International Journal of Engineering Science Invention (IJESI)
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org ||Volume 8 Issue 02 Series. I || Feb 2019 || PP 41-47

Intelligent Traffic Networking System: An IoT Application

Mr. Pranav Ruparelia[1], Ms. Reshma Malik[2]
[1](B.E., Information Technology Engineering, Thadomal Shahani Engineering College/Mumbai University,India)
[2](Assistant Professor, Department of Information Technology Engineering, Thadomal Shahani Engineering College/Mumbai University, India)
Corresponding Author: Pranav Ruparelia

Abstract: *The objective of this paper is to manage and reduce traffic congestion effectively. Traffic congestion may be a major downside in cities across the globe and has become a nightmare for the commuters in those cities. Conventional traffic signals are based on fixed time concept allotted to each side of the junction. Sometimes higher traffic density at one aspect of the junction demands longer Green-Time as to the assigned straightforward time.*
Intelligent Traffic Signaling System (ITSS), measures traffic congestion on each junction on the road, based on the density of vehicles in each lane. Traveler Information System (TIS) is one of the functional areas of ITSS, aimed at providing real-time traffic information to travellers. This information would be most effective if the traveler receive it during or before the start of their trip. Therefore, a user can access the TIS User Interface which conveys reliable information about the current and future state of traffic.
Keywords: *Internet of Things (IoT), ITSS, TIS, Sensors, Traffic Density, Google API*

Date of Submission: 24-01-2019 Date of acceptance: 08-02-2019

I. Introduction

Traffic congestion leads to long and unpredictable commute times, environmental pollution and wastage of fuel. It is a major challenge in the area of transportation planning as well as traffic management. These negative effects are additional acutes in developing countries like India, where infrastructure growth is slow.

Intelligent traffic management and better access to traffic information for commuters will alleviate congestion issues. The traffic lights will make sure that vehicles from each direction would be able to get a chance to proceed through the intersection in an orderly fashion. Normally, the lights of the traffic signal were programmed for particular time intervals. It was usually observed that during the peak hours of the traffic, one side of a road was heavy on traffic when compared to the other. Hence, programming equal intervals of time attribute to congestion during hours of heavy traffic, resulting in traffic delays. But, here we proposed a system that manages traffic signals on a junction based on the vehicle density, contrary to the old method of allotting constant time intervals to all sections of the junction irrespective of their traffic density. We measured the density of traffic by various sensors, which were placed on the sides of the road.[1] The sensors output was given to micro-controller to digitise output. Thus, depending on the density of traffic the timing of traffic lights was suitably set.

Even if we are able to solve the congestion to some extent, the problem cannot be solved completely. Therefore, informing road users about congestion conditions of different roadways will always be an added advantage to them while traveling to make suitable routing choices. Road users primarily choose route depending on the traffic conditions (real-time) than on conditions at the beginning of the trip. The sensor data collected in the Intelligent Traffic Signalling System (ITSS) was stored in a database, then made available for the road users as part of Traveler Information System (TIS).Therefore, along with the recent, reliable information about the future state of traffic was also being provided to the commuters which are based on the statistics of the large data set provided by the ITSS.

II. Existing Work

S.P. Biswas, P. Roy, N. Patra, N. Dey: developed a system that was used for optimisation of Green Light to reduce the congestion of the traffic on the road. The Path-Optimisation Technique and collected statistical data helped them to make the system function as a real-time traffic monitoring system. [2]

Divya R, Sindhu S, Manohar P, VishnuDatha S: proposed and developed a system where they used RFID tags for vehicles and developed a system that will count the vehicles with the reference of RFID tags equipped with vehicle ID for determining the Green Light of the signal. [3]

D. Singh, G. Tripathi, and A. J. Jara: developed an idea of how the Internet of Things with RFID tags can be used to make a smart embedded system along with semantic logic and semantic value based Information into an intelligent system.[4]

A real-time system that helps traveler with better information about the traffic congestion, with accurate prediction models, that helped in conveying the reliable information related the future state of the traffic was developed by Jithin R. and his colleagues. [5]

III. Design Requirements

3.1 Hardware Requirements

3.1.1 Micro-Controller:

Arduino UNO - Arduino Uno is based on ATmega328P single-chip micro-controller board. It has fourteen digital input/output pins, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header, and a reset button. [6]

