

TABLE OF CONTENTS

PROJECT SYNOPSIS	1
CHAPTER 1:EDLC.....	
1.1 Need.....	3
1.2 Conceptualization/concept scope	3
1.3 Targeted end user.....	3
1.4 Analysis.....	3
1.4.1Functional Requirement	3
1.4.2Non-Functional Requirement	3
1.5 Circuit diagram	4
CHAPTER 2: INTRODUCTION	
2.1 BACKGROUND.....	12
2.2 OBJECTIVES	12
2.3 PURPOSE, SCOPE AND APPLICABILITY	12
2.3.1PURPOSE.....	12
2.3.2SCOPE.....	12
2.4 ACHIEVEMENTS	12
CHAPTER 3: SURVEY OF TECHNOLOGY.....	13
CHAPTER 4: REQUIREMENTS AND ANALYSIS	
4.1 Problem definition:	14
4.1.1 Problem description:.....	14
4.1.2 Sub-problems:	14
4.2 Requirement specification:.....	14
4.2.1 Requirement Gathering:	15
4.2.2 Requirement analysis:	16
4.3 planning and scheduling	19
4.3.1 Activity table.....	19
4.3.2 Gantt chart	20
4.4 Hardware and software requirement.....	23
4.5Conceptual Model	
4.5.1 Data Flow Diagram (DFD):	24
4.5.1.1 Level 0 (Context level DFD):	25
4.5.1.2 Level 1 DFD:.....	25

4.5.2 Activity Diagram-	27
4.5.3 State-chart Diagram	29
CHAPTER 5: SYSTEM DESIGN.....	
5.1 Test cases.....	30
CHAPTER 6: CODING AND IMPLEMENTATION	
6.1 Implementing Approaches and testing.....	33
6.2 Code efficiency.....	33
6.3 Testing approach.....	43
6.3.1 Unit Testing.....	44
6.3.2 Integration Testing.....	44
6.3.3 Beta Testing.....	44
CHAPTER 7: RESULT AND DISCUSSION.....	
7.1 Test Report	46
7.2 User Documentation.....	46
CHAPTER 8: CONCLUSION	
8.1 Conclusion	50
8.2 Limitation.....	50
8.3 Future Scope	50
References	
Book Reference	52
Website Reference	52
Referred for uml diagram.....	52

List of Figures

No.	Diagram name	Page number
1.	Architecture	2
1.1	Arduino Uno	5
1.2	Ultrasonic sensor	5
1.3	Soil moisture sensor	6
1.4	Resistors	6
1.5	Light Dependent Resistor	7
1.6	Buzzers	7
1.7	GPS NEO 6M	8
1.8	ESP8266 NodeMCU	9
1.9	Jumper wires	9
1.10	Circuit diagram of Arduino uno	10
1.11	Circuit diagram of wifi module	11
4.1	Google forms	15
4.2	Requirement gathering	17
4.3	Gantt chart	20
4.4	Gantt chart 2	23
4.5	Level 0 DFD	25
4.6	Level 1 DFD	25
4.7	Systems Activity diagram	27
4.8	State chart diagram obstacle detection	29
4.9	State chart diagram water detection	29
4.10	State chart diagram Insufficient light detection	29

4.11	State chart diagram location	30
7.1	Obstacle detection	46
7.2	Water detection	46
7.3	Map link Through text	47
7.4	Map location through text	48
7.5	Live location through map	49

List of Tables

Table. No.	Table Name	Page No.
4.1	Task Table	19
4.2	Task Table 2	22
4.3	DFD diagram Notations	24
4.4	Activity diagram Notations	26
4.5	State diagram Notations	28
5.1	Test case	31
6.1	Test case approach	45

- **Title:**

SMART BLIND STICK

- **Statement:**

The project will basically detect the obstacles in front of them,

water also & insufficient light. It has feature of GPS tracking which is used for the persons caretaker or parents to track location.

- **Why this project?**

In our country there are many blind person and they typically use the white colour stick to navigate and detect obstacle with the help of physical touch to that obstacle. But this device is useful for them to detect the obstacle with the specific range and the sensors which are used in this system i.e ultrasonic sensors, soil moisture sensor, light dependent resistor sensor will help to detect if water is ahead and insufficient light and main feature is GPS tracking which is used to track the location of that visually impaired person and from the safety point of view also .so this will be the modern type of stick which is more useful or more advanced than the earlier one.

- **Objective or scope of object:**

Smart blind stick is specially designed to detect obstacles whether its solid or water in front of visually impaired person which may help to navigate freely like sighted person and from the safety point of view of GPS tracking used.

Scope:

From this project visually impaired person can walk freely like sighted person. And the persons parents or caretaker can track the location so they will also don't have to worry about that person with the help of SMS to authenticated device.

- **Methodology for developing project:**

EDLC Model and SDLC Model

- **SDLC Model**

In this system I am going to use iterative model.

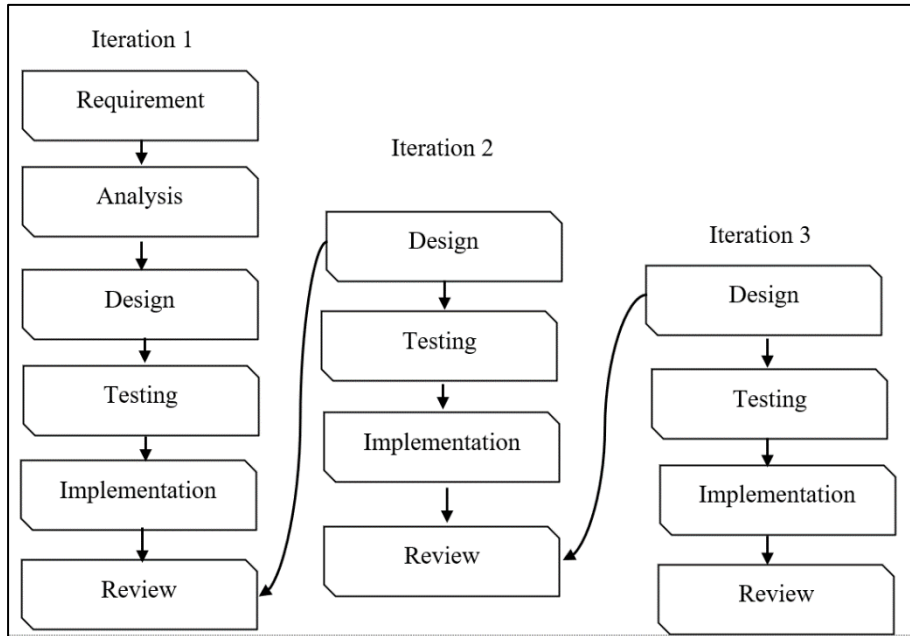


Fig 1 Architecture

- **Proposed architecture of project:**

1. Standalone project

- **Requirements:**

1. **Software requirements:**

Operating system: Windows 7,8,10 or Mac OS, Arduino IDE for coding, Arduino software, Blynk app.

2. **Hardware requirements:**

Arduino uno, ultrasonic sensor, LDR, Soil moisture sensor, resistor, breadboard, GPS module, Node MCU Esp8266, power supply.

- **Platform required:**

Arduino IDE

- **Contribution of society:**

With this project the visually impaired person can walk with the confidence as sighted person and main feature is the GPS location through Blynk app.

1.1 Need:

In our country the people who are visually impaired have the the blind stick which can detect obstacle by only physical touch, but this system which I am developing through sensors it can detect obstacle, water, & insufficient light and main feature is live location through Blynk app for the visually impaired persons parents or caretaker so that they can walk freely like sighted person .

1.2 Conceptualization/concept scope:

From this project visually impaired person can walk freely like sighted person and person's parents or caretaker can track the live location so they don't have to worry about that person. It is all possible through sensors.

1.3 Targeted end user:

Visually impaired person

1.4 Analysis:

1.4.1 Functional requirements:

- 1) The system should detect the obstacles in front of them with the help of sensors
- 2) The system should track the live location of person with the help of Blynk app.
- 3) While tracking the location, systems Wi-Fi or network should be on.
- 4) While doing this things system should have the stable power supply.

1.4.2 Non-functional requirements:

- 1) The system should be secured
- 2) It should be reliable and accurate also easy to use.
- 3) It should be easy to maintain.

1.5 Circuit diagram:

1.5.1 Arduino Uno

The Arduino UNO is an open-source microcontroller board based on the MicrochipATmega328P microcontroller and developed by Arduino cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 Digital pins, 6 Analog pins, and programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform. The ATmega328 on the Arduino Uno comes preprogrammed with a boot loader that allows uploading new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol. The Uno also differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

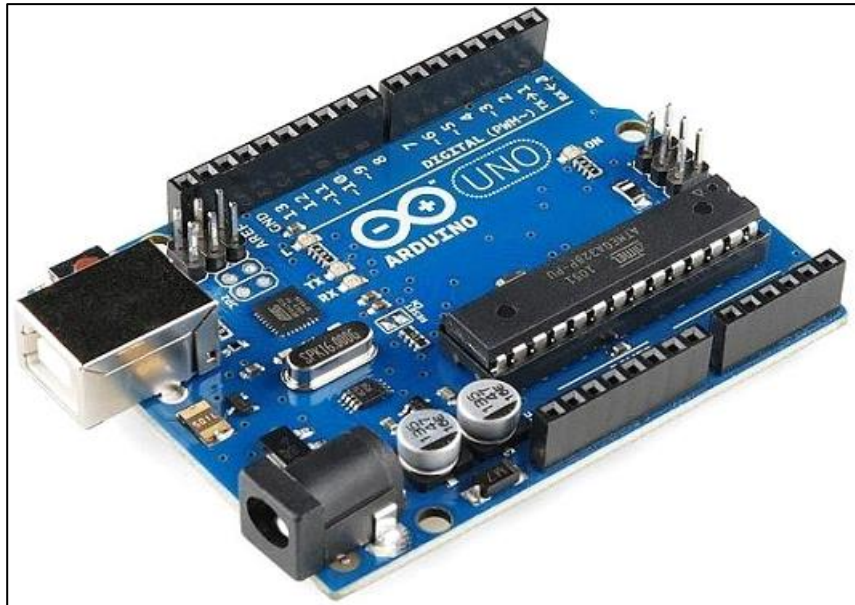


Fig 1.1 Arduino Uno

1.5.2 Ultrasonic Sensor

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound. Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).



Fig 1.2 Ultrasonic sensor

1.5.3 Soil Moisture Sensor

Soil moisture sensors measure or estimate the amount of water in the soil. These sensors can be stationary or portables such as handheld probes. Stationary sensors are placed at the predetermined locations and depths in the field, whereas portable soil moisture probes can measure soil moisture at several locations.

