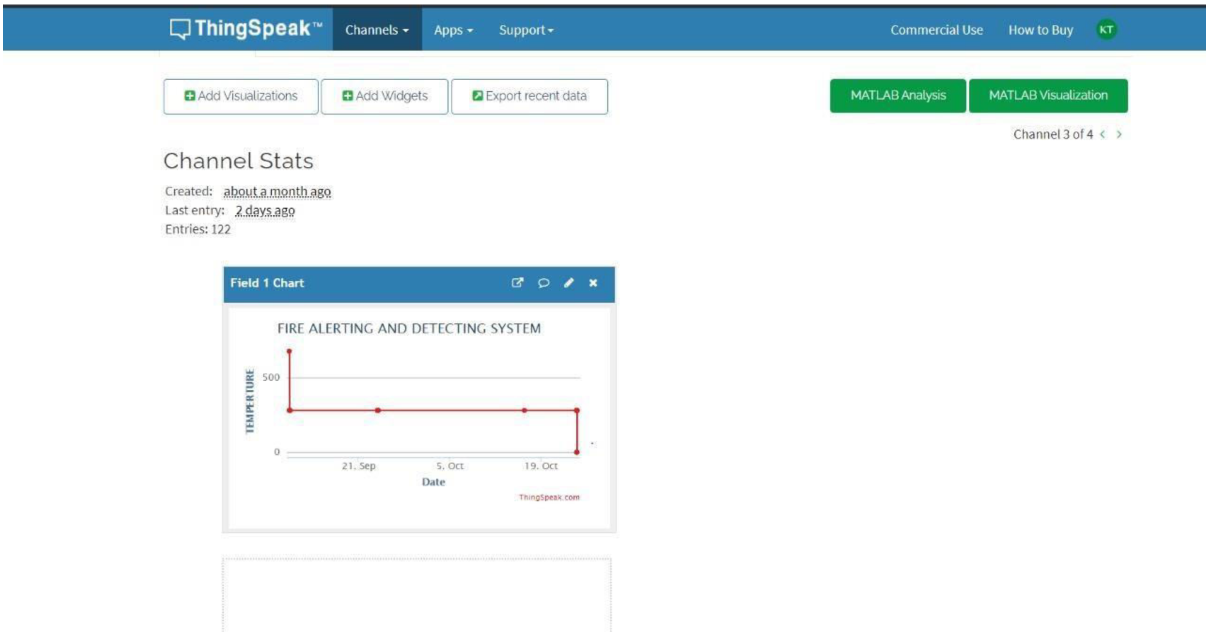


FIRE ALERTING AND DETECTING SYSTEM

Channel ID: 1139012
Author: trinathraja28
Access: Private

Channel Stats

Created: about a month ago
Last entry: 2 days ago
Entries: 122

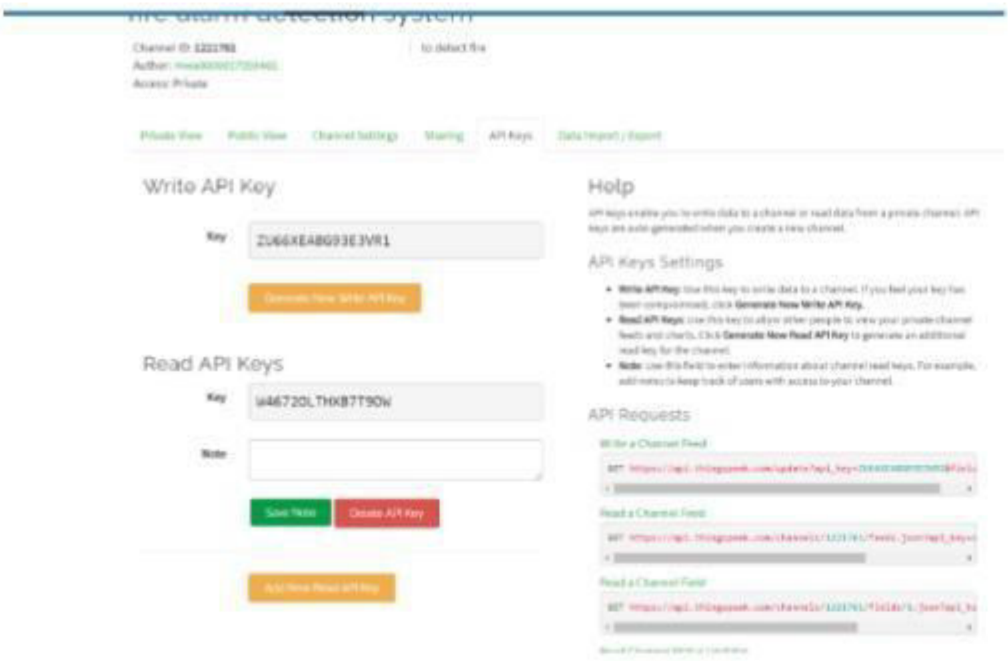


Channel Stats

Created: about a month ago

Last entry: 2 days ago

Entries: 122



Fire alarm detection system

Channel ID: 1221761

Author: [illegible]

Access: Private

to detect fire

