A Project Report on

SMART EMERGENCY SAVING SYSTEM

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

D.SRI LAKSHMI PRASANNA	**16541A0509**
N.S.S.S.PRAVALLIKA	**16P31A0538**
S.B.KIRAN REDDY	**16P31A0548**
T.KARTHIK KUMAR	**16P31A0552**

Under the esteemed supervision of

Mrs. Sarita Kilarapu, M. Tech

Assistant Professor, CSE Department

Department of Computer Science and Engineering

ADITYA COLLEGE OF ENGINEERING AND TECHNOLOGY (Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada) Surampalem, East Godavari District Andhra Pradesh - 533437

2016-2020

ADITYA COLLEGE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE, New Delhi & Affiliated to JNTUK, Kakinada) Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work entitled, "**SMART EMERGENCY SAVING SYSTEM**", is a bonafide work carried out by **D.SRI LAKSHMI PRASANNA (16541A0509), N.S.S.S.PRAVALLIKA (16P31A0538), S.B.KIRAN REDDY (16P31A0548), T.KARTHIK KUMAR (16P31A0552)** in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in **COMPUTER SCIENCE AND ENGINEERING** from Aditya College of Engineering and Technology, during the academic year 2016-2020.

PROJECT SUPERVISOR
Mrs. Sarita Kilarapu, M. Tech
Assistant Professor

HEAD OF THE DEPARTMENT
Mr. M. Anil Kumar, M. Tech, (Ph. D)
Associate Professor

External Examiner

DECLARATION

We hereby declare that this project entitled "**SMART EMERGENCY SAVING SYSTEM**" has been undertaken by us and this work has been submitted to **ADITYA COLLEGE OF ENGINEERING AND TECHNOLOGY** affiliated to JNTUK, Kakinada, in partial fulfillment of the requirements for the award of the degree of Bachelor of technology in **COMPUTER SCIENCE AND ENGINEERING.**

We further declare that this project work has not been submitted in full or part for the award of any degree of this in any other educational institutions.

PROJECT ASSOCIATES

D.SRI LAKSHMI PRASANNA (16541A0509)

N.S.S.S.PRAVALLIKA (16P31A0538)

S.B.KIRAN REDDY (16P31A0548)

T.KARTHIK KUMAR (16P31A0552)

ACKNOWLEDGEMENT

It is with immense pleasure that we would like to express our indebted gratitude to our supervisor **Mrs. Sarita Kilarapu, M. Tech**, Assistant Professor who has guided us a lot and encouraged us in every step of project work, his valuable moral support and guidance throughout the project helped us greater extent.

We wish to express our sincere thanks to **Mr. M. Anil Kumar, M. Tech, (Ph. D),Head of the Department of CSE** for his valuable guidance given to us throughout the period of the project.

We feel elated to thank vice principal, **Dr. A. Rama Krishna** of **ADITYA COLLEGE OF ENGINEERING AND TECHONOLOGY**, for his cooperation and help in completion of our project and throughout our course.

We feel elated to thank principal, **Dr. T. K. Rama Krishna Rao** of **ADITYA COLLEGE OF ENGINEERING AND TECHONOLOGY** for their cooperation and helping completion of our project and throughout our course.

We wish to express our sincere thanks to all faculty members, lab programmers for their valuable guidance given to us throughout the period of the project.

We avail this opportunity to express our deep sense and heart full thanks to the **Management** of **ADITYA COLLEGE OF ENGINEERING AND TECHONOLOGY** for providing a great support for us in completing our project by arranging the trainees, and facilities needed to complete our project and for giving us the opportunity of doing the project.

PROJECT ASSOCIATES

D.SRI LAKSHMI PRASANNA (16541A0509)

N.S.S.S.PRAVALLIKA (16P31A0538)

S.B.KIRAN REDDY (16P31A0548)

T.KARTHIK KUMAR (16P31A0552)

ABSTRACT

According to Government of India, around 1,06,000 people lost their lives in three hundred and fifty thousand of accidents in 2013. In 2018, around 1,46,000 people lost their lives in five hundred thousand of accidents, where 7% of lives could have been saved if they would have got medical attention before-hand(Just within the span of five years). As the usage of vehicles is increasing drastically, the hazards due to vehicles is also increased. The main cause for accidents is high speed, drunk and drive, diverting minds, over stress and due to electronic gadgets. This project deals with accident detection system that occurs due to carelessness of the person who is driving the vehicle. If the person is not in a position to control the vehicle then the accident occurs. Once the accident occurs to the vehicle this system will send information to registered mobile number. For detecting accident we will use accident detection device which will be already attached to the vehicle to know the information of accident right away. Hence they can be transported to hospital immediately. Hence we could reduce death rate due accidents.

CONTENTS

Figure	PAGE NO

Table

PAGE NO

Chapter -1

INTRODUCTION

1.INTRODUCTION

1.1 Introduction

In present days the rate of accidents can be increased rapidly. Due to employment the usage of vehicles like cars, bikes can be increased, because of this reason the accidents can be happened due to over speed. Around 1,46,000 people lost their lives in five hundred thousand of accidents, where 7% of lives could have been saved if they would have got medical attention before-hand(Just within the span of five years).

The proposed system will check whether an accident has occurred and notifies to ambulance system, police station and registered mobile numbers about the place of accident using GSM and GPS modules. The location can be sent through tracking system to cover the geographical coordinates over the area.

The accident can be detected by a Accelerometer sensor which is used as major module in the system. Hence we decided to implement this project. Our project is IOT based. IOT means Internet of Things which connects devices with each oth1.2 Current Scenario through internet. Here we will attach accident detection device to the vehicles.

If there was any accident, it detects right away and sends the information about accident and accident location to the ambulance system through internet.

For receiving all the information and for the user interaction, we will create platforms such as dashboard app, website.

1.2 Current Scenario

The existing ambulance system was not effective enough to save lives and they are not quick. In the existing ambulance system, the information of the accident occurrence and the accident location has to be informed manually by someone. Due to these drawbacks in the existing ambulance system, many people were losing their lives helplessly.

Fig 1.2.1 Existing system

1.3 Problem Statement

Around 1,46,000 people lost their lives in five hundred thousand of accidents, where 7% of lives could have been saved if they would have got medical attention before-hand(Just within the span of five years).

People are going under risk because of their over speed, due to unavailability of advanced techniques, the rate of accidents can't be decreased.

The main objective is to send a message to the registered mobile using wireless communications techniques.

Implementation cost of project:

Arduino Uno board	=	Rs. 470
GPS module (NEO-6M)	=	Rs. 1100
Wi-Fi module (ESP8266)	=	Rs. 300
Accelerometer (MPU6050)	=	Rs. 300
GSM module (SIM900A)	=	Rs. 900
ESP8266 Adaptor (ESP-01)	=	Rs. 195
Climbing Car (toy)	=	Rs. 4000
Others	=	Rs. 900
Total cost of implementation	=	Rs. 8165

1.4 Objectives

- Provide the user with a sense of security
- Detecting an accident by effectively using the accelerometer and gyrometer
- Send real-time messages and inform to relevant parties of the system
- Provide information more accurately
- Generation of reports and graphical representation

1.5 Proposed Solutions

There are two applications one for server and the other for the client. We will attach this accident detection device to the vehicles which detects accident basing on acceleration due to accident and also detects the impact of accident. The information about accident sends to the ambulances that are nearest to that location along with the details of the location and the impact using internet. Hence ambulances can reach there and transport patient to hospital immediately.

1.6 Organization of Project

1. Chapter 1 introduces project. It describes problem statement, current scenario, proposed solutions.
2. Chapter 2 states about the past work that have done related to the developed project.
3. Chapter 3 explains hardware and software used in the project.
4. Chapter 4elaborates working principle, system architecture and circuit design of the project.
5. Chapter 5 deals with System Design.
6. Chapter 6 implementation of the Project.
7. Chapter 7 testing results and output screens of the project are discussed.

Chapter-2

Review of Past Work

2. REVIEW OF PAST WORK

Existing System

To protect the vehicle and tracking so many advanced technologies are available in now a days. In olden days the information of accident can be Revised Manuscript Received on December 22, 2018. T Kalyani, Assistant Professor, Department of Electronics and Communications Engineering, MLR Institute of Technology, Hyderabad, Telangana, India. S Monika, Assistant Professor, Department of Electronics and Communications Engineering, MLR Institute of Technology, Hyderabad, Telangana, India.

B Naresh, Assistant Professor, Department of Electronics and Communications Engineering, MLR Institute of Technology, Hyderabad, Telangana, India. Mahendra Vucha, Assistant Professor, Department of Electronics and Communications Engineering, MLR Institute of Technology, Hyderabad, Telangana, India. Transferred, but the place of accident spot cannot be identified. In any vehicle airbags are designed, air bags are used for security and safety travels. The air bag system was introduced in the year of 1968.

Technologies used in the Past So far:

1.GSM +GPS : It is one of effective method used in the past, but the main drawback it won't give you the status acknowledgement and features the like tracking, it will give you the coordinates of Long, Lat. So it will need extra work to get the live location as well.

2.IOT (without UI): The main advantage of this method would be it can give you the status acknowledgement and features the like tracking without extra work, but the device can't handle more mobile numbers (once installed you can't add a new number) to which it is going to send the SMS.

CHAPTER-3

Software and Hardware Requirements

3.Software and Hardware Requirements

3.1 Platform and Cloud services

ThingSpeak: This is an open, IoT platform with MATLAB support. The core of this platform is the IoT analytics and data visualization. It provides real-time data visualization, and with the support of MATLAB, it is possible to add data analysis and processing. Internet of Things (IoT) describes an emerging trend where a large number of embedded devices (things) are connected to the Internet. These connected devices communicate with people and other things and often provide sensor data to cloud storage and cloud computing resources where the data is processed and analysed to gain important insights. Cheap cloud computing power and increased device connectivity is enabling this trend.

IoT solutions are built for many vertical applications such as environmental monitoring and control, health monitoring, vehicle fleet monitoring, industrial monitoring and control, and home automation.

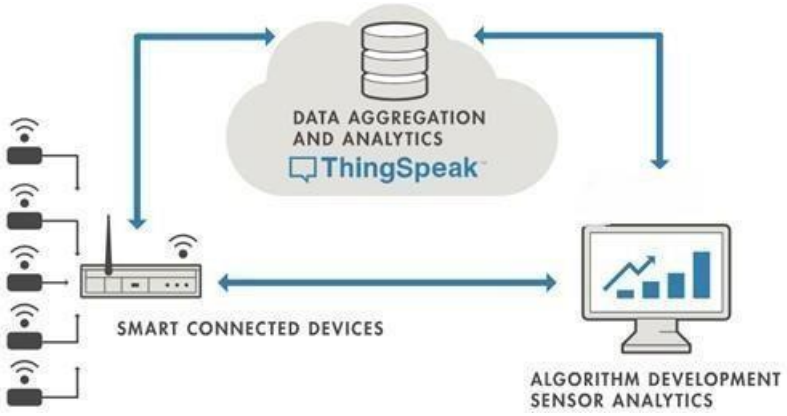


Fig 3.1.1 IoT System

The main features provided by this IoT platform includes:

- Real-time sensor data visualization

- Data aggregation from third-partyproviders
- Schedule IoT analytics tasks to analyse thedata
- Event scheduling
- Run actions according to data acquired

This Internet of Things platform supports several devices, such as Arduino, Raspberry Pi, ESP, Particle, and so on.

3.2 List of Hardware Components and their Specifications

Arduino UNO

ESP8266 (WIFI Module)

ESP v1 supporting Adaptor

MPU6050

GSM SIM900A

GPS

3.2.1 Arduino UNO

The name Arduino comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduino of Ivrea, who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014.

Fig 3.2.1.1 Arduino Board

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead()functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 KOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB- to-TTL Serial chip.

- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

3.2.2 ESP8266 (WIFI Module)

The **ESP8266** is a low-cost Wi-Fi microchip, with a full TCP/IP stack and microcontroller capability, produced by Espressif Systems in Shanghai, China.

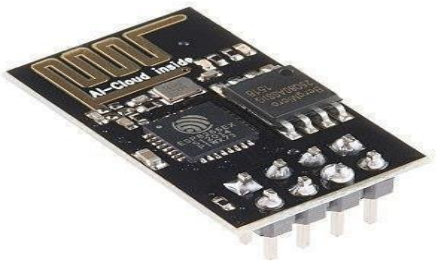


Fig 3.2.2.1 ESP8266 Physical View

- Processor: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz[5]
- Memory:
  - 32 KiB instruction RAM
  - 32 KiB instruction cache RAM
  - 80 KiB user-data RAM
  - 16 KiB ETS system-data RAM
- External QSPI flash: up to 16 MiB is supported (512 KiB to 4 MiB typically included)
- IEEE 802.11 b/g/n Wi-Fi
  - Integrated TR switch, balun, LNA, power amplifier and matching network
  - WEP or WPA/WPA2 authentication, or open networks
- 16 GPIO pins
- SPI
- I²C (software implementation)[6]
- I²S interfaces with DMA (sharing pins with GPIO)
- UART on dedicated pins, plus a transmit-only UART can be enabled on GPIO2
- 10-bit ADC (successive approximation ADC)

3.2.3 ESP v1 supporting Adaptor

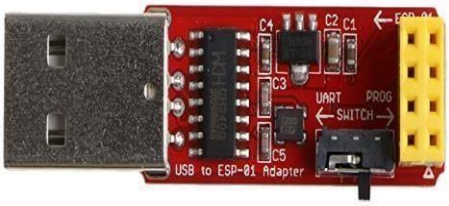


Fig 3.2.3.1 ESP8266 Programming adaptor

The pinout is as follows for the common ESP-01 module:

1. VCC, Voltage (+3.3 V; can handle up to 3.6 V)
2. GND, Ground (0 V)
3. RX, Receive data bit X
4. TX, Transmit data bit X
5. CH_PD, Chip power-down
6. RST, Reset
7. GPIO 0, General-purpose input/output No. 0
8. GPIO 2, General-purpose input/output No. 2

3.2.3.1 Features

USB to ESP-01 adapter module has CH340G USB to TTL driver IC onboard, so you can easily use your computer to do ESP-01 functional debugging. ESP- 01 serial WIFI module can be directly plugged into the yellow pin header without any connection wires.

Features:

- Working voltage: 4.5V - 5.5V (On-board 3.3v LDO Regulator)
- Working current: 300mA(LDO regulator can supply)
- Selectable working mode: On-board toggle switch. UART side for serial TTL debugging by AT commands, PROG for firmware programming
- USB to serial TTL chip: CH340G
- Logic level: 3.3V
- Applications: Home automation, sensor networks, industrial wirelesscontrol

Size: 5*1.4cm/1.97"*0.55"

3.2.4 MPU6050

Fig 3.2.4.1 MPU6050 Physical View

The MPU6050 IMU has both 3-Axis accelerometer and 3-Axis gyroscope integrated on a single chip.

The gyroscope measures rotational velocity or rate of change of the angular position over time, along the X, Y and Z axis. It uses MEMS technology and the Coriolis Effect for measuring. The outputs of the gyroscope are in degrees per second, so in order to get the angular position we just need to integrate the angular velocity. On the other hand, the MPU6050 accelerometer measures acceleration. So, if we fuse, or combine the accelerometer and gyroscope data we can get very accurate information about the sensor orientation.

There is a main eight PINOUT of MPU6050, which are described here:

Pin	Pin Name	Description
1	Vcc	This pin used for Supply Voltage. Its input voltage is +3 to +5V.
2	GND	This pin use for ground
3	SCL	This pin is used for clock pulse for I2C compunction
4	SDA	This pin is used for transferring of data through I2C communication.
5	Auxiliary Serial Data (XDA)	It can be used for other interfaced other I2C module with MPU6050.
6	Auxiliary Serial Clock (XCL)	It can also be used for other interfaced other I2C module with MPU6050.
7	AD0	If more than one MPU6050 is used a single MCU, then this pin can be used to vary the address.
8	interrupt (int)	This pin is used to indicate that data is available for MCU to read.

Table 3.2.4 MPU6050 PINOUT.

3.2.4.1 Features of MPU6050

These are some features of MPU6050.

- **MPU6050** is the world's first integrated six motion tracking device

- The communication protocol at which it operates is I2C.
- It is built in 16 BIT ADC, which provide high accuracy.
- Its operating voltage is 3 to 5 volts.
- It consists of a digital motion processor, which provide high computational power.
- It is inbuilt in the temperate sensor.
- It can be used to interfaces with IIC devices like magnetometer.
- The pitch of its pins is 0.1 inch.
- Its Acceleration Range is +/-2g, +/-4g, +/-8g, +/-16g.
- Its Dimensions (excluding pins) are, 21.2mm (0.84″) length x 16.4mm(0.65″) width x 3.3mm (0.13″) height.
- Its weight is 2.1g.
- It has the smallest and thinnest QFN package for portable devices, 4x4x0.9 mm.
- Its operating current is 3.9 mA when its six motion sensing axes and DMP are in motion.
- It also has gyroscope feature like its Gyroscope operating current is 3.6 mA.
- Its gyroscopic stand by current is 5µA. It also has low improved frequency noise performance.
- It works at Gyroscope range, ± 250 500 1000 2000 °/ s.

