**SMART AMBULANCE GUIDANCE SYSTEM**

**PROJECT REPORT**

***Submitted By***

**T AJAIY PRAVEEN 312216205006**

**BALAJI U 312216205016**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

**SSN COLLEGE OF ENGINEERING**

**KALAVAKKAM 603110**

**ANNA UNIVERSITY :: CHENNAI - 600025**

**April 2020**

**ANNA UNIVERSITY : CHENNAI 600025**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“ SMART AMBULANCE GUIDANCE SYSTEM** ” is the bonafide work of “ **T AJAIY PRAVEEN. (312216205006), and BALAJI U. (312216205016)**” who carried out the project work under my supervision.

**DR T. NAGARAJAN MR V. ARULKUMAR**

**HEAD OF THE DEPARTMENT SUPERVISOR**

Professor Assistant Professor

Department of IT Department of IT

SSN College of Engineering SSN College of Engineering

Kalavakkam - 603110 Kalavakkam - 603110

Place:

Date:

Submitted for the examination held on . . . . . . . . . . . .

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENTS**

We would like to thank and show our deep sense of gratitude to our guide **Mr. V. Arulkumar**, Assistant Professor, Department of Information Technology, for his valuable advice and suggestions as well as his continued guidance, patience and support that helped us to shape and reﬁne our work.

Our sincere thanks to **Dr. T. NAGARAJAN**, Professor and Head of the Department of Information Technology, for his words of advice and encouragement and we would like to thank our Project Coordinator **Dr. K. R. UTHAYAN**, Associate Professor, Department of Information Technology and Dr.

**R. SRINIVASAN**, Professor, Department of Information Technology for their valuable suggestions throughout the project.

We express our deep respect to the founder **Dr. SHIV NADAR**, Chairman, SSN Institutions. We also express our appreciation to our **Dr. S. SALIVAHANAN**, Principal, for all the help he has rendered during this course of study.

We would like to extend our sincere thanks to all the teaching and non-teaching staffs of our department who have contributed directly and indirectly during the course of our project work. Finally, we would like to thank our parents and friends for their patience, cooperation and moral support throughout our life.

**T AJAIY PRAVEEN BALAJI U**

# **ABSTRACT**

Increase in population has resulted in the growth of automobiles leading to a surge in traffic. Often traffic jams occur due to the escalation in the number of vehicles wqon the road. Adhering to the absence of lane systems in countries like India an ambulance or any emergency vehicle find it tenuous to navigate through the traffic and the journey is disrupted. This project aims at providing a solution to the above problem by controlling the traffic signal as an ambulance approaches the traffic signal, using the concept of Internet of Things (IoT) integrates with an android application. The system also alerts the general public at the signal about the approaching ambulance by activating the white light indicator placed at each signal-junction. This system uses a central server (Firebase) to control the traffic controllers. The traffic signal controller is implemented using Arduino NANO and NodeMCU. The mobile application keeps track of the current location of the ambulance and automatically intimates the nearby traffic signal and turns it to green. Once the ambulance crosses the junction the signals are restored to its previous state. As this system of trafficking incurs low cost, it can be implemented throughout the city thereby reducing the number of deaths by allowing the ambulance to quickly skip the traffic and reach the destination as early as possible. This application can be extended by sending patient details to the hospital for future endeavours.

Table of Contents

[ABSTRACT iii](#_Toc35984489)

[LIST OF FIGURES iv](#_Toc35984490)

[1 INTRODUCTION 1](#_Toc35984491)

[1.1 Overview 1](#_Toc35984492)

[1.2 Objective 2](#_Toc35984493)

[1.3 Motivation 2](#_Toc35984494)

[1.3 Scope 3](#_Toc35984495)

[2 LITERATURE SURVEY 5](#_Toc35984496)

[2.1. Intelligent Ambulance with Traffic Control - RF communication 5](#_Toc35984497)

[2.2 IR Sensor Based Intelligent Ambulance with some Advance features of Telecommunication 5](#_Toc35984498)

[2.3 Smart Ambulance to Control Traffic Signals using IoT and Cloud 6](#_Toc35984499)

[2.4 Smart Traffic Control System Using Image Processing 7](#_Toc35984500)

[2.5 A VANET-based emergency vehicle warning system 7](#_Toc35984501)

[3 SYSTEM DESIGN 9](#_Toc35984502)

[3.1 System Architecture 9](#_Toc35984503)

[3.1.1 Android Application 10](#_Toc35984504)

[3.1.2 Realtime Cloud Server 11](#_Toc35984505)

[3.1.3 Internet Of Things 12](#_Toc35984506)

[3.2 Components Used 13](#_Toc35984507)

[3.2.1 Arduino Nano 13](#_Toc35984508)

[3.2.2 NodeMCU 18](#_Toc35984509)

[3.3 Components and Environment Used 21](#_Toc35984510)

[3.3.1 Arduino IDE 21](#_Toc35984511)

[3.3.2 Firebase Console 21](#_Toc35984512)

[3.3.3 Android Studio 21](#_Toc35984513)

[3.3.4 Adobe XD 22](#_Toc35984514)

[3.4 Languages and packages used 22](#_Toc35984515)

[3.4.1 Java 22](#_Toc35984516)

[3.4.2 Kotlin 23](#_Toc35984517)

[3.4.3 C++ 23](#_Toc35984518)

[3.4.4 Maps SDK for Android 24](#_Toc35984519)

[3.4.5 GeoFire 25](#_Toc35984520)

[3.4.6 ESP8266 Arduino core 25](#_Toc35984521)

[3.4.7 Firebase Arduino 26](#_Toc35984522)

[4 IMPLEMENTATION 27](#_Toc35984523)

[4.1 Building Android application 27](#_Toc35984524)

[4.1.1 Detecting current location 27](#_Toc35984525)

[4.1.2 Generating polyline 29](#_Toc35984526)

[4.1.3 Simulation with Animation Package 31](#_Toc35984527)

[4.1.4 Integrating Firebase using GeoFire 32](#_Toc35984528)

[4.2 Building IoT Module 36](#_Toc35984529)

[4.2.1 Software Serial communication between NANO and NodeMCU 36](#_Toc35984530)

[4.2.2 WiFi connection for ESP8266 – NodeMCU 37](#_Toc35984531)

[4.2.3 Connecting Arduino to Firebase 38](#_Toc35984532)

[5 RESULT AND ANALYSIS 40](#_Toc35984533)

[6 CONCLUSION AND FUTURE SCOPE 45](#_Toc35984534)

[6.1 Conclusion 45](#_Toc35984535)

[6.2 Future Scope 45](#_Toc35984536)

[7 REFERENCES 46](#_Toc35984537)

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **No.** | **Title** | **Page No** |
| **3.1** | System Architecture | **9** |
| **3.2(a)** | Arduino NANO Pinout | **15** |
| **3.2(b)** | NodeMCU Pinout | **20** |
| **4.1(a)**  **4.1(b)** | Current location  Polyline | **28**  **30** |
| **4.1(c)** | Simulation | **32** |
| **4.1(d)**  **4.1(e)**  **4.2 (a)** | Notification  Firebase Console  Serial Communication between Nano and NodeMCU | **33**  **35**  **36** |
| **4.2 (b)** | Flow Diagram | **39** |
| **5.1** | Iot Simulation before signal changes | **40** |
| **5.2** | In App Before Signal Changes | **41** |
| **5.3** | In App Signals are changed | **42** |
| **5.4** | Blue Stroboscopic LED | **43** |
| **5.5** | In App Signals are restored | **44** |

# 

# 

# 1 INTRODUCTION

## Overview

The pace at which the world is developing is very high today. Reformations in technology every day is evolving and improving efficiency in healthcare sector which is one of the most difficult and challenging jobs. Due to Industrialization and Urbanization, the population increases day by day leading to a surge in the number of vehicles on the roads which results in high traffic jams in big cities. Traffic congestion causes many adversary effects on the country’s transportation. One of the widely affected service due to traffic jams is that of an ambulance. Many a times, ambulance consist of emergency or critical patients who need to be taken to the hospital in minimum amount of time, providing proper treatment to the patient so that chances of surviving increases in critical condition. A patient may lose his life if there is delay in reaching of ambulance to the hospital. According to the surveys 95% of the heart attacks cases can be treated, if the ambulance can reach the hospital at the right time without getting stuck in the traffic. In order to achieve this, the surrounding vehicles need to clear a path allowing smooth navigation for the ambulance. We can overcome these limitations by using emerging technologies such as IoT i.e. Internet of Things. Various software implementations and hardware devices can be connected with the help of wireless networking tools or wired tools. In IoT the components are connected and controlled by the internet. Thus the impact of IoT in today’s era is significant as it helps to represent the object digitally and makes itself something greater than the object by itself. In this project, we have come up with the idea of a ‘Smart Ambulance Guidance System’ to alleviate the above mentioned issues. The main objective of this system is to make it possible for the ambulance to reach a particular location without having it to stop anywhere until the destination is reached. This project proposes an automatic monitoring of traffic lights where the driver need not be involved. The driver enters the desired destination in the android application and a route map is generated. Using the current location of the ambulance, the nearest traffic signals are accessed and turns it to green if red. Once it crosses the junction, the signals are restored to its previous flow. Using this method, the traffic is cleared providing a way for the ambulance allowing it to reach the hospital in minimum time.

