HOME SAFE - HOME SECURITY SYSTEM

by

Jaisuraj Bantupalli	20BLC1005
Sanjukta Roy	20BLC1026
Ashwin Santhosh Nair	20BLC1036

A project report submitted to

Dr. SOFANA REKA S

SCHOOL OF ELECTRONICS ENGINEERING

in partial fulfilment of the requirements for the course of

ECE4003 – EMBEDDED SYSTEM DESIGN

in

B. Tech. ELECTRONICS AND COMPUTER ENGINEERING

Vandalur – Kelambakkam Road
Chennai – 600127

NOVEMBER 2022

BONAFIDE CERTIFICATE

Certified that this project report entitled "**HOME SECURITY SYSTEM"** is a bonafide work of **Jaisuraj Bantupalli - 20BLC1005, Sanjukta Roy - 20BLC1026** and **Ashwin Santhosh Nair - 20BLC1036** who carried out the Project work under my supervision and guidance for **ECE4003 – EMBEDDED SYSTEM DESIGN.**

Dr. SOFANA REKA S

Assistant Professor

School of Electronics Engineering (SENSE),

VIT University, Chennai

Chennai – 600 127.

ABSTRACT

In today's world, safety is the most vital requirement. People need to feel safe in the environment they are in. With rapid advancement in science, it only seems logical that there is also an advancement in how we deal with security in our living spaces. The days of using locks and bolts alone to provide home security are long gone. The main goal of the proposed 'Home Security System' is to give users an easy-to-use and affordable alarm protection system for their homes so they can make their environment safer. This might entail enhanced monitoring for a house, place of business, or neighbourhood, according to the requirements of the user. The system uses the NUCLEO-L4R5ZI development board along with other components such as an ultrasonic sensor, a microphone, a keypad, a 16x2 LCD unit, an active buzzer and LEDs. The system is first armed by the user by entering a passcode into the system using the keypad system. The LCD screen acts as the user interface which displays all the appropriate messages for each scenario. The ultrasonic sensor and microphone work to detect any loud noises or presence of intruder based on the proximity respectively so as to detect intruders. The buzzer rings immediately and LED glows in a certain pattern, when intrusion is detected so as to alert the user.

Keywords: Home security system, Security, Intruder detection.

ACKNOWLEDGEMENT

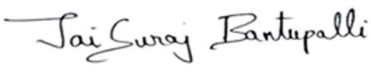
We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Sofana Reka S**, Assistant Professor Senior (Grade 2), School of Electronics Engineering, for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Susan Elias**, Dean of School of Electronics Engineering (SENSE), VIT Chennai, for extending the facilities of the School towards our project and for her unstinting support.

We express our thanks to our Head of the Department, **Dr. Jaya Vignesh T** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.



Jaisuraj Bantupalli **Sanjukta Roy** **Ashwin Santhosh Nair**

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 OBJECTIVES

The main objectives of this project are as described below:

- The purpose of this system is to provide an affordable and user-friendly home security system which consumers can use to create a safer space. This could mean having a better protected home or any area chosen by the user, in the form of enhanced monitoring of their environment.
- The ultrasonic sensor must be able to detect any movement and after a certain proximity it is triggered and recognises intruders and create alerts. It needs to be able to distinguish stationary things from those that shouldn't set off the ultrasonic sensor. This means that objects such as walls, furniture etc. should not activate the system.
- The microphone must be able to detect loud noises. Its mechanism shouldn't turn on under typical environmental circumstances. In further detail, it should be able to tune out background noises like the wind, the rain, and other mild noises.
- The active buzzer must create an auditory alert immediately when intruders are detected by the ultrasonic sensor and/or the microphone.
- Additionally, the system must be able to be engaged and disabled by the user using the keypad input.
- When alarms are activated or sensors are operational, the appropriate indicator LEDs or displays should be produced.

1.2 APPLICATIONS

In 2021 alone, there were over 966 theft cases per 100,000 inhabitants reported in the union territory of Delhi in India. Followed by Haryana and Mizoram with over 91 and 88 cases per 100 thousand people in the states respectively. [1] These statistics show an increasing need for safety systems in our environments to ensure safety of the people. Home security systems have formed an integral part in ensuring safety in our society. These systems are constantly vigilant and react faster than any human. It can significantly reduce the number of casualties in home break-ins and

other such safety breaches. The proposed home security system can find applications in the security firms. Security firms can utilise this system to guarantee safety of their clients by setting up the designed system in their homes.

1.3 FEATURES

The proposed Home Security System will have the following features:

- A microphone that will be used to pick up loud noises from the environment.
- An ultrasonic sensor to keep track of objects in its adjacent radius so as to detect any movement when the intruder is in proximity, i.e below the threshold.
- A keypad that will be used by the user to manage the device. Users will be able to arm and disarm the system by entering the password into the system via the keypad.
- A 16x02 LCD display will be used to handle the user interface (UI). This will display all the appropriate messages as and when necessary.
- An active buzzer will work as the triggered system's auditory cue.
- A number of LEDs that will be utilised as system indicators to show when sensors are active, alarms are triggered, or system input is recognised.

