SMART TRAFFIC MANAGEMENT SYSTEM

Phase – 5

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 :

Smart traffic management systems using IoT are becoming increasingly popular in many countries around the world. These systems use a variety of techniques to identify traffic congestion, such as image processing, laser tracking, and inductive loop.

However, there are significant problems with the available methods, and hence, the proposed model makes use of infrared sensors, which play a constructive role in handling traffic. Infrared sensors are used to count the density of automobiles using which the signals are controlled by ESP8266-based Node MCU and the data is sent to the central cloud system.

This system can be interfaced with existing models and takes less time to install. The proposed system allocates a smart period for green lights on roads. To overcome the existing traffic problem, a solution of profile signal control system

called Autonomous and Real-Time signal control based on Estimation traffic demand for Minimization of Signal waiting time (ARTEMIS) has been proposed. ARTEMIS is a new signal-based control system that makes optimized signaling by predicting traffic flow from upstream intersections. Using the proposed system, traffic congestion can be reduced by 35%.

OBJECTIVES :

An IOT-based real-time traffic monitoring system is proposed for dynamic handling of traffic signals based on traffic density. The system provides a real-time dashboard to monitor the traffic updates, saving time expansion for reaching the proposed destination and preventing the loss of human life up to great extent. Infrared sensor, inductive loop detection, video data analysis, wireless sensor network, and other are used to manage the traffic smartly.

1.*Traffic Flow Optimization:* The primary objective is to optimize traffic flow by reducing congestion and minimizing traffic jams. This involves dynamically adjusting traffic signals, lane assignments, and speed limits based on real-time traffic data.

2.*Reduced Traffic Congestion:* Smart traffic management aims to reduce traffic congestion and bottlenecks, resulting in shorter commute times, decreased fuel consumption, and lower carbon emissions.

3.*Enhanced Safety:* Improve road safety by monitoring traffic conditions and quickly identifying and responding to accidents or hazardous road conditions. IoT devices can detect and report incidents in real-time, allowing for swift emergency responses.

4.*Real-Time Monitoring:* Continuously monitor traffic conditions and gather data from various IoT sensors, cameras, and devices to provide up-to-the-minute information to traffic management centers and commuters.

5.*Public Awareness and Education:* Use IoT technologies to educate the public about traffic rules, safety, and the benefits of the traffic management system.

6.Environmental Benefits: Reduce fuel consumption and greenhouse gas emissions by optimizing traffic patterns, reducing idling times, and promoting the use of public transportation and alternative modes of transportation.

7.Improved Public Transportation: Integrate traffic management with public transportation systems, ensuring buses and trains are in sync with traffic patterns to provide efficient service.
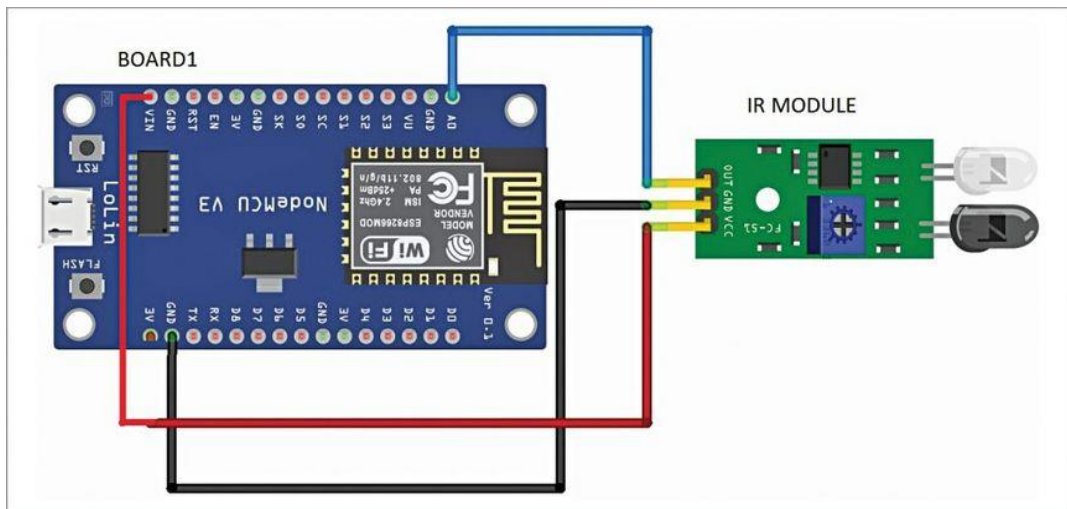
8.Sustainability: Encourage sustainable transportation choices by promoting walking, cycling, and electric or hybrid vehicles through traffic management policies.

COMPONENTS REQUIRED :

1. Raspberry Pi 3B+

2. IR Sensors

3. RGB LED

4. Traffic Light

5. USB Camera

6. Arduino ESP8266
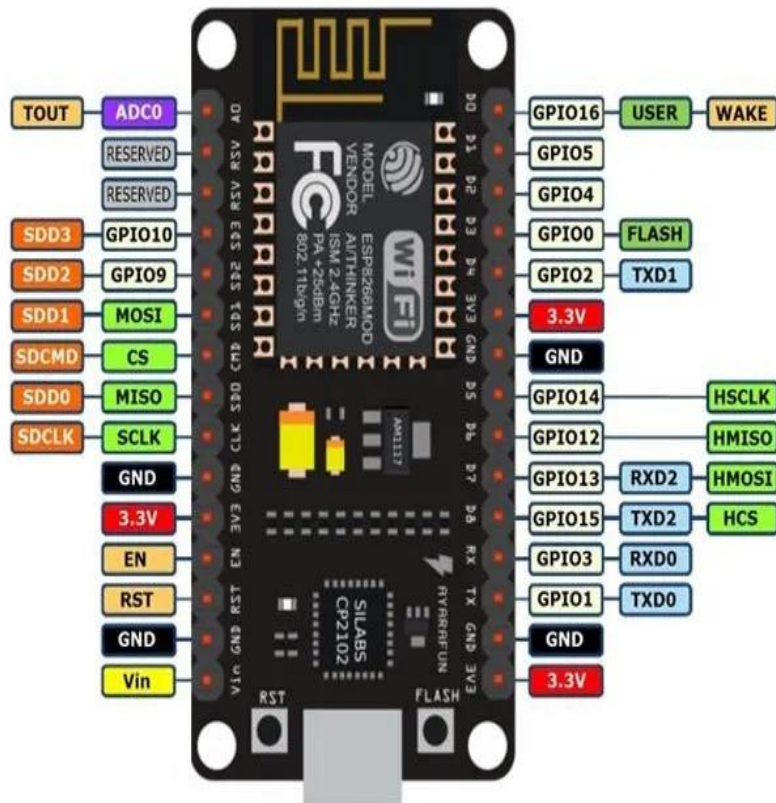
CIRCUIT DIAGRAM :
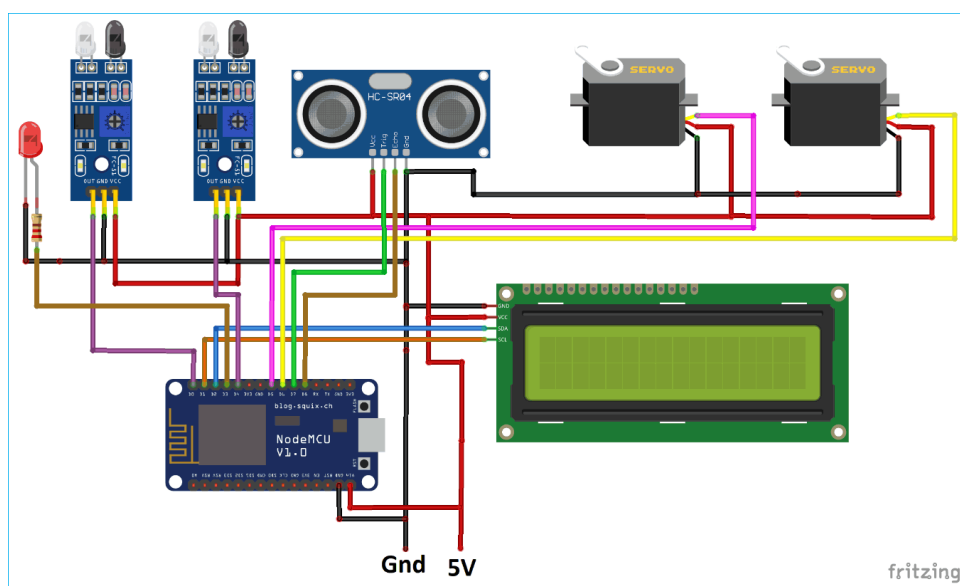
ESP8266 :

The ESP8266 uses a 32bit processor with 16 bit instructions. It is Harvard architecture which mostly means that instruction memory and data memory are completely separate.

The ESP8266 has on die program Read-Only Memory (ROM) which includes some library code and a first stage boot loader.