## 1.2 Objective

The proposed system connects the traffic light and the ambulance via a mobile application through a cloud server. Microcontrollers will be implanted in each signal which are centrally controlled by the server. An android application will be used to provide realtime coordinates of the ambulance to the server and accesses the nearest signal accordingly. Moreover the surrounding vehicles will be intimated of an approaching ambulance by installing an additional stroboscopic light in each of the signals.

## 1.3 Motivation

Road traffic congestion becomes a major issue for highly crowded metropolitan cities.

According to the National Crime Records Bureau, nearly 24,012 people die each day due to a delay in getting medical assistance. These patients have suffered heart attacks, brain haemorrhage, suicide attempts, accidents and strokes.  Nearly 4,40,042 cases are reported across the country each year, of which 1,39,091 people lose their life. The first hour after the incident, or the Golden Hour, is critical. Many accident victims wait for help at the site, and a delay costs them their life. Ambulances getting stuck in traffic is one of the key reasons for this.

Many people suspect that ambulances that rush around may not really carry a patient in need of urgent medical treatment. But the truth is, even if there is no patient on board, an ambulance that turns on its sirens is definitely heading to pick up a patient.

Ambulance service is one of the major services which gets affected by traffic jams. To smoothen the ambulance movement we have come up with the solution of “Smart Ambulance Guidance System”.

## 1.3 Scope

One of the major problems faced by heavy traffic is by ambulances. But due to heavy traffic, one can often see the ambulances stuck in traffic for long durations thus causing danger to patient’s life. As the number of road users constantly increases, and resources provided by current infrastructures are limited, intelligent control of traffic has become the need of the hour. Hence the movement of ambulances during peak hours of traffic needs to be smooth. This project aims at clearing the traffic for the ambulance by the changing the signal to green if it is red. This ensures that ambulance reach the destination without any restraint in between. Many lives can be saved as the route for the ambulance would be cleared to reach the emergency spot and back to the hospital in an even quicker time. Since the time taken to reach either the emergency spot or hospital is considerably reduced, this system could prove to be very useful and efficient in preventing a lot of fatalities.

Unlike western countries, Indian cities cannot think of having separate lanes for emergency purpose due to lack of road planning and infrastructure. With the lives of the patients depending on the speedy arrival of the ambulances to hospital, an alternative solution to the above problem is the need of the hour. Thus, this project finds great use in large metropolitan cities where ambulances struggle to reach the destination within the stipulated time.

# 2 LITERATURE SURVEY

## 2.1. Intelligent Ambulance with Traffic Control - RF communication

In this project the authors have used hardware to calculate the health parameters. Serial communication is used to store it in PC which is in ambulance through which they are transferred to the hospital. RF communication is used to control the traffic. The two systems which are combined in this paper are - health monitoring and traffic controlling systems. Data acquisition will take place in Health monitoring system and parameters will be sent to the hospital server via PC. The driver of the ambulance controls the traffic using the keypad in the ambulance. Both the systems will work simultaneously. The doctor in the hospital monitors the patients’ health parameters. The signals could be manipulated by the driver of the ambulance at the same time. By adding a GPS navigation system with a congestion detection module, this system can be improved for the real time scenario.

## 2.2 IR Sensor Based Intelligent Ambulance with some Advance features of Telecommunication

The main aim of the paper is to design a Microcontroller based intelligent ambulance system which can change the traffic lights upon its arrival at traffic light junction using IR(Infrared) sensors. The ambulance system also has Global System for Mobile Communication based information device that alerts the doctors about the patient’s condition and informs the doctors to report to the nearest hospital for patient’s quick recovery. There could be a case where two Ambulances are exactly at equal distance from traffic light, in this case the traffic light receiver will give chance to the transmitter of any one Ambulance randomly without considering any fact.

## 2.3 Smart Ambulance to Control Traffic Signals using IoT and Cloud

In this project, the large amount of data that is generated by these devices can be handled by cloud computing and it can also be used to send command to those devices to perform a task. This project is based on the IoT and cloud. This project is to establish the communication between the traffic signals and the ambulance so that the traffic signal can respond to the arrival of the ambulance. The application needs a required bandwidth for the instantaneous communication between the ambulance and the traffic signal. B. Janani Saradha, et al. proposed the microcontroller based RFID system that is used to alter the traffic lights upon arrival at traffic light junction. The system creates an android app that connects both the ambulance and the traffic signal station using cloud network. This system makes use of Radio frequency identification technology to implement the intelligent traffic signal control. The basic idea behind the proposed system is, if the Ambulance halts on the way due to a traffic signal, RFID installed at the traffic signal tracks the RFID tagged ambulance and sends the data to the cloud. After the acknowledgment for the user through the mobile app, the particular signal is made Green for some time and after the ambulance passes by, it regains its original flow of the sequence of signaling, if this scheme is fully automated, it finds the ambulance spot, controls the traffic lights. This system controls the traffic lights and saves the time during emergency periods.

## 2.4 Smart Traffic Control System Using Image Processing

The system will be an image processing based adaptive signal controlling. Proposed system will be based on traditional system along with automated signal. Digital camera is mounted on the motor for rotation. This faces the lanes and gets the sense of the traffic. The artificial vision is captured with the help of the digital camera. The camera’s direction changes in the steps of 90 degrees, it faces each lane and captures the image. In order to change the direction of the camera, it is controlled by the PC through microcontroller. Load of the traffic on each lane is estimated by Image processing techniques. The accuracy of the image processing compared to GPS is low. When an ambulance is detected, and the signal is green, the timing is extended by 20 seconds and if the signal is red, the timing is extended for a longer duration of 40 seconds. This can be extended to any number of signals along the ambulance’s path so that the emergency cases can be served immediately. If a vehicle of a bigger size than an ambulance is in front of the ambulance, then the camera will not be able to capture the ambulance.

## 2.5 A VANET-based emergency vehicle warning system

Buchenscheit, Andreas, et al. have implemented VANET-based emergency vehicle warning system. They have designed an emergency vehicle warning system that makes use of inter-vehicle communication and also encompasses roadside infrastructure like traffic lights. In this system, other vehicles are warned of an approaching emergency vehicle and also receive detailed route information. Based on this information, a timely and appropriate reaction of other drivers is possible.

# 3 SYSTEM DESIGN

## 3.1 System Architecture

**A drawing of a cartoon character

Description automatically generated** **Figure 3.1: System Architecture**

The architecture of the system has mainly four components as shown in figure 3.1

The system consists of the following modules

* Android application
* Realtime Cloud Server (firebase)
* IOT (microcontrollers and traffic signals)

### 3.1.1 Android Application

The android application is built using android studio. The current location of the ambulance will be visible in the application. A search bar is present in the application where the user can enter the name of the hospital and get the results according to the search. Once the user chooses a particular destination, a get directions button pops up which provides the shortest route to reach the entered destination from the current location of the ambulance. The shortest route is calculated using Google directions API. A polyline is generated as soon as we hit the get directions button so that the ambulance can trace its route while moving. When the ambulance reaches the destination the trip gets completed.

Google maps API and Google directions API were extensively used to server the purpose of this project. The app is further connected to a real time cloud server (Firebase) where the real time coordinates of the ambulance are constantly updated. A functionality is used in the android application which is used to constantly check the coordinates of nearest the signal if any exists based on the real time coordinates of the ambulance. If a signal is found in a junction, the direction in which the ambulance is moving in updated in the real time database and the signal present in that direction in the junction is accordingly changed to green if it is red or if it is green the same signal flow is maintained. When the ambulance crosses the junction, there is another functionality present in the android application which restores the previous signal flow of the junction. The commuters in the other signals are intimated of an approaching ambulance by using an additional stroboscopic light in the signal which indicates the commuters that an ambulance is going to pass through.

### 3.1.2 Realtime Cloud Server

The realtime cloud server used for this project is firebase. The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. The Realtime Database provides a flexible, expression-based rules language, called Firebase Realtime Database Security Rules, to define how our data should be structured and when data can be read from or written to. When integrated with Firebase Authentication, we can define who has access to what data, and how they can access it. Instead of typical HTTP requests, the Firebase Realtime Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds. The Realtime Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Realtime Database API is designed to only allow operations that can be executed quickly.

Thus all the realtime data used in this project are stored in this realtime cloud hosted database. The current coordinates (latitude and longitude) of the ambulance is constantly updated in the database. The coordinates of the signals present in a junction are also added to the database. The direction in which an ambulance is moving is also constantly updated in the database so that once the ambulance comes near a junction, the signal in that junction is accessed based on the direction in which the ambulance is moving and the signal flow is updated accordingly. Values are assigned for each colour in the signals which keeps changing dynamically as and when an ambulance approaches the junction.