3.2.5 GSM SIM 900A

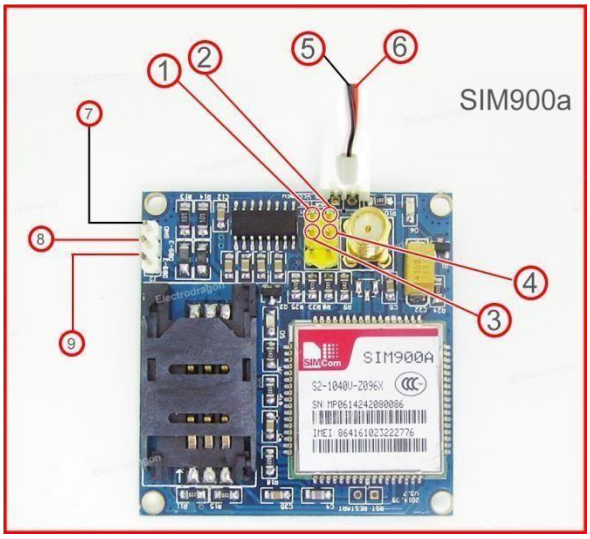


Fig 3.2.5.1 GSM Module Physical View

6 - VDC (+5V)
5 - GND
4 - RX (connect to Arduino TX - pin 1 for UNO)
3 - TX (connect to Arduino RX - pin 0 for UNO)

3.2.5.1 General features

- Quad-band 850/900/1800/1900MHz
- GPRS multi-slot class 12/10
- GPRS mobile station class B
- Compliant to GSM phase 2/2+
    - Class 4 (2 W @ 850/900MHz)
    - Class 1 (1 W @ 1800/1900MHz)
- Bluetooth: compliant with 3.0+EDR
- Dimensions: 24.0*24.0*3.0mm
- Weight: 3.14g
- Control via AT commands (3GPP TS 27.007,27.005 and SIMCOM enhanced AT Commands)
- Supply voltage range 3.4 ~ 4.4V
- Low power consumption
- Operation temperature:-40℃ ~85℃

3.2.6 GPS

- GPS or Global Positioning System is a satellite navigation system that furnishes location and time information in all climate conditions to the user.
- GPS is used for navigation in planes, ships, cars and trucks also. The system gives critical abilities to military and civilian users around theglobe.
- GPS provides continuous real time, 3-dimensional positioning, navigation and timing worldwide.

  VCC: Power supply 3.3-6V

  GND: Ground

  TX: Transmit data serially which gives information about location, time etc.

  RX: Receive data serially. It is required when we want to configure GPS module.

Fig 3.2.6.1GPS Module Physical View

3.2.6.1 Features:

- 5Hz position update rate
- Operating temperature range: -40 TO 85°CUART TTL socket
- EEPROM to save configuration settings
- Rechargeable battery for Backup
- The cold start time of 38 s and Hot start time of 1 s
- Supply voltage: 3.3 V
- Configurable from 4800 Baud to 115200 Baud rates. (default 9600)
- Super sense ® Indoor GPS: -162 dBm tracking sensitivity
- Support SBAS (WAAS, EGNOS, MSAS, GAGAN)
- Separated 18X18mm GPS antenna

3.3 List of SOFTWARE COMPONENTS

Arduino IDE
Xampp Server
C#
Php
Html

3.3.1Arduino IDE

The **Arduino Integrated Development Environment (IDE)** is a cross-platform application (for Windows, macOS, 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 source code for the IDE is released under the GNU General Public License, version 2.

The Arduino IDE supports the languages C and C++ using special rules of code structuring.

The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution.

The Arduino IDE employs the program *argued* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

In October 2019 the Arduino organization began providing early access to a new Arduino Pro IDE with debugging and other advanced features.

Fig 3.3.1.1 Arduino IDE

3.3.2 Xampp Server

XAMPP (/ˈzæmp/ or /ˈɛks.æmp/[2]) is a free and open-source cross-platform web server solution stack package developed by Apache Friends,[2] consisting mainly of the Apache HTTP Server, Maria DB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

XAMPP's ease of deployment means a WAMP or LAMP stack can be installed quickly and simply on an operating system by a developer, with the advantage that common add-in applications such as WordPress and Joomla! can also be installed with similar ease using Bitnami.

The term XAMPP is an apparent acronym. However, there is no official acronym expansion specified on the Apache Friends website. Their homepage header reads "XAMPP Apache + MariaDB + PHP + Perl", indicating that this abbreviation is a recursive acronym.

The term can be unofficially broken down as follows:

Letter	Meaning
X	as an ideographic letter referring to cross-platform
A	Apache or its expanded form, Apache HTTP Server
M	MariaDB (formerly: MySQL[])
P	PHP
P	PERL

Table 3.3.2Xampp Server Abbreviation table

3.3.3 C#

C# (pronounced see sharp, like the musical note C♯, but written with the number sign) is a general-purpose, multi-paradigm programming language encompassing strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines.

It was developed around 2000 by Microsoft as part of its .NET initiative, and later approved as an international standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270:2018). Mono is the name of the free and open-source project to develop a compiler

and runtime for the language. C# is one of the programming languages designed for the Common Language Infrastructure (CLI).

C# was designed by Anders Hejlsberg, and its development team is currently led by Mads Torgersen. The most recent version is 8.0, which was released in 2019 alongside Visual Studio 2019 version 16.3.

The Ecma standard lists these design goals for C#:

- The language is intended to be a simple, modern, general-purpose, object-oriented programming language.
- The language, and implementations thereof, should provide support for software engineering principles such as strong type checking, array bounds checking, detection of attempts to use uninitialized variables, and automatic garbage collection. Software robustness, durability, and programmer productivity are important.
- The language is intended for use in developing software components suitable for deployment in distributed environments.
- Portability is very important for source code and programmers, especially those already familiar with C and C++.
- Support for internationalization is very important.
- C# is intended to be suitable for writing applications for both hosted and embedded systems, ranging from the very large that use sophisticated operating systems, down to the very small having dedicated functions.
- Although C# applications are intended to be economical with regard to memory and processing power requirements, the language was not intended to compete directly on performance and size with C or assembly language.

3.3.4 Php

PHP is a popular general-purpose scripting language that is especially suited to web development. It was originally created by Rasmus Lerdorf in 1994; the PHP reference implementation is now produced by The PHP Group. PHP originally stood for *Personal Home Page*, but it now stands for the recursive initialism *PHP: Hypertext Preprocessor*. PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon or as a Common Gateway Interface (CGI) executable. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the *de facto* standard which other implementations aimed to follow. Since 2014, work has gone on to create a formal PHP specification. As of February 2020, over half of sites on the web using PHP are still on discontinued/"EOLed" version 5.6 or older;[14] and over 55% of all websites in the world run versions prior to 7.2, that are neither officially supported by The PHP Development Team,

An example of the early PHP syntax:

```
<!--include/text/header.html-->

<!--getenvHTTP_USER_AGENT-->
<!--ifsubstr$exec_resultMozilla-->
Hey,youareusingNetscape!<p>
<!--endif-->

<!--sqldatabaseselect*fromtablewhereuser='$username'-->
<!--ifless$numentries1-->
Sorry,thatrecorddoesnotexist<p>
<!--endifexit-->
Welcome<!--$user-->!<p>
Youhave<!--$index:0-->creditsleftinyouraccount.<p>

<!--include/text/footer.html-->
```

3.3.5 Html

Hypertext Markup Language (**HTML**) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <img/> and <input/> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

3.4 Database

According to its developers, "**ThingSpeak** is an open-source, Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP and MQTT protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates".

We have ThingSpeak cloud environment to store the data. ThingSpeak provides users with free time-series data storage in channels. Each channel can include up to eight data fields. This tutorial provides an introduction to some of the applications of ThingSpeak, a conceptual overview of how ThingSpeak stores time-series data, and how MATLAB analysis is incorporated in ThingSpeak. We have used six fields to store the data. The data is displayed in form of charts. From that it is sent to website and dashboard app. In those every value is displayed, when the new value is updated in the ThingSpeak.

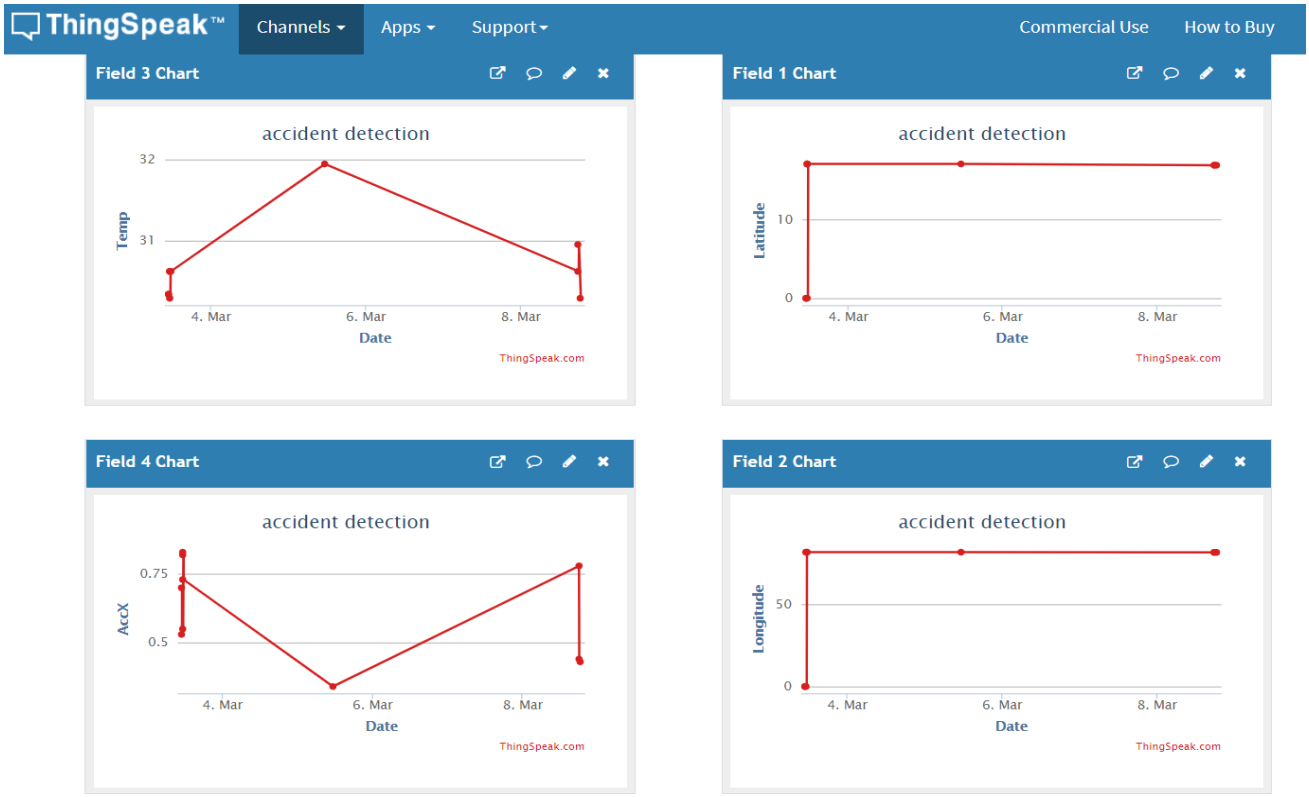


Fig 3.4.1Thingspeak

CHAPTER-4

Methodology

4. METHODOLOGY

4.1 System Architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages.

The system architecture and mechanism of the project can be observed from below figures

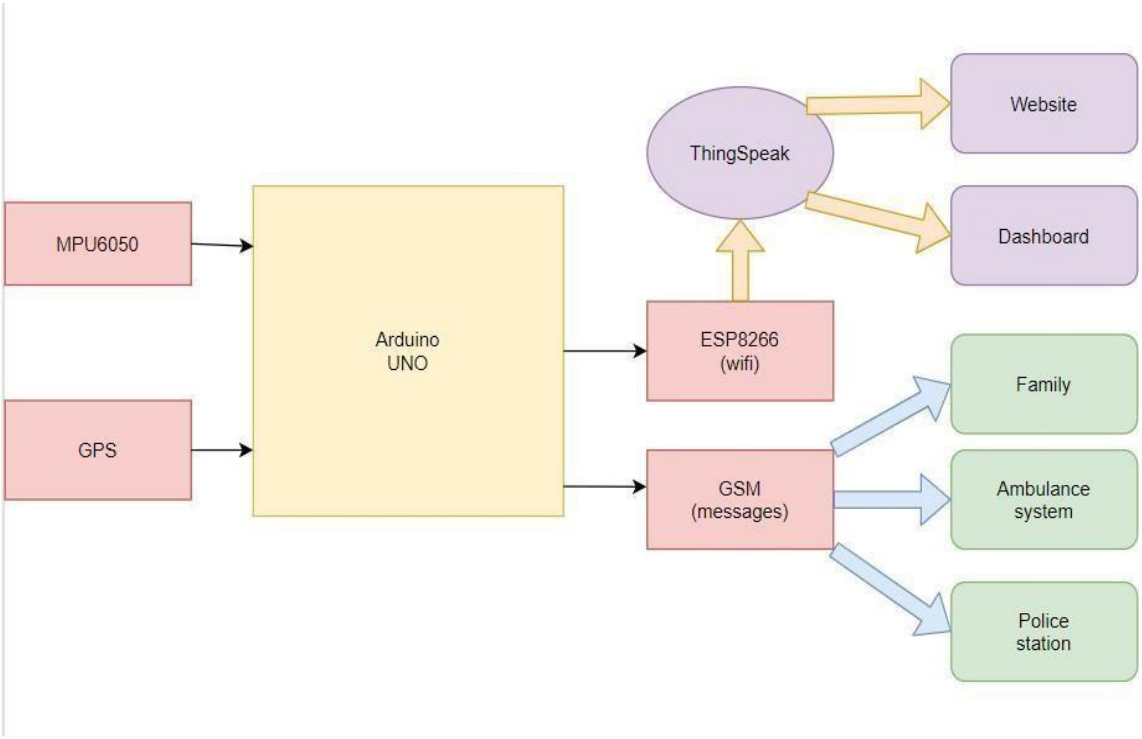


Fig 4.1.1 Proposed System Architecture

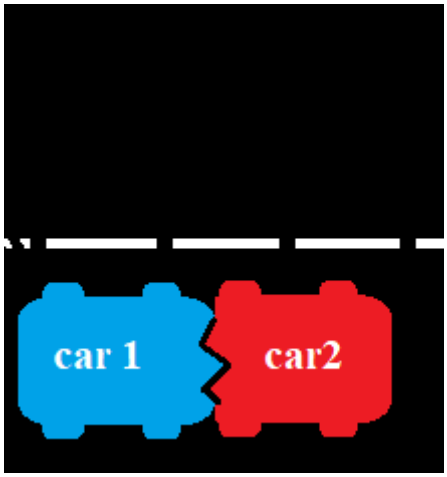
4.2 Working Principle:

Working of the proposed system is very simply we created a webpage for the user registration and to view the location on google map, to see where the accident was happened. MPU6050 Sensor has been used to detect the accident, as it was gyroscopic sensor it can measure the vibration in x,y,z axis, if the car hits anything the vibration in that

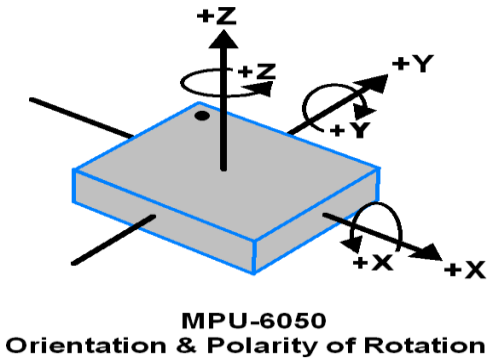
x/y/z/ axis will cause the micro-controller to activate the system which consists of GPS and GSM module, if the accident was happened then the Arduino will fetch the longitude and the latitude values then these values will be added by the Arduino in the Google maps URL in their respective fields, the same will be send to the ThingSpeak, and as SMS to the registered mobile numbers. If you click on the link you will be navigated to the Google maps. With that you can get an SMS at the incident when accident was happened which saves lot of time and many human lives.

From the below diagrams you can understand better:

If collision happens as show below

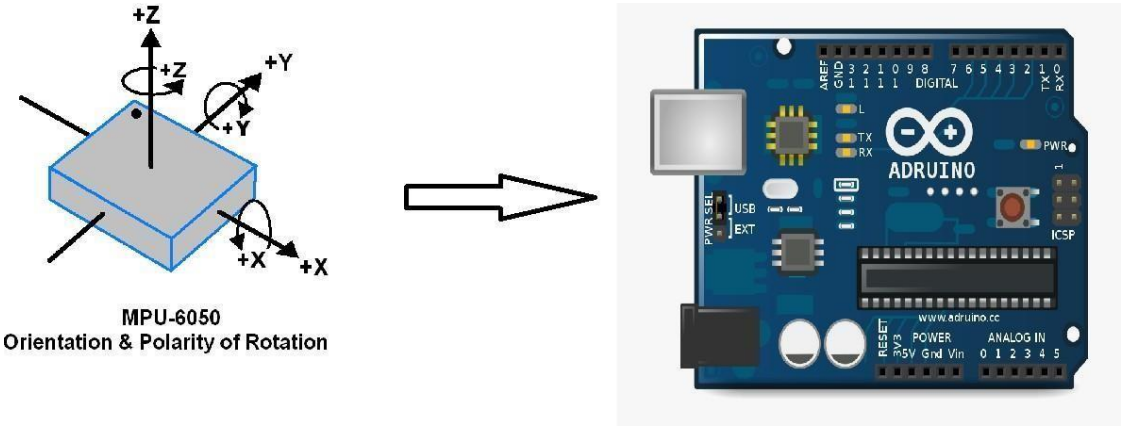


The Gyroscope MPU6050 gets disturbed on the axis as shown below



MPU-6050
Orientation & Polarity of Rotation

Arduino will continuously monitors the data coming from the MPU6050



As per the code it is going to detect weather there is accident happened or not , now if the accident happens then it will send the alert signal to cloud and generates sms. This data includes longitude and latitude as well.