Arduino Mega 2560 - The Arduino MEGA 2560 is intended for projects that need more I/O lines, more sketch memory and more RAM. With 54 digital I/O pins, 16 analog inputs and a larger space for your sketch it is the recommended board for 3D printers and robotics projects. [6]

3.1.2 Ethernet / Wi-Fi Shield:

Ethernet shield -The Arduino Ethernet Shield is used to connect your Arduino to the internet in minutes. Just plug ethernet module onto your Arduino Board, connect it to your network using an RJ45 cable. [7]

WiFi shield - The Arduino WiFi shield allows an Arduino board to connect to the internet using the WiFi library and to read and write data on SD card using the SD library. [8]

3.1.3 Sensors:

IR Reflectance Sensors - This type of hardware has both an IR source and an IR detector in a single housing in such a way that light from emitter LED bounces off an external object and is mirrored into a detector. The reflectivity of the surface of the object will determine the intensity of reflected light (into the detector). [9]

Ultrasonic detectors - These are sensors that operate by transmitting ultrasonic energy and measuring the energy reflected by the target. The multiple types of ultrasonic sensors available are presence-only and speed measuring. Both types measurements are processed and calculated to obtain measurements of vehicle presence, speed, and occupancy. [10]

3.1.4 I2C:

I2C is a serial protocol for a two-wire interface that adds up the low-speed devices like micro-controllers, EEPROMs, A/D and D/A converters, I/O interfaces and other alike peripherals in embedded systems. [11]

3.2 Software Requirements

3.2.1 Google API:

Google API Key - A user must register their app project on the Google API Console and get a Google API key which then can add to the app or website. The API key is used to authenticate the application's API, in the Google API Console.

Places API - The Places API for Web allows users to query for information regarding a place on a variety of categories, such as organisations, geo-locations, and more. A user can explore for places either by proximity or entering a text string. A Place Search returns a list of all the places along with summary information about each place. [12]

Geocoding API - Geocoding is the process of converting address strings (like "Mumbai, Maharashtra")into geographic coordinates (like latitude 19.0760° N and longitude 72.8777° E), which are then used as place markers on a map or to position the map. [13]

Directions API - A service that can calculate directions between users entered locations,by initiating an HTTP request is implemented by Direction API. There are multiple modes of transport for which a user can search the directions simultaneously. [14]

Google Maps Javascript API - Customisations in UI according to the users is carried out by The Maps JavaScript API. It lets users customise maps UI with their altered content and imageries can be display on web pages as well as their mobile devices. Maps JavaScript API is available with basic four features namely roadmap, satellite, hybrid, and terrain. The maps can be customised according to the users choice using layers, styles, controls and events, and various services/libraries. [15]

Place Autocomplete Maps - Autocomplete feature of the Places library in the Maps JavaScript API is a very handy tool for the users as it autocompletes to offer users applications type-ahead-search behaviour in the search field. When a user starts entering an address of a location, autocomplete feature will fill in the rest. [15]

Google Map Marker - A marker is used to distinguish a location on a map. Markers use a default icon and can display user-defined images, which are usually referred to as "icons". [16]

Google Map Poly lines - Poly lines are used to define lines on the map. A linear overlay of connected line segments on the map can be defined using the PolyLine Class. A PolyLine object is attributed with an array of every LatLng locations and a series of line segments is created which then connects those locations in an orderly manner. [15]

3.2.2 Arduino - Integrated Development Environment (IDE) :

The IDE is a text editor where a user-specific functioning code can be written along with that a message area, a text console, a toolbar buttons which are available for common functions. It is then used as a platform to upload programs to the Arduino board. The user-defined programs written using the IDE are called sketches. [6]

IV. System Flow

The system developed follows many steps in order to work as a complete functioning application. The design and the implementation of the system were done in such a way that the system's hardware and the software work in accordance with each other. The implementation of the system was developed in these following stages :