CHAPTER 2
DESIGN

2.1 BLOCK DIAGRAM

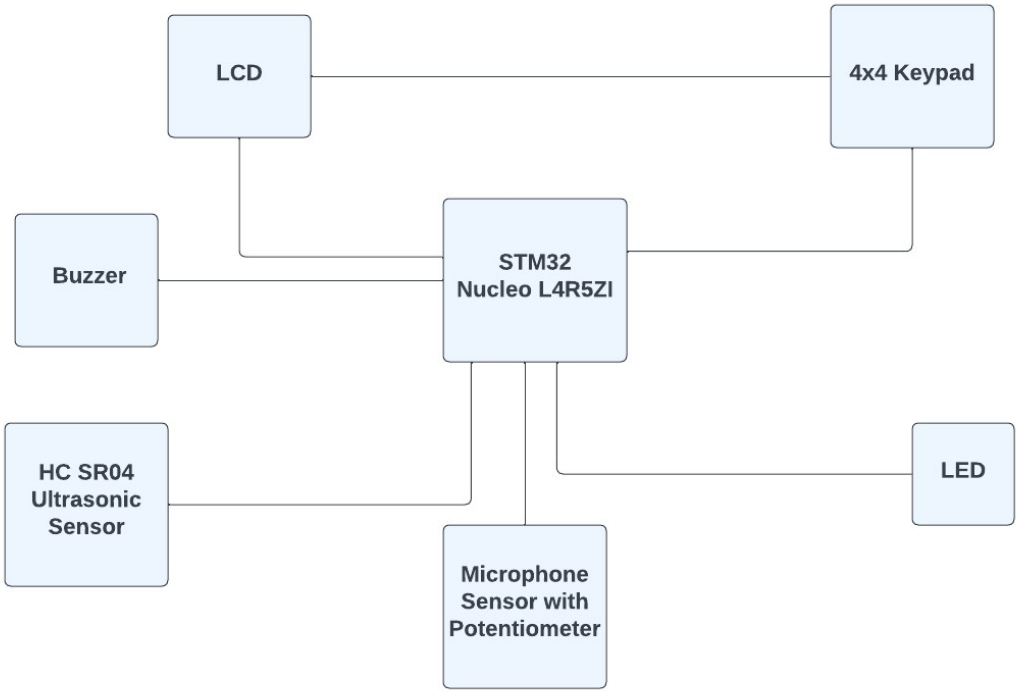


Figure 1: Block Diagram of the Home Security System

2.2 HARDWARE/SOFTWARE BLOCK ANALYSIS

As observed from the block diagram in Figure 1, we can see that there are various components, namely, the HC SR05 Ultrasonic Sensor, Microphone, LEDs, 16x02 LCD, Active buzzer and a 4x4 Keypad, attached to the STM32 NUCLEO L4R5ZI.

For first time use, the home security system needs to be set up first. When the user initially switches on the system, they need to set the passcode using the 4x4 keypad which will display the entered characters on the LCD. After the password is set, the system is initiated. For it to work, the user

has to press ‘A’ on the keypad and it arms the system. The system needs to be armed to make it function while the user is away or wants to turn the system on at night.

After the system is armed, the Buzzer, Microphone and Ultrasonic Sensor are activated to function. If there is a sound then the buzzer gets activated making noise and the LED light blinks signifying that an intruder has been heard.
If there is someone nearby, the ultrasonic sensor detects their presence and the buzzer gets activated and the LED light blinks making noise that the Intruder is detected.
In order to disarm the system, the user needs to press the key ‘A’ and has to enter the passcode again. If the pass code matches, the buzzer deactivates.

The software part of this project was coded and developed using the Keil Studio ARM MBED compiler online. The use of this online compiler makes developing codes for ARM based microcontrollers quicker and easier.

CHAPTER 3
SOFTWARE CODING AND ANALYSIS

3.1 CODE IMPLEMENTATION

Code for LCD:

```
#include "lcdIni.h"
#include "mbed.h"

LCD_func::LCD_func(unsigned char lcd_cols, unsigned char lcd_rows,
                   unsigned char charsize, PinName sda, PinName scl)
  : i2c(sda, scl) {

 _addr = LCD_ADDRESS_1602;
 _cols = lcd_cols;
 _rows = lcd_rows;
 _charsize = charsize;
 _backlightval = LCD_BACKLIGHT;
}

void LCD_func::begin() {
 _displayfunction = LCD_4BITMODE | LCD_1LINE | LCD_5x8DOTS;

 if (_rows > 1) {
   _displayfunction |= LCD_2LINE;
 }

 if ((_charsize != 0) && (_rows == 1)) {
   _displayfunction |= LCD_5x10DOTS;
 }
```

```
thread_sleep_for(50);

expanderWrite(
   _backlightval);
thread_sleep_for(1000);

write4bits(0x03 << 4);
wait_us(4500);

write4bits(0x03 << 4);
wait_us(4500);

write4bits(0x03 << 4);
wait_us(150);

write4bits(0x02 << 4);

command(LCD_FUNCTIONSET | _displayfunction);

_displaycontrol = LCD_DISPLAYON | LCD_CURSOROFF | LCD_BLINKOFF;
display();

clear();

_displaymode = LCD_ENTRYLEFT | LCD_ENTRYSHIFTDECREMENT;

command(LCD_ENTRYMODESET | _displaymode);

home();
backlight();
}
```

```
void LCD_func::clear() {
 command(LCD_CLEARDISPLAY);
 wait_us(2000);
}

void LCD_func::home() {
 command(LCD_RETURNHOME);
 wait_us(2000);
}

void LCD_func::setCursor(unsigned char col, unsigned char row) {
 int row_offsets[] = {0x00, 0x40, 0x14, 0x54};
 if (row > _rows) {
   row = _rows - 1;
 }
 command(LCD_SETDDRAMADDR | (col + row_offsets[row]));
}


void LCD_func::noDisplay() {
 _displaycontrol &= ~LCD_DISPLAYON;
 command(LCD_DISPLAYCONTROL | _displaycontrol);
}
void LCD_func::display() {
 _displaycontrol |= LCD_DISPLAYON;
 command(LCD_DISPLAYCONTROL | _displaycontrol);
}


void LCD_func::noCursor() {
 _displaycontrol &= ~LCD_CURSORON;
```

```
 command(LCD_DISPLAYCONTROL | _displaycontrol);
}
void LCD_func::cursor() {
 _displaycontrol |= LCD_CURSORON;
 command(LCD_DISPLAYCONTROL | _displaycontrol);
}


void LCD_func::noBlink() {
 _displaycontrol &= ~LCD_BLINKON;
 command(LCD_DISPLAYCONTROL | _displaycontrol);
}
void LCD_func::blink() {
 _displaycontrol |= LCD_BLINKON;
 command(LCD_DISPLAYCONTROL | _displaycontrol);
}


void LCD_func::scrollDisplayLeft(void) {
 command(LCD_CURSORSHIFT | LCD_DISPLAYMOVE | LCD_MOVELEFT);
}
void LCD_func::scrollDisplayRight(void) {
 command(LCD_CURSORSHIFT | LCD_DISPLAYMOVE | LCD_MOVERIGHT);
}


void LCD_func::leftToRight(void) {
 _displaymode |= LCD_ENTRYLEFT;
 command(LCD_ENTRYMODESET | _displaymode);
}
```

```
void LCD_func::rightToLeft(void) {
 _displaymode &= ~LCD_ENTRYLEFT;
 command(LCD_ENTRYMODESET | _displaymode);
}


void LCD_func::autoscroll(void) {
 _displaymode |= LCD_ENTRYSHIFTINCREMENT;
 command(LCD_ENTRYMODESET | _displaymode);
}


void LCD_func::noAutoscroll(void) {
 _displaymode &= ~LCD_ENTRYSHIFTINCREMENT;
 command(LCD_ENTRYMODESET | _displaymode);
}

void LCD_func::createChar(unsigned char location, unsigned char charmap[]) {
 location &= 0x7; // we only have 8 locations 0-7
 command(LCD_SETCGRAMADDR | (location << 3));
 for (int i = 0; i < 8; i++) {
   write(charmap[i]);
 }
}

void LCD_func::noBacklight(void) {
 _backlightval = LCD_NOBACKLIGHT;
 expanderWrite(0);
}

void LCD_func::backlight(void) {
 _backlightval = LCD_BACKLIGHT;
```

```
 expanderWrite(0);
}
bool LCD_func::getBacklight() { return _backlightval == LCD_BACKLIGHT; }


inline void LCD_func::command(unsigned char value) { send(value, 0); }

inline int LCD_func::write(unsigned char value) {
 send(value, Rs);
 return 1;
}

void LCD_func::send(unsigned char value, unsigned char mode) {
 unsigned char highnib = value & 0xf0;
 unsigned char lownib = (value << 4) & 0xf0;
 write4bits((highnib) | mode);
 write4bits((lownib) | mode);
}

void LCD_func::write4bits(unsigned char value) {
 expanderWrite(value);
 pulseEnable(value);
}

void LCD_func::expanderWrite(unsigned char _data) {
 char data_write[2];
 data_write[0] = _data | _backlightval;
 i2c.write(_addr, data_write, 1, 0);
 i2c.stop();
}

void LCD_func::pulseEnable(unsigned char _data) {
```

```
 expanderWrite(_data | En);
 wait_us(1);

 expanderWrite(_data & ~En);
 wait_us(50);
}

void LCD_func::load_custom_character(unsigned char char_num,
                                     unsigned char *rows) {
 createChar(char_num, rows);
}

void LCD_func::setBacklight(unsigned char new_val) {
 if (new_val) {
   backlight(); // turn backlight on
 } else {
   noBacklight(); // turn backlight off
 }
}

int LCD_func::print(const char *text) {

 while (*text != 0) {
   send(*text, Rs);
   text++;
 }
 return 0;
}
```