PIN—DIAGRAM :

Our ESP8266 has 17 GPIO pins but only 11 can be used (among 17 pins, 6 are used for communication with the on-board flash memory chip). It also has an analog input (to convert a voltage level into a digital value that can be stored and processed in the ESP8266).

IOT Sensor Setup :

After gathering all the information and main parts for the traffic signal management system here are three

components: traffic lights, queue detectors buried in the road and/or cameras, and a central control system. Main components in traffic management system

USB CAMERA :

USB cameras can be used in a traffic management system for various purposes, providing valuable visual data for monitoring, analysis, and control.

USES :

Traffic Monitoring: USB cameras are used to monitor traffic flow and congestion at key locations such as intersections, highways, and bridges. They provide live video feeds that can be analyzed to assess traffic conditions.

Traffic Surveillance: USB cameras can act as surveillance cameras to capture video footage of roadways. This footage can be useful in identifying accidents, traffic violations, or incidents for investigation and law enforcement.

License Plate Recognition (LPR): USB cameras equipped with LPR software can automatically read and recognize license plates on vehicles. This technology is useful for tracking vehicles, identifying stolen vehicles, or monitoring entry and exit points in parking lots and restricted areas.

Red-Light Camera Systems: USB cameras can be installed at traffic signal intersections to capture images and video of vehicles running red lights. These images are used to issue traffic violation tickets.

Traffic Control: USB cameras can be integrated into traffic control systems to provide visual data for intelligent traffic signal control. The cameras can detect the presence of vehicles at intersections and adjust signal timings accordingly.

RGB LED :

RGB (Red, Green, Blue) lights can be used in a traffic management system for various purposes, primarily for signaling and enhancing traffic safety. RGB lights, when used in conjunction with traditional traffic lights or other signaling devices, can provide additional information and improve communication with drivers and pedestrians.
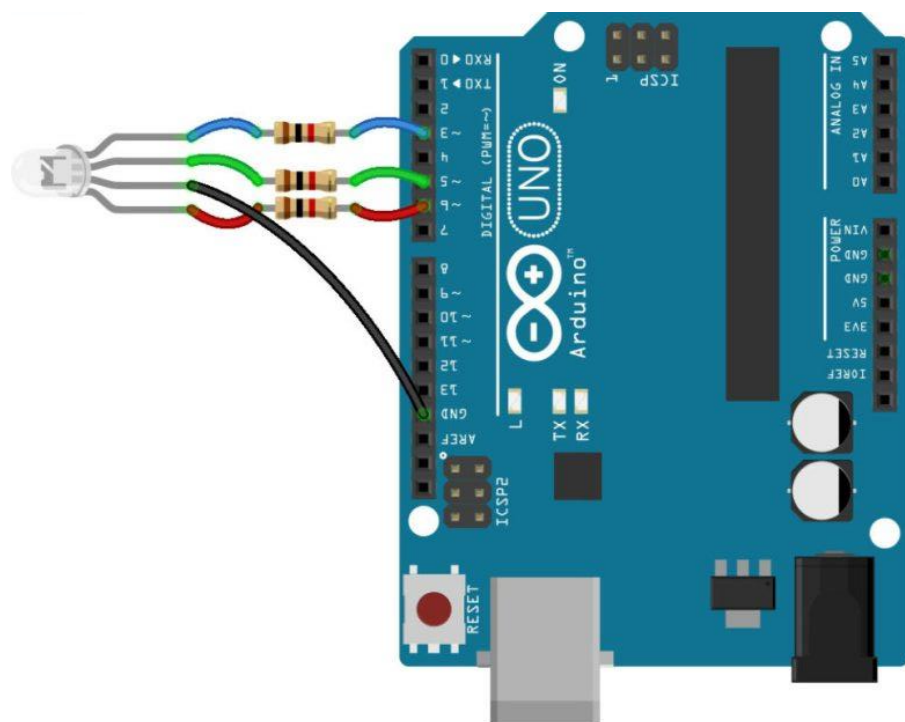
USES :
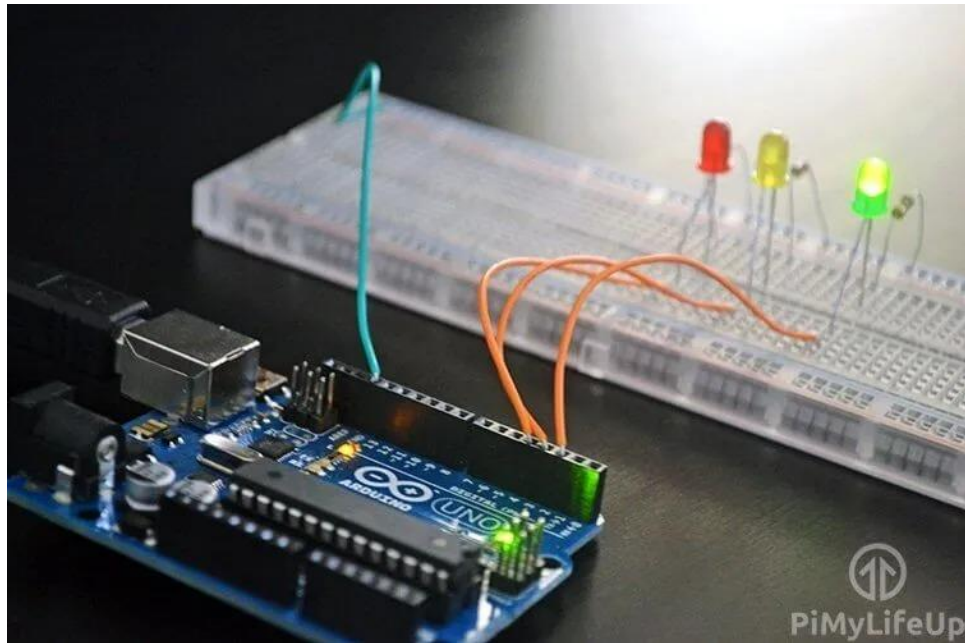
Emergency Vehicle Preemption: RGB lights can be used to signal the approach of an emergency vehicle. The lights can change to a specific color (e.g., blue) to alert other road users to yield the right-of-way and make way for the emergency vehicle.

Dynamic Lane Assignment: RGB lights can be incorporated into lane control systems to indicate when lanes are open or closed. For example, a green light can indicate that a lane is open for traffic, while a red light can signal lane closure.

Pedestrian Crosswalk Safety: In pedestrian crosswalks, RGB lights can enhance safety. They can change colors to signal pedestrians when it's safe to cross (green) and when it's not safe (red).

Traffic Density Information: RGB lights can provide real-time information about traffic density on a particular lane or road. For instance, a color code (e.g., green for light traffic, yellow for moderate, and red for heavy traffic) can be displayed.

Intersection Control: At complex intersections, RGB lights can provide additional signals to improve intersection control. They can indicate which lanes are reserved for specific types of vehicles, turning restrictions, or special conditions.
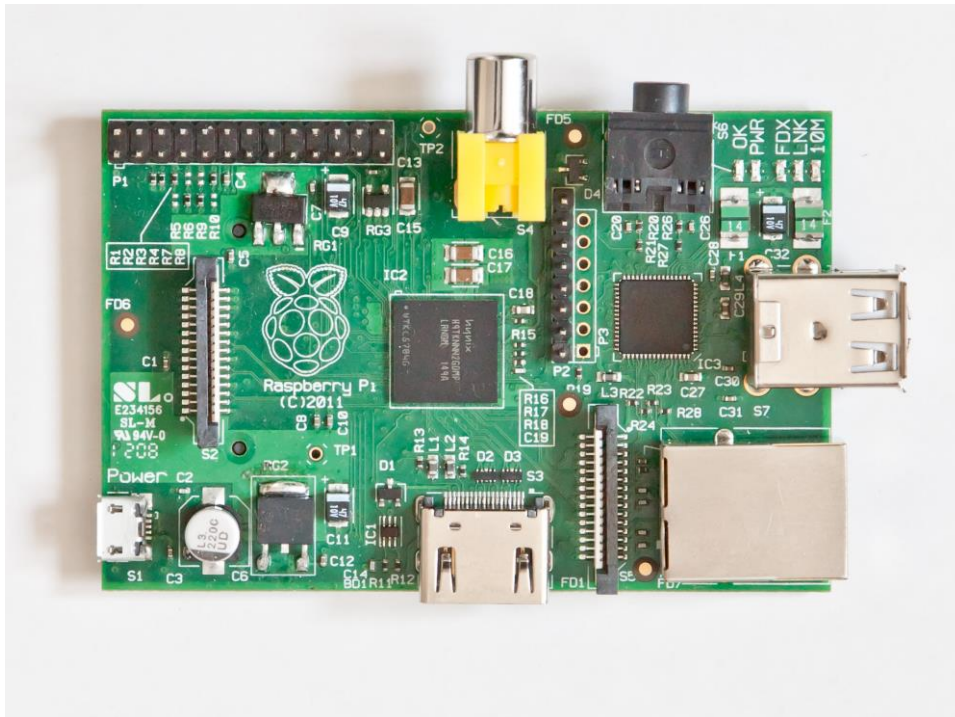
IR SENSOR :
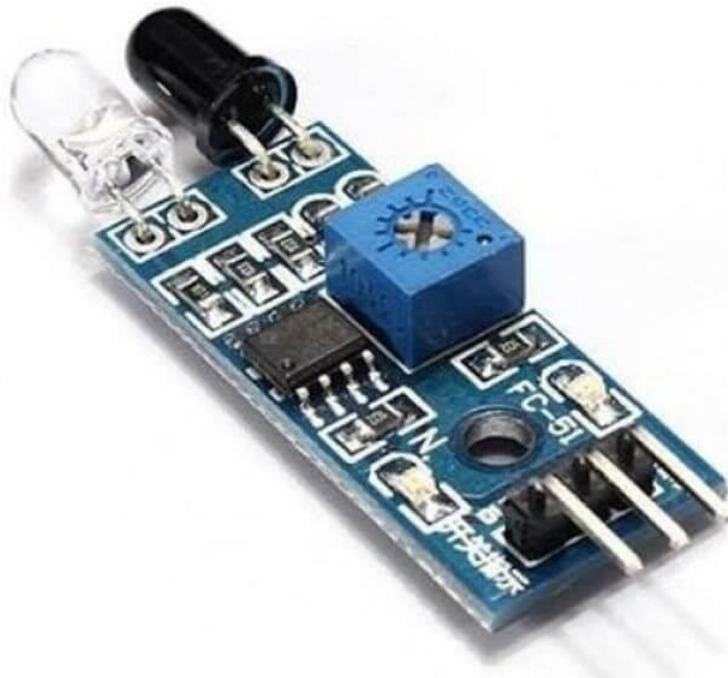
USES :