Write API Key

Key: ZU66XEABG93E3VR1

Read API Keys

Key: W46720LTHKB7T90W

Note:

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key.**
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Write a Channel Feed

```
GET https://api.thingspeak.com/update?api_key=ZU66XEABG93E3VR1&field
```

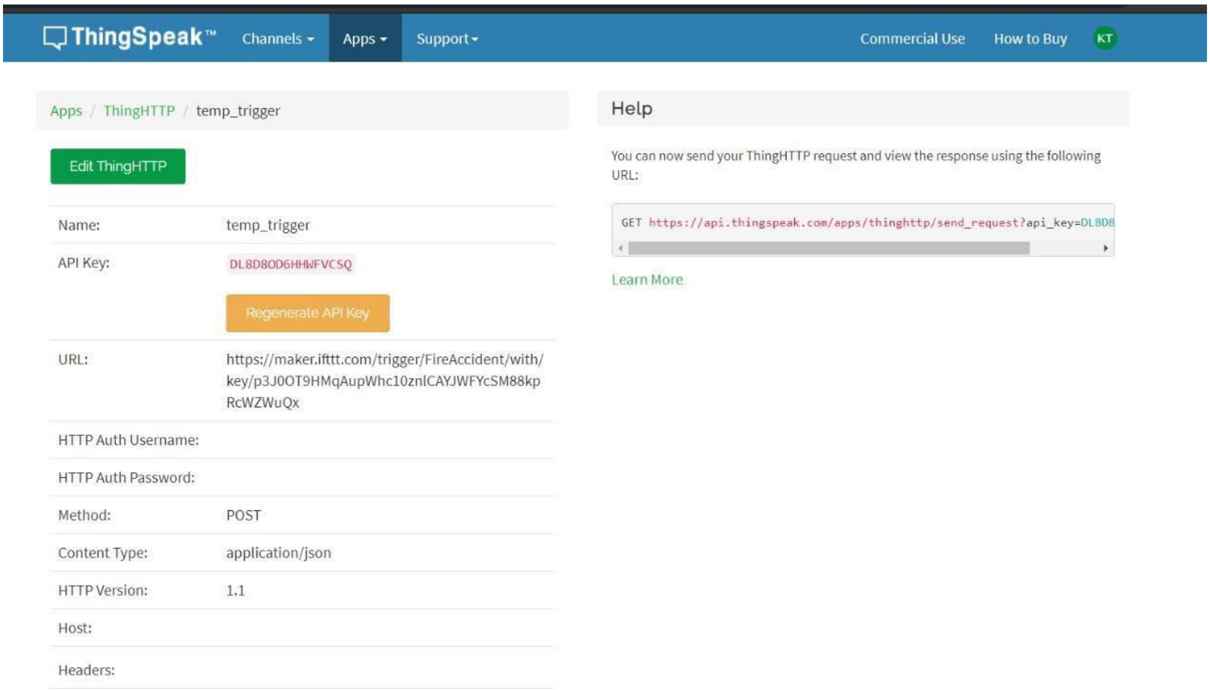
Read a Channel Feed

```
GET https://api.thingspeak.com/channels/1221761/feeds.json?api_key=
```

