GAYATRI VIDYA PARISHAD

COLLEGE FOR DEGREE AND PG COURSES (A)

(Affiliated to Andhra University)

Rushikonda, Visakhapatnam

Master of Computer Applications

VISION

"Creating human excellence for a better society"

MISSION

"Unfold into a world class organization with strong academic and research base, producing responsible citizens to cater to the changing needs of the society."

SMART STICK FOR VISUALLY IMPAIRED

A project report submitted in partial fulfilment of

the requirements for the award of the Degree of

Master of Computer Applications

Submitted by

KANDREGULA DEVA AKHIL

(Regd. No: PG212202027)

Under the Guidance of

Sri. Ch. S. S. Rajesh Patnaik,

Assistant Professor

Department of Computer Applications

GAYATRI VIDYA PARISHAD

COLLEGE FOR DEGREE AND PG COURSES(A)

(Affiliated to Andhra University)

Rushikonda, Visakhapatnam.

2021-2023

GAYATRI VIDYA PARISHAD

COLLEGE FOR DEGREE AND PG COURSES (A)

(Affiliated to Andhra University)

Rushikonda, Visakhapatnam

Department of Computer Applications

C E R T I F I C A T E

This is to certify that the project report titled **"SMART STICK FOR VISUALLY IMPAIRED"** is the bona-fide record work carried out by **KANDREGULA DEVA AKHIL (Regd. No. PG212202027),** a student of this college, during the academic year 2021-2023, in partial fulfilment of the requirements of the award of the degree of Master of Computer Applications.

Project Guide:	**Director of MCA**
Sri. Ch. S. S. Rajesh Patnaik	**Prof. I. S. Pallavi**

External Examiner

DECLARATION

I, Mr. **Kandregula Deva Akhil** hereby declare that the project report titled as "**Smart Stick For Visually Impaired**" an original work done at Gayatri Vidya Parishad College for Degree and PG Courses (A), Visakhapatnam, submitted in partial fulfilment of the requirements for the award of Master of Computer Applications to Gayatri Vidya Parishad College for Degree and PG Courses (A), affiliated to Andhra University. I assure that this project is not submitted by me in any other University or College.

KANDREGULA DEVA AKHIL

ACKNOWLEDGEMENT

I consider this as a privilege to thank all those people who helped me a lot for successful completion of the project "**Smart Stick For Visually Impaired**".

I would like to thank, **Principal of Gayatri Vidya Parishad College for Degree and PG Courses (A), Prof. S. Rajani**, who has provided full-fledged lab and infrastructure for successful completion of my project work.

I would like to thank our **Director of MCA**, **Prof. I. S. Pallavi**, who has obliged in responding to every request though she is busy with her hectic schedule of administration and teaching and suggested me with several changes a lot in completion of this project.

I would like to thank our ever-accommodating my project guide **Sri. Ch. S. S. Rajesh Patnaik, Assistant Professor**, who has very obliged in responding to every request though he is busy with his hectic schedule of teaching.

I thank all the **Teaching & Non-Teaching staff** who had been a constant source of support and help during the tenure of completion of this project.

.

KANDREGULA DEVA AKHIL

ABSTRACT

ABSTRACT

The “Smart Stick for Visually Impaired” is an innovative device designed to assist blind individuals with navigating their surroundings more safely and independently. The device incorporates with different sensors to detect obstacles, stairs and drainage holes and the system provide warnings through buzzer and also contains Emergency Push Button. Overall, the smart stick is an excellent example of how technology can be used to improve the lives of individuals with disabilities.

SMART STICK FOR VISUALLY IMPAIRED

TABLE OF CONTENTS

INTRODUCTION

1. INTRODUCTION

1.1. Introduction to Smart Stick for Visually Impaired

The innovative project utilizes the power of the Arduino Uno board to process data from various sensors, enhancing the safety and independence of individuals with visual impairments. The system incorporates an ultrasonic sensor, infrared sensor, and LDR sensor, working together to provide valuable information about the surrounding environment. With the ultrasonic sensor, our Smart Stick detects obstacles and objects within its range, alerting the user and ensuring a safer navigation experience. The infrared sensor assists in detecting nearby objects or individuals, further enhancing awareness and preventing collisions. Additionally, the LDR sensor enables the Smart Stick to measure light levels, aiding users in determining whether they are in a well-lit or dimly lit area. We have also included an emergency push button, allowing the visually impaired user to send an immediate distress message to their family members, alerting them to the potential need for assistance. Furthermore, the Smart Stick is equipped with a buzzer, which can provide audible alerts, helping the user to stay informed about important events or potential hazards. Our project aims to empower visually impaired individuals, granting them greater mobility, security, and peace of mind as they navigate the world around them.

1.2. Introduction to Arduino

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments.

1.2.1. Why Arduino

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers – can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community. There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handy board, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems.

- **Inexpensive -** Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than 800 Rs.
- **Cross-platform -** The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- **Simple, clear programming environment -** The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

- **Open source and extensible software -** The Arduino software is published as opensource tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
- **Open source and extensible hardware -** The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

1.2.2. Pin Configuration of Arduino UNO

Arduino board is an open-source platform used for building electronics projects. Arduino is a programmable circuit's board which we can write a program based on your projects. Arduino program will be uploading with IDE (Integrated Development Environment) software that runs on your computer, it is used to write and upload computer code to the Arduino physical board. Arduino language is merely a set of C/C++ functions that can be called from the code.

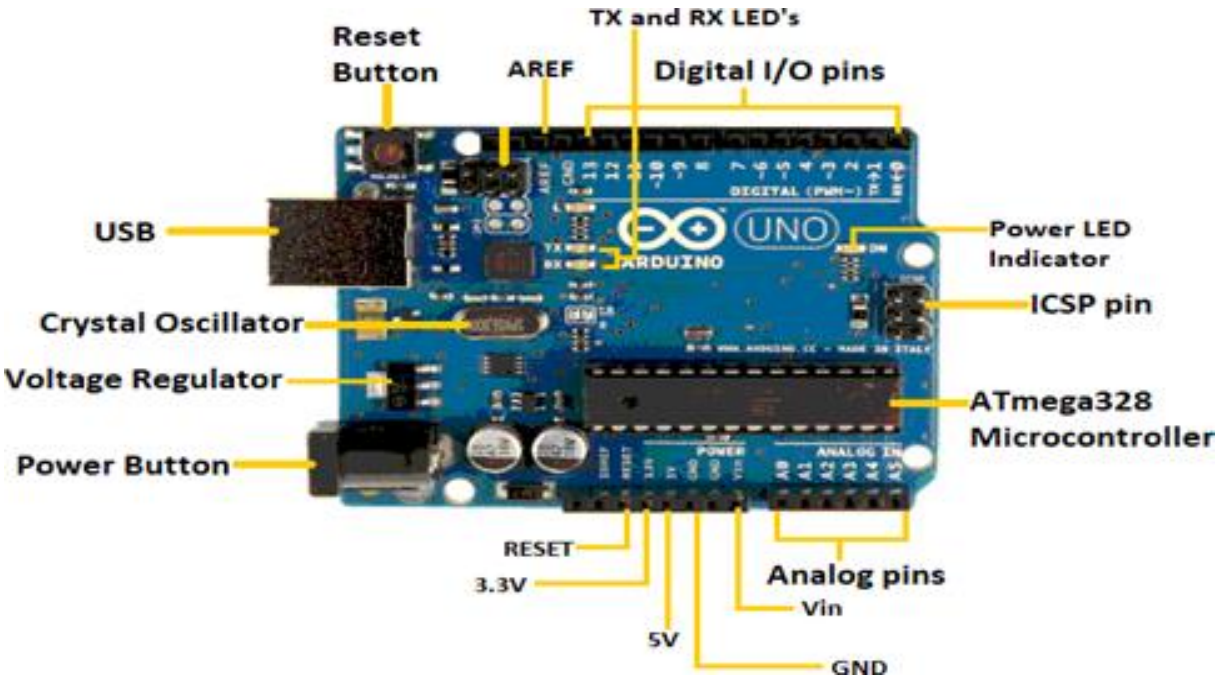

Figure 1.1: Arduino UNO

- **ATmega328 Microcontroller:** It is a single chip Microcontroller of the ATmega328 family. The processor code inside it is of 8-bit. It combines Memory (SRAM, EEPROM, and Flash), Analog to Digital Converter, SPI serial ports, I/O lines, registers, timer, external and internal interrupts, and oscillator.
- **ICSP pin:** The In-Circuit Serial Programming pin allows the user to program using the firmware of the Arduino board.
- **AREF:** The Analog Reference (AREF) pin is used to feed a reference voltage to the Arduino UNO board from the external power supply.
- **USB:** It allows the board to connect to the computer. It is essential for the programming of the Arduino UNO board.
- **Crystal Oscillator:** The Crystal oscillator has a frequency of 16MHz, which makes the Arduino UNO a powerful board.
- **Ground Pins:** The Arduino UNO has several ground pins, which are used to provide a common ground reference for external devices connected to the board.
- **Digital Pins:** The Arduino UNO has 14 digital input/output pins (labelled 0 to 13), which can be used as either input or output. These pins are used to communicate with external devices using digital signals, and can be set to either high or low voltage levels (5V or 0V, respectively).
- **Voltage Regulator:** The voltage regulator converts the input voltage to 5V.
- **Analog Pins:** The Arduino UNO has six analog input pins (labelled A0 to A5), which can be used to read analog signals from sensors such as temperature sensors or potentiometers. These pins can also be used as digital input/output pins if needed.
- **Power Pins:** The Arduino UNO has several power pins, including a 5V pin, a 3.3V pin, and a Vin pin. These pins can be used to power external devices connected to the board.
- **Reset Pin:** The Reset pin is used to reset the microcontroller on the board.

