Project Report for ESDI: Women Safety in Automobiles

Table of Contents

1. Student Information

Student Name:

Sham Ganesh M

Address:

193/C, 3rd Floor, Vanniyar Street, Padi

Chennai – 600 050

Contact:

+91 76399 71487

shamganesh7slan@gmail.com

University Name and Location:

Vellore Institute of Technology – Chennai Campus

Vandalur – Kelambakkam Road,

Chennai – 600 127

University Serial Number:

22BLC1341

Submission Date:

11th July 2024 (11 – 07 – 2024)

2. Introduction

In recent years, the issue of women's safety has made a huge impact on society. As an effort to enhance security measures, the "Woman Safety in Automobiles" project has been developed. This project aims to address safety concerns by utilizing technology to create a more secure environment for women while traveling. The core functionality involves sending alert message to predefined phone number and initiate an alarm in emergencies. As incidents of harassment and assault during travels have been frequently, by integrating this system into automobiles, we can provide an additional layer of security and peace of mind for female passengers. This project has a wide span of applications including personal vehicles, taxis, ride-sharing services, and public transportation, making it versatile and widely applicable. As we advance towards smarter and safer cities, incorporating women's safety features in automobiles becomes a crucial step in fostering an inclusive and secure environment for all.

3. Project Specification

3.1 Functional Requirement

- System should have a push button, when pressed the system will enter alert mode.
- On alert, an SMS with an appropriate message is sent to 2 predefined phone numbers.
- On alert, the latitude and longitude information (position information is sent to the same phone numbers).
- A buzzer connected to the system will start buzzing on alert and it will keep buzzing for the next 2 hours.
- The system will stay in alert mode, once initiated.
- Buzzer will stop once a STOP message is received from a predefined phone number.

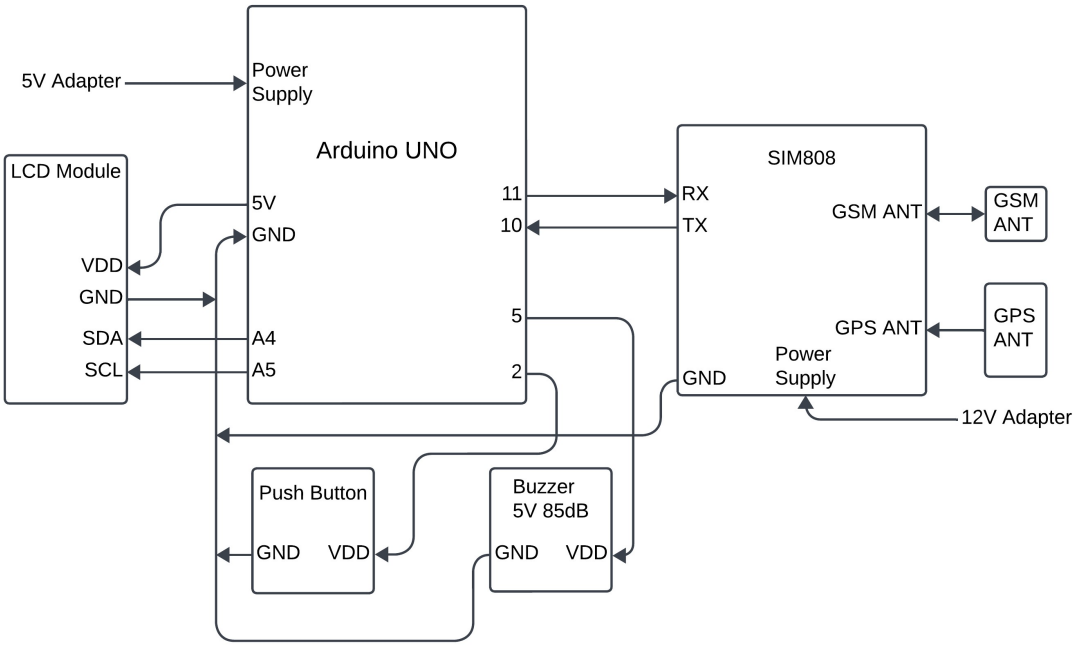
3.2 Non-Functional Requirement

- Performance requirements:
  - SMS should be sent within 30 seconds from the time the switch is detected.
  - The buzzer should be audible outside the closed car.

