Security Panel System Using Arduino

Trushil Patel, *CS 807 Student, University of Regina*
Akash Singh, *CS 807 Student, University of Regina*

***Abstract*—Security is one of the most crucial things to act as a shield to any organization. We are focusing on digital locking systems which works on the concept of interactive hardware. In this research project, we will suggest some modifications to an existing system. The aim is to improve efficiency and extend the usability of the system. We have chosen a security panel system developed by Mert Arduino[1]. The system serves the purpose of security to any organization. The content flow of this article goes like this. First, we will provide a background of similar existing projects. Second, an overview of the new design which includes additional components required. Third, discuss the working of the new system in detail. Fourth, provide a user manual which includes directions to use the system. Fifth, develop a prototype according to the new proposed design. Sixth, test the prototype. Seventh, analyze the results obtained from testing the prototype. At last, provide directions to extend this project in the future.**

I. Background

There was a time when there was no other option than including a latch on the door to open and close it. Using physical components like wood or metal for the keys was the only option to make a locking system function. With time technology evolved and the new keys were invented which improved the security aspect of the system. In this modern era, keys like RFID tags, biometrics, passcode, etc. are widely used in a big organization. The main advantage of using a digital locking system over traditional locking system is that its difficult to make a copy of the key. For example, it is almost next to impossible to make the same eye for a retina scan. Fig. 1 shows some digital locking systems available at The Home Depot which is the leading company in North America for household products. These locking systems make use of unique identities as key. They seem to be the best choice nowadays when a person thinks of building a new house. However, these products do not fit into everyone's budget. The systems shown in Fig.1 were the cheapest. So, we are more concerned about the cost of the product. The purpose of this research is also to design a locking system which is far cheaper than those in Fig1. In the next section, we will provide an overview of the existing system which we have chosen for our research. And we will suggest some modifications to transform it as a competitor to the system available in the market.

There are many do-it-yourself projects on door locking system using Arduino. Fig. 2 shows a door locking system developed by Ioannis[9]. The main component in this locking system was the ultrasonic sensor. The door opens automatically when any person comes close to the door. The idea is similar to those locking systems in shopping malls. The ultrasonic sensor measures the distance every 3 seconds. Each reading after 3 seconds is input for the password. The door opens when the password matches with the one predefined into the system. The system is OK to use but does not provide enough security.

Another do-it-yourself project by Jayesh Nawami[8] showed in Fig. 3 is also an attractive one. Source to enter the password is different here. In this project, a keypad is used to input password. This project is similar to the conventional digital locking system which is available in the market. The security level of this system is much better. However, it gives an idea of the length of the password to the intruder.

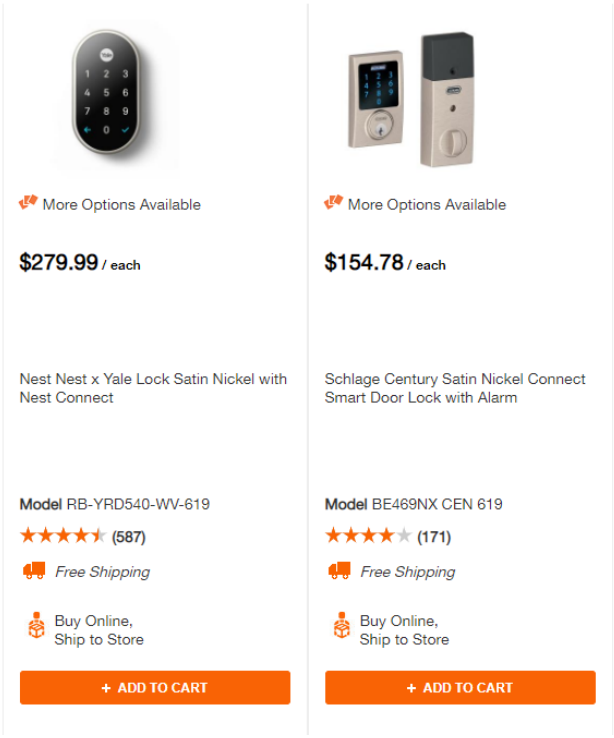

Fig. 1. Digital locks available at The Home Depot[6]

II. Existing System

The Security Panel System developed by Mert Arduino was a simple door locking system. The purpose of choosing this project for our research was to suggest modifications to improve the design and make it more usable. Adding additional features to this system will make it more attractive and be a good competitor in the market. The system developed by Mert Arduino serves the purpose of security in a logical way. Which means that some physical movements should be included to make it a complete working model. In this research, we are proposing a new design which we will discuss in the next section.

