NANDHA ENGINEERING COLLEGE

ERODE–638052 (Autonomous)

(Affiliated to Anna University, Chennai)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

22AIC14 – INTERNET OF THINGS AND ITS APPLICATIONS

MINI PROJECT REPORT ON

TOPIC – LIFELINE DIALER

Submitted by

REGISTER NUMBER	NAME
22AI006	BHAVIYASHREE M
22AI033	PAVITHRA P
22AI058	YATHIGAA T S

NANDHA ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

This is to certify that the project work entitled "LIFELINE DIALER" is the Bonafide work of BHAVIYASHREE M(22AI006),PAVITHRA P(22AI033), YATHIGAA T S(22AI058) who carried out the work under my supervision.

Signature of the Supervisor
Dr. K. Lalitha,
Professor,
Department of AI & DS,
Nandha Engineering College,
Erode – 638052.

Signature of the HOD
Dr. P. Karunakaran,
Head of the Department,
Department of AI & DS,
Nandha Engineering College,
Erode – 638052.

Submitted for End semester PBL review held on _____

LIFELINE DIALER

AIM:

To develop a simple and efficient emergency communication system that allows users to make quick calls to predefined contacts with the press of a button, ensuring safety and accessibility.

SCOPE:

The project provides an affordable and user-friendly solution for emergency communication, particularly for the elderly and differently-abled. It ensures ease of operation, reliability, and flexibility, with scope for future enhancements like SMS alerts and GPS integration

BRIEF HISTORY:

Emergency dialing systems have evolved significantly, starting with basic landline emergency services. With the advancement of mobile technology, emergency calling systems became more accessible through mobile phones. The concept of Lifeline Dialers emerged as a solution to provide quick access to emergency contacts for individuals with limited mobility or technical expertise. Over time, these devices have been enhanced with features like predefined numbers and customizable buttons to cater to a wide range of users, particularly the elderly and differently-abled, ensuring their safety in critical situations.

PROPOSED METHODOLOGY:

The Lifeline Dialer project follows a simple, efficient, and cost-effective approach to ensure quick emergency communication. The proposed methodology is as follows**:**

Hardware Design:

The system is built using key components like the Node MCU ESP32 microcontroller, push buttons, LCD, SIM800L GSM module, and a step-down transformer for power regulation.

Button Configuration:

The system is designed with multiple push buttons, each linked to a predefined contact number. When a button is pressed, the corresponding number is dialed using the GSM module.

Microcontroller Programming:

The Node MCU ESP32 is programmed to handle button inputs, initiate calls through the GSM module, and display relevant information on the LCD.

Power Supply:

A step-down transformer is used to supply the required voltage to the circuit, ensuring stable and continuous operation.

Testing and Debugging:

The system will undergo rigorous testing to ensure seamless functionality, such as making calls and displaying the correct information on the LCD.

This methodology emphasizes simplicity, functionality, and ease of use, aiming to provide a reliable emergency dialing system for users with minimal technical expertise**.**

COMPONENTS REQUIRED:

S. No	Component	Quantity
1.	Node MCU ESP32 Microcontroller	1
2.	GSM Module (SIM800L)	1
3.	Step-down Transformer (12V to 5V)	1
4.	LCD Display (16x2)	1
5.	Push Buttons	4
6.	Jumper Wires	As required

7.	**PCB (for circuit assembly)**	**1**
8.	**Power Supply (5V, for ESP32)**	**1**
9.	**Connecting Cables**	**As required**

DESCRIPTION:

The components used in the Lifeline Dialer project:

Node MCU ESP32 Microcontroller: The Node MCU ESP32 is the heart of the system. It is a low-cost microcontroller with built-in Wi-Fi and Bluetooth capabilities, which allows it to handle button inputs and control other components like the GSM module. It is programmed to detect button presses, send signals, and display output on the LCD screen.

GSM Module (SIM800L): The GSM module is used to send and receive SMS and initiate voice calls. It connects to the microcontroller (Node MCU) and is responsible for dialing emergency numbers when a button is pressed. It also requires a SIM card for connectivity to the cellular network.

Step-down Transformer (12V to 5V): The step-down transformer is used to convert high voltage (12V) to the required low voltage (5V) to power the system. The transformer ensures the proper voltage supply for components like the microcontroller and GSM module.

