ABSTRACT

The **"Safe Guardian Real-Time alert System"** is a comprehensive safety device designed to protect elderly individuals, women, and children. This system integrates GPS, GSM 900A, a speaker, a microphone, a push button, and an accelerometer to provide real-time monitoring and emergency response. Key features include Single Push Button Press Sends an emergency message indicating a serious health issue or potential kidnapping, along with the exact GPS location, and initiates a phone call to a designated contact. Double Push Button Press sends a confirmation message with the GPS location without making a call, suitable for non-urgent updates like taking medication or confirming safe arrival home. Accident Detection when the accelerometer detects a fall, it automatically sends an alert message stating Accident has occurred with the exact GPS location and makes an Accident Alert call. This system ensures timely communication and safety for vulnerable individuals, providing a reliable way to manage and respond to emergencies in real time.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

The "Safe Guardian Real-Time Alert System" is an advanced safety device meticulously crafted to offer robust protection for elderly individuals, women, and children. This innovative system merges GPS, GSM 900A, a speaker, a microphone, a push button, and an accelerometer to deliver comprehensive real-time monitoring and emergency response.

Central to the Safe Guardian system is its intuitive push button feature. A single press activates an emergency alert, transmitting the user's GPS location and initiating a call to emergency services, ensuring immediate assistance. For non-urgent situations, a double press of the button sends updates without triggering an emergency response, providing flexibility in communication. By integrating these technologies, the Safe Guardian Real-Time Alert System ensures timely communication and effective emergency management, significantly enhancing the safety and security of its user.

Moreover, the built-in microphone and speaker allow for real-time communication between the user and emergency responders, ensuring clear and direct interaction during critical moments. The GSM 900A module ensures reliable connectivity, enabling the device to function seamlessly across various geographical locations. With a focus on user-friendly design, the system is easy to operate, making it accessible even for those who may not be tech- savvy.

For added convenience, the device can be programmed with multiple emergency contacts, ensuring that help is always available from trusted sources. The GPS tracking feature provides continuous location updates, allowing family members and caregivers to monitor the user's whereabouts in real-time.In case of false alarms, the system allows for easy cancellation, preventing unnecessary emergency responses.

Overall, the Safe Guardian Real-Time Alert System offers a comprehensive and reliable solution for personal safety, combining advanced technology with user-friendly features to deliver peace of mind for both users and their loved ones.

CHAPTER 2

LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

2.1 Literature Survey

1. De Silva, L., Yatigammana, M. Y. A., & Dissanayake, D. A. (2016).Design and Development of a Real-Time Emergency Alert System for Elderly and Disabled People. This project created an emergency alert system using health sensors, a GSM module, and a microcontroller to monitor the user's health and send SMS alerts when abnormalities were detected. **Limitation:** It lacked GPS for location tracking and did not consider emergencies like falls or kidnappings. **Our Implementation:** The "Safe Guardian Real-Time Alert System" adds GPS for location tracking, multi-functional emergency alerts, accident detection with an accelerometer, and real-time communication via phone calls, offering enhanced emergency response capabilities.

2. Nair, M. N., Athira, K. S., & Pranav, K. S. (2017). Development of an Accident Detection and Alerting System for Vehicles Using Arduino and GSM Module. This system used an accelerometer, microcontroller, GSM, and GPS to detect vehicle collisions and send SMS alerts with GPS coordinates. **Limitation:** It was designed for vehicle accidents and lacked applicability for personal safety. **Our Implementation:** Our project extends this concept to include personal safety, fall detection, child monitoring, and real-time alerts for various emergencies, providing a more comprehensive solution.

3. Rajasekaran, A., Jeyanthi, S., & Geetha, K. (2018). A Review on GSM Based Automatic Accident Alert System for Vehicles. This study reviewed GSM-based accident alert systems, focusing on accelerometers, GSM, GPS, and microcontrollers. **Limitation:** It was theoretical, without practical implementation or testing. **Our Implementation:** Our project offers a practical, tested prototype with multi-functional alerts and comprehensive safety features, ensuring real-world applicability and effectiveness.

4. Gaikwad, R. V., Kulkarni, P. S., & Pagar, S. D. (2019). A Survey on GPS and GSM Based Tracking Systems. This survey highlighted the importance of real-time location tracking and SMS updates in safety systems. **Limitation:** It focused solely on tracking, lacking emergency alerts and additional safety features. **Our Implementation:** Our project integrates emergency alerts, multi-functional push buttons, and accident detection for comprehensive safety, combining tracking with proactive emergency management.

5. Singh, S. K., & Singh, A. K. (2021). A Survey on Child Safety System Using IoT. This survey examined IoT-based child safety systems, focusing on GPS, GSM, microcontrollers, and IoT sensors. **Limitation:** It was theoretical with limited practical insights. **Our Implementation:** Our project provides a practical IoT-based child safety system with real-time monitoring, multi-functional alerts, and real-time communication via phone calls, offering effective, actionable safety solutions.

2.2 Consolidated Literature Review

The Safe Guardian Real-Time Alert System combines GPS tracking, GSM connectivity, an intuitive push button, an accelerometer for fall detection, and a speaker and microphone for two-way communication. These features, coupled with long battery life, water resistance, and affordability, make it a comprehensive and reliable solution for personal safety. By integrating advanced technologies with user-friendly design, the system ensures timely communication and effective emergency management, significantly enhancing the safety and security of its users.

CHAPTER 3
PROBLEM ANALYSIS

CHAPTER 3
PROBLEM ANALYSIS

The Safe Guardian Real-Time Alert System addresses a critical need for timely and effective communication during safety emergencies and potential threats. In today's fast-paced world, traditional safety measures often fall short in providing immediate, actionable information. Delays in alerting users or response teams can exacerbate the impact of emergencies, leading to increased risk and potentially severe consequences. Current systems may lack integration with diverse data sources, resulting in fragmented or incomplete information. Additionally, users often face challenges with outdated or non-intuitive interfaces that hinder prompt action. The problem is further compounded by the need for scalable solutions that can handle varying levels of data and user demands. Safe Guardian aims to bridge these gaps by delivering a cohesive, real-time alert mechanism that ensures rapid notification, integrates multiple data sources, and offers a user-friendly experience. This system is designed to enhance situational awareness, streamline communication, and ultimately improve safety outcomes across different contexts. The Safe Guardian Real-Time Alert System is a cutting-edge solution designed to meet the urgent need for fast and effective emergency communication. Unlike traditional systems that often fail to provide timely, actionable alerts, Safe Guardian ensures immediate notification to users and response teams. It seamlessly integrates a variety of data sources—such as live surveillance, environmental sensors, and social media updates—to offer a comprehensive overview of unfolding situations. This integration eliminates fragmented information and enhances decision-making. The system features a modern, intuitive interface that simplifies user interaction, even under stress. It is built to be scalable, adapting to different levels of data and user demand efficiently. Safe Guardian's real-time analytics help prioritize alerts and responses, improving overall situational awareness. By streamlining communication and response coordination, Safe Guardian significantly boosts safety outcomes and reduces the impact of emergencies.

CHAPTER 4

PROPOSED SYSTEM

CHAPTER 4

PROPOSED SYSTEM

The Safe Guardian Real-Time Alert System is designed to enhance safety and security by providing immediate notifications about potential threats or emergencies. This comprehensive system integrates a user-friendly mobile app and web dashboard for seamless interaction. It leverages a network of sensors and external data sources to monitor various safety parameters, such as environmental changes or unusual behavior. The system's core is its robust processing engine, which analyzes incoming data to detect anomalies and generate real-time alerts based on predefined criteria. Notifications are dispatched instantly via push notifications, SMS, or email, ensuring that users and administrators are promptly informed. Additionally, the Safe Guardian system includes customizable user settings, historical data access, and integration with emergency services to facilitate a rapid response. Designed with scalability and security in mind, the system provides a reliable and efficient solution for personal, campus, or community safety management.