Read a Channel Field

```
GET https://api.thingspeak.com/channels/1221761/fields/1.json?api_ke
```

[illegible]



Apps / ThingHTTP / temp_trigger

Edit ThingHTTP

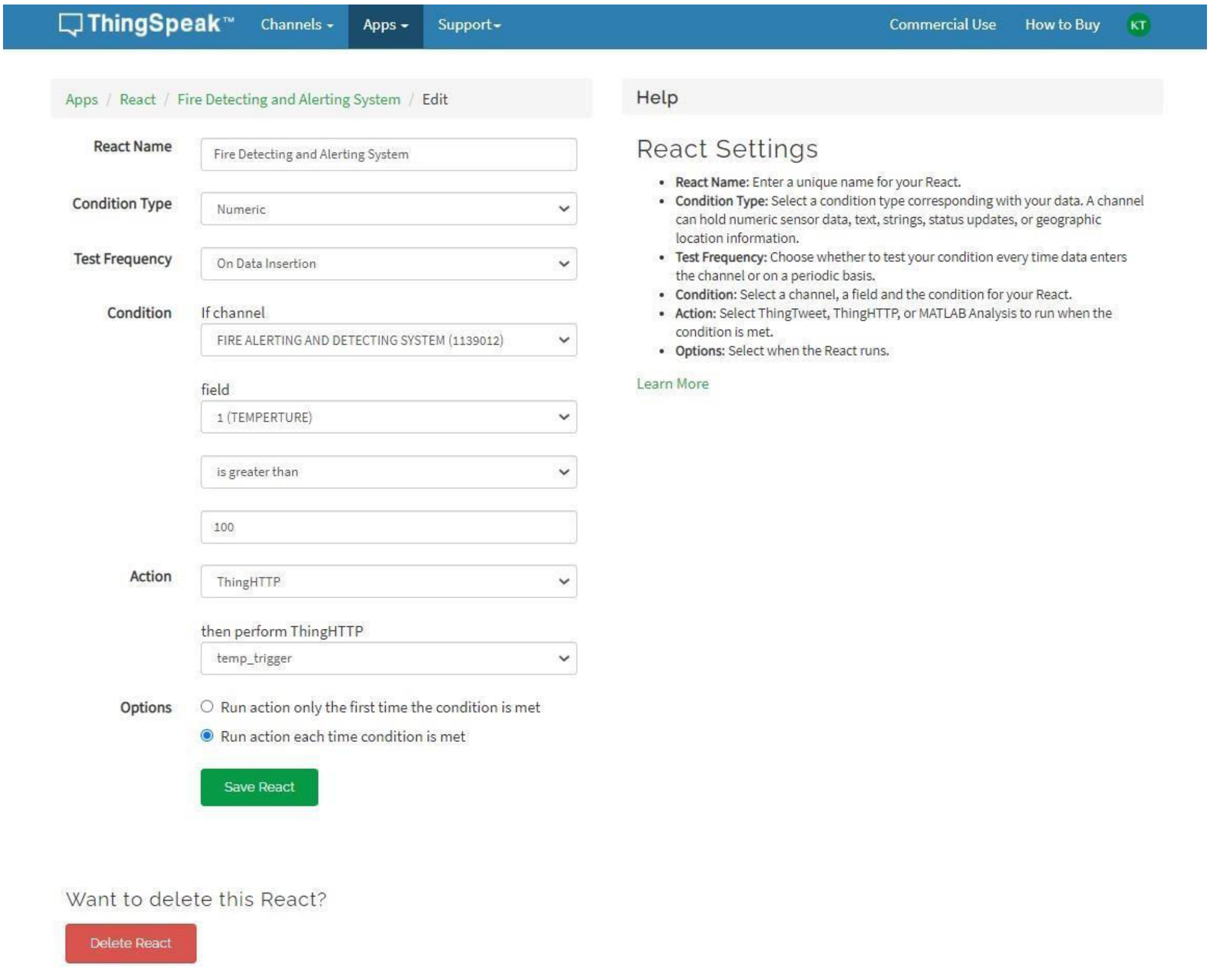
Name:	temp_trigger
API Key:	DL8D8OD6HHWFVCSQ Regenerate API Key
URL:	https://maker.ifttt.com/trigger/FireAccident/with/key/p3J0OT9HMqAupWhc10znlCAYJWFYcSM88kpRcWZWuQx
HTTP Auth Username:	
HTTP Auth Password:	
Method:	POST
Content Type:	application/json
HTTP Version:	1.1
Host:	
Headers:	