Vehicle Detection: IR sensors are commonly used to detect the presence of vehicles at traffic intersections and along roadways. When a vehicle passes over or through the sensor's detection area, it triggers the sensor to send a signal, indicating the presence of a vehicle.

Traffic Light Control: IR sensors are integrated into traffic light control systems to monitor the presence of vehicles waiting at an intersection. These sensors help in determining when to change traffic signal phases, making traffic flow more efficiently and reducing unnecessary waiting times.

Traffic Counting: IR sensors can be used to count the number of vehicles passing a specific point on the road. This data is valuable for traffic management and planning, allowing authorities to analyze traffic patterns and congestion.

RASPBERRY PI 3 B+ :

Raspberry Pi is a versatile, cost-effective, and popular single-board computer that can be used in various applications, including traffic management systems

USES :

Traffic Light Control: Raspberry Pi can be used to control traffic lights at intersections. It can receive real-time traffic data from sensors and adjust signal timings to optimize traffic flow and reduce congestion.

Vehicle Detection: Raspberry Pi can interface with various sensors (including IR sensors, cameras, or ultrasonic sensors) to detect the presence of vehicles at intersections or in parking areas. This data can be used for traffic monitoring and control.

Data Processing: Raspberry Pi is capable of processing and analyzing data from traffic sensors in real-time. It can perform tasks like vehicle counting, traffic flow analysis, and identifying traffic congestion.

Image and Video Processing: Raspberry Pi can process images and video feeds from cameras to monitor traffic conditions, recognize license plates, or detect traffic violations.

TRAFFIC LIGHT LED

CIRCUIT DIAGRAM :
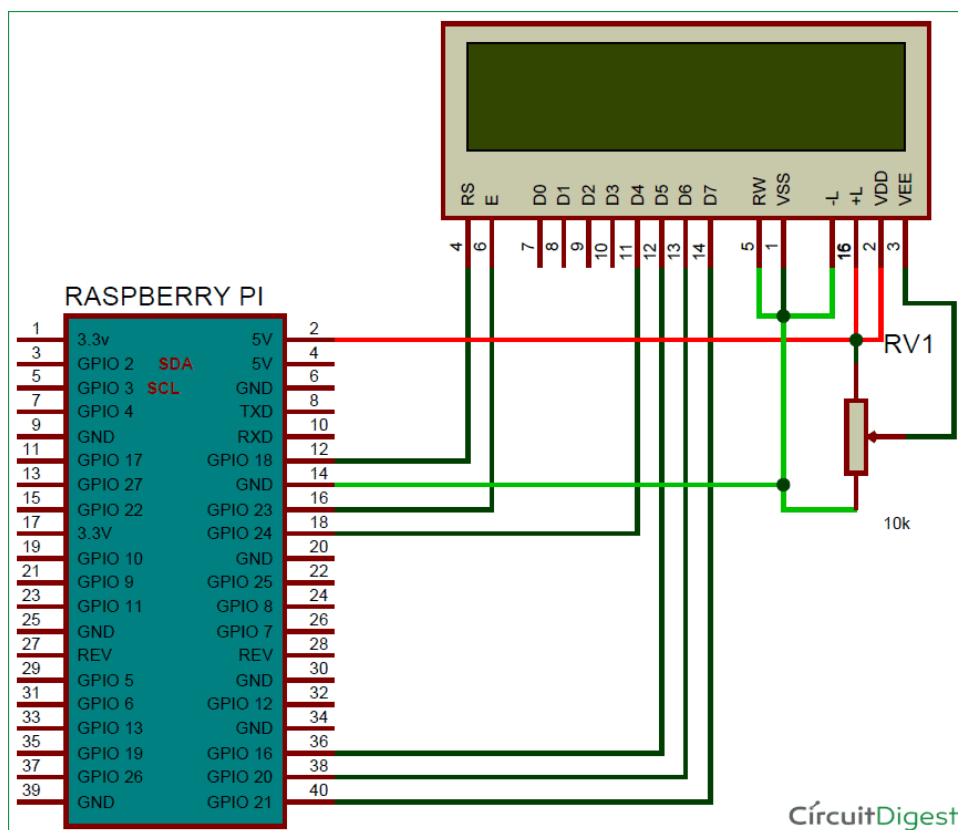
PROGRAM :

```
import RPi.GPIO as GPIO

import time
```

```python
# Set the GPIO mode to BCM
GPIO.setmode(GPIO.BCM)

# Define the GPIO pins for the traffic lights
RED_PIN = 17
YELLOW_PIN = 27
GREEN_PIN = 22

# Setup the GPIO pins as outputs
GPIO.setup(RED_PIN, GPIO.OUT)
GPIO.setup(YELLOW_PIN, GPIO.OUT)
GPIO.setup(GREEN_PIN, GPIO.OUT)

# Function to turn on the red light
def turn_on_red():
    GPIO.output(RED_PIN, GPIO.HIGH)
    GPIO.output(YELLOW_PIN, GPIO.LOW)
    GPIO.output(GREEN_PIN, GPIO.LOW)

# Function to turn on the yellow light
def turn_on_yellow():
    GPIO.output(RED_PIN, GPIO.LOW)
    GPIO.output(YELLOW_PIN, GPIO.HIGH)
    GPIO.output(GREEN_PIN, GPIO.LOW)
```

```python
# Function to turn on the green light
def turn_on_green():
    GPIO.output(RED_PIN, GPIO.LOW)
    GPIO.output(YELLOW_PIN, GPIO.LOW)
    GPIO.output(GREEN_PIN, GPIO.HIGH)

try:
    while True:
        # Start with the red light
        turn_on_red()
        time.sleep(5)  # Red light for 5 seconds

        # Switch to yellow for 2 seconds
        turn_on_yellow()
        time.sleep(2)

        # Finally, turn on the green light for 5 seconds
        turn_on_green()
        time.sleep(5)

except KeyboardInterrupt:
    # Cleanup when the program is interrupted
    GPIO.cleanup()
```

VEHICLE DETECTION IN IOT :

Raspberry Pi can interface with various sensors (including IR sensors, cameras, or ultrasonic sensors) to detect the presence of vehicles at intersections or in parking areas. This data can be used for traffic monitoring and control.

In a smart traffic system, vehicle detection is a critical component for various applications, including traffic monitoring, traffic flow optimization, security, and incident detection. There are different methods and technologies to detect vehicles, and the choice depends on the specific requirements of your system.

PROGRAM :

Creating a vehicle detection program using a Raspberry Pi for a traffic management system involves using hardware such as IR sensors, cameras, or ultrasonic sensors. Below is a simplified Python program using OpenCV, a popular computer vision library, to detect vehicles in a video stream from a camera connected to a Raspberry Pi. This program assumes you have OpenCV installed and a camera connected to your Raspberry Pi

```python
import cv2

import numpy as np


# Load the pre-trained vehicle detection model

vehicle_cascade = cv2.CascadeClassifier('haarcascade_car.xml')


# Open a video stream from a camera (0 for the default camera)

cap = cv2.VideoCapture(0)


while True:
    ret, frame = cap.read()

    if not ret:
        break

    # Convert the frame to grayscale for vehicle detection
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Detect vehicles in the frame
    vehicles = vehicle_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))
```

```
    # Draw rectangles around detected vehicles

    for (x, y, w, h) in vehicles:

        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)


    # Display the frame with detected vehicles

    cv2.imshow('Vehicle Detection', frame)


    # Exit the program on 'q' key press

    if cv2.waitKey(1) & 0xFF == ord('q'):

        break
  # Release the video stream and close OpenCV windows

   cap.release()

   cv2.destroyAllWindows()
```

This program captures video from the camera, applies background subtraction to detect moving objects (vehicles), and draws bounding boxes around them. To use this code effectively in a real traffic management system, you may need to fine-tune the parameters, adjust the detection logic, and integrate it with other components of your system, such as data storage and communication with a central server. Additionally, consider using more advanced object detection techniques, like YOLO or SSD, for better accuracy in complex traffic scenarios.