ADVANTAGE OF PROPOSED SYSTEM:

The Safe Guardian Real-Time Alert System offers significant advantages by addressing critical gaps in current safety and emergency communication frameworks. Its ability to provide immediate and precise alerts directly tackles the problem of delayed responses, which can be detrimental in emergencies. By integrating data from diverse sources, including sensors and external APIs, Safe Guardian delivers a comprehensive view of potential threats, minimizing the risk of incomplete or fragmented information. The system's user-friendly interfaces, both on mobile and web platforms, enhance accessibility and ensure that users can quickly act upon alerts. Additionally, its scalable architecture supports growing user bases and data volumes, making it adaptable to various settings, from personal safety to large-scale community protection. This proactive approach not only improves response times and situational awareness but also contributes to overall safety by streamlining communication and enabling more effective emergency management.

CHAPTER 5

HARDWARE & SOFTWARE REQUIREMENTS

CHAPTER 5

HARWARE & SOFTWARE REQUIREMENTS

5.1 Hardware Requirements

5.1.1. Arduino UNO

5.1.2. GSM Module 900A

5.1.3. GPS Neo M6

5.1.4. 4 ohm Speaker

5.1.5. Microphone

5.1.6. Accelerometer

5.1.7. Push button

5.1.8. 7V Power supply (lithium ion 18650 battery)

5.1.9. Jumper wires

5.2 Software Requirements

5.2.1. Installation of code to Arduino UNO

5.1.1. Arduino UNO

Fig 5.1.1. Arduino UNO

The Arduino UNO is an open-source microcontroller board based on the Microchip ATmega328P microcontroller (MCU) and developed by Arduino.cc and initially released in 2010.The microcontroller 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 I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.It can be powered by a USB cable or a barrel connector that accepts voltages between 7 and 20 volts, such as a rectangular 9-volt battery.

5.1.2. GSM Module 900A

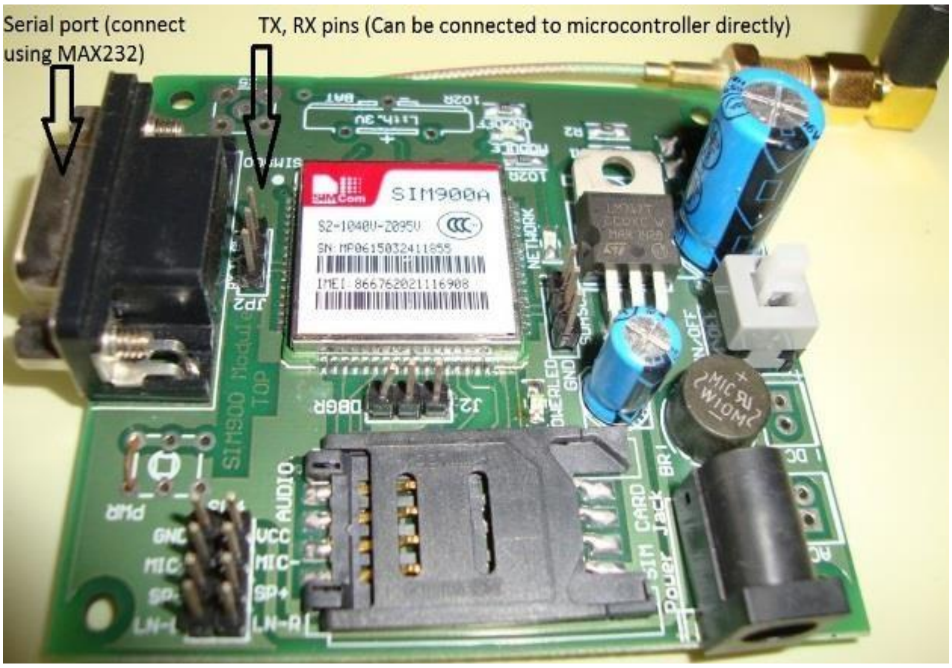

Fig 5.1.2. GSM Module 900A

This is a GSM/GPRS-compatible Quad-band cell phone, which works on a frequency of 850/900/1800/1900MHz and which can be used for various applications such as access the Internet, make a voice call, send and receive SMS, etc. The frequency bands of the GSM modem can be set by AT Commands. The baud rate is configurable from 1200-115200 through AT command. The GSM/GPRS Modem is having an internal TCP/IP stack which enables us to connect with the internet via GPRS. This is an SMT type module and designed with a very powerful single-chip processor integrating AMR926EJ-S core, which is very popular in various industrial products.

Technical Specifications:

- Supply voltage: 3.4V – 4.5V.
- Sleep Mode power consumption: 0.5mA.
- Frequency bands: Dual-band SIM900A (EGSM900, DCS1800).
- Operating temperature: -30ºC to +80ºC.
- Supports MIC, audio input, and speaker.
- UART interface support with firmware upgrade via debug port.

5.1.3. GPS Neo M6

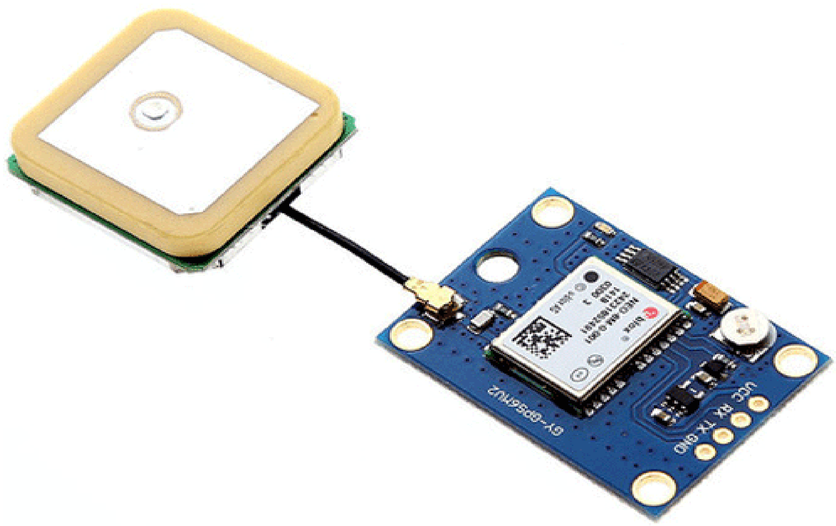

Fig 5.1.3. GPS Neo M6

The NEO-6M GPS module is a popular GPS receiver with a built-in ceramic antenna, offering strong satellite search capabilities. It can track up to 22 satellites globally and features an on-board signal indicator to monitor network status. The module includes a backup battery to retain data if power is lost.

The NEO-6M chip by u-blox, the core of the module, supports 50 channels with a sensitivity of -161 dBm and achieves a Time-To-First-Fix (TTFF) of under 1 second. It supports baud rates from 4800 to 230400 bps, with a default of 9600 bps.

Features:

- Operating voltage: 2.7-3.6V DC

- Operating current: 67 mA

- Baud rate: 4800-230400 bps (9600 default)

- Communication Protocol: NEMA

- Interface: UART

- External antenna with built-in EEPROM

GPS Module Pinout:

- VCC: Input voltage

- GND: Ground

- RX, TX: UART communication with Microcontroller

This GPS module has been used in various Arduino projects, including vehicle tracking.