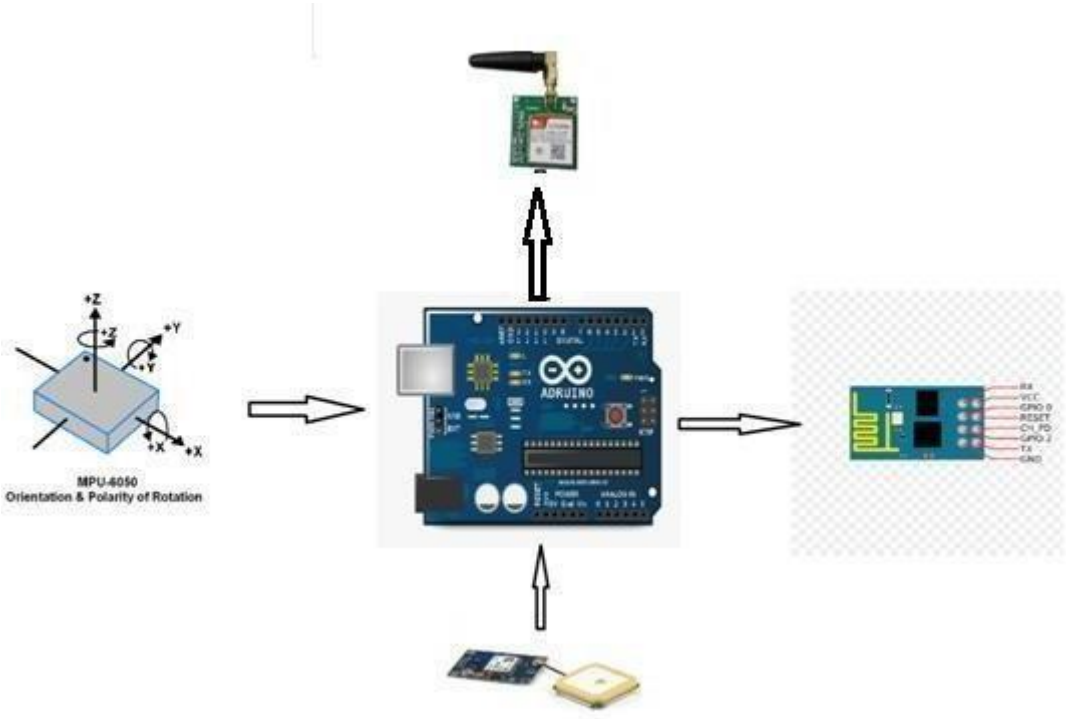


Fig 4.2.1 Working of Project

4.3 Circuit Design

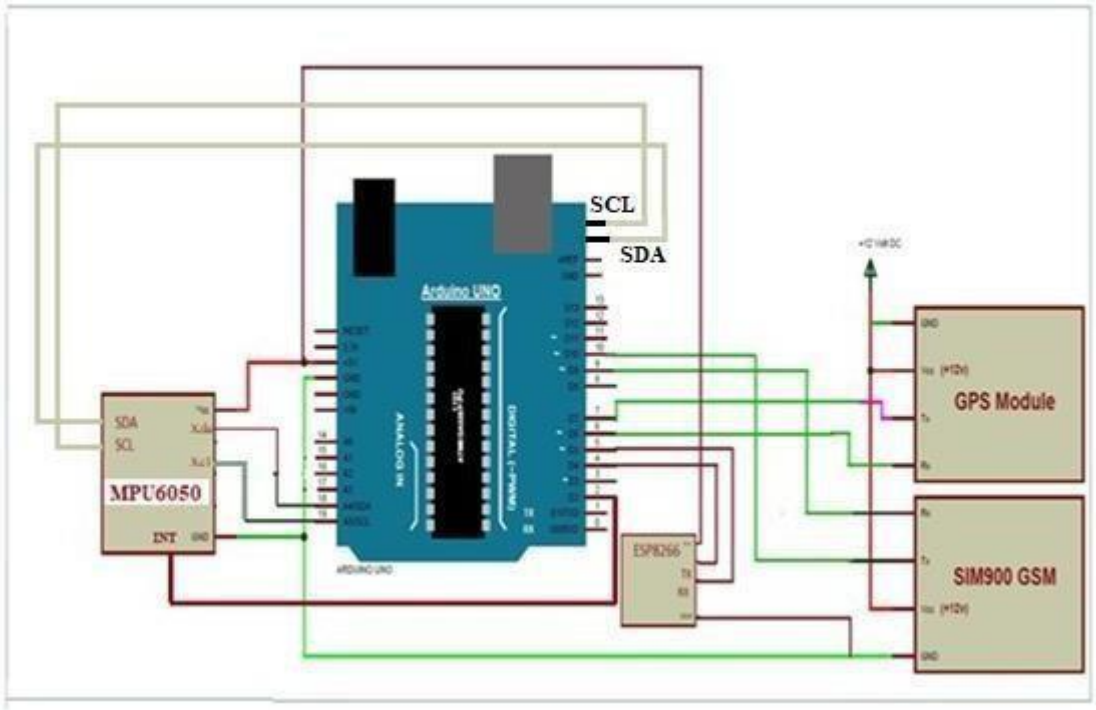


Fig 4.3.1 Circuit Diagram of the proposed system

CHAPTER-5

System Design

5. SYSTEM DESIGN

5.1 Introduction

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Beginning, once system requirement has been specified and analyzed, system design is the first of the three technical activities-designs, code and test that is required to build and verify software. The importance can be stated with a single word "quality". Design is the phase where quality is fostered in software development.

During design, progressive refinement of data structure, program structure and procedural details are developed, reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view design is comprised of four activities - Architectural design, Data structure design, Interface design and Procedural design.

5.2 Use Cases

5.2.1 Definition

A use case diagram is a set of use cases, actors and relationships between actors and use cases. A use case is represented by oval shape.

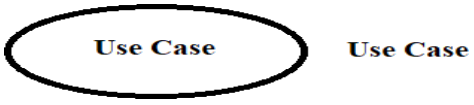
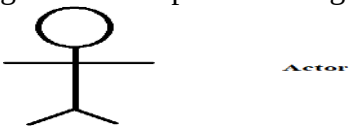


Fig 5.2.1.1 Simple use case diagram

An actor represents a coherent set of roles that users of use cases play when interacting with these use cases. Typically, an actor represents a role that human, a hardware device or even other system plays with a system. Actors are rendered as stick figures.

Fig 5.2.1.2 Simple actor diagram



- Use case is used to capture high level functionalities of a system.
- Actor is used in a use case diagram to describe the internal or external entities.
- Use case diagrams specify the events of a system and their flows.
- But, use case diagram never describes how they are implemented.

- Use case diagram can be imagined as a black box where only the input, output and the function of the black box is known.
- Use case diagrams can be used for
- Requirement analysis and high-level design.
- Model the context of a system.
- Reverse engineering.
- Forward engineering.

5.2.2 Actors

- Admin
- User

5.2.3 List of Use Cases

- Register
- Login
- Connect to sensors in vehicle
- Connect user to sensors
- Get sensors information
- Send notification alerts
- Logout

5.2.4 Use case Diagram for User and Admin:

A use case describes how a user uses a system to accomplish a particular goal. A use case diagram consists of the system, the related use cases and actors and relates these to each other to visualize.

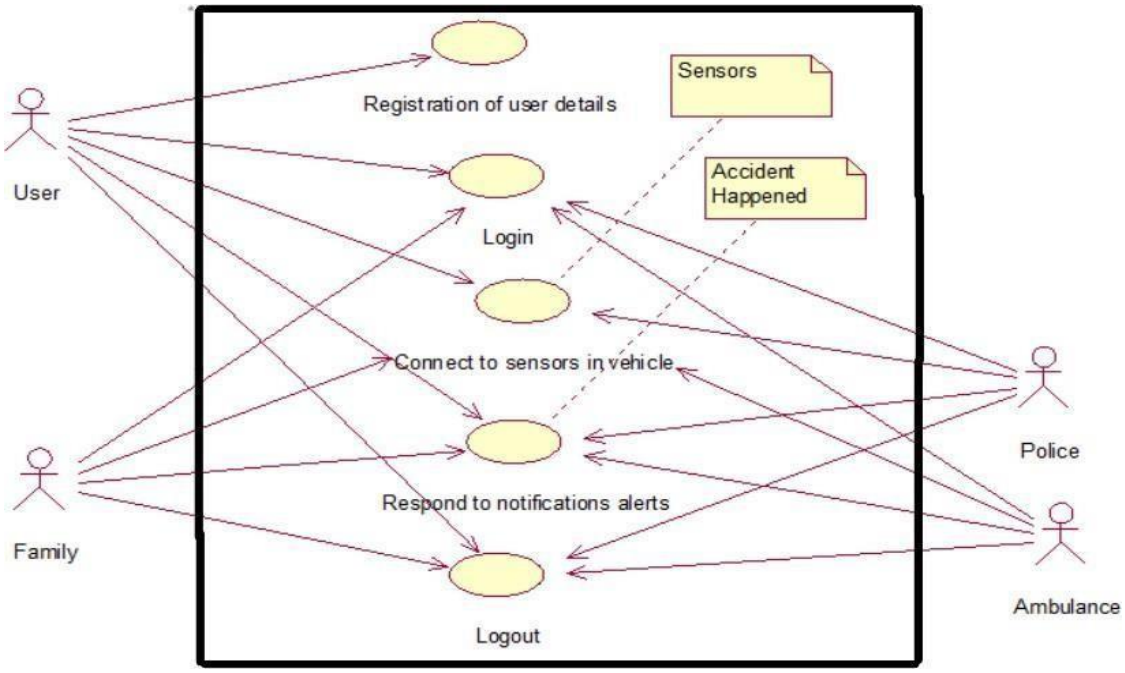


Fig 5.2.4.1 Use case diagram for user

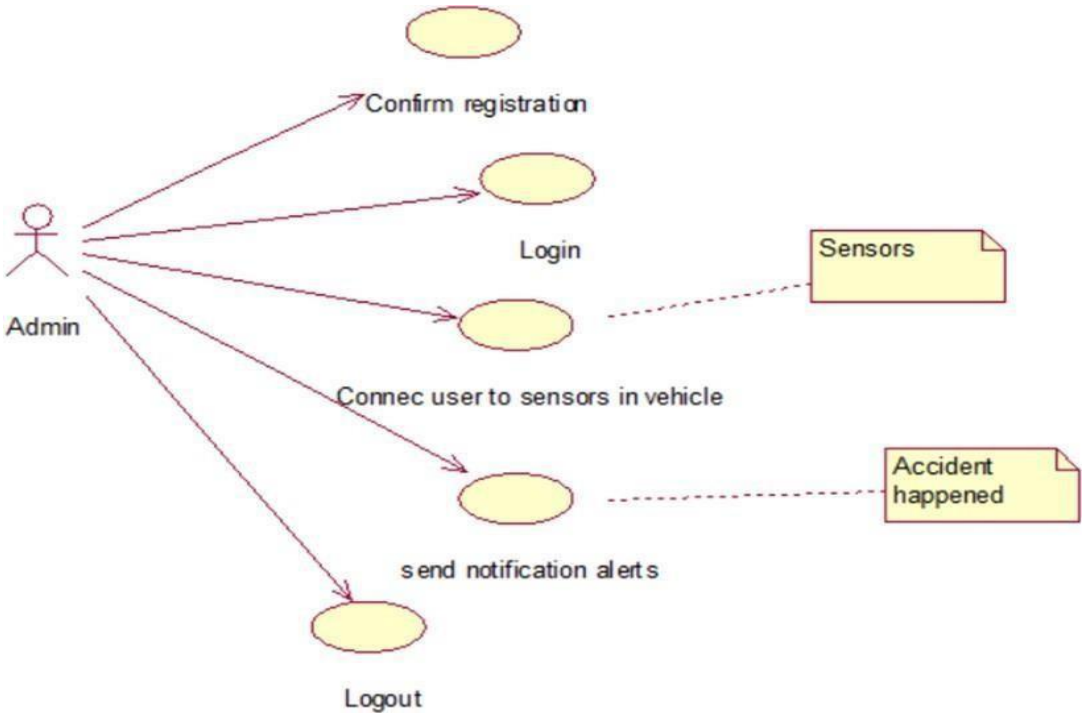


Fig 5.2.4.2 Use case diagram for Admin

5.3 Component Diagram

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus, from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment.

A single component diagram cannot represent the entire system but a collection of diagrams is used to represent the whole.

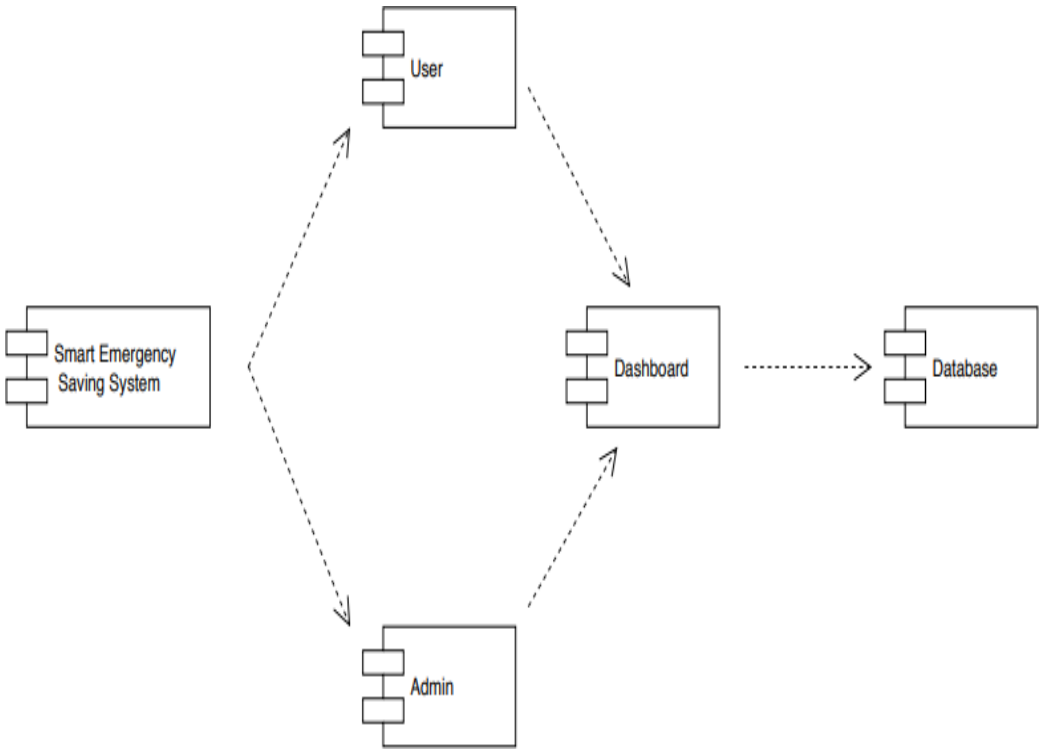


Fig 5.3.1 Single Component diagram

5.3.1 Object Descriptions

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams. Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagram and object diagrams. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment. Object diagrams are used to render a set of objects and their relationships as an instance.

5.3.1.1 Objects

As the name, object-oriented implies, objects are key to understanding object-oriented technology.

These real-world objects share two characteristics: they all have state and they all have behavior. A software object maintains its state in variables and implements its behavior with methods.

An object is a software bundle of variables and related methods. Class object is the root of class hierarchy. Every class has object as a super class. All objects, including arrays, implement the method of this class.

5.4 Class Diagrams

A class diagram gives an overview of a system by showing its classes and the relationships among them. UML class is represented as RECTANGLE divide into class name, attributes and operations. Class diagram has three kinds of relationships.

- **Association:** A relationship between instances of the two classes.
- **Aggregation:** An association in which one class belongs to a collection. An aggregation has a diamond end pointing to the part containing the whole.
- **Generalization:** An inheritance link indicating one class is a super class of the other.
- **Dependency:** Properties of one class depend on properties of another class.

It shows set of classes, interfaces and collaborations & their relationships. This addresses the static design view of a system that include active classes address the static process view of a system.

5.4.1 Class

A Class defines the attributes and the methods of a set of objects. All objects of this class share the same behavior, and have the same set of attributes. The term—Type1 is sometimes used instead of class, but it is important to mention that these two are not the same, and Type is a general term.

In UML, classes are represented but rectangles, with the name of the class, and can also show the attributes and operations of the class in other –compartments inside the rectangle.

5.4.2 Attributes

In UML, Attributes are shown with at least their name and can also show their type, initial value and other properties. Attributed can also displayed with their visibility.
+ Stands for public operation
# Stands for protected operation
- Stands for private operation

5.4.3 Operations

Operations are displayed with at least their name and can also show their parameters and return types. Operations can, just as Attributes can also displayed with their visibility.
+ Stands for public operation
# Stands for protected operation
- Stands for private operation

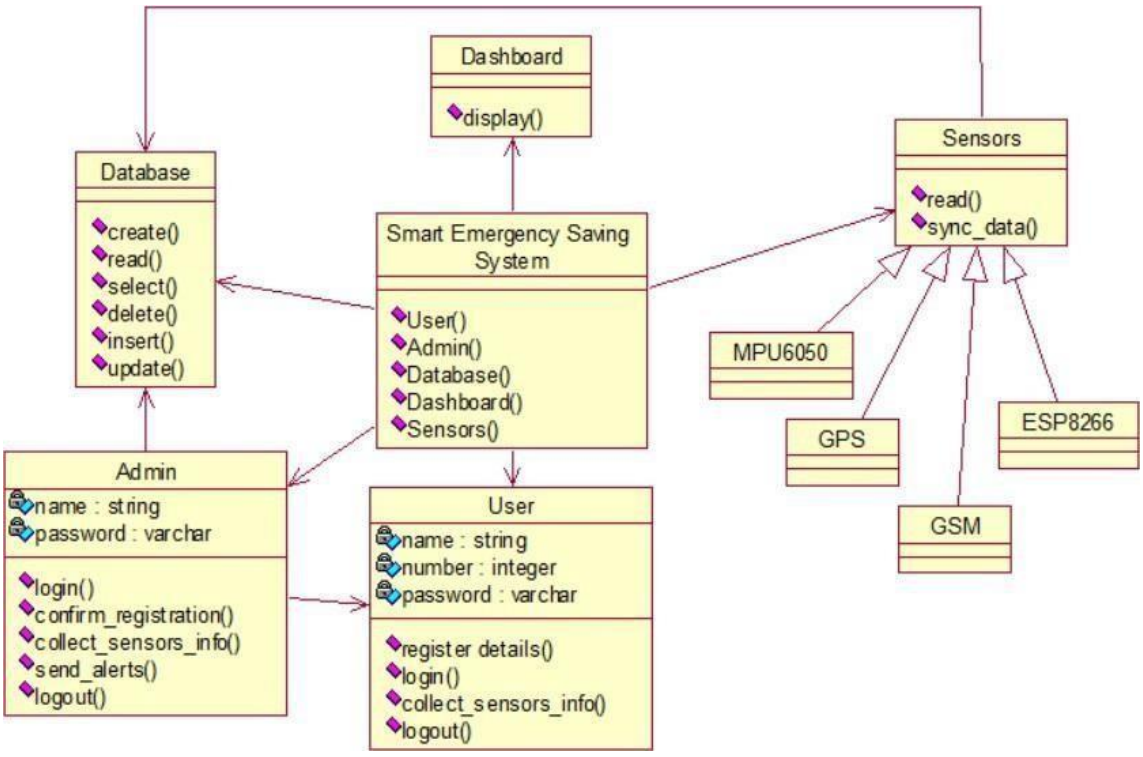


Fig 5.4.1.1 Class diagram

5.5 Object Collaboration

A collaboration diagram also called a communication diagram or interaction diagram is an illustration of the relationships and interactions among software objects in the unified modeling language.

