

Intelligently Automated Traffic Management System

Presented By:

Madhyam Patra

N Sriya Setty

Shubhankar Gadad

Sneha Dabbiru

1. INTRODUCTION

1.1 Overview

In cities, where the number of vehicles continuously increases faster than the available traffic infrastructure to support them, congestion is a difficult issue to deal with and it becomes even worse in case of car accidents. This problem affects many aspects of the modern society, including economic development, traffic accidents, increase in greenhouse emissions, time spent, and health damages. In this context, modern societies can rely on traffic management system to minimize traffic congestion and its negative effects. Traffic management systems are composed of a set of application and management tools to improve the overall traffic efficiency and safety of the transportation systems. Furthermore, to overcome such issue, traffic management system gathers information from heterogeneous sources, exploits such information to identify hazards that may potentially degrade the traffic efficiency, and then provides services to control them. With this question in mind, this article presents a classification, review, challenges, and future perspectives to implement a traffic management system.

1.2 Purpose

It has become very easy for a common person to own a vehicle with affordability and higher purchasing power. It creates a problem in terms of road congestion and increasing traffic in big cities though this has led to comfortable lifestyles. Smart traffic management system is an alternative way to solve road traffic problem and this system will support existing operation system. Traffic congestion is a severe problem in almost every modern city around the world. Traffic congestion has been causing many critical problems and challenges in the major and most populated cities. It is becoming more difficult and time-consuming to travel to different places within the city. Due to these congestion problems, people lose time, miss opportunities, and get frustrated. Due to traffic congestion, there is a loss in productivity from workers, trade opportunities are lost, delivery gets delayed, and thereby the costs go on increasing.

2. LITERATURE SURVEY

2.1 Existing Problem

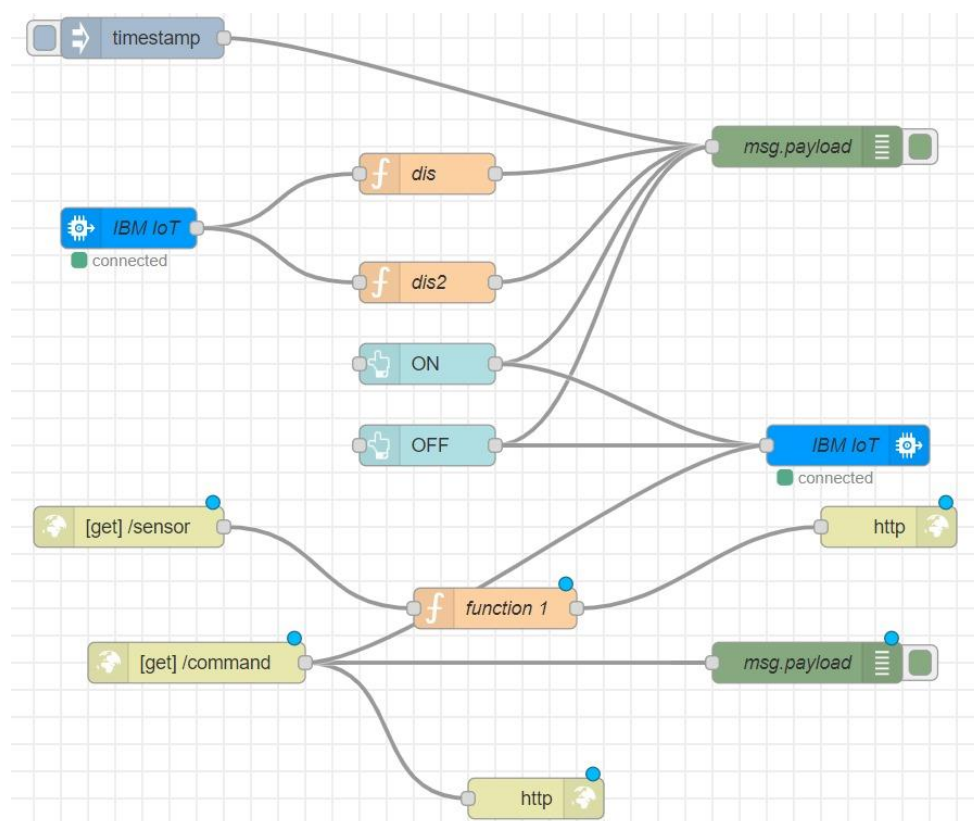
Due to the increasing population in the metropolitan cities and the urban areas, traffic management has become a major problem causing a havoc in the day-to-day life of the people. The number of accidents that take place keep increasing due to increase in the number of vehicles used by the people. By managing the traffic efficiently, this would avoid umpteen number of mishaps that take place. We have identified that congestion clearance and police alert system to prevent mischievous activities are the major drawbacks of today's managements systems, which we have tried to eradicate with the help of this project.