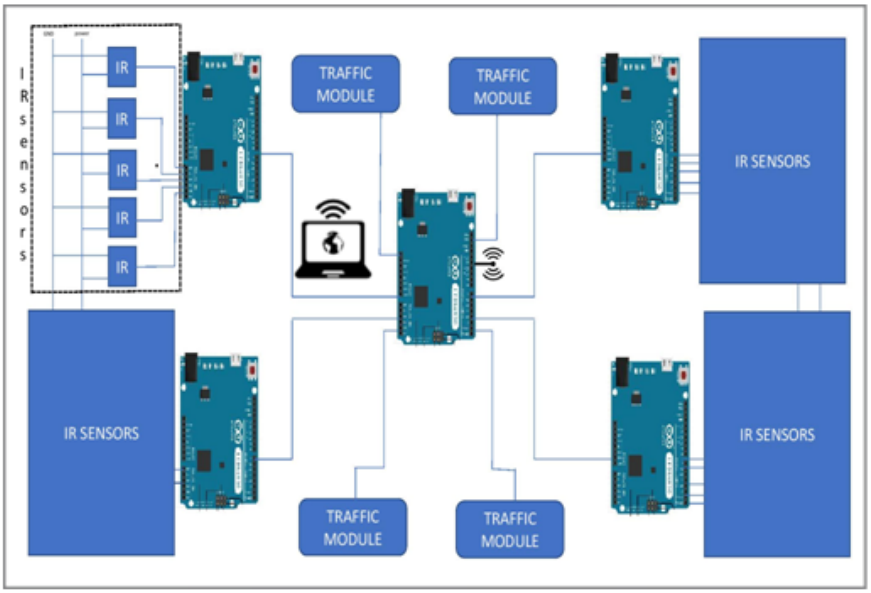

Fig. 1: Block Diagram of ITSS at a junction

Stage 1:

The master controller initiates the signalling system and accordingly, slaves are initialised. This stage was developed to collect the information on the real-time basis from the sensors that were aligned on every lane of the system as shown in Fig. 1. There were various modules developed, which included a micro-controller along with the sensors that were then utilised to read the input data.

Stage 2:

The input data collected from the sensors was processed and the density of the lane is being determined. This generated density of the system on request was sent to the master micro-controller. The Master Controller then operated on the master algorithm along with the received density and the generated Green Time of that particular junction was passed to the slave controller. The generated density of every cycle is stored and used by the application for further computations of Traveler Information System (TIS).

Stage 3:

The Master controller also sends the data to the Database of the system where the data can be made available using a UI which the User or the local commuter can have access too.

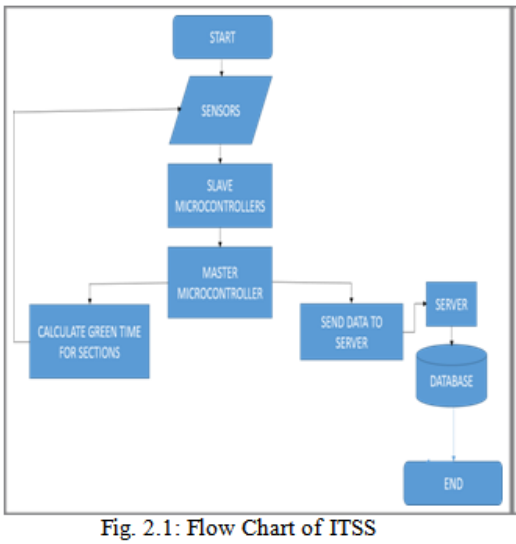

Fig. 2.1: Flow Chart of ITSS

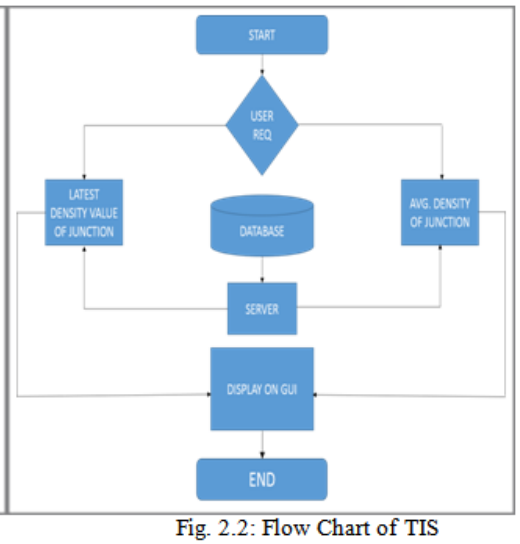

Fig. 2.2: Flow Chart of TIS

Algorithm for MASTER ARDUINO:

