

# SMART TRAFFIC MANAGEMENT SYSTEM

## PHASE 3 – PROJECT

### TEAM MEMBERS

- 1) au820421106006 : ASWIN V
- 2) au820421106001 : AKSHAY KUMAR S
- 3) au820421106020 : JAYA SURYA V
- 4) au820421106022 : KALAIYARASAN C
- 5) au820421106003 : ARUN KUMAR K

### ABSTRACT:

Traffic is the major problem in so many cities of India and there are so many countries facing the same problem now a days . the problem of the traffic is the failure of the signal lights and the bad traffic management has to lead to traffic congestion .And how it is the high time to manage the traffic congestion problem . with the various methods we can control the traffic management system and there are wireless sensor network and inductive loop detection and video data and analysis and sensors and there are many more like these etc. but there is the only problem with this that was the it occurs too much of cost and it takes so much of time and the maintenance of the system is also very high rated .

A better result in the short period of time itself to overcome from these challenges a new method raises called Radio frequency . Identification (RFID). After applying this we can except that traffic will become less and the traffic will be monitored and management.

### INTRODUCTION:

In order to avoid the accident sand environment pollution, etc .Internet of Things is appears to be a new trend setter for intelligent traffic management due to advancement of data communication through internet, cloud utilization using various machine learning methodologies.

It will reduce the traffic tensions for civilians such as vehicle drivers , elderly peoples, ambulance , and shipping services.

This is the kind of intelligent traffic management system based on the IOT leads that to smart city management in the future.

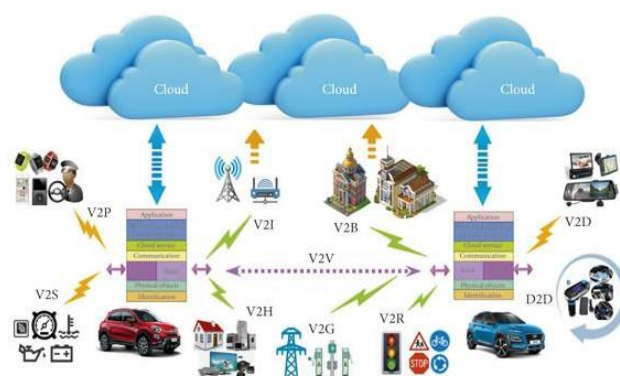
It includes an effective traffic information acquisition, suitable processing, analyzing various conditions and the categories of bulk traffic information in the crowded areas which takes to modern traffic management. applying this we can expect that traffic will become less and the traffic will be monitored and management.

## **PROPOSED SYSTEM:**

### Technologies to Implement ITMS:

The direct communication between vehicles is using an Ad Hoc network is compared to as intervehicle communication (IVC) communication types. Vehicle-to-Vehicle communication that will allow vehicles to exchange messages between them on the road. Vehicles can communicate with infrastructure deployed alongside the roads using Vehicle-to-Infrastructure communication. Each vehicle has Onboard Unit that similar to the vehicle computer with extra features allowing the services and layers of VANET. The infrastructure is a network of Roadside Units that is installed on the roadside.

The next generation of the VANET is referred as the Internet of Vehicles that will extends the functionality of VANET and inherits some many features of the Internet of Things . IOV involves Vehicle-to-Pedestrian, allowing the communication with vulnerable road users ,Vehicle-to-Sensor, on the inside of the vehicle ,Vehicle-to-Home, of the owner of the vehicle, Vehicle-to-Building, the surrounding buildings in the smart city. Vehicle-to-Grid, for electric charging, Vehicle-to-Device, for all the onboard devices, and Vehicle-to-Road signs.