2.2 Proposed Solution

Our approach is the Intelligently Automated System, which uses automated traffic lights to control traffic congestion. The system will automatically empty the lane for the emergency vehicle (i.e., ambulances) once the controllers with the Traffic Management Authority staff have been activated to identify the emergency car in the lane. The police can use our system's alert function to automatically open the barricades on the lane and apprehend any criminals they are pursuing by pressing the alert button while doing so.

3. THEORETICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware/Software Designing

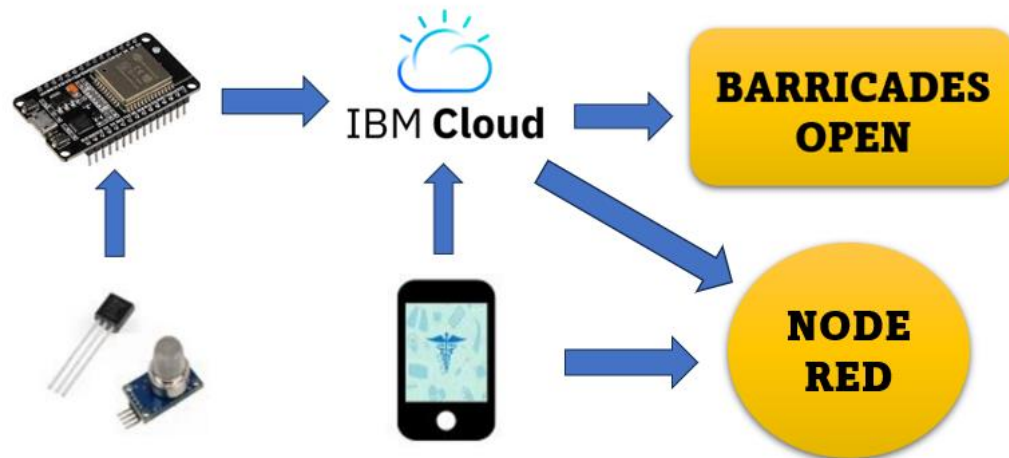
Hardware Used to Prepare: ESP32, LED's, Switch, Jumper Wires, Ultrasonic sensors