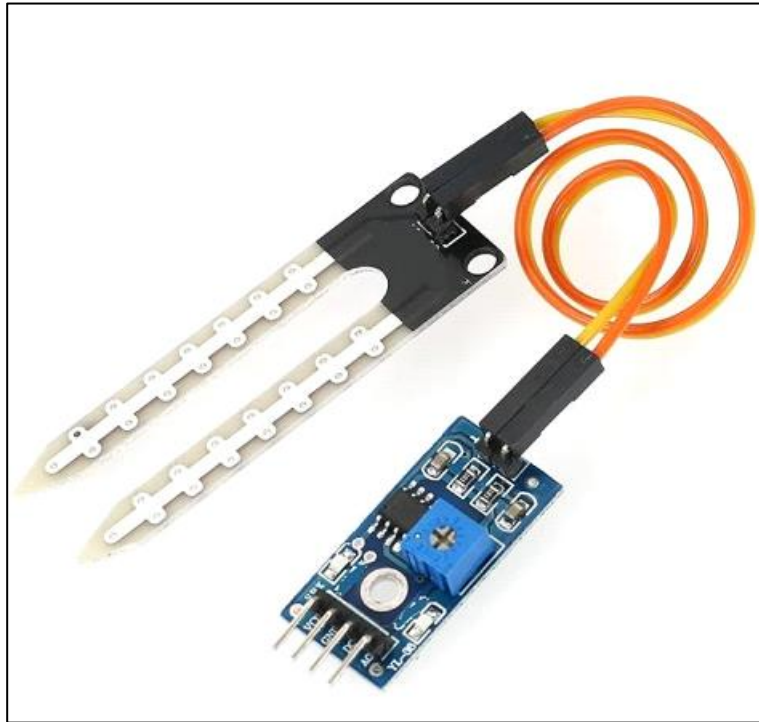


Fig 1.3 Soil moisture sensor

1.5.4 Resistors

It is a device having resistance to the passage of an electric current. A resistor is an electrical component that limits or regulates the flow of electrical current in an electronic circuit.



Fig 1.4 Resistors

1.5.5 LDR (Light Dependent Resistor)

An LDR is a component that has a (variable) resistance that changes with the light intensity that falls upon it. This allows them to be used in light sensing circuits. A photoresistor is a light-controlled variable resistor. The resistance of a photoresistor decreases with increasing incident light intensity; in other words, it exhibits photoconductivity. A photoresistor can be applied in light-sensitive detector circuits, and light-activated and darkactivated switching circuits. Working Principle of LDR.

This resistor works on the principle of photo conductivity. It is nothing but, when the light falls on its surface, then the material conductivity reduces and also the electrons in the valence band of the device are excited to the conduction band.

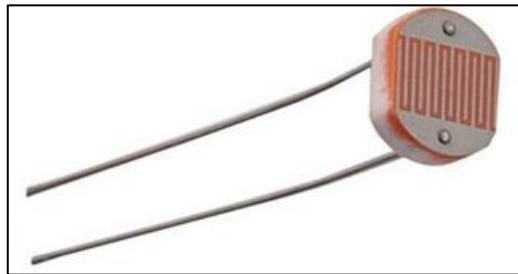


Fig 1.5 Light Dependent Resistor

1.5.6 Buzzers

A buzzer or beeper is an audio signaling device,[which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.



Fig 1.6 Buzzers

1.5.7 GPS NEO 6M

GPS stands for Global Positioning System by which anyone can always obtain the position information anywhere in the world. GPS receiver module gives output in standard (National Marine Electronics Association) NMEA string format. It provides output serially on Tx pin with default 9600 Baud rate. This NMEA string output from GPS receiver contains different parameters separated by commas like longitude, latitude, altitude, time etc. Each string starts with '\$' and ends with carriage return/line feed sequence.



Fig 1.7 GPS NEO 6M

1.5.8 ESP8266 NodeMCU

The NodeMCU (*Node MicroControllerUnit*) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WiFi), and even a modern operating system and SDK.

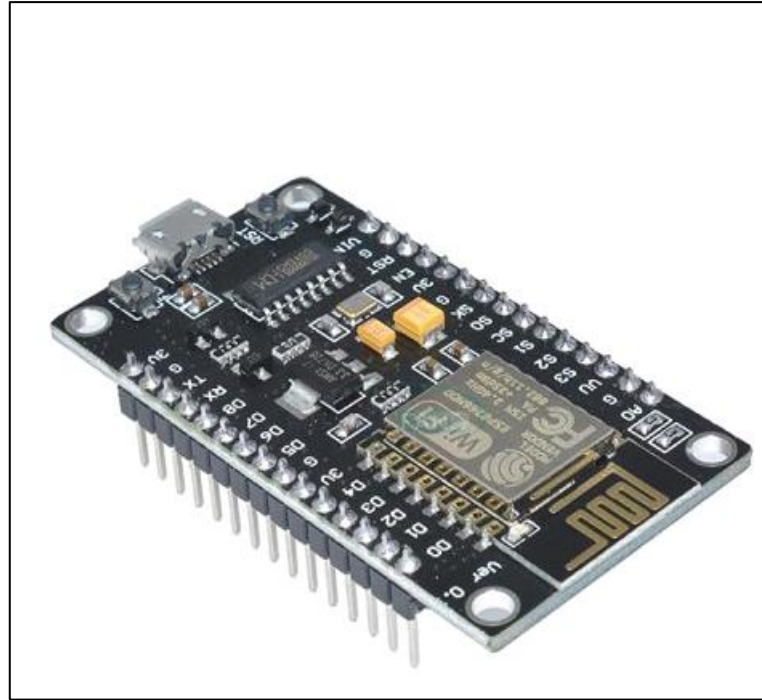


Fig 1.8 ESP8266 NodeMCU

1.5.9 Jumper wires

A wire is a single, usually cylindrical, flexible strand or rod of metal. Wires are used to bear mechanical loads or electricity and telecommunications signals. Wire is commonly formed by drawing the metal through a hole in a die or draw plate. Here male to male wires are also used. Wire gauges come in various standard sizes, as expressed in terms of a gauge number. The term wire is also used more loosely to refer to a bundle of such strands, as in "multi stranded wire", which is more correctly termed a wire rope in mechanics, or a cable in electricity.

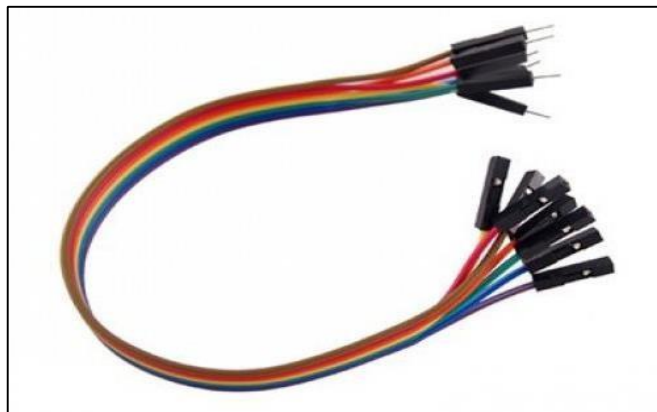


Fig 1.9 Jumper wires

1.5.10 Sim800l

SIM800L GSM/GPRS module is a miniature GSM modem, which can be integrated into a great number of IoT projects. You can use this module to accomplish almost anything a normal cell phone can; SMS text messages, make or receive phone calls,

connecting to the internet through GPRS, TCP/IP, and more! To top it off, the module supports quad-band GSM/GPRS network, meaning it works pretty much anywhere in the world

First is made of wire (which solders directly to NET pin on PCB) – very useful in narrow places. Second – PCB antenna – with double-sided tape and attached pigtail cable with IPX connector. This one has better performance and allows to put your module inside a metal case – as long the antenna is outside.

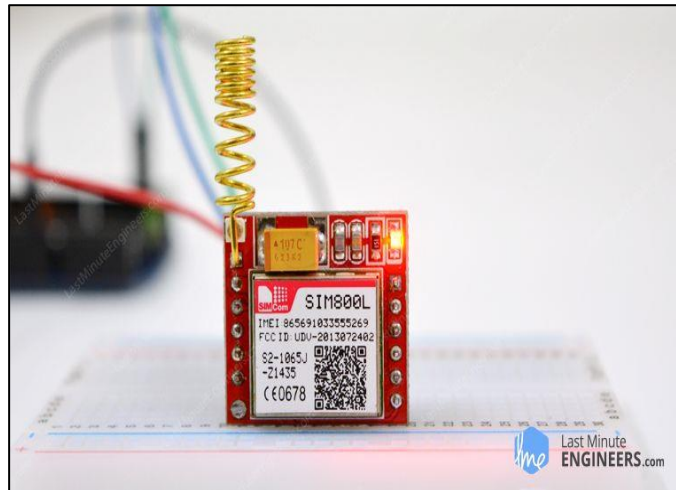


Fig 1.10 Sim800l

Circuit diagram of Arduino uno:

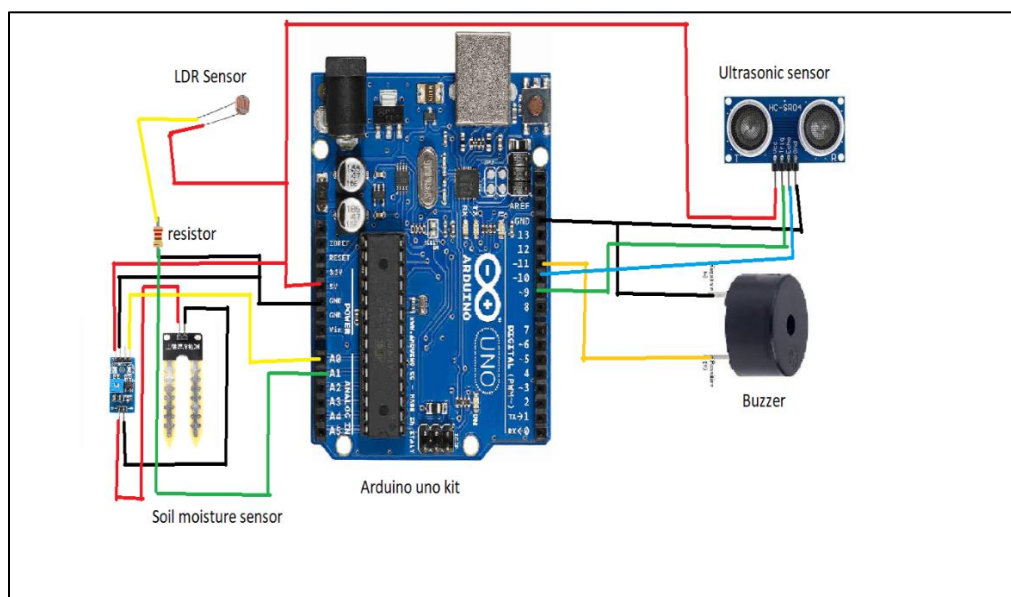
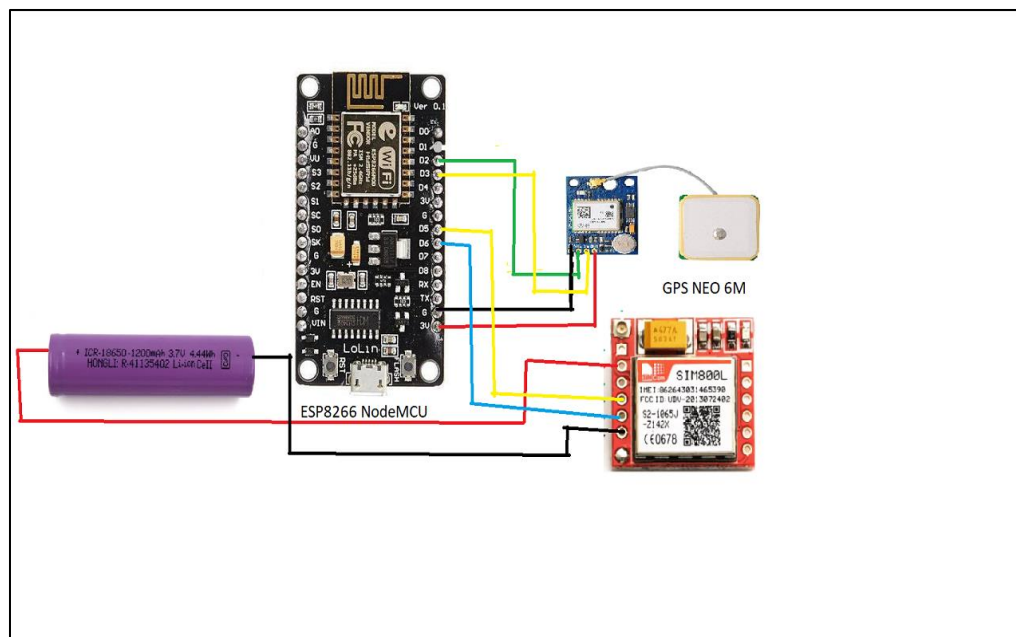


Fig 1.11 Circuit diagram of Arduino uno

In this diagram the main Arduino uno is connected to ultrasonic sensor, soil moisture sensor, ldr and buzzer. The positive side of the buzzer is connected to pin 11 and negative side is connected to ground, when obstacle or water is detected it beeps sound. The ultrasonic sensors trig pin is connected to 9 and echo pin is connected to 10 and vcc is connected to 5V and ground is connected to ground. In soil moisture sensor pins are connected to power and A0 respectively.