### 3.1.3 Internet Of Things

The components used in this module of the project are :

* Arduino Nano
* NodeMCU
* LED lights
* Mini breadboards
* Jumper wires

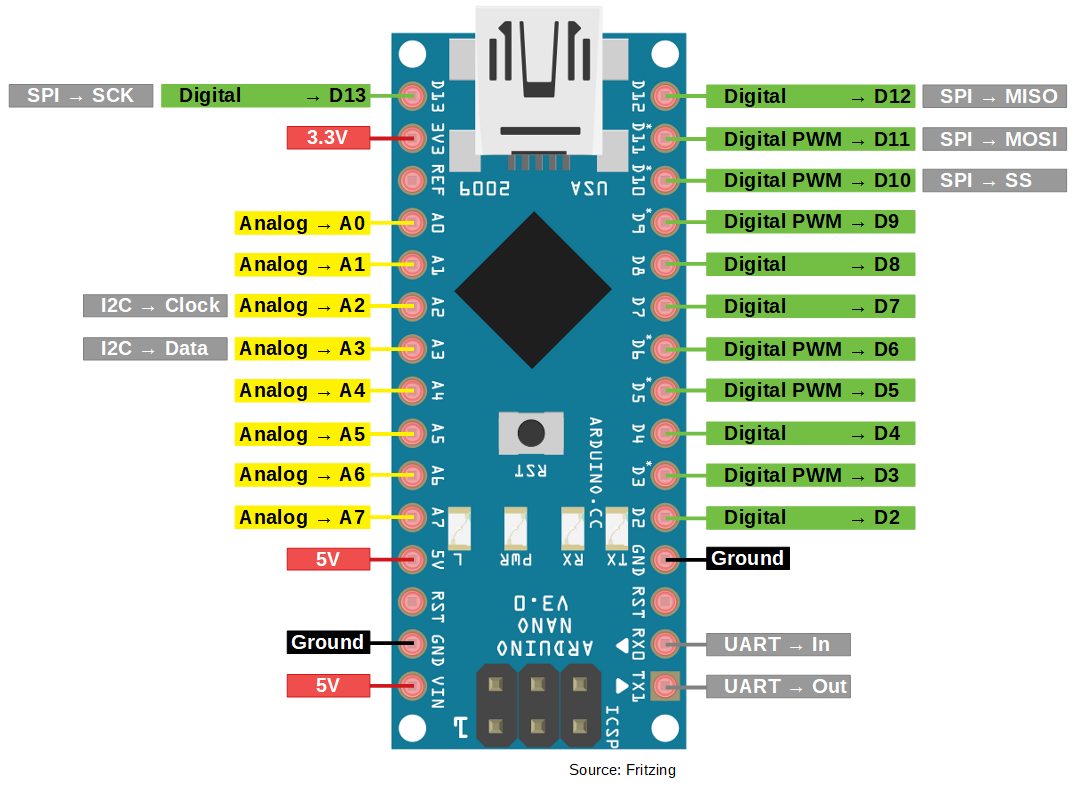
The microcontrollers Arduino Nano and NodeMCU were programmed using the Arduino IDE. The general purpose input/output pins (GPIO pins) in both these microcontrollers were exploited to provide input to the LED lights. NodeMCU has an inbuilt wifi module(ESP8266 12E) which is used to obtain the realtime values of each of the signals from the database and is fed as input to the GPIO pins in both the microcontrollers. The NodeMCU receives all the values of the signals from the database in realtime and constantly keeps updating the GPIO pins to which the LEDs are connected. The NodeMCU also serves another purpose by serially sending the values obtained from the realtime database to Arduino Nano which inturn keeps constantly updating its GPIO pins to which another set of LEDs are connected. Thus based on the realtime values of the signals obtained from the database, the boards are programmed to simulate the realtime working of the signals with the help of LEDs once an ambulance approaches a junction and how the signal flow is restored once the ambulance crosses the junction.

## 3.2 Components Used

### 3.2.1 Arduino Nano

* **Arduino Nano**  is a small, compatible, flexible and breadboard friendly Microcontroller board. It comes with exactly the same functionality as in Arduino UNO but quite in small size.
* It comes with an operating voltage of 5V, however, the input voltage can vary from 7 to 12V.
* **Arduino Nano Pinout** contains 14 digital pins, 8 analog Pins, 2 Reset Pins & 6 Power Pins.
* Each of these Digital & Analog Pins are assigned with multiple functions but their main function is to be configured as input or output.
* They are acted as input pins when they are interfaced with sensors, but if you are driving some load then use them as output.
* Functions like pinMode() and digitalWrite()  are used to control the operations of digital pins while analogRead() is used to control analog pins.
* The analog pins come with a total resolution of 10bits which measure the value from zero to 5V.
* The serial communication is carried out by the digital pins like pin 0 (Rx) and pin 1 (Tx) where Rx is used for receiving data and Tx is used for the transmission of data. The serial monitor is added on the Arduino Software which is used to transmit textual data to or from the board. FTDI drivers are also included in the software which behave as a virtual com port to the software.
* The Tx and Rx pins come with an LED which blinks as the data is transmitted between FTDI and USB connection to the computer.
* Arduino Software Serial Library is used for carrying out a serial communication between the board and the computer.

**PINOUT**

****

**FIGURE: 3.2(a) Arduino NANO Pinout**

**Vin.** It is input power supply voltage to the board when using an external power source of 7 to 12 V.

**5V.** It is a regulated power supply voltage of the board that is used to power the controller and other components placed on the board.

**3.3V.** This is a minimum voltage generated by the voltage regulator on the board.

**GND.**  These are the ground pins on the board. There are multiple ground pins on the board that can be interfaced accordingly when more than one ground pin is required.

**Reset.** Reset pin is added on the board that resets the board. It is very helpful when running program goes too complex and hangs up the board. LOW value to the reset pin will reset the controller.

**Analog Pins.** There are 8 analog pins on the board marked as A0 – A7. These pins are used to measure the analog voltage ranging between 0 to 5V.

**Rx, Tx.** These pins are used for serial communication where Tx represents the transmission of data while Rx represents the data receiver.

**REF.** This pin is used as a reference voltage for the input voltage.

**PWM.** Six pins 3,5,6,9,10, 11 can be used for providing 8-pit PWM (Pulse Width Modulation) output. It is a method used for getting analog results with digital sources.

**SPI.**  Four pins 10(SS),11(MOSI),12(MISO),13(SCK) are used for SPI (Serial Peripheral Interface). SPI is an interface bus and mainly used to transfer data between microcontrollers and other peripherals like sensors, registers, and SD card.

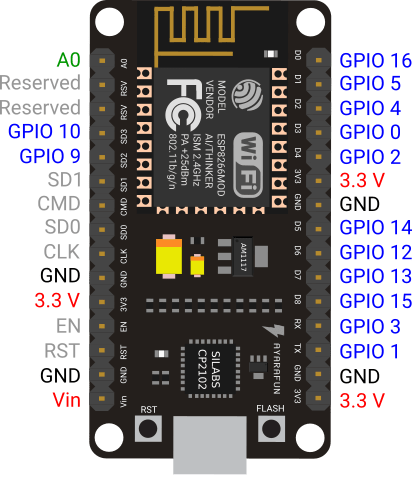
**External Interrupts.** Pin 2 and 3 are used as external interrupts which are used in case of emergency when we need to stop the main program and call important instructions at that point. The main program resumes once interrupt instruction is called and executed.

**I2C.** I2C communication is developed using A4 and A5 pins where A4 represents the serial data line (SDA) which carries the data and A5 represents the serial clock line (SCL) which is a clock signal, generated by the master device, used for data synchronization between the devices on an I2C bus.

### 3.2.2 NodeMCU

* NodeMCU is an open-source firmware and development kit that plays a vital role in designing your own IoT product using a few Lua script lines.
* The module is mainly based on [ESP8266](https://www.theengineeringprojects.com/2018/08/introduction-to-esp8266.html) that is a low-cost Wi-Fi microchip incorporating both a full TCP/IP stack and microcontroller capability.
* Multiple GPIO pins on the board allow you to connect the board with other peripherals and are capable of generating PWM, I2C, SPI, and UART serial communications.
* The interface of the module is mainly divided into two parts including both Firmware and Hardware where former runs on the ESP8266 Wi-Fi SoC and later is based on the ESP-12 module.
* The firmware is based on Lua – A scripting language that is easy to learn, giving a simple programming environment layered with a fast scripting language that connects you with a well-known developer community.
* And open source firmware gives you the flexibility to edit, modify and rebuilt the existing module and keep changing the entire interface until you succeed in optimizing the module as per your requirements.
* USB to UART converter is added on the module that helps in converting USB data to UART data which mainly understands the language of serial communication.
* Instead of the regular USB port, MicroUSB port is included in the module that connects it with the computer for dual purposes: programming and powering up the board.
* The board incorporates status LED that blinks and turns off immediately, giving you the current status of the module if it is running properly when connected with the computer.
* We use ARDUINO IDE software for programming this module.