Software Used to Prepare: Wokwi, IBM Cloud, Node Red, MIT App Inventor

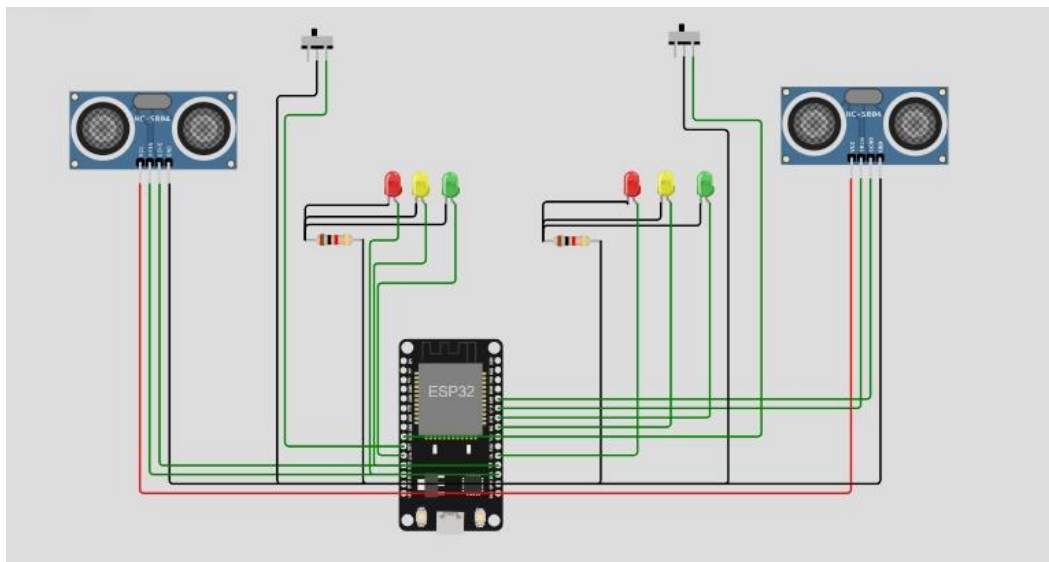
4. EXPERIMENTAL INVESTIGATIONS

1. **Hardware Setup:** Connect the Ultrasonic sensors to an IoT-enabled device like an ESP32 microcontroller. Ensure proper power supply and wiring connections.
2. **Data Transmission:** Connect the ultrasonic sensors to an IoT-enabled device like an ESP32 microcontroller. Ensure proper power supply and wiring connections.
3. **Cloud Integration:** Set up a cloud-based platform to receive and store the sensor data. Services like AWS IoT, Azure IoT, or Google Cloud IoT can be utilized for this purpose. Here we have made use of IBM Cloud.
4. **Alert Generation:** Define criteria for Police Alert and generate alerts and or notifications when pursuit detected. This can be accomplished by the press of button.
5. **Visualization And Reporting:** Develop a user interface to alert the respective personnel and send data via the cloud services.
6. **Testing And Validation:** Conduct controlled experiments to simulate the congestion control and trigger alert mechanism system.
7. **Iterative Improvements:** Based on the experimental results and feedback, refine the system, make necessary adjustments to improve accuracy and enhance overall performance.

5. FLOWCHART

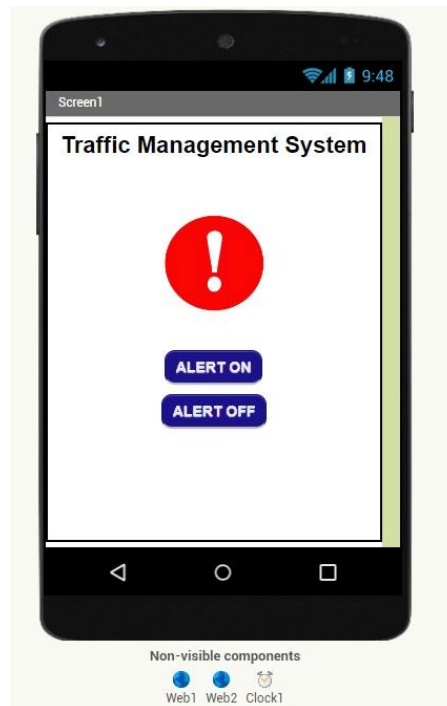


Circuit:



6. RESULT

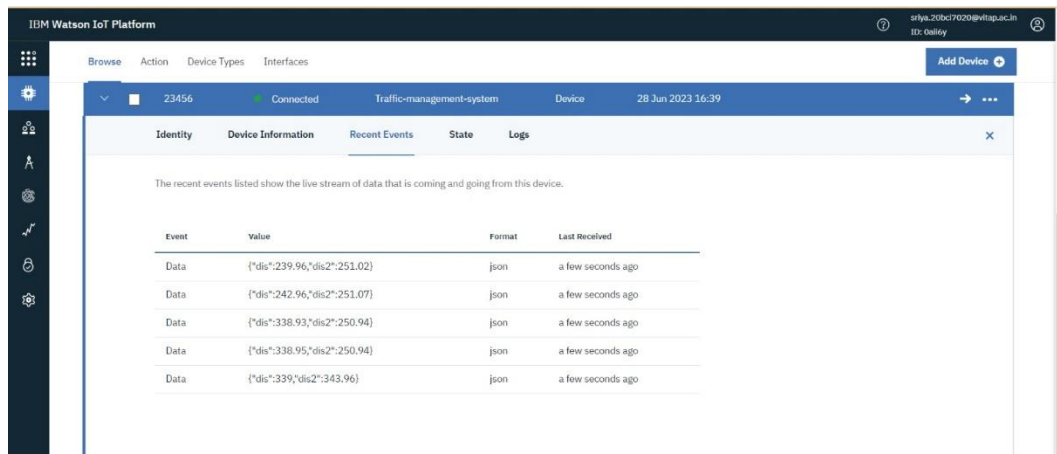
MIT-App Frontend:



MIT-App Backend:



IBM Platform:



IBM Watson IoT Platform

23456 Connected Traffic-management-system Device 28 Jun 2023 16:39

Identity Device Information **Recent Events** State Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"dis":239.96,"dis2":251.02}	json	a few seconds ago
Data	{"dis":242.96,"dis2":251.07}	json	a few seconds ago
Data	{"dis":338.93,"dis2":250.94}	json	a few seconds ago
Data	{"dis":338.95,"dis2":250.94}	json	a few seconds ago
Data	{"dis":339,"dis2":343.96}	json	a few seconds ago

Barricades Opening Output:

```
S1: 329.94 S2: 284.94 Sending payload: {"dis":329.94,"dis2":284.94}
Publish ok
callback invoked for topic: iot-2/cmd/command/fmt/String
data: {"command":"alerton"}
{"command":"alerton"}
Barricades opened
S1: 329.97 S2: 284.94 Sending payload: {"dis":329.97,"dis2":284.94}
Publish ok
```

Barricades Closing Output:

```
S1: 329.94 S2: 284.94 Sending payload: {"dis":329.94,"dis2":284.94}
Publish ok
callback invoked for topic: iot-2/cmd/command/fmt/String
data: {"command":"alertoff"}
{"command":"alertoff"}
Barricades closed
S1: 330.00 S2: 284.95 Sending payload: {"dis":330.00,"dis2":284.95}
Publish ok
```

When Emergency Buttons Are Pressed:

```
Emergency btn 1
S1: 329.95 S2: 284.99 Sending payload: {"dis":329.95,"dis2":284.99}
Publish ok
Emergency btn 2
S1: 329.95 S2: 285.01 Sending payload: {"dis":329.95,"dis2":285.01}
Publish ok
```

7. ADVANTAGES AND DISADVANTAGES

7.1 Advantages:

1. Improved Traffic Flow: By dynamically adjusting traffic signals and optimizing routes, the system can alleviate congestion and reduce travel times for all vehicles.
2. Enhanced Safety: Integration with a police alert system enables real-time communication and response to incidents.
3. Emergency Vehicle Prioritization: By giving priority to emergency vehicles, the system enables faster response times, potentially saving lives during critical situations.
4. Efficient Resource Allocation: The system can optimize the use of existing infrastructure, reducing the need for costly expansions and providing cost-effective solutions for traffic management.
5. Real-Time Monitoring and Control: Continuous monitoring of traffic conditions allows for quick detection of issues and immediate adjustments to optimize traffic flow and reduce delays.

7.2 Disadvantages:

1. Initial Implementation Costs: Setting up the infrastructure and deploying the necessary sensors, cameras, and communication systems can be expensive.
2. Dependency on Technology: The system relies heavily on technology, and any malfunctions or technical glitches could disrupt its effectiveness and potentially cause traffic issues.
3. Data Privacy Concerns: Collecting and analyzing real-time data raises concerns about privacy and the potential misuse or unauthorized access to sensitive information.
4. Limited Human Judgment: While the system is designed to make intelligent decisions, it may lack the nuanced judgment that human operators possess, particularly in complex or unpredictable situations.
5. Maintenance and Upgrades: Regular maintenance and updates are required to ensure the system remains efficient and up-to-date with changing traffic patterns and technological advancements.