- Environment requirements:
  - The system should work in the temperature range 0 to 60 degrees centigrade respectively.

4. Hardware Architecture

Block Diagram:



Arduino UNO:

It is a micro-controller, which has 14 digital pins and 6 analog pins with 6 PWM pins. The reason I used UNO is because it can utilise an external DC power supply, and as it uses ATMEGA328P it consumes less power than NANO and relatively UNO costs lesser than most micro-controllers. Operational temperature: -40 C to 85 C.

SIM808 Module:

It has GPS, GSM and GPRS interfaces. This module supports Quad band 850/900/1800/1900 MHz, consumes less power, supports 1.5V and 3V SIM cards, Serial communication compatibility for easy use, supports DC power supply, Li-ion battery supply, and it can be used for several projects. Also, by combining both GSM and GPS into single module we can reduce a lot of power consumption and space requirements and pin usage of micro-controller. Connections for this module is to connect the TX of module to RX of controller and RX of module to TX of controller and attach GND pin of module to GND of controller.

Operational Temperature: -40 C to 85 C.

5V Active Buzzer:

Sound pressure: 80 dB

As it is of active type, this buzzer can be controlled by directly attaching it to a power source or by giving a wave signal to it. Unlike passive type which can only be controlled by giving a wave signal. So we can just connect the positive terminal of buzzer to Arduino digital pin (5) and negative terminal to GND.

Operational Temperature: -20 C to 65 C.

LCD Module (I2C Interface):

This module is a 16 character and 2-line LCD display with I2C type communication. Using I2C type, we are reducing the number of pins required by the micro-controller to control the module. It also comes with a dedicated library for easy access. Connection follows as SDA to A4 of UNO and SCL to A5 of UNO as default for the library.

Operational Temperature: ~0 C to ~60C.

4.1 List of Hardware Components used

S. No	Hardware Description	Quantity	Remarks
1	SIM808 with GPS, GSM, GPRS	1	Used this product as it has both GPS and GSM, and it satisfied the requirements.
2	Arduino UNO	1	Micro-controller with necessity needed for this project.
3	LCD Module (I2C)	1	Used an I2C type LCD module for easy access.
4	Buzzer (5V) 85dB	1	Checked the product to be audible outside even thick walls.
5	Push Button	1	-
6	Power source for SIM808	1	Used an 12V adapter to power the SIM808 Module
7	Power source for Arduino UNO	1	Used an 5V Adapter to power the Arduino

5. Software Architecture

We used functions such as getGPSdata(), getGPSdata(float&, float&) to get GPS data (Latitude and Longitude) from the SIM808 module, here getGPSdata(float&, float&) gets the gps data and stores it in the argument variable, whereas the getGPSdata() gets the gps data and displays it in the LCD display by utilizing a displayGPSdata(string, string) sub-function.

Functions such as displayGPSData(string, string), displayData(int, string, int, string, unsigned long) and displayData(int, string, int, string) are used to display data on the LCD, displayGPSdata is used to display the latitude and longitude coordinates, and the two displayData functions differs by a single argument (unsigned long) which is used to have the data on LCD to be erased after a specific period of time (in milliseconds).

Function Alert(char*) is used to initiate the Alert mode which turns on the Buzzer and sends SMS to the predefined phone numbers by using SendSMS(char*) function, and the buzzer state is maintained for the specified duration (2 hrs) and during the duration of the buzzer we will be constantly checking for message from the predefined phone number to STOP the alert mode, and also we are continuously checking the gps and displaying it on the lcd screen. Once a interrupt message from the predefined phone numbers is received, by the function checkForSTOP() the buzzer gets turned off and alert mode ends.

Function SendSMS(char*) is used to send message to the predefined phone numbers through SIM808.

Function checkForSTOP() has a return type of bool, it checks if any unread SMS is left on the SIM808 and reads it, if the message is “STOP” (in any case format) and is from any one of the predefined phone number we return true and false for all other conditions. We use checkPhone(string) to check whether the message belongs to one of the predefined phone numbers.