**PINOUT**

****

**FIGURE: 3.2(b) NodeMCU Pinout**

## 3.3 Components and Environment Used

### 3.3.1 Arduino IDE

The [Arduino](https://en.wikipedia.org/wiki/Arduino) Integrated Development Environment ([IDE](https://en.wikipedia.org/wiki/Integrated_development_environment)) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform)  open source application (for [Windows](https://en.wikipedia.org/wiki/Windows), [mac OS](https://en.wikipedia.org/wiki/MacOS), [Linux](https://en.wikipedia.org/wiki/Linux)) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards. The environment is written in Java and based on Processing and other open-source software.

### 3.3.2 Firebase Console

Firebase is a technology that allows you to create web applications without server-side programming, making development faster and easier. It supports Web, iOS, OS X and Android clients. Apps that use Firebase can use and control data without thinking about how data is stored and synchronized across different instances of the application in real-time.

### 3.3.3 Android Studio

Android Studio is Android's official IDE. It is purpose built for Android to accelerate your development and help you build the highest-quality apps for every Android device. It offer tools custom-tailored for Android developers, including rich code editing, debugging, testing, and profiling tools.

### 3.3.4 Adobe XD

Adobe XD is a [vector-based](https://en.wikipedia.org/wiki/Vector_graphics_editor) [user experience design tool](https://en.wikipedia.org/wiki/User_experience_design) for [web apps](https://en.wikipedia.org/wiki/Web_app) and [mobile apps](https://en.wikipedia.org/wiki/Mobile_app), developed and published by [Adobe Inc](https://en.wikipedia.org/wiki/Adobe_Inc.). It is available for [macOS](https://en.wikipedia.org/wiki/MacOS) and [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), although there are versions for [iOS](https://en.wikipedia.org/wiki/IOS) and [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) to help preview the result of work directly on mobile devices. XD supports [website wireframing](https://en.wikipedia.org/wiki/Website_wireframe) and creating click-through prototypes.

## 3.4 Languages and packages used

### 3.4.1 Java

The Java language is a key pillar in [Android](https://en.wikipedia.org/wiki/Android_(operating_system)), an [open source](https://en.wikipedia.org/wiki/Open_source_software) [mobile operating system](https://en.wikipedia.org/wiki/Mobile_operating_system). Although Android, built on the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel), is written largely in C, the [Android SDK](https://en.wikipedia.org/wiki/Android_software_development#SDK) uses the Java language as the basis for Android applications but does not use any of its standard GUI, SE, ME or other established Java standards. The bytecode language supported by the Android SDK is incompatible with Java bytecode and runs on its own virtual machine, optimized for low-memory devices such as [smartphones](https://en.wikipedia.org/wiki/Smartphone) and [tablet computers](https://en.wikipedia.org/wiki/Tablet_computer). Depending on the Android version, the bytecode is either interpreted by the [Dalvik virtual machine](https://en.wikipedia.org/wiki/Dalvik_virtual_machine) or compiled into native code by the [Android Runtime](https://en.wikipedia.org/wiki/Android_Runtime).

Android does not provide the full Java SE standard library, although the Android SDK does include an independent implementation of a large subset of it. It supports Java 6 and some Java 7 features, offering an implementation compatible with the standard library.

### 3.4.2 Kotlin

**Kotlin** is a statically typed, general-purpose programming language developed by **JetBrains**, that has built world-class IDEs like IntelliJ IDEA, PhpStorm, Appcode, etc. It was first introduced by JetBrains in 2011 and a new language for the JVM. Kotlin is object-oriented language, and a “better language” than Java, but still be fully interoperable with Java code.

Kotlin is sponsored by **Google**, announced as one of the official languages for **Android** Development in 2017.  It is focused on interoperability, safety, clarity, and tooling support.

### 3.4.3 C++

**C++** is a [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language) created by [Bjarne Stroustrup](https://en.wikipedia.org/wiki/Bjarne_Stroustrup) as an extension of the [C programming language](https://en.wikipedia.org/wiki/C_(programming_language)), or "C with [Classes](https://en.wikipedia.org/wiki/Class_(programming))". The language has expanded significantly over time, and modern C++ has [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [generic](https://en.wikipedia.org/wiki/Generic_programming), and [functional](https://en.wikipedia.org/wiki/Functional_programming) features in addition to facilities for [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) [memory](https://en.wikipedia.org/wiki/Memory_(computing)) manipulation. C++ was designed with a bias toward [system programming](https://en.wikipedia.org/wiki/System_programming) and [embedded](https://en.wikipedia.org/wiki/Embedded_software), resource-constrained software and large systems, with [performance](https://en.wikipedia.org/wiki/Performance_(software)), efficiency, and flexibility of use as its design highlights.

The language supported by the Arduino IDE is basically C++ with some additional features implemented by the Arduino programmers. The Arduino language has a lot of abstraction built in, especially in the hardware interfaces, which makes it very simple to use.

### 3.4.4 Maps SDK for Android

With the Maps SDK for Android, you can add maps based on Google Maps data to your application. The API automatically handles access to Google Maps servers, data downloading, map display, and response to map gestures. You can also use API calls to add markers, polygons, and overlays to a basic map, and to change the user's view of a particular map area. These objects provide additional information for map locations, and allow user interaction with the map. The API allows you to add these graphics to a map:

* Icons anchored to specific positions on the map (Markers)
* Sets of line segments (Polylines).
* Enclosed segments (Polygons).
* Bitmap graphics anchored to specific positions on the map (Ground Overlays).
* Sets of images which are displayed on top of the base map tiles (Tile Overlays).

### 3.4.5 GeoFire

GeoFire is an open-source library for Android that allows you to store and query a set of keys based on their geographic location.

At its heart, GeoFire simply stores locations with string keys. Its main benefit however, is the possibility of querying keys within a given geographic area - all in realtime.

GeoFire uses the [Firebase Realtime Database](https://firebase.google.com/products/realtime-database/) for data storage, allowing query results to be updated in realtime as they change. GeoFire selectively loads only the data near certain locations, keeping your applications light and responsive, even with extremely large datasets.

GeoFire is designed as a lightweight add-on to the Firebase Realtime Database. However, to keep things simple, GeoFire stores data in its own format and its own location within your Firebase database. This allows your existing data format and security rules to remain unchanged and for you to add GeoFire as an easy solution for geo queries without modifying your existing data.

### 3.4.6 ESP8266 Arduino core

This project brings support for the ESP8266 chip to the Arduino environment. It lets you write sketches, using familiar Arduino functions and libraries, and run them directly on ESP8266, with no external microcontroller required.

ESP8266 Arduino core comes with libraries to communicate over WiFi using TCP and UDP, set up HTTP, mDNS, SSDP, and DNS servers, do OTA updates, use a file system in flash memory, and work with SD cards, servos, SPI and I2C peripherals.

ESP8266 is all about Wi-Fi. The [Wi-Fi library for ESP8266](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi) has been developed based on [ESP8266 SDK](https://bbs.espressif.com/viewtopic.php?f=51&t=1023), using the naming conventions and overall functionality philosophy of the [Arduino WiFi library](https://www.arduino.cc/en/Reference/WiFi). The scope of functionality offered by the [ESP8266WiFi](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi) library is quite extensive.

### 3.4.7 Firebase Arduino

Firebase Arduino is a library to simplify connecting to the Firebase database from Arduino clients.

It is a full abstraction of Firebase’s REST API exposed through C++ calls In a wiring friendly way. All Json parsing is handled by the library and you may deal in pure C/Arduino types.

Firebase Arduino now depends on [ArduinoJson library](https://github.com/bblanchon/ArduinoJson) instead of containing it's own version of it.