8. APPLICATIONS

1. Congestion Control: The system can monitor traffic conditions in real-time and dynamically adjust traffic signal timings and lane configurations to optimize traffic flow, reduce congestion, and minimize delays during peak hours or special events.

2. **Emergency Vehicle Prioritization:** The system can identify and prioritize emergency vehicles, such as ambulances or fire trucks, by giving them green signal priority or creating clear paths to reach their destinations faster. This helps in improving emergency response times and potentially saving lives.
3. **Incident Management:** The system can detect incidents such as accidents or breakdowns and automatically notify relevant authorities. Traffic can be redirected, and emergency services can be dispatched promptly, improving incident response and minimizing traffic disruptions.
4. **Public Transportation Integration:** The system can coordinate with public transportation networks, providing real-time data on bus or train schedules and optimizing traffic signals to prioritize the movement of public transportation vehicles, reducing delays, and improving overall efficiency.
5. **Future Mobility Integration:** As autonomous vehicles become more prevalent, the automated traffic management system can integrate with them, enabling efficient traffic coordination and smooth interaction between autonomous and human-driven vehicles.

9. CONCLUSION

In conclusion, an automated traffic management system with congestion control, integrated police alert system, and emergency vehicle prioritization represents a significant advancement in urban transportation management. By harnessing advanced technologies, this system offers numerous benefits. It optimizes traffic flow, reducing congestion and travel times for all vehicles. The prioritization of emergency vehicles ensures faster response times and potentially life-saving interventions. However, challenges such as initial implementation costs, data privacy concerns, and the need for regular maintenance should be addressed. Overall, this intelligent system has the potential to revolutionize traffic management, making cities more efficient, safer, and responsive to the needs of commuters, pedestrians, and emergency services alike.

10.FUTURE SCOPE

1. **Advanced AI and Machine Learning:** Continued advancements in artificial intelligence and machine learning will enable the system to become even smarter and more efficient in analyzing and predicting traffic patterns. This will lead to more accurate congestion control measures and improved emergency vehicle prioritization.
2. **Connected and Autonomous Vehicles:** As connected and autonomous vehicles become more prevalent, the integration of these vehicles with the traffic management system will become crucial. The system can communicate with these vehicles, providing real-time traffic information, optimizing their routes, and ensuring smooth interaction between autonomous and human-driven vehicles.
3. **Smart Infrastructure:** The deployment of smart infrastructure components such as intelligent traffic lights, road sensors, and vehicle-to-infrastructure communication systems will enhance the capabilities of the traffic management system. These advancements will improve data collection, accuracy, and responsiveness, leading to more effective congestion control and emergency response.
4. **Integration with Smart City Initiatives:** The automated traffic management system can be integrated with broader smart city initiatives, including energy management, environmental sustainability, and public transportation planning. This integration will

create synergies and enable comprehensive urban management strategies that consider multiple aspects of urban life.

5. **Real-Time Predictive Analytics:** The system's data analysis capabilities can be further enhanced to provide real-time predictive analytics. By analyzing historical data, current traffic conditions, and external factors such as weather and events, the system can anticipate congestion and incidents, allowing for proactive measures to mitigate their impacts.
6. **Multi-Modal Transportation Integration:** The future system can integrate various transportation modes, including public transportation, bicycles, and pedestrians. By optimizing the flow and coordination between different modes, the system can provide seamless and efficient mobility options, reducing congestion and improving overall transportation accessibility.
7. **Sustainability and Environmental Considerations:** The future scope of the system includes incorporating environmental considerations into traffic management. By analyzing emissions data, promoting eco-friendly transportation options, and optimizing traffic patterns to reduce carbon footprint, the system can contribute to sustainable and eco-conscious urban development.