Fig 5.1.3. Pin out of GPS Module

5.1.4. 4 Ohm Speaker

Fig 5.1.4. 4 ohm Speaker

An 4-ohm speaker is a common choice for home audio systems due to its standard impedance, which ensures compatibility with most amplifiers. This impedance rating provides a good balance between power handling and efficiency, allowing the speaker to produce clear and balanced sound with a typical home amplifier. Speakers with an 4-ohm impedance are versatile, often found in various audio setups including home theaters and stereo systems. When matched properly with an amplifier, they deliver reliable performance and can handle a broad range of frequencies, contributing to a rich listening experience.

5.1.5. Microphone

Fig 5.1.5. Microphone

A microphone captures sound waves and converts them into electrical signals, allowing you to record or transmit audio. For optimal listening, a good microphone should have a wide frequency response and low self-noise to accurately capture the nuances of sound. High-quality microphones are essential for clear voice recordings, whether for podcasts, streaming, or professional audio work. Directional microphones help focus on specific sound sources while minimizing background noise, enhancing overall audio clarity. Proper microphone placement and selection are crucial for achieving the best sound quality in any recording or communication setup.

5.1.6. Accelerometer

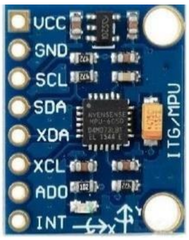

Fig 5.1.6. Accelerometer

An accelerometer is a sensor that measures acceleration forces, detecting changes in velocity and movement. In accident detection systems, accelerometers monitor sudden or abnormal changes in motion, such as rapid deceleration or impacts. When these unusual acceleration patterns are detected, the system can trigger alerts or automatic responses, such as notifying emergency services or deploying airbags. The sensitivity and accuracy of the accelerometer are crucial for distinguishing between normal driving conditions and genuine accidents. By integrating with vehicle safety systems, accelerometers enhance overall safety and response times in critical situations.

5.1.7. Push button

Fig 5.1.7. Push button

A push button for emergency detection is a simple yet effective safety tool designed to alert others during urgent situations. When pressed, the button sends an immediate signal to emergency services or a designated contact, ensuring a quick response. These buttons are often used in various settings, including homes, offices, and public spaces, to enhance safety and provide rapid assistance. They can be part of a larger safety system, such as a panic alarm or alert system, and may also be connected to automated responses like unlocking doors or activating security cameras. Their ease of use and reliability make them a crucial component in emergency preparedness. In addition to their primary function of alerting emergency services, push buttons for emergency detection can be customized to fit specific needs. They may include features such as silent alarms to avoid drawing attention or integration with mobile apps for notifications. Regular maintenance and testing are essential to ensure their functionality in critical moments. These devices also often come with user-friendly designs, allowing people of all ages and abilities to operate them effortlessly.

5.1.8. 7V Power supply(lithium ion 18650 battery)

Fig 5.1.8. 7V Power supply (lithium ion 18650 battery)

A 7-volt power supply using a lithium-ion 18650 battery provides a reliable and compact energy source for various electronic devices. The 18650 battery is known for its high energy density, long cycle life, and stable voltage output, making it suitable for portable and battery-powered applications. With a nominal voltage of 3.7 volts per cell, two 18650 cells in series can deliver a combined voltage of approximately 7.4 volts, which is close to the 7-volt requirement. This setup ensures a consistent power supply for devices requiring 7 volts. Proper management of charging and discharging is crucial to maximize battery life and maintain performance.

5.1.9. Jumper wire

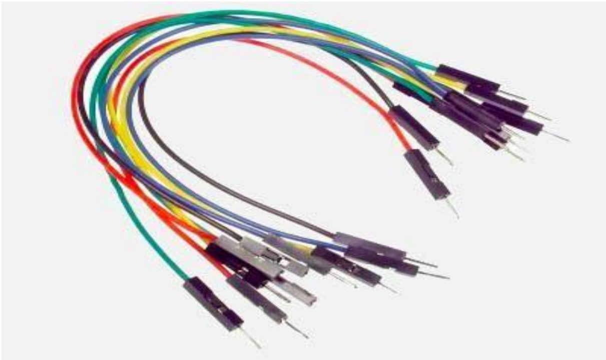

Fig 5.1.9. Jumper wires

Jumper wires are essential components in electronics and prototyping, used to establish connections between different points on a circuit board. They come in various colors and lengths, making it easy to organize and trace connections. Jumper wires are often employed in breadboards, where they facilitate the quick and flexible setup of electronic circuits for testing and development. They can be used for connecting components, transferring signals, or interfacing with microcontrollers and sensors. Available in male-to-male, female-to-female, and male-to-female configurations, jumper wires offer versatility for diverse project needs. Their ease of use and reusability make them invaluable tools for both hobbyists and professionals.

5.2. Software Requirements

Arduino software (IDE)

Arduino Software (IDE) makes it easy to write code and upload it to the board offline. We recommend it for users with poor or no internet connection. This software can be used with any Arduino board. There are currently two versions of the Arduino IDE, one is the IDE 1.x.x and the other the is IDE 2.x. The IDE 2.x is new major release that is faster and even more powerful to the IDE 1.x.x. In addition to a more modern editor and a more responsive interface it includes advanced features to help users with their coding and debugging.

5.2.1 Installation of code to Arduino UNO

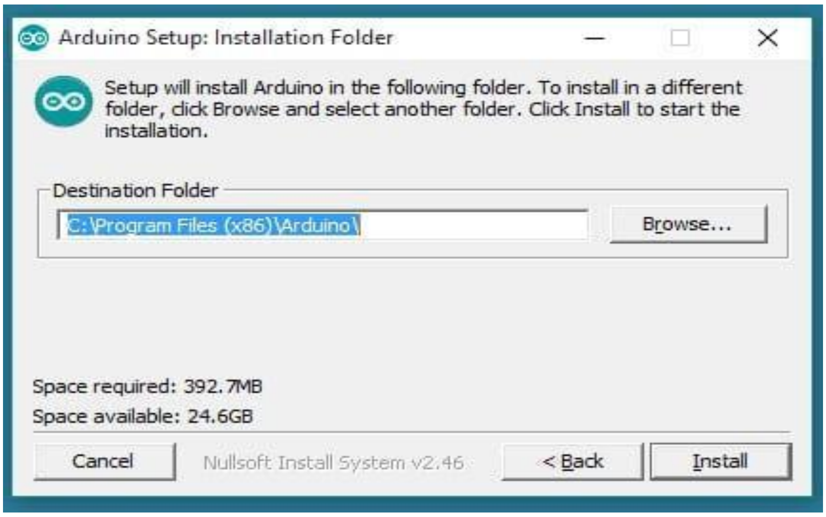

Fig 5.2.1. Installation of code to Arduino UNO

Get the latest version from the download page. You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a portable installation. When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system. The process will extract and install all the required files to execute properly the Arduino Software (IDE)

The text of the Arduino getting started guide is licensed under a Creative Commons Attribution-Share Alike 3.0 License. Code samples in the guide are released into the public domain.