Step 1: Start
Step 2: Declare the No of Lanes and Cycle Duration variables
Step 3: Assign values as NoOfLanes <-4, CycleDur <-240
Initialise array for Green Duration of Lanes (GDL) <-CycleDur / NoOfLanes and Right Ratio of Lanes (RRL) <-0.75
Step 4: Set the Starting Green Time of every Section numbered from 1 to 6 as Green Start Duration (GSD).
SET GSD Section 1 <-0, GSD Section 2 <-(GDL 1 * RRL 1),
SET GSD Section 3 <-GDL 1 + (GDL 3 * (1- RRL 3),
SET GSD Section 4 <-GDL 1 + GDL 3,
SET GSD Section 5 <-GSD Section 4 + (GDL 2 * RRL 2),
SET GSD Section 6 <-GSD Section 4 + GDL 3 + (GDL 4 * (1 – RRL 4))

Step 5: Initialise the communication with the slaves. Define the cycle of the sections of every Lane according to the Current time

```
WHILE TRUE
        Initialise the variable startTime (ST) <-Current Time and
        currentTime (CT) <-Current Time (in seconds)
        WHILE CT <= (ST + CycleDur)
            IF CT = ST THEN SET GSD Section 4 HIGH
            ELSE IF CT = GSD Section 1 THEN SET GSD Section 2 HIGH
            ELSE IF CT = GSD Section 2 THEN SET GSD Section 3 HIGH
            ELSE IF CT = GSD Section 3 THEN SET GSD Section 4 HIGH
            ELSE IF CT = GSD Section 4 THEN SET GSD Section 5 HIGH
            ELSE IF CT = GSD Section 5 THEN SET GSD Section 6 HIGH
            ELSE IF CT >= CycleDur – YellowLightTime THEN
                Initialise a variable Sum <-0
                FOR I = 1 to NoOfLanes
                        INITIALISE an Array Vehicle Density (VD).
                        GET VD <-SLAVE Calculate Sum of all densities
                        Sum <-Sum + VD [I]
                ENDFOR
                FOR I = 1 to NoOfLanes
                        GDL [I] <-CycleDur / Sum * VD [I]
                ENDFOR
                Repeat Step 4
        ENDIF
ENDWHILE
ENDWHILE
```

Step 6: Stop

Algorithm for SLAVE ARDUINO:

Step 1: Start

Step 2: Declare the No of Sensors, Sensor Read Time, Cycle Time and Yellow Light Time variables.

Step 3: Assign a value as NoOfSensors <-5, cycleTime <-240, sensorReadTime (SRT) = 15 AND YellowLight-Time (YLT) = 5

Step 4: Initialise Input Pins to the sensors and Communication with MASTER

GET Sections Green Start Time (SGST) from MASTER

Step 5: Read the Sensors Data and calculate the Density of the corresponding Lane. Synchronise the sensors to by adding delay. Send the Data to the Master on request and wait till the Green-Time Duration from the master is received.

```
WHILE TRUE
    SET currentTime (CT) <-Current Time (in sec)
    IF CT >= SGST – SRT – YLT AND CT <= SGST – YLT THEN
      WHILE CT <= SGST – YLT
        FOR I =1 to NoOfSensors
          INITIALISE AND READ sensorData [I] from Sensor I
        ENDFOR
        FOR I =1 to NoOfSensors
          IF sensorValue [I] = FALSE THEN
            IF sensorData [I] = 1 THEN
              INITIALISE countData Array AND
SET countData [I] <-countData [I] +1
            ELSE
SET countData [I] <-0
            ENDIF
            IF countData [I] = 200 THEN
              SET sensorValue [I] <-TRUE
            ENDIF
          ENDIF
        ENDFOR
        DELAY of 1 millisecond
        SET CT <-Current Time in seconds
      ENDWHILE
      INITIALISE vehicleDensity <-0
      FOR I = NoOfSensors to 1
        IF sensorValue [I] = TRUE THEN
          SET vehicleDensity <-I
          FOR J = I-1 to 1
            IF sensorValue [J] = TRUE THEN
              BREAK FROM LOOP
            ELSE
              SET vehicleDensity <-J
            ENDIF
          ENDFOR
        ENDIF
      ENDFOR
     WRITE vehicleDensity to MASTER once Communication is set.
GET SGST from the MASTER once it is computed.
DELAY of 1 second
    ENDIF
  ENDWHILE
```

Step 6: Stop

V. Implementation

SLAVE Arduino: The Slave Arduino powers the IR ReflectanceSensors to obtain the data (in the binary format). The received data from the sensors was then evaluated and the density of that particular lane was determined by the SLAVE and was passed to the MASTER micro-controller.