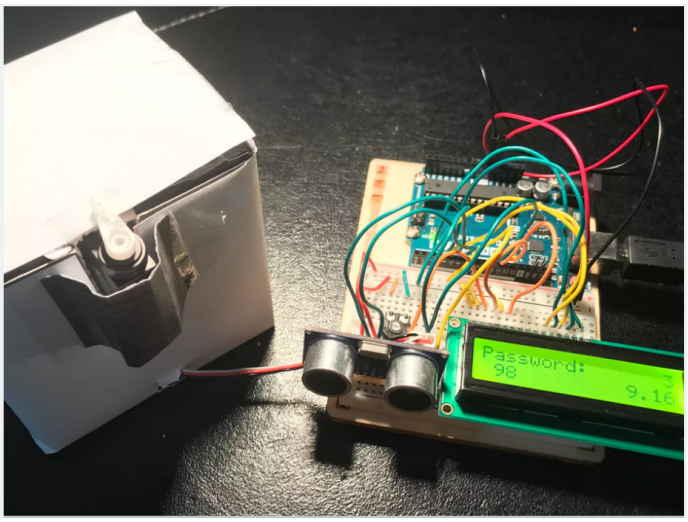

Fig. 2. Locking system using Arduino by Ioannis[9]

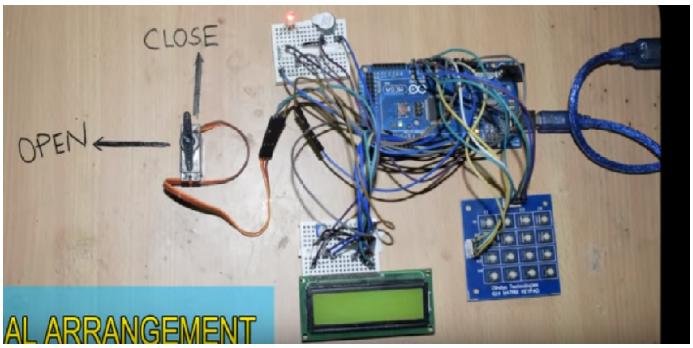

Fig. 3. Door locking system developed by Jayesh Nawami[8]

III. Suggested Modifications

Fig. 4 shows the project by Mert Arduino[1] which we selected for this research.There are many aspects where the existing system lacks in providing more usability. First, it does not have a sort of physical movement for the locking system. Second, there are no limitations for the user to enter the password. The smart home is the new trend in this modern era. We cannot integrate the system with the smart home as it does not have any wireless connectivity. In order to tackle the limitations of the existing system, we will suggest modifications. For the physical movement of the door lock, we will use a stepper motor. The user will get only three attempts to enter the password. An alarm will go on if the person fails to enter the correct password in those three attempts. For wireless connectivity of the system, we will use a Wifi module to establish a client-server based architecture which we will discuss in the next section.

IV. Design Description

A. Overview

Fig. 5 gives an idea of how the components would be interacting in a real-world scenario. Generally, it shows the communication protocols between the components. Arduino Mega and ESP8266 WiFi module will communicate with each

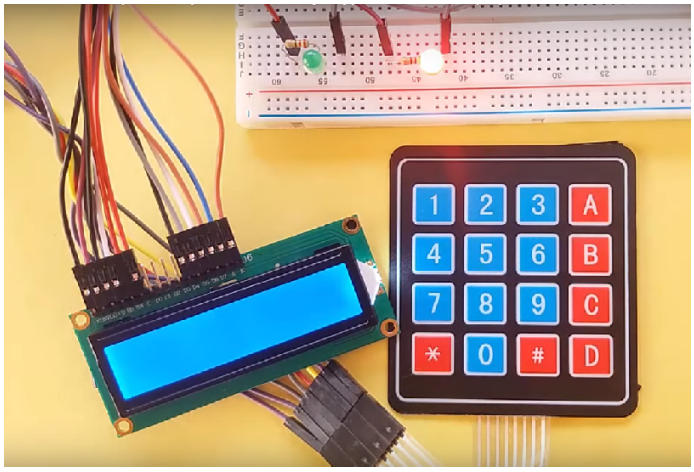

Fig. 4. Project by Mert Arduino[1]

other through serial communication. The ESP8266 WiFi module can operate in three modes. Over here we are configuring it as a web server. The access point in Fig. 5 can be a personal hotspot or the home router. The client can be a mobile phone or a laptop which will be connected to the same access point. First, the client sends an HTTP request to the server that is the ESP8266 WiFi module. Then, the server interprets the request and pass the appropriate changes to the Arduino Mega through serial communication. After the Arduino Mega receives the information from the ESP8266 WiFi module it sends confirmation in the form of the current status of the system. Meanwhile, the ESP8266 Wifi Module responds to the client through HTTP response. So, this illustrates the two-way communication between Arduino Mega and ESP8266 WiFi module. The idea to make the ESP8266 WiFi module function as a web server was taken from Random Nerd Tutorials[5].

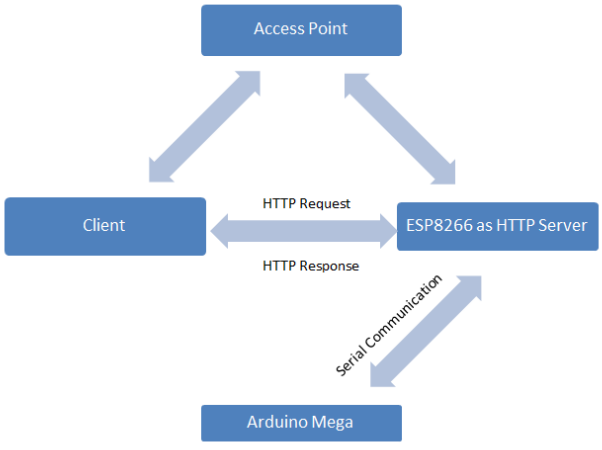

Fig. 5. Block diagram showing the interaction between entities

B. Detailed Description

1) Connections: In this section, we will discuss the design of the system in more detail. Components used in the proposed

design are listed below:-

- 1 x Arduino Mega 2560
- 1 x Active Buzzer or piezo speaker
- 1 x Red LED
- 1 x Green LED
- 1 x Blue LED
- 1 x Keypad(4 X 4)
- 1 x 28BYJ-48 Stepper Motor
- 1 x ULN2003A- Stepper Driver Board
- 3 x 220ohm Resistors
- 1 x Loudspeaker
- 1 x LCD Display(16 X 2)
- 1 x 10K ohm Potentiometer
- 1 x ESP8266 Wifi Module
- 1 x Breadboard
- Jumper Wires