Help

You can now send your ThingHTTP request and view the response using the following URL:

```
GET https://api.thingspeak.com/apps/thinghttp/send_request?api_key=DL8D8
```

Learn More



Apps / React / Fire Detecting and Alerting System / Edit

React Name	Fire Detecting and Alerting System
Condition Type	Numeric
Test Frequency	On Data Insertion
Condition	If channel FIRE ALERTING AND DETECTING SYSTEM (1139012) field 1 (TEMPERTURE) is greater than 100
Action	ThingHTTP then perform ThingHTTP temp_trigger
Options	○ Run action only the first time the condition is met ● Run action each time condition is met

Save React

Want to delete this React?

Delete React

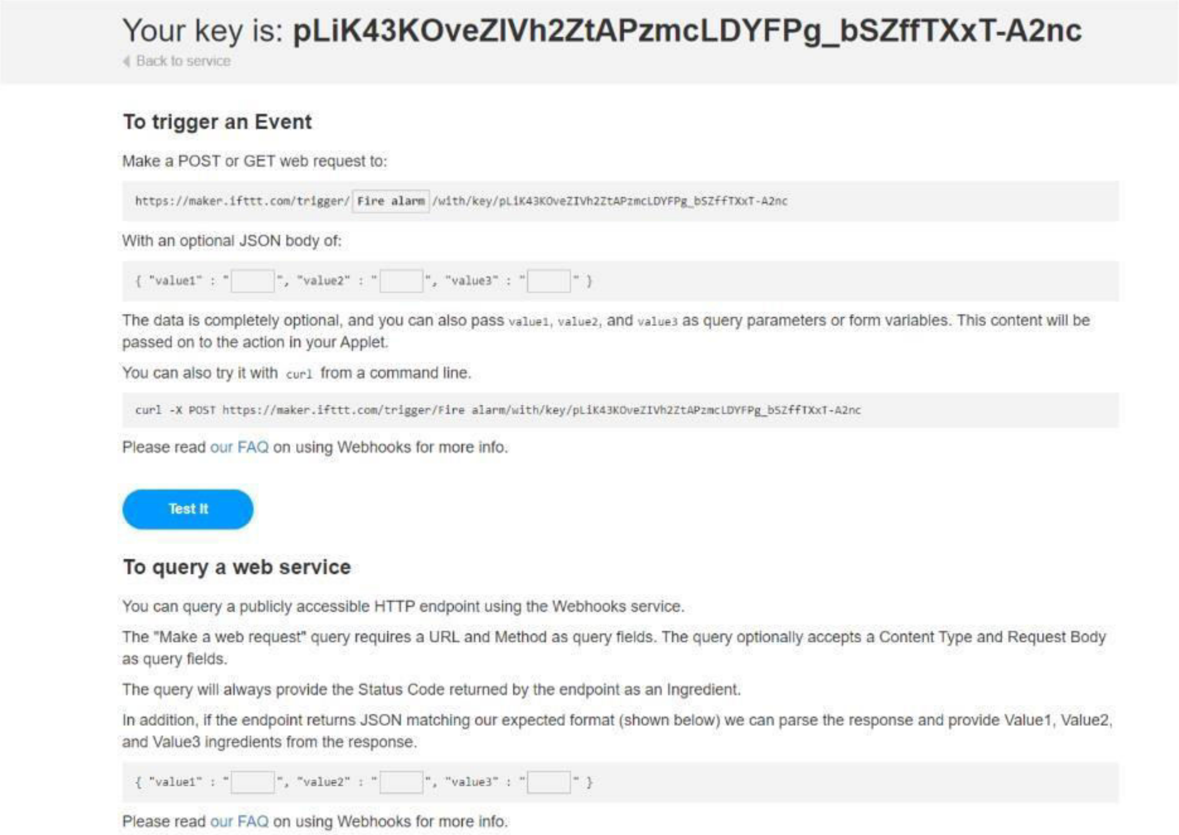
Help

React Settings

- **React Name:** Enter a unique name for your React.
- **Condition Type:** Select a condition type corresponding with your data. A channel can hold numeric sensor data, text, strings, status updates, or geographic location information.
- **Test Frequency:** Choose whether to test your condition every time data enters the channel or on a periodic basis.
- **Condition:** Select a channel, a field and the condition for your React.
- **Action:** Select ThingTweet, ThingHTTP, or MATLAB Analysis to run when the condition is met.
- **Options:** Select when the React runs.

Learn More

ifttt Configuration