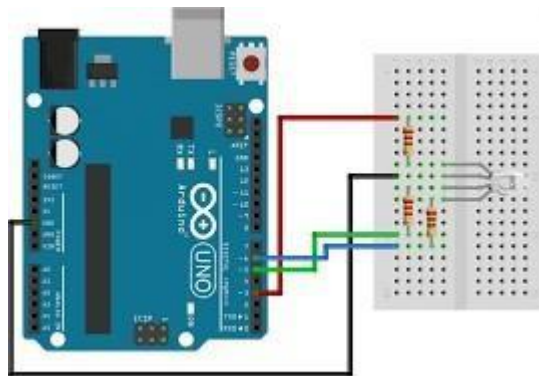
## DESIGN & DEVELOPMENT:

In method of collecting data for this project and I have refer so many sites to work on it . most of the information I have got it from articles and some of the web sites in chrome and journals about the topic of intelligent traffic management system.

## HARDWARE PARTS OF INTELLIGENT TRAFFIC MANAGEMENT SYSTEMS:

### 1. RGB Led:

RGB led is a combination of three LEDs one package: RGB-Red,Green,Blue.



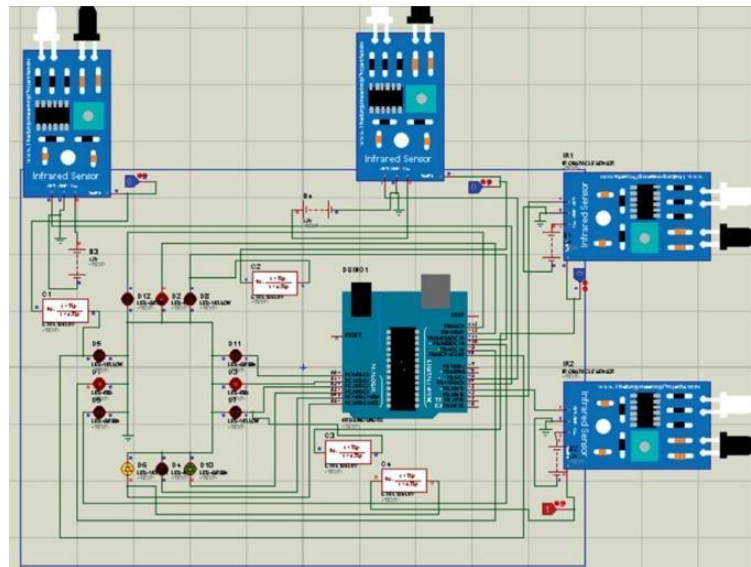
### 2. USB camera:

Universal Serial Bus (USB) it is a common platform that allows communication between devices & a host controller such as PC computer.



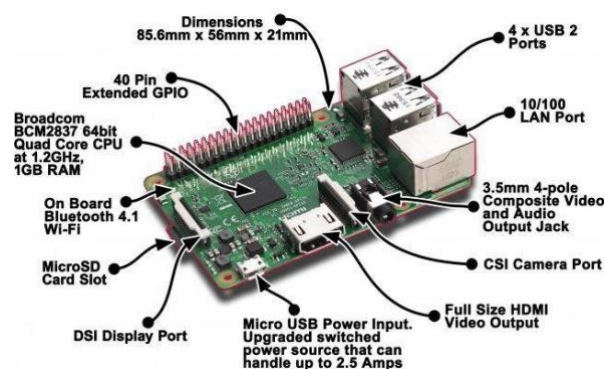
### **3. IR sensor:**

IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings.