The variables mentioned below are used in LCD which are stored in lcdIni.h:

```
#include "mbed.h"
```

```
#define LCD_CLEARDISPLAY 0x01
#define LCD_RETURNHOME 0x02
#define LCD_ENTRYMODESET 0x04
#define LCD_DISPLAYCONTROL 0x08
#define LCD_CURSORSHIFT 0x10
#define LCD_FUNCTIONSET 0x20
#define LCD_SETCGRAMADDR 0x40
#define LCD_SETDDRAMADDR 0x80

#define LCD_ENTRYRIGHT 0x00
#define LCD_ENTRYLEFT 0x02
#define LCD_ENTRYSHIFTINCREMENT 0x01
#define LCD_ENTRYSHIFTDECREMENT 0x00


#define LCD_DISPLAYON 0x04
#define LCD_DISPLAYOFF 0x00
#define LCD_CURSORON 0x02
#define LCD_CURSOROFF 0x00
#define LCD_BLINKON 0x01
#define LCD_BLINKOFF 0x00

#define LCD_DISPLAYMOVE 0x08
#define LCD_CURSORMOVE 0x00
#define LCD_MOVERIGHT 0x04
#define LCD_MOVELEFT 0x00

#define LCD_8BITMODE 0x10
#define LCD_4BITMODE 0x00
#define LCD_2LINE 0x08
#define LCD_1LINE 0x00
#define LCD_5x10DOTS 0x04
```

```
#define LCD_5x8DOTS 0x00

#define LCD_BACKLIGHT 0x08
#define LCD_NOBACKLIGHT 0x00
#define LCD_ADDRESS_1602 0x4E
#define En 0x04
#define Rw 0x02
#define Rs 0x01

class LCD_func {
public:
    /**
   * Constructor
   *
   * @param lcd_cols  Number of columns your LCD display has.
   * @param lcd_rows  Number of rows your LCD display has.
   * @param charsize  The size in dots that the display has, use LCD_5x10DOTS or LCD_5x8DOTS.
   * @param sda       Pin to use for SDA connection of I2C for LCD
   * @param scl       Pin to use for the SCL connection of I2C for LCD
   */
  LCD_func( unsigned char lcd_cols, unsigned char lcd_rows, unsigned char charsize = LCD_5x8DOTS,PinName sda=PB_9, PinName scl=PB_8);
  void begin();
  void clear();
  void home();
  void noDisplay();
  void display();
  void noBlink();
  void blink();
  void noCursor();
  void cursor();
```

```
    void scrollDisplayLeft();
    void scrollDisplayRight();
    void printLeft();
    void printRight();
    void leftToRight();
    void rightToLeft();
    void shiftIncrement();
    void shiftDecrement();
    void noBacklight();
    void backlight();
    bool getBacklight();
    void autoscroll();
    void noAutoscroll();
    void createChar(unsigned char, unsigned char[]);
    void setCursor(unsigned char, unsigned char);
    virtual int write(unsigned char);
    void command(unsigned char);
    inline void blink_on() { blink(); }
    inline void blink_off() { noBlink(); }
    inline void cursor_on() { cursor(); }
    inline void cursor_off() { noCursor(); }
    void setBacklight(unsigned char new_val);
    void load_custom_character(unsigned char char_num, unsigned char *rows);
    int print(const char* text);
  private:
    void send(unsigned char, unsigned char);
    void write4bits(unsigned char);
    void expanderWrite(unsigned char);
    void pulseEnable(unsigned char);
    unsigned char _addr;
    unsigned char _displayfunction;
    unsigned char _displaycontrol;
```

```
  unsigned char _displaymode;
  unsigned char _cols;
  unsigned char _rows;
  unsigned char _charsize;
  unsigned char _backlightval;

  I2C i2c;
};
```

Code for methods of few functions:

```
#include <mbed.h>

#define UPPERCASE 65
#define LOWERCASE 97

void set_pin_mode(unsigned int pin, GPIO_TypeDef *port, unsigned int mode);
void enable_rcc(unsigned int port);
void write_to_pin(unsigned int pin, GPIO_TypeDef *port, unsigned int value);

void enable_rcc(unsigned int port) {
 unsigned int offset =
    (port - LOWERCASE) < 0
       ? port - UPPERCASE
       : port - LOWERCASE;
 RCC->AHB2ENR |= (0x1 << offset);
}

void set_pin_mode(unsigned int pin, GPIO_TypeDef *port, unsigned int mode) {

 unsigned int offset = pin * 2;
 if (mode) {
  port->MODER &= ~(0x2 << (offset));
```

```
  port->MODER |= (0x1 << (offset));
 } else {
  port->MODER &= ~(0x3 << (offset));
 }
}

void write_to_pin(unsigned int pin, GPIO_TypeDef *port, unsigned int value) {
 value ? port->ODR |= (0x1 << pin) : port->ODR &= ~(0x1 << pin);
}
```