# 4 IMPLEMENTATION

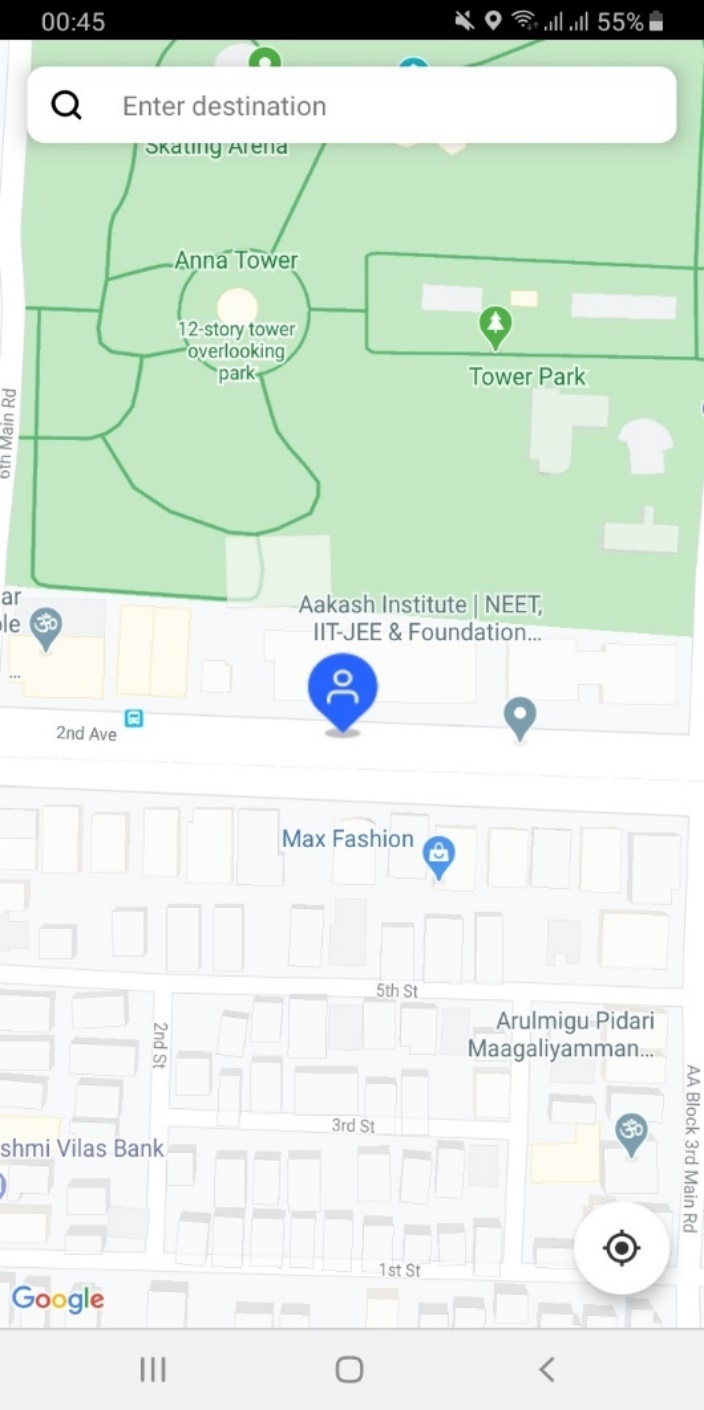
* A simulation of an ambulance following the nearest traced route to the destination entered is done in the android application.
* The android application is connected to a realtime cloud server (Firebase) and the realtime data of the location(coordinates) of the ambulance along with the signal values of the signals present in a junction are constantly updated.
* Microcontrollers are programmed to get the realtime database signal values and feed them as inputs to the LEDs through GPIO pins present in the microcontrollers.

## 4.1 Building Android application

A set of functionalities were implemented to develop this application. They are briefly described below.

### 4.1.1 Detecting current location

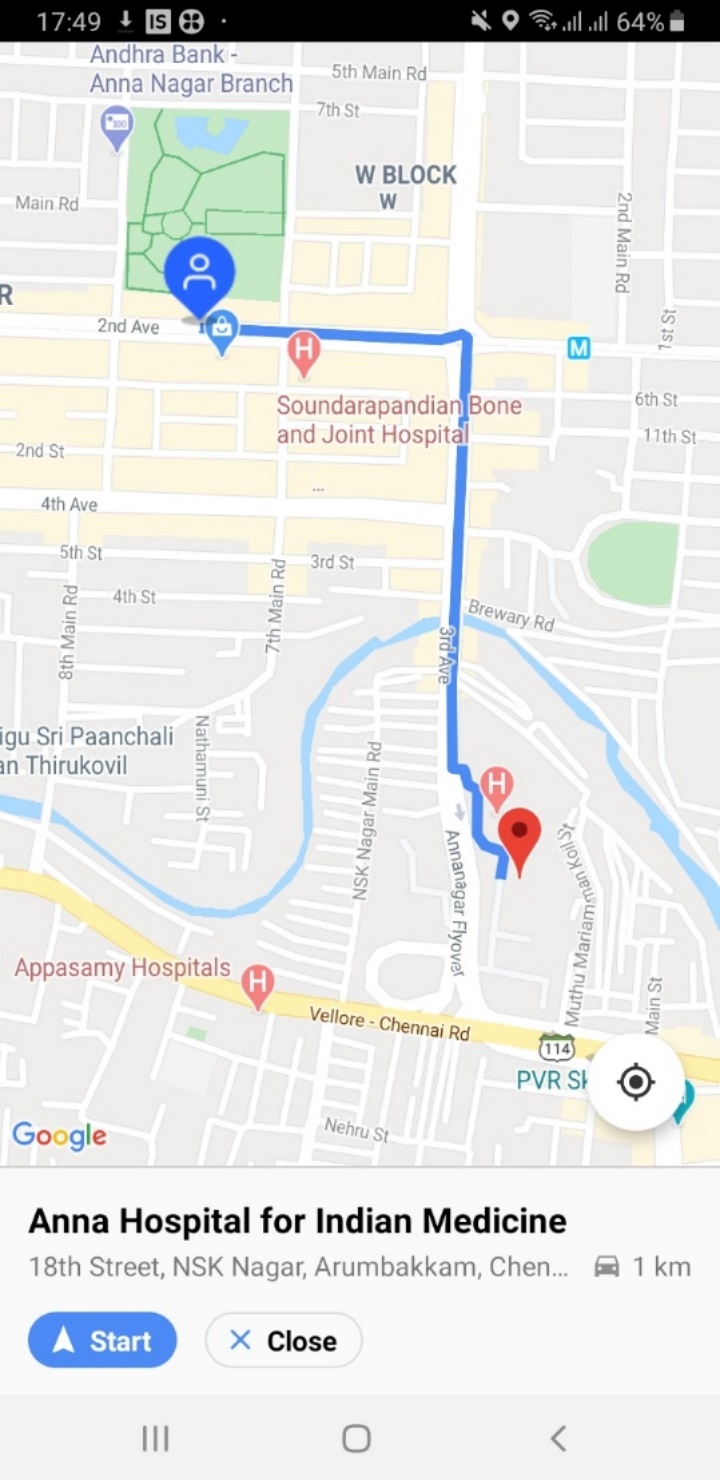
* Maps layout is used to form the base layout for this application where Google maps API is used to extract the world map and use it in our application.
* The current location of the user is detected using the locations API in android.
* The location API available in Google Play services facilitate adding location awareness to your app with automated location tracking, geofencing, and activity recognition.
* A marker icon is set to mark the location of the user so that it can be easily identified by the user using the app.



**FIGURE: 4.1(a) Current location**

### 4.1.2 Generating polyline

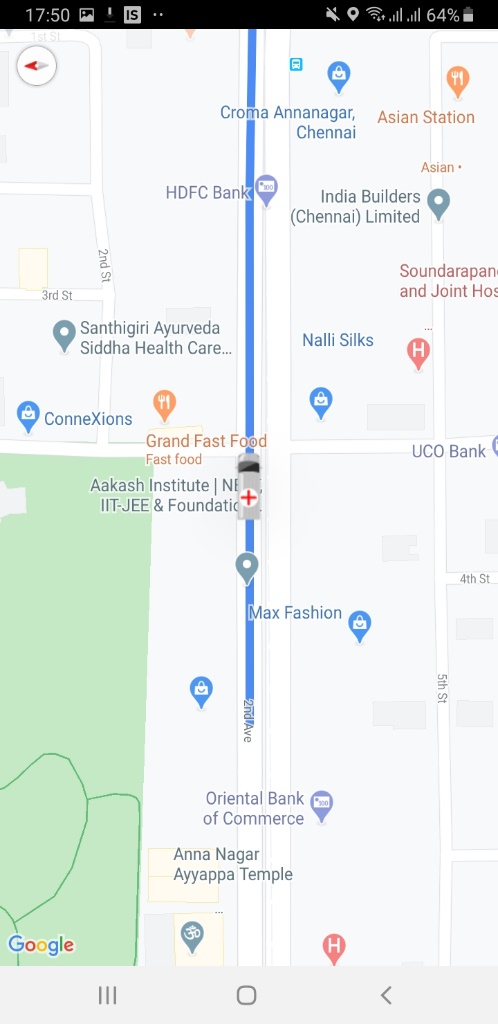
* Google directions API and Google places SDK for Android are used to serve this purpose.
* The Directions API is a service that calculates directions between locations using a HTTP request.
* Directions API helps users find the best way to get from A to Z with comprehensive data and real-time traffic.
* The Places SDK for Android allows us to build location-aware apps that respond contextually to the local businesses and other places near the user's device.
* The polyline is generated by taking into account the maximum number of points (the coordinates) in between the start and end destination and building a path that connects all such points.