In addition to the standard pins, the Arduino UNO board also has several other pins for special functions, such as the TX and RX pins for serial communication, and the PWM (Pulse Width Modulation) pins for controlling the brightness of LEDs or the speed of motors.

ESP8266 Wi-Fi module

The ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturing company Espressif Systems.

Each ESP8266 Wi-Fi module comes pre-programmed with an AT command set firmware, it can simply hook this up to Arduino device and get as much Wi-Fi ability as a Wi- Fi Shield offers.

- **3V3: –** 3.3 V Power Pin.
- **GND: –** Ground Pin.
- **RST: –** Active Low Reset Pin.
- **EN: –** Active High Enable Pin.
- **TX: –** Serial Transmit Pin of UART.
- **RX: –** Serial Receive Pin of UART.
- **GPIO0 & GPIO2: –** General Purpose I/O Pins. It also known as TX/RX pins are used for Programming the module or for serial I/O purpose.

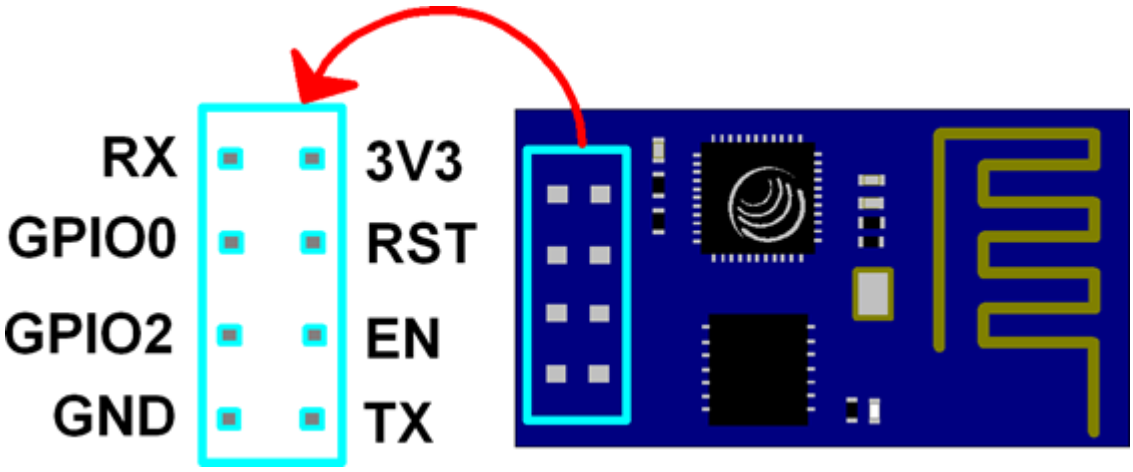

Figure 1.2: ESP8266 Wi-Fi module

Jumper Wires:

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

Figure 1.3: Jumper wires

Ultrasonic sensor

The module has two eyes like structure where one transmits a wave and the other receives it. This sensor is widely used for obstacle detection and also for measuring distances. The working frequency of the sensor is 40 HZ and the maximum range is 4m and min range.

Figure 1.4: Ultrasonic Sensor

Infrared sensor

IR Sensor functioning is also same as that of the Ultrasonic sensor since it also detects the obstacles. But the range is only 0-12 cm so, it is used here for detecting very small distance obstacle.

- The white LED here is an IR LED which works as a transmitter.
- The component next to the IR LED is a photodiode that works as the receiver in the IR sensor.

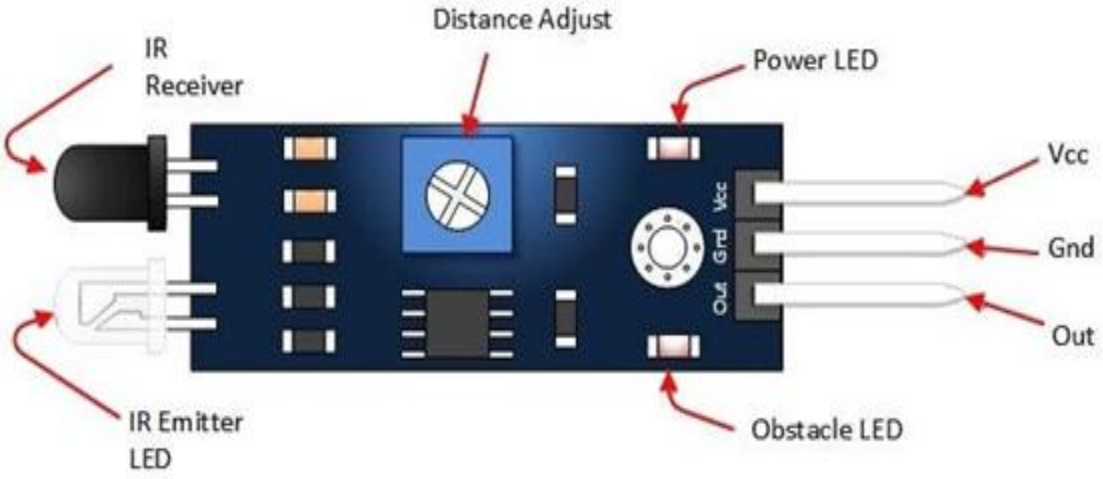

Figure 1.5: IR Sensor

LDR sensor

Light dependent resistor is a 3-pin module. This sensor has sensitivity to the light and it gives a high output as soon as the darkness is observed and vice versa.

Figure 1.6: LDR Sensor

Buzzer

The Buzzer is a device that produces sound when an electric current is passed through it. The buzzer can be directly connected to the board and produce different tones by giving different frequency electric pulses to the buzzer. The pin configuration of the buzzer is shown below.

Figure 1.7: Buzzer

Battery

A battery is a device that stores chemical energy and converts it into electricity.

Figure 1.8: Battery 9V

Bread board

A Bread board is a prototyping tool used to build and test electronic circuits without soldering. It has a grid of interconnected holes that allow easy component insertion and temporary circuit connections.

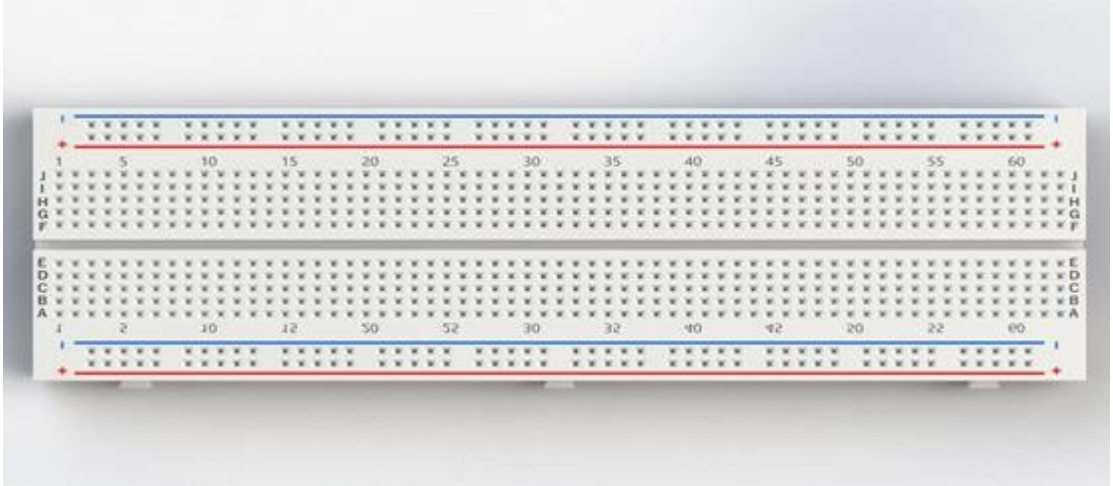

Figure 1.9: Bread board

Obstacle Detection Range Table

Obstacle Type	Sensors Used	Actual Range of Detection	Range of Sensor	Alert Type
Small Object like Stones, Steps, etc.	Infrared	12cm	0 - 9cm	Buzzer (change in Frequency Ex:20)
Humans	Ultrasonic	250cm/8feet	10 - 20cm	Buzzer (change in Frequency Ex:50)
Large objects like Walls, Poles, trees, etc.	Ultrasonic	250cm/8feet	10 - 30cm	Buzzer (change in Frequency Ex:500)
Moving Objects like cars, motor vehicles, etc.	Ultrasonic	360cm/10feet	10 - 40cm	Buzzer (change in Frequency Ex:750)
Man Holes, Pot Holes, Darkness	LDR	Has to Dip	Has to Dip	Buzzer (change in Frequency Ex:150)

Table 1.1: Obstacle Detection Range Table

1.3. Introduction to Internet of Things

The Internet of things (IoT) refers to the concept of extending internet connectivity beyond conventional computing platforms such as personal computers and mobile devices, and into any range of traditionally “dumb” or non-internet-enabled physical devices and everyday objects. Embedded with electronics, Internet connectivity, and other forms of hardware (such as sensors), these devices can communicate and interact with others over the Internet, and they can be remotely monitored and controlled. The definition of the Internet of things has evolved due to convergence of multiple technologies, real-time analytics, machine learning, commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the “smart home”, covering devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smart phones and smart speakers.

1.3.1. Applications

The extensive set of applications for IoT devices is often divided into consumer, commercial, industrial, and infrastructure spaces.

1.3.2. User-centric applications

Smart Home Automation: IoT enables the automation and control of various devices within a home, such as lighting, thermostats, security systems, and appliances. Users can remotely manage and monitor their homes using smartphones or voice commands.

Industrial Automation: IoT is extensively used in industries to optimize processes, monitor equipment, and enhance productivity. It enables real-time data collection, predictive maintenance, and remote monitoring of machines and systems.

Healthcare Monitoring: IoT devices can collect and transmit health data, enabling remote patient monitoring and proactive healthcare. This technology is used in wearable devices, smart medical equipment, and telemedicine applications.