11.BIBLIOGRAPHY

- ❖ <https://aryaomnitalk.com/advanced-traffic-management-system-atms/>
- ❖ <https://www.parknsecure.com/automatic-traffic-management-system/>
- ❖ <https://www.hindawi.com/journals/wcmc/2020/8841893/>
- ❖ <https://highways.dot.gov/public-roads/winter-1995/congestion-control-and-demand-management>
- ❖ <https://www.sciencedirect.com/science/article/pii/S1474667017526683/pdf?md5=d284d17dadda5110758b1b58ea758132&pid=1-s2.0-S1474667017526683-main.pdf>
- ❖ <https://transportation.conduent.com/urban-congestion-management/>
- ❖ <https://ieeexplore.ieee.org/document/8916879https://www.ijert.org/accident-alert-system-including-traffic-sign-classification>
- ❖ <https://www.mdpi.com/2079-9292/11/4/510>
- ❖ <https://www.quickcompany.in/patents/automatic-ambulance-alert-system-for-traffic-hotspots>

APPENDIX

A. Source Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
int red=13;
int yellow=12;
int green=14;
int trigpin=15;
int echopin=2;

int red2=4;
int yellow2=5;
int green2=18;
int trigpin2=19;
int echopin2=21;

int btn=27;
int btn2=26;
String data3;
float dis,dis2;

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

//-----credentials of IBM Accounts-----

#define ORG "0aii6y"//IBM ORGANITION ID
#define DEVICE_TYPE "Traffic-management-system"//Device type mentioned in ibm
watson IOT Platform
#define DEVICE_ID "23456"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "23456789" //Token

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and
format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

//-----
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient);
void setup() {
    // put your setup code here, to run once:
```

```
Serial.begin(115200);
pinMode(trigpin,OUTPUT);
pinMode(echopin, INPUT);
pinMode(red, OUTPUT);
pinMode(yellow, OUTPUT);
pinMode(green, OUTPUT);

pinMode(trigpin2,OUTPUT);
pinMode(echopin2, INPUT);
pinMode(red2, OUTPUT);
pinMode(yellow2, OUTPUT);
pinMode(green2, OUTPUT);

pinMode(btn,INPUT_PULLUP);
pinMode(btn2,INPUT_PULLUP);

wificonnect();
mqttconnect();
}

void loop() {
  // put your main code here, to run repeatedly:
  // this speeds up the simulation
  int btnstat=digitalRead(btn);
  int btnstat2=digitalRead(btn2);

  if(btnstat==0){
    Serial.println("Emergency btn 1");
    digitalWrite(red, LOW);
    digitalWrite(yellow, LOW);
    digitalWrite(green, HIGH);
    digitalWrite(red2, HIGH);
    digitalWrite(yellow2, LOW);
    digitalWrite(green2, LOW);

    delay(2000);
    // digitalWrite(btn, LOW);
  }
  if(btnstat2==0){
    Serial.println("Emergency btn 2");
    digitalWrite(red, HIGH);
    digitalWrite(yellow, LOW);
    digitalWrite(green, LOW);
    digitalWrite(red2, LOW);
    digitalWrite(yellow2, LOW);
    digitalWrite(green2, HIGH);

    delay(2000);
    // digitalWrite(btn, LOW);
  }

  digitalWrite(trigpin, LOW);
  digitalWrite(trigpin,HIGH);
```

```
delayMicroseconds(10);  
digitalWrite(trigpin, LOW);  
float time=pulseIn(echopin,HIGH);  
float dis=time*0.034/2;
```

```
digitalWrite(trigpin2, LOW);  
digitalWrite(trigpin2,HIGH);  
delayMicroseconds(10);  
digitalWrite(trigpin2, LOW);  
float time2=pulseIn(echopin2,HIGH);  
float dis2=time2*0.034/2;
```

```
Serial.print("S1: ");  
Serial.print(dis);  
Serial.print(" ");  
Serial.print("S2: ");  
Serial.print(dis2);  
Serial.print(" ");
```

```
if(dis<200){  
    signal1Function();  
}  
if(dis2<200){  
    signal2Function();  
}  
else{  
    switchoff();  
}
```

```
PublishData(dis, dis2);  
delay(2000);  
if (!client.loop()) {  
    mqttconnect();  
}  
}
```

