Internet of Things

UE19CS313

Covid-19 Monitoring Device

Team

Abhishek V	PES2UG19CS012
Balendra DP	PES2UG19CS100
Pavan N	PES2UG19CS277

Use case

One of the best ways to avoid the pandemic are social distancing, staying at home, avoiding groups of people, washing hands frequently, and to avoid touching the face. Wearing a mask is one of the best methods to stop the spreading of the virus. Wearing a mask will stop almost all of the droplets for 90% (when sneezing), that is why wearing a mask is mandatory during this pandemic. Almost all of Covid-19 patients experienced fever, it can be detected by measuring the body's temperature. WHO said that the body's temperature for fever is above 38 Celsius. This physical computing device installed in public places such as shopping malls, restaurants, metro stations, etc helps monitor the body temperature of the general public visiting these places and dispense sanitizer automatically.
This also provides useful insights to the owners about their customers and helps take the required precautionary measures on time.

Benefits of Non-contact Temperature Assessment Devices

These non-contact devices can quickly measure and display a temperature reading so a large number of people can be evaluated individually at points of entry.
Non-contact infrared thermometers require minimal cleaning between uses.
Using non-contact temperature measurement devices may help reduce the risk of spreading COVID-19 infections

Features:

1. Body temperature screening
2. Sanitizer dispensor
3. Ensure social distancing is maintained
4. Data analytics of customers visiting the place at a particular time
5. Can be portable if connected to a battery power supply, however the DC motor consumes a lot of power.

Software requirements:

1. Tinkercad - Online simulation tool
2. Arduino IDE - Open-source electronic prototyping and code editor
3. Thingspeak Cloud - Cloud data analytics platform

Hardware requirements:

1. Arduino Uno R3
2. Power supply
3. DC motor
4. NPN transistor (BJT)
5. LED
6. Resistor
7. Ultrasonic distance sensor
8. Potentiometer
9. Photodiode
10. Neopixel ring 12
11. Temperature sensor(TMP36)
12. Piezo
13. LCD 16x12
14. Bread board
15. Connecting wires

Logic to implement the features:

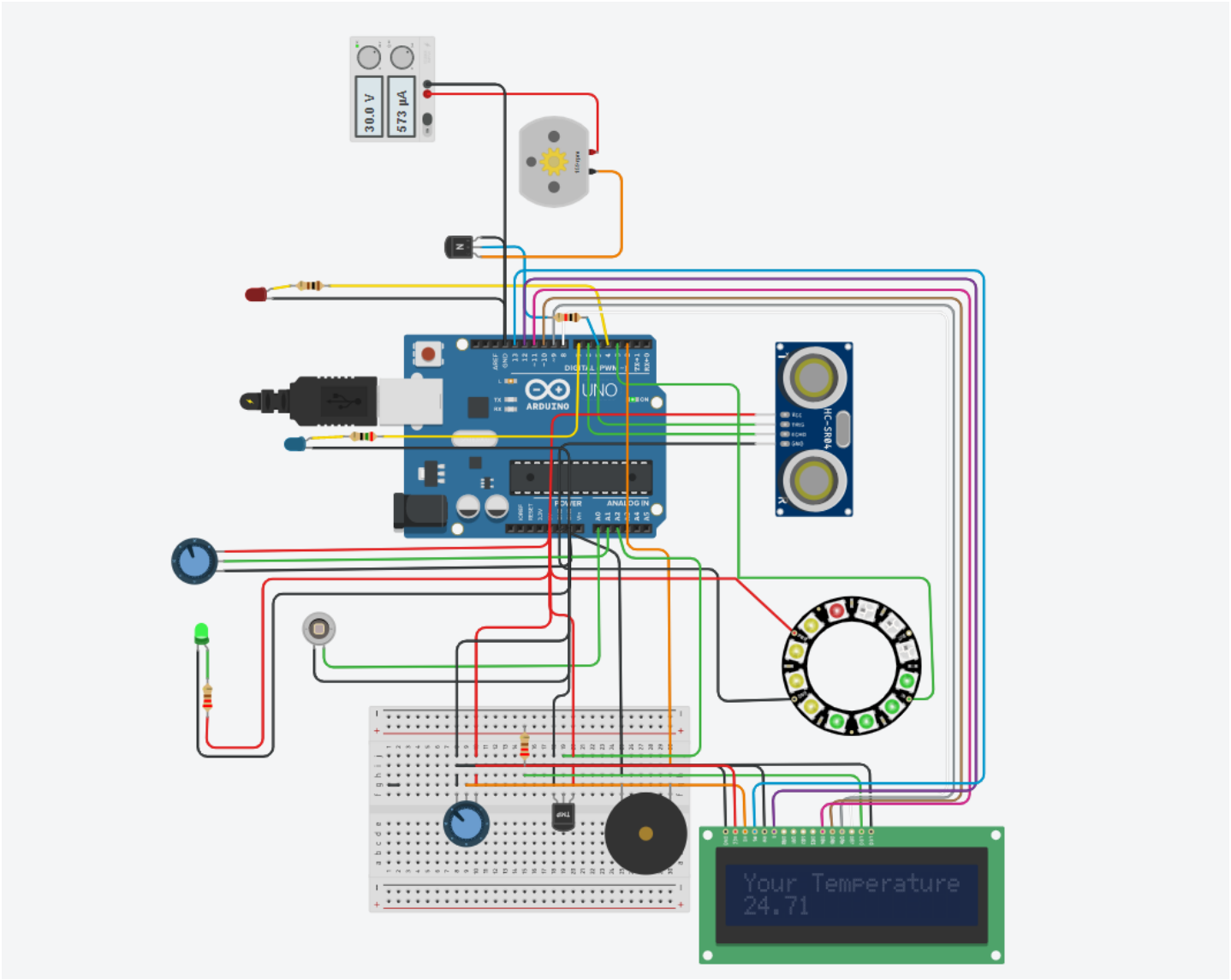
- The TMP36 temperature sensor is an easy way to measure temperature using an Arduino. The sensor can measure a fairly wide range of temperature (-50°C to 125°C), and is fairly precise (0.1°C resolution). In our project, we connect it with the LCD display and display it.