CHAPTER 6

IMPLEMENTATION

CHAPTER 6

IMPLEMENTATION

6.1 Proposed Methodology

6.1.1 Block Diagram

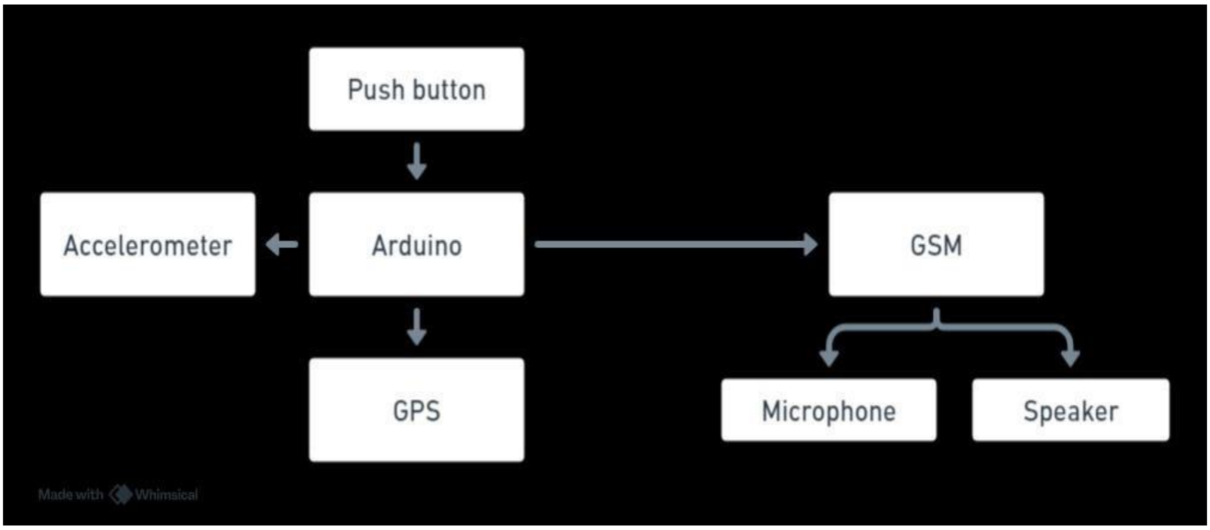

Fig 6.1.1. Block Diagram

The Long Press feature uses a button connected to an Arduino input pin, detecting a LOW signal when pressed. If held for over 5 seconds, the Arduino sends an SMS with the GPS location and makes a call to an emergency contact via the GSM module.

For the Short Press function, two presses within 5 seconds trigger the Arduino to send a confirmation SMS with the GPS location without making a call.

The Call Pickup Button, connected to another pin, sends an ATA command to the GSM module, answering incoming calls and enabling communication through the speaker and microphone.

The Accelerometer Sensor monitors movement across the X, Y, and Z axes. If a fall or significant impact is detected, the Arduino sends an emergency SMS with the GPS location and makes a call using the GSM module.

6.1.2 Flowchart

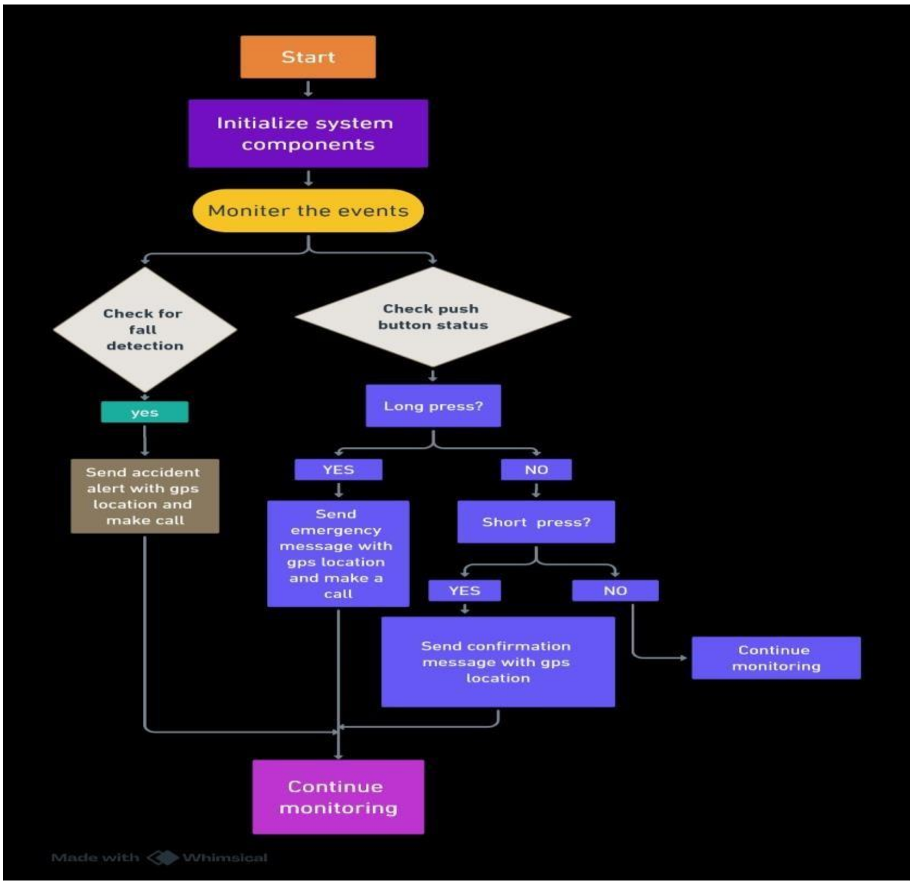

Fig 6.1.2. Flowchart

- Start: The system initializes all hardware components, including Arduino, GSM module, GPS module, accelerometer, push buttons, microphone, and speaker.
- Initialize Components: Arduino configures communication with the GSM module, initializes the GPS for location data, and calibrates the accelerometer.
- Monitor Events: The system continuously checks for events, reading data from the accelerometer, push buttons, and GPS module.
- Fall Detection: Analyzes accelerometer data for sudden movements or falls. If detected, the system checks the push button status. If no fall is detected, it continues monitoring.
- Push Button Status: Determines if the push button was long-pressed (single press) or short-pressed (double press).
- Accident Alert: If a fall is detected, the system sends an accident alert with GPS location and makes a call to the emergency contact.

- Long Press: If the button is held down, it sends an emergency message with GPS location and makes a call. If not, it checks for a short press.
- Short Press: If the button is pressed twice quickly, it sends a confirmation message with GPS location without making a call. If not, the system resumes monitoring.
- Continue Monitoring: After sending messages and making calls, the system returns to monitoring for fall detection and button presses.

6.1.3 Circuit Diagram

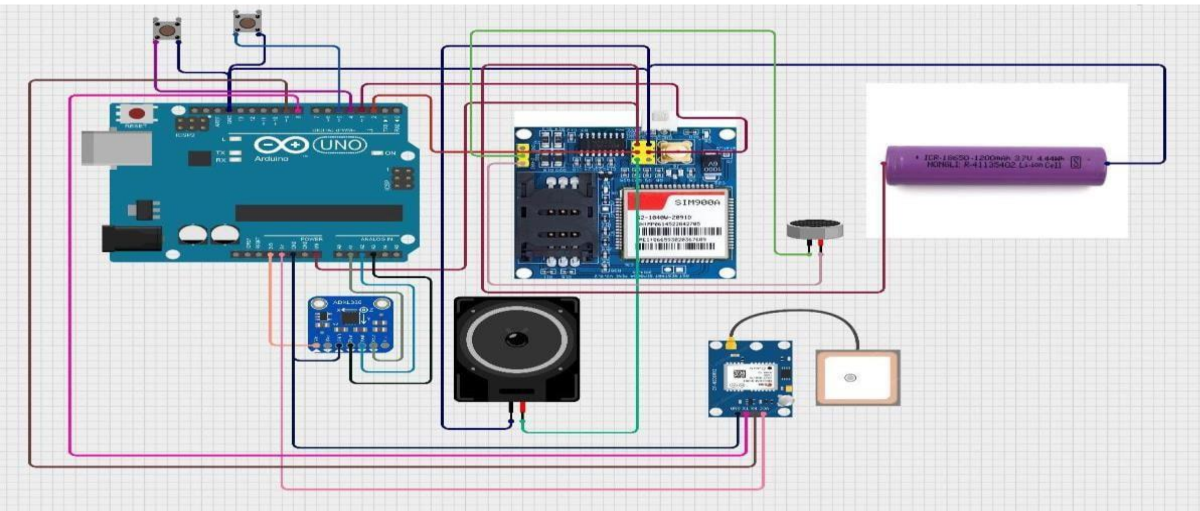

Fig 6.1.3 Circuit Diagram

- Power Supply: 12V 2A connected to the GSM module ensures stable operation during high current demands like calls or messages.
- GSM Module: TX to Pin 2 (Arduino) sends data to Arduino; RX to Pin 3 receives commands to control the GSM module.
- Push Buttons: Pin 4 detects a single button press. A long press sends an emergency message with GPS and initiates a call; a short press sends a confirmation message. Pin 5 is connected to another button for picking up incoming calls by sending an "ATA" command.
- Accelerometer: Pins A1, A2, A3 read X, Y, Z axis data to detect falls or significant movement, triggering an emergency alert if needed.
- GPS Module: TX to AltSoftSerial (Arduino) provides GPS data (latitude and longitude) for emergency and confirmation messages.