The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

These diagrams are used to portray the dynamic behavior of a particular use case and define the role of each object. These diagrams are created by first identifying the structural elements required to carry out the functionality of an interaction. A model is then built using the relationships between those elements.

The four major components of a collaboration diagram are:

1. Objects- Objects are shown as rectangles with naming labels inside. The naming label follows the convention of object name: class name. If an object has a property or state that specifically influences the collaboration, this should be noted.
2. Actors- Actors are instances that invoke the interaction in the diagram. Each actor has a name and a role, with one actor initiating the entire use case.
3. Links- Links connect objects with actors and are depicted using a solid line between two elements. Each link is an instance where messages can be sent.
4. Messages- Messages between objects are shown as a labeled arrow placed near a link. These messages are communications between objects that convey information about the activity and can include the sequence number.

A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. Objects are shown as rectangles with naming labels inside.

The relationship between the objects are shown as line connecting the rectangles. The message between objects are shown as arrows connecting the relevant rectangles along with labels that define the message sequencing.

5.5.1 Object Collaboration Diagrams

Collaboration diagrams are best suited to the portrayal of simple interactions among relatively small number of objects. As the number of objects and messages grow, collaboration diagram can become difficult to read.

Several vendors offer software for creating and editing collaboration diagrams.

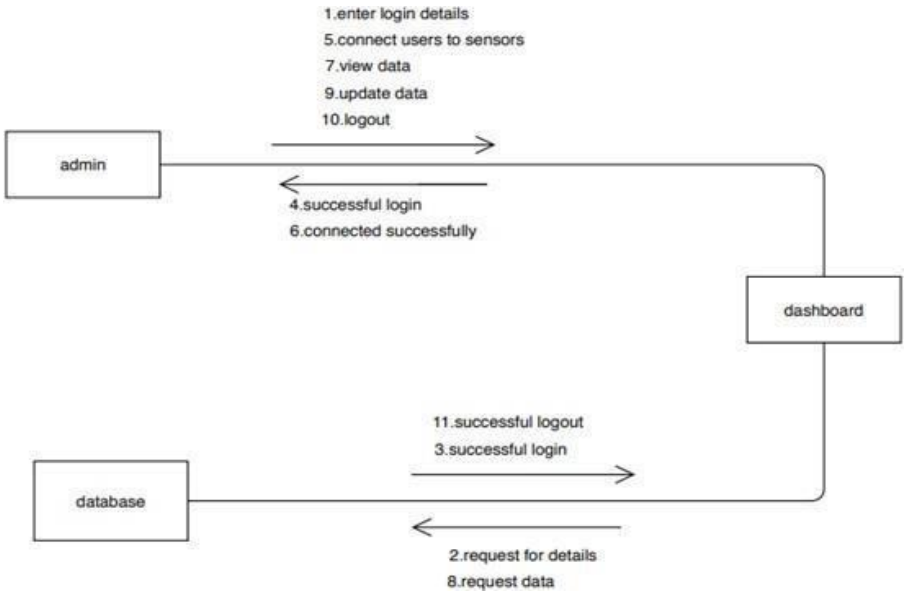


Fig 5.5.1.1 Object Collaboration Diagram for Admin

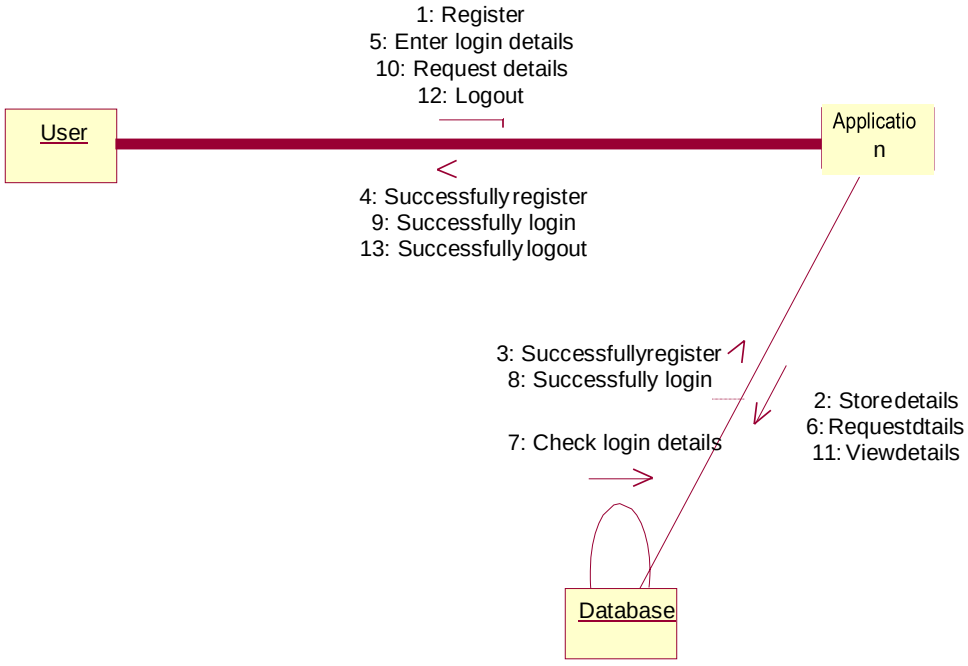


Fig 5.5.1.2 Object Collaboration Diagram for User

5.6 Database Design

Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems. Properly designed database is easy to maintain, improves data consistency and are cost effective in terms of disk storage space. The database designer decides how the data elements correlate and what data must be stored.

The main objectives of database designing are to produce logical and physical designs models of the proposed database system. The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically. The physical data design model involves translating the logical design of the database onto physical media using hardware resources and software systems such as database management systems (DBMS).

Database designs also includes ER (Entity diagrams) is a diagram that helps to design databases in an effective way. Attributes in ER diagrams are usually modeled as an oval with entity or relationship that contains the attributes.

5.7 Entity-Relationship Diagram

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties. By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases.ER diagrams are used to sketch out the design of a database.

5.7.1 Benefits of Entity Relationship Diagram

- **Provides a visual representation of the design:** It is normally a key to have on ERD if we are looking to come up with a valuable database design. This is as the behavior allows the designer in focusing on the method of the database will primarily work with all the data flows and relationships. It is general to the ERD being used collectively with data flow diagrams so as to make a better visual representation.
- **Makes effective communication:** An ERD actually conveys the key entities in some database and their relationship with each other. ERD usually provides symbols for representing three different kinds of information. Diamonds are used for representing the relationships, ovals are usually used for representing attributes and boxes represent the entities. This allows a designer to effectively communicate what exactly the database may be like.

- **Simple to understand:** ERD is simple to understand and simple to create. In impact, this design can be used to be offered to the Representatives for both authorization and evidence. The representatives can also make their contributions to the design, giving the ways of rectifying and developing the design.
- **Highly flexible:** The ERD model is so flexible to use as additional relationships can be derived instantly from the currently available ones. This can be done using other relational tables and mathematical formulas.

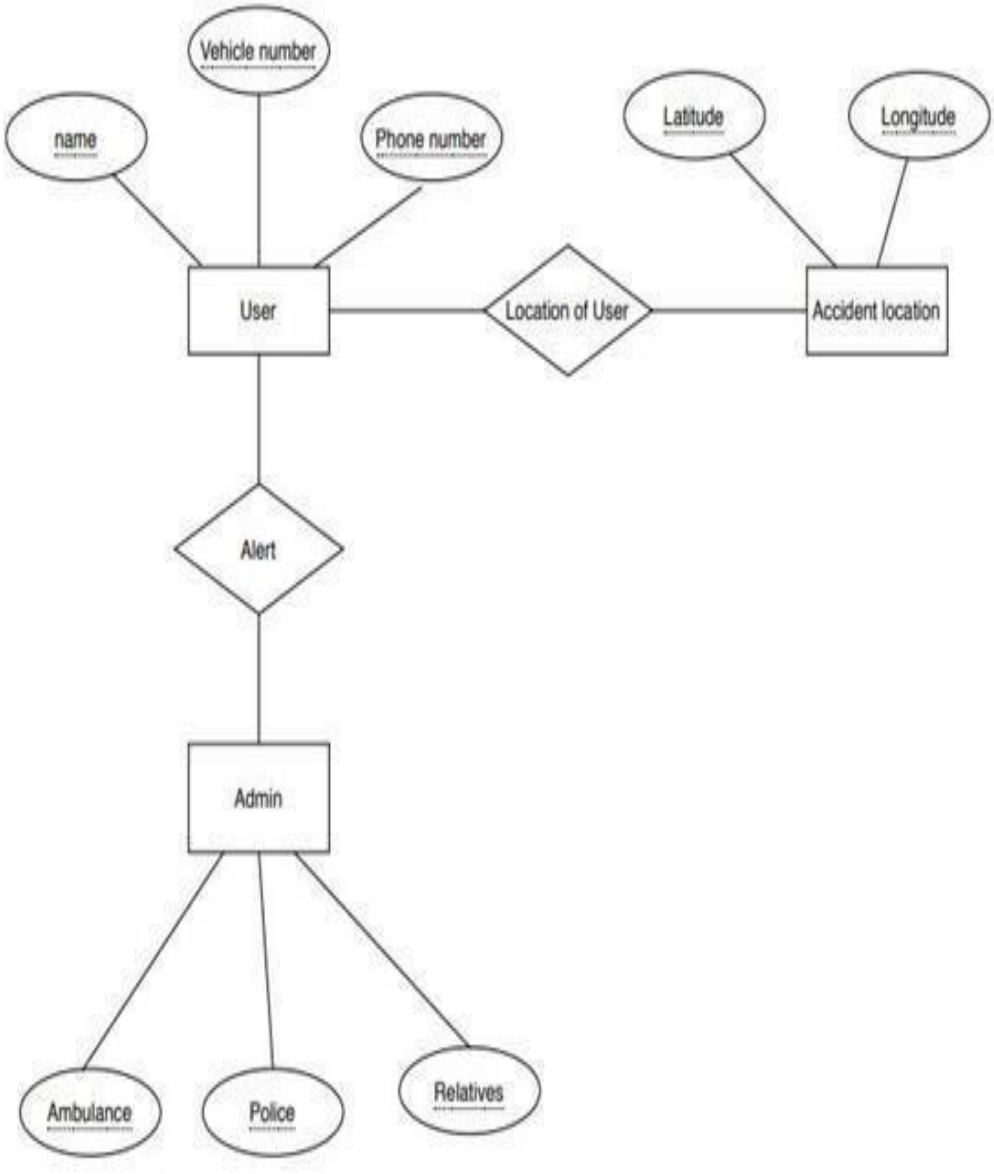


Fig 5.7.1 Entity-Relationship Diagram

5.8 Dynamic Model

5.8.1 Sequence Diagrams

A sequence diagram is the most commonly used interaction diagram. An interaction diagram is used to show the interactive behavior of a system. Since visualizing the interactions in a system can be a cumbersome task, we use different types of interaction diagrams to capture various features and aspects of interaction in a system.

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

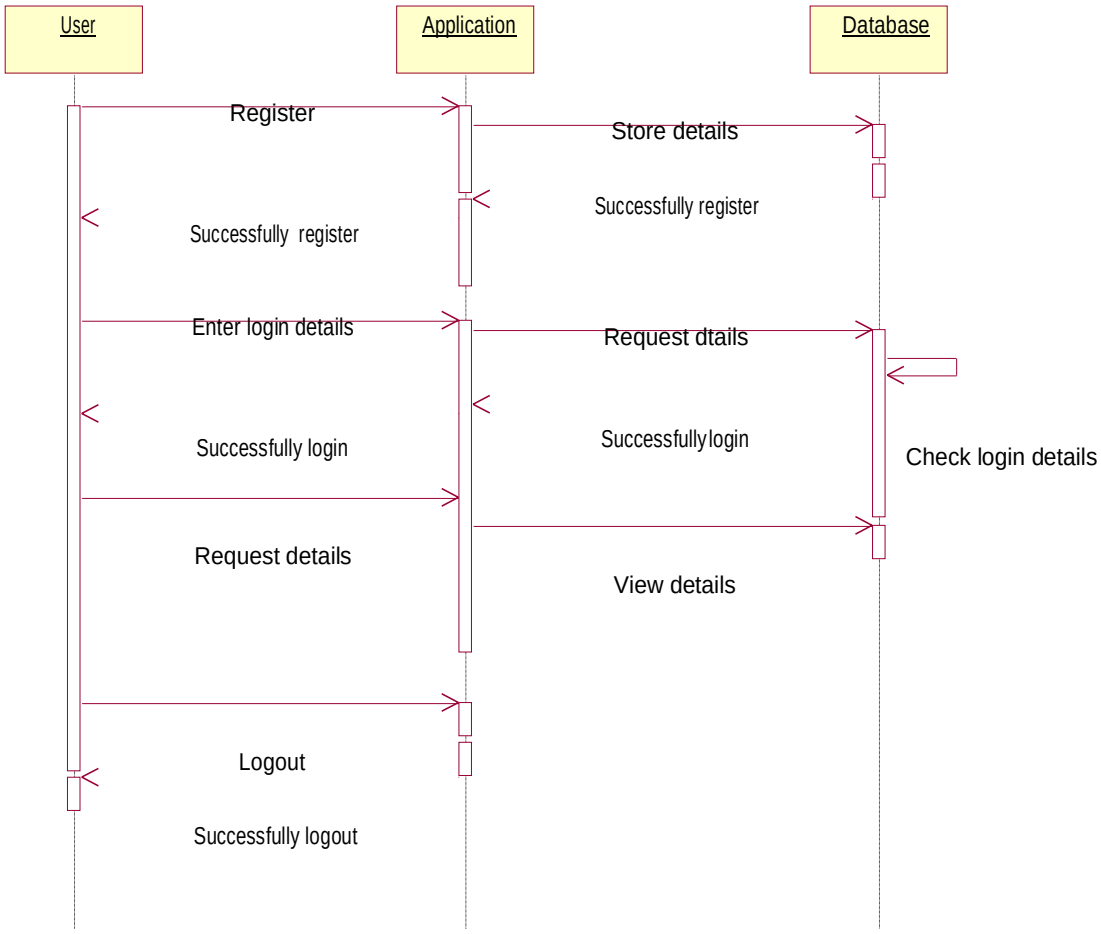


Fig 5.8.1.1 Sequence Diagram for User

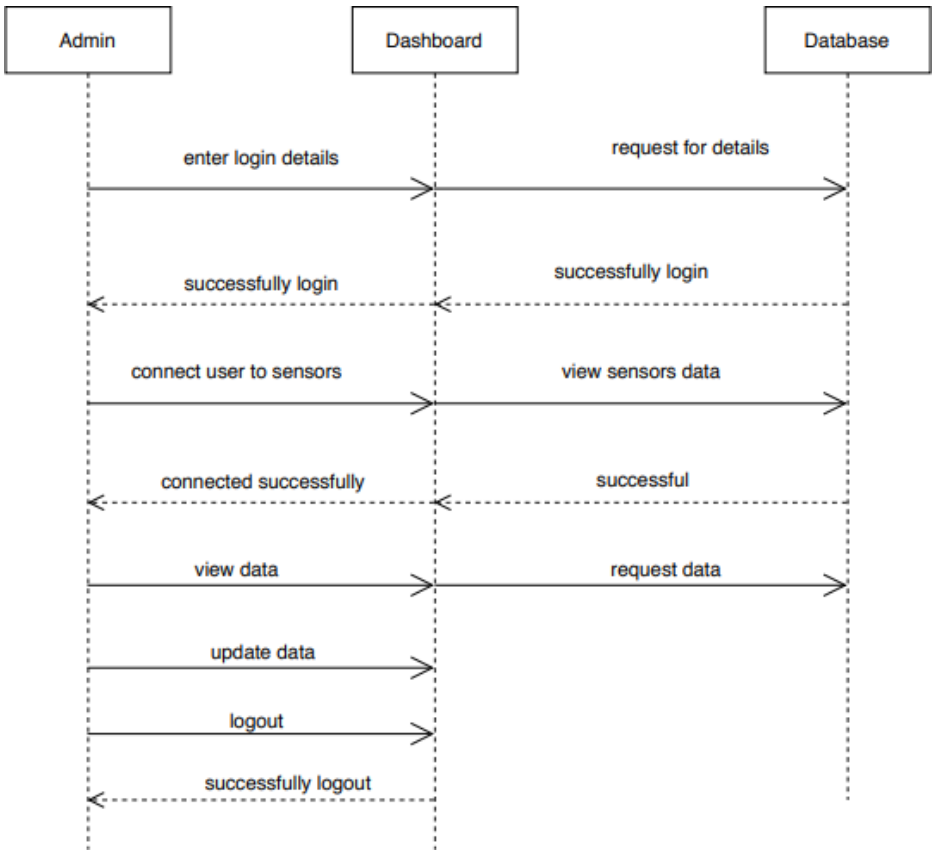


Fig 5.8.1.2 Sequence Diagram for Admin

5.8.2 State Chart Diagrams

State machine diagrams are also called as state chart diagrams. State machine diagrams are used to capture the behavior of a software system. UML State machine diagrams can be used to model the behavior of a class, a subsystem, a package, or even an entire system. It is also called a State chart or State Transition diagram. State chart diagrams provide us an efficient way to model the interactions or communication that occurs within the external entities and a system. These diagrams are used to model the event-based system. A state of an object is controlled with the help of an event. State chart diagrams are used to describe various states of an entity within the applicationsystem.