LCD Display (16x2): The 16x2 LCD display is used to provide real-time feedback to the user. It displays the status of the dialer, such as which number is being dialed and whether the call is connected. This helps the user understand the system's operation.

Push Buttons: The system includes four push buttons. Each button is connected to the microcontroller and is assigned to dial a specific number. The user can press a button to initiate an emergency call to a predefined contact.

Jumper Wires: Jumper wires are used to make all the necessary electrical connections between the components, including the microcontroller, GSM module, buttons, and the LCD display.

PCB (Printed Circuit Board): The PCB is used to securely mount and connect the components. It organizes the electrical connections in a compact form, allowing for a neat and stable construction of the circuit.

Power Supply (5V): The power supply ensures that the system receives the correct voltage. The Node MCU and GSM module both require 5V, and a reliable power source is crucial for the proper functioning of the system.

Battery (for Backup): A battery is included in the design for backup power. In case of a power failure, the battery ensures that the system remains operational, especially for emergency dialing functions.

Capacitors & Resistors: Capacitors and resistors are used for stabilizing the voltage, reducing noise, and protecting the components from excessive current. They ensure smooth operation and prevent electrical damage.

Connecting Cables: Connecting cables are used to link the various components together. These cables provide the necessary connections for data transfer and power distribution across the system.

Each of these components plays a critical role in ensuring that the Lifeline Dialer works efficiently and reliably in emergency situations.

PROTOCOLS

The Lifeline Dialer project utilizes several protocols to ensure smooth and reliable communication between its components. These include the AT Command Protocol, GPIO Communication, and Serial Communication Protocol (UART), all of which are essential for the project's functionality.

The AT Command Protocol is primarily used to communicate with the GSM module, enabling the microcontroller to initiate calls. AT commands, such as ATD<number>; are sent to the GSM module to perform specific actions like dialing a phone number. This protocol is both simple and effective for controlling GSM-based systems, making it an ideal choice for the Lifeline Dialer.

The GPIO Communication Protocol handles the interaction between the buttons and the microcontroller. When a button is pressed, the GPIO pins read the input signals and trigger the microcontroller to execute the appropriate action, such as dialing a preconfigured contact. This direct communication ensures quick and accurate responses to user input.

The Serial Communication Protocol (UART) is used for data transmission between the microcontroller and the GSM module. By utilizing TX (transmit) and RX (receive) pins, the microcontroller sends AT commands and receives responses from the GSM module. This protocol is highly reliable and commonly used in microcontroller-based projects.

CODING:

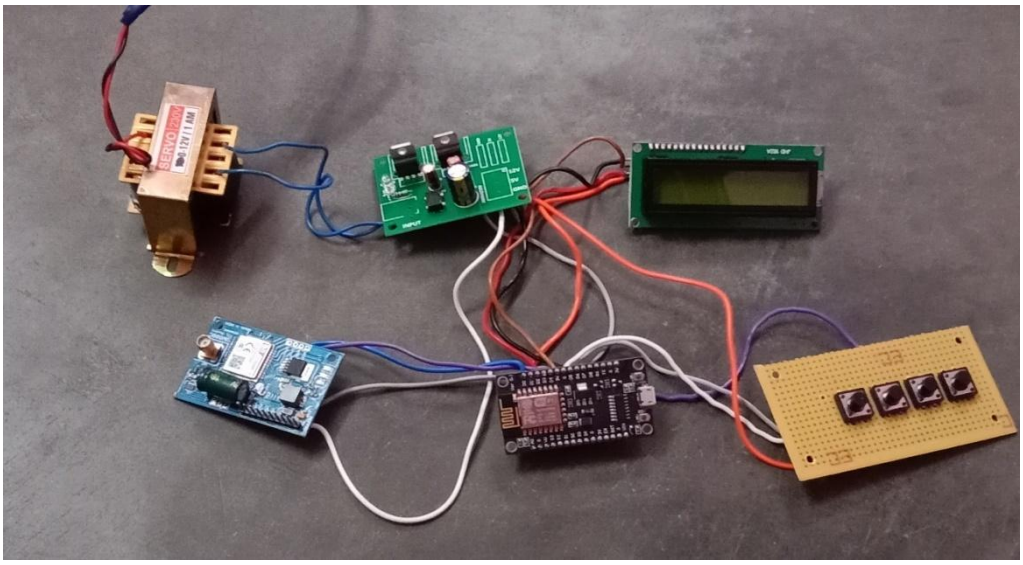
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>