Pseudocode:

Alert(message)

Start time = current time

Start buzzer

Send sms to predefined phone numbers

Till current time – start time exceeds duration

Check for interrupt message from the predefined phone numbers

If interrupt message received break out of loop

Constantly get GPS data and display on LCD

Turn off buzzer

checkForSTOP()

check for unread sms

if unread sms present

read sms and store message, phone number, date and time

mark sms as read and delete

check if message equals “STOP” and phone number matches any one of the predefined phone numbers

if true return true

return false

checkPhone(phone number)

check if phone number belongs to predefined phone numbers

if true return true

return false

getGPSdata(lat, lon)

get gps data

store latitude and longitude data in lat and lon

getGPSdata()

get gps data

display latitude and longitude data in LCD display

displayGPSData(lat, lon)

clear lcd data

display latitude data on first line from 1st column

display longitude data on second line from 1st column

displayData(pos1, data1, pos2, data2, dur)

clear lcd data

display data1 in pos1+1 th column in 1st row

display data2 in pos2+1 th column in 2nd row

wait till time dur passes

clear lcd data

displayData(pos1, data1, pos2, data2)

clear lcd data

display data1 in pos1+1 th column in 1st row

display data2 in pos2+1 th column in 2nd row

SendSMS(message)

Send sms to predefined phone numbers

setup()

initialise lcd module

display welcome address to user using lcd

define pinMode for pins and turn off the buzzer

initialise the sim808 module

display successful message to user via lcd

turn ON the gps of sim808

display gps turned ON in lcd

loop()

get gps data from sim808

display the data on lcd

create a message and store the gps data

if pushbutton is pressed

inititiate alert mode

lcd.init() to initialise lcd module

lcd.backlight() to turn ON the backlight of lcd module

lcd.clear() to clear data on lcd

lcd.setCursor(col,row) to set cursor to specified col and row on lcd

lcd.print(data) to display data from the cursor as starting point

sim808.init() to intialise sim808 module

sim808.attachGPS() to turn ON the gps of sim808

sim808.getGPS() to get gps data from sim808

sim808.isSMSunread() to check if any unread sms available and returns the index

sim808.readSMS() to read the indexed SMS with phone number, message, date and time

sim808.deleteSMS() to delete the indexed SMS

sim808.sendSMS(phone, msg) to send message to defined phone number

pinMode(pin, type) to initialise the type of pin

types used -> OUTPUT, INPUT_PULLUP

INPUT_PULLUP reverses the data interpretation at the pin, that is, it is read as high(4.8V) always and whenever short circuited to gnd it goes to low (0V)

dtostrf(float_var, int_part, decimal_part, string_var) to convert a float to string with precision