- Microphone and Speaker: Connected to GSM module (MCP, MCN, Sp, GND) for clear two-way audio communication during calls.
- Arduino Control: Pin 2 (RX) and Pin 3 (TX) manage communication with the GSM module. Analog Pins A1, A2, A3 monitor the accelerometer, while Digital Pins 4 and 5 detect button presses to send emergency or confirmation messages or answer calls.

CHAPTER 7

RESULTS AND ANALYSIS

CHAPTER 7

RESULTS AND ANALYSIS

7.1 Arduino IDE working image

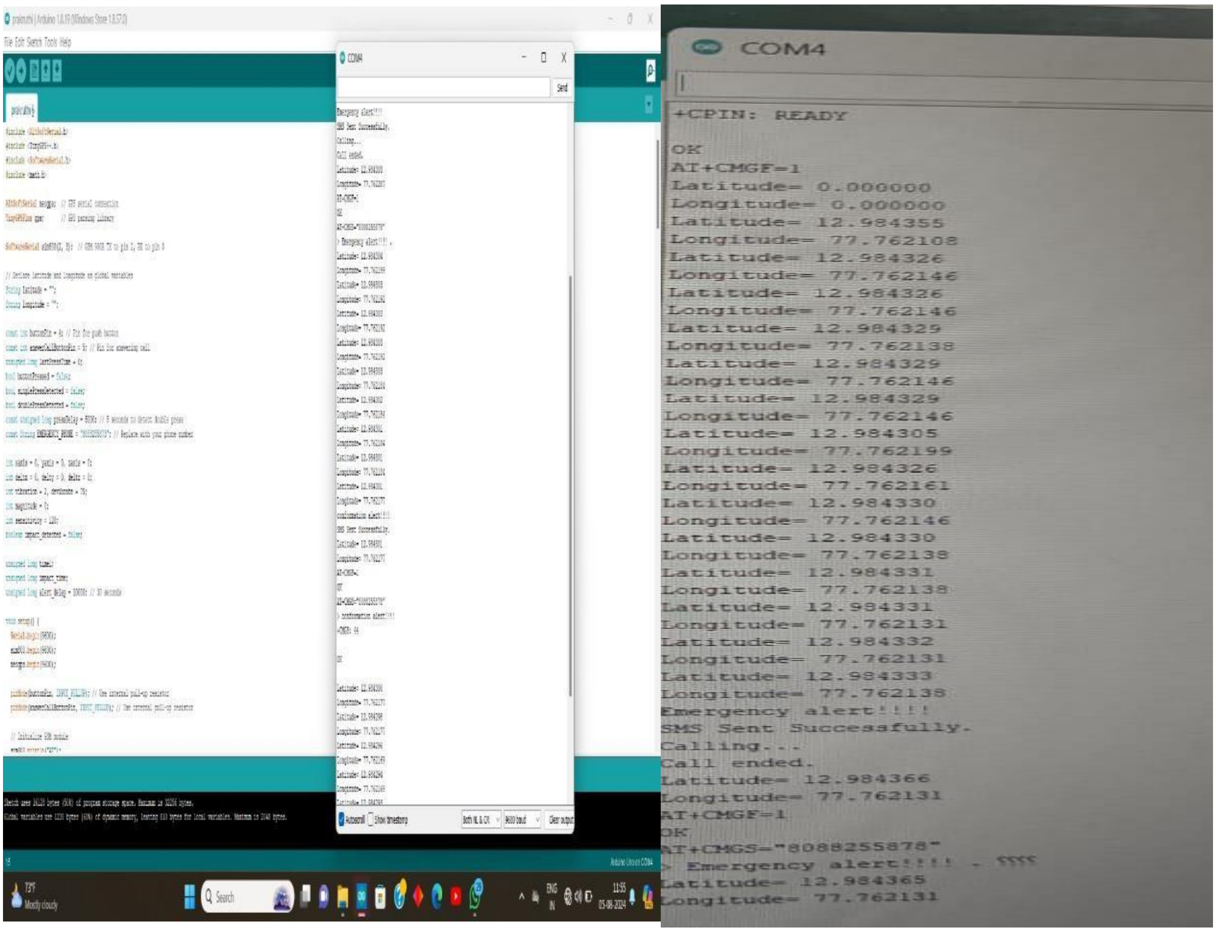

Fig 7.1.1. Arduino IDE working image

The Arduino IDE code for the "Safe Guardian Real-Time Alert System" starts by initializing libraries and setting up pins for components like the GSM module, push buttons, and accelerometer. It configures serial communication with `Serial.begin(9600)` for monitoring and `GSM Serial.begin(9600)` for GSM communication. In the `loop()` function, the system monitors button presses, accelerometer data, and GPS coordinates. A long press on the emergency button triggers an SMS with GPS location and an emergency call via AT commands like `AT+CMGS` and `ATD`. A short press sends a confirmation message with the GPS location. The code distinguishes between long and short presses by the duration the button is held. Another push button allows answering calls by sending the `ATA` command to the GSM module. The serial monitor displays real-time messages such as "Sending Emergency Message..." or "GPS Location: 12.9716, 77.5946", confirming system operations and aiding debugging. During initialization, it shows messages like "Initializing GSM Module..." and "Waiting for GPS signal...", indicating the setup process. As events like button presses or falls are detected, the monitor updates with corresponding messages, allowing users to track system actions and status.