// Define pin connections for GSM
#define SIM800_RX D0  // ESP8266 D0 pin connected to GSM module's RX pin
#define SIM800_TX D3  // ESP8266 D3 pin connected to GSM module's TX pin

// Define button pins
#define BUTTON1 D5    // Button to call +919514740046
#define BUTTON2 D6    // Button to call +919489550344
#define BUTTON3 D7    // Button to call +919514740046

// Define the phone numbers
const char phoneNumber1[] = "+916374822646";  // First phone number
const char phoneNumber2[] = "+919345718779";  // Second phone number

SoftwareSerial gsmSerial(SIM800_RX, SIM800_TX);  // RX, TX for software serial

// Initialize the LCD, set the I2C address of your display (e.g., 0x27 or 0x3F)
LiquidCrystal_I2C lcd(0x27, 16, 2);  // I2C address 0x27, 16 columns, 2 rows
```

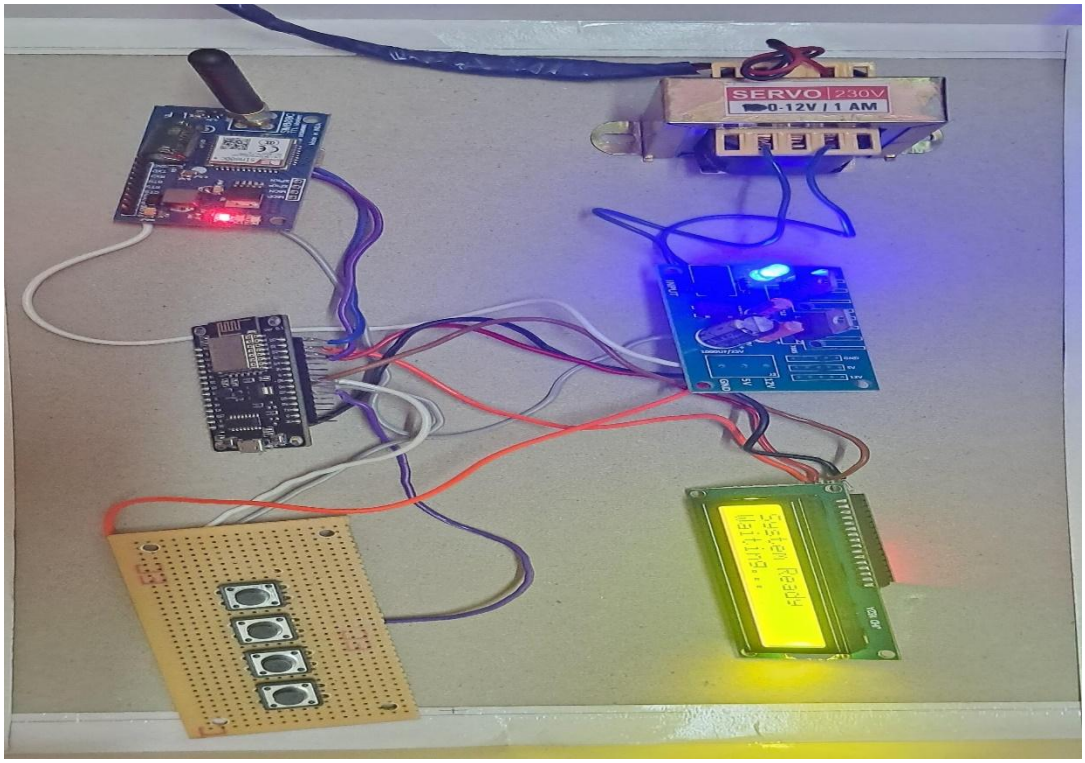
```
void setup() {
  // Initialize serial communication with SIM800C
  gsmSerial.begin(9600);  // SIM800C works best at 9600 baud rate
  Serial.begin(115200);   // For debug purposes, to monitor on Serial Monitor

  // Setup buttons as input with internal pull-up resistors
  pinMode(BUTTON1, INPUT_PULLUP);
  pinMode(BUTTON2, INPUT_PULLUP);
  pinMode(BUTTON3, INPUT_PULLUP);

  // Initialize the LCD and print a welcome message
  lcd.init();
  lcd.backlight();  // Turn on the backlight
  lcd.clear();

  lcd.setCursor(0, 0);  // Set cursor to the first column and row
  lcd.print("System Ready");
  lcd.setCursor(0, 1);
  lcd.print("Waiting...");