CIRCUIT :
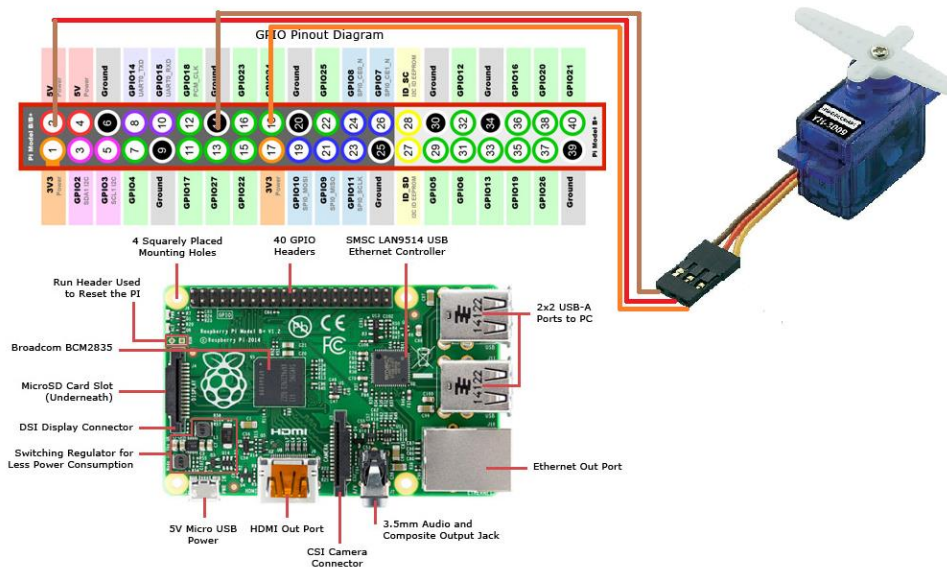
PROS :

Improved Traffic Flow: Vehicle detection systems help optimize traffic flow by adjusting traffic signals, lane assignments, and signal timings in real-time based on the actual presence and movement of vehicles. This reduces congestion and minimizes traffic jams.

Reduced Congestion: By actively managing traffic signals and flow, vehicle detection systems reduce traffic congestion and bottlenecks, resulting in shorter travel times and reduced fuel consumption.

Enhanced Safety: Vehicle detection systems contribute to road safety by monitoring traffic conditions and quickly identifying and responding to accidents, breakdowns, or other hazardous situations. They can trigger timely alerts and emergency responses.

Real-time Monitoring: These systems provide real-time monitoring of traffic conditions, allowing traffic management centers to respond promptly to changing situations, such as accidents, road closures, and adverse weather conditions.

Data-Driven Decision Making: Vehicle detection systems collect valuable traffic data, which can be used for data-driven decision making. This data is instrumental in implementing effective traffic policies, planning infrastructure improvements, and adapting to changing traffic patterns.

Public Transportation Integration: Vehicle detection can be integrated with public transportation systems to ensure buses and trains are in sync with traffic conditions, providing more efficient and reliable service.

Efficient Traffic Signaling: Adaptive traffic signal systems can be implemented, which adjust signal timings based on real-time traffic data. This minimizes waiting times at traffic signals and reduces fuel consumption.

VEHICLE SPEED MEASURING IN RASPBERRY PI

Measuring vehicle speed in a Raspberry Pi-based traffic management system can be achieved using various methods, such as cameras, ultrasonic sensors, or radar sensors. Below, I'll outline a basic example using a Raspberry Pi and a camera with OpenCV to calculate the speed of vehicles based on image analysis:

Prerequisites:

1. Raspberry Pi with Raspbian OS installed.
2. Camera module connected to the Raspberry Pi.
3. OpenCV library installed on the Raspberry Pi (you can install it using pip install opencv-python).

CIRCUIT :
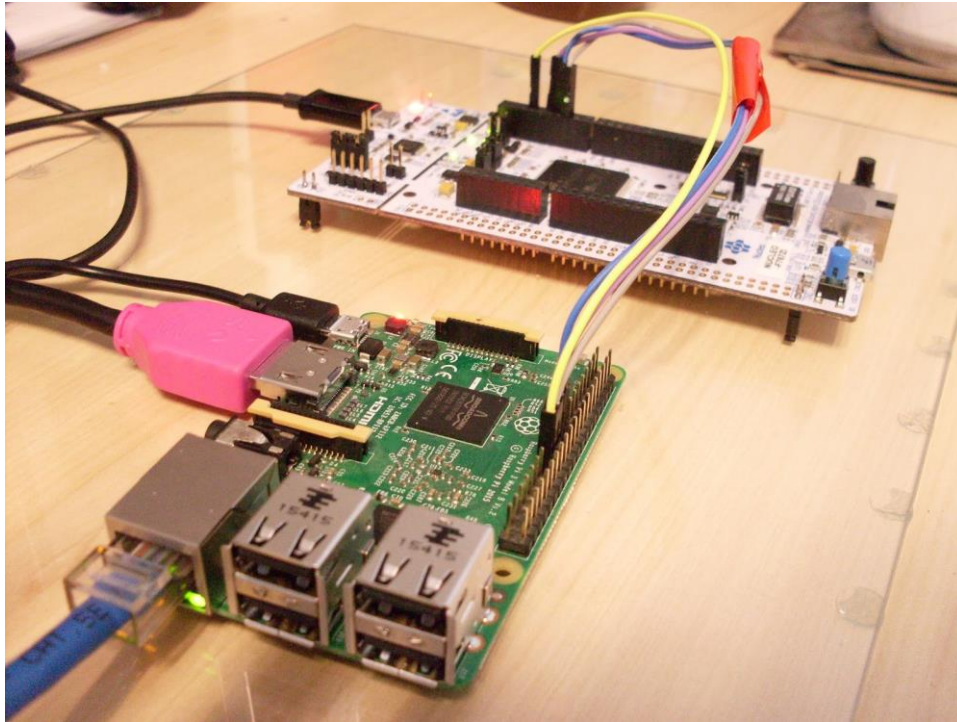
PROGRAM :

```
import cv2

import numpy as np

import time


# Initialize the camera

cap = cv2.VideoCapture(0)  # Use 0 for the default camera


# Parameters for speed calculation

fps = 30  # Frames per second

pixel_to_meter = 0.1  # Conversion factor (adjust as needed)

distance_between_points = 5  # Distance between the two points (in meters)

previous_frame = None

previous_time = time.time()
```

```python
while True:
    ret, frame = cap.read()
    if not ret:
        break

    # Convert the frame to grayscale for processing
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    if previous_frame is not None:
        # Calculate optical flow using Lucas-Kanade method
        flow = cv2.calcOpticalFlowPyrLK(
            previous_frame, gray, None, None, winSize=(15, 15), maxLevel=2
        )

        # Filter out points that have moved
        good_new = flow[:, 0]
        good_old = flow[:, 1]

        # Calculate speed based on the points that moved
        speeds = np.linalg.norm(good_new - good_old, axis=1) * pixel_to_meter / (time.time() - previous_time)
        average_speed = np.mean(speeds)

        # Check if a vehicle has passed the measurement point
        if average_speed > 0:
            print(f"Vehicle speed: {average_speed} m/s")
```

```
    previous_frame = gray

    previous_time = time.time()


    # Display the video stream with speed info

    cv2.putText(frame, f"Speed: {average_speed:.2f} m/s", (10, 50),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)

    cv2.imshow("Traffic Management System", frame)


    if cv2.waitKey(1) & 0xFF == 27:

        break


cap.release()

cv2.destroyAllWindows()
```

In this code, we capture video from the camera, calculate the optical flow between consecutive frames to track vehicle movement, and estimate vehicle speed. You may need to adjust the pixel_to_meter value based on your camera's setup and measurement needs. Additionally, you should set up your camera to point at the road segment you want to monitor.

PROS :

Safety Enhancement: Vehicle speed monitoring helps in enforcing speed limits, reducing the risk of accidents, and improving overall road safety. It deters drivers from exceeding safe speeds and contributes to accident prevention.

Accident Detection: Real-time monitoring of vehicle speeds allows for the rapid detection of speeding incidents and accidents. This enables immediate emergency responses and quicker clearance of accident scenes.