**FIGURE: 4.1(b) Polyline**

### 4.1.3 Simulation with Animation Package

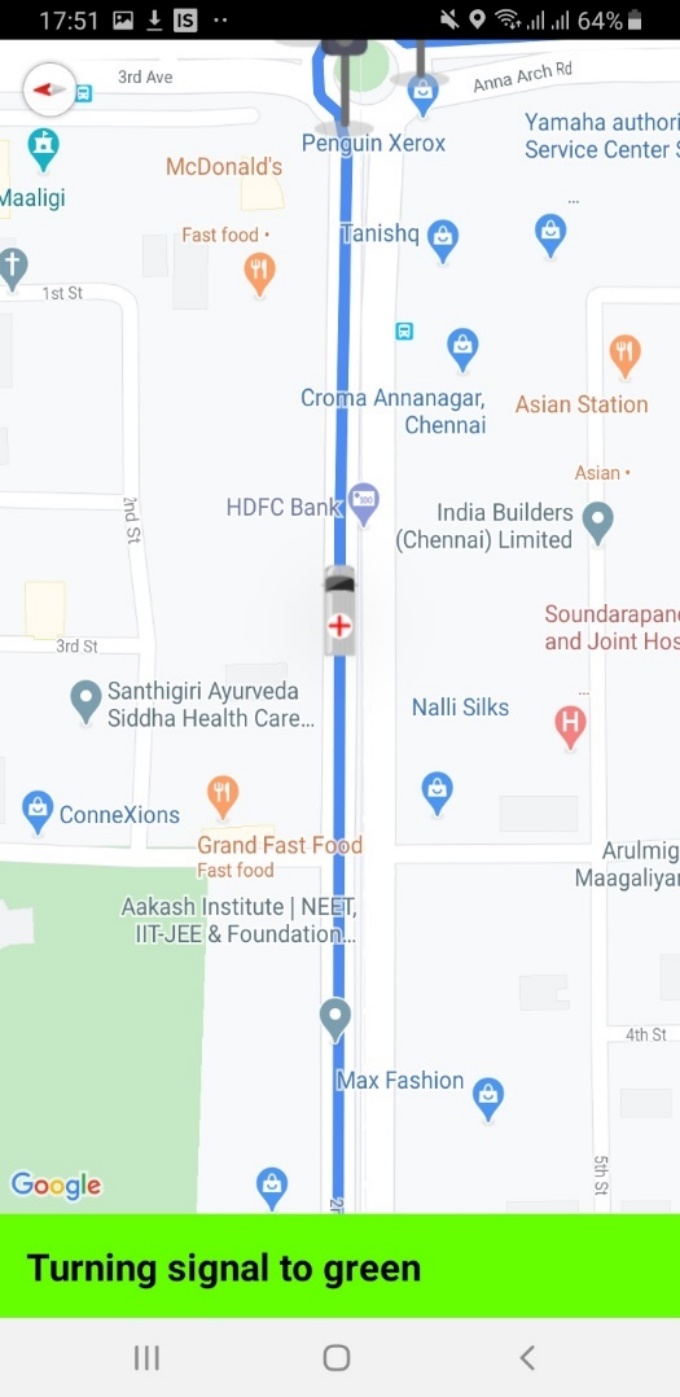
* The Animator class provides the basic structure for creating animations.
* A Subclass of the Animator class (ValueAnimator) is used which is the main timing engine for property animation that also computes the values for the property to be animated.
* Bearing/Heading values represent the degree in which the ambulance is pointing towards in the map. It can range from 0 to 360 degrees which depicts the direction in which the ambulance is heading towards.
* Many location points exist between the start and the end location which when joined together form the polyline.
* Bearing/Heading values of the current location are used to simulate the motion of an ambulance.
* When the simulation starts the ambulance heads towards each of these location points through animation which is done by the animator class.
* In this similar fashion the ambulance simulates from the start point to the end point.



**FIGURE: 4.1(c) Simulation**

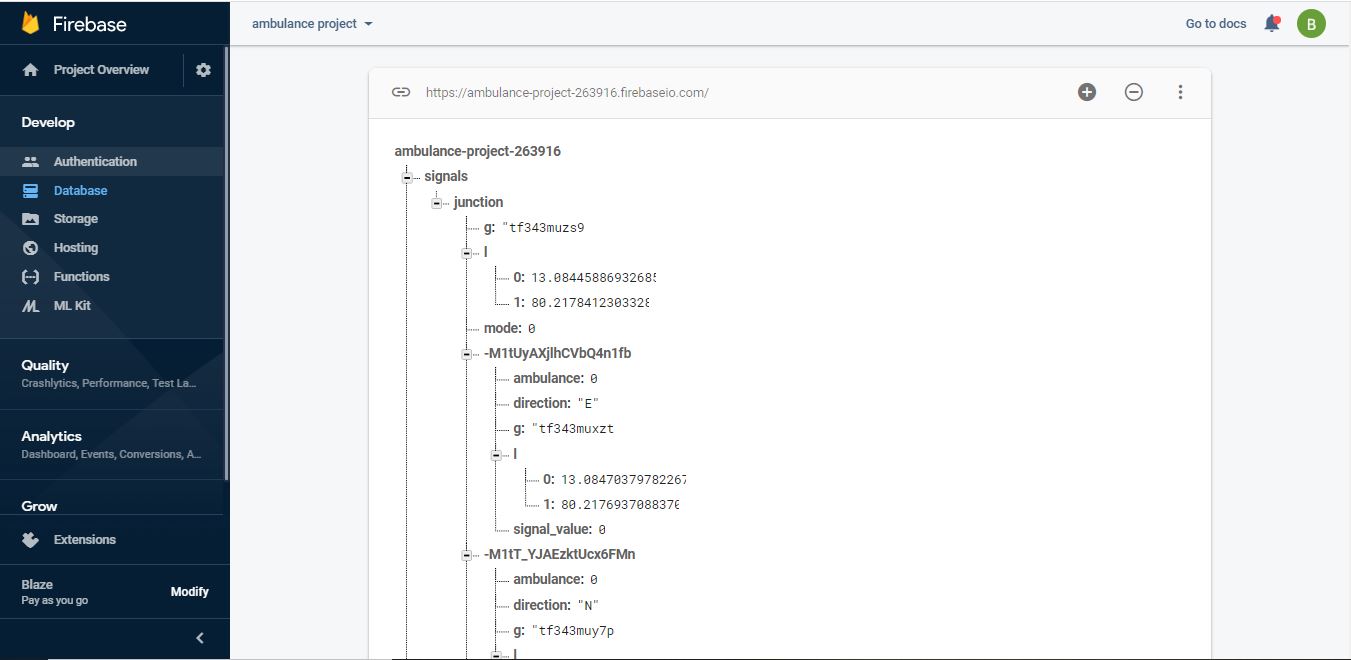
### 4.1.4 Integrating Firebase using GeoFire

* GeoFire (library for android) uses geoquery that queries a set of keys in the realtime database(Firebase).
* The application scans the database for signals within a specified radius of 500 metres from the current location regularly.
* When the ambulance comes under the proximity of a signal an alert is displayed notifying the signal turning to green when the ambulance is 100 metres away from the signal.



**FIGURE: 4.1(d) Notification**

* A database reference is used to either get the signal value of the signals from Firebase or update Firebase with new signal values.
* As the current location of the ambulance moves towards the junction where the signals are present, the nearest signal is accessed and the signal values of all the signals present in the junction are changed accordingly which is reflected immediately in Firebase.
* The signal present in the direction in which the ambulance is approaching is changed to green and the rest all signals are changed to red.
* The signal flow is restored once the ambulance crosses the junction.
* All these real time changes are reflected in Firebase as shown below.

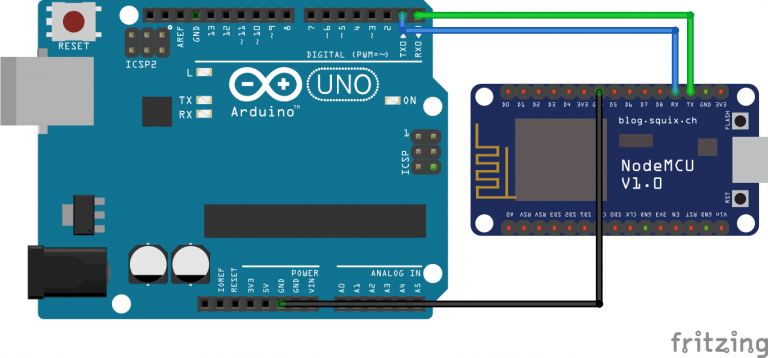


**FIGURE: 4.1(e) Firebase Console**

## 4.2 Building IoT Module

### 4.2.1 Software Serial communication between NANO and NodeMCU

* Serial communication is needed to make use of all the pins in both the boards. Data can be transmitted in the form of bytes serially.
* A circuit as per the given diagram is made after uploading the program. The RX pin of NodeMCU is connected to the TX pin of Arduino Nano. Also, the TX pin of NodeMCU is connected to the RX pin of Arduino Nano. Both the Grounds of NodeMCU and Arduino are common as per diagram.