Code for variables in method:

```
#include "mbed.h"

void set_pin_mode(unsigned int pin, GPIO_TypeDef *port, unsigned int mode);
void enable_rcc(unsigned int port);
void write_to_pin(unsigned int pin, GPIO_TypeDef *port, unsigned int value);
```

Main file Code:

```
#include "DigitalOut.h"
#include "ThisThread.h"
#include "Ticker.h"
#include "mbed_thread.h"
#include <lcdIni.h>
#include <methods.h>
#include <cstdio>
#include <mbed.h>
#include <string>
#include <time.h>

void isr_col(void);
void isr_falling_edge(void);
```

```
void isr_microphone(void);

void isr_ultrasonic(void);
void isr_ultrasonic_falling_edge(void);

void ultrasonic_handler(void);
void trigger_ultrasonic_sensor(void);

void microphone_handler(void);

void row_handler(void);
void key_handler(void);

void power_on_mode(void);
void unarmed_mode(void);
void armed_mode(void);
void triggered_mode(void);
void trigger_mode_transition(void);

void idle_timeout_handler(void);
void set_display_off(void);
const uint32_t TIMEOUT_MS = 5000;
int key_pressed = 0;
int debounced = 0;
int display_on = 1;
volatile int echo_on = 0;
string password = "1234";
string password_entered = "1234";
int password_position = 0;
int entering_password = 0;
```

```
LCD_func LCD(16, 2, LCD_5x8DOTS, PB_9, PB_8);
InterruptIn col_0(PF_14, PullDown);
InterruptIn col_1(PE_11, PullDown);
InterruptIn col_2(PE_9, PullDown);
InterruptIn col_3(PF_13, PullDown);

InterruptIn microphone(PD_7, PullDown);
InterruptIn ultrasonic_echo(PD_5, PullDown);

DigitalOut ultrasonic_trigger(PD_6);
DigitalOut active_buzzer(PD_4);
DigitalOut microphone_enable(PF_12);
DigitalOut alarm_leds(PD_15);

Thread row_thread;
Thread key_thread;

Mutex resource_lock;
EventQueue queue;
Timeout idle_timeout;
Timeout ultrasonic_timeout;
Ticker ultrasonic_ticker;
char keypad[4][4] = {{'1', '2', '3', 'A'},
                     {'4', '5', '6', 'B'},
                     {'7', '8', '9', 'C'},
                     {'*', '0', '#', 'D'}};
int mode = 0;
int row = 0;
int main() {
   LCD.clear();
  col_0.enable_irq();
 col_1.enable_irq();
```

```
col_2.enable_irq();
col_3.enable_irq();

enable_rcc('a');
enable_rcc('c');

set_pin_mode(3, GPIOA, 1);

set_pin_mode(0, GPIOC, 1);

set_pin_mode(3, GPIOC, 1);
 set_pin_mode(1, GPIOC, 1);

LCD.begin();
LCD.print("Set Passcode: ");
LCD.setCursor(0, 1);

col_0.rise(&isr_col);
col_1.rise(&isr_col);
col_2.rise(&isr_col);
col_3.rise(&isr_col);

microphone.rise(&isr_microphone);

ultrasonic_echo.rise(&isr_ultrasonic);
ultrasonic_echo.fall(&isr_ultrasonic_falling_edge);

col_0.fall(&isr_falling_edge);
col_1.fall(&isr_falling_edge);
col_2.fall(&isr_falling_edge);
col_3.fall(&isr_falling_edge);
```

```
 idle_timeout.attach(&idle_timeout_handler, 10s);
  ultrasonic_ticker.attach(&trigger_ultrasonic_sensor, 500ms);
 row_thread.start(row_handler);
 key_thread.start(key_handler);

 Watchdog &watchdog = Watchdog::get_instance(); // Initialize watchdog
 watchdog.start(TIMEOUT_MS);

 queue.dispatch_forever();
}

void isr_col(void) { key_pressed = 1; }

void isr_falling_edge(void) {
 key_pressed = 0;
 debounced = 0;
}

void isr_microphone(void) {
 microphone_enable = 0;
 queue.call(&microphone_handler);
}

void isr_ultrasonic(void) {
 echo_on = 1;
 ultrasonic_timeout.attach(&ultrasonic_handler, 888us);
}

void isr_ultrasonic_falling_edge(void) { echo_on = 0; }

void microphone_handler() {
 resource_lock.lock();
```

```
 if (mode == 2) {
   mode = 3;
   alarm_leds = 1;
   password_position = 0;
   entering_password = 0;
   active_buzzer = 1;
   LCD.clear();
   LCD.print("Triggered");
 }
 else {
     microphone_enable = 1;
 }
 resource_lock.unlock();
}

void ultrasonic_handler() {
 if (mode == 2 && !echo_on) {
   queue.call(&trigger_mode_transition);
 }
}

void trigger_mode_transition() {

 mode = 3;
 alarm_leds = 1;
 password_position = 0;
 entering_password = 0;
 microphone_enable = 0;
 active_buzzer = 1;
 LCD.clear();
 LCD.print("Triggered");
}
```

```
void key_handler() {
 while (1) {
  resource_lock.lock();
  if (key_pressed) {
   if (!debounced) {
    thread_sleep_for(10);
    if (key_pressed) {
     debounced = 1;
     if (!display_on) {
      display_on = 1;
      LCD.backlight();
     }
     idle_timeout.detach();
     idle_timeout.attach(&idle_timeout_handler, 10s);
     switch (mode) {
     case 0:
      power_on_mode();
      break;
     case 1:
      unarmed_mode();
      break;
     case 2:
      armed_mode();
      break;
     case 3:
      triggered_mode();
      break;
     }
    }
   }
  }
```

```
   resource_lock.unlock();
 }
}

void row_handler() {
 while (1) {
   resource_lock.lock();
   if (!key_pressed) {
     row++;
     row %= 4;
     switch (row) {
     case 0:
       write_to_pin(0, GPIOC, 0);
       write_to_pin(3, GPIOC, 0);
       write_to_pin(1, GPIOC, 0);
       write_to_pin(3, GPIOA, 1);
       break;
     case 1:
       write_to_pin(3, GPIOA, 0);
       write_to_pin(3, GPIOC, 0);
       write_to_pin(1, GPIOC, 0);
       write_to_pin(0, GPIOC, 1);
       break;
     case 2:
       write_to_pin(3, GPIOA, 0);
       write_to_pin(0, GPIOC, 0);
       write_to_pin(1, GPIOC, 0);
       write_to_pin(3, GPIOC, 1);