```
void signal1Function()  
{  
    Serial.println("1");  
    low();  
    // Make RED LED LOW and make Green HIGH for 5 seconds  
    digitalWrite(red, LOW);  
    digitalWrite(green, HIGH);  
    delay(5000);  
    // if there are vehicels at other signals  
    if(dis2<200)  
    {  
        // Make Green LED LOW and make yellow LED HIGH for 2 seconds  
        digitalWrite(green, LOW);  
        digitalWrite(yellow, HIGH);  
        delay(2000);  
    }  
}
```

```
void signal2Function()
{
  Serial.println("2");
  low();
  digitalWrite(red2, LOW);
  digitalWrite(green2, HIGH);
  delay(5000);

  if(dis<200)
  {
    digitalWrite(green2, LOW);
    digitalWrite(yellow2, HIGH);
    delay(2000);
  }
}

void low()
{

  digitalWrite(yellow, LOW);
  digitalWrite(green, LOW);
  digitalWrite(yellow2, LOW);
  digitalWrite(green2, LOW);

  digitalWrite(red, HIGH);
  digitalWrite(red2, HIGH);

}

void switchoff(){
  digitalWrite(red, LOW);
  digitalWrite(yellow, LOW);
  digitalWrite(green, LOW);
  digitalWrite(red2, LOW);
  digitalWrite(yellow2, LOW);
  digitalWrite(green2, LOW);
}

void PublishData(float dis, float dis2) {
  mqttconnect();//function call for connecting to ibm
  /*
    creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"dis\":";
  payload += dis;
  payload += "," "\"dis2\":";
  payload += dis2;
  payload += "}";

  Serial.print("Sending payload: ");
  Serial.println(payload);
}
```

```
if (client.publish(publishTopic, (char*) payload.c_str())) {  
    Serial.println("Publish ok");// if it successfully upload data on the cloud then it will print  
    publish ok in Serial monitor or else it will print publish failed  
} else {  
    Serial.println("Publish failed");  
}  
  
}
```

```
void mqttconnect() {  
    if (!client.connected()) {  
        Serial.print("Reconnecting client to ");  
        Serial.println(server);  
        while (!client.connect(clientId, authMethod, token)) {  
            Serial.print(".");  
            delay(500);  
        }  
    }  
  
    initManagedDevice();  
    Serial.println();  
}
```

```
void wificonnect() //function definition for wificonnect  
{  
    Serial.println();  
    Serial.print("Connecting to ");  
  
    WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the  
    connection  
    while (WiFi.status() != WL_CONNECTED) {  
        delay(500);  
        Serial.print(".");  
    }  
    Serial.println("");  
    Serial.println("WiFi connected");  
    Serial.println("IP address: ");  
    Serial.println(WiFi.localIP());  
}
```

```
void initManagedDevice() {  
    if (client.subscribe(subscribetopic)) {  
        Serial.println((subscribetopic));  
        Serial.println("subscribe to cmd OK");  
    } else {  
        Serial.println("subscribe to cmd FAILED");  
    }  
}
```

```
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)  
{  
  
    Serial.print("callback invoked for topic: ");
```

```

Serial.println(subscribetopic);
for (int i = 0; i < payloadLength; i++) {
  //Serial.print((char)payload[i]);
  data3 += (char)payload[i];
}
Serial.println("data: " + data3);
if(data3=="alerton")
{
Serial.println(data3);
Serial.println("Barricades opened");
}
else if(data3=="{"command":"alerton"}")
{
Serial.println(data3);
Serial.println("Barricades opened");
}
else
{
Serial.println(data3);
Serial.println("Barricades closed");
}
data3="";
}

```

B. Connections Code:

```

{
  "version": 1,
  "author": "NUKALA SRIYASETTY 20BCI7020",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 116.11, "left": 58.28, "attrs": {} },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": -60.86,
      "left": 32.62,
      "attrs": { "color": "red" }
    },
    {
      "type": "wokwi-led",
      "id": "led2",
      "top": -60.8,
      "left": 60.83,
      "attrs": { "color": "yellow" }
    },
    {
      "type": "wokwi-led",
      "id": "led3",
      "top": -60.35,
      "left": 91.03,

```