Smart Cities: IoT enables efficient management of resources in urban areas. It can be used for smart traffic management, waste management, environmental monitoring, public safety, and energy management to create sustainable and livable cities.

Agriculture and Farming: IoT solutions help optimize agricultural operations by monitoring soil moisture, temperature, and humidity levels. Farmers can use this data to automate irrigation systems, control pests, and optimize resource utilization for improved crop yield and reduced costs.

Supply Chain Management: IoT enables real-time tracking and monitoring of goods throughout the supply chain. It improves inventory management, reduces losses, and enhances efficiency in logistics operations.

Energy Management: IoT can be used for smart grid systems, enabling monitoring and control of energy consumption in homes, buildings, and industries. It facilitates energy optimization, demand response, and efficient use of renewable energy sources.

Environmental Monitoring: IoT devices can monitor and collect data on air quality, water quality, noise levels, and other environmental factors. This information helps in environmental conservation, early warning systems for natural disasters, and pollution control.

Retail and Customer Experience: IoT is used in retail to track inventory, personalize shopping experiences, and optimize store layouts. It also enables location-based marketing, interactive displays, and seamless online and offline shopping integration.

Transportation and Fleet Management: IoT technology is applied in tracking and managing vehicles, optimizing routes, and monitoring driver behavior. It enhances safety, efficiency, and maintenance of transportation systems.

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1. Introduction

The Smart Stick for Visually Impaired is an innovative project aimed at providing visually impaired individuals with enhanced mobility and safety. This project utilizes Internet of Things (IoT) technology to develop a smart stick that incorporates various sensors for obstacle detection and location sharing. To ensure the success and effectiveness of this project, it is crucial to conduct a comprehensive literature survey. The literature survey allows for the exploration and analysis of existing research and development efforts in the field of assistive technologies for visually impaired individuals.

The primary objective of this literature survey is to gather insights, knowledge, and best practices from relevant research papers, articles, and conference proceedings. By reviewing the existing literature, we can identify the state-of-the-art technologies, methodologies, and design considerations in the domain of smart sticks for visually impaired individuals. This survey will serve as a foundation for designing an efficient and user-friendly smart stick system.

Literature Survey Objectives:

Explore Assistive Technologies for Visually Impaired People: The literature survey aims to provide an overview of the various assistive technologies developed for visually impaired individuals. It will encompass topics such as obstacle detection systems, wearable devices, and navigation aids. By examining existing research, we can gain valuable insights into the strengths, limitations, and advancements in the field.

Investigate IoT-based Solutions for Visually Impaired Individuals: With the rapid advancement of IoT technology, it is crucial to explore IoT-based solutions specifically tailored for visually impaired individuals. The literature survey will focus on IoT-based navigation systems, communication protocols, and integration of sensors to enhance the functionality of the smart stick.

Identify Sensor Technologies for Obstacle Detection: Obstacle detection plays a critical role in ensuring the safety and mobility of visually impaired individuals. The literature survey will

delve into the utilization of different sensor technologies such as ultrasonic sensors, infrared sensors, and LDR sensors for obstacle detection. This exploration will help us understand the effectiveness of these sensors and their integration into the smart stick system.

Examine Emergency Notification Systems: In emergency situations, immediate assistance is crucial for visually impaired individuals. The literature survey will explore the incorporation of an emergency push button and location sharing mechanisms within the smart stick system. Examining the existing research in this area will provide insights into effective emergency notification systems and their integration with IoT technologies.

Block diagram

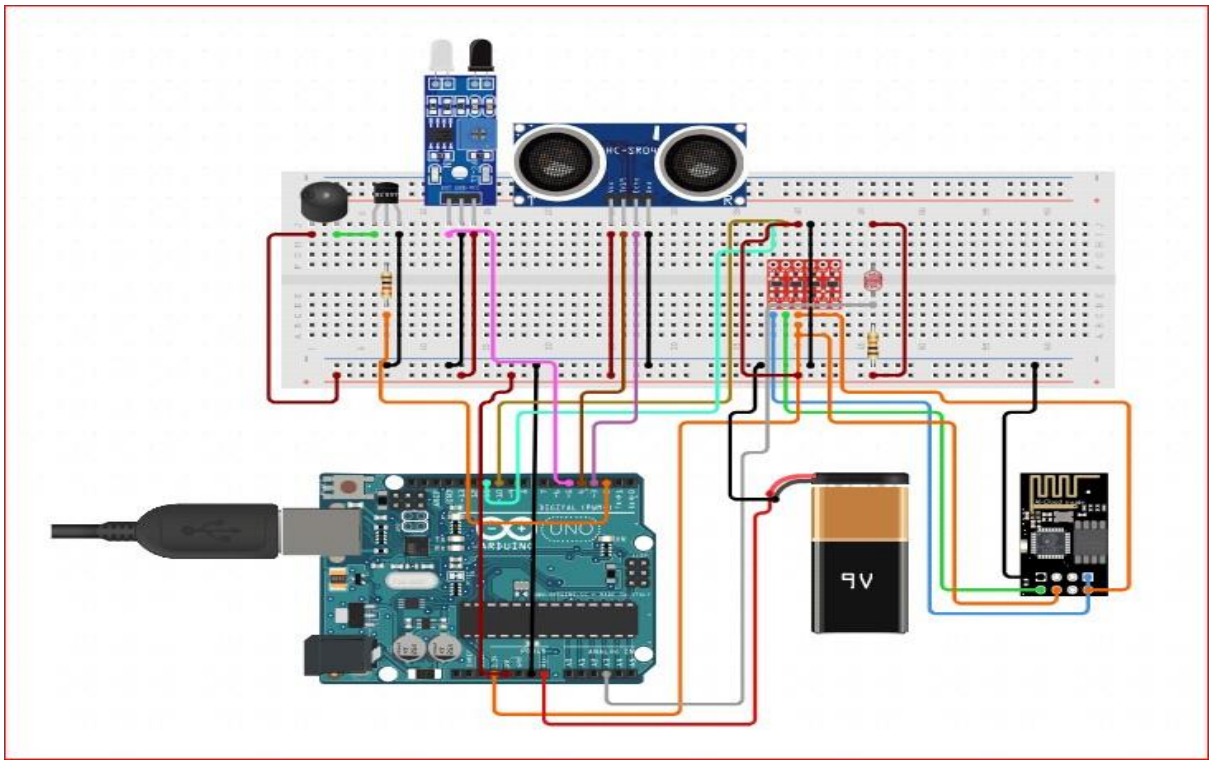

Figure 2.1: Block diagram

2.2. History

The history of blind people predates the advent of the Internet of Greece, and Rome. Blind individuals often relied on the assistance of family members or friends for basic needs. Some societies developed rudimentary forms of education for blind individuals, primarily focused

on religious or philosophical teachings of Things (IoT) by centuries. Throughout history, blind individuals faced various challenges in their daily lives, including mobility, education, and social integration. Here is a brief overview of the history of blind people before the emergence of IoT:

Medieval and Renaissance Periods: During the medieval and Renaissance periods, blind individuals were often dependent on charity and religious institutions for support. In some cases, blind individuals were provided with alms or given shelter in specialized institutions known as alms houses. The first known educational institution for blind people, the Institution for the Blind in Paris, was established in the late 18th century.

Braille: One significant milestone in the history of blind people came with the development of Braille in the early 19th century. Louis Braille, a blind Frenchman, invented the Braille system, which consists of raised dots representing letters, numbers, and musical notations. Braille revolutionized education and literacy for blind individuals, enabling them to read, write, and communicate independently.

Mobility Aids: Before the introduction of IoT, blind individuals used various tools and aids to assist with mobility. The white cane, which originated in the early 20th century, became a widely recognized symbol for blind individuals and provided basic assistance in detecting obstacles. Guide dogs also became popular mobility aids, offering increased independence and navigation support.

Assistive Technologies: Prior to the IoT era, there were assistive technologies specifically designed for blind individuals. These included devices like talking books, which provided audio recordings of written materials, and specialized tools such as tactile maps and raised-line drawings to help with spatial understanding. These technologies helped blind individuals access information and participate in educational and professional settings.

Supportive Organizations: Throughout history, various organizations and societies were established to support blind individuals. These organizations focused on providing education, vocational training, employment opportunities, and social integration for blind individuals. Many of these organizations continue to operate today, advocating for the rights and well-beings of blind people.

It is important to note that the history of blind people is diverse and encompasses different cultures and regions. The advancements and support systems varied across countries and time periods. The emergence of IoT and related technologies has brought new possibilities and improvements in the lives of blind individuals, enhancing their access to information, communication, and mobility.

2.3. Problem statement

The problem statement for the Arduino "Third Eye" project in the context of IoT can be stated as follows:

Visually impaired individuals face challenges in navigating their surroundings independently and safely. Traditional mobility aids, such as white canes, have limitations in detecting obstacles at head or chest height. There is a need for an affordable, portable, and efficient assistive device that leverages IoT technology to provide real-time obstacle detection and navigation support for visually impaired individuals.

The Arduino "Third Eye" project aims to address this problem by utilizing IoT capabilities to develop a smart, wearable device that enhances the mobility and safety of visually impaired individuals. The device should be able to detect obstacles in the environment and provide immediate feedback to the user, enabling them to navigate their surroundings with increased confidence and independence.

Key aspects to consider in addressing this problem include:

1. Obstacle Detection: The device should be equipped with sensors, such as ultrasonic or infrared sensors or LDR sensors, to detect obstacles in the environment. These sensors should have an appropriate range and accuracy to detect objects at various distances.

2. Real-time Feedback: The device should provide real-time feedback to the user about the presence and proximity of obstacles. This feedback can be in the form of audible alerts, vibrations, or haptic feedback to ensure that visually impaired individuals can perceive and understand the information easily.

3. IoT Connectivity: The device should be connected to an IoT platform or a smartphone application to enable advanced functionalities. This connectivity allows for remote monitoring, data logging, and the potential for integrating with other assistive technologies or services.