Traffic Flow Optimization: By collecting data on vehicle speeds, traffic management systems can adjust traffic signal timings and optimize traffic flow, reducing congestion and enhancing the overall efficiency of road networks.

Data for Decision-Making: Vehicle speed data is valuable for traffic engineers and city planners. It provides insights into traffic patterns, congestion points, and the impact of speed on overall traffic conditions, aiding in data-driven decision-making for infrastructure improvements.

Compliance Monitoring: Vehicle speed monitoring helps in enforcing speed limits and identifying drivers who violate these limits. Automated enforcement, such as speed cameras, can issue tickets to violators.

CONNECTING TO MOBILE APPLICATION (BLYNK APK) :
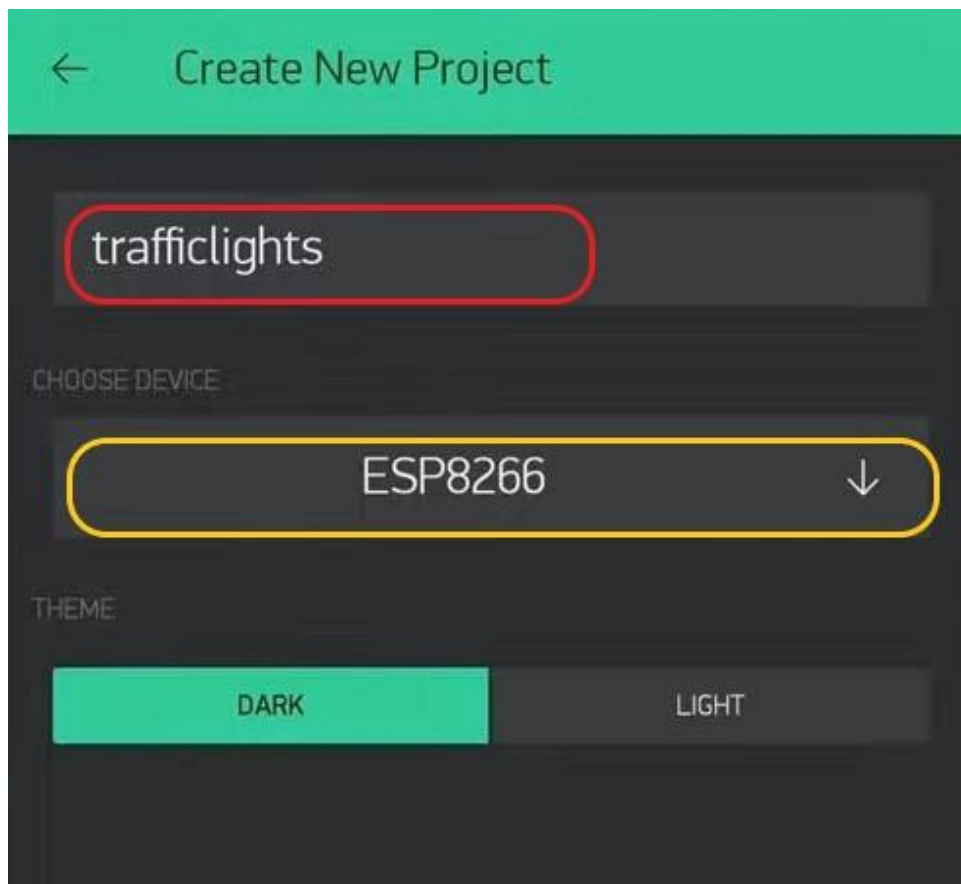
Blynk is an IoT software platform that provides infrastructure for the internet of Things. It supports 400+ development boards, SBC's, and other modules. It allows users to build and manage connected products with no code, and to remotely control, monitor, and automate them with mobile and web applications. It also offers secure cloud, data analytics, user and access management, alerts, and Over-The-Air firmware updates

1 . To create a device using Blynk, you can manually create a device using Blynk.Console for initial prototyping (works for any hardware) or use Static Tokens for cellular, Ethernet, and other non-WiFi connection methods. Activating devices with manually generated AuthTokens is recommended for prototyping stages or when you build a device for yourself.

• First we need to install the Blynk app from PlayStore and create an account.

 • Click on the create button and create your new project
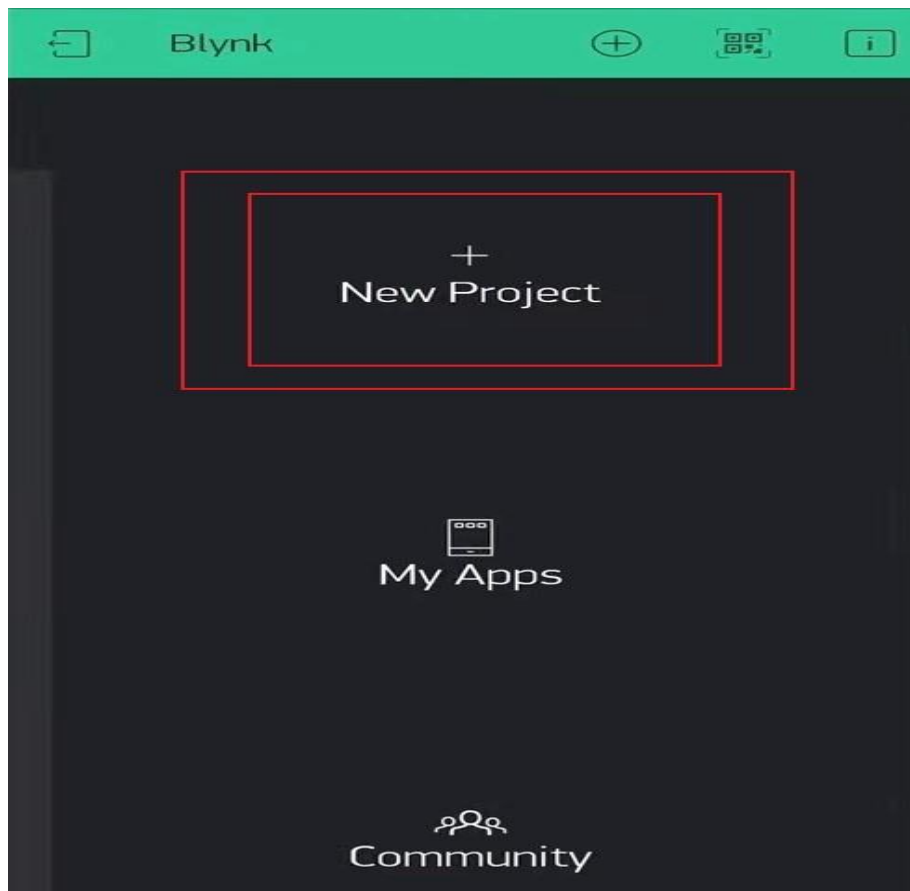
•Give your project a name and choose the board as NodeMCU and connection type as Wi-Fi.
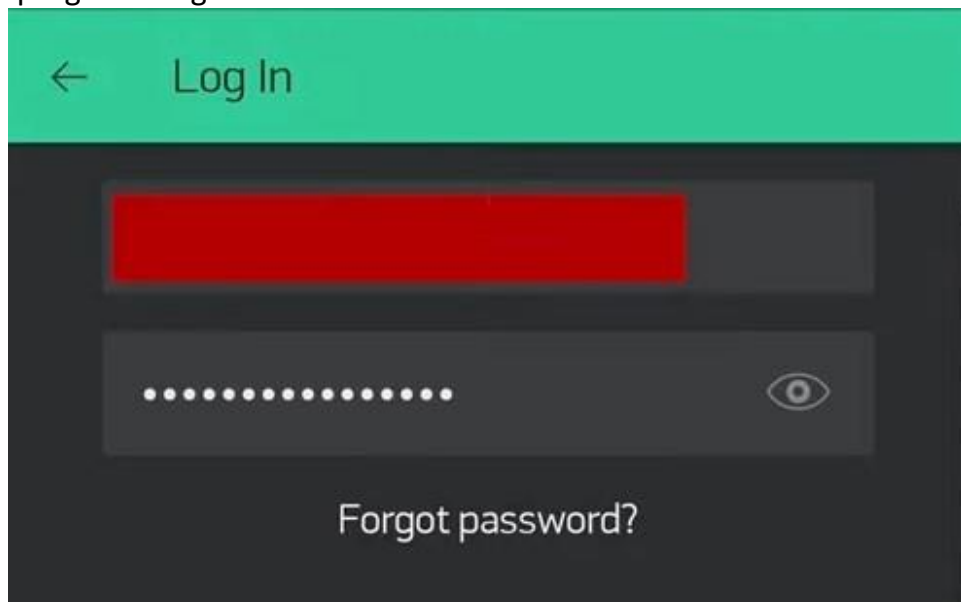
• An auth token will be sent to your registered email id. Keep it safe as it will be used later on while programming.