       break;
     case 3:
       write_to_pin(3, GPIOA, 0);
       write_to_pin(0, GPIOC, 0);
```

```
      write_to_pin(3, GPIOC, 0);
      write_to_pin(1, GPIOC, 1);
      break;
    }
  }
  resource_lock.unlock();
  Watchdog::get_instance().kick();
 }
}

void power_on_mode() {
 if (col_0.read() && keypad[row][0] != '*') {
  password[password_position] = keypad[row][0];
 } else if (col_1.read()) {
  password[password_position] = keypad[row][1];
 } else if (col_2.read() && keypad[row][2] != '#') {
  password[password_position] = keypad[row][2];
 }
 if ((col_0.read() && keypad[row][0] != '*') || col_1.read() ||
    (col_2.read() && keypad[row][2] != '#')) {
  password_position++;
  LCD.print("*");
  if (password_position == 4) {
    password_position = 0;
    mode = 1;
    LCD.clear();
    LCD.print("Unarmed");
  }
 }
}

void unarmed_mode() {
```

```
if (col_0.read() && keypad[row][0] != '*' && entering_password) {
  password_entered[password_position] = keypad[row][0];
} else if (col_1.read() && entering_password) {
  password_entered[password_position] = keypad[row][1];

} else if (col_2.read() && keypad[row][2] != '#' && entering_password) {
  password_entered[password_position] = keypad[row][2];
} else if (col_3.read() && keypad[row][3] == 'A' && !entering_password) {
  entering_password = 1;
  LCD.clear();
  LCD.print("Enter Passcode: ");
  LCD.setCursor(0, 1);
}
if ((col_0.read() && keypad[row][0] != '*' && entering_password) ||
    (col_1.read() && entering_password) ||
    (col_2.read() && keypad[row][2] != '#' && entering_password)) {
  password_position++;
  LCD.print("*");
  if (password_position == 4) {
    password_position = 0;
    entering_password = 0;
    if (password_entered == password) {
      mode = 2;
      microphone_enable = 1;
      LCD.clear();
      LCD.print("Armed");
    } else {
      LCD.clear();
      LCD.print("Incorrect");
      LCD.setCursor(0, 1);
      LCD.print("Passcode");
      thread_sleep_for(2000);
```

```
      LCD.clear();
      LCD.print("Unarmed");
    }
  }
 }
}

void armed_mode() {
 if (col_0.read() && keypad[row][0] != '*' && entering_password) {
  password_entered[password_position] = keypad[row][0];
 } else if (col_1.read() && entering_password) {
  password_entered[password_position] = keypad[row][1];

 } else if (col_2.read() && keypad[row][2] != '#' && entering_password) {
  password_entered[password_position] = keypad[row][2];
 } else if (col_3.read() && keypad[row][3] == 'A' && !entering_password) {
  entering_password = 1;
  LCD.clear();
  LCD.print("Enter Passcode: ");
  LCD.setCursor(0, 1);
 }
 if ((col_0.read() && keypad[row][0] != '*' && entering_password) ||
    (col_1.read() && entering_password) ||
    (col_2.read() && keypad[row][2] != '#' && entering_password)) {
  password_position++;
  LCD.print("*");
  if (password_position == 4) {
    password_position = 0;
    entering_password = 0;
    if (password_entered == password) {
      mode = 1;
      LCD.clear();
```

```
      LCD.print("Unarmed");
    } else {
      LCD.clear();
      LCD.print("Incorrect");
      LCD.setCursor(0, 1);
      LCD.print("Passcode");
      thread_sleep_for(2000);
      LCD.clear();
      LCD.print("Armed");
    }
  }
 }
}

void triggered_mode() {
 if (col_0.read() && keypad[row][0] != '*' && entering_password) {
  password_entered[password_position] = keypad[row][0];
 } else if (col_1.read() && entering_password) {
  password_entered[password_position] = keypad[row][1];

 } else if (col_2.read() && keypad[row][2] != '#' && entering_password) {
  password_entered[password_position] = keypad[row][2];
 } else if (col_3.read() && keypad[row][3] == 'A' && !entering_password) {
  entering_password = 1;
  LCD.clear();
  LCD.print("Enter Passcode: ");
  LCD.setCursor(0, 1);
 }
 if ((col_0.read() && keypad[row][0] != '*' && entering_password) ||
    (col_1.read() && entering_password) ||
    (col_2.read() && keypad[row][2] != '#' && entering_password)) {
  password_position++;
```

```
    LCD.print("*");
    if (password_position == 4) {
      password_position = 0;
      entering_password = 0;
      if (password_entered == password) {
        mode = 1;
        LCD.clear();
        LCD.print("Unarmed");
        active_buzzer = 0;
        alarm_leds = 0;
      } else {
        LCD.clear();
        LCD.print("Incorrect");
        LCD.setCursor(0, 1);
        LCD.print("Passcode");
        thread_sleep_for(2000);
        LCD.clear();
        LCD.print("Triggered");
      }
    }
  }
}

void idle_timeout_handler() {
  if (display_on) {
    display_on = 0;
    queue.call(&set_display_off);
  }
}

void set_display_off() {
  LCD.noBacklight();
```

```
 LCD.clear();
 password_position = 0;
 entering_password = 0;
 switch (mode) {
 case 0:
   LCD.print("Set Passcode: ");
   LCD.setCursor(0, 1);
   break;
 case 1:
   LCD.print("Unarmed");
   break;
 case 2:
   LCD.print("Armed");
   break;
 case 3:
   LCD.print("Triggered");
   break;
 }
}

void trigger_ultrasonic_sensor() {
 if (!echo_on) {
   ultrasonic_trigger = 1;
   wait_us(10);
   ultrasonic_trigger = 0;
   if (mode == 3) {
     alarm_leds = !alarm_leds;
   }
 }
}
```