  Serial.println("System ready. Press button to make a call.");
}

void loop() {
  // Check if Button 1 is pressed (D5)
  if (digitalRead(BUTTON1) == LOW) {
    Serial.println("Calling number 1 from Button 1...");
    lcd.clear();
    lcd.setCursor(0, 0);
```

```
  lcd.print("Calling...");
  lcd.setCursor(0, 1);
  lcd.print(phoneNumber1);  // Display number being called
  makeCall(phoneNumber1);
  delay(10000);  // Debounce delay
  displayReadyMessage();  // Display "System Ready" after call
}

// Check if Button 2 is pressed (D6)
if (digitalRead(BUTTON2) == LOW) {
  Serial.println("Calling number 2 from Button 2...");
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Calling...");
  lcd.setCursor(0, 1);
  lcd.print(phoneNumber2);  // Display number being called
  makeCall(phoneNumber2);
  delay(10000);  // Debounce delay
  displayReadyMessage();  // Display "System Ready" after call
}

// Check if Button 3 is pressed (D7)
if (digitalRead(BUTTON3) == LOW) {
  Serial.println("Calling number 1 from Button 3...");
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Calling...");
  lcd.setCursor(0, 1);
  lcd.print(phoneNumber1);  // Display number being called
```

```
    makeCall(phoneNumber1);  // Call the first phone number again
    delay(10000);  // Debounce delay
    displayReadyMessage();  // Display "System Ready" after call
  }
}

// Function to make a call using the SIM800C
void makeCall(const char* phoneNumber) {
  gsmSerial.println("ATD" + String(phoneNumber) + ";");  // Dial the number
  Serial.println("Dialing: " + String(phoneNumber));
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Dialing:");
  lcd.setCursor(0, 1);
  lcd.print(phoneNumber);  // Display the number being dialed