- In the case of the sanitizer dispenser the photodiode is used for measuring the infrared and the potentiometer to calculate how much volume of sanitizer to dispense. The motor acts like the dispenser here.

- For social distancing an Ultrasonic Distance Sensor is used to check how far a person is from the user. When someone gets too close to the user, the neopixel ring changes color from green to red. When the color changes to red the piezo starts producing a sound alarming the user that they're standing too close to someone.

- The data collected by the ultrasonic distance sensor and temperature sensor is sent to the the Thingspeak cloud for data analytics to draw useful insights out of it.

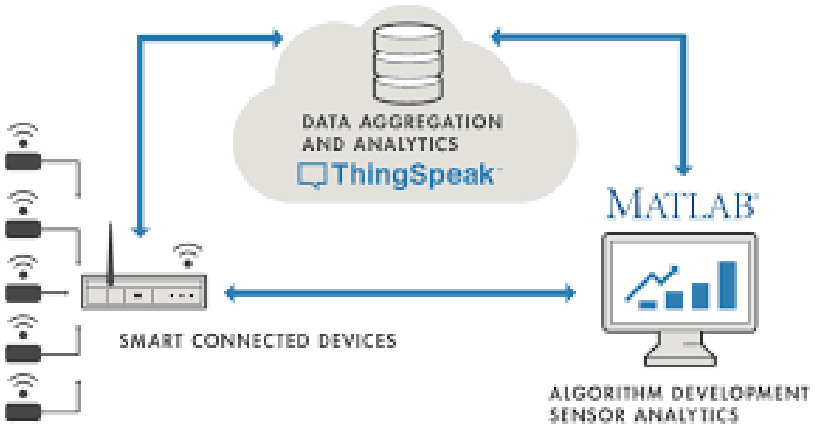
Circuit Design:



Link to circuit design and code:

https://www.tinkercad.com/things/gC15uE2S9ZI-covidscreening/editel?sharecode=8zOHFhZZ6nGF4au9m6hnOsn9r_MxCWzp1ezMhJBs9co

Pushing data to ThingSpeak Cloud:



Connecting to cloud via an API key and sending data through a GET request

```
GET https://api.thingspeak.com/update?api_key=CLZR9GMXUU9X1T1X&field1=0
```