3.2 SNAPSHOTS OF IMPLEMENTATION

Figure 2: Code and its blocks in the build

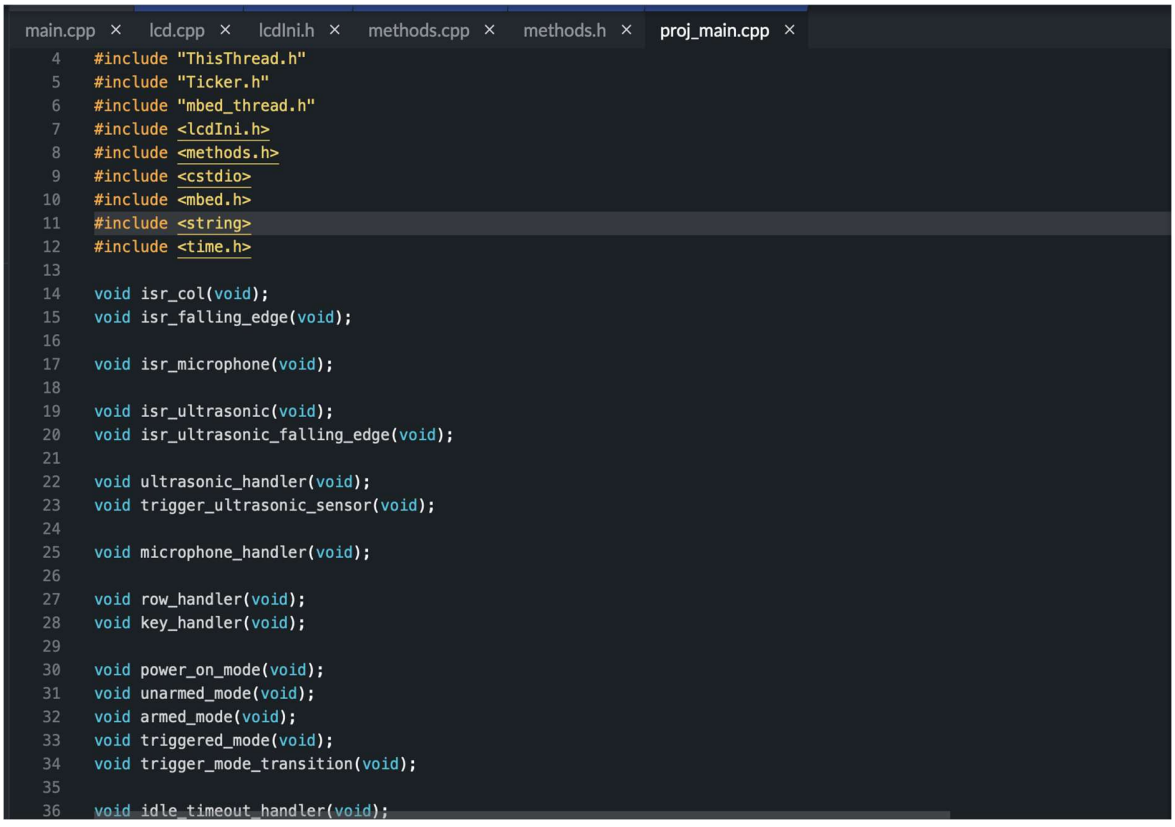


Figure 3: "Main" file of the Home Security System

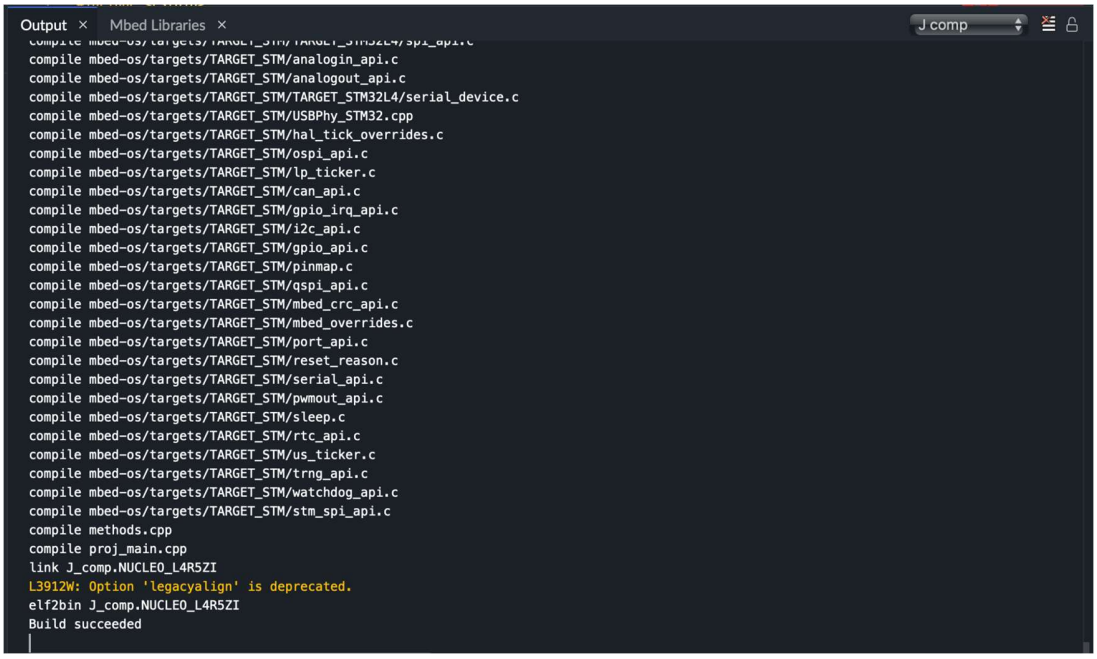


Figure 4: Output Compilation screen - Build Succeeded

3.3 SNAPSHOTS OF RESULTS

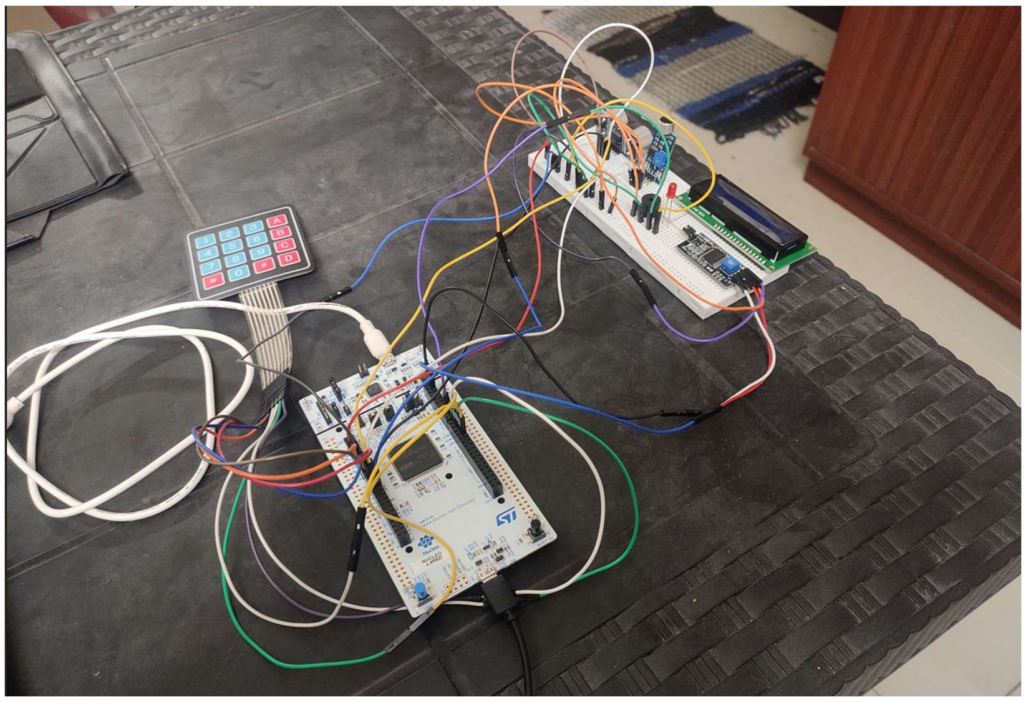


Figure 5: Complete Circuit in standby state