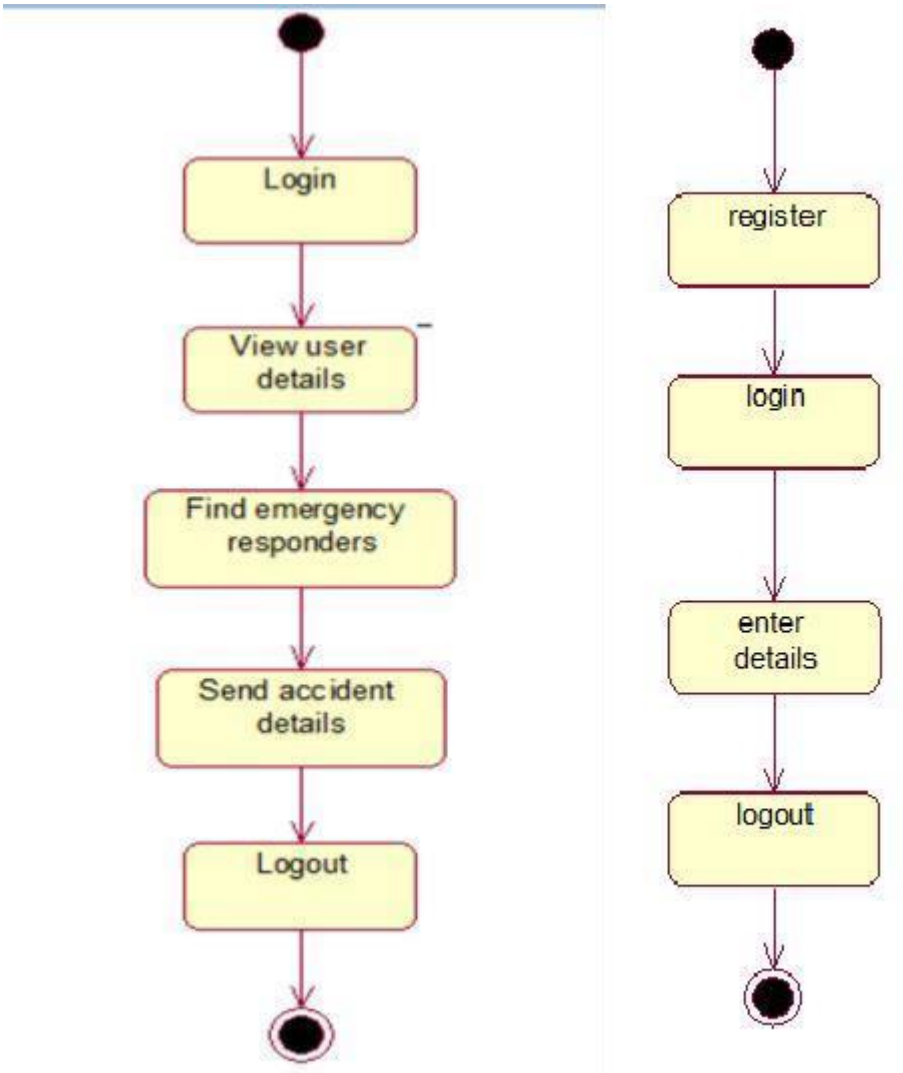


Fig 5.8.2.1 State Chart Diagram for Admin and User

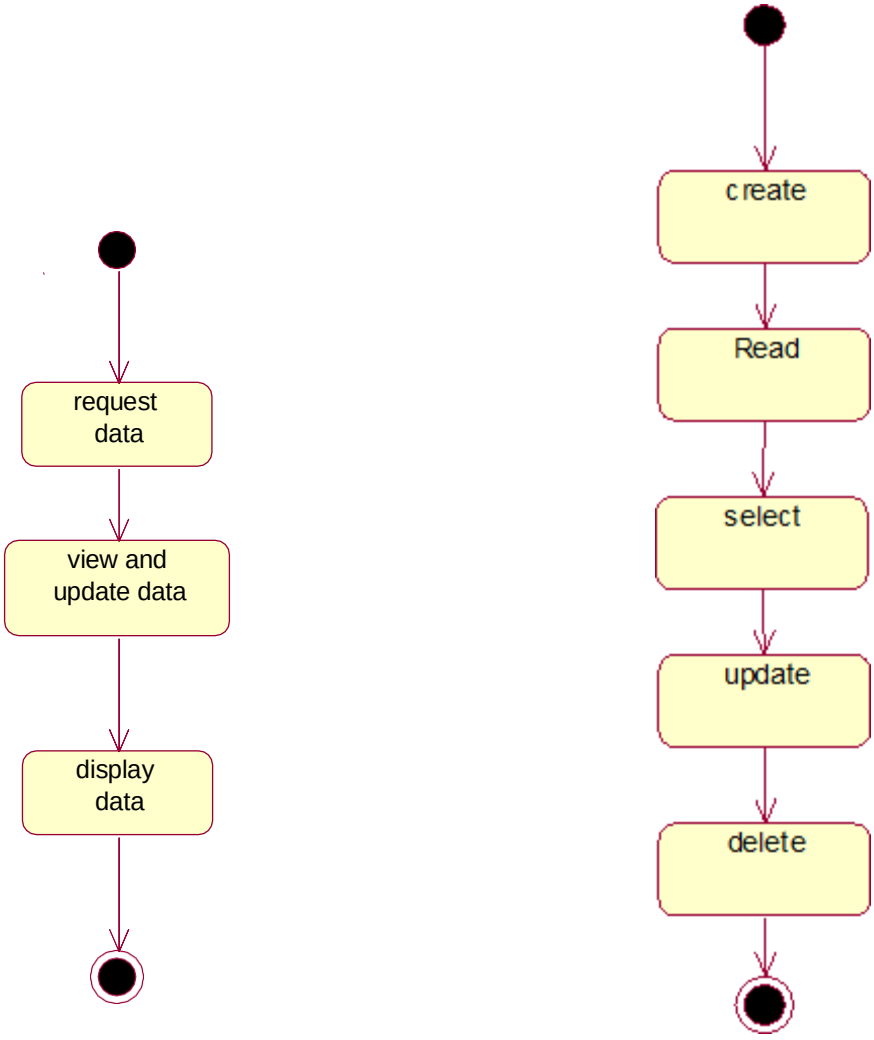


Fig 5.8.2.2 State Chart Diagram for Database and Dashboard

5.8.3 Activity Diagrams

Activity diagram is defined as a UML diagram that focuses on the execution and flow of the behavior of a system instead of implementation. It is also called object- oriented flowchart. Activity diagrams consist of activities that are made up of actions which apply to behavioral modeling technology.

Activity diagram allows you to create an event as an activity which contains a collection of nodes joined by edges. An activity can be attached to any modeling element to model its behavior. Activity diagrams are used to model processes and workflows. The essence of a useful activity diagram is focused on communicating a specific aspect of a system's dynamic behavior. Activity diagrams capture the dynamic elements of a system.

Activity diagram is similar to a flowchart that visualizes flow from one activity to another activity. Activity diagram is identical to the flowchart, but it is not a flowchart. The flow of activity can be controlled using various control elements in the UML diagram. In simple words, an activity diagram is used to activity diagrams that describe the flow of execution between multiple activities.

Activity Diagram Notations

Activity diagrams symbol can be generated by using the following notations:

- Initial states: The starting stage before an activity takes place is depicted asthe initial state
- Final states: The state which the system reaches when a specific process ends is known as a Final State
- State or an activity box:
- Decision box: It is a diamond shape box which represents a decision withalternate paths. It represents the flow of control.

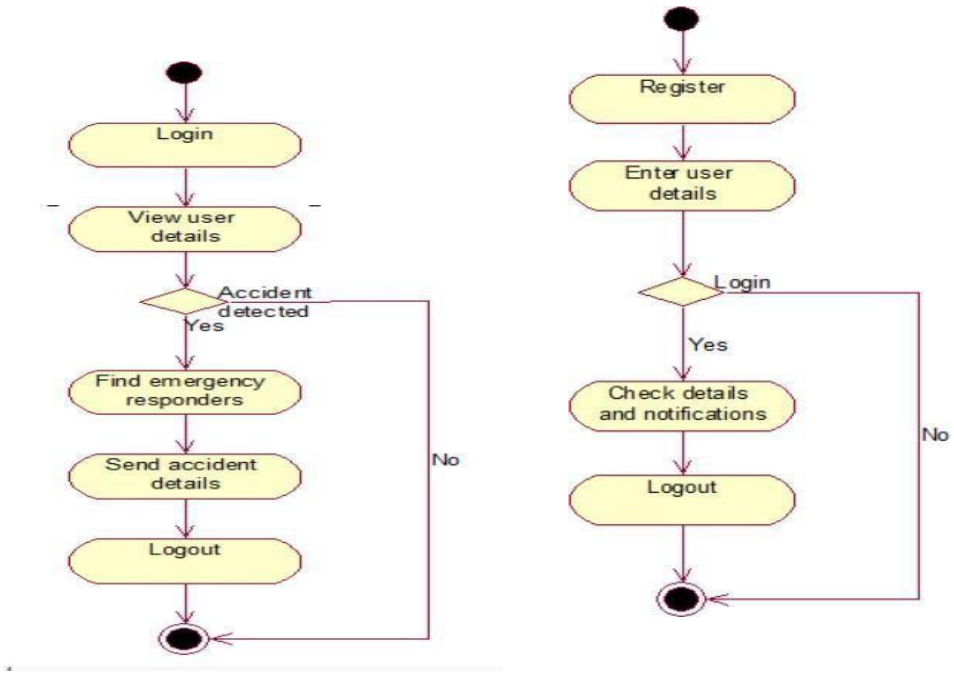


Fig.5.8.3.1 Activity Diagram Admin and User

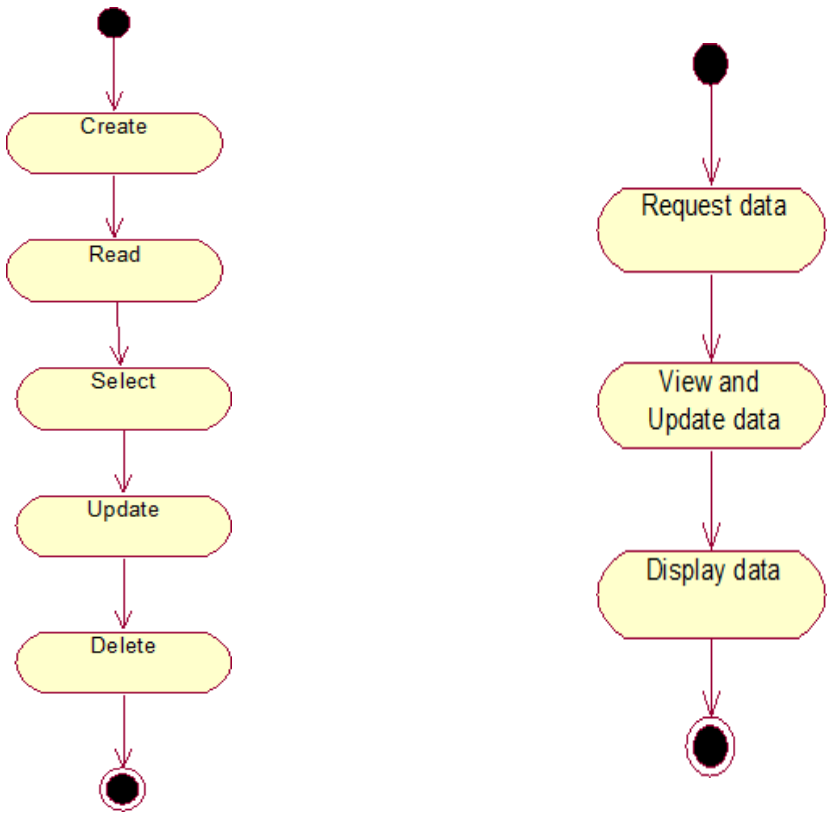


Fig 5.8.3.2 Activity Diagram for Database and Dashboard

5.8.4 Deployment Diagrams

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them. Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware. Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.

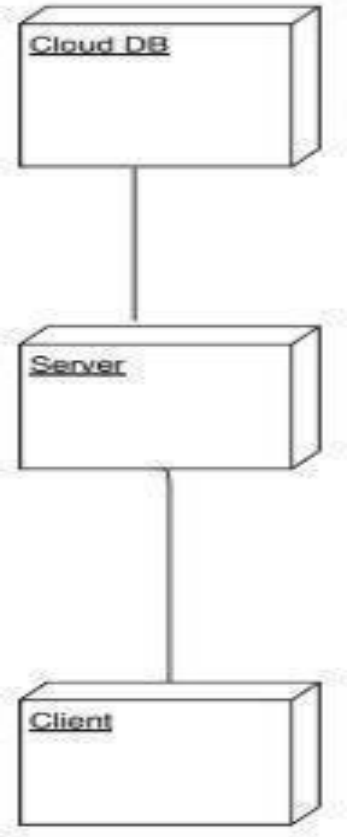


Fig.5.8.4.1 Deployment Diagram

Chapter-6

Implementation

6. IMPLEMENTATION

6.1 Arduino Coding

6.1.1 To Send Messages

```
#include <SoftwareSerial.h>
#include<MPU6050_tockn.h>
#include <Wire.h>
#include <TinyGPS++.h>
MPU6050 mpu6050(Wire);
SoftwareSerial ss(6,7);//gps
long timer = 0;
int i=0;
TinyGPSPlusgps;
String lat,lng;
SoftwareSerialmySerial(9, 10);//sim900a
String msg="";

void sendmsg();
void sendmsg2();
void sendmsg3();
void setup() {
mySerial.begin(9600); // Setting the baud rate of GSM Module
Serial.begin(9600);
ss.begin(9600);// Setting the baud rate of Serial Monitor (Arduino)
Wire.begin();
  mpu6050.begin();
  mpu6050.calcGyroOffsets(true);
  delay(100);
```

```
}

void loop() {
  mpu6050.update();
  while(ss.available())//While there are characters to come from the GPS
      {
gps.encode(ss.read());//This feeds the serial NMEA data into the library one char at a time
      }
  if(millis() - timer > 100){
Serial.print("temp : ");Serial.println(mpu6050.getTemp());
Serial.print("acc : ");Serial.println(mpu6050.getAccX());

    if(mpu6050.getAccX()>0.3){

      if(mpu6050.getAccX()<0.5){msg="Car just met with an obstacle.No harm. ";}
      else if(mpu6050.getAccX()<0.7){msg="Do not panic.Injuries are very mild.";}
      else {msg="Accident is very severe.Hurry up.";}
      while(ss.available())//While there are characters to come from the GPS
      {
gps.encode(ss.read());//This feeds the serial NMEA data into the library one char at a time
      }
      // lat=gps.location.lat();
      //lon="82.0571";
lat="17.0814";
lng=gps.location.lng();

sendmsg();
```

```
    sendmsg2();
    //sendmsg3();
Serial.println("Accident happened");// The SMS text you want to send
Serial.println(msg);
Serial.print("temp : ");mySerial.println(mpu6050.getTemp());
Serial.print("acc : ");mySerial.println(mpu6050.getAccX());
Serial.println("Location:");// The SMS text you want to send
Serial.print("https://www.google.com/maps/?q=");
Serial.print(lat);
Serial.print(",");
Serial.print(lng);

   timer = millis();

 }
 }

}
void sendmsg(){
 while(ss.available())//While there are characters to come from the GPS
     {
gps.encode(ss.read());//This feeds the serial NMEA data into the library one char at a time
     }
mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
    delay(1000); // Delay of 1 second
mySerial.println("AT+CMGS=\"+917330844898\"\r"); // Replace x with mobile number
```

```
    delay(1000);
     delay(1000);
mySerial.println("Accident happened");// The SMS text you want to send
mySerial.println(msg);
mySerial.print("temp : ");mySerial.println(mpu6050.getTemp());
mySerial.print("acc : ");mySerial.println(mpu6050.getAccX());
mySerial.println("Location:");// The SMS text you want to send
mySerial.print("https://www.google.com/maps/?q=");
mySerial.print(lat);
mySerial.print(",");
mySerial.print(lng);
     delay(100);
mySerial.write(26);// ASCII code of CTRL+Z for saying the end of sms to the module
    delay(1000);

}
void sendmsg2(){
  while(ss.available())//While there are characters to come from the GPS
     {
gps.encode(ss.read());//This feeds the serial NMEA data into the library one char at a time
     }
mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
    delay(1000); // Delay of 1 second

mySerial.println("AT+CMGS=\"+919701989631\"\r"); // Replace x with mobile number
    delay(1000);
mySerial.println("Accident happened");// The SMS text you want to send
```

```
mySerial.println(msg);
mySerial.print("temp : ");mySerial.println(mpu6050.getTemp());
mySerial.print("acc: ");mySerial.println(mpu6050.getAccX());
mySerial.println("Location:");// The SMS text you want to send
mySerial.print("https://www.google.com/maps/?q=");
mySerial.print(lat);
mySerial.print(",");
mySerial.print(lng);
    delay(100);
mySerial.write(26);// ASCII code of CTRL+Z for saying the end of sms to the module
    delay(1000);

}
```