### **4. RASPBERRY PI 3 B+:**

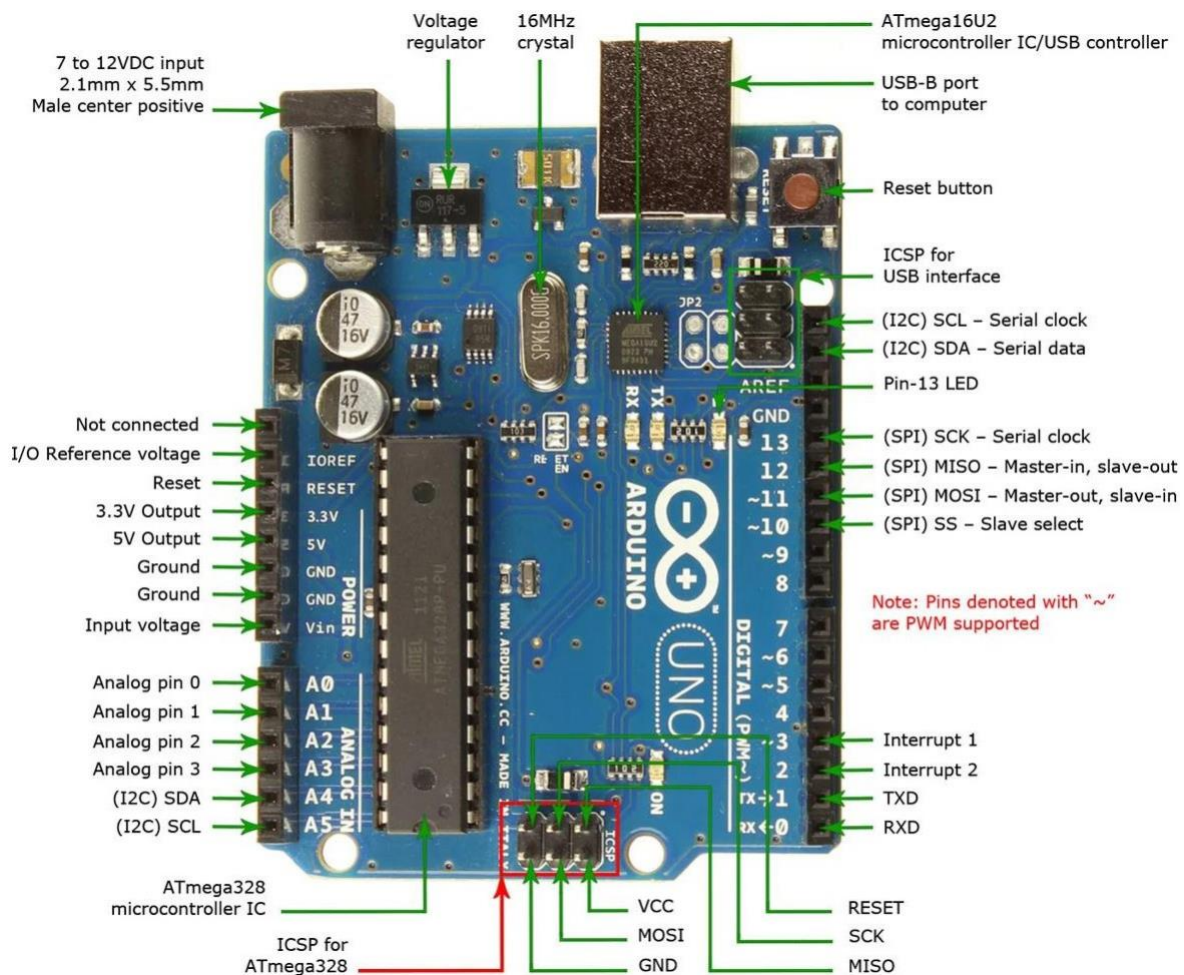
It is interfaced with IR sensor to detect the vehicles, camera to capture the vehicles on the road & the motor which is connected to IR sensor used to avoid the vehicles crossing the road during red light.



## 5. ARDUINO:

The led have been powered by Arduino UNO board. It contains a code which uploaded to the board. And once it simulated LED start's blinking like a traffic light. In this 15 seconds for **Red** light, 6 seconds for **Yellow** light, 20 seconds for **Green** light.

The Internet Of Things (IOT) technology facilitates direct communication between intelligent vehicles at intersection or other important touchpoints on roads.