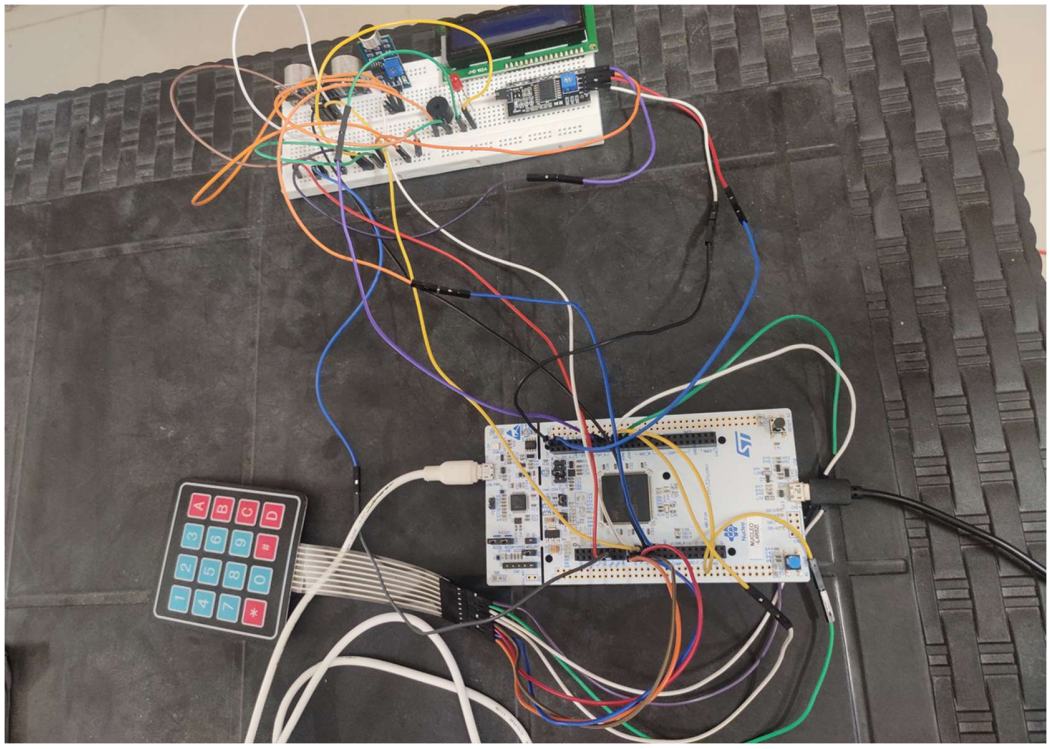


Figure 6 : Complete Circuit in Standby state

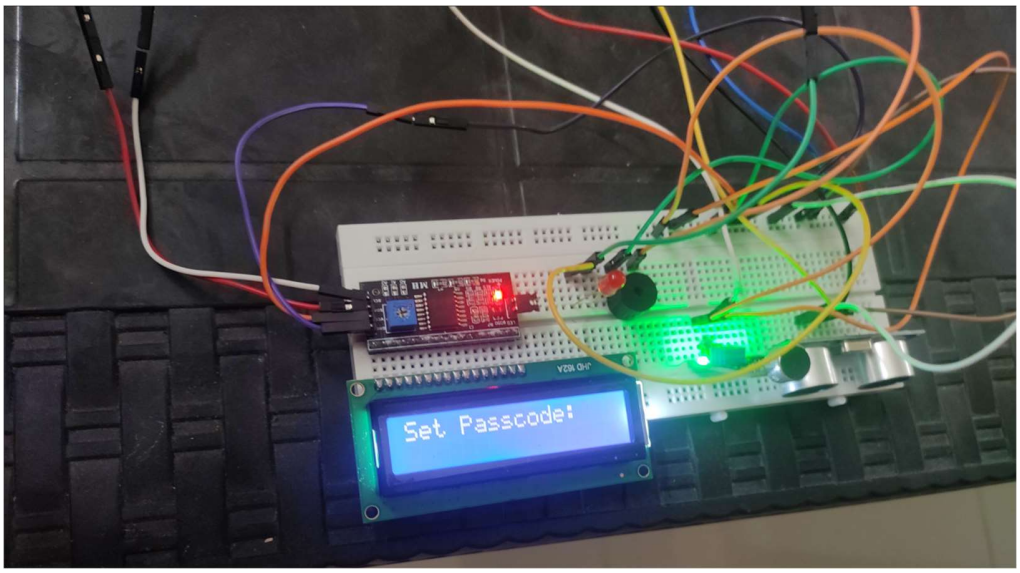


Figure 7: GUI on LCD displaying “Set Passcode”

This prompts the user to set the passcode on the security system of their choice so as to arm and disarm the system and to access the system using the keypad.