As the connection between Master and Server is already established, the MASTER sends the Lane Time back. We can see in the above figure Fig. 3.1 that after the inputs have been accepted, the algorithm proceeds to determine the density of the Lane. Since the vehicles in Lane were aligned till the Sensor No 3, the density was accordingly calculated and displayed in the Slave Serial Monitor.

MASTER Arduino: Master Arduino then evaluates this received density using the algorithm. On the basis of this density, the Lane Time gets calculated. The lane with the highest density is allocated with maximum Green Time and others will get on the basis of density calculations. These values will then be forwarded back to the SLAVE. Along with this, the server stores the density of each lane in the database which then can be used for further computation.

Finally based on the values of the density of each lane, the calculated LANE TIME (in milliseconds) is displayed on the Master Serial Monitor shown in Fig. 3.2.

Google Maps API: In Fig. 3.3, we can see that the data from the Database was used to display the traffic of every junction that lies in the path of source and destination provided by the user. With the help of this data, the user could plan their trip accordingly.

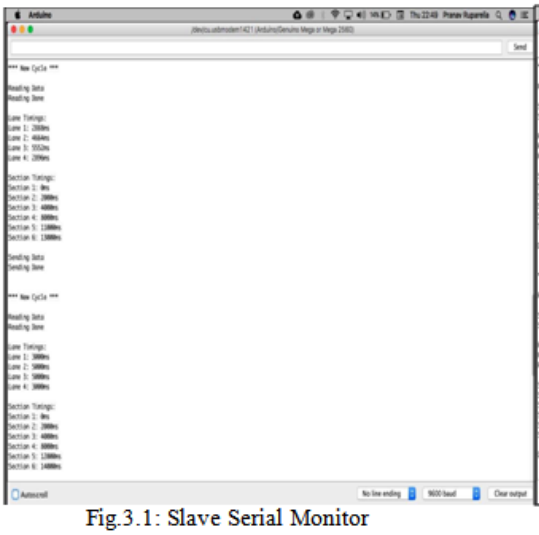

Fig.3.1: Slave Serial Monitor

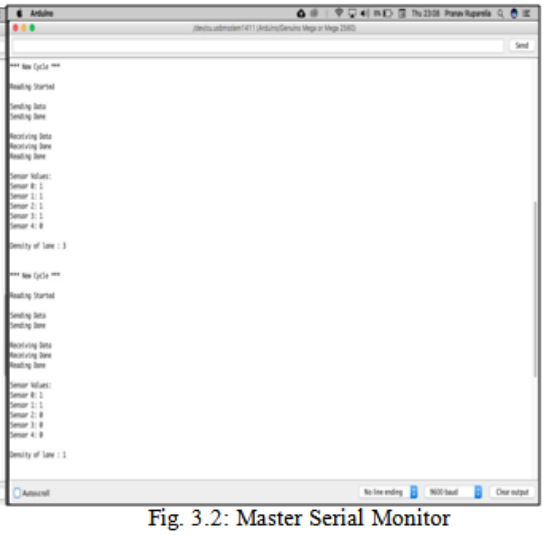

Fig. 3.2: Master Serial Monitor

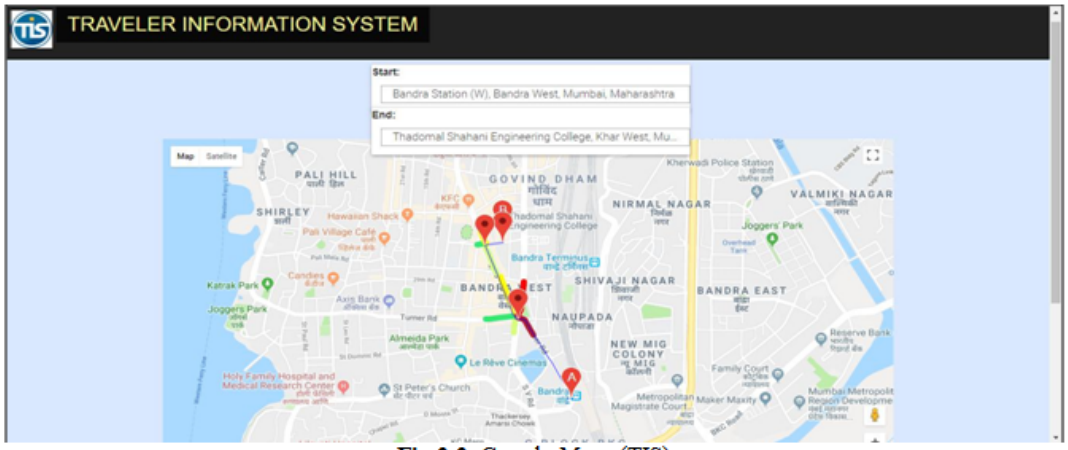

Fig.3.3: Google Maps (TIS)