Your key is: **pLiK43KOveZIVh2ZtAPzmcLDYFPg_bSZffTXxT-A2nc**

◂ Back to service

To trigger an Event

Make a POST or GET web request to:

```
https://maker.ifttt.com/trigger/ Fire alarm /with/key/pLiK43KOveZIVh2ZtAPzmcLDYFPg_bSZffTXxT-A2nc
```

With an optional JSON body of:

```
{ "value1" : "    ", "value2" : "    ", "value3" : "    " }
```

The data is completely optional, and you can also pass `value1`, `value2`, and `value3` as query parameters or form variables. This content will be passed on to the action in your Applet.

You can also try it with `curl` from a command line.

```
curl -X POST https://maker.ifttt.com/trigger/Fire alarm/with/key/pLiK43KOveZIVh2ZtAPzmcLDYFPg_bSZffTXxT-A2nc
```

Please read our FAQ on using Webhooks for more info.

To query a web service

You can query a publicly accessible HTTP endpoint using the Webhooks service.

The "Make a web request" query requires a URL and Method as query fields. The query optionally accepts a Content Type and Request Body as query fields.

The query will always provide the Status Code returned by the endpoint as an Ingredient.

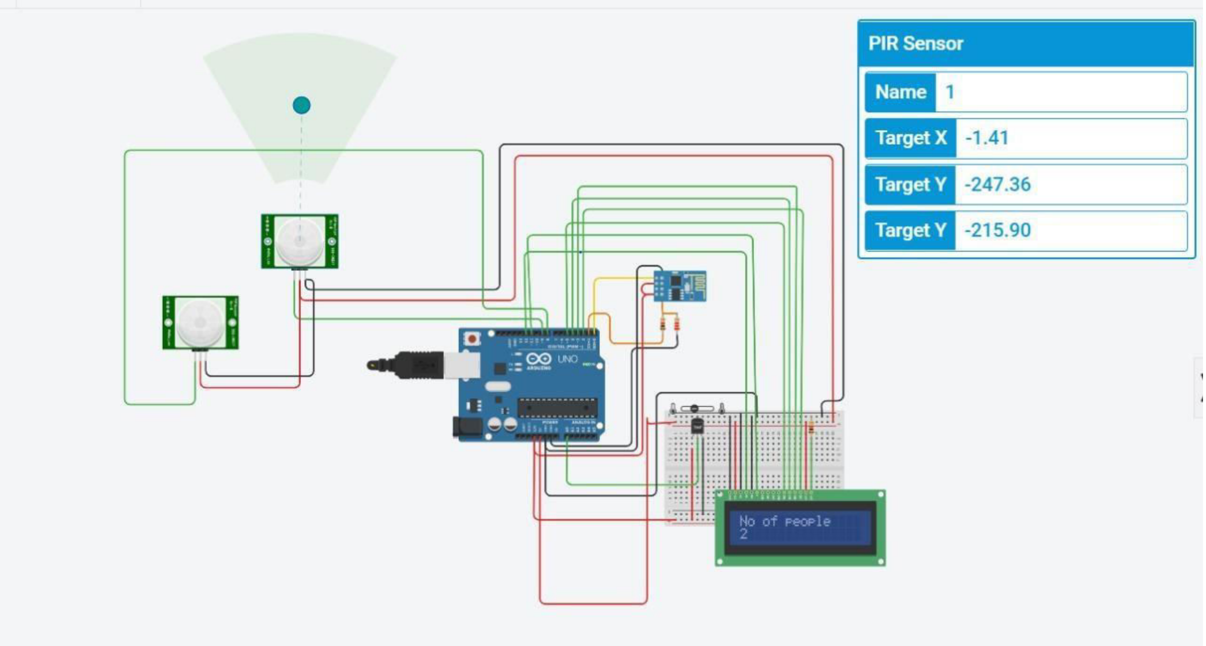
In addition, if the endpoint returns JSON matching our expected format (shown below) we can parse the response and provide Value1, Value2, and Value3 ingredients from the response.

```
{ "value1" : "    ", "value2" : "    ", "value3" : "    " }
```