4. Portability and Wearability: The device should be designed to be lightweight, compact, and wearable, ensuring ease of use and comfort for the user. It should be easily attachable to clothing or accessories, enabling users to carry it conveniently in their daily activities.

5. Affordability and Accessibility: The solution should aim to be cost-effective and accessible to a wide range of visually impaired individuals. Utilizing open-source platforms like Arduino can help in achieving affordability and fostering a community of developers and users who can contribute to the project's improvement and sustainability.

By addressing these aspects, the Arduino "Third Eye" project seeks to provide visually impaired individuals with a reliable and affordable IoT-based assistive device that enhances their mobility, safety, and overall independence in navigating their surroundings.

2.4. Existing System

In existing system “Arduino based third eye for blind people" is prepared to provide help for the blind people to tackle the lack of eyesight vision sense. This device uses the audio and vibration signal to alert the people about the upcoming obstacle in his/her path by using ultrasonic sensor. It helps the blind people only to identify the obstacle by detecting it. The existing system has limited functionalities and it is designed to detect obstacles and provide audio and vibration feedback

2.5. Proposed system

The Proposed System “Smart Stick for Visually Impaired” provides a more extensive view of the user’s surroundings, making it easier to navigate unfamiliar environments. Its ability is to detect darkness, stairs and drainage holes is particularly important since these hazards are not always detectable by traditional aids. Warning system that uses a buzzer sensor to alert users about hazards in their path and the warnings gives different alert for different hazards. The Project provides Emergency Push Button to users that is for emergency situations faced by visual impaired people, to get help from family members the user has to press the push button

then the location of the user will be sent to their family members through cloud, the family members are added earlier. By providing increased mobility, independence, and safety, this device can empower individuals with disabilities to lead more fulfilling lives.

2.6. Requirement Analysis

The functional requirements describe the inputs and outputs of the application. The functional requirements are as follows:

2.6.1. Functional Requirements

- **Navigation assistance:** The Smart stick provide navigation assistance to the user by giving real-time information about obstacles.
- **Object detection:** The stick is able to detect nearby objects such as walls, furniture, and other obstacles to prevent collisions.
- **Emergency assistance:** The smart stick contains a panic button that the user can press in case of an emergency, which sends a signal to a designated family member.

HARDWARE REQUIREMENTS:

- Arduino UNO
- ESP8266 Wi-Fi Module
- Ultrasonic sensor
- Infrared sensor
- LDR (Light Dependent Resistor) sensor

SOFTWARE REQUIREMENTS:

- Operating System: Windows 7 and Above
- Coding language: C++
- Framework: Arduino IDE

2.6.2. Non-Functional Requirements

Reliability: The Smart stick has the ability to function without any failure or interruption to produce accurate expected results to produce accurate and expected result.

Usability: The Smart stick is easy to use.

Maintainability: The Smart stick is easy to maintain and repair, with a modular and easily replaceable design.

Scalability: The Smart stick can be able to upgrade or downgrade easily for changes demanded by the user.

UML MODELING

3. UML MODELING

3.1. Introduction to UML

UML is a standard language for specifying, visualizing, constructing, and documenting the artefacts of software system. UML was created by the Object Management Group (OMG) and UML 1.0 specification drift was proposed to the OMG in January 1997. OMG is continuously making efforts to create a truly industry standard for Unified Modelling Language.

UML is different from the other common programming language such as C++, java, etc.

UML is a pictorial language used to make software blueprints.

UML can be described as a general purpose visual modelling language to visualize, specify, construct and document software system.

Although UML is generally used to model software system, it is not limited within this bound. It is generally used to model software system as well. For example, the process flows in a manufacturing unit, etc.

UML is not a programming language, but tools can be used to generate code in various language using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard.

3.2. Goals of UML

A picture is worth a thousand words, this idiom absolutely fits describing UML. Object-oriented concepts were introduced much earlier than UML. At that point of time, there were no standard methodologies to organize and consolidate the Object-oriented development. It was then that UML came into picture.

There are number of goals for developing UML but the most important is to define some general-purpose modelling language, which all models can use and it also need to be made simple to understand and use.

UML diagram are not only made for developers but also for business users, common people, and anybody interested to understand the system. The system can be a software or non-software system. Thus, it must be clear that UML is not a development method rather it accompanies with processes to make it a successful system. In conclusion, the goal of UML can be defined as a simple modeling mechanism to model all possible practical system in today's complex environment.

3.3. UML Standard Diagrams

The elements are like components which can be associated in diverse ways to make a complete UML picture, which is known as diagram. Thus, it is very important to understand the different diagrams to implement the knowledge in real-life system. Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. If we look around, we will realize that the diagram is not a new concept but it is used widely in different forms in different industries. We prepare UML diagram to understand the system in a better and simple way. A single diagram is not enough to cover all the aspects of the system. UML defines various kinds of diagrams to cover most of the aspects of a system. You can also create your own set of diagrams to meet your requirements. Diagrams are generally made in an incremental and iterative way. There are two broad categories of diagram and they are again divided into subcategories:

- Structural Diagrams
- Behavioral Diagrams

3.3.1. Structural Diagrams

The structural diagram represents the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and are therefore stable. These static parts are represented by classes, interfaces, object, components, and nodes. The four structural diagrams are:

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

3.3.2. Behavioral Diagrams

Any system can have two aspects, static and dynamic. So, a model is considered as complete when both the aspects are fully covered. Behavioral diagram captures the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system.

UML has the following five types of behavioral diagrams:

Use case diagram Sequence diagram State chart diagram

3.4. UML Diagrams

3.4.1. Use Case Diagram

A Use case diagram is a graphical representation that depicts the interactions between users (actors) and a system, showcasing the different use cases or functionalities provided by the system. It illustrates the system's behavior from a user's perspective and helps visualize the relationships between actors and use cases.

Key elements of a use case diagram include:

1. Actors: Represent individuals, roles, or external systems that interact with the system being modelled. Actors are typically depicted as stick figures.

2. Use Cases: Represent specific functionalities or tasks that the system provides to its users. Use cases are depicted as ovals or rectangles.

3. Relationships: Show how actors and use cases are related to each other. The relationships are represented by lines or arrows.

- **Association:** Shows a communication link between an actor and a use case.
- **Generalization:** Represents an inheritance relationship, where one use case inherits properties from another.
- **Include:** Indicates that one use case includes another as a part of its behavior.
- **Extend:** Represents an optional or alternative behavior that can be added to a use case.

Use Case Diagram:

Use Case Diagram for Smart Stick for Visually Impaired

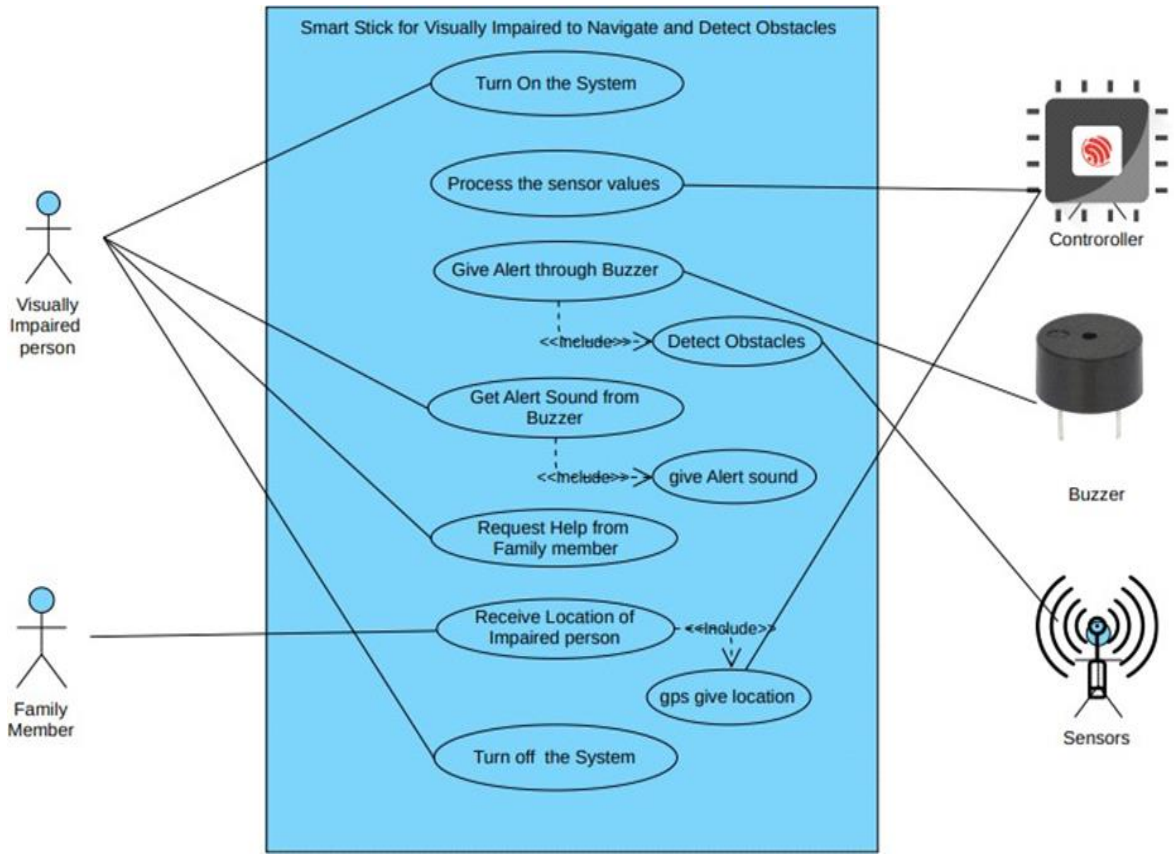

Figure 3.1: Use case diagram

Description:

In this use case diagram, the sensors check the whether there any obstacles or not. If any obstacle is detected then it provides warning sound through the buzzer. If there is no obstacle Detected then the user can keep moving. This process will be repeated continuously.