sprint(var, data) to force the data onto the character array var

millis() – gets the current runtime clock of the processor

Code

```
#include <LiquidCrystal_I2C.h>
#include <DFRobot_SIM808.h>
#include <SoftwareSerial.h>

LiquidCrystal_I2C lcd(0x27,16,2);

#define pushButton 2
#define buzzer 5

#define TX 10 // TX of SIM 808
#define RX 11 // RX of SIM 808
SoftwareSerial mySerial(TX, RX);
DFRobot_SIM808 sim808(&mySerial);

char phone1[] = "+916379161303";  // predefined phone number 1
char phone2[] = "+918608071697";  // predefined phone number 2
```

```
unsigned long duration = 2 * 60 * 60 * 1000; // 2 hours in milliseconds

void setup() {
  Serial.begin(9600);
  // Initializing LCD Module
  lcd.init();
  lcd.backlight();

  displayData(0, "Welcome", 4, "User", 500); // user can be replaced with
their name max 12 chars

  pinMode(buzzer, OUTPUT);
  pinMode(pushButton, INPUT_PULLUP);
  digitalWrite(buzzer, 0);

  mySerial.begin(9600);

  displayData(0, "Initialising", 0, "");
  // Initializing SIM808 Module
  while(!sim808.init()) {
    delay(1000);
    Serial.print("Sim808 init error\r\n");
  }
  displayData(0, "Initialising", 4, "Successful", 500);
  // Turning ON GPS
  while(!sim808.attachGPS());
  Serial.println("GPS powered ON successfully");
  displayData(0, "GPS", 4, "Turned ON", 500);
}

void loop() {
  float lat, lon; // to store latitude and longitude coordinates
  displayData(0, "Getting", 4, "GPS Data");
  getGPSdata(lat, lon); // Getting GPS data - latitude, longitude
  displayGPSData(String(lat, 6), String(lon, 6)); // displaying the GPS data
on LCD with 6 decimal places
  char la[11], lo[11];
  dtostrf(lat, 4, 6, la); dtostrf(lon, 4, 6, lo);

  char message[100];
  sprintf(message, "Emergency!!! \nLatitude  : %s\nLongitude : %s\n",
la,  lo); // prepare the message with the GPS data

  if(!digitalRead(pushButton))  // need to press the button for atleast 0.5
sec due to delays and functions
    Alert(message);

  delay(500);
```

```
}

// Function to initiate the alert and buzzer
void Alert(char* message) {
  unsigned long start_time = millis(); // taking the starting time
  digitalWrite(buzzer, 1); // turn ON the buzzer
  SendSMS(message);  // sending SMS
  digitalWrite(buzzer, 0); // turn OFF the buzzer once the message sent to
make sure the sms has been sent (a small beep will be generated to show that
message has been sent)
  displayData(0,"Alert",4,"Sent",1000); // displaying that sms has been sent
for 1 sec

  while(millis()-start_time < duration) { // checking the current time with
starting time and running the program till the required duration
    digitalWrite(buzzer, 1);  // turn ON the buzzer
    if(checkForSTOP())  break;
    getGPSdata(); // function to always show the gps coordinate of the user
    digitalWrite(buzzer, 0);  // turn off the buzzer
  }
  digitalWrite(buzzer,0); // turning off the buzzer
  displayData(0,"Buzzer",4,"Turned OFF",500);
}

// Function to check for the STOP message
bool checkForSTOP(void) {
  int messageIndex = sim808.isSMSunread();  // checks for any unread SMS

  if(messageIndex > 0) {
    char message[160];  // stores the message received
    char phoneN[16];    // stores the phone number the message is from
    char datetime[24];  // stores the time and date at which the message is
received
    sim808.readSMS(messageIndex, message, 160, phoneN, datetime);  // reads
the unread SMS
    String msg = message;
    sim808.deleteSMS(messageIndex); // delete the SMS after reading
    if(msg.equalsIgnoreCase("STOP") && checkPhone(phoneN))  // if message is
Stop (any case format) from pre-defined phone number, returns true
      return true;
  }
  return false;
}

// Function to check whether the phone number belongs to predefined phone
number
bool checkPhone(String phoneN) {
  if(phoneN.equals(String(phone1)) || phoneN.equals(String(phone2)))
```

```
    return true;
  return false;
}

// Function to get the GPS data and store them in the input argument
void getGPSdata(float& lat, float& lon) {
  while(!sim808.getGPS());  // getting gps
  lat = sim808.GPSdata.lat; // latitude data
  lon = sim808.GPSdata.lon; // longitude data
}

// Function to get the GPS data and displays the data on lcd by itself
void getGPSdata() {
  while(!sim808.getGPS());  // getting gps
  float lat = sim808.GPSdata.lat; // latitude data
  float lon = sim808.GPSdata.lon; // longitude data

  displayGPSData(String(lat, 6), String(lon, 6)); // displaying the GPS data
on LCD with 6 decimal places
}

// Function to display GPS data on a LCD with two rows r1 -> latitude, r2 ->
longitude
void displayGPSData(String lat, String lon) {
  lcd.clear(); delay(1);
  lcd.setCursor(0, 0);  lcd.print("Lat : " + lat);  // row 1 -> latitude
  lcd.setCursor(0, 1);  lcd.print("Lon : " + lon);  // row 2 -> longitude
}

// Function to display data on a LCD with two rows, with a duration to display
attached
void displayData(int pos1, String data1, int pos2, String data2, unsigned long
dur) {
  lcd.clear(); delay(1);
  lcd.setCursor(pos1, 0);  lcd.print(data1);  // row1 -> pos1 defines the
column
  lcd.setCursor(pos2, 1);  lcd.print(data2);  // row2 -> pos2 defines the
column
  delay(dur); // delay for the screen to get clear
  lcd.clear();
}

// Function to display data on a LCD with two rows
void displayData(int pos1, String data1, int pos2, String data2) {
  lcd.clear(); delay(1);
  lcd.setCursor(pos1, 0);  lcd.print(data1);  // row1 -> pos1 defines the
column
```

```
  lcd.setCursor(pos2, 1);  lcd.print(data2);  // row2 -> pos2 defines the
column
}

// Function to send an SMS to a predefined phone number and a desired message
void SendSMS(char* Message) {
  sim808.sendSMS(phone1, Message); // sends SMS to the pre-defined phone
number 1
  sim808.sendSMS(phone2, Message); // sends SMS to the pre-defined phone
number 2
  // can edit these phone numbers as per necessary
}
```