Fig. 7 and Fig. 8 in the appendix section shows the schematic and circuit diagram of the connections. The connections are a bit complex as it involves lots of components. Please refer to the GitHub[7] repository of this project for proper guidelines on how to upload the code to Arduino Mega and ESP8266Wifi Module. Writing the sketch for Arduino Mega was done in different phases. First, the basic locking system without any serial communication with the ESP8266 WiFi module was built. Then, after testing each functionality of the system ESP8266 WiFi module was introduced. The serial communication between Arduino Mega and ESP8266 WiFi module was a bit challenging. But in the end, we made it to the point where everything was working as expected.

2) Working: The functioning of the system is simple and straightforward. The system will turn on as soon as power supply is provided. Initially, it will prompt to enter the password to open the lock. Now, the person will enter the password through the keypad followed by D provided on the panel. The person will only have three attempts to enter the password. If the person fails to enter the correct password in the last attempt, the alarm will go on. If the password is correct, the door opens. The system also speaks out voice commands through the speaker. For example, if the password is incorrect it says-try again. The door can be locked by pressing the * key on the keypad. Other than this the door lock can also be controlled from the web browser. For this, the client device and the security panel system should be in the same network. Which means the client device and the security system should be connected to the same access point. To access the web portal of the security system the person needs to enter the IP address of the web server in the URL on the client device. The state of the security system can be monitored and controlled through this web portal. The client sends an HTTP request to the server each time any button is pressed. The web page gets updated every three seconds which keeps track of the status of the system.

C. Use

The design of the system proposed by us is a universal locking system which has many applications. According to us, this system is the perfect option for an office or home use. The system is simple and easy to use. The device should get constant power supply of 9V. Using an external adapter for power supply is highly recommended. To open the door enter the password through the keypad and then press D. To close the door press * on the keypad. To adjust the contrast of the LCD display, rotate the knob provided at the back of the device. To control the device through webpage enter the IP address of the web server which was configured during development. Now, it is mandatory that the device should be connected to the same access point. Press the OPEN/CLOSE button on the webpage to open or close the door. To turn off the alarm press the OFF button on the webpage.

V. Evaluation

A. Overview

During the entire development cycle of this design, we followed a spiral model. This lead to many changes after every phase of the development cycle. At last, we developed a prototype according to the final design of which we will talk in the next section. During the testing of the prototype, we used the serial monitor of the Arduino IDE. Location for testing the prototype was our home.

B. Prototype

Fig. 6 shows the prototype which we developed according to the proposed design. For the base of the prototype, we used a cardboard sheet. According to the measurements of the components, we created space on the cardboard to mount on it. We managed to fit the entire circuit on the back of the cardboard sheet.

Fig. 6. Prototype developed according to the proposed design

C. Testing and Results

In this section, we will be testing our prototype in a real-world scenario. Table I, II, III, IV, V, VI and VII demonstrates how the device is reacting actually.

TABLE I
TEST SUITE 1

Testing the Buzzer						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test whether the buzzer goes ON after the number of attempts are exceeded	The user should enter wrong passcode in each attempt	When the LCD screen display "Exceeded"	The buzzer should go ON	The buzzer started making noise	PASS
2	To test whether the buzzer goes OFF when the OFF button is pressed on the webpage of client	The client should be connected to the same access point	When the OFF button is pressed on the webpage	The buzzer should go OFF and the system should reset	The buzzer goes OFF and the system resets	PASS

TABLE II
TEST SUITE 2

Testing the Speaker						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test the voice output from the speaker	The system should be ON	When the user enters wrong password	The sound should come from speaker saying "Try Again"	The speaker sounds saying "Try Again"	PASS

TABLE III
TEST SUITE 3

Testing the Potentiometer						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test the whether the contrast of the LCD screen reduces when the potentiometer is rotated anti-clockwise	The system should be ON	When the potentiometer is rotated anti-clockwise	The contrast of the LCD screen should decrease	The contrast of the LCD screen decreases	PASS
2	To test the whether the contrast of the LCD screen increases when the potentiometer is rotated clockwise	The system should be ON	When the potentiometer is rotated clockwise	The contrast of the LCD screen should increase	The contrast of the LCD screen increases	PASS

TABLE IV
TEST SUITE 4

Testing the 4X4 Keypad						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test the whether the LCD screen display '*' when any key on the keypad is pressed	LCD screen should display "Enter password"	When any key on the keypad is pressed	Consecutive '*' should be added when any key is pressed on the keypad	Consecutive '*' is added when any of the key is pressed on the keypad	PASS