## **6. TRAFFIC SIGNAL LIGHT:**

### Traffic Signals

#### Steady lights

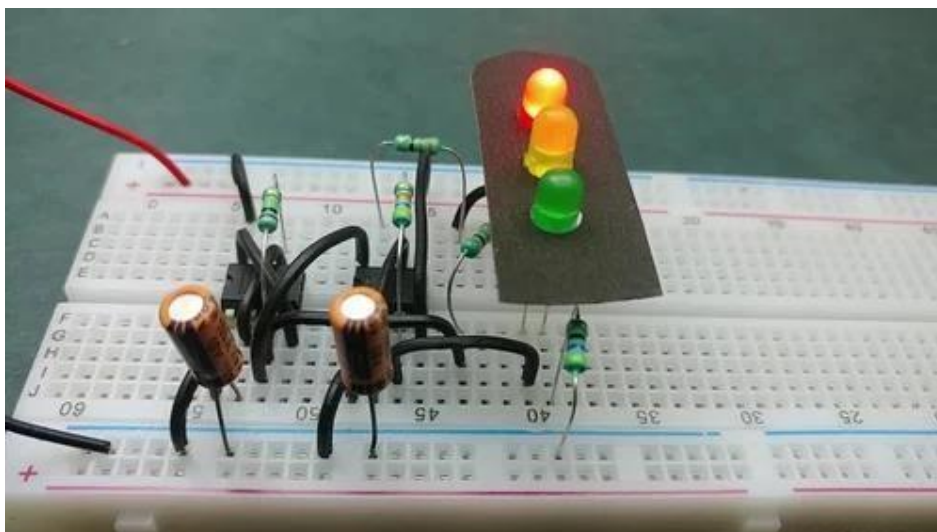
- **Red** –stop and remain stopped as long as the signal is red
- **Yellow** - a steady yellow light or arrow indicates that the light is about to change
  - Clear the intersection
- **Green** – go IF the way is clear