7.2 Confirmation of code update

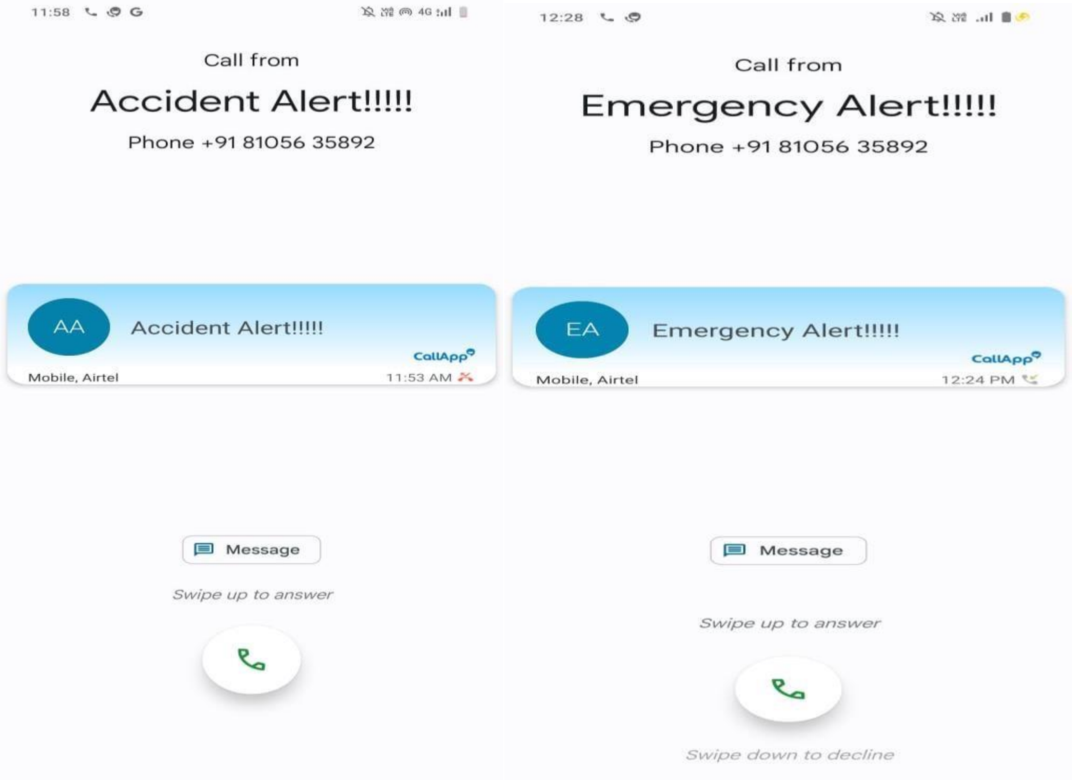

Fig 7.2.1. Phone Call

- Call Initiation: The GSM module sends a call request to the network using AT commands. This triggers the network to establish a connection with the designated phone number.
- Call Notification: Your phone receives a call notification, typically showing the caller's number or an identifier for the GSM module.
- Ringing: Your phone will ring to alert you of the incoming call from the GSM module. The display may show "Incoming Call" with the number provided by the GSM module.
- Answering: When you answer the call, the GSM module establishes a two-way communication channel, allowing you to converse with the person or system that

initiated the call.

- Audio Transmission: Audio from the GSM module is transmitted to your phone's speaker.

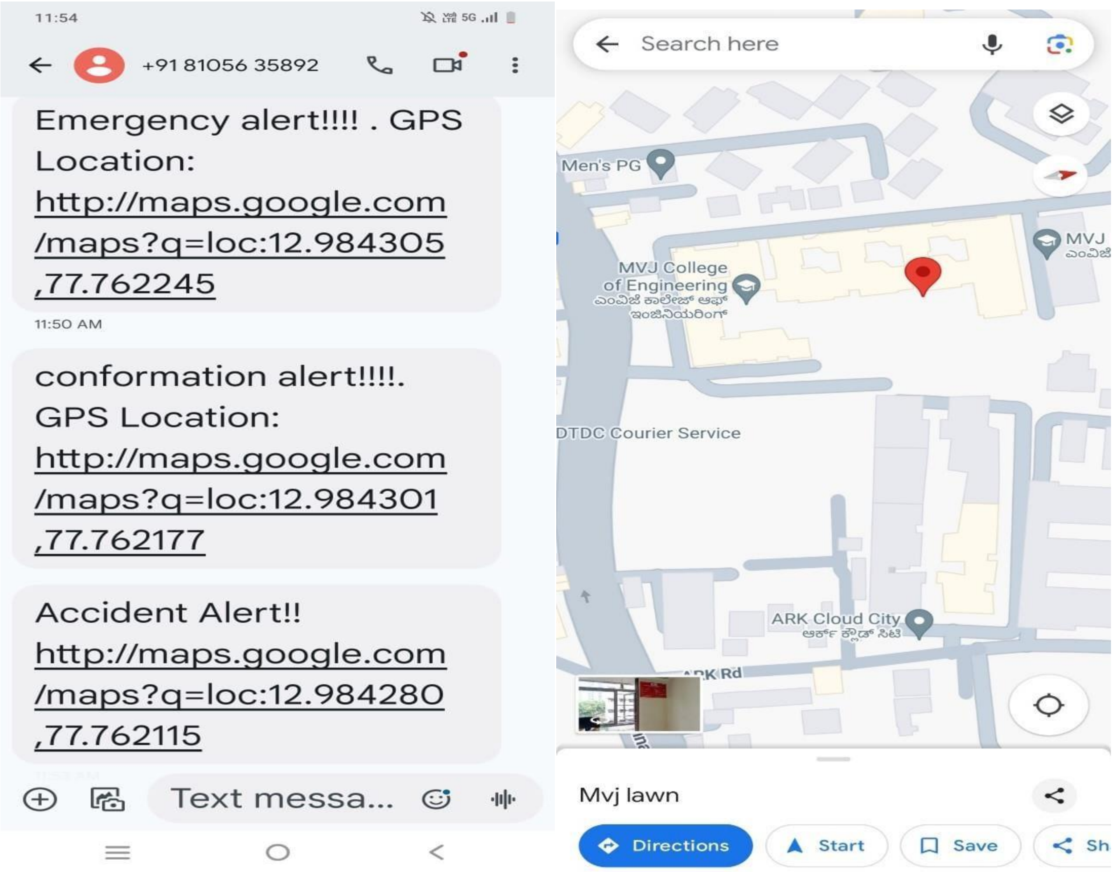

Fig 7.2.2. Message with GPS location

- Message Preparation: The Arduino formats an SMS containing text and GPS location data (latitude and longitude).
- Sending SMS: The GSM module transmits the SMS over the cellular network to the specified recipient.
- Message Receipt: The recipient's phone displays the message, which includes the current GPS coordinates.
- GPS Location Image: The message includes a link to a map or the raw GPS coordinates, allowing the recipient to view the exact location on a map application.
- Visualization: On receiving the SMS, the recipient can open the GPS link or coordinates on a mapping application to see the sender's precise location.

7.3 Final prototype

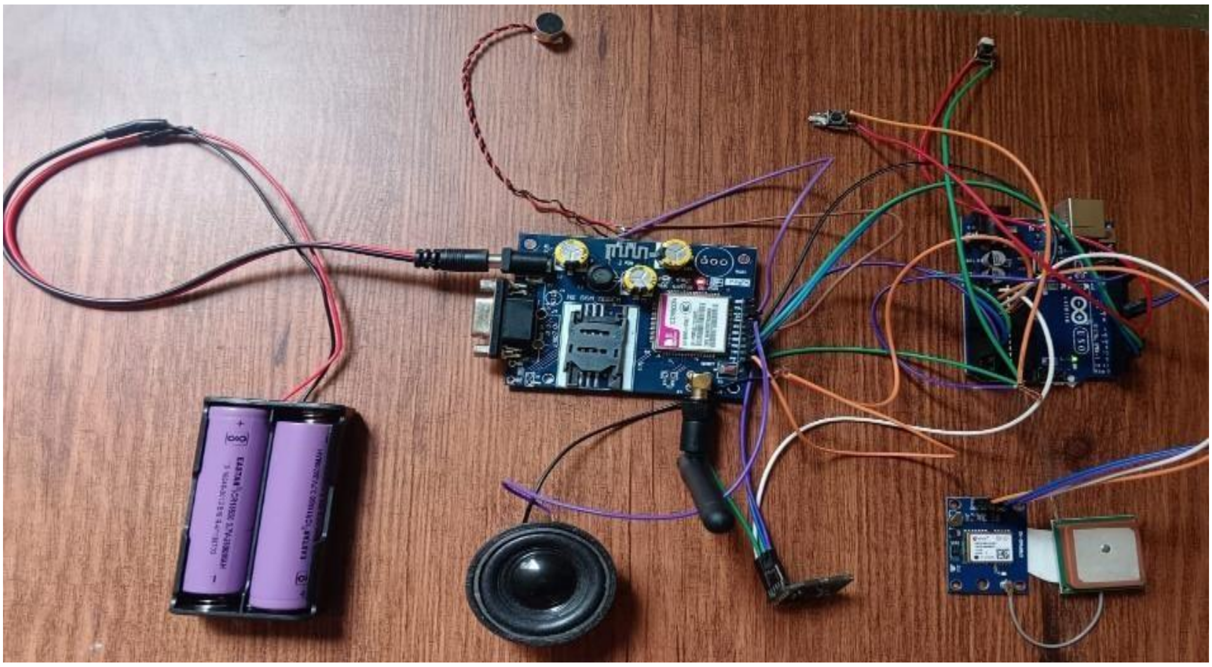

Fig 7.3.1. Final Prototype

1. **Long Press Button (Emergency Alert)**

Function: When the user presses the button once, the system immediately sends an emergency alert.