**FIGURE 4.2(a) Serial Communication between Nano and NodeMCU**

* Serial.write(data) - is the code for Sender NodeMCU. First the SoftwareSerial Library is included and  SoftwareSerial at RX and TX pin is started. In Setup, SoftwareSerial is started at 9600 Baud.
* Serial.read() - is code for Receiver Arduino. First a variable s declared to store received data. In Setup, Serial Begin at 9600 Baud. In the loop, the serial data is read and stored it in a variable.
* The code is uploaded to NodeMCU and Arduino Board. A circuit as per the given diagram is made after uploading code. ‘data’ will be transmitted from Sender NodeMCU to Receiver Arduino. Similarly all the data can be transferred in such a similar way.
* Using this method 16 IO pins can be utilized on both the boards to control individual LEDs in the traffic signal.
* The data is first brought into the NodeMCU from the server and is then the serially communicated to the Nano board.

### 4.2.2 WiFi connection for ESP8266 – NodeMCU

Connection to a wireless access point is relatively straightforward: the SSID and the password of the network to be connected to is entered, and the WiFi.begin function is called to establish it. Then wait for the connection to complete, the ESP8266 is now connected to the Local Area Network. The coding is done in the Arduino IDE itself by choosing the device as NodeMCU and a desired Port.

#include <ESP8266WiFi.h> - header to be included

const char\* ssid = "SSID" – SSID of the LAN

const char\* password = "PASSWORD" – password of the LAN

**WiFi**.begin(ssid, password) - connect to the Wifi

Using this WiFi connection, we can access data from the server (firebase) in realtime to operate the traffic signals accordingly.

### 4.2.3 Connecting Arduino to Firebase

It is possible to read or transfer data from your database (firebase) to Arduino and ESP8266. A Host name and an authentication key of your firebase project is needed. Then, the Firebase Arduino library is added and the code is uploaded.

Firebase Arduino-Master is a library to simplify connecting to the Firebase database from arduino clients.

It is a full abstraction of Firebase’s REST API exposed through C++ calls in a wiring friendly way. All Json parsing is handled by the library and you may deal in pure C/Arduino types. A few parameters are essential to connect it to firebase.



**FIGURE: 4.2(b) Flow Diagram**

Parameters

* Host : The firebase database host, usually X.firebaseio.com.
* Auth: Optional credentials for the db, a secret or token.

The value stored at a given Firebase database key can actually be a structure containing a mixture of numeric and string fields, which is useful for representing richer data structures.

String getString (const String &path) can be used to access the individual children from the database

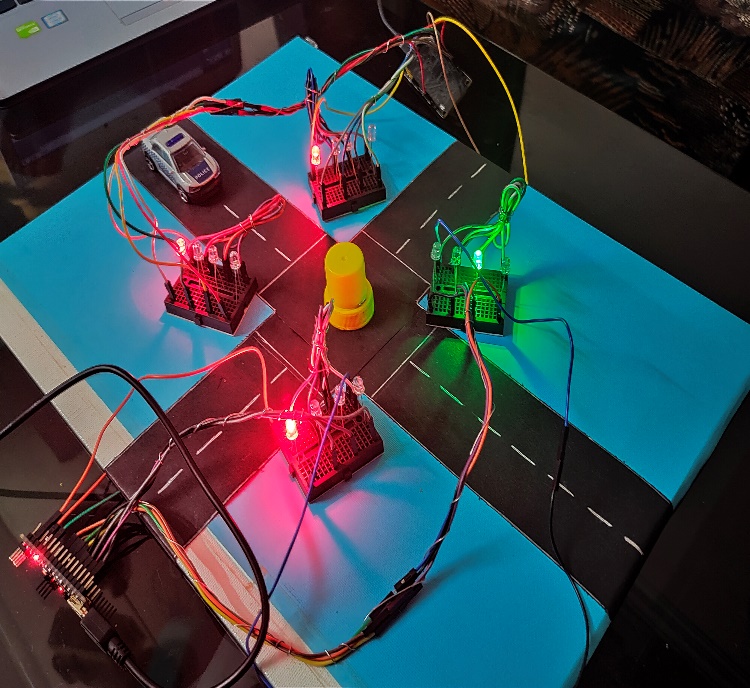
* Gets the string value located at path.
* Necessary to check the success() after calling. This ensures that the retrieval from the database is successful

It is important to use the newest fingerprint available and update it in the master before connection.

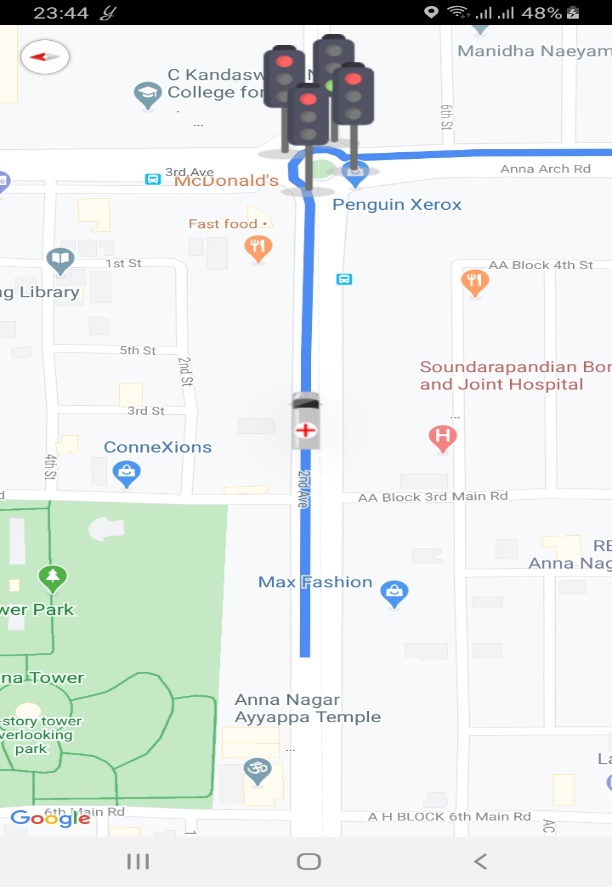
# 5 RESULT AND ANALYSIS

In this section, we evaluate the results of the proposed system.

* The traffic signals at the junction will function in their regular states as long as there is no ambulance within their proximity.