```
"attrs": { "color": "limegreen" }
},
{
  "type": "wokwi-resistor",
  "id": "r1",
  "top": 11.51,
  "left": -36.9,
  "attrs": { "value": "1000" }
},
{
  "type": "wokwi-hc-sr04",
  "id": "ultrasonic1",
  "top": -133.72,
  "left": -276.44,
  "attrs": { "distance": "232" }
},
{
  "type": "wokwi-led",
  "id": "led4",
  "top": -60.99,
  "left": 276.28,
  "attrs": { "color": "red" }
},
{
  "type": "wokwi-led",
  "id": "led5",
  "top": -61.94,
  "left": 310.01,
  "attrs": { "color": "yellow" }
},
{
  "type": "wokwi-led",
  "id": "led6",
  "top": -60.99,
  "left": 350.15,
  "attrs": { "color": "limegreen" }
},
{
  "type": "wokwi-resistor",
  "id": "r2",
  "top": 12.88,
  "left": 205.02,
  "attrs": { "value": "1000" }
},
{
  "type": "wokwi-hc-sr04",
  "id": "ultrasonic2",
  "top": -138.61,
  "left": 447.15,
  "attrs": { "distance": "339" }
},
{ "type": "wokwi-slide-switch", "id": "sw1", "top": -197.81, "left": -41.09, "attrs": {} },
{ "type": "wokwi-slide-switch", "id": "sw2", "top": -202.19, "left": 332.42, "attrs": {} }
```

```

],
"connections": [
  [ "esp:TX0", "$serialMonitor:RX", "", [] ],
  [ "esp:RX0", "$serialMonitor:TX", "", [] ],
  [ "led1:C", "r1:1", "black", [ "v0.84", "h-91.66" ] ],
  [ "led2:C", "r1:1", "black", [ "v12.75", "h-111.32" ] ],
  [ "led3:C", "r1:1", "black", [ "v19.99", "h-141.52" ] ],
  [ "r1:2", "esp:GND.2", "black", [ "v0" ] ],
  [ "led1:A", "esp:D13", "green", [ "v36.68", "h-28.73", "v239.05" ] ],
  [ "led2:A", "esp:D12", "green", [ "v60.05", "h-52.84", "v207.41" ] ],
  [ "led3:A", "esp:D14", "green", [ "v79.53", "h-78.94", "v177.53" ] ],
  [ "ultrasonic1:VCC", "esp:3V3", "red", [ "v0" ] ],
  [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ],
  [ "ultrasonic1:TRIG", "esp:D15", "green", [ "v0" ] ],
  [ "ultrasonic1:ECHO", "esp:D2", "green", [ "v0" ] ],
  [ "led4:C", "r2:1", "black", [ "v-2.32", "h-84.78" ] ],
  [ "led5:C", "r2:1", "black", [ "v17.06", "h-118.51" ] ],
  [ "led6:C", "r2:1", "black", [ "v21.94", "h-158.65" ] ],
  [ "r2:2", "esp:GND.1", "black", [ "v0" ] ],
  [ "led4:A", "esp:D4", "green", [ "v0" ] ],
  [ "led5:A", "esp:D5", "green", [ "v0" ] ],
  [ "led6:A", "esp:D18", "green", [ "v0" ] ],
  [ "ultrasonic2:VCC", "esp:3V3", "red", [ "v0" ] ],
  [ "ultrasonic2:GND", "esp:GND.1", "black", [ "v0" ] ],
  [ "ultrasonic2:TRIG", "esp:D19", "green", [ "v0" ] ],
  [ "ultrasonic2:ECHO", "esp:D21", "green", [ "v0" ] ],
  [ "sw1:2", "esp:GND.2", "black", [ "v34.86", "h-40.24", "v389.56" ] ],
  [ "sw1:3", "esp:D27", "green", [ "v54.35", "h-41.9", "v335.67" ] ],
  [ "sw2:2", "esp:GND.1", "black", [ "v73.81", "h45.31", "v335.18" ] ],
  [ "sw2:3", "esp:D26", "green", [ "v64.09", "h69.55", "v324.2" ] ],
],
"dependencies": {}
}

```