3.4.2. Sequence Diagram

A Sequence diagram is a type of interaction diagram in UML (Unified Modeling Language) that illustrates the dynamic behavior of a system by showing the sequence of interactions between objects over time.

Key elements of a sequence diagram include:

1. Lifelines: Represent individual objects or actors participating in the sequence of interactions.

2. Messages: Represent the communication or interaction between lifelines. Messages can be synchronous (denoted by a solid arrow) or asynchronous (denoted by a dashed arrow).

3. Activation Boxes: Represent the period of time during which an object is actively processing a message or performing an operation.

4. Return Messages: Represent the response or return message sent by an object to another object after completing a requested operation.

Sequence Diagram:

Sequence Diagram for Smart Stick for Visually Impaired

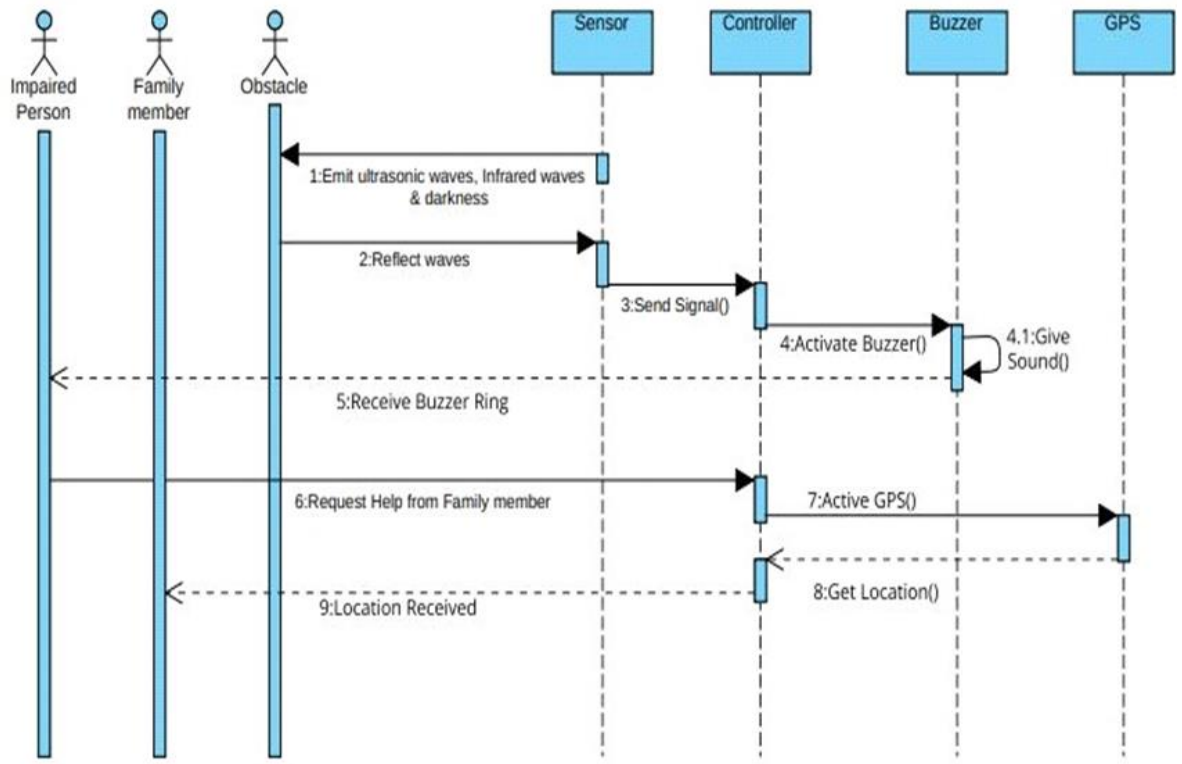

Figure 3.2: Sequence diagram

Description:

The sequence diagram starts with the user initiating the system by starting the smart stick. The smart stick activates its sensors to detect obstacles in the user's surroundings. Once an obstacle is detected, the smart stick sends the information to be analyzed. The smart stick analyzes the obstacle data received from the sensors to determine its nature and location. Based on the analysis, the smart stick generates vibrations or sound feedback to alert the user about the obstacle. The user can interact with the smart stick by pressing buttons or using other input methods. The smart stick responds to the user's interaction, which may involve.

3.4.3. State Chart Diagram

A State chart diagram, also known as a state machine diagram, is a type of behavioral diagram in UML (Unified Modeling Language) that represents the various states and transitions of an object or system over time.

Key elements of a state chart diagram include:

State chart diagrams consist of several components, including states, transitions, events, actions, and conditions. Here's a brief description of each:

State: A state represents a specific condition or mode in which an object or system can exist. It describes the behavior and attributes of an object at a particular point in time.

Transition: A transition signifies a change of state that occurs when a specific event is triggered. It represents the flow from one state to another and may be triggered by internal or external events.

Event: An event is an occurrence that triggers a transition from one state to another. Events can be internal, such as a timer expiration, or external, such as a user input.

Action: Actions represent the operations or behaviors performed when a transition occurs. They depict the activities or tasks that take place during a state change.

Condition: Conditions, also known as guards, are logical expressions that determine whether a transition can be taken. They specify the constraints or requirements that must be satisfied for a transition to occur.

State Chart Diagram:

State Chart Diagram for Smart Stick for Visually Impaired

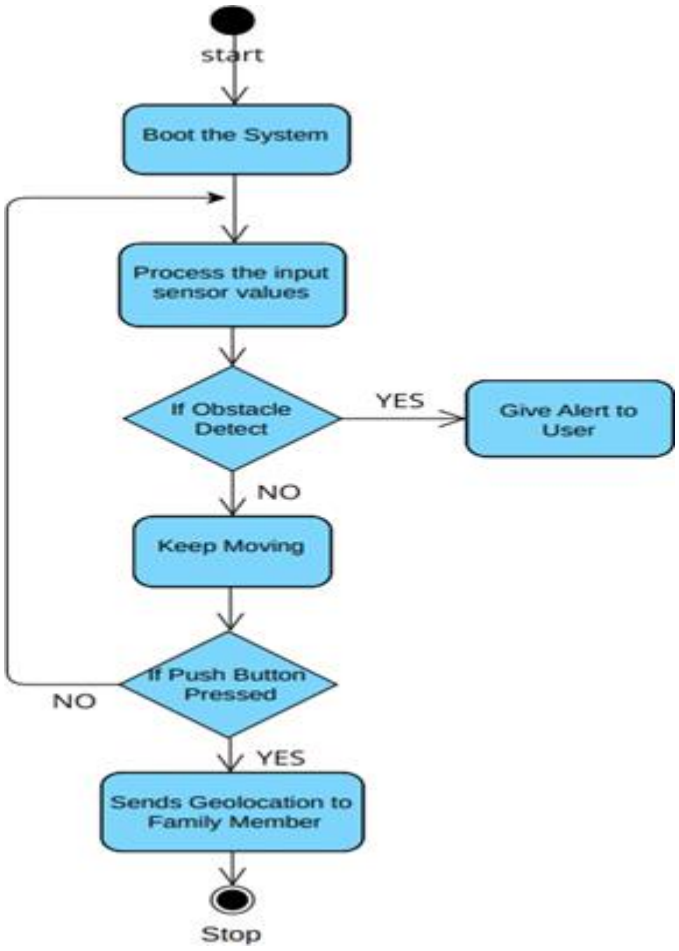

Figure 3.3: State Chart

DESCRIPTION:

The State chart diagram for a Smart stick designed for visually impaired consists of several states and transitions. The system starts in the initial state, where it waits for input or events. When an obstacle is detected through the sensor values the buzzer produce a sound. If there is no obstacle detected then the user will keep moving. The smart stick helps the user to navigate and send the location to the family member by pressing push button.

DESIGN

4. DESIGN

System Design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. In System design, developers:

- Define design goals of the project
- Decompose the system into smaller sub systems
- Design hardware/software strategies
- Design persistent data management strategies
- Design global control flow strategies
- Design access control policies and
- Design strategies for handling boundary conditions.

System design is not algorithmic. It is decomposed of several activities. They are:

- Identify Design Goals
- Design the initial subsystem decomposition
- Refine the subsystem decomposition to address the design goals.

System Design is the transformation and analyzes a model into a system design model. Developers define the design goals of the project and decompose the system into smaller subsystems that can be realized by individual teams. Developers also select strategies for building the system, such as the hardware/software platform on which the system runs, the result of the system design is model that includes a clear description of each of these strategies, subsystem decomposition.

4.1. Design Goals

Design goals are the qualities that the system should focus on. Many design goals can be inferred from the non-functional requirements or from the application domain.

- **Readable:** The arrangement of fonts and words in order to make written content flow in a simple and easy to read manner.
- **Usability:** This targets for the percentage of users who clearly understands the interface of the application

- **Reliable:** Ensures the system perform a specified function within a given system for and expected lifecycle.
- **Reusable:** Reusing several components previously used makes it quicker and easier to design a new project.
- **Extendable:** It is a measure of the ability to extend a system and the level of effort required to implement extension.
- **Flexible:** The capacity to support multiple functions without altering the architecture of the system.
- **Efficient:** Efficiency means that the project does not waste any resources like time and memory in order to process the application.