Please read our FAQ on using Webhooks for more info.

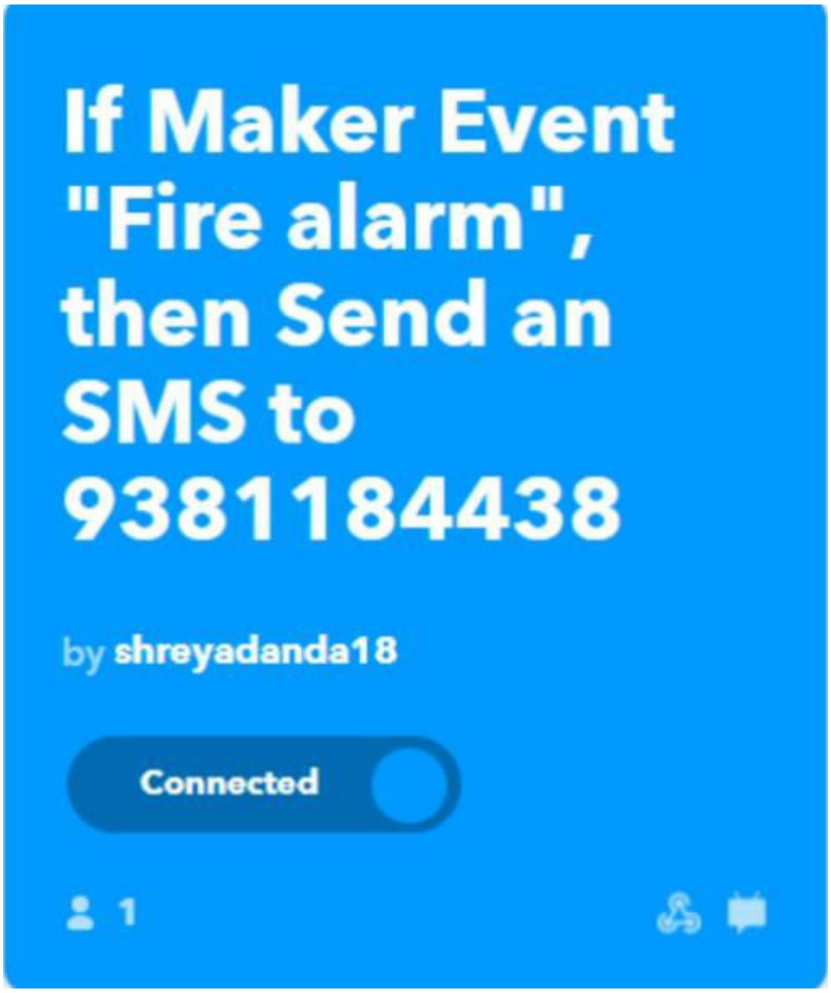
Results Obtained

- LCD screen displays no of ppl in room.
- PIR sensors used at doors which keeps the count when a person enters orleaves the room.