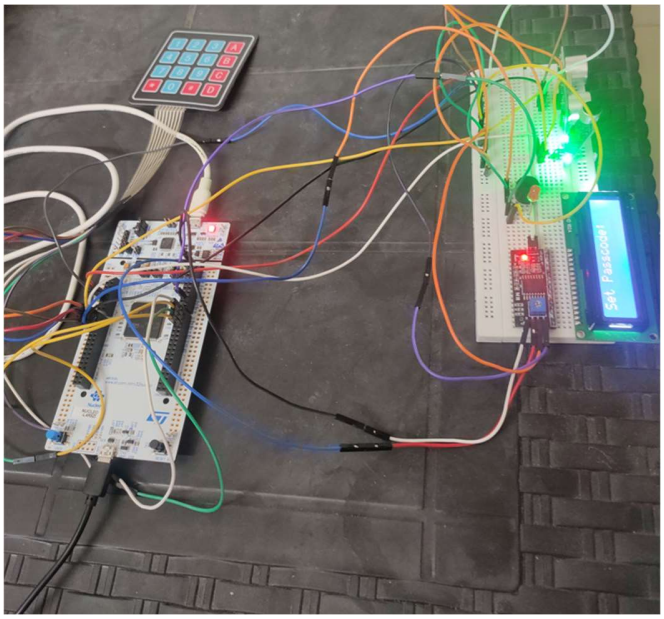


Figure 8: Circuit in the set passcode state where in it takes input from user for passcode

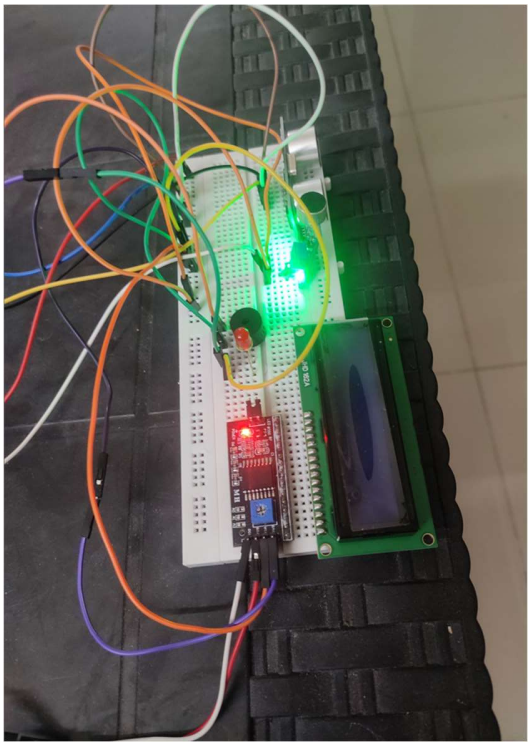


Figure 9: Standby state

The circuit goes into a standby state when it is not working and the LCD turns off when the system has a 10 second duration of inactivity and turns on immediately whenever a key is pressed on the keypad.

CHAPTER 4
CONCLUSION AND FUTURE WORK

4.1 CONCLUSION

Home security creates a flexible, comfortable and safe environment for residents, enhancing their quality of life. The key element of home automation is a security system. In recent decades, traditional alarm-based security methods have become increasingly popular. However, today's embedded systems are built to ensure security thanks to significant advancements made in microcontroller technologies. The proposed home security system can prove to be very useful for anyone who wants to ensure safety of their living spaces at an affordable rate. It was designed using the STM NUCLEO-L4R5ZI development board along with other components such as an ultrasonic sensor, a microphone, a keypad, a 16x2 LCD unit, an active buzzer and LEDs. It can successfully detect the presence of an intruder using the ultrasonic sensor within its threshold and also detect noises using the microphone. On detecting intruders, the active buzzer immediately creates auditory cues to alert users and LED glows in a certain pattern as well to provide for a visual alert. Furthermore, the LCD also displays the appropriate messages according to sensor inputs. Thus, to conclude, we can say that the proposed Home Security System was designed and implemented successfully.

4.2 FUTURE WORK

The home security system has a lot of scope for enhancements in the future. Its efficiency can be increased in a number of ways by adding additional modules to make intruder detection more accurate. A few of the areas where we plan to implement in the future to further develop this project is by introducing a GSM module so as to facilitate sending of remote text alerts to the user’s phone in case of a break-in. Another method would be to also link CCTV camera feed monitoring systems for cameras attached at all entry points in the house such as doors, windows, balconies, etc.

REFERENCES

[1] https://www.statista.com/statistics/632912/reported-theft-rate-by-state-india/ (Last accessed on 12th November, 2022)

[2] https://os.mbed.com/platforms/NUCLEO-L4R5ZI/ (Last accessed on 12th November, 2022)

[3] Paul Kocher, Ruby Lee, Gary McGraw, Anand Raghunathan, and Srivaths Ravi. 2004. Security as a new dimension in embedded system design. In Proceedings of the 41st annual Design Automation Conference (DAC '04). Association for Computing Machinery, New York, NY, USA, 753–760. https://doi.org/10.1145/996566.996771

[4] https://controllerstech.com/i2c-lcd-in-stm32/ (Last accessed on 12th November, 2022)

[5] https://www.azosensors.com/article.aspx?ArticleID=229 (Last accessed on 12th November, 2022)

[6] https://www.digikey.com/htmldatasheets/production/1979760/0/0/1/hc-sr04.html
(Last accessed on 12th November, 2022)

[7] Huang, H., Xiao, S., Meng, X., & Xiong, Y. (2010, April). A remote home security system based on wireless sensor network and GSM technology. In *2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing* (Vol. 1, pp. 535-538). IEEE.

[8] Lee, J. V., Chuah, Y. D., & Chai, C. T. (2013). A multilevel home security system (mhss). International Journal of Smart Home, 7(2), 49-60.

[9] Sahani, M., Nanda, C., Sahu, A. K., & Pattnaik, B. (2015, March). Web-based online embedded door access control and home security system based on face recognition. In 2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015] (pp. 1-6). IEEE.

[10] Parab, A. S., Joglekar, A., & Parab, A. S. (2015). Implementation of home security system using GSM module and microcontroller. International Journal of Computer Science and Information Technologies, 6(3), 2950-2953.

BIODATA

Name: Jaisuraj Bantupalli

Mobile Number: 9440129033

Email: jaisuraj.bantupalli2020@vitstudent.ac.in

Permanent Address: 26-15-129, Jayalakshmi Metal Mart, Poirna Market, Main Road, Visakhapatnam-530001

Name: Sanjukta Roy

Mobile Number: 7358001420

Email: sanjukta.roy2020@vitstudent.ac.in

Permanent Address: B4 F3, VGN Minerva, 273, Guruswamy Road, Nolambur, Chennai - 600095

Name: Ashwin Santhosh Nair

Mobile number: 9741498624

E-mail: ashwin.2020@vitstudent.ac.in

Permanent Address: Door 103, SSK Residency, Near Aditya Kalyana Mantapa, AECS Layout B Block, Singasandra, Bengaluru – 560068