6.1.2 To Upload Data To Thingspeak:

```
#include <SoftwareSerial.h>
#include <MPU6050_tockn.h>
#include <Wire.h>
#include <TinyGPS++.h>
TinyGPSPlusgps;
MPU6050 mpu6050(Wire);

long timer = 0;

SoftwareSerialespSerial(3, 4); //Pin 2 and 3 act as RX and TX. Connect them to TX and RX of ESP8266
SoftwareSerial ss(6,7); //rx=6 and tx=7
```

```
#define DEBUG true
String mySSID = "My phone";       // WiFi SSID
String myPWD = "luckylove"; // WiFi Password
String myAPI = "63QP1EEXGD5YKV23"; // API Key
String myHOST = "api.thingspeak.com";
String myPORT = "80";
String myFIELD1 = "field1";
String myFIELD2 = "field2";
String myFIELD3 = "field3";
String myFIELD4 = "field4";
String myFIELD5 = "field5";
String myFIELD6 = "field6";
//String myFIELD7 = "field7";
//String myFIELD8 = "field8";
String lat,lng,t,x,y,z;
void setup()
{
Serial.begin(9600);
Wire.begin();
mpu6050.begin();
espSerial.begin(115200);
Serial.println("WiFi begin");
ss.begin(9600);
Serial.println("gps begin");
espData("AT+RST", 1000, DEBUG);                    //Reset the ESP8266 module
espData("AT+CWMODE=1", 1000, DEBUG);               //Set the ESP mode as station
mode
```

```
espData("AT+CWJAP=\""+ mySSID +"\",\""+ myPWD +"\"", 1000, DEBUG);
//Connect to WiFi network

  delay(1000);

}


  void loop()

  {

    mpu6050.update();

    /* while(ss.available())//While there are characters to come from the GPS

  {

gps.encode(ss.read());//This feeds the serial NMEA data into the library one char at a time

  }

lat=String(gps.location.lat());

lng=String(gps.location.lng());

Serial.print("latitude: ");

Serial.println(gps.location.lat());

Serial.print("longitude: ");

Serial.println(gps.location.lng());*/

    if(millis() - timer > 100){

    while(ss.available())//While there are characters to come from the GPS

  {

gps.encode(ss.read());//This feeds the serial NMEA data into the library one char at a time

  }

 if(gps.location.isUpdated())//This will pretty much be fired all the time anyway but will at
least reduce it to only after a package of NMEA data comes in

  {

lat=String(gps.location.lat());

lng=String(gps.location.lng());
```

```
Serial.print("latitude: ");

Serial.println(gps.location.lat());

Serial.print("longitude: ");

Serial.println(gps.location.lng());


Serial.println("================================================
==");

Serial.print("temp  :  ");Serial.println(mpu6050.getTemp());

Serial.print("accX  :  ");Serial.println(mpu6050.getAccX());

Serial.print("  accY  :  ");Serial.print(mpu6050.getAccY());

Serial.print(" accZ : ");Serial.println(mpu6050.getAccZ());

   t=String(mpu6050.getTemp());

   x=String(mpu6050.getAccX());

   y=String(mpu6050.getAccY());

   z=String(mpu6050.getAccZ());


  if(mpu6050.getAccX()>0.3)//This will pretty much be fired all the time anyway but will
at least reduce it to only after a package of NMEA data comes in

  {


    String sendData = "GET /update?api_key="+ myAPI +"&"+ myFIELD1
+"="+lat+"&"+ myFIELD2 +"="+lng+"&"+ myFIELD3 +"="+t+"&"+ myFIELD4
+"="+x+"&"+ myFIELD5 +"="+y+"&"+ myFIELD6 +"="+z;//+"&"+ myFIELD7
+"="+String("6300997006")+"&"+ myFIELD8 +"="+String("7330844898");


espData("AT+CIPMUX=1", 1000, DEBUG);       //Allow multiple connections

espData("AT+CIPSTART=0,\"TCP\",\""+ myHOST +"\","+ myPORT, 1000,DEBUG);

espData("AT+CIPSEND=0," +String(sendData.length()+4),1000,DEBUG);

espSerial.find(">");

espSerial.println(sendData);
```

```
Serial.print("latitude: ");
Serial.println(lat);
Serial.print("longitude: ");
Serial.println(lng);
espData("AT+CIPCLOSE=0",1000,DEBUG);
    delay(100);
  }
 }
  timer = millis();


  }
  }
  String espData(String command, const int timeout, boolean debug)
{
Serial.print("AT Command ==> ");
Serial.print(command);
Serial.println("     ");

  String response = "";
espSerial.println(command);
long int time = millis();
  while ( (time + timeout) >millis())
  {
    while (espSerial.available())
    {
      char c = espSerial.read();
      response += c;
```

```
    }
  }
  if (debug)
  {
    //Serial.print(response);
  }
  return response;
}
```

6.2 Website Code

home_page.html:

```
<!DOCTYPE html>
<html>
<head>
<title></title>
<link rel="stylesheet" type="text/css" href="style_home.css">
</head>
<body>
<div class=bgimage>
<div class="menu">
<div class="top">
<h1 align="center" style="color: green"> SMART EMERGENCY SAVING SYSTEM
</h1>
<h2 align="center" style="color: red">(SESS)</h2>
<h3 align="center" style="color: purple"> To save lives with in no time </h3>
</div>
</div>
<div class="bottom">
```

```
<nav>
<ul>

<li><a href="login.php" ><h3> LOGIN: Existing users </h3></a></li>
<li><a href="registration.php"><h3> REGISTER: For new users </h3></a></li>

</ul>
</nav>
</div>
</div>
</body>
</html>
```

registration.php:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>REGISTER</title>
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css" type="text/css" >
<link href="style_register.css" rel="stylesheet" type="text/css">
</head>
<body>
<h1 align="center" style="color: purple"> SMART EMERGENCY SAVING SYSTEM </h1>
```

```
<div class="register">
<h1>REGISTER (New User)</h1>
<form method="post" >

<label for="username">
<i class="fas fa-user"></i>
</label>
<h4 style="color: blue"> User Name :
<input type="text" name="username" placeholder="User_Name" required></h4>
<label for="mobile">
<i class="fas fa-mobile"></i>
</label>
<h4 style="color: blue"> Mobile Number :
<input type="text" name="Phone_Number" placeholder="Phone Number" required></h4>
<label for="password">
<i class="fas fa-lock"></i>
</label>
<h4 style="color: blue"> Password :
<input type="password" name="password" placeholder="Password" required></h4>
<label for="Family1">
<i class="fas fa-mobile"></i>
</label>
<h4 style="color: blue"> Family Number_1 :
<input type="text" name="Family_Number_1" placeholder="Family Number 01"
required></h4>


<input type="submit" name="submit" value="Register">
```

```
</form>
<h3 align="center" style="color: red"> OR </h3>
<nav>
<ul>

<li><a href="home_page.html.html" ><h4 style="color: green"> HOME: To main page
</h4></a></li>

</ul>
</nav>

</div>
</body>
</html>

<?php
$host = "localhost";
$user = "root";
$password = "";
$db = "sess_db";
// Create connection
$conn=mysqli_connect($host,$user,$password,$db);
if(isset($_POST['submit'])){
$username = $_POST['username'];
$Phone_Number = $_POST['Phone_Number'];
$password = $_POST['password'];
$Family = $_POST['Family_Number_1'];
```

```
$sql = "INSERT INTO user_details(User_Name,Mobile,Password,Family_Number_1)
values('$username','$Phone_Number','$password','$Family')";


if (mysqli_query($conn,$sql))

{

echo "<script>alert(' Registered sucessfully');</script>";

}

else

{

echo "<script>alert('Registration failed');</script>";

}

//mysqli_close($conn);

}


?>
```

login.php:

```
<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>USER LOGIN</title>

<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.1/css/all.css"
type="text/css" >

<link href="style_login.css" rel="stylesheet" type="text/css">

</head>

<body>

<h1 align="center" style="color: purple"> SMART EMERGENCY SAVING SYSTEM
</h1>
```

```
<div class="login">

<h1>LOGIN (Existing User)</h1>

<form method="post" autocomplete="off">


<label for="username">

<i class="fas fa-user"></i>

</label>

<h4 style="color: blue"> User Name :

<input type="text" name="username" placeholder="Username" id="username"
required></h4>


<label for="password">

<i class="fas fa-lock"></i>

</label>

<h4 style="color: blue"> Password :

<input type="password" name="password" placeholder="Password" id="password"
required></h4>


<input type="submit" name="login" value="Login">

<h3 align="center" style="color: red"> OR </h3>

<nav>

<ul>


<li><a href="home_page.html.html" ><h4 style="color: green"> HOME: To main page
</h4></a></li>


</ul>
```

```
</nav>
</form>
</div>
</body>
</html>
<?php

$host="localhost";
$user="root";
$password="";
$db="sess_db";

$con=mysqli_connect($host,$user,$password,$db);

if(isset($_POST['login'])){

    $uname=$_POST['username'];
    $password=$_POST['password'];

    $sql="select * from user_details where User_Name='".$uname."'AND
Password='".$password."' limit 1";

    $result=mysqli_query($con,$sql);

    if(mysqli_num_rows($result)==1){
        //header('Location:thingspeak.php');
        echo "<script>alert(' You Have Successfully Logged in');</script>";
        header('Location:thingspeak.php');
```

```
    }
    else{
        echo "<script>alert('You Have Entered Incorrect Password');</script>";

    }
     $con->close();
}
?>
```

thingspeak.php:

```
<!DOCTYPE html>
<html>
<head>

<title>USER LOGGED IN</title>
</head>
<body>
        <div class="User loged in">
        <h1>WELCOME</h1>

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/972273/charts/1?bgcolor=%23ffffff&color=%23d62
020&dynamic=true&results=60&type=line&update=15"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/972273/charts/2?bgcolor=%23ffffff&color=%23d62
020&dynamic=true&results=60&type=line&update=15"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/972273/charts/3?bgcolor=%23ffffff&color=%23d62
020&dynamic=true&results=60&type=line&update=15"></iframe>
```

```
<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/972273/charts/4?bgcolor=%23ffffff&color=%23d62
020&dynamic=true&results=60&type=line&update=15"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/972273/charts/5?bgcolor=%23ffffff&color=%23d62
020&dynamic=true&results=60&type=line&update=15"></iframe>

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/972273/charts/6?bgcolor=%23ffffff&color=%23d62
020&dynamic=true&results=60&type=line&update=15"></iframe>

<nav>

<a href="logout.php" ><h3 align="center" style="color: purple"><input type="logout"
name="Logout" value="Logout"></h3></a>

</nav>

</div>

</body>

</html>
```

6.3 Dashboard Code

Login:

```
using SQLite;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SQLite;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DashBoard
{
   public partial class Login : Form
   {
      public string path = null;
      public Login()
      {
```

```
InitializeComponent();
        }

        private void btnLogin_Click(object sender, EventArgs e)
        {
            string user = null;
            string Phone = null;

            string querry = "SELECT * from Db WHERE Username='"+txtUser.Text+"' and
Password='"+txtPassword.Text+"'";
System.Data.SQLite.SQLiteConnection _conn = new
System.Data.SQLite.SQLiteConnection(@"Data Source="+path+"; Version = 3; New =
True; Compress = True; ");
            _conn.Open();
System.Data.SQLite.SQLiteCommand _select = new
System.Data.SQLite.SQLiteCommand(querry, _conn);
SQLiteDataReaderdr = _select.ExecuteReader();
            var count = 0;
            while (dr.Read())
            {
                count += 1;
                 user = dr.GetString(1);
                 Phone = dr.GetInt64(3).ToString();
            }
            if (count == 1)
            {

                //this.Hide();
                Form1 fr = new Form1(user,Phone);
fr.Show();

            }else
            {
MessageBox.Show("Invalid Username and Password");
            }

        }

        private void btnClose_Click(object sender, EventArgs e)
        {
            Close();
        }

        private void Login_FormClosing(object sender, FormClosingEventArgs e)
        {
Application.Exit();
```

```
        }

        private void btnLoad_Click(object sender, EventArgs e)
        {
            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                path = openFileDialog1.FileName;
            }
        }
    }
}
```

Form1.Designer:

```
namespace DashBoard
{
    partial class Form1
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
components.Dispose();
            }
base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
```

```
        {
System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea1 = new
System.Windows.Forms.DataVisualization.Charting.ChartArea();
System.Windows.Forms.DataVisualization.Charting.Legend legend1 = new
System.Windows.Forms.DataVisualization.Charting.Legend();
System.Windows.Forms.DataVisualization.Charting.Series series1 = new
System.Windows.Forms.DataVisualization.Charting.Series();
System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea2 = new
System.Windows.Forms.DataVisualization.Charting.ChartArea();
System.Windows.Forms.DataVisualization.Charting.Legend legend2 = new
System.Windows.Forms.DataVisualization.Charting.Legend();
System.Windows.Forms.DataVisualization.Charting.Series series2 = new
System.Windows.Forms.DataVisualization.Charting.Series();
System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea3 = new
System.Windows.Forms.DataVisualization.Charting.ChartArea();
System.Windows.Forms.DataVisualization.Charting.Legend legend3 = new
System.Windows.Forms.DataVisualization.Charting.Legend();
System.Windows.Forms.DataVisualization.Charting.Series series3 = new
System.Windows.Forms.DataVisualization.Charting.Series();
System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea4 = new
System.Windows.Forms.DataVisualization.Charting.ChartArea();
System.Windows.Forms.DataVisualization.Charting.Legend legend4 = new
System.Windows.Forms.DataVisualization.Charting.Legend();
System.Windows.Forms.DataVisualization.Charting.Series series4 = new
System.Windows.Forms.DataVisualization.Charting.Series();
System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea5 = new
System.Windows.Forms.DataVisualization.Charting.ChartArea();
System.Windows.Forms.DataVisualization.Charting.Legend legend5 = new
System.Windows.Forms.DataVisualization.Charting.Legend();
System.Windows.Forms.DataVisualization.Charting.Series series5 = new
System.Windows.Forms.DataVisualization.Charting.Series();
            this.dataGridView1 = new System.Windows.Forms.DataGridView();
            this.chart1 = new System.Windows.Forms.DataVisualization.Charting.Chart();
            this.chart2 = new System.Windows.Forms.DataVisualization.Charting.Chart();
            this.chart3 = new System.Windows.Forms.DataVisualization.Charting.Chart();
            this.chart4 = new System.Windows.Forms.DataVisualization.Charting.Chart();
            this.chart5 = new System.Windows.Forms.DataVisualization.Charting.Chart();
            this.button1 = new System.Windows.Forms.Button();
this.lblUser = new System.Windows.Forms.Label();
this.lblPhone = new System.Windows.Forms.Label();
            ((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.chart1)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.chart2)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.chart3)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.chart4)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.chart5)).BeginInit();
```

```
this.SuspendLayout();
        //
        // dataGridView1
        //
        this.dataGridView1.BackgroundColor =
System.Drawing.SystemColors.ButtonHighlight;
        this.dataGridView1.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
        this.dataGridView1.Location = new System.Drawing.Point(3, 12);
        this.dataGridView1.Name = "dataGridView1";
        this.dataGridView1.RowHeadersWidth = 51;
        this.dataGridView1.RowTemplate.Height = 24;
        this.dataGridView1.Size = new System.Drawing.Size(728, 243);
        this.dataGridView1.TabIndex = 0;
        //
        // chart1
        //
        chartArea1.Name = "ChartArea1";
        this.chart1.ChartAreas.Add(chartArea1);
        legend1.Name = "Legend1";
        this.chart1.Legends.Add(legend1);
        this.chart1.Location = new System.Drawing.Point(891, 3);
        this.chart1.Name = "chart1";
        series1.ChartArea = "ChartArea1";
        series1.Legend = "Legend1";
        series1.Name = "Series1";
        this.chart1.Series.Add(series1);
        this.chart1.Size = new System.Drawing.Size(751, 275);
        this.chart1.TabIndex = 1;
        this.chart1.Text = "chart1";
        //
        // chart2
        //
        chartArea2.Name = "ChartArea1";
        this.chart2.ChartAreas.Add(chartArea2);
        legend2.Name = "Legend1";
        this.chart2.Legends.Add(legend2);
        this.chart2.Location = new System.Drawing.Point(12, 266);
        this.chart2.Name = "chart2";
        series2.ChartArea = "ChartArea1";
        series2.Legend = "Legend1";
        series2.Name = "Series1";
        this.chart2.Series.Add(series2);
        this.chart2.Size = new System.Drawing.Size(705, 372);
        this.chart2.TabIndex = 2;
        this.chart2.Text = "chart2";
```

```
//
// chart3
//
chartArea3.Name = "ChartArea1";
this.chart3.ChartAreas.Add(chartArea3);
legend3.Name = "Legend1";
this.chart3.Legends.Add(legend3);
this.chart3.Location = new System.Drawing.Point(723, 284);
this.chart3.Name = "chart3";
series3.ChartArea = "ChartArea1";
series3.Legend = "Legend1";
series3.Name = "Series1";
this.chart3.Series.Add(series3);
this.chart3.Size = new System.Drawing.Size(1005, 354);
this.chart3.TabIndex = 3;
this.chart3.Text = "chart3";
//
// chart4
//
chartArea4.Name = "ChartArea1";
this.chart4.ChartAreas.Add(chartArea4);
legend4.Name = "Legend1";
this.chart4.Legends.Add(legend4);
this.chart4.Location = new System.Drawing.Point(30, 666);
this.chart4.Name = "chart4";
series4.ChartArea = "ChartArea1";
series4.Legend = "Legend1";
series4.Name = "Series1";
this.chart4.Series.Add(series4);
this.chart4.Size = new System.Drawing.Size(759, 343);
this.chart4.TabIndex = 4;
this.chart4.Text = "chart4";
//
// chart5
//
chartArea5.Name = "ChartArea1";
this.chart5.ChartAreas.Add(chartArea5);
legend5.Name = "Legend1";
this.chart5.Legends.Add(legend5);
this.chart5.Location = new System.Drawing.Point(737, 666);
this.chart5.Name = "chart5";
series5.ChartArea = "ChartArea1";
series5.Legend = "Legend1";
series5.Name = "Series1";
this.chart5.Series.Add(series5);
this.chart5.Size = new System.Drawing.Size(991, 343);
```

```
            this.chart5.TabIndex = 5;
            this.chart5.Text = "chart5";
            //
            // button1
            //
            this.button1.Location = new System.Drawing.Point(740, 15);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(196, 239);
            this.button1.TabIndex = 6;
            this.button1.Text = "Track";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.button1_Click);
            //
            // lblUser
            //
this.lblUser.AutoSize = true;
this.lblUser.Font = new System.Drawing.Font("Microsoft Uighur", 25.8F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
this.lblUser.Location = new System.Drawing.Point(1617, 15);
this.lblUser.Name = "lblUser";
this.lblUser.Size = new System.Drawing.Size(0, 52);
this.lblUser.TabIndex = 7;
            //
            // lblPhone
            //
this.lblPhone.AutoSize = true;
this.lblPhone.Font = new System.Drawing.Font("Microsoft Uighur", 22.2F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,((byte)(0)));
this.lblPhone.Location = new System.Drawing.Point(1613, 95);
this.lblPhone.Name = "lblPhone";
this.lblPhone.Size = new System.Drawing.Size(0, 46);
this.lblPhone.TabIndex = 8;
            //
            // Form1
            //
this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.BackColor = System.Drawing.SystemColors.ControlLightLight;
this.ClientSize = new System.Drawing.Size(1757, 1055);
this.Controls.Add(this.lblPhone);
this.Controls.Add(this.lblUser);
this.Controls.Add(this.button1);
this.Controls.Add(this.chart5);
this.Controls.Add(this.chart4);
this.Controls.Add(this.chart3);
this.Controls.Add(this.chart2);
```