TABLE V
TEST SUITE 5

Testing the Stepper Motor						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test the whether the stepper motor rotates anti-clockwise when the lock is opened	The lock should be CLOSE	When correct passcode is entered followed by 'D'	The stepper motor should rotate 180 degree anti-clockwise and "Verified" should be displayed on LCD screen	The stepper motor rotates 180 degrees anti-clockwise and "Verified" is displayed on LCD screen	PASS
2	To test the whether the stepper motor rotates clockwise when the lock is closed	The lock should be OPEN	When user press '*' on the keypad	The stepper motor should rotate 180 degree clockwise and "Door Locked" should be displayed on LCD screen	The stepper motor rotates 180 degrees clockwise and "Door Locked" is displayed on LCD screen	PASS

TABLE VI
TEST SUITE 6

Testing the LEDs						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test whether Red LED glows when the door is lock	The lock should be CLOSE	When the lock is in close state and "Enter password" is displayed on LCD screen	The Red LED should glow when the lock is closed	The Red LED glows when the lock is closed	PASS
2	To test whether Red LED goes OFF when the door is unlocked	The lock should be CLOSE	When the user enters the correct passcode on the keypad followed by 'D'	The Red LED should go OFF	The Red LED goes OFF when the the lock is opened	PASS
3	To test the whether Green LED glows when the door is unlocked	The lock should be CLOSE	When the user enters the correct passcode on the keypad followed by 'D'	The Green LED should glow when the lock is opened	The Green LED glows when the lock is opened	PASS
4	To test whether Green LED goes OFF when the door is locked	The lock should be OPEN	When the user press '*' on the keypad	The Green LED should go OFF when the lock is closed	The Green LED goes OFF when the lock is closed	PASS
5	To test whether Blue LED glows when the user exceeds the number of attempts to enter the passcode	The lock should be CLOSE	When the user exceeds the number of attempts to enter the passcode	The Blue LED should glow when the user exceeds the number of attempts to enter the passcode and the buzzer should start making noise	The Blue LED glows when the user exceeds the number of attempts to enter the passcode and the buzzer started making noise	PASS
6	To test whether Blue LED goes OFF when the user press the OFF button on the webpage	The lock should be CLOSE and buzzer should be making noise	When the user presses the OFF button from the webpage	The Blue LED should go OFF when the user presses OFF button on the webpage and the buzzer should stop making noise	The Blue LED goes OFF when the user presses OFF button on the webpage and the buzzer stoped making noise	PASS

TABLE VII
TEST SUITE 7

Testing the serial communication between Arduino Mega and ESP8266 Wifi Module						
Test Case ID	Test Case Objective	Prerequisite	Action	Expected Output	Actual Output	Result
1	To test the serial communication between Arduino Mega and ESP8266 Wifi Module	The lock can be OPEN or CLOSE	When the user press any button on the webpage	The message received from the ESP8266 Wifi module should be printed on serial monitor	"open" message is printed as received string in serial monitor when the user presses OPEN button on the webpage	PASS

D. Assessment

After testing the prototype developed according to the proposed design we can say that the system is working as expected. Monitoring and controlling through mobile phone or laptop makes it more attractive. The two-way communication between Arduino Mega and ESP8266 Wifi module made sure that the status on the webpage is updated on time. The voice commands were also audible enough to the person. Messages displayed on the LCD screen was working according to the logic. Overall, the new system is more efficient and extends the usability of the existing system developed by Mert Arduino.

E. Workload Distribution

The tasks of this research project was divided among the team members as follows:

- Trushil Patel
  - Documentation, coding and construction
- Akash Singh
  - Designing, testing and coding.

F. Scheduling

Several milestones were set to achieve the goals of this research. We eliminated some of them as they were not required. The list below outlines the timeline of developing the project:

- **Milestone #1:**
  - Gathering the required components
  - Due March 11th
  - This milestone was achieved as it was straight forward.
- **Milestone #2:**
  - Complete assembling the items
  - No coding done yet
  - Due March 13th
  - This milestone was also achieved.
- **Milestone #3:**
  - Complete construction of the existing system with coding
  - Testing the prototype
  - Due March 21st
  - This milestone was also achieved.
- **Milestone #4:**
  - Text-to-Speech functionality is added
  - IR Remote control interface is implemented
  - Alarm is implemented and tested
  - Motor is integrated
  - Due April 1st
  - This milestone was achieved. But, we eliminated the IR Remote control interface because the alarm could be controlled through the web page.
- **Milestone #5:**
  - Arduino Security Panel System Panel prototype works with no failures
  - Try to implement remote access to the device through WIFI
  - Due April 9th
  - This milestone was also achieved and we were able to control and monitor the system through web interface.
- **Goal Milestone :**
  - Testing of final prototype
  - Complete documentation of the implemented design
  - Due April 17th
  - This milestone was also achieved and the report of the research was submitted on time.

VI. Conclusion

In this research project, we proposed a design for a security panel system using Arduino. First, we talked about the commercial products available in the market. Second, we talked about different do-it-yourself projects using Arduino. Third, we chose an existing project and suggested changes to the design. After the changes were made we thoroughly performed the testing of the prototype. User Manual for the new design was also given in this report to explain in brief how the new system works. Milestones were met properly during the design phase. In the end, we accomplished our goal of making a similar system at a lower cost with more efficiency and usability.