4.2. Flow Chart of the System

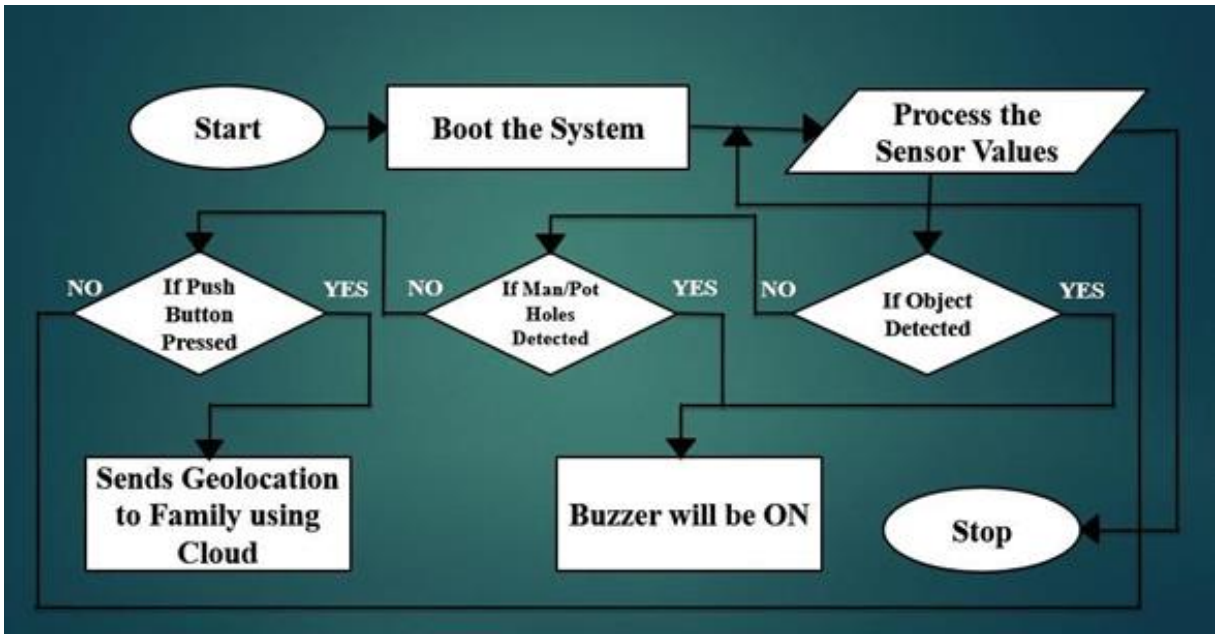

Figure 4.1: Flow Chart

4.3. Algorithm

Step-1: Start

Step-2: Check sensor readings

Step-3: Is there any obstacle detected? Step-4 If yes, Buzzer on.

Step-5: Return to step 2.

Step-6: Keep moving

4.4. Overview Screen of Smart Stick for Visually Impaired

Figure 4.2. Smart stick with integrated sensors

4.3.1. Download the IDE

First, you must download the IDE and install it. Start by visiting Arduino's software page. The IDE is available for most common operating systems, including Windows, Mac OS X, and Linux, so be sure to download the correct version for your OS. If you are using Windows 7 or older, do not download the Windows app version, as this requires Windows 8.1 or Windows 10.

The Arduino IDE

The Arduino IDE is incredibly minimalistic, yet it provides a near-complete environment for most Arduino-based projects. The top menu bar has the standard options, including “File” (new, load, save, etc.), “Edit” (font, copy, paste, etc.), “Sketch” (for compiling and programming), “Tools” (useful options for testing projects), and “Help”.

Projects made using the Arduino are called sketches, and such sketches are usually written in a cut-down version of C++ (a number of C++ features are not included). Because programming a microcontroller is somewhat different from programming a computer, there are a number of device-specific libraries (e.g., changing pin modes, output data on pins, reading analog values, and timers). This sometimes confuses users who think Arduino is programmed in an “Arduino language.” However, the Arduino is, in fact, programmed in C++. It just uses unique libraries for the device.

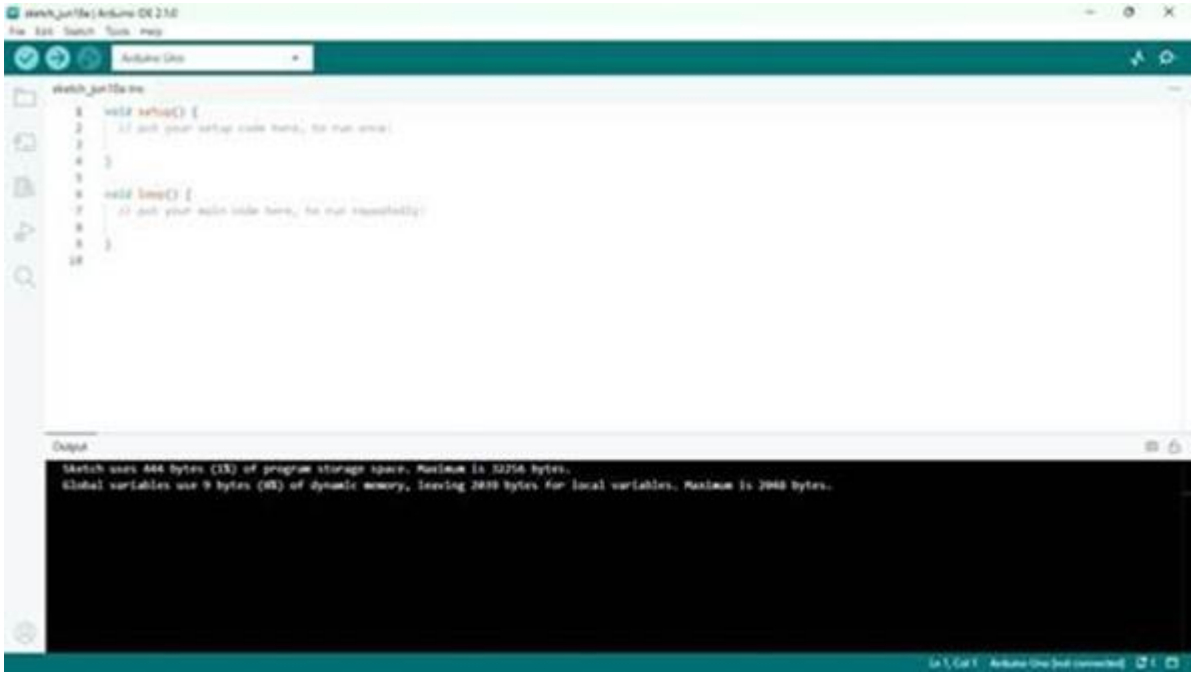

Figure 4.3: Arduino IDE

The 6 Buttons while more advanced projects will take advantage of the built-in tools in the IDE, most projects will rely on the six buttons found below the menu bar.

Figure 4.4: The Button bar

The button bar

- The check mark is used to verify your code. Click this once you have written your code.
- The arrow uploads your code to the Arduino to run.
- The dotted paper will create a new file.
- The upward arrow is used to open an existing Arduino project.
- The downward arrow is used to save the current file.
- The far-right button is a serial monitor, which is useful for sending data from the Arduino to the PC for debugging purposes.

CODING

5. CODING

The goal of coding or programming phase is to translate the design of the system produced during the phase into code in a given programming language, which can be executed by a computer and the performs the computation specified by the design.

The coding phase affects both testing and maintenance. The goal of coding is not to reduce the implementation cost, but the goal should be to reduce the cost of later phase. In other words, the goal is not to simplify the job of programmer. Rather the goal should be to simplify the job of the tester and maintainer.

5.1. Coding Approach

There are two major approaches for coding any software system. They are top-Down approach and bottom-up approach.

Bottom-up approach can suit for developing the object-oriented systems. During system design phase of reduce the complexity. We decompose the system into appropriate number of subsystems, for which objects can be modelled independently. These objects exhibit the way the subsystems perform their operations.

Once object have been modeled they are implemented by means of coding. Even though related to the same system as the objects are implemented of each other the Bottom-Up approach is more suitable for coding these objects.

In this approach, we first do the coding of objects independently and then we integrate these modules into one system to which they belong. In this project, top-Down approach is followed. For registration and Login. User will click and stores the intruder information intensely.

5.2. Information Handling

Any software system requires some amount of information during its operation selection of appropriate data structures can help us to produce the code so that objects of the system can better operate with the available information decreased complexity.

In this project, Encryption and decryption will not be possible if the image fields are vacant. System will not have any default values. User must specify each secret file name in encryption and locate all required operations in decryption.

5.3. Programming Style

Programming style deals with act of rules that a programmer must follow so that the characteristics of coding such as Traceability, Understands the ability, Modifiability, and Extensibility can be satisfied.

In the current system, we followed the coding rules for naming the variables and methods. The system is developed in a very interactive and users friendly manner.

5.4. Verification and Validation

Verification is the process of checking the product built is right. Validation is the process of checking whether the right product is built. During the Development of the system coding for the object has been thoroughly verified from various aspects regarding their design, in the way they are integrated and etc. The various techniques that have been followed for validation discussed in testing the current system. Validations applied to the entire system at two levels:

5.5. Form level validation

Validations of all the inputs given to the system at various points in the forms are validated while navigating to the next form. System raises appropriate custom and predefined exceptions to alert the user about the errors occurred or likely to occur.