My Devices

trafficlights

HARDWARE MODEL

NodeMCU ↓

CONNECTION TYPE

Wi-Fi ↓

AUTH TOKEN

****************************BuiF

Refresh   Email

Widget Box

+

Displays

3.141	Value Display		ⓘ
25°C	Labeled Value	UPGRADE	ⓘ
LED icon	LED		ⓘ
Gauge icon	Gauge		ⓘ
Radial Gauge icon	Radial Gauge	UPGRADE	ⓘ
Enhanced Gauge icon	Enhanced Gauge	UPGRADE	ⓘ
LCD icon	LCD	UPGRADE	ⓘ
SuperChart icon	SuperChart		ⓘ
SimpleChart icon	SimpleChart	UPGRADE	ⓘ
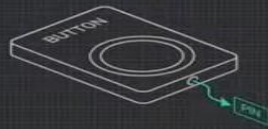
Button

OUTPUT

D0	0	1

MODE

PUSH ⬤ SWITCH

ON/OFF LABELS

OFF   ON

OFF	ON

DESIGN

FONT SIZE   TEXT
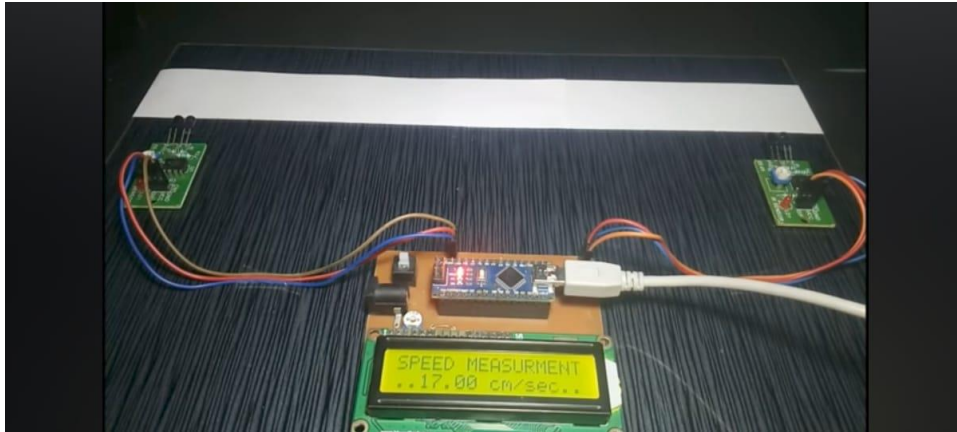
T **T** ᴛ   ⬤

[×] Delete
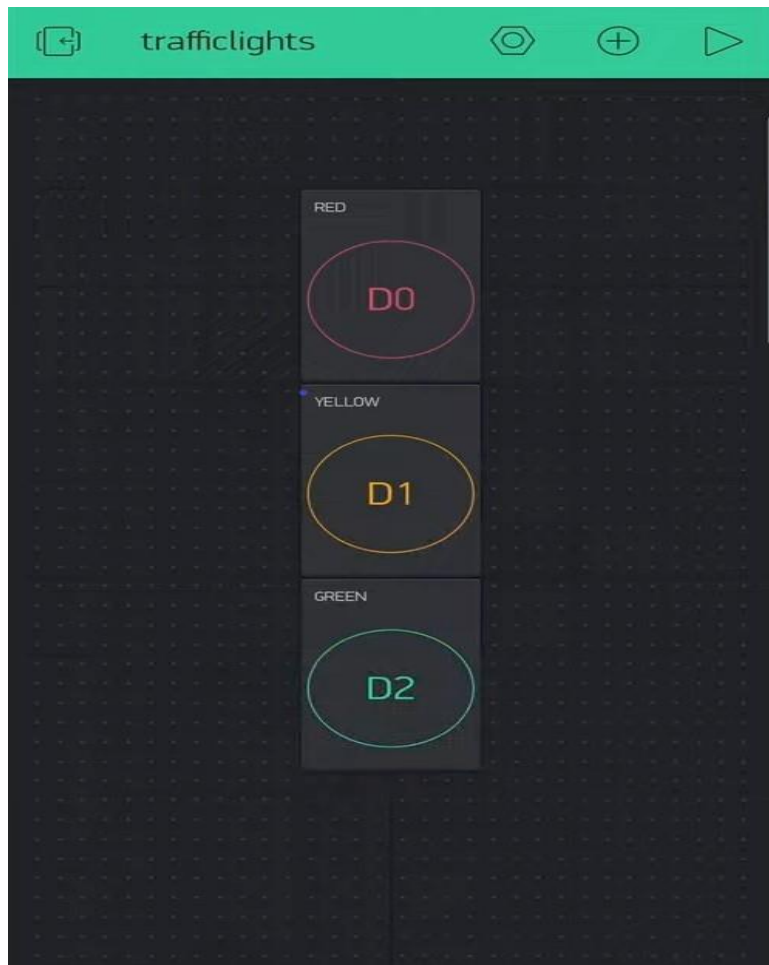
MOBILE APP OUTPUT :
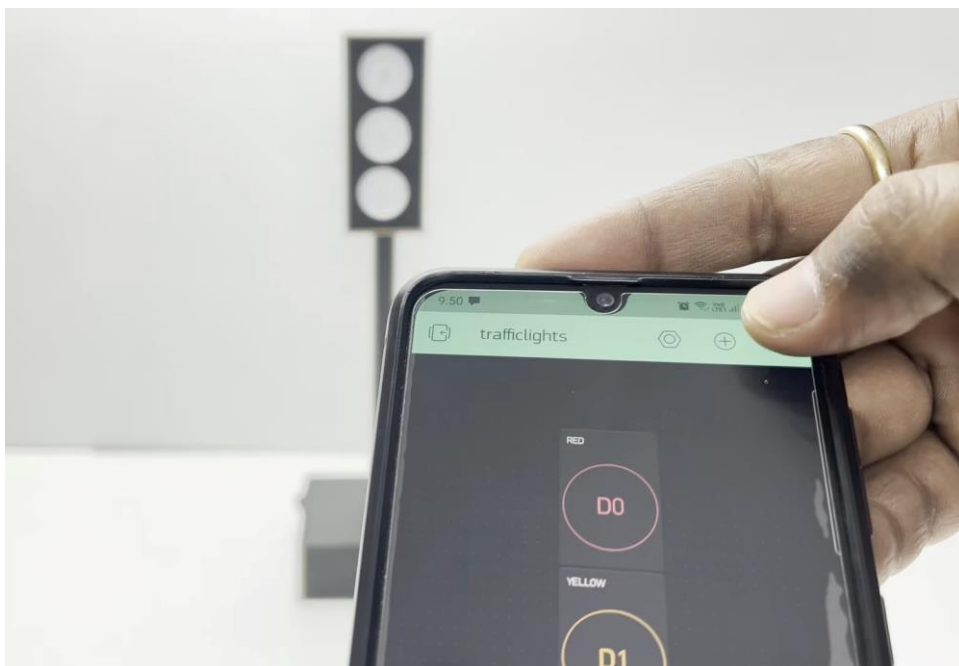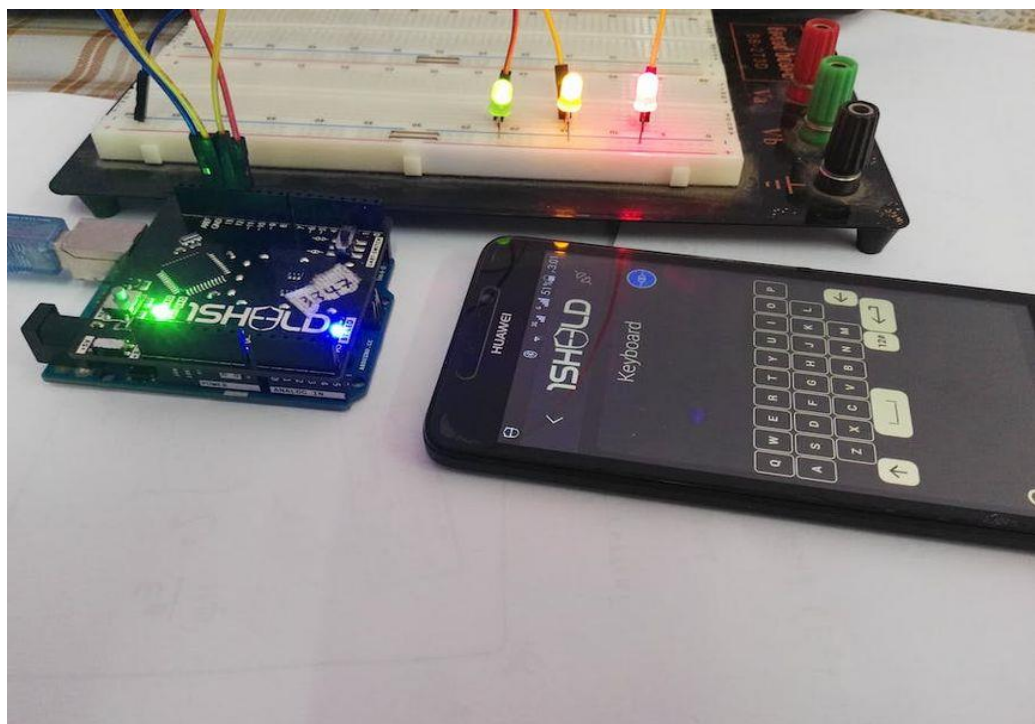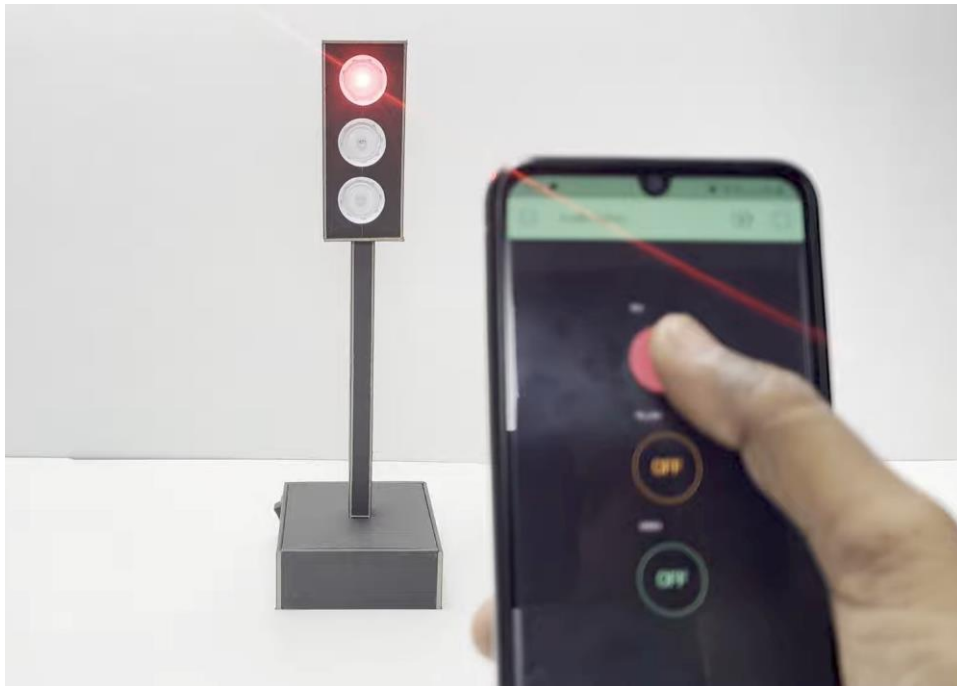
VEHICLE SPEED DETECTOR :
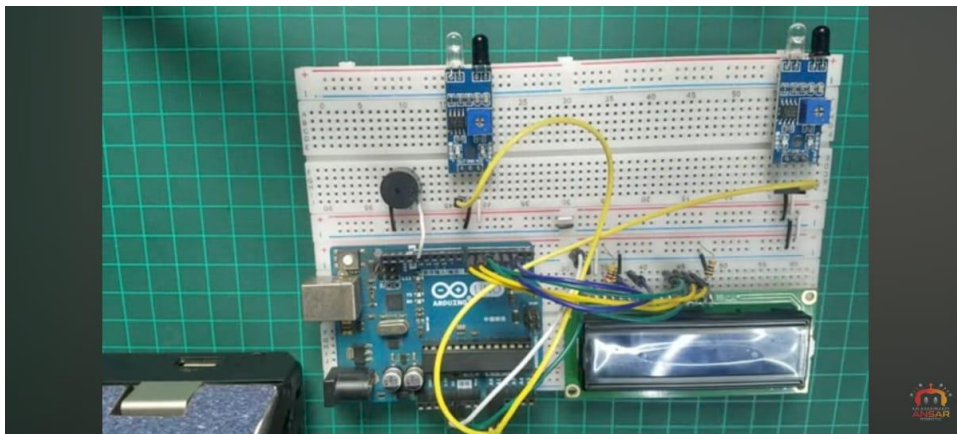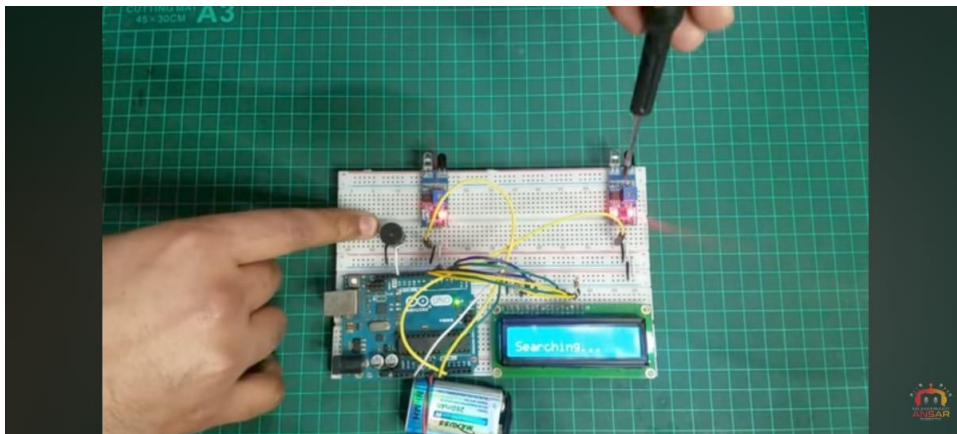
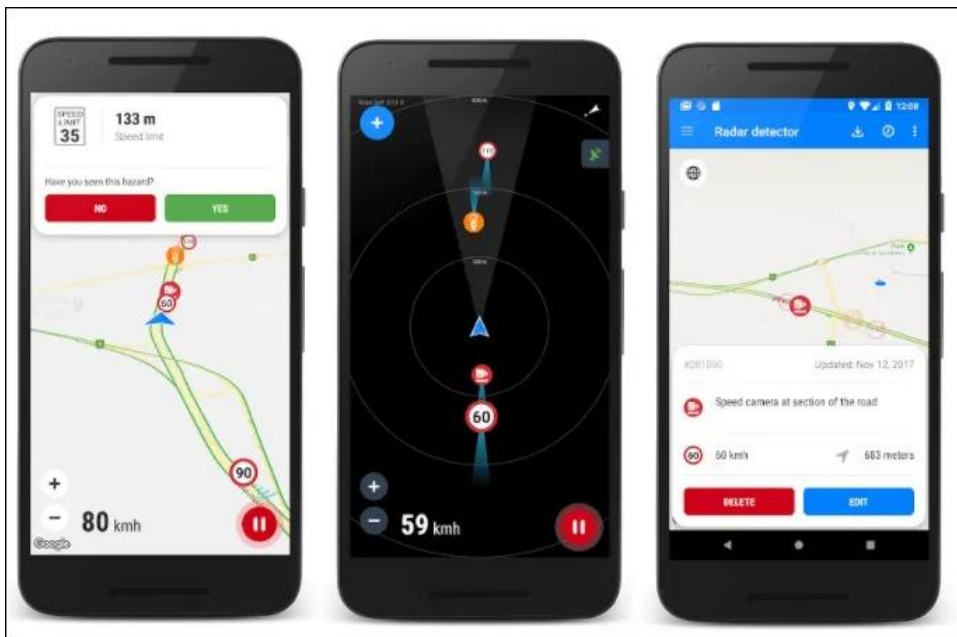# HTML PROGRAM :

```html
<!DOCTYPE html>
<html>
<head>