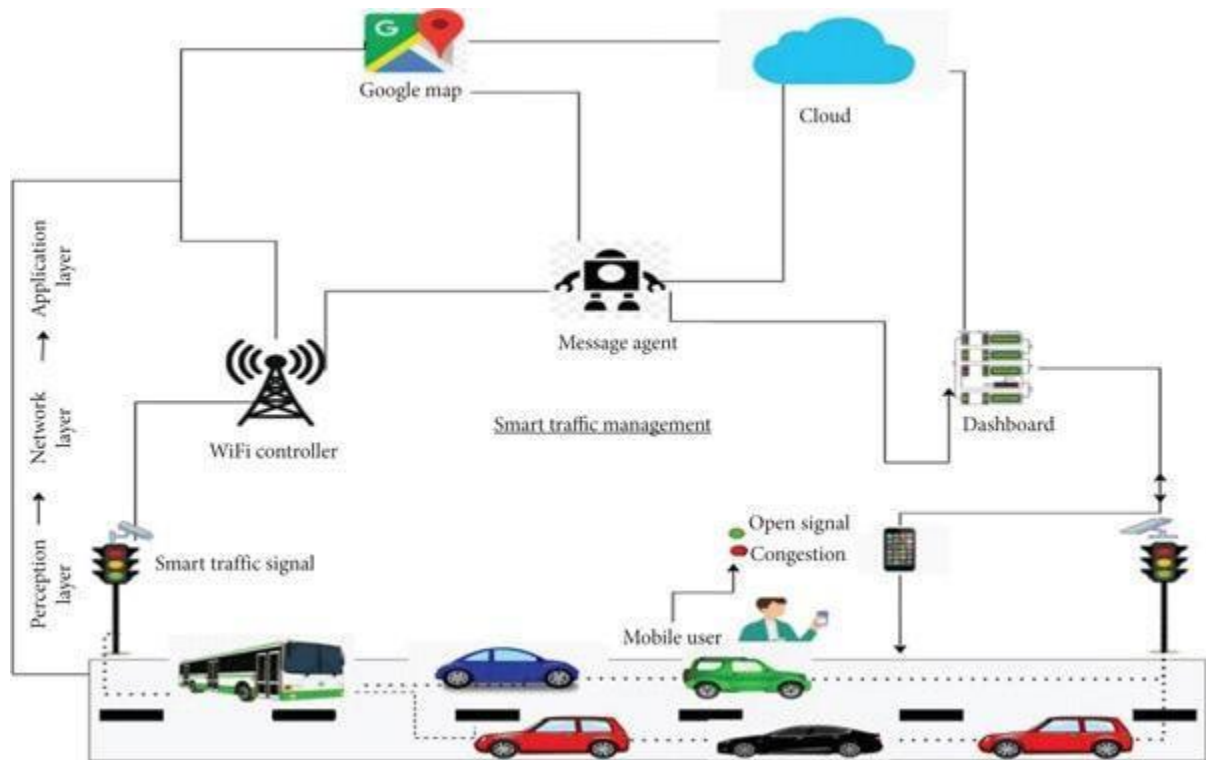
2

## **7. TRAFFIC LIGHT CIRCUITS WITH LED:**

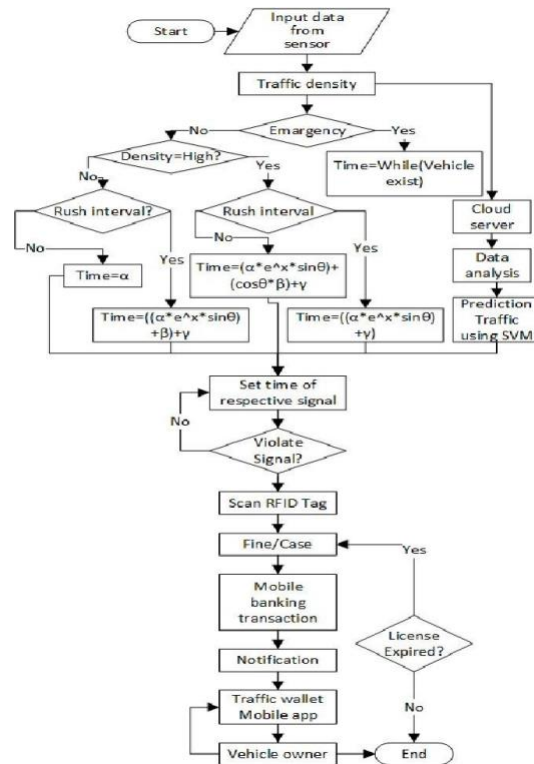
Traffic light circuit with LED in traffic management refers to the LED (light emitting diode) light used in traffic signals, commonly known as Traffic lights, and the associated electronic circuitry used to control & manage the operation of these lights.



## ARCHITECTURE:



## FLOW CHART:



## **HARDWARE SETUP:**

1. Connect the RGB LED to the Arduino. Typically, an RGB LED has four pins (common anode/cathode, and three color pins).
2. Connect the IR sensor to the Arduino to detect objects or motion.
3. Attach the USB camera to the Arduino, making sure it's compatible with the board.
4. Arrange the LEDs on a breadboard to simulate a traffic light (Red, Yellow, Green).

## **ARDUINO SETUP:**

1. Arduino sketch to control the RGB led based on input from the IR sensor.
2. Interface the USB camera with the Arduino.