}

// Function to display "System Ready" after the call ends
void displayReadyMessage() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("System Ready");
  lcd.setCursor(0, 1);
  lcd.print("Waiting...");
}
```

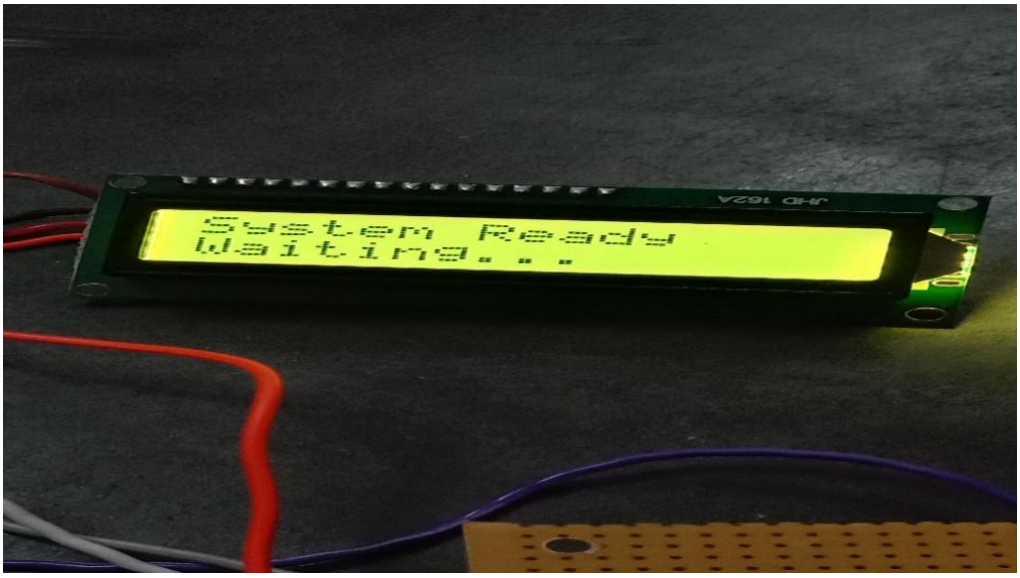
SCREENSHOTS:

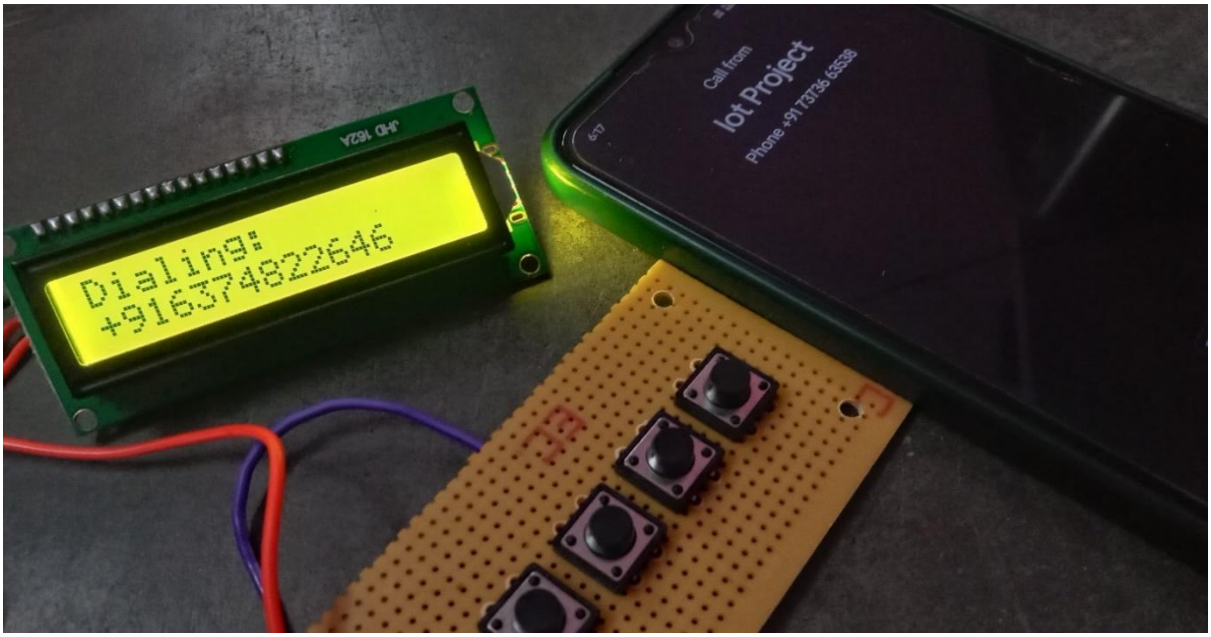




OUTPUTS:

Lifeline Dialer System: Dialing and Call Status





LIMITATIONS:

1. **Limited Functionality**: The system can only make calls to pre-programmed numbers and lacks flexibility for dynamic input.
2. **No Feedback Mechanism**: Lacks confirmation for successful call connection or notification if the call fails.
3. **No Internet Integration**: Cannot send messages, notifications, or provide additional services like real-time location sharing due to the absence of internet connectivity
4. **Non-Wearable Design**: The current design may not be portable or suitable for users needing constant accessibility, such as elderly or disabled individuals.
5. **No User Authentication**: Anyone can press the buttons to make a call, which might lead to misuse or unauthorized calls.

FUTURE ENHANCEMENTS:

1. **Integration with IoT**: Connect the dialer to a mobile app or cloud platform to allow remote access and monitoring of call logs or dialing activity.
2. **Emergency Alert System**: Add a feature to send automated SMS alerts or notifications to emergency contacts when a specific button is pressed.
3. **Voice Assistance**: Implement voice commands to trigger calls instead of using buttons, making it more user-friendly for elderly or differently-abled individuals.
4. **GPS Tracking**: Include a GPS module to send the location of the user during emergency calls for better assistance.
5. **Multiple Contact Storage**: Expand functionality to store and manage a list of contacts, allowing the user to choose whom to call dynamically.

6. **Fall Detection**: Use sensors to detect a fall and trigger an automatic call to emergency contacts.
7. **Dual Communication Modes**: Support text-based communication (e.g., sending pre-defined emergency messages) alongside calls.
8. **Compact Design**: Redesign the hardware to make it smaller, wearable (like a wristband), or more portable.

CONCLUSION:

Thus, the Lifeline Dialer system successfully allows users to initiate emergency calls with just a press of a button, providing a quick and reliable way to contact predefined emergency numbers. The system uses the Node MCU ESP32, GSM module, and LCD to ensure seamless communication during critical situations. With the integration of various components, such as the step-down transformer and push buttons, the system is simple, effective, and easy to use, ensuring safety and peace of mind for the user.