    <title>Smart Traffic Management</title>
</head>
<body>
    <header>
        <h1>Smart Traffic Management</h1>
    </header>

    <section id="traffic-status">
        <h2>Traffic Status</h2>
        <p id="status-info">Loading...</p>
    </section>

    <section id="camera-feed">
        <h2>Live Camera Feed</h2>
        <img id="camera-image" src="camera.jpg" alt="Live Camera Feed">
    </section>

    <section id="traffic-lights-control">
        <h2>Traffic Lights Control</h2>
        <button id="green-light">Green Light</button>
        <button id="yellow-light">Yellow Light</button>
        <button id="red-light">Red Light</button>
    </section>
```
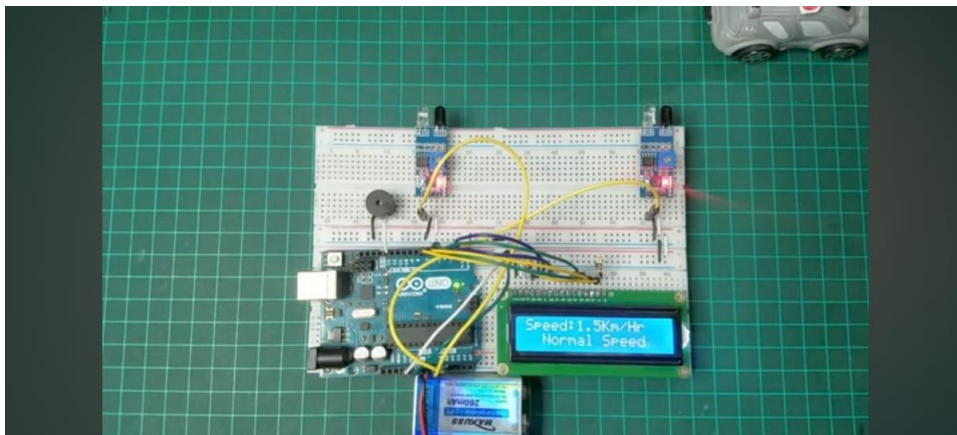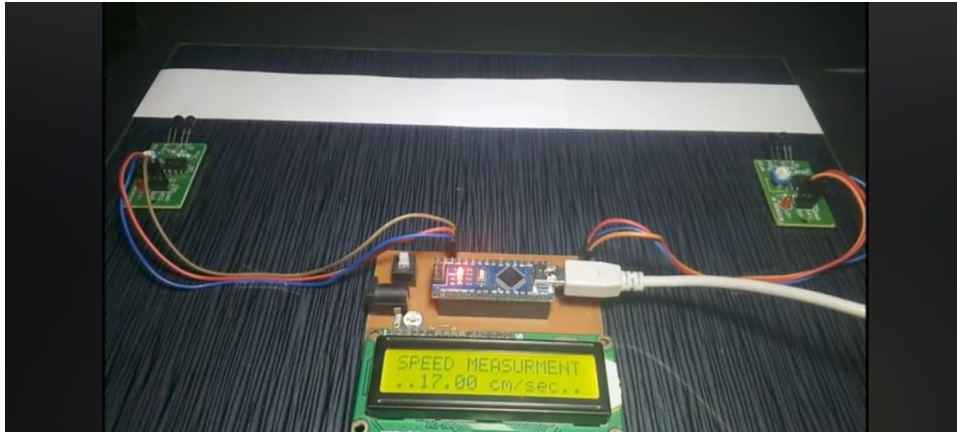
```html
<section id="sensors-data">
    <h2>Sensor Data</h2>
    <p id="temperature">Temperature: N/A</p>
    <p id="traffic-density">Traffic Density: N/A</p>
</section>

<footer>
    <p>&copy; 2023 Smart Traffic Management</p>
</footer>
</body>
</html>
```