PIR Sensor

Name	1
Target X	-1.41
Target Y	-247.36
Target Y	-215.90



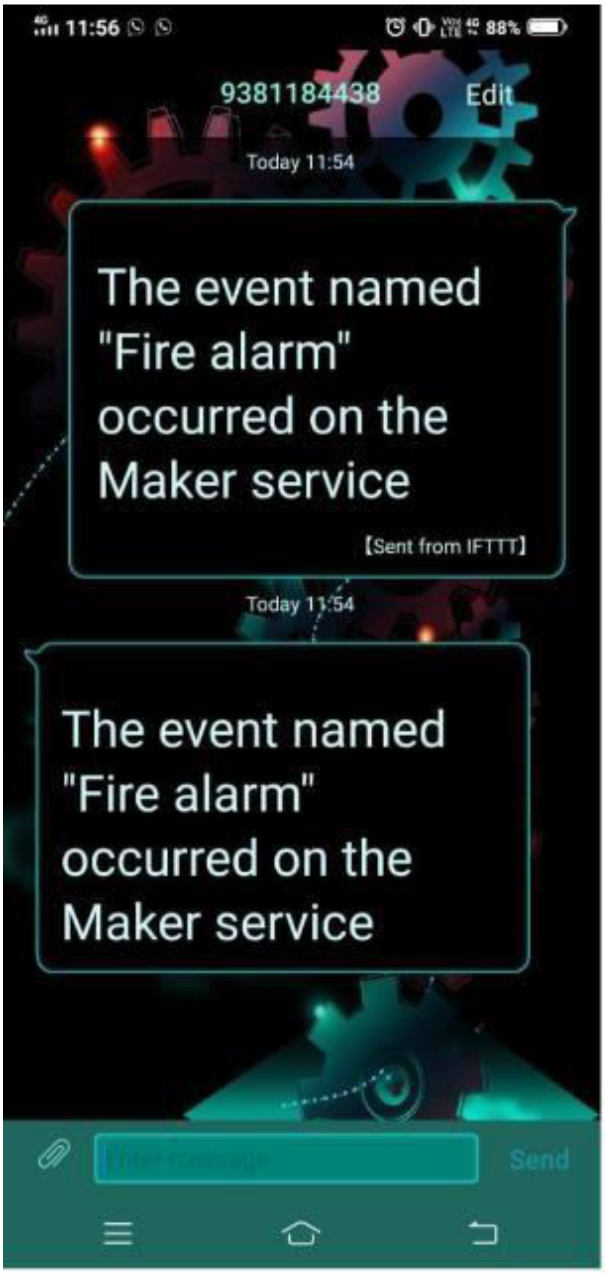
If Maker Event "Fire alarm", then Send an SMS to 9381184438

by shreyadanda18

Connected

1

SMS to the registered number,



Today 11:54

The event named "Fire alarm" occurred on the Maker service

[Sent from IFTTT]

Today 11:54

The event named "Fire alarm" occurred on the Maker service

3. CONCLUSION AND FUTURE WORK

Result and Conclusion

Therefore, the proposed system which alerts with the help of a notification system to the Fire Department or the required personal, when the recordings of the sensor passes the preset threshold value. The number of the people in separate areas(rooms) in the affected area(building) will be displayed in the LCD.

Inference

A person counter implemented to the alerting system will make sure that the correct amount of equipment is taken and there won't be shortage when required.

4.2 FUTURE WORK

Adding cameras, for real-time monitoring of the affected areas which only switch on when the temperature crosses the preset value.

Researching on ways to reduce the number of IR sensors used.

REFERENCES

https://ieeexplore.ieee.org/document/7942622

https://ieeexplore.ieee.org/document/8835265

https://ieeexplore.ieee.org/document/7306741

https://ieeexplore.ieee.org/document/8666329

https://ieeexplore.ieee.org/abstract/document/7231402

https://ieeexplore.ieee.org/abstract/document/8058270

https://ieeexplore.ieee.org/document/8724380

https://ieeexplore.ieee.org/document/4721350

https://ieeexplore.ieee.org/document/5477835

https://ieeexplore.ieee.org/document/5764264

https://ieeexplore.ieee.org/document/4350581

https://ieeexplore.ieee.org/document/7794000

https://ieeexplore.ieee.org/document/5539616/authors#authors

https://ieeexplore.ieee.org/document/8093388/authors#authors

https://ieeexplore.ieee.org/document/8934458

https://ieeexplore.ieee.org/document/8728383

https://ieeexplore.ieee.org/document/8389132

https://ieeexplore.ieee.org/document/8625538

https://ieeexplore.ieee.org/document/6820527