## **SOURCE CODE:**

```
import RPi.GPIO as GPIO
import time

# Setup GPIO pins for traffic lights
red_pin = 17
yellow_pin = 18
green_pin = 27

GPIO.setmode(GPIO.BCM)
GPIO.setup(red_pin, GPIO.OUT)
GPIO.setup(yellow_pin, GPIO.OUT)
GPIO.setup(green_pin, GPIO.OUT)

# Function to control traffic lights
def control_traffic_lights():
    while True:
```



```
# Red light (stop)
GPIO.output(red_pin, GPIO.HIGH)
GPIO.output(yellow_pin, GPIO.LOW)
GPIO.output(green_pin, GPIO.LOW)
time.sleep(5) # Delay for red light

# Yellow light (prepare to stop)
GPIO.output(yellow_pin, GPIO.HIGH)
time.sleep(2) # Delay for yellow light

# Green light (go)
GPIO.output(red_pin, GPIO.LOW)
GPIO.output(yellow_pin, GPIO.LOW)
GPIO.output(green_pin, GPIO.HIGH)
time.sleep(5) # Delay for green light

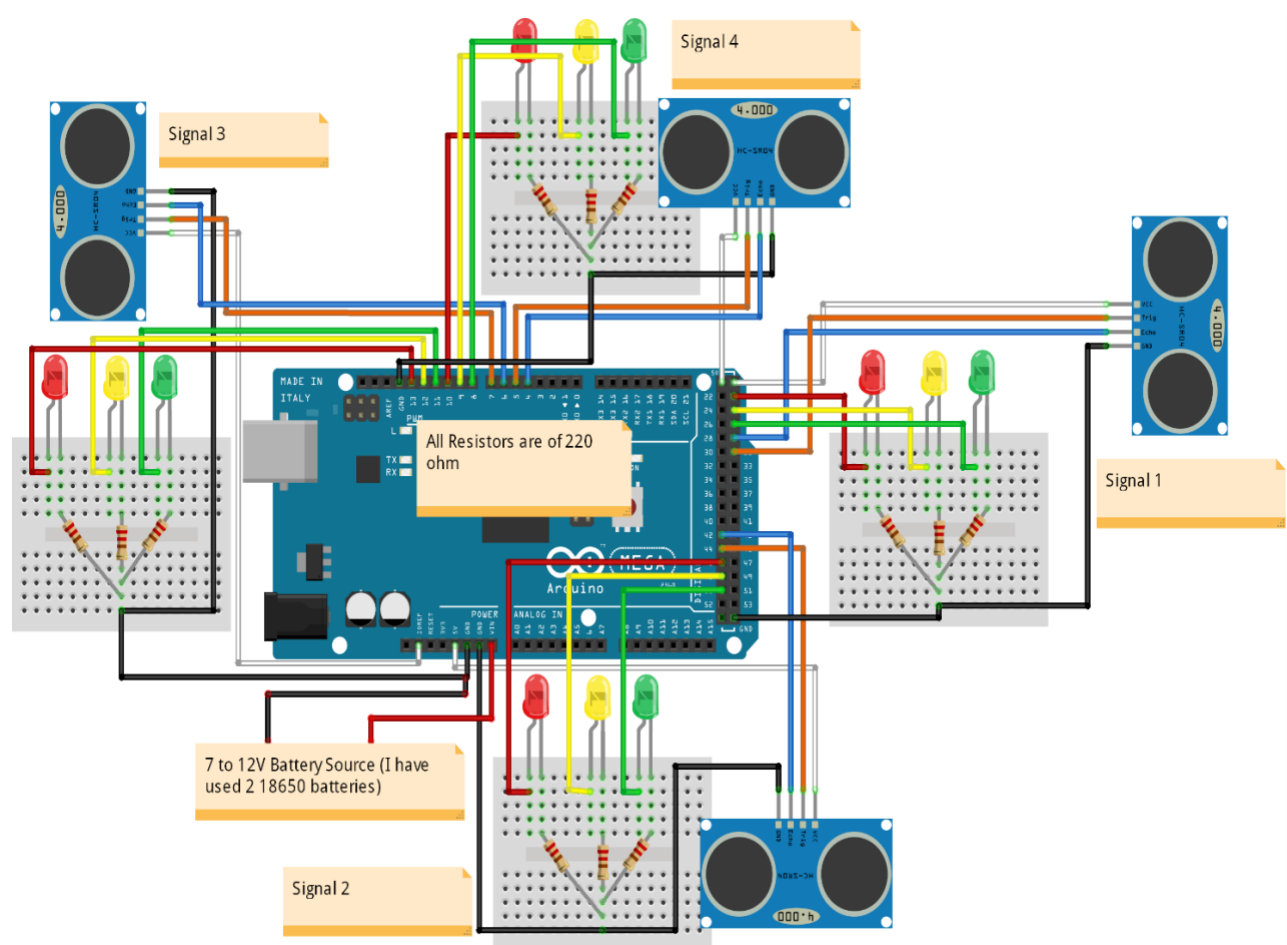
# Main loop
try:
    control_traffic_lights()
except KeyboardInterrupt:
    GPIO.cleanup()
```