```
this.Controls.Add(this.chart1);
this.Controls.Add(this.dataGridView1);
this.Name = "Form1";
this.Text = "DashBoard";
this.Load += new System.EventHandler(this.Form1_Load);
        ((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.chart1)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.chart2)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.chart3)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.chart4)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.chart5)).EndInit();
this.ResumeLayout(false);
this.PerformLayout();

    }

    #endregion

    private System.Windows.Forms.DataGridView dataGridView1;
    private System.Windows.Forms.DataVisualization.Charting.Chart chart1;
    private System.Windows.Forms.DataVisualization.Charting.Chart chart2;
    private System.Windows.Forms.DataVisualization.Charting.Chart chart3;
    private System.Windows.Forms.DataVisualization.Charting.Chart chart4;
    private System.Windows.Forms.DataVisualization.Charting.Chart chart5;
    private System.Windows.Forms.Button button1;
    private   System.Windows.Forms.LabellblUser;
    privateSystem.Windows.Forms.LabellblPhone;
  }
}
```

TrackMap.Designer:

```
namespace DashBoard
{
  partial class TrackMap
  {
    /// <summary>
    /// Required designer variable.
    /// </summary>
    private System.ComponentModel.IContainer components = null;

    /// <summary>
    /// Clean up any resources being used.
    /// </summary>
```

```
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
components.Dispose();
            }
base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.gMapControl1 = new GMap.NET.WindowsForms.GMapControl();
this.SuspendLayout();
            //
            // gMapControl1
            //
            this.gMapControl1.Bearing = 0F;
            this.gMapControl1.CanDragMap = true;
            this.gMapControl1.EmptyTileColor = System.Drawing.Color.Navy;
            this.gMapControl1.GrayScaleMode = false;
            this.gMapControl1.HelperLineOption =
GMap.NET.WindowsForms.HelperLineOptions.DontShow;
            this.gMapControl1.LevelsKeepInMemmory = 5;
            this.gMapControl1.Location = new System.Drawing.Point(23, 12);
            this.gMapControl1.MarkersEnabled = true;
            this.gMapControl1.MaxZoom = 2;
            this.gMapControl1.MinZoom = 2;
            this.gMapControl1.MouseWheelZoomEnabled = true;
            this.gMapControl1.MouseWheelZoomType =
GMap.NET.MouseWheelZoomType.MousePositionAndCenter;
            this.gMapControl1.Name = "gMapControl1";
            this.gMapControl1.NegativeMode = false;
            this.gMapControl1.PolygonsEnabled = true;
            this.gMapControl1.RetryLoadTile = 0;
            this.gMapControl1.RoutesEnabled = true;
            this.gMapControl1.ScaleMode = GMap.NET.WindowsForms.ScaleModes.Integer;
```

```
            this.gMapControl1.SelectedAreaFillColor =
System.Drawing.Color.FromArgb(((int)(((byte)(33)))), ((int)(((byte)(65)))),
((int)(((byte)(105)))), ((int)(((byte)(225)))));
            this.gMapControl1.ShowTileGridLines = false;
            this.gMapControl1.Size = new System.Drawing.Size(1199, 658);
            this.gMapControl1.TabIndex = 0;
            this.gMapControl1.Zoom = 0D;
            //
            // TrackMap
            //
this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(1278, 770);
this.Controls.Add(this.gMapControl1);
this.Name = "TrackMap";
this.Text = "TrackMap";
this.Load += new System.EventHandler(this.TrackMap_Load);
this.ResumeLayout(false);

        }

        #endregion

        private GMap.NET.WindowsForms.GMapControl gMapControl1;
    }
}
```

Login.Designer:

```
namespace DashBoard
{
    partial class Login
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
```

```
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
components.Dispose();
            }
base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.label1 = new System.Windows.Forms.Label();
            this.panel1 = new System.Windows.Forms.Panel();
this.btnClose = new System.Windows.Forms.Button();
this.btnLogin = new System.Windows.Forms.Button();
this.txtPassword = new System.Windows.Forms.TextBox();
this.txtUser = new System.Windows.Forms.TextBox();
            this.label3 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
this.btnLoad = new System.Windows.Forms.Button();
            this.openFileDialog1 = new System.Windows.Forms.OpenFileDialog();
            this.panel1.SuspendLayout();
this.SuspendLayout();
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Font = new System.Drawing.Font("Microsoft Uighur", 25.8F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.ForeColor = System.Drawing.Color.LightSeaGreen;
            this.label1.Location = new System.Drawing.Point(51, 47);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(353, 51);
            this.label1.TabIndex = 0;
            this.label1.Text = "Accident Detection Login";
            //
            // panel1
            //
            this.panel1.BackColor = System.Drawing.Color.LightSeaGreen;
            this.panel1.Controls.Add(this.btnLoad);
```

```
            this.panel1.Controls.Add(this.btnClose);
            this.panel1.Controls.Add(this.btnLogin);
            this.panel1.Controls.Add(this.txtPassword);
            this.panel1.Controls.Add(this.txtUser);
            this.panel1.Controls.Add(this.label3);
            this.panel1.Controls.Add(this.label2);
            this.panel1.Location = new System.Drawing.Point(33, 124);
            this.panel1.Name = "panel1";
            this.panel1.Size = new System.Drawing.Size(377, 351);
            this.panel1.TabIndex = 1;
            //
            // btnClose
            //
this.btnClose.BackColor = System.Drawing.Color.DeepPink;
this.btnClose.ForeColor = System.Drawing.SystemColors.ButtonFace;
this.btnClose.Location = new System.Drawing.Point(231, 212);
this.btnClose.Name = "btnClose";
this.btnClose.Size = new System.Drawing.Size(111, 39);
this.btnClose.TabIndex = 5;
this.btnClose.Text = "Close";
this.btnClose.UseVisualStyleBackColor = false;
this.btnClose.Click += new System.EventHandler(this.btnClose_Click);
            //
            // btnLogin
            //
this.btnLogin.BackColor = System.Drawing.Color.DeepSkyBlue;
this.btnLogin.ForeColor = System.Drawing.SystemColors.ButtonHighlight;
this.btnLogin.Location = new System.Drawing.Point(114, 212);
this.btnLogin.Name = "btnLogin";
this.btnLogin.Size = new System.Drawing.Size(111, 39);
this.btnLogin.TabIndex = 4;
this.btnLogin.Text = "Login";
this.btnLogin.UseVisualStyleBackColor = false;
this.btnLogin.Click += new System.EventHandler(this.btnLogin_Click);
            //
            // txtPassword
            //
this.txtPassword.Location = new System.Drawing.Point(103, 136);
this.txtPassword.Name = "txtPassword";
this.txtPassword.PasswordChar = '*';
this.txtPassword.Size = new System.Drawing.Size(233, 22);
this.txtPassword.TabIndex = 3;
            //
            // txtUser
            //
this.txtUser.Location = new System.Drawing.Point(107, 75);
```

```
this.txtUser.Name = "txtUser";
this.txtUser.Size = new System.Drawing.Size(233, 22);
this.txtUser.TabIndex = 2;
        //
        // label3
        //
        this.label3.AutoSize = true;
        this.label3.ForeColor = System.Drawing.SystemColors.ButtonHighlight;
        this.label3.Location = new System.Drawing.Point(27, 141);
        this.label3.Name = "label3";
        this.label3.Size = new System.Drawing.Size(69, 17);
        this.label3.TabIndex = 1;
        this.label3.Text = "Password";
        //
        // label2
        //
        this.label2.AutoSize = true;
        this.label2.ForeColor = System.Drawing.SystemColors.ButtonHighlight;
        this.label2.Location = new System.Drawing.Point(27, 78);
        this.label2.Name = "label2";
        this.label2.Size = new System.Drawing.Size(75, 17);
        this.label2.TabIndex = 0;
        this.label2.Text = "UserName";
        //
        // btnLoad
        //
this.btnLoad.Location = new System.Drawing.Point(79, 19);
this.btnLoad.Name = "btnLoad";
this.btnLoad.Size = new System.Drawing.Size(256, 37);
this.btnLoad.TabIndex = 6;
this.btnLoad.Text = "LoadDB";
this.btnLoad.UseVisualStyleBackColor = true;
this.btnLoad.Click += new System.EventHandler(this.btnLoad_Click);
        //
        // openFileDialog1
        //
        this.openFileDialog1.FileName = "openFileDialog1";
        //
        // Login
        //
this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.BackColor = System.Drawing.SystemColors.ButtonHighlight;
this.ClientSize = new System.Drawing.Size(441, 567);
this.Controls.Add(this.panel1);
this.Controls.Add(this.label1);
```

```
this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.None;
this.Name = "Login";
this.Text = "Login";
this.FormClosing += new
System.Windows.Forms.FormClosingEventHandler(this.Login_FormClosing);
        this.panel1.ResumeLayout(false);
        this.panel1.PerformLayout();
this.ResumeLayout(false);
this.PerformLayout();

    }

    #endregion

    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.Panel panel1;
    private System.Windows.Forms.ButtonbtnClose;
    private System.Windows.Forms.ButtonbtnLogin;
    private System.Windows.Forms.TextBoxtxtPassword;
    private System.Windows.Forms.TextBoxtxtUser;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.ButtonbtnLoad;
    private System.Windows.Forms.OpenFileDialog openFileDialog1;
  }
}
```

TrackMap:

```
using GMap.NET;
using GMap.NET.WindowsForms;
using GMap.NET.WindowsForms.Markers;
using System;
usingSystem.Collections.Generic;
using System.ComponentModel;
using System.Data;
usingSystem.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DashBoard
{
```

```
   public partial class TrackMap : Form
   {
      public DataTabledts;
      public List<PointLatLng> _point;
      public TrackMap(DataTable dt)
      {
dts = dt;
         _point = new List<PointLatLng>();
InitializeComponent();
      }

      private void toolStripContainer1_ContentPanel_Load(object sender, EventArgs e)
      {

      }

      private void toolStripContainer1_TopToolStripPanel_Click(object sender, EventArgs
e)
      {

      }

      private void toolStripContainer1_LeftToolStripPanel_Click(object sender, EventArgs
e)
      {

      }

      private void TrackMap_Load(object sender, EventArgs e)
      {
         gMapControl1.DragButton = MouseButtons.Right;
         gMapControl1.ShowCenter = false;
         gMapControl1.MapProvider =
GMap.NET.MapProviders.GoogleMapProvider.Instance;
GMap.NET.GMaps.Instance.Mode = GMap.NET.AccessMode.ServerOnly;
         //gMapControl1.Position = new GMap.NET.PointLatLng(16.96, 82.23);
         //gMapControl1.SetPositionByKeywords("India, Andhrapradesh");
         gMapControl1.MinZoom = 0;
         gMapControl1.MaxZoom = 32;
         gMapControl1.Zoom = 14;
         for (int i = 0; i<dts.Rows.Count; i++)
         {
            _point.Add(new
PointLatLng(Convert.ToDouble(dts.Rows[i]["field1"].ToString()),
Convert.ToDouble(dts.Rows[i]["field2"].ToString())));
```

```
            gMapControl1.Position = new
PointLatLng(Convert.ToDouble(dts.Rows[i]["field1"].ToString()),
Convert.ToDouble(dts.Rows[i]["field2"].ToString()));
            var markers = new GMapOverlay("marker");
            var marker = new GMarkerGoogle(new
PointLatLng(Convert.ToDouble(dts.Rows[i]["field1"].ToString()),
Convert.ToDouble(dts.Rows[i]["field2"].ToString())), GMarkerGoogleType.red_small);
markers.Markers.Add(marker);
            gMapControl1.Overlays.Add(markers);

        }
      }
   }
}
```

Form1:

```
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Net;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;

namespace DashBoard
{
   public partial class Form1 : Form
   {
      public DataTabledistinctValues;
      public string user;
      public string mobile;
      public Form1(string username,string phone)
      {
         user = username;
         mobile = phone;
InitializeComponent();
      }
```

```
        private void Form1_Load(object sender, EventArgs e)
        {
lblUser.Text = user;
lblPhone.Text = mobile;
            var url =
"https://api.thingspeak.com/channels/972273/feeds.json?api_key=XKUZJGKT8OPZVMK
J&results=200";
            // for a simple get, use WebClient
            var json = new WebClient().DownloadString(url);
            var root = JsonConvert.DeserializeObject<Bustracking>(json);
DataTable dt = new DataTable();
dt.Columns.Add("created_at", typeof(string));
dt.Columns.Add("entry_id", typeof(string));
dt.Columns.Add("field1", typeof(double));
dt.Columns.Add("field2", typeof(double));
dt.Columns.Add("field3", typeof(double));
dt.Columns.Add("field4", typeof(double));
dt.Columns.Add("field5", typeof(double));
            foreach (var i in root.feeds)
            {
dt.Rows.Add(i.created_at, i.entry_id, i.field1, i.field2, i.field3, i.field4, i.field5);
            }


DataView view = new DataView(dt);
DataTable distinctValues1 = view.ToTable(true, "created_at", "field1");
DataTable distinctValues2 = view.ToTable(true, "created_at", "field2");
DataTable distinctValues3 = view.ToTable(true, "created_at", "field3");
DataTable distinctValues4 = view.ToTable(true, "created_at", "field4");
DataTable distinctValues5 = view.ToTable(true, "created_at", "field5");

distinctValues = view.ToTable(true, "field1", "field2");
            dataGridView1.DataSource = dt;
            dataGridView1.Refresh();
            Chart(distinctValues1,"field1",chart1,"Latitude",SeriesChartType.Radar,
Color.Blue);
            Chart(distinctValues2,"field2",chart2,"Longitude", SeriesChartType.Radar,
Color.Green);
            Chart(distinctValues3,"field3",chart3,"Temprature", SeriesChartType.Line,
Color.Red);
            Chart(distinctValues4,"field4",chart4,"Accelerometer X", SeriesChartType.Area,
Color.Coral);
            Chart(distinctValues5,"field5",chart5, "Accelerometer Y",
SeriesChartType.Bubble, Color.Pink);

        }
```

```
        private void Chart(DataTable distinctValues1,string
y,Chartchart,stringtitle,SeriesChartTypety,Color x)
        {
chart.Series["Series1"].ChartType = ty;
chart.DataSource = distinctValues1;
chart.Series["Series1"].XValueMember = "created_at";
chart.Series["Series1"].YValueMembers = y;
chart.Series["Series1"].Color = x;