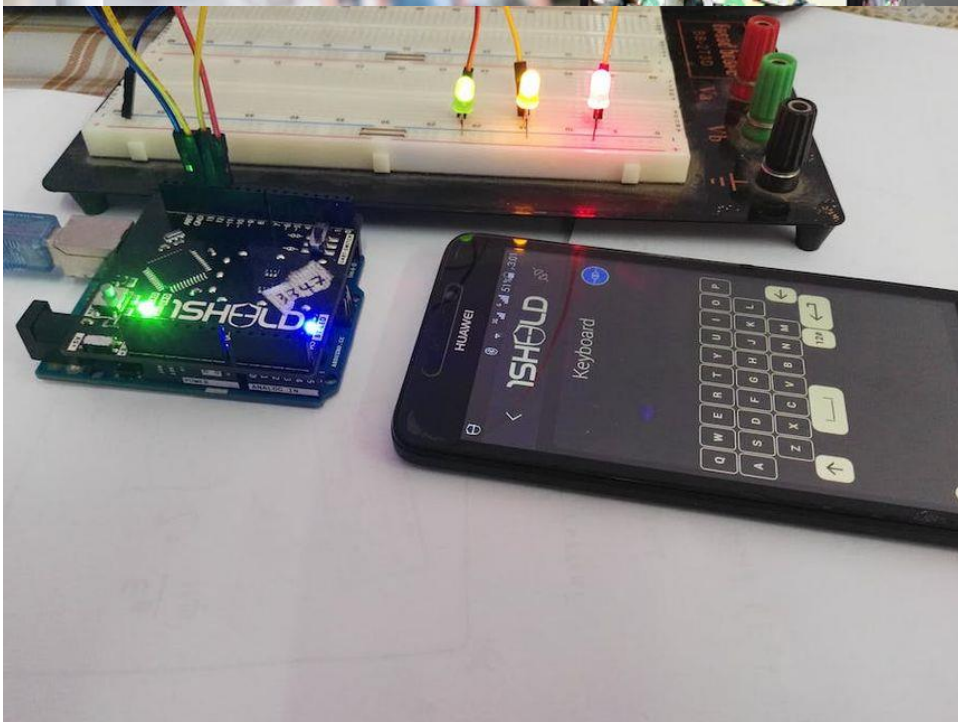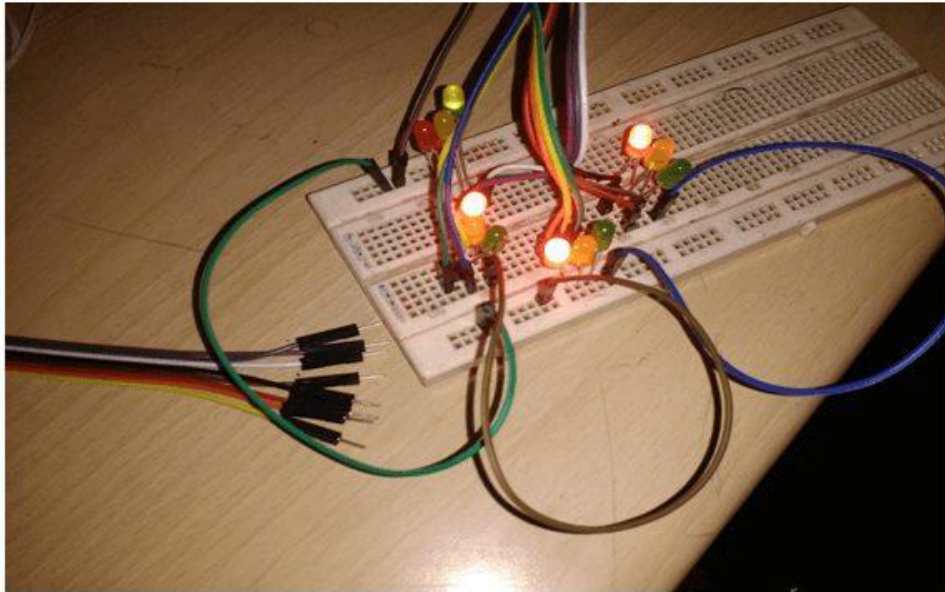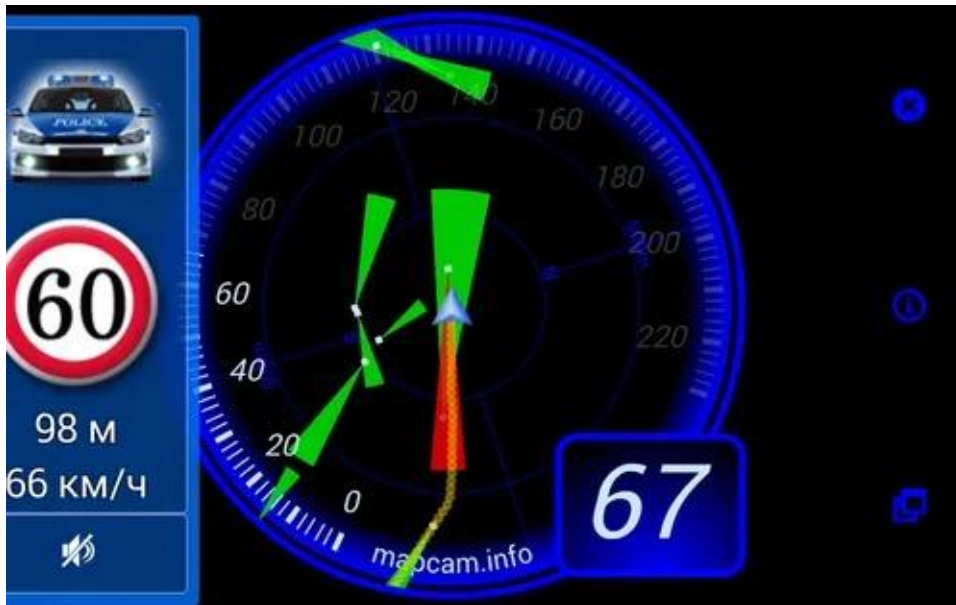
SAMPLE OUTPUT :

CONCLUSION :

A smart traffic management system in the context of the Internet of Things (IoT) holds great promise for addressing traffic congestion, enhancing safety, and improving overall traffic efficiency. The proposed work focuses on Smart Traffic management System using RFID which will eliminate the drawbacks of the existing system such as high implementation cost, dependency on the environmental conditions, etc. The proposed system aims at effective management of traffic congestion. It is also cost effective than the existing system. In conclusion, a smart traffic management system leveraging IoT has the potential to revolutionize urban mobility and address critical traffic-related challenges. With effective planning, collaboration between stakeholders, and addressing the associated challenges, IoTbased traffic management systems can create safer, more efficient, and sustainable urban environments.