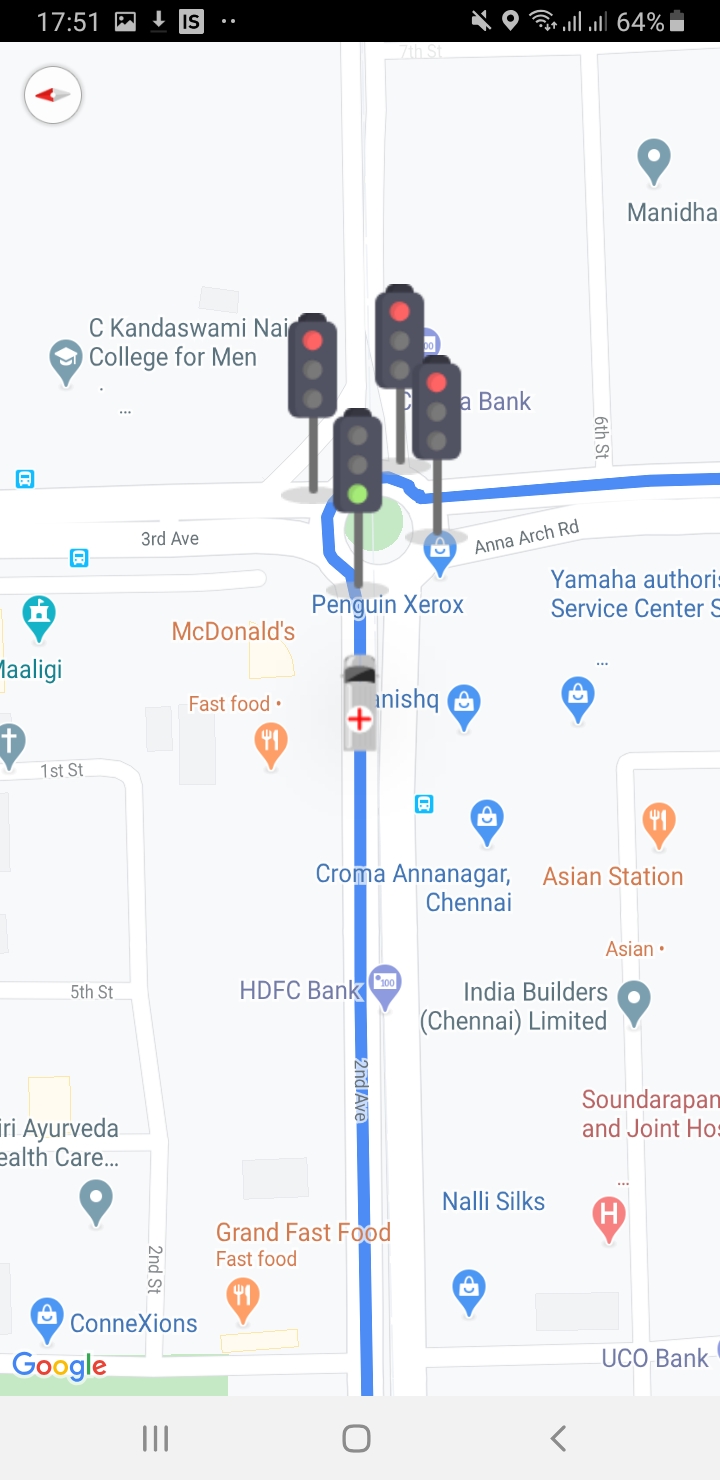


**FIGURE: 5.1: Iot Simulation before signal changes**



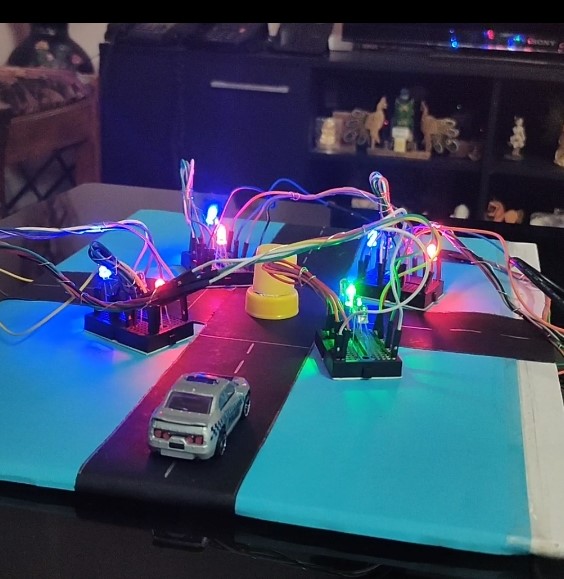
**FIGURE: 5.2: In App Before Signal Changes**

* The driver enters the desired destination and starts the navigation. When an ambulance reaches the 100m proximity threshold, the signals will begin to function in emergency mode. Using the GPS location the bearing and offset value of the approaching ambulance is calculated and the desired signal will turn green while the remaining ones turn red simultaneously allowing smooth passage.



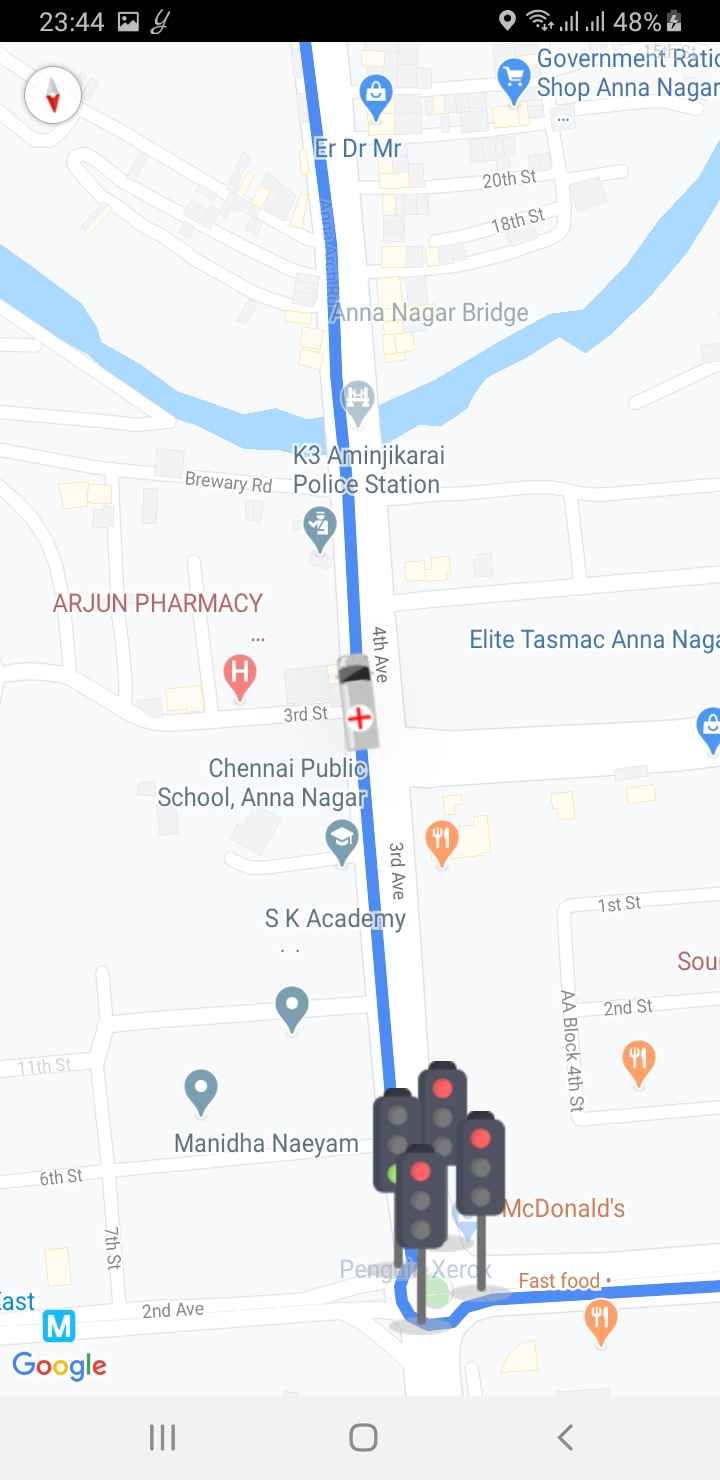
**FIGURE: 5.3: In App Signals are changed**

* A blue stroboscopic LED fitted in each signal will also glow at the same time to intimate the surrounding vehicles of the incoming emergency vehicle.



**FIGURE: 5.4 Blue Stroboscopic LED**

* Once the ambulance crosses the junction, which is known using a distance measure, the traffic signal will return back to their previous state.



**FIGURE: 5.5: In App Signals are restored**

# 6 CONCLUSION AND FUTURE SCOPE

## 6.1 Conclusion

In this project, we have demonstrated the working prototype of “Smart Ambulance Guidance System” by integrating IoT, an application and a server which is suitable to run on any smartphone. The goal of this project was to address the difficulties faced by emergency vehicles like ambulances to bypass heavy traffic and road congestions in countries like India. The project aims at providing a solution to solve this issue. A central server, Firebase is used to monitor the states of the traffic signals by connecting the android application and IoT module . The app provides the current location of the ambulance to the server. When the ambulance approaches within proximity of the junction, the LEDs of the traffic signals are changed accordingly to allow smooth passage. A stroboscopic light is also added to intimate the surrounding vehicles about the incoming ambulance. The traffic lights return to their previous states after the ambulance passes through.

## 6.2 Future Scope

Further this project can be improved by sending the patient details to nearby hospitals. Also a cross platform application can be developed using web apps allowing it to run on different kind of devices.

# 7 REFERENCES

1. Intelligent Ambulance with Traffic Control (Gargi Beri, Pankaj Ganjare, Amruta Gate, Ashwin Channawar, Vijay Gaikwad)
2. An Intelligent Ambulance with some Advance features of Telecommunication (Pratyush Parida, Sudeep Kumar Dhurua, P. Santhi Priya)
3. Automated Emergency System in Ambulance to Control Traffic Signals using IoT (Dr. A. Balamurugan, G. Navin Siva Kumar, S. Raj Thilak, P. Selvakumar)
4. Smart Traffic Control System Using Image Processing (Prashant Jadhav, Pratiksha Kelkar, Kunal Patil, Snehal Thorat)
5. Ambulance Controlled Traffic System Using RFID Technology with LabVIEW Simulation (S. Chandrakanth Sagar, Dr. M. Narayana)
6. Intelligent Ambulance Rescue System (Nangare Yogini K. and Prof. Hate S.G)
7. Mr.S.Iyyappan 1, Mr.V.Nandagopal “Automatic Accident Detection And Ambulance Rescue with Intelligent Traffic Light System”
8. Mrs Manasi Patil, Aanchal Rawat, Prateek Singh, Srishtie dixit” Accident Detection and Ambulance Control using Intelligent Traffic Control System”
9. Prof.Mrs.Bhagya Lakshmi V, Prof.Savitha Hiremath, Prof.Sanjeev Mhamane “ FPGA Based Vehicle Tracking and Accident Warning using GPS”
10. Niketa Chellani, Chirag Tahilyani Student, Electronics and Telecommunication Dept., TSEC, Mumbai University “ Traffic Congestion Detection and Control using RFID Technology “