chart.Titles.Add(title);
        }

        private void button1_Click(object sender, EventArgs e)
        {
TrackMaptrc = new TrackMap(distinctValues);
trc.Show();
        }
    }
    public class Bustracking
    {
        public List<Result> feeds { get; set; }
    }
    public class Result
    {
        public string created_at { get; set; }
        public string entry_id { get; set; }
        public string field1 { get; set; }
        public string field2 { get; set; }
        public string field3 { get; set; }
        public string field4 { get; set; }
        public string field5 { get; set; }
    }
}
```

Chapter 7

Testing

7. TESTING

7.1 Introduction

Testing is one of the most important phases in the software development activity. In software development life cycle (SDLC), the main aim of testing process is the quality, the developed software is tested against attaining the required functionality and performance. The success of the testing process in determining the errors is mostly depends upon the test cases criteria, for testing any software we need to have a description of the expected behavior of the system and the method of determining whether the observed behavior confirmed to the expected behavior.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well- planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

7.2 Strategies

A Strategy for software testing indicates software test case design methods in a well- planned series of steps that results in the successful construction of the software. This strategy provides a road map that describes the steps to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources are required. It must be rigid enough to promote reasonable planning and management tracking as the project progresses. The strategies for testing are envisioned by the following methods. A number of software testing strategies have been proposed. All provide the software developer with a template for testing and all the following generic characteristics.

- To perform effective testing, a software team should conduct effective formal technical reviews. By doing this, many errors will be eliminated before testing commences.
- Testing begins at the component level and works “outward” toward the integration of entire computer-based system.

- Different testing techniques are appropriate at different points in time.
- The developer of the software and an independent test group conducts testing.
- Testing and debugging are different activities, but debugging must accommodate in a testing strategy.

A strategy for software testing must accommodate low level test that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validates major system functions against customer requirements.

7.3 Types of Testing

7.3.1 Unit Testing

Unit test focuses verification effort on the smallest unit of software design- the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The unit test focuses on the internal processing login and data structures within the boundaries of a component. This type of testing can be conducted in parallel for multiple components. The module interface is tested to ensure that information properly flows into and out of the program until undertest.

7.3.2 Integration Testing

A neophyte in the software world might ask a seemingly legitimate question once all modules have been unit testes." if they all work individually", the problem, of course, is "putting them together" interfacing. Data can be lost across an interface; one module can have an inadvertent, adverse effect on another, sub functions, when combined, may not produce the desired major functions; global data structures can present problems.

7.3.2.1 Top down Integration Testing

Top down Integration testing is an incremental approach to construction of the software architecture. Modules are integrated by moving downward through the control.

Coterie Line 60 hierarchy, beginning with the main control module. Modules subordinate to the main control module are incorporate into the structure in either a depth-first or breadth- first manner.

7.3.2.2 Bottom up Integration Testing

Bottom up Integration testing as its name implies, begins construction and testing with atomic modules. Because components are integrated from the bottom up, processing required for component subordinate to a given level is always available and the need for stubs is eliminated.

7.3.3 Black Box Testing

Knowing the specified that product has been designed to perform, test can be conducted that demonstrate each function is fully operational while at the same time searching for errors in each function.

7.3.4 White Box Testing

White Box testing, sometimes called glass-box testing is a test case design philosophy that uses the control structure described as part of component-level design to derive test case. Using white-box testing methods, the software engineer can define test cases that:

7.3.4.1 Guarantee that all independent paths within a module have been exercised at least once.
7.3.4.2 Exercise all logical decisions on their true and false sides.
7.3.4.3 Exercise all loops at their boundaries and within their operational bounds.

Validation:

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed.

Validations performed during login page. Proper validations are performed such that users with invalid login ID or password cannot access this application.

7.4 Test cases:

Testcase id/Name	Input	Description	Expected Result	Actual Result	Status
ESP8266	Checks for the internet to get Connected	The uploading of data to cloud takes place after getting connected to internet.	Gets connected to internet.	Gets connected to internet.	Pass
ESP8266	Checks for the internet to get Connected	The uploading of data to cloud takes place after getting connected to internet.	Gets connected to internet.	Doesn't get connected to the internet.	Fail

GPS	Checks for the satellite signal.	GPS should get connected to satellite signals in Order to show us the current location.	GPS sends the location coordinates.	GPS sends the location co-ordinates.	Pass
GPS	Checks for the satellite signal.	GPS should get connected to satellite signals in Order to show us the current location.	GPS sends the location coordinates.	GPS couldn't lock the satellite signal to send location co-ordinates	Fail
GSM	Check for the Network	GSM is used for cellular communication and it should connect to the network.	if we call then it should ring	send SMS alert	**Pass**
GSM	Check for the Network	GSM is used for cellular communication and it should connect to the network.	if we call then it should ring	Didn't connected to network	**Fail**
MPU6050	Move its axis	It consists of 3-axis Accelerometer, 3-axis Gyroscope.	Arduino will display the xyz axes values	Accident detected	**Pass**
MPU6050	Move its axis	It consists of 3-axis Accelerometer, 3-axis Gyroscope.	Arduino will display the xyz axes values	Accident is not happened.	**Fail**

Table 7.4.1 Test Case for Circuit

Test case ID	Input	Description	Expected result	Actual Result	Pass / Fail
1	Blank password and username	A blank username and password given by the user	Display username and password is required	Display username and password is required	Pass
2	Valid username and blank password	A valid username and blank password given by the user	Display password is required	Display password is required	Pass
3	Username blank and valid password	A blank username and password given by the user	Display username is required	Display username is required	Pass
4	Valid Username and valid password	A valid username and valid password given by the user	Login successfully	Login successfully	Pass
5	Invalid username and Invalid	An invalid username and invalid password are given by the user	Display user don't exist	Display user don't exist	Pass
6	Valid username with special character and valid password with special character	An valid username and password with special characters are given by the user	Login successfully	Login successfully	Pass
7	Invalid username and password with special characters	An invalid username and Password with special characters are given by user	Display user don't exist	Display user don't exist	Pass

Table 7.4.2 Test Case for Web page

Test case ID	Input	Description	Expected Result	Actual Result
1.	Blank user id	A blank user id is given by the user	Display user id is required	Display user id is required
2.	Username with numbers	A username with numbers is given by the user	Display valid username is required	Display valid username is required
3.	Blank password	A blank password is given by the User	Display password is required	Display password is required
4.	Username<minimum length	A username less than the minimum length is given by the user	Display valid username is required	Display valid username is required
5.	Password<minimum length	A password less than the minimum length is given by user	Display valid password is required	Display valid password is required

Table 7.4.3 Test Case for Web page/ Data Base

Chapter 8

Output Screens

8. OUTPUT SCREENS

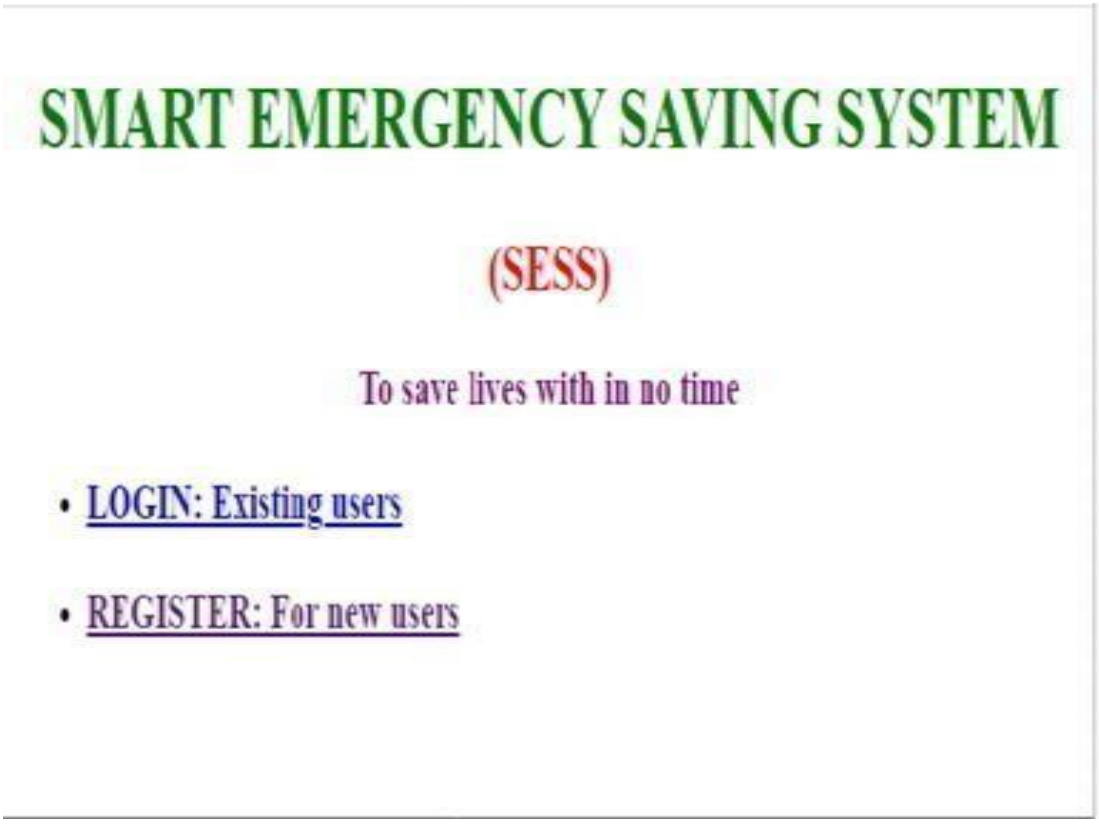


Fig 8.1 Main Web Page

Fig8.2 Website Registration Page

Fig 8.3 Website Login Page

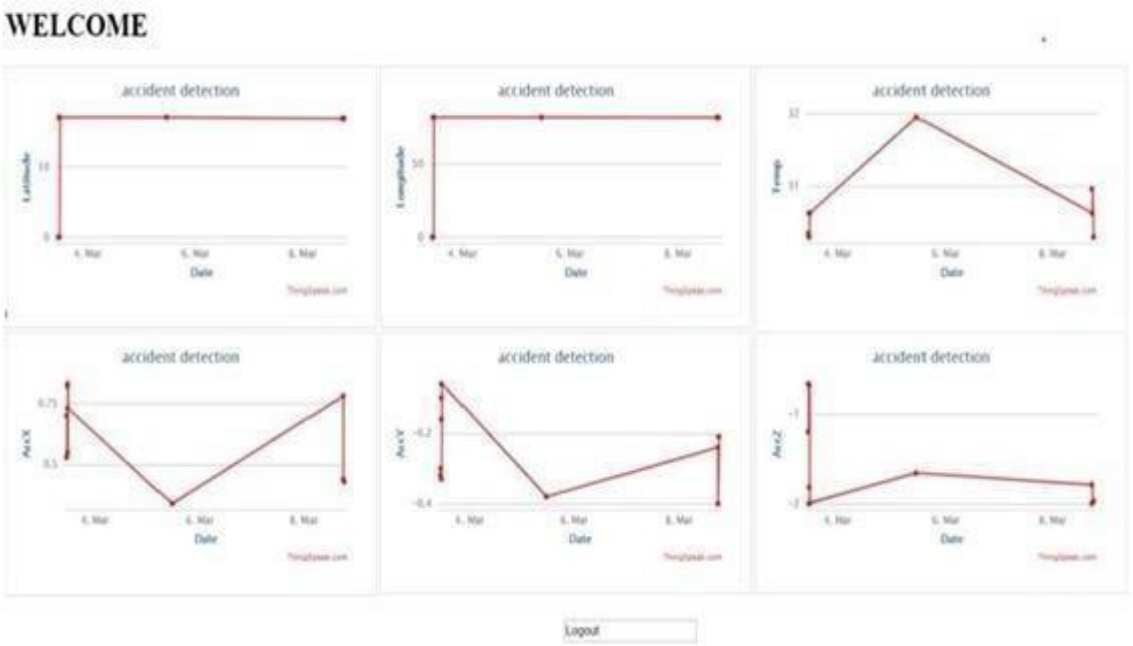


Fig 8.4 After Logging In (Main Page)

Fig 8.5 Dashboard Login Page

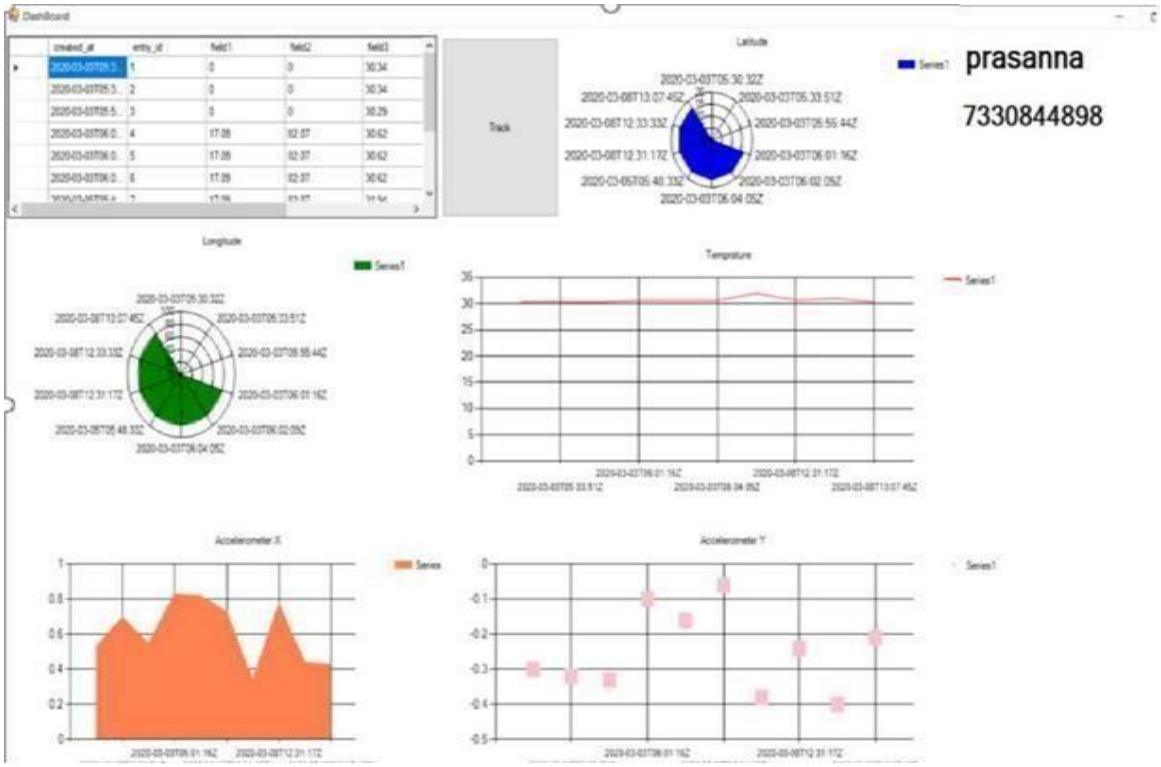


Fig 8.6 User Dashboard

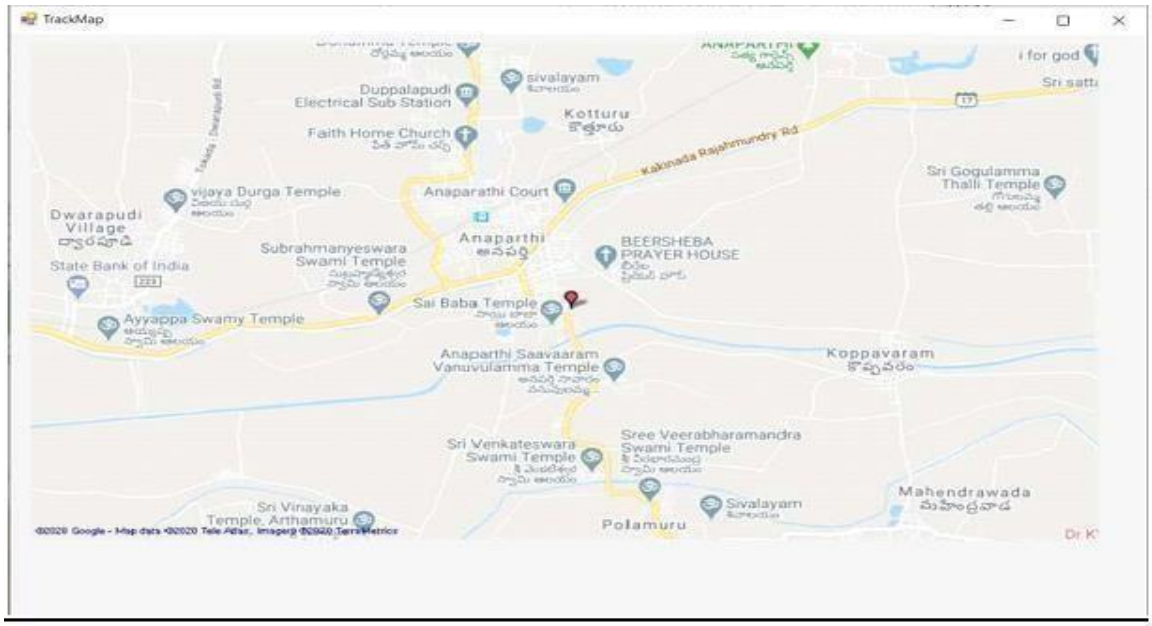


Fig 8.7 Google maps location

Fig 8.8 SMS Alert

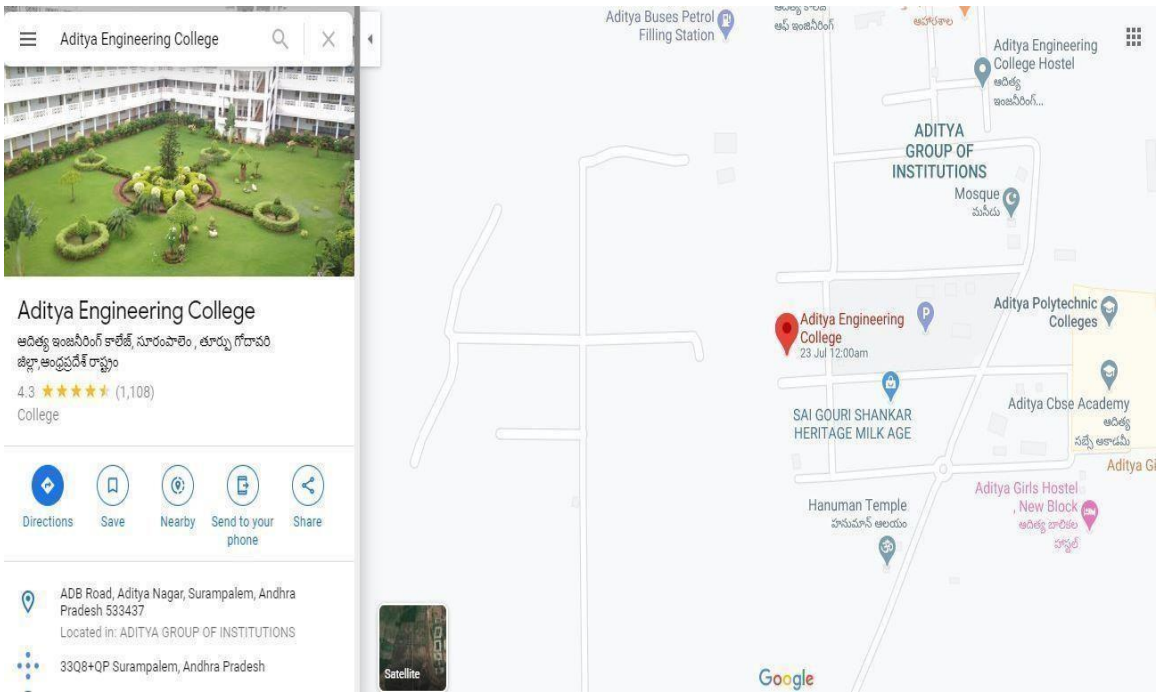


Fig 8.9 Accident location

Conclusion and Future Work

The proposed system is much better that the existing system what we have been using right now. As we have multiple alert systems in the proposed system this can be used in any kind of emergency applications. The proposed system detects the accident and it's impact then sends alerts to family members, ambulance system and police station. The proposed system also sends accident location along with the alerts. By this project we can say that our system can help much better and effective as far as the process and working concern.

The future scope will like integrating the same with drones, which would be used to dispense any kind of emergency medicine kits in case of accident. Also sending accident and victim details to the nearby hospitals.

References

1. Grady Booch, James Rumbaugh, Ivar Jacobson: The Unified Modelling Language User Guide, Person Education.

2. Database Management System, Raghu Rama Krishnan, Johannes Gehrke, TATA McGraw Hill 3rd edition

3. Web programming, building internet applications, Chris Bates 2nd edition, WILEY- Dreamtech.

4. Software Engineering, A practitioner's Approach-Roger S. Pressmen, 6th edition, McGraw Hill International Edition.

- https://circuits4you.com/2017/12/09/thingspeak-esp8266/
- https://www.instructactables.com
- https://www.electronicshub.org/connect-esp8266-to-thingspeak/
- https://www.electroschematics.com/neo-6m-gps-module/
- http://www.tutorialspoint.com/php/php_and_mysql.htm
- http://www.w3schools.com/php/default.asp
- http://www.w3schools.com/html/default.asp
- http://www.mysql.com/downloads/index.html