Circuit diagram of wifi module:



1.12 circuit diagram of wifi module

The ESP8266 NodeMCU model is connected to sim800l and GPS NEO6M and a 3.7v cell is attached to sim800l as it supports power from 3.7v to 4.2v. GPS NEO6M rx is connected to D2 and tx is connected to D3 and vcc is connected to power and ground is connected to wifimodule's ground. And Sim800l vcc is connected to 3.7v + side and ground is connected to 3.7v cell - side. Rx pin is connected to D5 and Tx pin is connected to D6.

2.1 Background:

The previous blind stick which is in use is just a simple and foldable stick which can detect the obstacle by physical touch only and some modern blind sticks made from IOT can detect obstacle in front from sensor. The smart blind stick can detect obstacle from front other than that water , and insufficient light with the help of sensors and main feature is live GPS location to the person who is authentic (parents or caretaker).

2.2 Objective:

This system is specially designed to detect obstacles whether its solid, water in front of visually impaired person which may help to navigate freely like sighted person and from the safety concern there is GPS tracking is used.

2.3 Purpose ,Scope, Applicability:

2.3.1 Purpose:

In our country the people who are visually impaired have the blind stick which can detect obstacle by only physical touch. But in this system there are sensors which can detect obstacle, water and insufficient light and main feature is the live location through GPS, people can view like sighted person through this system.

2.3.2 Scope:

From this project visually impaired person can walk freely like sighted person and persons parents or caretaker can track the live location so they don't have to worry about that person .

2.3.3 Applicability:

By implementing this project the visually impaired person can walk freely like sighted person. The main use of this project is for the visually impaired person in our society.

2.4 Achievements :

In Smart Blind Stick the input stage is consist of 3 sensors; i.e. Ultrasonic sensors, Soil Moisture sensor and LDR sensor which senses the obstacle, water and light respectively. A C Program is loaded into microcontroller. The live location is shared so it will be easier for caretaker to track the live location of the person.

Various Technologies Studied :

- **Arduino** : Arduino is the leading company on the iot market that produces electronic devices and software for them. They are having software products and represented by Arduino ide, Arduino cloud, iot cloud remote and coding is done in c.
- **Flutter**: Another hardware product for iot solution is flutter a programmable processor core. It's based on Arduino. It has complete kit , solar panel ,3D printed parts for device parts. For device dart language is used.
- **Raspberry pi**: It's formerly known as Raspbian .It's official operating system for raspberry pi hardware and programming are done in python.
- **Kinoma**: It's open source software and hardware product for iot and embedded solutions. kinoma create, kinomastudio(IDE), kinoma connect(for android ios) are products.
- **Node-red**: It's a free programming tool based on node.js .It works primarily in linux environment but can be installed on android and windows.as well. You will need linux subsystem for windows.
- **Eclipse Iot_**: A wide range of open source project for iot development is gathered under eclipse umbrella. They include software development platforms framework ,tools and many more. It uses java language.

Technology I am going to use for this project is :Arduino

Reason for selecting this technology:

The Arduino software is easy to use for beginners and it runs on mac, windows, and linux. The main reason for selecting this technology is it's inexpensive, cross platform support ,simple clear programming environment , open source and extensible hardware.

4.1 Problem definition:

Smart blind stick is specially designed to detect obstacles whether it's solid or water in front and also insufficient light which help visually impaired person to navigate freely like sighted person. And for the safety point of view of that person GPS tracking system is also used with the help of app.

4.1.1 Problem Description:

This stick will help the blind person to walk freely like sighted person, as it detect obstacles, water, insufficient light with the help of sensors and main feature of it is that it can track the live location of person also.

4.1.2 sub problems:

- Obstacle detection
 - Water detection
 - Insufficient light detection
 - GPS tracking
-
- Obstacle detection : In this with the help of ultrasonic sensor obstacle detection is done with the help of this we can detect the obstacle in front by 1 to 25 feet of range before.
 - Water detection: If the water is present in front of the person it will beep, for that soil moisture sensor is used.
 - Insufficient light detection: If there is absence of light particular room or on that specific road or lane also. It will beep so that person will not go in that area. It is done using light dependent resistor(LDR) sensor.
 - GPS tracking : For the safety point of view the feature of GPS tracking is there it can give the current location of that person to the authorized device with the help of app.

4.2 Requirement specification:

For my system I have used google forms to collect feedback from users and according to the responses I have added the functional , non-functional and system requirements for the system.

4.2.1 Requirement Gathering:

The link of the google form that I provided to the users is given below:

<https://docs.google.com/forms/d/1YxGAKoBTpvLbhDwQ49kYo5k1PchBtLSVXWYTSUfATIs/edit>

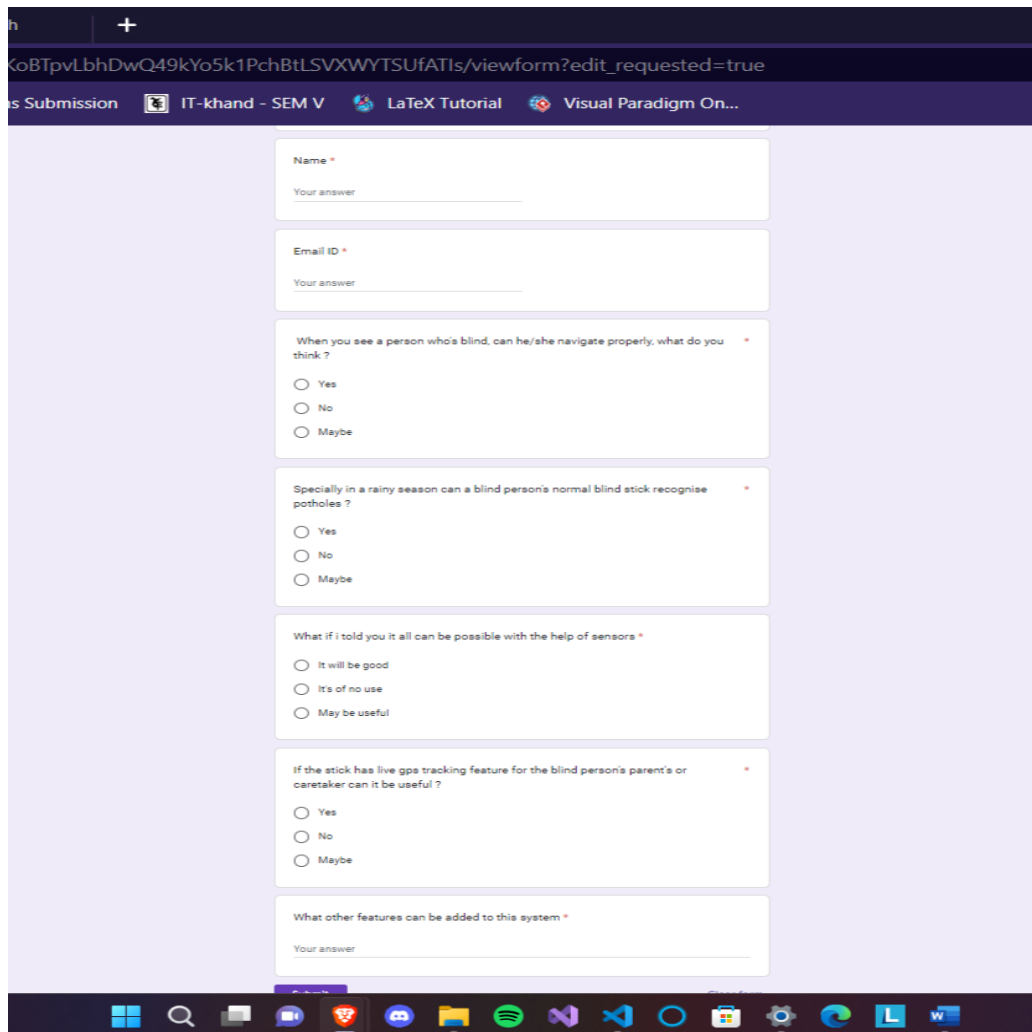
A screenshot of a Google Form titled "IT-khand - SEM V". The form contains several questions with radio button options. The questions are: 1. "When you see a person who's blind, can he/she navigate properly, what do you think?" with options Yes, No, and Maybe. 2. "Specially in a rainy season can a blind person's normal blind stick recognise potholes?" with options Yes, No, and Maybe. 3. "What if i told you it all can be possible with the help of sensors?" with options It will be good, It's of no use, and May be useful. 4. "If the stick has live gps tracking feature for the blind person's parent's or caretaker can it be useful?" with options Yes, No, and Maybe. 5. "What other features can be added to this system?" with a text input field. The form is displayed in a web browser window with a Windows taskbar at the bottom.

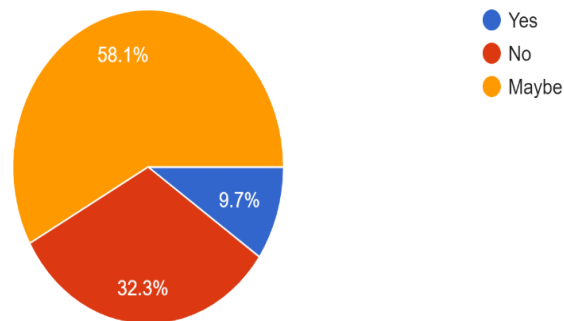
Fig 4.1 google form

Based on the responses I got from the users, I may conclude that:

According to the response 58.1% people think that blind people may navigate properly. Almost 61.3% people think that blind people can't recognize potholes during the rainy days. 90.3% people think that it will be good if it is done with the help of sensors. 96.8% people think that the live GPS tracking feature can be helpful.