Validations at the level of individual controls are also applied whenever necessary. System pops up appropriate and sensuous dialogs whenever necessary

5.6. Source Code

```
#include <SoftwareSerial.h>

SoftwareSerial espSerial(2, 3); // RX, TX pins for ESP8266

// Wi-Fi credentials

const char* ssid = "MrDev";

const char* password = "developer";

// IFTTT Webhooks details

const char* iftttEventName = "push_button";

const char* iftttKey = "chv7_4xyttRqPBue7Iw7OJ";

// Push button details

const int buttonPin = 11;  // Pin connected to the push button

bool buttonPressed = false;

// Pins connected to the ultrasonic sensor

#define trigPin 4

#define echoPin 5

// Pins for IR Sensor

#define irdetect 6

// Pin connected to the LDR

#define ldr 7

// Pin connected to the piezo buzzer

#define alarm 10

void connectToWiFi() {

  // Connect to Wi-Fi network
```

```
  espSerial.println("AT+CWMODE=1"); // Set ESP8266 to station mode
  delay(1000);
  String cmd = "AT+CWJAP=\"" + String(ssid) + "\",\"" + String(password) + "\"";
  espSerial.println(cmd);
  delay(5000);
  if (espSerial.find("OK")) {
    Serial.println("Connected to Wi-Fi");
  } else {
    Serial.println("Failed to connect to Wi-Fi");
  }
}
void triggerIFTTTMessage() {
  // Trigger IFTTT message
  String message = "Button pressed on Arduino Uno";
  espSerial.print("AT+CIPSTART=\"TCP\",\"maker.ifttt.com\",80\r\n");
  delay(5000);
  if (espSerial.find("OK")) {
    Serial.println("TCP connection established");
    start(1500, "Emergency situation", 1000);
  } else {
    Serial.println("Failed to establish TCP connection");
    stop();
    return;
  }
```

```
 String postRequest = "POST /trigger/" + String(iftttEventName) + "/with/key/" + String(iftttKey) + " HTTP/1.1\r\n" +
                      "Host: maker.ifttt.com\r\n" +
                      "Content-Type: application/x-www-form-urlencoded\r\n" +
                      "Content-Length: " + String(message.length()) + "\r\n\r\n" +
                      message;
  espSerial.print("AT+CIPSEND=");
  espSerial.println(postRequest.length());
  delay(2000);
  if (espSerial.find(">")) {
    Serial.println("Sending POST request");
  } else {
    Serial.println("Failed to send POST request");
    return;
  }
  espSerial.print(postRequest);
  delay(5000);
  if (espSerial.find("OK")) {
    Serial.println("IFTTT message triggered successfully");
  } else {
    Serial.println("Failed to trigger IFTTT message");
  }
}
void setup() {
```

```
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(115200);
  while (!Serial) continue;
  espSerial.begin(9600);
  Serial.println("ESP8266 module initialized");
  connectToWiFi();
  // Initialize the sensor pins
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(irdetect, INPUT);
  //pinMode(alarm, OUTPUT);
  pinMode(ldr, INPUT);
  Serial.println("Server started");
}
void loop() {
  int buttonState = digitalRead(buttonPin);
  if (buttonState == LOW) {
    if (!buttonPressed) {
      buttonPressed = true;
      triggerIFTTTMessage();
    }
  } else {
    buttonPressed = false;
   // Establish variables for duration of the ping and the distance result in inches and centimeters
```

```
long duration, inches, cm;
// The PING))) is triggered by a HIGH pulse of 2 or more microseconds
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(5);
digitalWrite(trigPin, LOW);
// Take reading on echo pin
duration = pulseIn(echoPin, HIGH);
// Convert the time into a distance
inches = microsecondsToInches(duration);
cm = microsecondsToCentimeters(duration);
int irStatus = digitalRead(irdetect);
// Checking objects with the ultrasonic sensor
if (inches >= 6 && inches <= 15) {
  start(2000, "Ultrasonic -- object detect", 50);
} else if (inches >= 15 && inches < 30) {
  start(1700, "Ultrasonic -- object detect", 750);
} else if (inches >= 30 && inches < 40) {
  start(1500, "Ultrasonic -- object detect ", 500);
} else {
  stop();
}
```

```
    // If nothing detected in the ultrasonic sensor, check the IR sensor
    if (irStatus == 0) {
      start(2000, "IR sensor -- object detect", 20);
    } else {
      stop();
      // If nothing detected in the ultrasonic sensor and IR sensor, check the LDR sensor
      if (digitalRead(ldr) == 1) {
        start(2000, "LDR sensor -- obstacle detect", 150);
      } else {
        stop();
      }
    }
  }
}
void start(int freq, const char* msg, long ms) {
  Serial.println(msg);
  tone(alarm, freq);
  delay(ms);
}
void stop() {
  noTone(alarm);
  delay(100);
}
long microsecondsToInches(long microseconds) {
```

```
  return microseconds / 74 / 2;
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

TESTING

6. TESTING

Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior of the system. Testing is a critical role in quality assurance and ensuring the reliability of development and these errors will be reflected in the codes the application should be thoroughly tested and validated.

Unit testing finds the differences between the object design model and its corresponding components. Structural testing finds differences between the system design model and a subset of integrated subsystems. Functional testing finds differences between the use case model and the system.

Finally, performance testing, finds differences between non-functional requirements and actual system performance. Form modeling point of view, testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems.

6.1. Testing Activities Unit Testing

Unit testing focuses on testing individual components or modules of the smart stick system. It involves verifying the functionality of the sensors, such as ultrasonic sensors, infrared sensors, and LDR sensors, to ensure accurate obstacle detection. Unit testing also includes testing the emergency push button and its functionality for sending location information.

Integration Testing:

Integration testing aims to validate the proper integration and interaction of different components within the smart stick system. It involves testing the seamless communication between the sensors, microcontroller or processing unit, and the buzzer for obstacle alerts. Integration testing also verifies the integration of the emergency push button with the location-sharing mechanism.

Functional Testing:

Functional testing evaluates the overall functionality and performance of the smart stick system. It includes testing various scenarios of obstacle detection using different sensors and

verifying the appropriate generation of alerts through the buzzer. Functional testing also involves testing the emergency push button to ensure it triggers the location-sharing mechanism accurately.

Performance Testing:

Performance testing assesses the system's performance under different conditions and loads. It includes testing the response time of obstacle detection and the accuracy of the alert system. Performance testing may involve subjecting the smart stick to various environmental conditions and obstacles to evaluate its reliability and responsiveness.