VII. Future Work

As we have already achieved the milestones for the project, we do not end our research at this point. Having that said we would still like to develop this project and enhance it to become more efficient and user-friendly. We have a few ideas in mind like adding GSM to the current design. GSM will help directly send the notification of the state of the lock to the cell phone of the user. We are also thinking of adding another feature in which the user will have the power to change the password. We can also use IR sensor to improve security. Reason for adding this will be that in the case of failure of intranet connection there will be another method to turn off the alarm system. Some database on the back end of the web portal can be created to keep track of the people entering or leaving the premises.

APPENDIX A
DIAGRAMS AND PICTURES

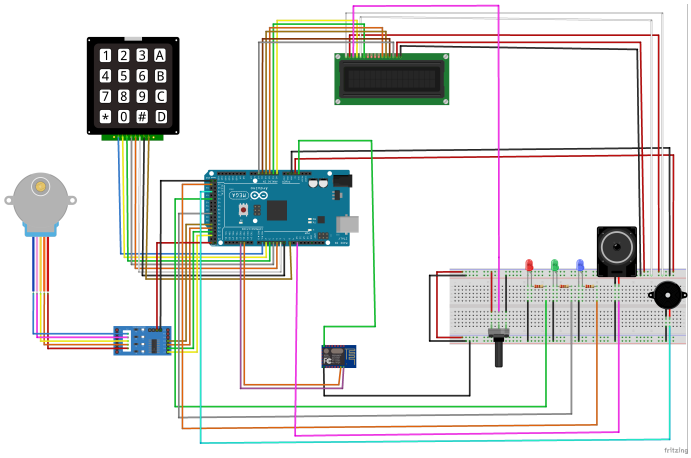

Fig. 7. The schematic of the proposed design

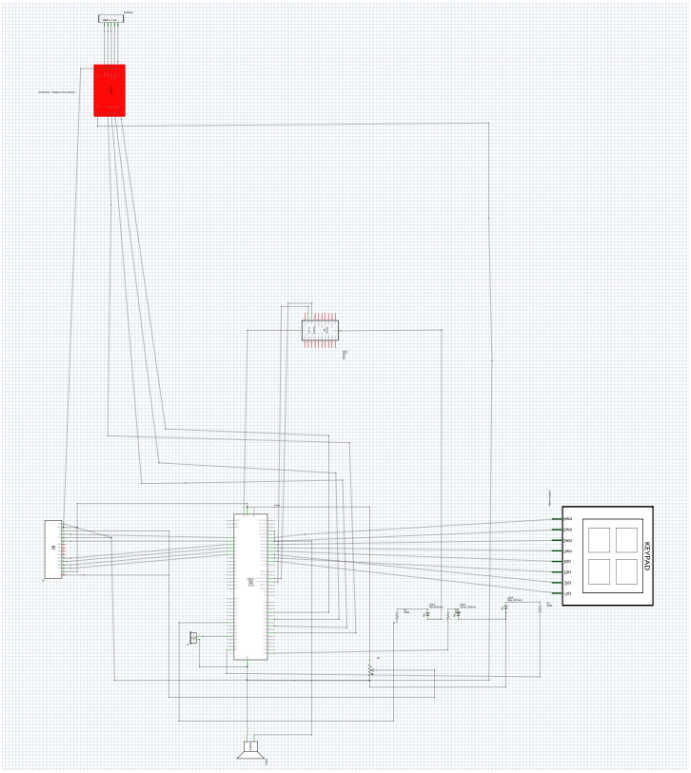

Fig. 8. The circuit diagram of the proposed design

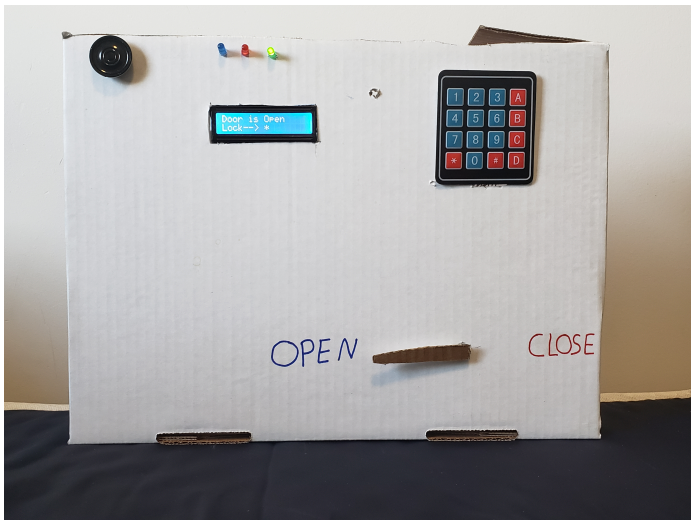

Fig. 9. The front view of the prototype in open state

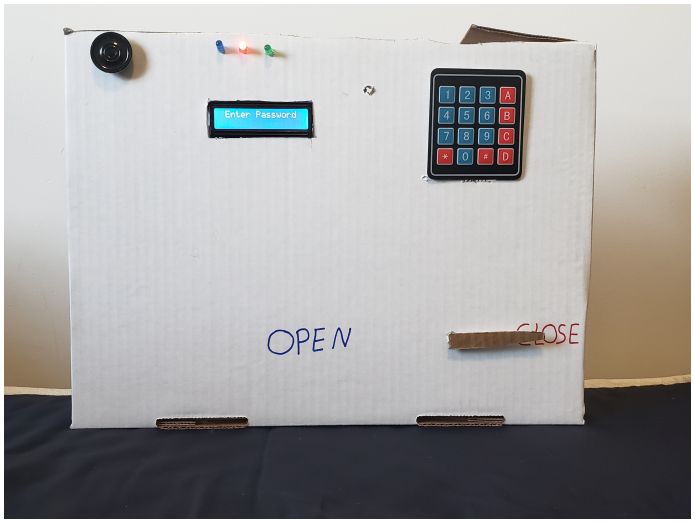

Fig. 10. The front view of the prototype in close state

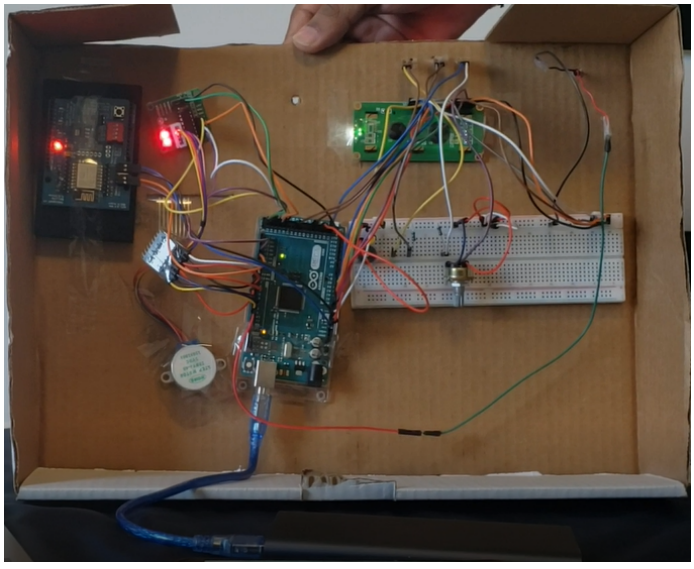

Fig. 11. The back view of the prototype

APPENDIX B
SOURCE CODE FOR ARDUINO MEGA 2560

```
/* This sketch is developed for 2-way serial
    communication between the Arduino Mega
    and ESP8266 Wifi Module */
/* This sketch just demonstrates the main
    part of the code. The whole code can be
    found at github repository of this
    project*/
/*Declare strings for PCM voice file */
/*copy the string from the github repository
    and paste it between the curly brackets*/
const unsigned char enterpassword[] PROGMEM =
    { (paste here)};
const unsigned char verified[] PROGMEM =
    {(paste here)};
const unsigned char tryagain[] PROGMEM =
    {(paste here)};


/*Declare object of Stepper class for using
    stepper motor */
Stepper
    small_stepper(STEPS,motorpin1,motorpin3,motorpin2,motorpin4);
int Steps2Take;

void setup(){

  /*Set the baud rate of TX0 and RX0 pin of
     Arduino Mega to 9600 */
  Serial.begin(9600);
  /*Set the baud rate of TX2 and RX2 pin of
     Arduino Mega to 9600 */
  Serial2.begin(9600);
}

/*Custom function to reset the system*/
void reset()
{
  Num_Of_Attempts=3;
  alarmstate=false;
  pos=0;
  curpos=0;
  lockstate=0;
  Num_Of_Attempts=3;
  match=false;
  lcd.clear();
   lcd.setCursor(0, 0);
   empty(epassword);
   whichKey='\0';
   digitalWrite(buzzer,LOW);
   digitalWrite(blueled,LOW);
}

void loop(){

  /*Send system state to ESP8266 Wifi module*/
  if(millis() % 10 ==0)
  {
    if(lockstate==0 && alarmstate==false)
    {
     Serial2.write("11");

    }
    else if(lockstate==1 && alarmstate==false)
    {
     Serial2.write("01");
    }
    else if(lockstate==0 && alarmstate==true)
    {
     Serial2.write("10");

    }

  }

  /*Receive data from ESP8266 Wifi module*/
  String IncomingString="";
  boolean StringReady = false;

  while (Serial2.available()){
    IncomingString=Serial2.readString();
    StringReady= true;
  }

  /*print received data from ESP8266 Wifi
     module to the serial monitor for
     testing*/
  if (StringReady)
  {
    Serial.println("Received String: " +
       IncomingString);
  }

/*Trigger actions after receiving data from
    ESP8266 Wifi module*/
if(IncomingString.substring(0,10)=="open"
    && lockstate==0)
{

  whichKey= '*';
  lockstate=1;

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("*** Verified ***");