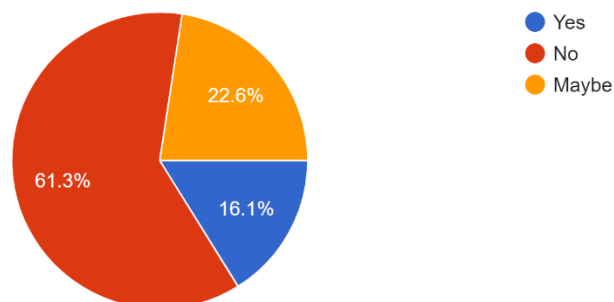
When you see a person who's blind, can he/she navigate properly, what do you think ?

31 responses



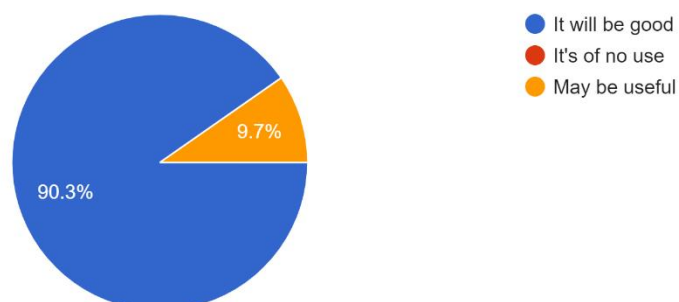
Specially in a rainy season can a blind person's normal blind stick recognise potholes ?

31 responses



What if i told you it all can be possible with the help of sensors

31 responses



If the stick has live gps tracking feature for the blind person's parent's or caretaker can it be useful ?

31 responses

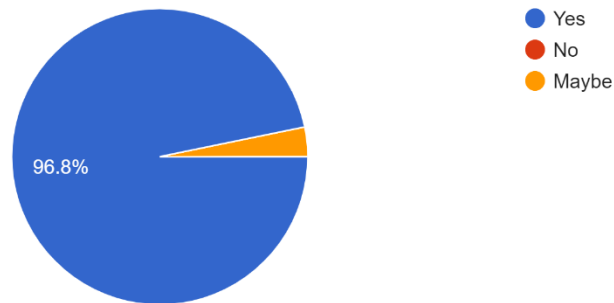


Fig 4.2 Requirement gathering

4.2.2 Requirement Analysis:

4.2.2.1 Functional Requirements:

The system should detect the obstacles in front with the help of sensors. The system should track the live location of the person with the help of Blynk app. While tracking the location, systems wifi network should be on.

While doing these things, the system should have power supply.

4.2.2.2 Non Functional Requirements:

The system should be secured. It should be reliable and accurate and easy to use. It should be easy to maintain. Sensors should detect the respected obstacle correctly in to the person.

4.2.2.3 System Requirement:

- **Obstacle detection:**
 - **Input:** Any object which is physically present.
 - **Source:** sensors
 - **Output:** beep sound will be generated.
 - **Destination:** user will get alerted through beep sound.
 - **Action:** After detection user will get alerted.
 - **Pre-condition:** user must have to walk.

- **Post-condition:**User should skip or go by another way.

- **Water detection:**
 - **Input:** water is present ahead.
 - **Source:** sensors
 - **Output:** beep sound will be generated.
 - **Destination:** user will get alerted through beep sound.
 - **Action:** After detection user will get alerted.
 - **Pre-condition:** user must have to walk.
 - **Post-condition:**User should skip or go by another way.

- **Insufficient light detection :**
 - **Input:**if insufficient light is there
 - **Source:** sensors
 - **Output:** beep sound will be generated.
 - **Destination:** user will get alerted through beep sound.
 - **Action:** After detection user will get alerted.
 - **Pre-condition:** -
 - **Post-condition:**User should skip or go by another way.

- **GPS Tracking:**
 - **Input:**From GPS module.
 - **Source:** Blynk app
 - **Output:** show current location on Blynk app.
 - **Destination:** users caretaker or parent will get the current location of that person.
 - **Action:** parents get live location after of the person after tracking.
 - **Pre-condition:** Network connection should be on.
 - **Post-condition:-**

- **Sound beep while detecting :**
 - **Input:**sensors.
 - **Source:** obstacle detection.
 - **Output:** it will beep.
 - **Destination:** with the help of sensors obstacle detected.
 - **Action:** -

- **Pre-condition:** Network connection and power supply should be on.
- **Post-condition:-**

4.3 Planning and scheduling

4.3.1 Activity table:

Task	Start date	End date
Synopsis	01/06/22	20/06/22
EDLC	20/06/22	27/06/22
Introduction	20/06/22	27/06/22
Survey of Technologies	20/06/22	27/06/22
Requirements specification and Analysis	27/06/22	04/07/22
System Requirements	11/07/22	18/07/22
Planning and scheduling	18/07/22	25/07/22
Data flow diagram	18/08/22	25/08/22
Circuit diagram	25/08/22	02/09/2002
Activity diagram	12/09/22	19/09/22
State diagram	12/09/2022	19/09/2022
Test cases	19-09-2022	26-09-2022

Table 4.1 Activity table

4.3.2 Gantt chart:

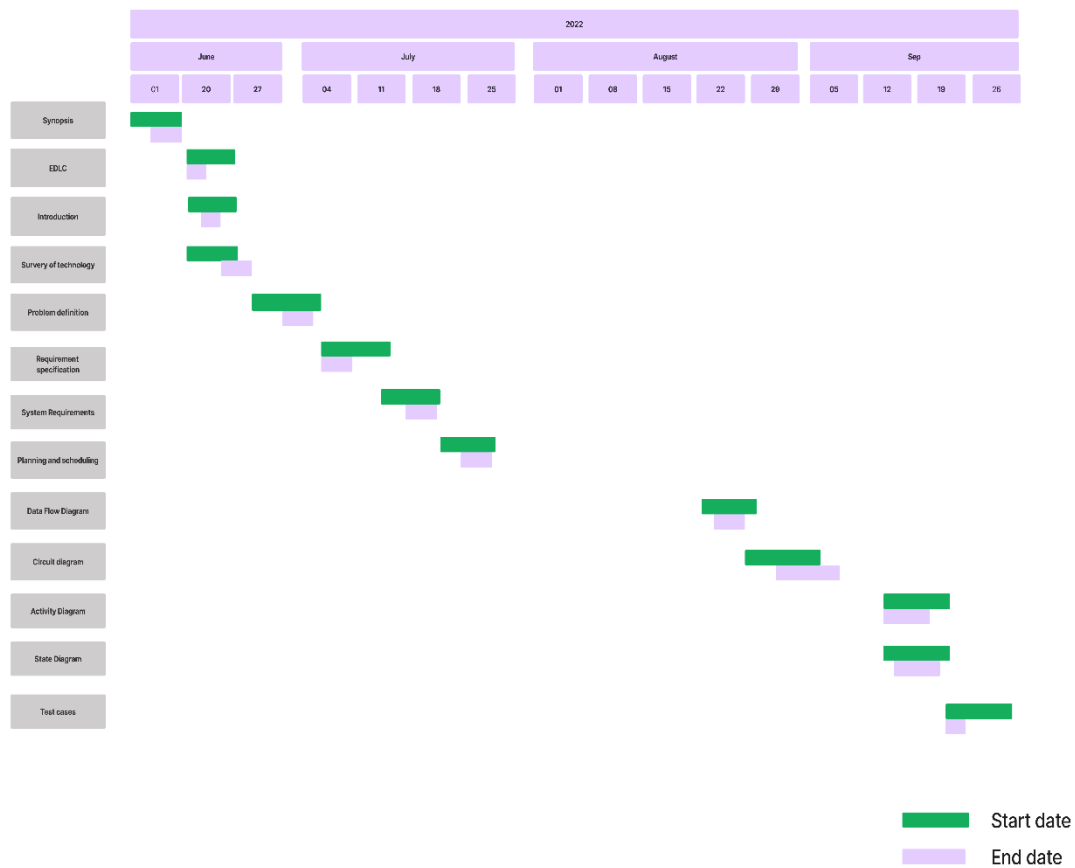


Fig 4.3 Gantt chart

Re-engineered Task management table:

Task No.	Task Name	Start Date	End Date	Estimated Days	Actual Start Date	Actual End Date	Actual Days
Re-Engineering							
T1	RE-chapter 1:Introduction	18-Dec-22	19-Dec-22	2	18-Dec-22	19-Dec-22	2
T2	RE-chapter 2:Survey of Technologies	20-Dec-22	21-Dec-22	2	20-Dec-22	21-Dec-22	2
T3	RE-chapter 3: Requirement of Analysis	22-Dec-22	23-Dec-22	2	22-Dec-22	23-Dec-22	2
T4	RE-chapter 4: System Design	23-Dec-22	23-Dec-22	1	23-Dec-22	23-Dec-22	1
Unit 1 : Ultrasonic sensor							
T5	Coding	26-Dec-22	29-Dec-22	4	26-Dec-22	30-Dec-22	5
T6	Testing	30-Dec-22	31-Dec-22	2	30-Dec-22	31-Dec-22	2
T7	Debugging	2-Jan-23	4-Jan-23	3	2-Jan-23	4-Jan-23	3
T8	Unit Testing	5-Jan-23	6-Jan-23	2	5-Jan-23	6-Jan-23	2
Unit 2 : Soil Moisture Sensor							
T9	Coding	8-Jan-23	11-Jan-23	4	8-Jan-23	11-Jan-23	4
T10	Testing	12-Jan-23	13-Dec-23	2	12-Jan-23	13-Dec-23	2
T11	Debugging	14-Jan-23	15-Jan-23	2	14-Jan-23	15-Jan-23	2
T12	Unit Testing	16-Jan-23	17-Jan-23	2	16-Jan-23	17-Jan-23	2
T13	Integration Testing	20-Jan-23	22-Jan-23	3	20-Jan-23	24-Jan-23	5
Unit 3 : Light Detector Resistor							
T14	Coding	8-Jan-23	11-Jan-23	4	8-Jan-23	11-Jan-23	4
T15	Testing	12-Jan-23	13-Dec-23	2	12-Jan-23	13-Dec-23	2
T16	Debugging	14-Jan-23	15-Jan-23	2	14-Jan-23	8Feb-23	24
T17	Unit Testing	16-Jan-23	17-Jan-23	2	16-Jan-23	8-Feb-23	22
Unit 4: Neo6m GPS tracking with app							

T18	Coding	28-Jan-23	31-Feb-23	4	28-Jan-23	2-Feb-23	6
T19	Testing	4-Feb-23	6-Feb-23	3	4-Feb-23	6-Feb-23	3
T20	Debugging	7-Feb-23	8-Feb-23	2	7-Feb-23	8-Feb-23	2
T21	Unit Testing	9-Feb-23	10-Feb-23	2	9-Feb-23	10-Feb-23	2
Unit 5: GPS tracking with Sim8001 module							
T22	Coding	11-Feb-23	14-Feb-23	4	11-Feb-23	15-Feb-23	5
T23	Testing	16-Feb-23	18-Feb-23	3	16-Feb-23	18-Feb-23	3
T24	Debugging	19-Feb-23	20-Feb-23	2	19-Feb-23	20-Feb-23	2
T25	Unit Testing	21-Feb-23	22-Feb-23	2	21-Feb-23	22-Feb-23	2
T26	Integration Testing	23-Feb-23	24-Feb-23	2	23-Feb-23	24-Feb-23	2
T27	System Testing	24-Feb-23	25-Feb-23	2	24-Feb-23	25-Feb-23	2
Further Chapters							
T28	Chapter 5: Implementation and Testing	25-Feb-23	26-Feb-23	2	25-Feb-23	26-Feb-23	2
T29	Chapter 6: Results and Discussions	27-Feb-23	27-Feb-23	1	27-Feb-23	27-Feb-23	1
T30	Chapter 7: conclusions	27-Feb-23	27-Feb-23	1	27-Feb-23	27-Feb-23	1