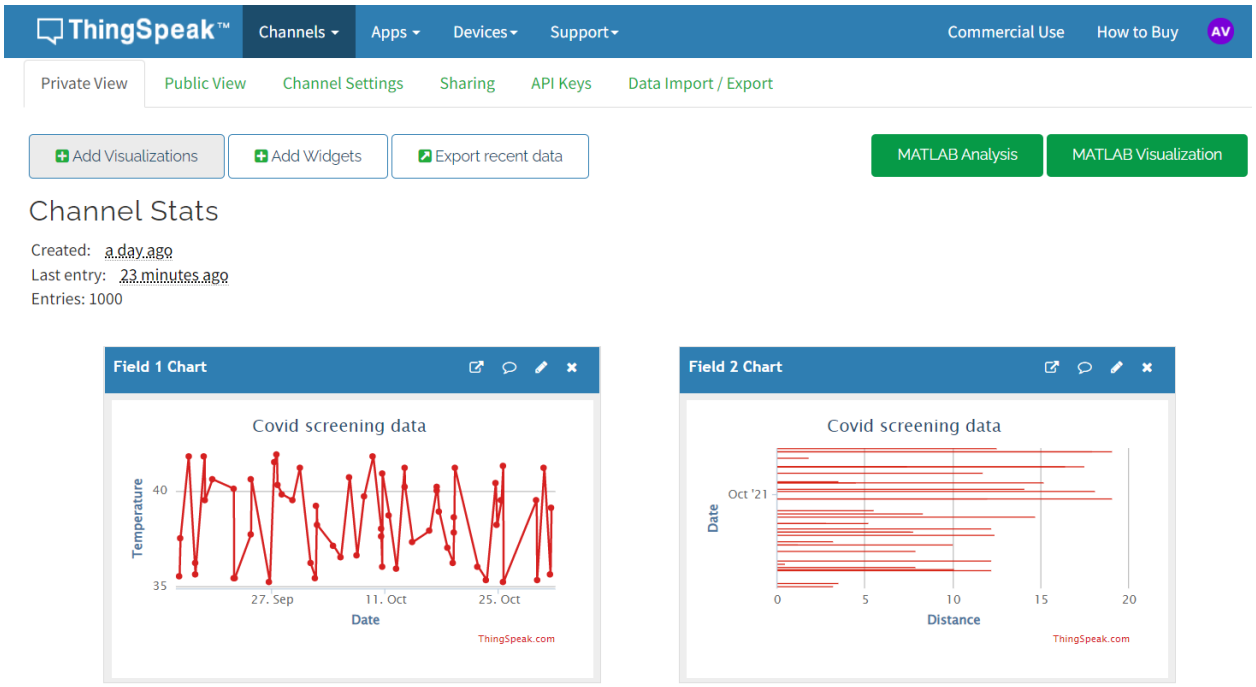
Data collected from different sensors:

Sensor	**Data collected**
Temperature sensor(TMP36)	Temperature (Degrees Celsius)
Ultrasonic Sensor	Distance (meters)

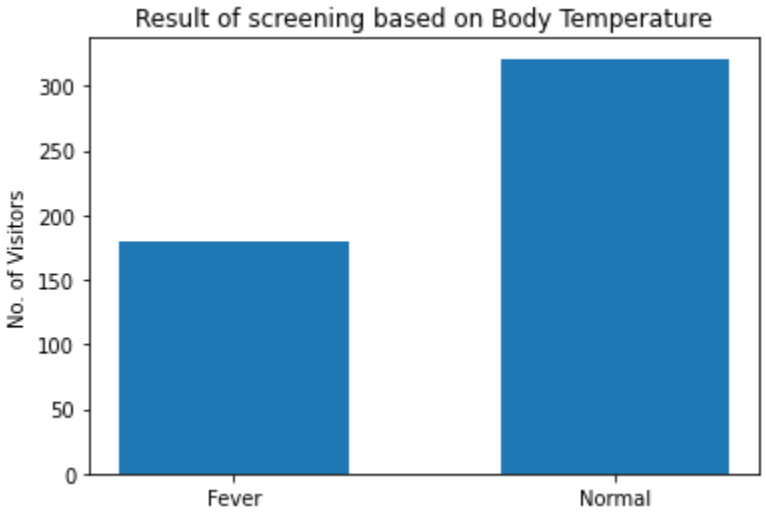
Data obtained from sensors:

	Timestamp	Temperature	Distance
0	2021-01-03 11:22:42	39.0	17.5
1	2021-06-24 07:11:59	38.0	0.4
2	2020-01-01 05:10:58	41.0	12.4
3	2021-03-22 03:17:39	39.0	14.3
4	2021-01-21 03:17:01	38.0	0.5
5	2021-04-04 11:44:08	40.0	5.1
6	2020-04-16 08:51:46	37.0	12.6
7	2021-06-27 01:56:04	35.0	4.0
8	2021-09-22 01:42:39	40.0	16.4
9	2020-03-05 06:26:43	40.0	2.3

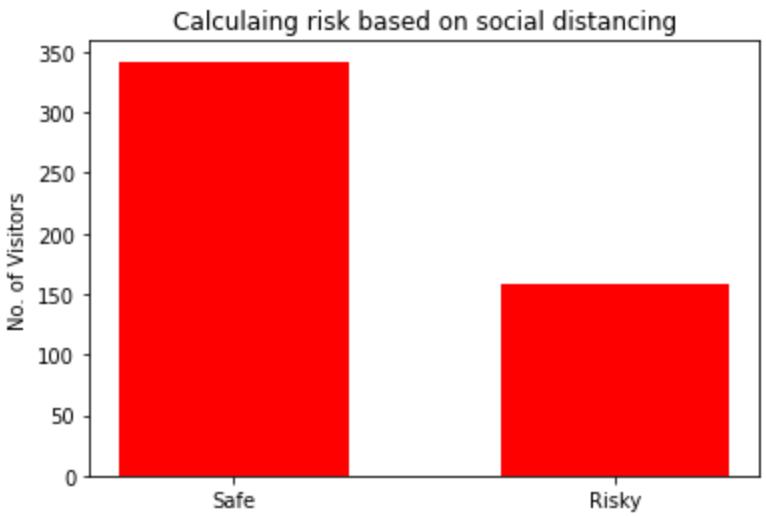
Visualizing the data sent to cloud:



ThingSpeak Dashboard



People with body temperature over 37.5 C are considered to have fever



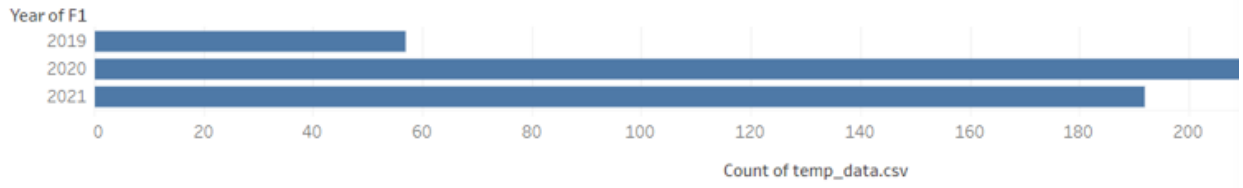
No. of visitors maintaining a safe social distance of 2m

Counting Number of visitors on a particular month and year

Year		Count
2019	10	17
	11	20
	12	20
2020	1	23
	2	23
	3	29
	4	14
	5	21
	6	22
	7	25
	8	25
	9	22
	10	17
	11	18
	12	15
2021	1	19
	2	17
	3	17
	4	15
	5	15
	6	17
	7	20
	8	23
	9	24
	10	22

Name: Timestamp, dtype: int64

Counting No. of visitors yearly:



Code Implementation:

```
#include <LiquidCrystal.h>
#include <Adafruit_NeoPixel.h>

int ledPin= 3;
int ledNo= 12;
int tempSensorPin = 2;
const int rs = 13, en = 12, d4 = 11, d5 = 10, d6 = 9, d7 = 8;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
Adafruit_NeoPixel strip=
Adafruit_NeoPixel(ledNo,ledPin,NEO_RGB+NEO_KHZ800);


int buzzerPin= 2;
int echoPin= 6;
int trigPin= 5;
int minDistance = 100;
int maxDistance = 300;

int onOffTime;
int IRSense;

int autoOffTrigger=0;
```

```
void setup()
{
  pinMode(5, OUTPUT); //Motor pump control pin
  pinMode(2, OUTPUT); //Sensor sensing Pin


  pinMode(buzzerPin, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial. begin(9600);
  strip.begin();
  lcd.begin(16, 2);
  lcd.print("Your Temperature:");
  for(int i = 0; i < ledNo; i++)
  {
   strip.setPixelColor(i,strip.Color(0,0,0));
  }
  strip.show();
}

void loop()
{

  //code for temperature sensor

  int tempReading = analogRead(tempSensorPin);  //Read the temperature
degree
  double temp;
  temp = (double)tempReading / 1024;            //find % of input reading
  temp = temp * 5;                              //multiply by 5V to get
voltage
  temp = temp - 0.5;                            //Subtract the offset
  temp = temp * 100;                            //Convert to degrees

  //code for LCD display

  lcd.setCursor(0, 1);          // set the cursor to column 0, line 1
  lcd.print(temp);

  //code for hand sanatizer dispenser
```

```
  int IRSense= analogRead(A0); // Read Sensor Value
  int onOffTime= analogRead(A1); // Read How much volume to dispense

  int time=map(onOffTime,0,1023,0,10);//convert to simple scale
  //Serial.println("IR: "+String(IRSense));
  //Serial.println("pot: "+String(onOffTime)+ ": "+String(time));

  if(IRSense >78) //If sense higher than 78 LED INDICATE
  {
    digitalWrite(7,1);
  }
  else
  {
    digitalWrite(7,0);
  }
  //IF IR sense higher than 78
  //Motor pump will ON for Sometime mentioned in "time"
  //Then Turn Off
  if(IRSense >78 && autoOffTrigger==0)
  {
    digitalWrite(4,1);
    digitalWrite(buzzerPin, HIGH); //buzzing when it dispenses the hand
sanitizer
    delay(time*1000); // 1000 is 1000 millisecond(s)
    digitalWrite(4,0);
    autoOffTrigger=1;
    Serial.println("Dispensing... ");
  }
  else if(IRSense <78)
  {
    //AutoOFFTrigger is used to cut off motor pump
    //Make for each sense, only onetime dispenser will come out
    autoOffTrigger=0;
  }

  //Code for social distancing

  int distance = calcDistance();
  //distance between the user and he person behind him
```

```
  int ledsToGlow = map(distance, minDistance, maxDistance, ledNo, 1);
  // no of leds that tell closeness between the users
  if(ledsToGlow == 12)
  {
    digitalWrite(buzzerPin, HIGH);
  }
  else
  {
    digitalWrite(buzzerPin, LOW);
  }
  for(int i = 0; i < ledsToGlow; i++)
  {
    if(i < 4)
    {
      strip.setPixelColor(i,strip.Color(50,0,0));//green,red,blue
    }
    else if(i >= 4 && i < 8)
    {
      strip.setPixelColor(i,strip.Color(50,50,0));//green,red,blue
    }
    else if(i >= 8 && i < 12)
    {
      strip.setPixelColor(i,strip.Color(0,50,0));//green,red,blue
    }
  }
  for(int i = ledsToGlow; i < ledNo; i++)
  {
    strip.setPixelColor(i,strip.Color(0,0,0));
  }
  strip.show();
  delay(50);
}
GET https://api.thingspeak.com/update?api_key=CLZR9GMXUU9X1T1X&field1=0
int calcDistance()
{
  long distance,duration;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
```

```
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration/29/2;
  if(distance >= maxDistance)
  {
    distance = maxDistance;
  }
  if(distance <= minDistance)
  {
    distance = minDistance;
  }
  return distance;
}
```

References:

1. https://www.arduino.cc/reference/en/
2. https://www.tinkercad.com/learn
3. https://www.tinkercad.com/learn/designs/learning
4. https://www.electronicshub.org/arduino-temperature-sensors
5. https://create.arduino.cc/projecthub/electropeak/the-beginner-sguide-to-control-motors-by-arduino-and-l293d-139307
6. https://create.arduino.cc/projecthub/electropeak/the-beginner-sguide-to-control-motors-by-arduino-and-l293d-139307
7. How To Collect, Analyze, and Act on IoT Data - ThingSpeak IoT