This code assumes that you have set up your Raspberry Pi with the necessary hardware for controlling LED lights as traffic signals. You would need to connect the LEDs to the specified GPIO pins and make sure you have the RPi.GPIO library installed.

For a complete smart traffic management system, you would need additional components like sensors, cameras, and a more sophisticated control algorithm. You might also need a web interface or a communication system to interact with a centralized server or other traffic management devices. The code provided here is just a basic starting point for controlling traffic lights on a Raspberry Pi.

Building a smart traffic management system using sensors and a camera on a Raspberry Pi is a complex project. Here's a simplified example that uses a PIR motion sensor and a Pi Camera module to detect vehicles and control traffic lights. Please note that this is a basic illustration, and a real-world implementation would require more advanced hardware and software components.

### CIRCUIT DIAGRAM:



## Code Explanation:

First of all, we included the timerone library. This library is used to repetitively measure a period of time in microseconds and at the end of each period, an interrupt function will be called.

We have used this library because we want to read from the sensors and control LED's at the same time. We will have to use the delay in between the traffic signal so we can't read from the sensors continuously. Therefore we have used this library which will allow us to call a function in which we will read from the sensors continuously and in the loop function, we will control the traffic signals.

### **PYTHON CODE:**

```
#include<TimerOne.h>

int signal1[] = {23, 25, 27};
int signal2[] = {46, 48, 50};
int signal3[] = {13, 12, 11};
int signal4[] = {10, 9, 8};

int redDelay = 5000;
int yellowDelay = 2000;

volatile int triggerpin1 = 31;
volatile int echopin1 = 29;
volatile int triggerpin2 = 44;
volatile int echopin2 = 42;
volatile int triggerpin3 = 7;
volatile int echopin3 = 6;
volatile int triggerpin4 = 5;
volatile int echopin4 = 4;

volatile long time;
volatile int S1, S2, S3, S4;

int t = 5;

void setup()
{
```

```
Serial.begin(115200);
Timer1.initialize(100000);
Timer1.attachInterrupt(softInterr);
```

```
for(int i=0; i<3; i++){
    pinMode(signal1[i], OUTPUT);
    pinMode(signal2[i], OUTPUT);
    pinMode(signal3[i], OUTPUT);
    pinMode(signal4[i], OUTPUT);
}
```

```
pinMode(triggerpin1, OUTPUT);
pinMode(echopin1, INPUT);
pinMode(triggerpin2, OUTPUT);
pinMode(echopin2, INPUT);
pinMode(triggerpin3, OUTPUT);
pinMode(echopin3, INPUT);
pinMode(triggerpin4, OUTPUT);
pinMode(echopin4, INPUT);
}
```

```
void loop()
{
    // If there are vehicles at signal 1
    if(S1<t)
    {
        signal1Function();
    }

    // If there are vehicles at signal 2
    if(S2<t)
    {
        signal2Function();
    }

    // If there are vehicles at signal 3
    if(S3<t)
    {
        signal3Function();
    }
}
```

```

// If there are vehicles at signal 4
if(S4<t)
{
    signal4Function();
}
}