Table 4.2 Activity table2

Gantt chart:

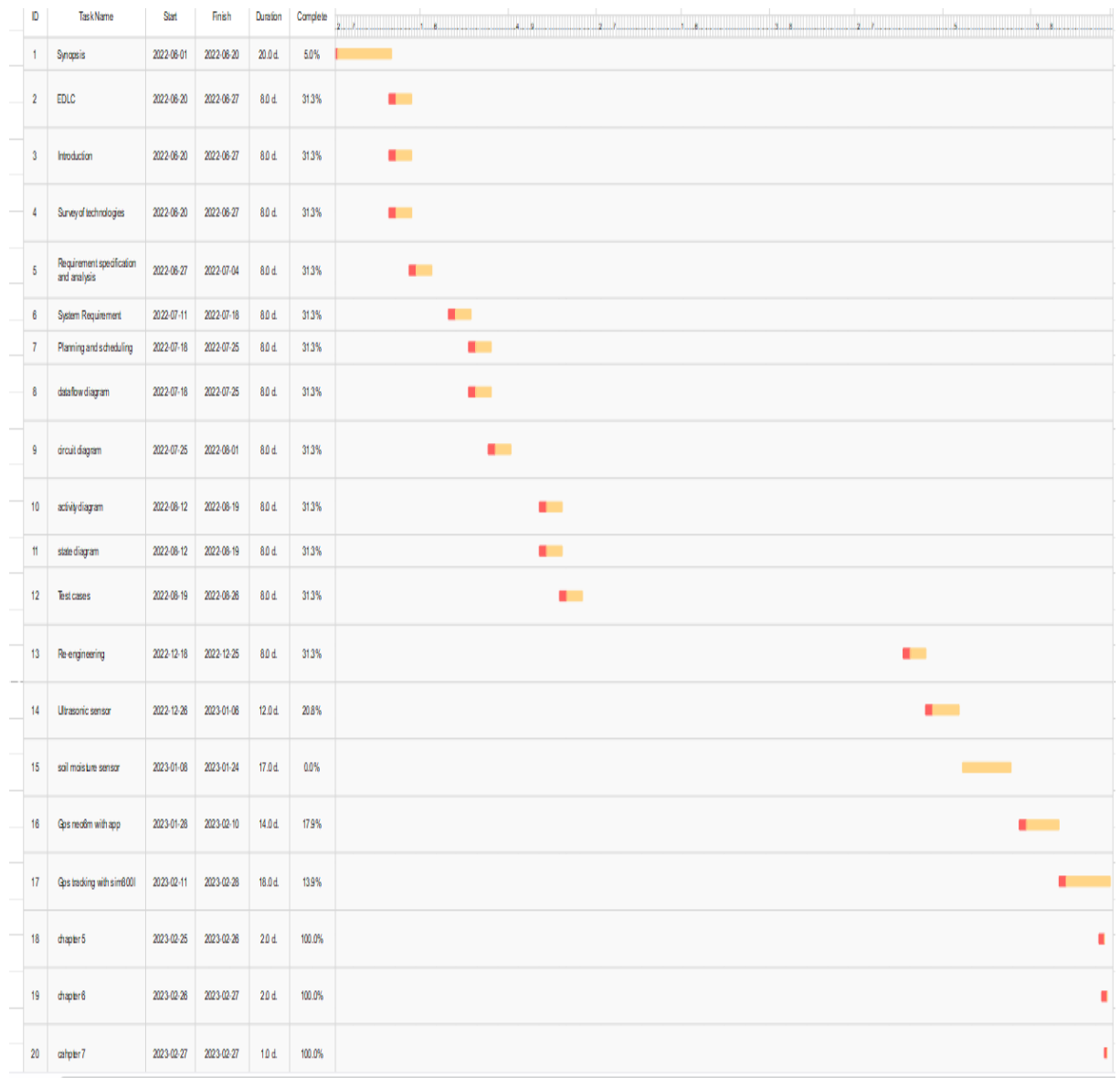


Fig 4.4 Gantt chart2

4.4 Software and Hardware Requirements:

- **Software requirements:**

Operating system: Windows 7,8,10 or more Mac os Blynk app.

Arduino IDE for coding,

For compiling and testing :Arduino software

- **Hardware requirements:**

Arduino uno, ultrasonic sensor, soil moisture sensor, LDR, Resistor,

Breadboard, Gps module, Node MCU ESP8266 , Power supply.

4.5 Conceptual Model:

4.5.1 Data Flow Diagram:

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

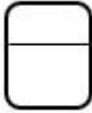

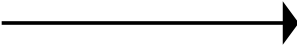

Name	Symbol	Description
Process		A process transforms incoming data flow into outgoing data flow.
Data Store		Data stores are repositories of data in the system.
Data Flow		Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.
External Entity		External entities are objects outside the system, with which the system communicates

Table 4.3 DFD diagram Notations

DFD level 0 :

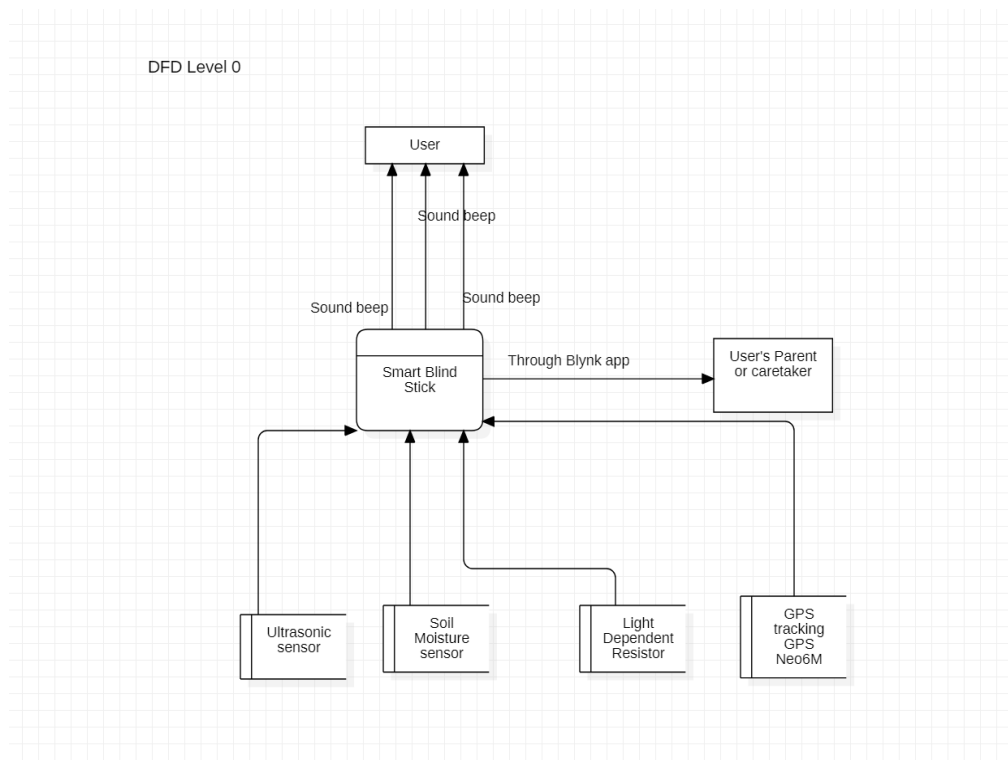


Fig 4.5 Level 0 DFD

DFD level 1:

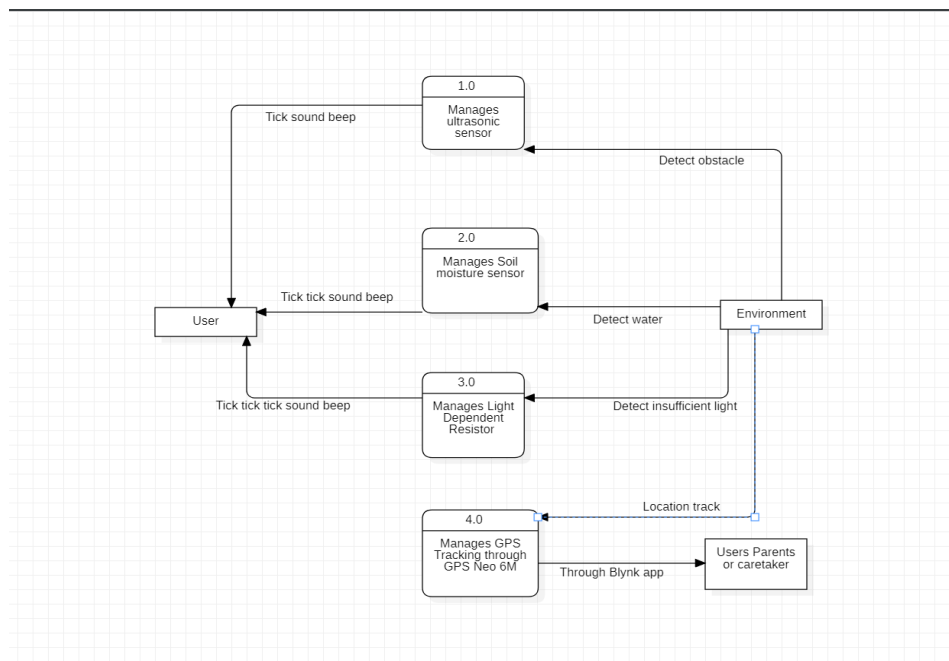


Fig 4.6 Level 1 DFD

4.5.2 Activity Diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.




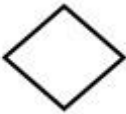
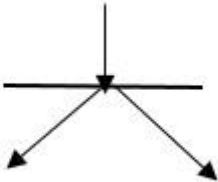

Name	Symbol	Description
Initial state		This shows the starting point or first activity of the flow.
Final state		The end of the Activity diagram, also called as a final activity.
Action		It represents the activity to be performed.
Decision		A logic where a decision is to be made is depicted by a diamond.
Synchronization		It combines or splits two activity flows.
Transition		A transition link represents control flow between nodes.