Working: The push button is connected to the Arduino, which detects the press. Upon detecting a single press, the Arduino triggers the GSM 900A module to send an SMS containing a pre-defined emergency message and the current GPS coordinates obtained from the GPS module. Simultaneously, the GSM module initiates a call to a designated emergency contact to ensure the alert is noticed promptly

2. **Short Press Button (Non-Urgent Update)**

Function: A double press sends a non-urgent update without making a call.

Working: The Arduino differentiates between single and double presses by measuring the time interval between presses. Upon detecting a double press, the Arduino instructs the GSM module to send an SMS with the current GPS location and a non-urgent message, such as confirming safe arrival or taking medication.

3. **Accident Detection**

Function: Automatically detects falls or accidents and sends an alert.

Working: The accelerometer continuously monitors the user’s movement and orientation. If a sudden change indicative of a fall or accident is detected (e.g., rapid deceleration or unusual angle), the Arduino processes this data and classifies it as an accident. The Arduino then commands the GSM module to send an SMS stating "Accident has occurred" along with the GPS location, and also initiates an "Accident Alert" call.

4. **Real-Time Monitoring**

Function: Continuously monitors the user’s status and location.

Working: The GPS module continuously updates the Arduino with the user’s current coordinates. The system can periodically send updates to a central monitoring system or caregivers, ensuring real-time tracking.

5. **Speaker and Microphone Integration**

Function: Allows two-way communication during emergencies.

Working: The GSM module, connected to a speaker and microphone, facilitates voice calls. During an emergency call initiated by the system, the speaker and microphone enable the user to communicate with the emergency contact, providing more details about their situation.

CHAPTER 8

CONCLUSION

CHAPTER 8

CONCLUSION

The Safe Guardian Real-Time Alert System is a significant advancement in personal safety for women, children, and the elderly, integrating cutting-edge technology and innovative features to address real-time safety concerns. By incorporating AI for predictive threat detection and seamless wearable device integration, users have constant access to protective measures and health monitoring.

Global coverage and enhanced geofencing will adapt to diverse regional needs and respond to high-risk areas. Streamlined communication with emergency services and multi-channel alerts ensure prompt assistance. Customizable safety profiles and smart home integration provide tailored protection and automated responses, while real-time health monitoring meets the specific needs of elderly users.

The system's reliance on crowdsourced data drives continuous improvement, and interactive safety maps and educational resources empower users with knowledge for effective navigation and emergency preparedness. Prioritizing privacy and data security maintains user trust, while community support networks foster a collaborative approach to safety.

In conclusion, the Safe Guardian Real-Time Alert System is designed to enhance personal safety through advanced sensors and communication technologies. It uses an accelerometer for fall detection, a GPS module for location data, and a GSM module for alerts and calls. Push buttons allow users to trigger emergency messages or confirmation alerts, and a separate button enables answering calls. The integration of these components ensures effective monitoring, critical event detection, and timely alerts, offering users immediate assistance and peace of mind in various situations.

CHAPTER 9

FUTURE ASPECTS

CHAPTER 9

FUTURE ASPECTS

The Safe Guardian Real-Time Alert System envisions transformative advancements in safety for women, children, and the elderly. Future versions will incorporate AI to predict threats based on behavior and context, and integrate with wearable technology for real-time health monitoring and instant alerts. The system will expand globally, adapting to regional emergency protocols and languages.

Enhanced geofencing will monitor locations precisely, triggering alerts in high-risk areas. Communication with emergency responders will be streamlined through encrypted channels, while customizable safety profiles allow users to set alert parameters based on personal routines.

Integration with smart home systems will enable automated safety responses like locking doors or activating alarms. For the elderly, features like real-time health monitoring and fall detection will ensure timely medical assistance. Crowdsourced data and user feedback will continually improve the system's algorithms and response protocols.

Multi-channel alerts via SMS, email, and push notifications will ensure timely communication, and direct interfacing with local emergency services will expedite responses. Interactive safety maps and educational resources will guide users in navigating safely and understanding system features.

Privacy and data security will be prioritized through robust encryption, and behavioral analytics will proactively address potential risks. Community support networks will connect users with local resources, fostering a collaborative approach to safety. Emergency preparedness features, voice activation, and continuous upgrades will ensure the system evolves with technological advancements, maintaining its relevance and effectiveness.

REFERENCES

1. Aggarwal, Deepti, et al. "An insight into android applications for safety of women: techniques and applications." *2022 IEEE Delhi section conference (DELCON)*. IEEE, 2022.

2. Brandl, Bonnie, et al. *Elder abuse detection and intervention: A collaborative approach*. Springer publishing company, 2006.

3. Mann, William C. "Smart technology for aging, disability, and independence." *The State of the Science* 2005 (2005).

4. Hughes, Donna M. "The use of new communications and information technologies for sexual exploitation of women and children." *Hastings Women's LJ* 13 (2002): 127.

5. Briggs, Freda. *Child protection: A guide for teachers and child care professionals*. Routledge, 2020.

6. Peterman, Amber, Alina Potts, Megan O'Donnell, Kelly Thompson, Niyati Shah, Sabine Oertelt-Prigione, and Nicole Van Gelder. *Pandemics and violence against women and children*. Vol. 528. Washington, DC: Center for Global Development, 2020.

7. Posetti, Julie, Nabeelah Shabbir, Diana Maynard, Kalina Bontcheva, and Nermine Aboulez. "The chilling: Global trends in online violence against women journalists." *New York: United Nations International Children's Emergency Fund (UNICEF)* (2021).

8. Landau, Ruth, and Shirli Werner. "Ethical aspects of using GPS for tracking

people with dementia: recommendations for practice." *International psychogeriatrics* 24.3 (2012): 358-366.

9. Hussain, Afzal, Anisha, Pallavi Murghai Goel, and Sudeshna Chakraborty. "Smart Child Safety Framework Using Internet of Things." *Decision Support Systems for Smart City Applications* (2022): 153-166.

10. Reddy, B. Ramesh, T. Sowjanya, N. Bala Subrahmanayam, G. Mahantesh, and S. Prudhvi. "IOT based smart protective equipment for women." *Materials Today: Proceedings* 80 (2023): 2895-2900.

11. Gao, Zhigang, Hongyi Guo, Yunfeng Xie, Yanjun Luo, Huijuan Lu, and Ke Yan. "ChildGuard: A child-safety monitoring system." *IEEE MultiMedia* 24, no. 4 (2017): 48-57.

12. Muli, Erick M. "Smart school bus system to enhance child transport safety using radio frequency identification (RFID) technology." PhD diss., University of Nairobi, 2018.