```

// This is interrupt **function** and it will run each time the timer period finishes. The timer period is set at 100 milli seconds.

```

void softInterr()
{
    // Reading from first ultrasonic sensor
    digitalWrite(triggerpin1, LOW);
    delayMicroseconds(2);
    digitalWrite(triggerpin1, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerpin1, LOW);
    time = pulseIn(echopin1, HIGH);
    S1= time*0.034/2;

    // Reading from second ultrasonic sensor
    digitalWrite(triggerpin2, LOW);
    delayMicroseconds(2);
    digitalWrite(triggerpin2, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerpin2, LOW);
    time = pulseIn(echopin2, HIGH);
    S2= time*0.034/2;

    // Reading from third ultrasonic sensor
    digitalWrite(triggerpin3, LOW);
    delayMicroseconds(2);
    digitalWrite(triggerpin3, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerpin3, LOW);
    time = pulseIn(echopin3, HIGH);
    S3= time*0.034/2;

    // Reading from fourth ultrasonic sensor
    digitalWrite(triggerpin4, LOW);

```

```

    delayMicroseconds(2);
    digitalWrite(triggerpin4, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerpin4, LOW);
    time = pulseIn(echopin4, HIGH);
    S4= time*0.034/2;

    // Print distance values on serial monitor for debugging
    Serial.print("S1: ");
    Serial.print(S1);
    Serial.print(" S2: ");
    Serial.print(S2);
    Serial.print(" S3: ");
    Serial.print(S3);
    Serial.print(" S4: ");
    Serial.println(S4);
}

void signal1Function()
{
    Serial.println("1");
    low();
    // Make RED LED LOW and make Green HIGH for 5 seconds
    digitalWrite(signal1[0], LOW);
    digitalWrite(signal1[2], HIGH);
    delay(redDelay);

    // if there are vehicels at other signals
    if(S2<t || S3<t || S4<t)
    {
        // Make Green LED LOW and make yellow LED HIGH for 2
seconds
        digitalWrite(signal1[2], LOW);
        digitalWrite(signal1[1], HIGH);
        delay(yellowDelay);
    }
}

void signal2Function()
{
    Serial.println("2");
    low();

```



```
digitalWrite(signal2[0], LOW);
digitalWrite(signal2[2], HIGH);
delay(redDelay);

if(S1<t || S3<t || S4<t)
{
    digitalWrite(signal2[2], LOW);
    digitalWrite(signal2[1], HIGH);
    delay(yellowDelay);
}
}

void signal3Function()
{
    Serial.println("3");
    low();
    digitalWrite(signal3[0], LOW);
    digitalWrite(signal3[2], HIGH);
    delay(redDelay);

    if(S1<t || S2<t || S4<t)
    {
        digitalWrite(signal3[2], LOW);
        digitalWrite(signal3[1], HIGH);
        delay(yellowDelay);
    }
}

void signal4Function()
{
    Serial.println("4");
    low();
    digitalWrite(signal4[0], LOW);
    digitalWrite(signal4[2], HIGH);
    delay(redDelay);

    if(S1<t || S2<t || S3<t)
    {
        digitalWrite(signal4[2], LOW);
        digitalWrite(signal4[1], HIGH);
        delay(yellowDelay);
    }
}
```

```

}

// Function to make all LED's LOW except RED one's.
void low()
{
    for(int i=1; i<3; i++)
    {
        digitalWrite(signal1[i], LOW);
        digitalWrite(signal2[i], LOW);
        digitalWrite(signal3[i], LOW);
        digitalWrite(signal4[i], LOW);
    }
    for(int i=0; i<1; i++)
    {
        digitalWrite(signal1[i], HIGH);
        digitalWrite(signal2[i], HIGH);
        digitalWrite(signal3[i], HIGH);
        digitalWrite(signal4[i], HIGH);
    }
}

```

## **CONCLUSION:**

This research of this project we have tested everything that we used in the project we have firstly checked the raspberry pi and IR sensor and then it is working and then we checked RGB led and USB camera by attaching them and it worked properly .

And in this intelligent traffic management system we have been used python programming language .we have tested on camera and it took photograph of the man and over speed when they don't follow the rules and then buy using the WIFI signals of the poll it send the picture to the control room and then the worker who are working there they will send them the picture and ask to pay fine . these is how the we tested the camera and every thing.