Table 4.4 Activity diagram notations

System:

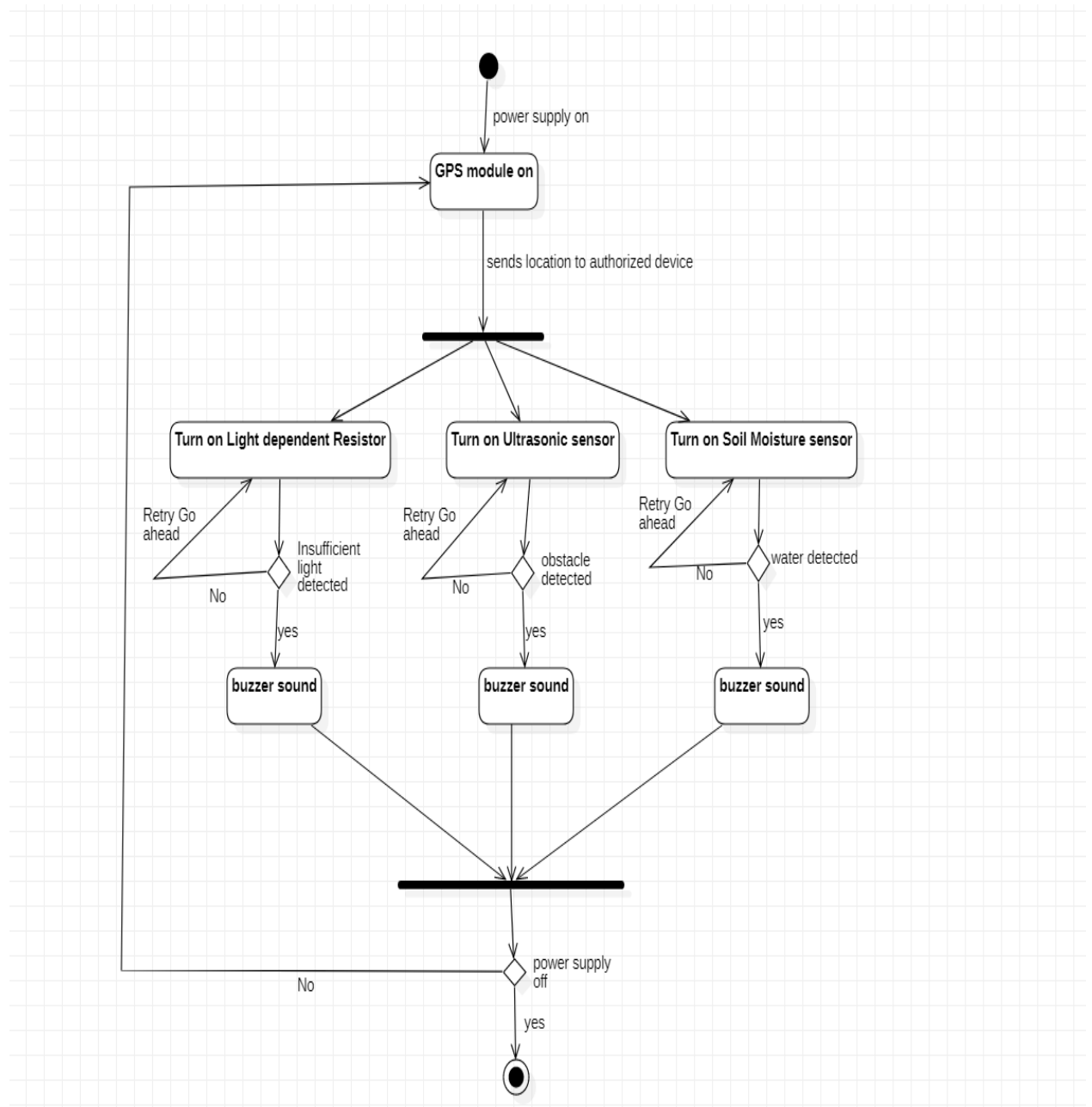


Fig 4.7 SystemActivity Diagram

4.5.3 State chart diagram:

A state diagram is used to represent the condition of the system or part of the system at finite instances of time. It's a behavioral diagram and it represents the behavior using finite state transitions. State diagrams are also referred to as State machines and State-chart Diagrams. These terms are often used interchangeably. So simply, a state diagram is used to model the dynamic behavior of a class in response to time and changing external stimuli.




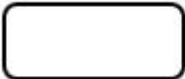
Name	Symbol	Description
Initial state		This represents the starting of the state diagram.
Final state		This represents the final state or end of the state diagram.
Transition		This represents the change of one state into another state.
State		This represents the state of the activity.

Table 4.5 State chart diagrams notations

Obstacle detection:

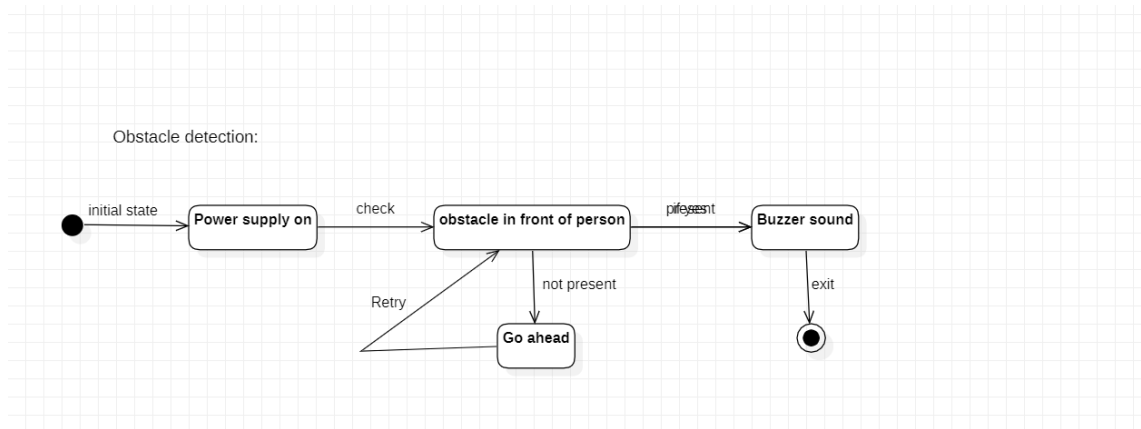


Fig 4.8 State chart diagram obstacle detection

Water detection:

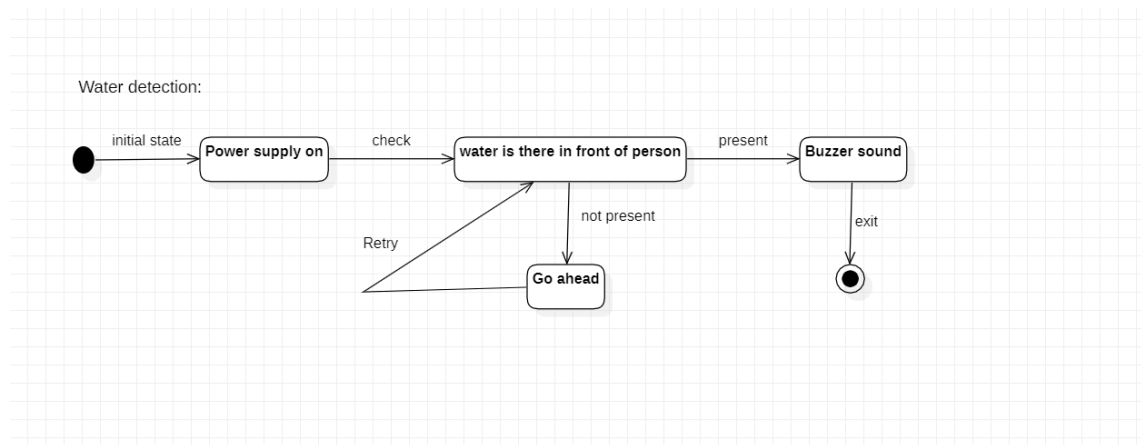


Fig 4.9 State chart diagram water detection

Insufficient light detection:

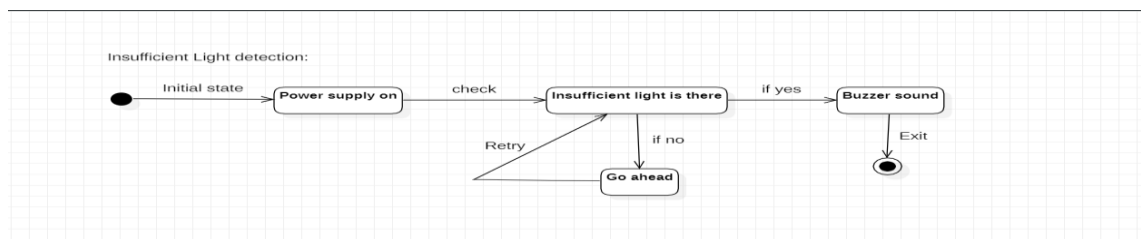


Fig 4.10 State chart diagram insufficient light detection

Location :

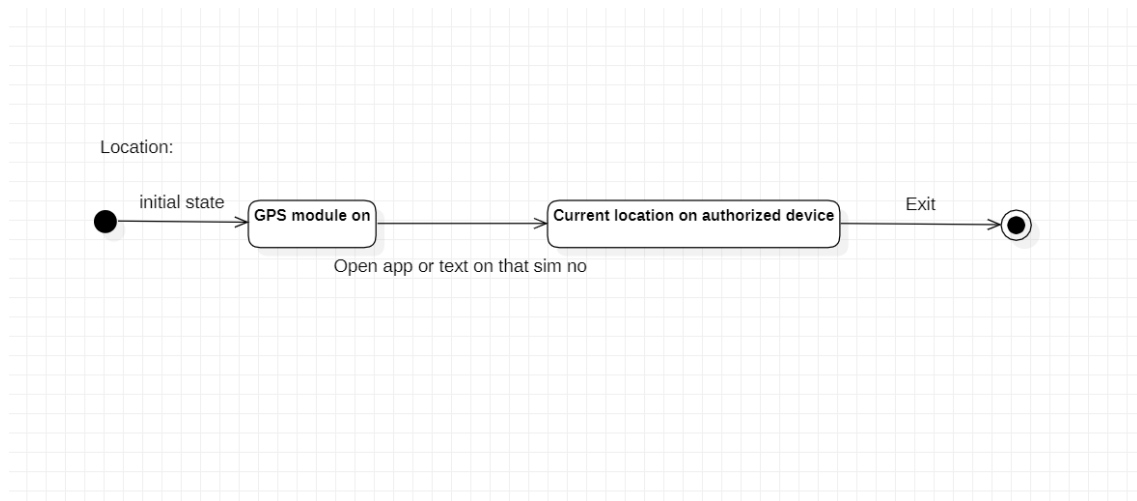


Fig 4.11 State chart diagram location

5.1 Test cases:

Test case no	Name	Expected output	Actual output	Remark
1	Obstacle detection	Beep sound need to be generated	Connection is not done properly	Fail
2	Water detection	Beep sound need to be generated	Beep sound detected	Pass
3	Light detector Resistor	Beep sound need to be generated	Absence of light is not catch	Fail
4	GPS Tracking	Need to show current location on app	Range is not caught	Fail
5	Sound beep while detecting	Need to beep at that time	Connection is not done properly	Pass
6	Gps tracking	Need to get location when texted on that sim	Sim is not registered from which text is send	Fail

Table 5.1 Test cases

Corrective actions taken for failed test case :

Test case 1: Obstacle Detection

Connection is checked and done properly after that it worked .

Test case 3: Light detector Resistor

Connection is checked and done properly but even though it not worked.

Test case 4 : GPS Tracking

For range go to open space where you can easily get range it includes road, terrace but sometimes not able to catch range in home.

Test case 6 : GPS Tracking through sim

For range go to open space where you can easily get range it includes road, terrace but sometimes not able to catch range in home and if number is not registered then it should not send message.

6.1 Implementation Approaches:

This project was implemented using the Iterative model. In incremental methodology, the model is designed, implemented and tested incrementally. If in case there is a change in the requirements of the user then that part can be redesigned, re-implemented and tested again iteratively. This model allows us to update the system at each increment and we can add new functionalities as well.

The project is implemented in Arduino IDE ,I first implemented the ultrasonic sensor and testing is done and after that I have implemented soil moisture sensor, after that both the sensors were tested combinely.