  setLocked(0);
  startPlayback(verified, sizeof(verified));
  delay(1000);
  stopPlayback();
  Serial.println("OPEN DOOR...!!!");
  small_stepper.setSpeed(500);
  Steps2Take= 1024;
  small_stepper.step(Steps2Take);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Door is Open");
  lcd.setCursor(0, 1);
  lcd.print("Lock--> *");
}

if(IncomingString.substring(0,10)=="close"
    && lockstate==1)
{
  reset();
  Serial.println("CLOSE DOOR...!!!");
  small_stepper.setSpeed(500);
  Steps2Take= -1024;
  small_stepper.step(Steps2Take);
  setLocked(1);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(" Enter Password");
  startPlayback(enterpassword,
      sizeof(enterpassword));
```

```
    delay(1000);
    stopPlayback();
  }

  if(IncomingString.substring(0,10)=="buzz"
      && alarmstate==true)
  {
    Serial.println("Turn OFF Buzzer...!!!");
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(" Enter Password");
    startPlayback(enterpassword,
        sizeof(enterpassword));
    delay(1000);
    stopPlayback();
    digitalWrite(buzzer,LOW);
    digitalWrite(blueled,LOW);
    reset();
  }


  if(Num_Of_Attempts==0 && alarmstate==false)
  {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Exceeded");
    alarmstate=true;
  }

  if(alarmstate==true && lockstate==0)
  {
    digitalWrite(buzzer,HIGH);
    digitalWrite(blueled,HIGH);
  }


  if(lockstate==0 && Num_Of_Attempts>0)
  {
    lcd.setCursor(0, 0);
    lcd.print(" Enter Password");
    match=false;
  }

  /*Read which key is pressed with getKey*/
  whichKey = myKeypad.getKey();

  if(whichKey == '*' && lockstate==1 )
  {

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("*** Door Locked ***");
    setLocked (true);
    small_stepper.setSpeed(500);
    Steps2Take= -1024;
    small_stepper.step(Steps2Take);
    delay(2000);
    reset();
    delay(2000);
    startPlayback(enterpassword,
        sizeof(enterpassword));
    delay(1000);
    stopPlayback();
   }
  if(whichKey == 'D' && Num_Of_Attempts>0)
  {

    Num_Of_Attempts--;

    for(int i=0;password[i]!='\0';i++)
    {
      if(password[i]==epassword[i] && pos<5)
      {
        match=true;
      }
      else
      {
        match=false;
      }
    }

    if(match)
    {
      setLocked (false);
      lockstate=1;
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("*** Verified ***");
      startPlayback(verified,
          sizeof(verified));
      delay(1000);
      stopPlayback();
      delay(500);
      small_stepper.setSpeed(500);
      Steps2Take= 1024;
      small_stepper.step(Steps2Take);
      delay(2000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Door is Open");
      lcd.setCursor(0, 1);
      lcd.print("Lock--> *");
      delay(500);
    }
    else
    {
      pos=0;
      curpos=0;
      empty(epassword);
      lcd.clear();
      lcd.setCursor(0, 0);
      setLocked (true);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(" Wrong Password!");
      delay(800);
      startPlayback(tryagain,
          sizeof(tryagain));
      delay(1000);
      stopPlayback();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Left: ");
      lcd.print(Num_Of_Attempts);
      delay(800);
      lcd.clear();
      if(Num_Of_Attempts > 0)
      {
        startPlayback(enterpassword,
            sizeof(enterpassword));
        delay(1000);
        stopPlayback();
      }
    }

}
```

```
  if(whichKey != 'D' && whichKey != '*' &&
      whichKey != '\0')
  {
    epassword[pos]=whichKey;
    lcd.setCursor(curpos,1);
    lcd.print("*");
    curpos++;
    pos++;
  }

 delay(100);
}

//end of the program.
```