6. Testing and Results

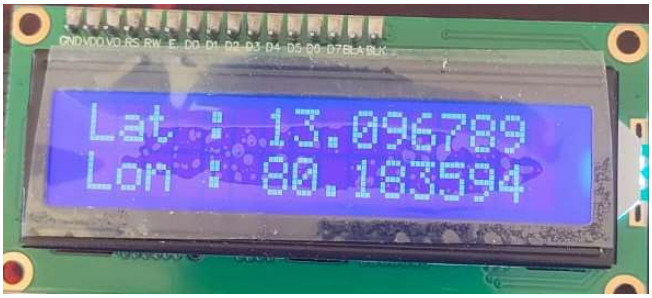
Welcome address during the start of program

Message denoting the initialisation of modules

After Initialisation,

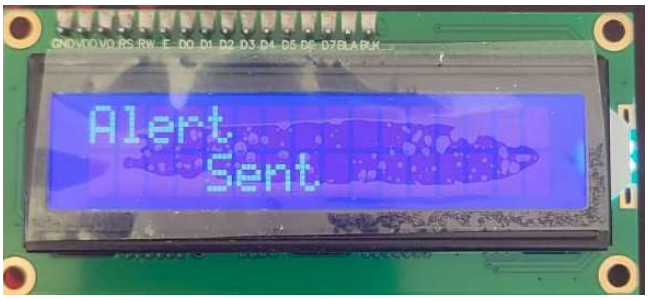
When push button is not pressed, gps data is driven and displayed

Message for getting gps and data



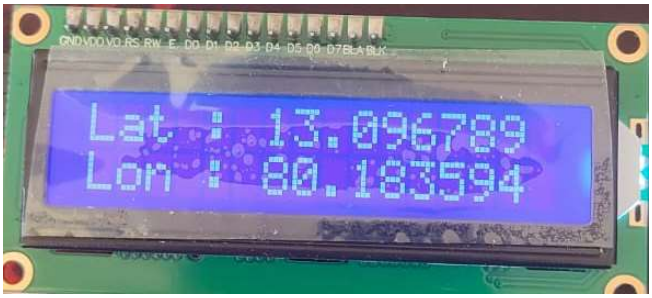
When pushbutton is pressed,
Alert mode is triggered, sms is sent to predefined phone numbers

Message denoting alert sent



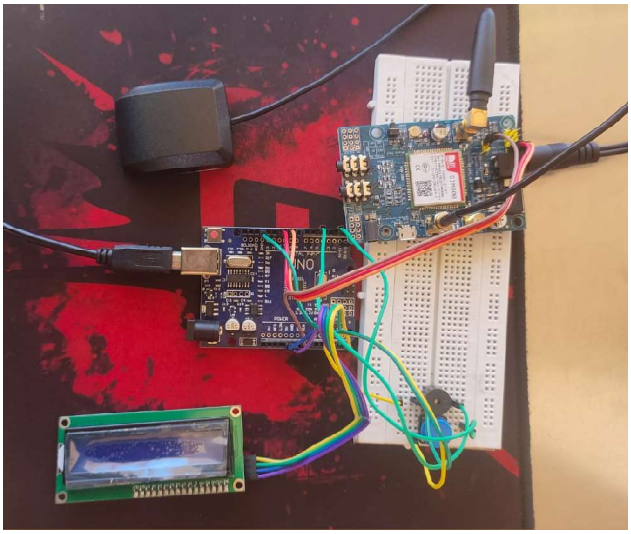
SMS received by pre-defined phone number

In alert mode, gps is displayed in lcd continuously until STOP message arrives



When “STOP” message is sent by predefined phone number,

Buzzer gets turned off and exits alert mode



Project Image

7. Future Work

This project can be further enhanced by implementing map functions instead of SMS which will be able to send the co-ordinates in a continuous manner and help the people to track them easily.

And incorporating emergency services such as police, fire services, etc to help them address their situation in a quicker manner, even if the people they contact is far away.

Moving to better quality equipment can provide better response and service.

Improving quality of user interface can help user to customize the data using IOT will make it easier to use.