After that I implemented GPS tracking with the help of GPS NEO6M with the help of app so that live location can be seen. After that with the help of sim800l module you can text on the sim which is inserted in that module and testing of that module is also done. The system is implemented by considering all the problems into view and final project should be fulfilling all the requirements.

6.2 Coding and Efficiency:

6.2.1 coding

Code for Arduino:

```
const int trigPin = 9;
const int echoPin = 10;
const int buzzer = 11;
const int waterPin = A0;
const int lightPin = A1;
int value = 825;
long duration;
int distance;
int safetyDistance;
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  pinMode(waterPin,INPUT);
  pinMode(buzzer, OUTPUT);
```

```

Serial.begin(9600); // Starts the serial communication
}

void loop() {
//Ultrasonic Sensor
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance= duration*0.034/2;
safetyDistance = distance;
if (safetyDistance <= 17){
  Serial.println("Obstacle is detected");
  digitalWrite(buzzer, HIGH);
  delay(1000);
  digitalWrite(buzzer, LOW);
  delay(1000);
}
else{

}

//Soil Moisture Sensor
int senseValue = analogRead(waterPin);
//Serial.print(senseValue);
if(senseValue < value)
{
  Serial.println("Water is detected");
  digitalWrite(buzzer,HIGH);

```

```

    delay(1000);
    digitalWrite(buzzer,LOW);
    delay(100);
  }
  else
  {
    digitalWrite(buzzer,LOW);
  }

  //Light detector resistor

void setup() {
  pinMode(lightPin,INPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600); // Starts the serial communication
}

void loop() {
  int lightValue = analogRead(lightPin);
  Serial.println(lightValue);
  if(lightValue <5)
  {
    Serial.println("Absence of Light!!!");
    digitalWrite(buzzer,HIGH);

    //delay(1000);
    //digitalWrite(buzzer,LOW);
    //delay(100);
  }
  else
  {

```

```
digitalWrite(buzzer,LOW);  
}  
}
```

Code for ESP8266 NodeMCU:

```
#include <TinyGPS++.h>  
#include <SoftwareSerial.h>  
#include <ESP8266WiFi.h>  
#include <FirebaseESP8266.h>  
  
#define FIREBASE_HOST "https://esp8266demo1-34e55-default-rtdb.firebaseio.com"  
#define FIREBASE_AUTH  
"CjBjqvuWWyVNqxZn0TN4HBwezAlUuZCt4MNLs6CC"  
const String PHONE = "+918828388979";  
  
//GSM Module RX pin to NodeMCU D6  
//GSM Module TX pin to NodeMCU D5  
#define rxGSM D5  
#define txGSM D6  
SoftwareSerial sim800(rxGSM, txGSM);  
  
#define rxGPS D2  
#define txGPS D3  
SoftwareSerial neogps(rxGPS, txGPS);  
TinyGPSPlus gps;  
  
String smsStatus, senderNumber, receivedDate, msg;  
const char* ssid = "MI_29";  
const char* password = "atharva29";
```



```

float latitude , longitude;
FirebaseData firebaseData;
FirebaseJson json;
void setup()
{
    Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
    Firebase.reconnectWiFi(true);

    Serial.begin(115200);
    neogps.begin(9600);
    sim800.begin(9600);
    Serial.println("SIM800L serial initialize");
    Serial.println();
    Serial.print("Connecting to ");
    //Serial.println(ssid);
    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED)
    {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    sim800.listen();
    neogps.listen();
    smsStatus = "";
    senderNumber = "";
    receivedDate = "";

```

```

msg = "";
sim800.print("AT+CMGF=1\r"); //SMS text mode
delay(1000);

// firebase
Serial.println("Connecting Firebase.....");

//Serial.println("Firebase OK.");
}
void loop()
{
while (neogps.available() > 0)
if (gps.encode(neogps.read()))
{
if (gps.location.isValid())
{
latitude = gps.location.lat();
String lat_str = String(latitude , 6);
//Serial.println(lat_str);
longitude = gps.location.lng();
String lng_str = String(longitude , 6);
//Serial.println(lng_str);
if (Firebase.setFloat(firebaseData, "/GPS/f_latitude", latitude))
{ //Serial.println('Done lat');
//Serial.println("PATH: " + firebaseData.dataPath());
//Serial.println("TYPE: " + firebaseData.dataType());
//Serial.println("ETag: " + firebaseData.ETag());
}
else
{

```

```

        //Serial.println('Not Done lat');
    }

    //-----

    if (Firebase.setFloat(firebaseData, "/GPS/f_longitude", longitude))
    { //Serial.println('Done log');
        //Serial.println("PATH: " + firebaseData.dataPath());
        //Serial.println("TYPE: " + firebaseData.dataType());
        //Serial.println("ETag: " + firebaseData.ETag());
    }
    else
    { //Serial.println('Not Done log');
    }
}

while (sim800.available()) {
    parseData(sim800.readString());
}

while (Serial.available())
{
    //String s = Serial.readString();
    //Serial.println(s);
    sim800.println(Serial.readString());
}
}

//main loop ends

void parseData(String buff) {
    Serial.println(buff);
    unsigned int len, index;
    index = buff.indexOf("\r");

```

```

buff.remove(0, index + 2);
buff.trim();

if (buff != "OK") {
    index = buff.indexOf(":");
    String cmd = buff.substring(0, index);
    cmd.trim();
    buff.remove(0, index + 2);

    if (cmd == "+CMTI") {
        //get newly arrived memory location and store it in temp
        index = buff.indexOf(",");
        String temp = buff.substring(index + 1, buff.length());
        temp = "AT+CMGR=" + temp + "\r";
        //get the message stored at memory location "temp"
        sim800.println(temp);
    }
    else if (cmd == "+CMGR") {
        extractSms(buff);
        if (senderNumber == PHONE) {
            if (msg == "get location") {
                sendLocation();
            }
        }
    }
    else {
        Serial.println(String(senderNumber) + ": Phone not registered");
    }
}
else {

```

```

    //The result of AT Command is "OK"
}
}

void extractSms(String buff) {
    unsigned int index;
    index = buff.indexOf(",");
    smsStatus = buff.substring(1, index - 1);
    buff.remove(0, index + 2);
    senderNumber = buff.substring(0, 13);
    buff.remove(0, 19);
    receivedDate = buff.substring(0, 20);
    buff.remove(0, buff.indexOf("\r"));
    buff.trim();
    index = buff.indexOf("\n\r");
    buff = buff.substring(0, index);
    buff.trim();
    msg = buff;
    buff = "";
    msg.toLowerCase();
}

void sendLocation()
{
    // Can take up to 60 seconds
    boolean newData = false;
    for (unsigned long start = millis(); millis() - start < 2000;)
    {
        while (neogps.available())
        {

```

```

    if (gps.encode(neogps.read()))
    {
        newData = true;
    }
}

//If newData is true
if (newData)
{
    Serial.print("Latitude= ");
    Serial.print(gps.location.lat(), 6);
    Serial.print(" Longitude= ");
    Serial.println(gps.location.lng(), 6);
    newData = false;
    delay(300);

    sim800.print("AT+CMGF=1\r");
    delay(100);
    sim800.print("AT+CMGS=\"" + PHONE + "\"\r");
    delay(1000);
    sim800.print("http://maps.google.com/maps?q=loc:");
    sim800.print(gps.location.lat(), 6);
    sim800.print(",");
    sim800.print(gps.location.lng(), 6);
    delay(1000);
    sim800.write(0x1A); //ascii code for ctrl-26 //sim800.println((char)26); //ascii code
for ctrl-26
    delay(1000);
    Serial.println("GPS Location SMS Sent Successfully.");
}

```

```

}
else {
    Serial.println("Invalid GPS data");
}
}

```

6.2.2 Coding efficiency:

I have tried to keep the codes as short as possible but there is no compromise in functionalities and reliability. As the module used in this project for ESP8266 NodeMcu consist only 32kb of memory so while integrating the sim800l code and code for live location tracking through map the memory gets full and integration is not done. But as the live location through app is more convenient its been taken as it continuously shows live location through app.

6.3 Testing Approach:

One effective testing approach for project would be to conduct a thorough review of the document to ensure accuracy and completeness. This review could include checking that all necessary information is present, such as the name and address of the site, the types of vaccines available, and the hours of operation. Additionally, testing could involve simulated scenarios, such as a patient arriving for their appointment or a staff member needing to access important information quickly. This can help identify any potential issues with the documentation and ensure that it is user-friendly and easy to navigate. Regular updates and revisions to the documentation based on feedback from staff and patients can also help improve its effectiveness over time.

6.3.1 Unit Testing

A unit testing approach can be effective. This involves testing individual components or units of the documentation, such as the instructions for administering a vaccine or the procedures for recording vaccine doses. Each unit should be tested in isolation, with any dependencies or interactions with other units mocked or simulated.

Tests should be designed to ensure that each unit performs as expected and produces the correct outputs for various inputs or scenarios. This can include testing for accuracy, completeness, and readability of the documentation. Unit tests can be automated using testing frameworks, making it easier to run tests quickly and consistently.

Overall, a unit testing approach can help ensure that the documentation for a vaccination site is reliable, accurate, and easy to use for healthcare professionals administering vaccines.

6.3.2 Integration Testing:

Here focus is on design and construction of the software architecture. Integration Testing is a systematic technique for constructing the program structure as well as the model structure (Integrating the components) while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The goal here is to see if modules can be integrated properly, the emphasis being on testing interfaces between modules. This testing activity can be considered as testing the design and hence the emphasis on testing module interactions. In this project the main system is formed by integrating all the modules. When integrating all the modules I have checked whether the integration effects working of any of the services by giving different combinations of inputs with which the two services run perfectly before integration.

6.3.3. Beta Testing:

Beta testing can be considered pre-release testing. Beta testing is also sometimes referred to as user acceptance testing (UAT) or end user testing. In this phase of software development, applications are subjected to real world testing by the intended audience for the software. The experiences of the early users are forwarded back to the developers who make final changes before releasing the software commercially.

Test case no	Name	Expected output	Actual output	Remark
1	Obstacle detection	Beep sound need to be generated	Beep sound generated	Pass
2	Water detection	Beep sound need to be generated	Beep sound generated	Pass
3	Light detector Resistor	Beep sound need to be generated	Beep sound not generated	Fail
4	GPS Tracking	Need to show current location on app	Location shown on app	Pass

5	Sound beep while detecting	Need to beep at that time	Beep at that time	Pass
6	Gps tracking	Need to get location when texted on that sim	Location is reverted back when sms is send	Pass

Table 6.1 Test approaches

7.1 Test Report

The testing part in project development is a very important phase. This phase helps to know whether all the functionalities are being performed the way that they are supposed to be executing. The testing phase started with designing the test cases for each module as well as the designing process for integration test cases was performed. Each module was analyzed and according to that test cases were formed. The test cases include input and the expected output to be seen after entering the values. After designing the test cases, the test cases were checked by actually entering the inputs and checking from different users if they are getting the expected output or not. If the expected outputs match the actual output then the test cases were remarked to be successful or else they were remarked as fail. Not all values were tried and tested but the process made sure the system would be able to cope up with any values. After performing all the test cases, there were no more errors. So, no further modifications were needed in the respective modules.