User Acceptance Testing:

User acceptance testing involves testing the smart stick system with visually impaired individuals or representatives from the target user group. It aims to gather feedback on usability, accessibility, and overall user experience. User acceptance testing helps identify any potential usability issues, allowing for improvements and refinements based on user feedback.

Field Testing:

Field testing involves deploying the smart stick system in real-world environments to evaluate its performance and reliability. It includes testing the system in various settings, such as indoor and outdoor environments, different terrains, and challenging conditions. Field testing provides valuable insights into the system's practical functionality and helps identify any additional improvements needed.

Each testing activity should be well-documented, including the test objectives, test scenarios, expected outcomes, and actual results. Testing activities should be conducted iteratively throughout the development process, with appropriate adjustments and improvements made based on the test results. The documentation should also include any test scripts, test data, and test reports generated during the testing phase.

6.2. Test Plan

The purpose of this project is to develop a smart stick embedded with IoT technologies to assist visually impaired individuals in navigating their surroundings.

It involves various functionalities like:

Obstacle Detection: Test the smart stick's ability to accurately detect obstacles in various environments and at different distances.

Distance Measurement: Verify the accuracy of distance measurement provided by the smart stick to help users assess their proximity to obstacles.

Audio Feedback: Test the audio feedback system to ensure it effectively provides the signals to visually impaired users.

Button and Control Testing: Validate the functionality of buttons, switches, or controls on the smart stick, ensuring they respond appropriately to user input.

6.3. Test Case Screens

Connection between Arduino UNO and Sensors:

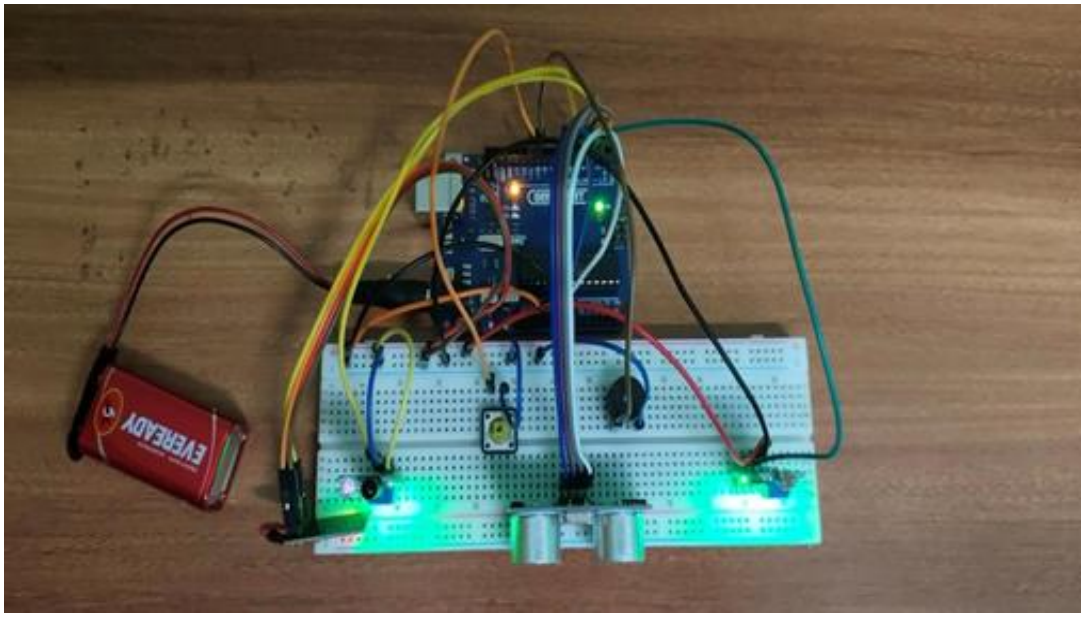

Fig 6.1: Connection between Arduino UNO and Ultrasonic sensor

Let’s connect the Ultrasonic sensor with the Arduino UNO. Connect the VCC and GND of the sensor to the 5volt and GND of the Arduino.

Connection between Arduino UNO and LDR sensor:

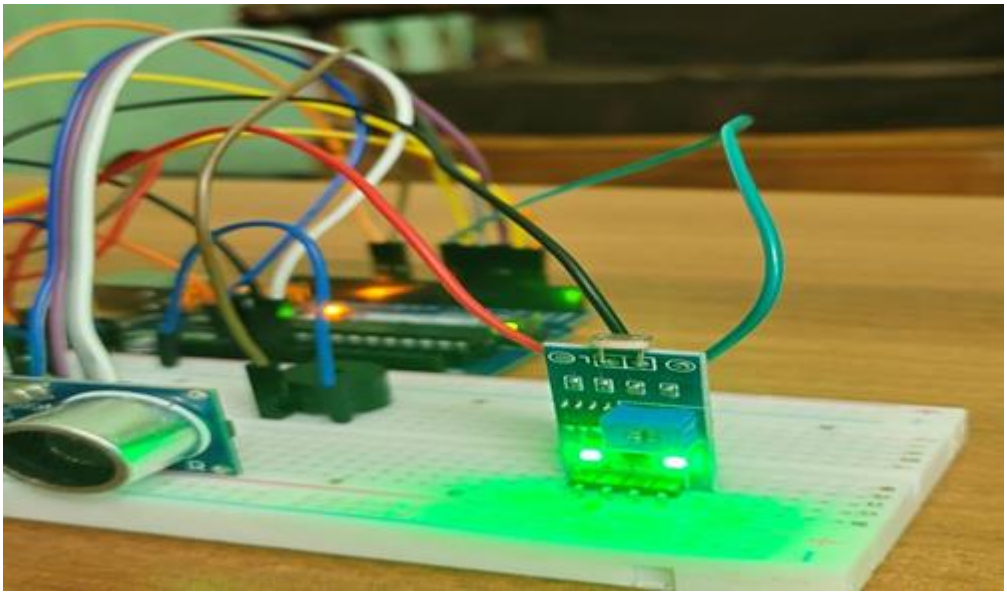

Fig 6.2: Light Dependent Resistor

The next step is to connect the LDR (Light Dependent Resistor) to the Arduino UNO. Connect the 3.3 volt and GND of the LDR to 3.3 volt and GND of the Arduino UNO. Connect the DO of the Arduino UNO to DO of the sensor.

Connection between Arduino UNO and IR sensor:

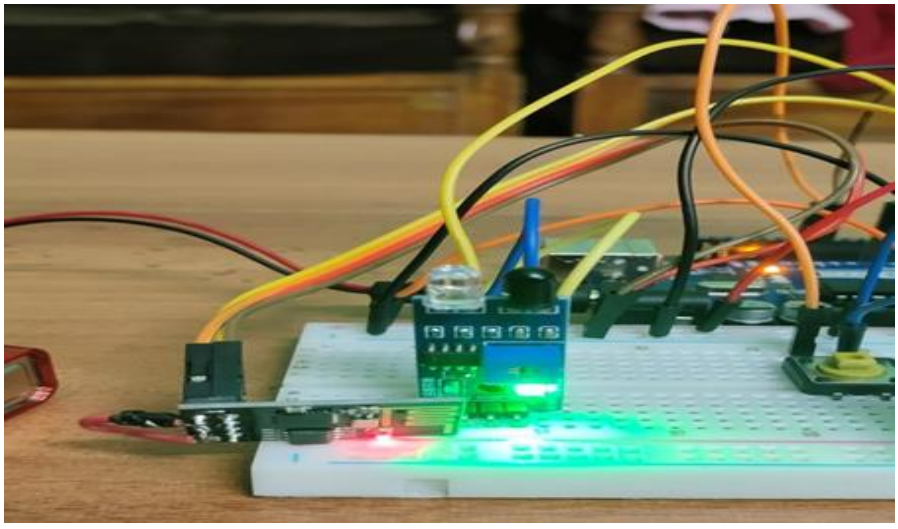

Fig 6.3.: Infrared sensor

IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings. Connect V pin and GND pin of the IR sensor to 3.3 volts and GND of Arduino UNO.

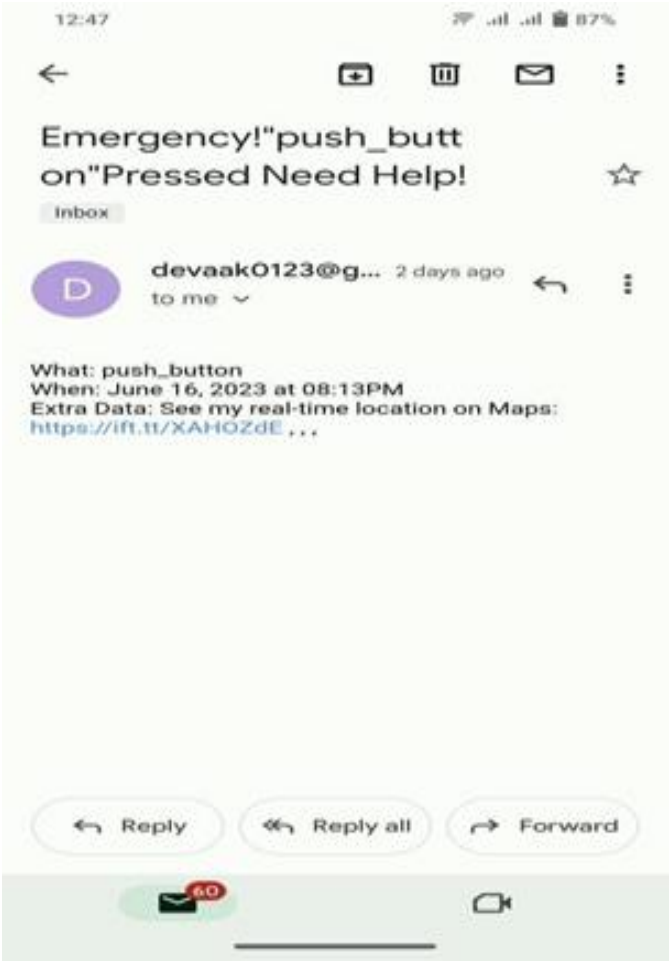

Fig 6.4: Email alert with Navigation

The Emergency push button triggers an action that sends an alert message when pressed by the user. It also navigates the location of the user and sends it through email.

6.4. Test Cases

Test Case	Test Case Description	Testing	Expected Result	Actual Result	Pass/ Fail
1	Wi-Fi Connected /Not	ESP 8266 Wi-Fi Module	Connected	Connected	Pass
2	Large Object detected using Ultra Sonic sensor	Ultra sonic sensor detects/not	Buzzer ON	Buzzer ON	Pass
3	Small Object detected using Infrared sensor	Infrared sensor detects/not	Buzzer ON	Buzzer ON	Pass
4	Manhole/Potholes detected	LDR sensor detects/not	Buzzer ON	Buzzer ON	Pass
5	Push Button pressed	Location Send via message	Message Trigger	Message Triggered	Pass

Table 6.1: Test Cases

Test Case 1

Test Case Description: Wi-Fi Connected /Not

Testing: ESP 8266 Wi-Fi Module

Actual Result: Connected

Test Case 2

Test Case Description: Large Object detected using Ultra Sonic sensor

Testing: Ultra sonic sensor detects/not

Actual Result: Buzzer ON

Test Case 3

Test Case Description: Small Object detected using Infrared sensor

Testing: Infrared sensor detects/not

Actual Result: Buzzer ON

Test Case 4

Test Case Description: Manhole/Potholes detected

Testing: LDR sensor detects/not

Actual Result: Buzzer ON

Test Case 5

Test Case Description: Push Button pressed

Testing: Location Send via message

Actual Result: Message Triggered

CONCLUSION

7. CONCLUSION

In conclusion, the development of the Smart Stick for Visually Impaired has been driven by advancements in technology and a growing need to provide visually impaired individuals with enhanced mobility, independence, and safety. By integrating ultrasonic sensors, infrared sensors, and LDR sensors, along with IoT technology, the smart stick offers accurate obstacle detection, reliable alerts, and an emergency push button for location sharing. Through rigorous testing and user feedback, we have ensured its functionality, usability, and durability. The Smart Stick for Visually Impaired represents a significant step forward in assistive technology, empowering visually impaired individuals to navigate their surroundings with confidence and regain their individuality.

REFERENCE

8. REFERENCE

1. Arduino Based Third Eye for Blind People by Ankush Yadav, Manish Kumar, Vijay Gupta, Shashi Bhushan - KIET Group of Institutions, Delhi-NCR, Ghaziabad, U.P., India Source: International Journal for Research in Applied Science & Engineering Technology (IJRASET) - ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538 Volume 10 Issue V May 2022- Available at www.ijraset.com.

2. "Smart Cane for Visually Impaired People Using Ultrasonic Sensors" by S. R. Polipalli, S. M. Biradar, and A. R. Birajdar. This research paper discusses the development of a smart cane using ultrasonic sensors for obstacle detection. (Link:https://www.researchgate.net/publication/320413225_Smart_Cane_for_Visually_Impaired_People_Using_Ultrasonic_Sensors)

3. "Smart Cane for Visually Impaired" by P. Dhakal and R. Mishra. This paper presents the design and implementation of a smart cane that incorporates ultrasonic sensors, GPS, and GSM technology for navigation and safety of visually impaired individuals. (Link: https://ieeexplore.ieee.org/abstract/document/8635876)

4. "Navigation Systems for the Blind: A Survey" by Shu-Hao Fan, et al. (2019) Source: IEEE Transactions on Human-Machine Systems, Vol. 49, Issue 6.

5. "Assistive Technologies for People with Diverse Abilities" by Giulio Lancioni, Nirbhay N. Singh, et al. (2014)

6. Udemy Tutorials – Arduino Bootcamp: Learning Through Projects by Lee Assam.

7. Arduino Documentation.

8. IFTTT Documentation.

9. Software Engineering Principles by Roger. S. Pressman.

10.Object Oriented S/W Engineering by Tata McGraw Hill.

APPENDIX

9. APPENDIX

9.1. List of Figures

9.2. List of Tables