VI. Conclusion

Traffic congestion, a major problem in the metropolitan, can be attributed to the static, non-adaptive nature of our Signalling System. This is where the ITSS comes into the picture. Being a dynamic, real-time & automatic system which adjusts the system according to the density of the vehicles, will help in reducing the traffic congestion which in turn would mean reduction in travel time, less pollution, lesser accidents and traffic violations, and less frustration of the road user. Along with this, to further help the road user to plan his travel, the TIS, one of the functional area of ITSS, provides reliable current and typical or future traffic scenario.

References

1. Density Based Traffic Signal System using Micro-controller, [Online] Available: http: // www. electronicshub.org/ density-based-traffic-signal- system-using-microcontroller
2. Satya Priya B., Paromita R., Nivedita P.,Amartya M. and Nilanjan D., Intelligent Traffic Monitoring System, ©Springer India 2016, S.C. Satapathy et al. (eds.), Proceedings of the Second International Conference on Computer and Communication Technologies, Advances in Intelligent Systems and Computing 380, DOI 10.1007/978-81-322-2523-2_52[Online] Available : https://www.researchgate.net/publication/280078500_Intelligent_Traffic_Monitoring_System

3. Divya R., Sindhu S., Manohar P., VishnuDatha S., Intelligent Traffic Monitoring System, Project Reference No.: 39S_BE_2073. Available : http://www.kscst.iisc.ernet.in/spp/39_series/SPP39S/02_Exhibition_Projects/185_39S_BE_2073.pdf
4. (2004) D. Singh, G. Tripathi and A. J. Jara, A survey of Internet-of-Things: Future vision, architecture, challenges and services, 2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul, 2014, pp. 287-292, DOI: 10.1109/WF-IoT.2014.6803174. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6803174&isnumber=6803102
5. Jithin R., Hareesh B., Lelitha Devi V., Application of data mining techniques for traffic density estimation and prediction, © 2016 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license. Peer-review under responsibility of the Department of Civil Engineering, Indian Institute of Technology Bombay DOI: 10.1016/j.trpro.2016.11.102 Available:http://www.sciencedirect.com/science/article/pii/S2352146516307177
6. "Arduino Micro_Controller Guide", Available:https://www.arduino.cc/en/Guide/HomePage
7. "Arduino Ethernet_Shield Guide", Available : https://www.arduino.cc/en/Main/ArduinoEthernetShieldV1
8. "Arduino WiFi_Shield Guide", Available : https://www.arduino.cc/en/Guide/ArduinoWiFiShield
9. Preeti J., IR Reflectance Sensors, Available : https://www.engineersgarage.com/articles/infrared-sensors
10. Ultrasonic Detection–Basics & Application, Available : https://www.elprocus.com/ultrasonic-detection-basics-application/
11. I2C Info – I2C Bus, Interface and Protocol,Available:http://i2c.info/
12. Overview | Places API | Google Developers, Available : https://developers.google.com/places/web-service/intro
13. Overview | Geocoding API | Google Developers,Available:https://developers.google.com/maps/documentation/geocoding/intro
14. Overview | Directions API | Google Developers, Available:https://developers.google.com/maps/documentation/directions/intro
15. Overview|MapsJavaScriptAPI|GoogleDevelopers, Available : https://developers.google.com/maps/documentation/javascript/tutorial
16. Overview | Markers | Google Developers, Available : https://developers.google.com/maps/documentation/javascript/markers
17. Ashutosh B., Traffic Control System, Available:https://www.engineersgarage.com/contribution/ambhatt/traffic-control-sytem
18. Ravindra Pal S., Smart Traffic Management With Real Time Data Analysis,Available:https://www.trafficinfratech.com/smart-traffic-management-with-real-time-data-analysis/
19. GHEN Paste-bin, Simple traffic light system with timer, Available: https:// pastebin.com/zbuAJkSm

Pranav Ruparelia" Intelligent Traffic Networking System: an Iot Application" International Journal Of Engineering Science Invention (Ijesi), Vol. 08, No. 02, 2019, Pp 41-47