7.2 User Documentation:

1. If obstacle is detected it will beep sound and user will get to know obstacle is ahead.

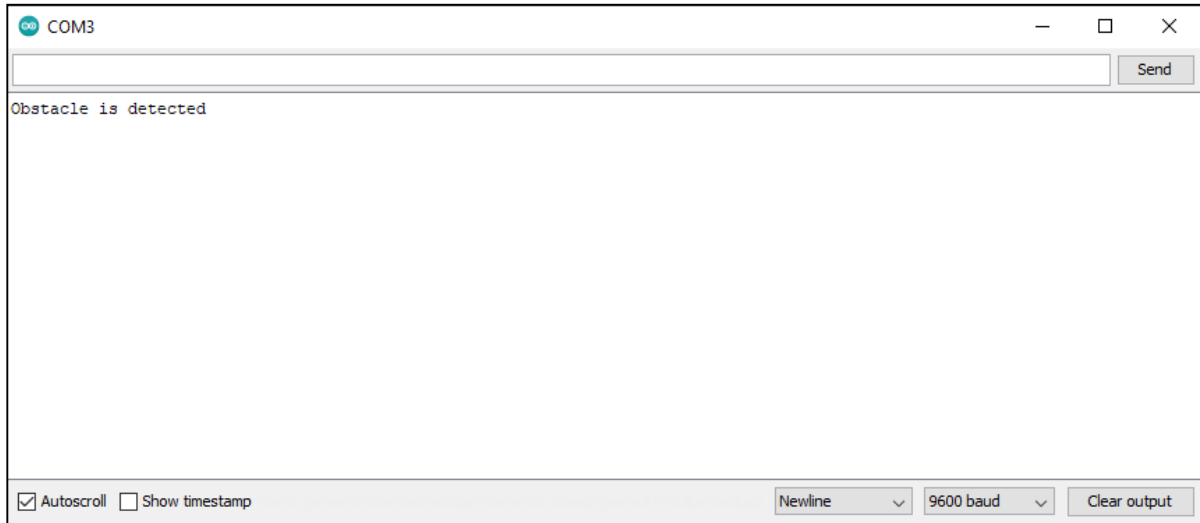


Fig 7.1 obstacle detection

- 2.If Water is detected it will beep sound in different pattern and user will get to know water present ahead.

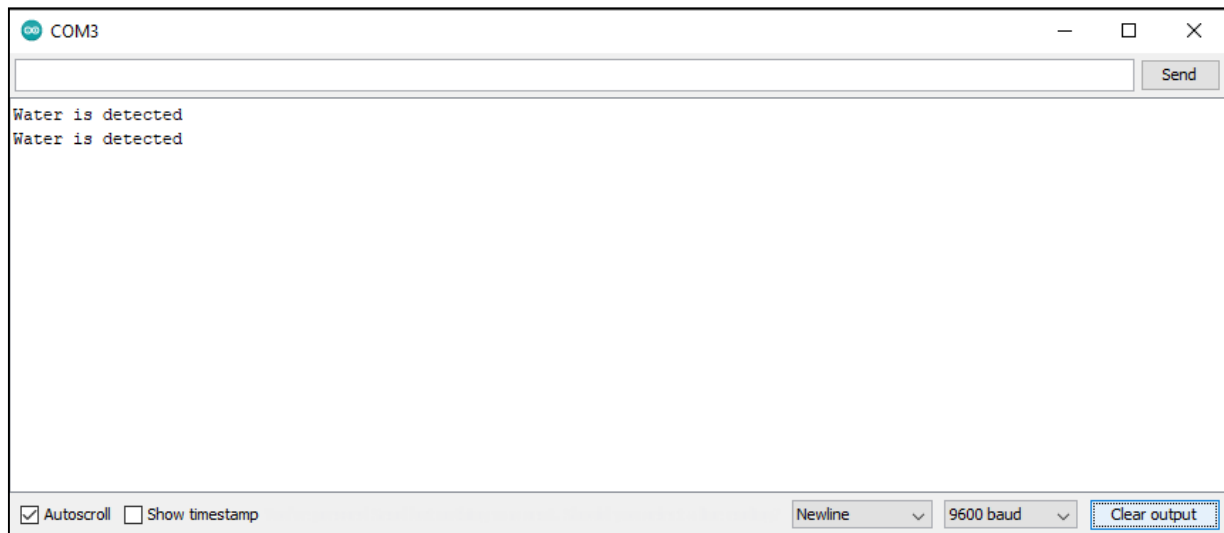


Fig 7.2 Water detection

2. As soon as ESP8266 NodeMCU is powered on the GPS connected to it powers on and it will send the live location as soon as GPS will get connected to nearby satellite. In the output the Red dot represents the location or GPS of the phone or device (from which you are searching for position of visually impaired person) Through sim800l module you can text get location on that number which is inserted in that module and it will give you location .

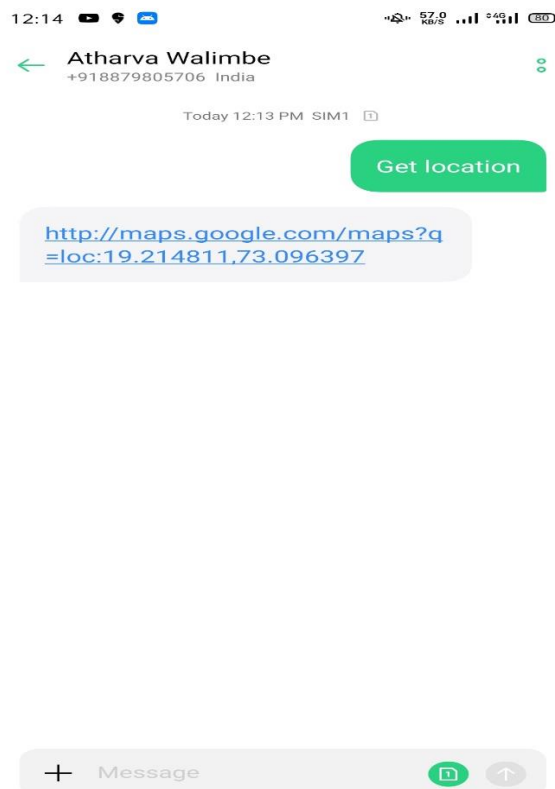


Fig 7.3 Maps link through Text



Fig 7.4 Map location through text

4. As soon as ESP8266 NodeMCU is powered on the GPS connected to it powers on and it will send the live location as soon as GPS will get connected to nearby satellite. In the output the Red dot represents the location or GPS of the phone or device (from which you are searching for position of visually impaired person) .

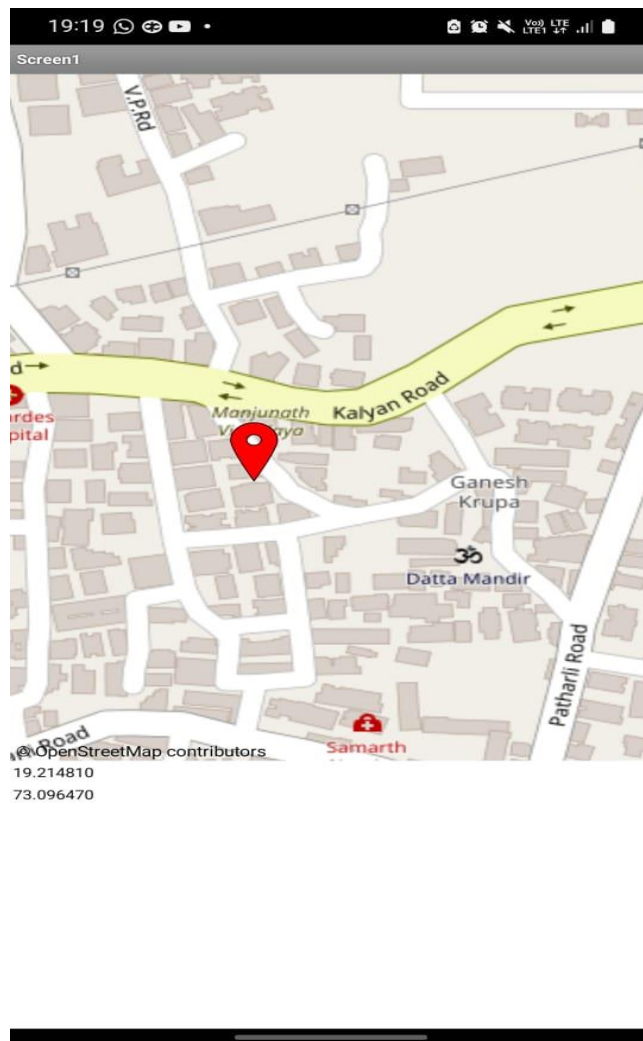


Fig 7.5 Map location through app

8.1 Conclusion:

From this project I learned for Coding you should have patience and faith on yourself. While implementing this project there're conditions where I got stuck up 2,3 days on one error, so you should have faith on yourself that I will solve that error.

In this project, an attempt has been done to design a Smart Blind Walking Stick using Ultrasonic Sensor, Soil moisture sensor, and microcontroller for navigating visually impaired person. The obstacles, presence of water will be detected by the ultrasonic sensor, soil moisture sensor respectively. Based on these results the buzzer connected is activated and produces sound in various pattern. This device can be used by blind or visually impaired person to navigate themselves. The specific pattern of buzzer helps the visually impaired person to know them which kind of obstacle is present in front of them. As soon as the smart blind stick is powered on then the GPS NEO 6M used will powered on and will search for nearby antenna. Whenever it gets connected to the signal it will give a live location of person on the app or through text message by sim8001 module.

8.2 Limitations:

- The ultrasonic sensor cannot sense the obstacle which is present in front of blind person if it is not in range of ultrasonic sensor.
- The device requires continuous internet connectivity for proper functioning of tracking GPS location.
- The GPS will not function properly if the weather is not clear to get the location through the satellite.
- The NodeMCU ESP8266 will not function properly if there is issue in network i.e. internet providing network.
- For the implementation of gps tracking we can't use both the module like through app and through sim as Nodemcu has 32kb memory size and if we integrate both then memory gets full.

8.3 Future Scope:

With the available time and resources, the objective of project was met. The project is able to implemented on a much larger scale. For future projects, one may consider the use of more effective sensors, but which are cost effective and consume little power. This would further enhanced efficiency while reducing costs. If there is possibility of further reducing the cost of this project, it would help a great deal. This is because whether or not such projects are embraced is dependent on how cheap they can be. We will be working more on this project in future on various modules.

They are as follows:

1. Raspberry pi
2. An RF module(wireless communication)

References:

Book References:

Introduction to Embedded system ‘Shibu K V’ McGraw hill education.

Software Engineering, “Ian Somerville”, 8th Edition, Pearson Education.

Website References :

Referred from 14/06/22 to 25/03/23

<https://online-journals.org/index.php/i-joe/article/view/7565https://www.tandfonline.com/doi/full/10.1080/23311916.2019.1692468https://www.maxbotix.com/articles/how-ultrasonic-sensorswork.htm#:~:text=An%20ultrasonic%20sensor%20is%20an,information%20about%20an%20object's%20proximity.>

Referred from 11/07/22 to 24/03/23

https://extension.umn.edu/irrigation/soil-moisture-sensors-irrigation-schedulinghttps://lastminuteengineers.com/neo6m-gps-arduino-tutorial/https://randomnerdtutorials.com/esp8266-pinout-reference-gpioshttps://blynk.io/https://play.google.com/store/apps/details?id=cc.blynk&hl=en_IN&gl=US

Referred for UML diagram:

Referred from 15/07/22 to 03/03/23

<https://www.lucidchart.com>