APPENDIX C
SOURCE CODE FOR ESP8266 WIFI MODULE

```
/*
   This sketch demonstrates how to set up
       ESP8266WiFi Module as simple HTTP-like
       server.
   This code is not complete the entire code
       can be found at the github repository
       of this project.
*/

void setup()
{
  /*Set the baud rate of ESP8266WiFi to 9600*/
  Serial.begin(9600);

  /*Connect to the access point*/
  Serial.print(F("Connecting to "));
  Serial.println(ssid);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  /*Wait for the ESP8266WiFi module to
      connect to access point*/
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(F("."));
  }

  /* Print the IP address of the server on
      serial monitor */
  Serial.println(WiFi.localIP());

}

void loop()
{

  /* Listen for the string coming from
      Arduino Mega*/
  while (Serial.available())
  {
    delay(20);
    IncomingString=Serial.readString();
    StringReady= true;
  }


  /*Set the status of system on webpage
      according to the received string from
      Arduino Mega*/
if (StringReady)
{
  if(IncomingString[0]=='0')
  {
    lock_status="OPEN";
  }
  if(IncomingString[0]=='1')
  {
    lock_status="CLOSE";
  }

  if(IncomingString[1]=='0')
  {
    buzz_status="ON";
  }
  if(IncomingString[1]=='1')
  {
    buzz_status="OFF";
  }
}



/*Read first line of the HTTP request*/
String req = client.readStringUntil('\r');
Serial.println(F("request: "));
Serial.println(req);

/*Send values to Arduino Mega to change the
    state of system from webpage*/

if (req.indexOf(F("/gpio/0")) != -1)
{
  delay(500);
  Serial.write("open");
}

if (req.indexOf(F("/gpio/2")) != -1)
{

  delay(500);
  Serial.write("buzz");

}
if (req.indexOf(F("/gpio/1")) != -1)
{
  delay(500);
  Serial.write("close");

}

/*Read request from the client if
    available*/
while (client.available())
{
  client.read();
}

/*Send the response to the client*/
client.print(F("HTTP/1.1 200
    OK\r\nContent-Type:
    text/html\r\n\r\n<!DOCTYPE
    HTML>\r\n<html><center><head><meta
    http-equiv=\"refresh\" content=\"5;
    url=http://192.168.43.12\"><style>table,
    th, td {border: 1px solid
    black;border-collapse: collapse;}th, td
    {padding: 5px;}th {text-align:
    center;}.button {background-color:
```

```
      #4CAF50;justify-content: center;border:
      none;color: white;padding: 15px
      32px;text-align:
      center;text-decoration: none;display:
      inline-block;font-size: 16px;margin:
      4px 2px;cursor: pointer;}.button3
      {background-color:
      #f44336;}</style></head>\r\n<h1>Door
      Locking System Portal</h1>\r\nControl
      the Lock From Here:- <br><br>"));


  client.print(F("<table
      style=\"width:100%\"><tr><th>Component</th><th></th><th>status</th></tr>"));
  client.print(F("<tr><td>LOCK</td><td><a
      href='http://"));
  client.print(WiFi.localIP());
  client.print(F("/gpio/0'><button
      class=\"button\">OPEN</button></a>"));

  client.print(F("<a href='http://"));
  client.print(WiFi.localIP());
  client.print(F("/gpio/1'><button
      class=\"button
      button3\">CLOSE</button></a></td><td>"));

  client.print(lock_status);
  client.print(F("</td></tr>"));

  client.print(F("<tr><td>ALARM</td><td><a
      href='http://"));
  client.print(WiFi.localIP());
  client.print(F("/gpio/2'><button
      class=\"button
      button3\">OFF</button></a></td><td>"));

  client.print(buzz_status);
  client.print(F("</td></tr>"));

  client.print(F("</table></center></html>"));

}


//end of the program.
```