13. Mirunalini, A. R., S. Shwetha, R. Priyanka, and V. Berlin Hency. "IoT-Based Unified Child Monitoring and Security System." *Integrated Green Energy Solutions Volume 1* (2023): 241-261.

14. Albu, Gabriela. *Global positioning system: Parental view on efficacy of GPS tracking devices for children's movement*. Florida Gulf Coast University, 2016.

15. Kavitha, A., & Swaminathan, J. N. (2023). Design of IoT based human bond communication in smart foam nanocomposite sensor through in-situ polymerization method for female child safety. *Journal of Electrical Engineering & Technology*, *18*(6), 4389-4397.

APPENDIX

```
#include <AltSoftSerial.h>
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <math.h>

AltSoftSerial neogps;  // GPS serial connection
TinyGPSPlus gps;       // GPS parsing library

SoftwareSerial sim800(2, 3);  // GSM 900A TX to pin 2, RX to pin 3

// Declare latitude and longitude as global variables
String latitude = "";
String longitude = "";

const int buttonPin = 4; // Pin for push button
const int answerCallButtonPin = 5; // Pin for answering call
unsigned long lastPressTime = 0;
bool buttonPressed = false;
bool singlePressDetected = false;
bool doublePressDetected = false;
const unsigned long pressDelay = 5000; // 5 seconds to detect double press
const String EMERGENCY_PHONE = "8088255878"; // Replace with your phone number

int xaxis = 0, yaxis = 0, zaxis = 0;
int deltx = 0, delty = 0, deltz = 0;
int vibration = 2, devibrate = 75;
int magnitude = 0;
int sensitivity = 20;
boolean impact_detected = false;

unsigned long time1;
unsigned long impact_time;
unsigned long alert_delay = 30000; // 30 seconds

void setup() {
  Serial.begin(9600);
  sim800.begin(9600);
  neogps.begin(9600);

  pinMode(buttonPin, INPUT_PULLUP); // Use internal pull-up resistor
  pinMode(answerCallButtonPin, INPUT_PULLUP); // Use internal pull-up resistor

  // Initialize GSM module
  sim800.println("AT");
  delay(1000);
  sim800.println("ATE1");
```

```
  delay(1000);
  sim800.println("AT+CPIN?");
  delay(1000);
  sim800.println("AT+CMGF=1");
  delay(1000);
  sim800.println("AT+CNMI=1,1,0,0,0");
  delay(1000);

  time1 = micros();

  xaxis = analogRead(A1);
  yaxis = analogRead(A2);
  zaxis = analogRead(A3);
}

void loop() {
  detectButtonPress();
  detectAnswerCallButtonPress();

  if (singlePressDetected) {
    Serial.println("Single press detected");
    handleSinglePress();
    singlePressDetected = false; // Reset flag
  }

  if (doublePressDetected) {
    Serial.println("Double press detected");
    handleDoublePress();
    doublePressDetected = false; // Reset flag
  }

  if (micros() - time1 > 1999) {
    Impact();
  }

  if (impact_detected == true) {
    if (millis() - impact_time >= alert_delay) {
      sendAlert(); // Send SMS with GPS location
      delay(5000); // Wait for 5 seconds before making the call
      makeCall(); // Make the call
      impact_detected = false;
      impact_time = 0;
    }
  }

  // Handle GPS data collection
  getGps();

  // Handle any incoming SMS messages or other GSM operations here if needed
```

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

  while (Serial.available()) {
    sim800.println(Serial.readString());
  }
}

void detectButtonPress() {
  bool currentButtonState = digitalRead(buttonPin) == LOW; // Active LOW

  if (currentButtonState && !buttonPressed) {
    // Button was just pressed
    buttonPressed = true;
    lastPressTime = millis();
  }
  else if (!currentButtonState && buttonPressed) {
    // Button was just released
    buttonPressed = false;
    unsigned long pressDuration = millis() - lastPressTime;

    if (pressDuration < pressDelay) {
      // Double press detected
      doublePressDetected = true;
    }
    else {
      // Single press detected
      singlePressDetected = true;
    }
  }
}

void detectAnswerCallButtonPress() {
  if (digitalRead(answerCallButtonPin) == LOW) {
    answerCall();
  }
}

void handleSinglePress() {
  String message = "Single press detected. GPS Location: ";
  sendSmsWithLocation(message);
  delay(5000); // Wait for 5 seconds before making the call
  makeCall();
}

void handleDoublePress() {
  String message = "Double press detected. GPS Location: ";
  sendSmsWithLocation(message);
```

```
}

void sendSmsWithLocation(String text) {
  String smsData = text;
  smsData += "http://maps.google.com/maps?q=loc:";
  smsData += latitude + "," + longitude;
  sendSms(smsData);
}

void sendSms(String text) {
  sim800.print("AT+CMGF=1\r");
  delay(1000);
  sim800.print("AT+CMGS=\"" + EMERGENCY_PHONE + "\"\r");
  delay(1000);
  sim800.print(text);
  delay(100);
  sim800.write(0x1A); // End of SMS
  delay(1000);
  Serial.println("SMS Sent Successfully.");
}

void makeCall() {
  Serial.println("Calling...");
  sim800.println("ATD" + EMERGENCY_PHONE + ";");
  delay(20000); // 20 seconds delay for the call
  sim800.println("ATH"); // Hang up
  Serial.println("Call ended.");
}

void answerCall() {
  Serial.println("Answering call...");
  sim800.println("ATA");
  delay(1000); // 1 second delay for answering the call
  Serial.println("Call answered.");
}

void getGps() {
  boolean newData = false;
  for (unsigned long start = millis(); millis() - start < 2000;) {
    while (neogps.available()) {
      if (gps.encode(neogps.read())) {
        newData = true;
        break;
      }
    }
  }

  if (newData) {
    latitude = String(gps.location.lat(), 6);
```

```
    longitude = String(gps.location.lng(), 6);
    Serial.print("Latitude= "); Serial.println(latitude);
    Serial.print("Longitude= "); Serial.println(longitude);
  }
  else {
    Serial.println("No GPS data is available");
    latitude = "";
    longitude = "";
  }
}

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") {
      index = buff.indexOf(",");
      String temp = buff.substring(index + 1, buff.length());
      temp = "AT+CMGR=" + temp + "\r";
      sim800.println(temp);
    }
    else if (cmd == "+CMGR") {
      if (buff.indexOf(EMERGENCY_PHONE) > 1) {
        buff.toLowerCase();
        if (buff.indexOf("get gps") > 1) {
          getGps();
          String sms_data;
          sms_data = "GPS Location Data\r";
          sms_data += "http://maps.google.com/maps?q=loc:";
          sms_data += latitude + "," + longitude;

          sendSms(sms_data);
        }
      }
    }
  }
}
```

```
void   Impact()   {
  time1 = micros();

  int oldx = xaxis;
  int oldy = yaxis;
  int oldz = zaxis;

  xaxis = analogRead(A1);
  yaxis = analogRead(A2);
  zaxis = analogRead(A3);

  vibration--;
  if (vibration < 0) vibration = 0;

  if (vibration > 0) return;

  deltx = xaxis - oldx;
  delty = yaxis - oldy;
  deltz = zaxis - oldz;

  magnitude = sqrt(sq(deltx) + sq(delty) + sq(deltz));

  if (magnitude >= sensitivity) {
    Serial.print("Impact detected!! Magnitude: ");
    Serial.println(magnitude);
    impact_detected = true;
    impact_time = millis();
  } else {
    magnitude = 0;
  }
}

void sendAlert() {
  String sms_data;
  sms_data = "Accident Alert!!\r";
  sms_data += "http://maps.google.com/maps?q=loc:";
  sms_data += latitude + "," + longitude;

  sendSms(sms_data);
}
```