TABLE VIII
ASSEMBLY LIST

Label	Part Type	Properties
J1	Piezo Speaker	
LED1	Red (633nm) LED	color Red (633nm); package 0805 [SMD]
LED3	Green (555nm) LED	color Green (555nm); package 0805 [SMD]
LED4	Blue (470nm) LED	color Blue (470nm); package 0805 [SMD]
Matrix keypad1	KEYPAD 4x4	variant variant 1
MOTOR1	28BYJ-48 Stepper Motor	variant variant 1
Part2	Arduino Mega 2560 (Rev3)	type Arduino MEGA 2560 (Rev3)
R1	220ÎÌ' Resistor	resistance 220ÎÌ'; package THT; bands 4; pin spacing 400 mil; tolerance Âś5%
R2	220ÎÌ' Resistor	resistance 220ÎÌ'; package THT; bands 4; pin spacing 400 mil; tolerance Âś5%
R3	220ÎÌ' Resistor	resistance 220ÎÌ'; package THT; bands 4; pin spacing 400 mil; tolerance Âś5%
SPKR1	Loudspeaker	
U1	LCD-16X2	variant silk; characters 16x2; package lcd-16x2
U2	POT	variant -rv16af-20; package pot_alpha_rv16af-20
U3	ESP8266 WiFi Module	variant variant 12E; flippedsmd true; part # ESP8266
ULN2003A - Stepper Driver Board1	ULN2003A - Stepper Driver Board	variant variant 2; pins 11; package board; chip label ULN2003A - Stepper Driver Board; spacing 300mil

REFERENCES

[1] Mertarduinotutorial.blogspot.com. (2017). Security Panel System with using Keypad and LCD. [online] Available at: http://mertarduinotutorial.blogspot.com/2017/01/arduino-tutorial-22-security-panel.html [Accessed 5 Mar. 2019].

[2] Overleaf.com. (n.d.). Tables - Overleaf, Online LaTeX Editor. [online] Available at: https://www.overleaf.com/learn/latex/Tables [Accessed 17 Feb. 2019].

[3] YouTube. (2018). Talking Arduino | Playing MP3 audio with Arduino | Arduino PCM audio without audio or mp3 module. [online] Available at: https://www.youtube.com/watch?v=F28Znry0qcw [Accessed 14 Apr. 2019].

[4] Medium. (2017). Arduino UNO + ESP8266 ESP-12E UART WIFI Shield. [online] Available at: https://medium.com/@manrick01/arduino-uno-esp8266-esp-12e-uart-wifi-wireless-shield-3a39858e5f25 [Accessed 14 Apr. 2019].

[5] Random Nerd Tutorials. (n.d.). Build an ESP8266 Web Server - Code and Schematics | Random Nerd Tutorials. [online] Available at: https://randomnerdtutorials.com/esp8266-web-server/ [Accessed 14 Apr. 2019].

[6] Anon, (2019). [online] Available at: https://www.homedepot.com/p/compare/?errorURL=ProductAttributeErrorViewlangId=-1storeId=10051catalogId=10053prodComp_0=304517994prodComp_1=203814067plpUrl=%2Fb%2FHardware-Door-Hardware-Door-Locks-Electronic-Door-Locks%2FN-5yc1vZc2bdN=5yc1vZc2bd [Accessed 15 Apr. 2019].

[7] Patel, T. (2019). trushil/Arduino-Security-Panel-System. [online] GitHub. Available at: https://github.com/trushil/Arduino-Security-Panel-System [Accessed 16 Apr. 2019].

[8] Arduino Project Hub. (2017). Door Lock System with Arduino. [online] Available at: https://create.arduino.cc/projecthub/jayesh_nawani/door-lock-system-with-arduino-fe95ab [Accessed 16 Apr. 2019].

[9] Arduino Project Hub. (2017). Ultrasonic Lock. [online] Available at: https://create.arduino.cc/projecthub/IoannisD/ultrasonic-lock-f6dd69?ref=tagref$_i d = lockoffset = 15[Accessed16Apr.2019]$.