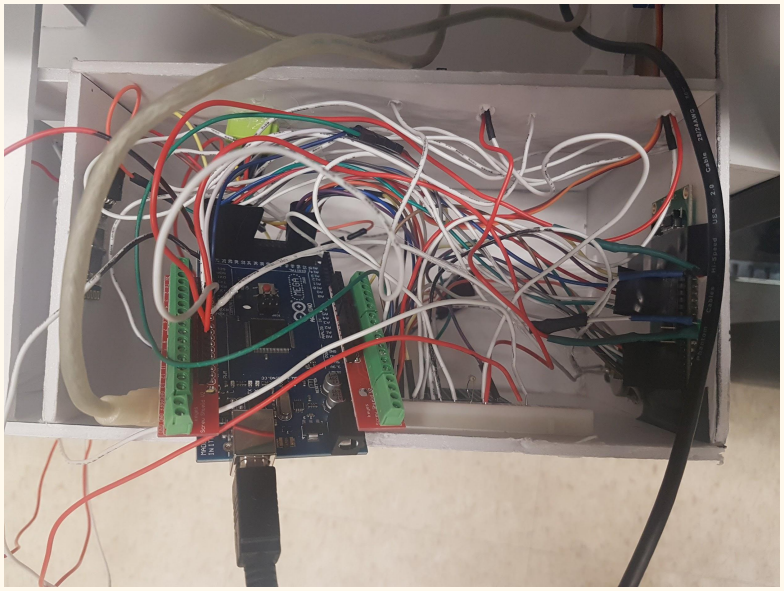
Smart Security Home

By Seong and Yunas



INTRODUCTION

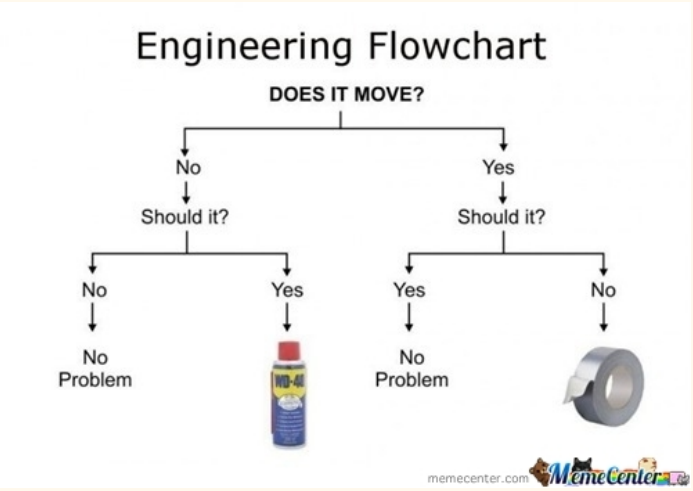
For the purpose of our project we wanted to go for an application that can be involved in the real world. The main focus was home renovation. From the start of stone age to now, home have undergone a significant amount of change on structural, materialistic, and design level. A problem today that is still lacking is technology associated with homes. There are add on features you can get such as the Google Nest, Google Home, Amazon Echo. However those come with a high price, our build is cost efficient and basic which is something we wanted our focus to be. An example of this is keys, keys have been used for centuries, they are a security mechanism for keep your belongings inside of a place and keep others out. Keys can get broken, lost, or stolen, which is a liability of having keys, and with that we designed our house to have a keypad instead. The keypad eliminates the purpose of having keys for your house. Now you can live happily with no hassle to remembering if you brought your keys. With led green for correct pin and red for incorrect and one buzzer for correct pin and 3 short buzz's for incorrect pin. Another issue we seen are porch lights are not automatic, they still have a switch you have to turn on and off, with ours we wanted to make it autonomous, with the help of an ldr. Another feature that comes with the house is ambient lighting, with bluetooth control it useful for when you want to kick back relax and cool off from a stressful day at work. With the android app the

connection is easy and the control are fairly simple with a colour wheel and you can select the colour on the wheel spectrum. Another feature is to have an electronic door, it's handy for when you have a bag full of groceries and you don't need to put them down and can go smoothly in and out of your house. Another feature in the house is your tradition light switch with an led attached to the wiring to show the cost efficiency of using led. Finally the best feature we find to the house is the sonic tripwire. It is designed so that you can leave your belongings in a room and it cannot be accessed without an alarm going off to alert people. All in all the project is a great mix between smart home and smart security features in one tiny barbie house.

Procedure

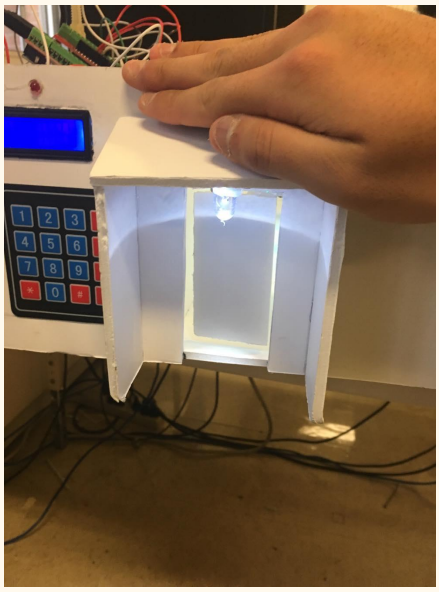
To build your very own smart barbie house, you will need the following:

- 1x Arduino Uno and Arduino Mega
- 1x keypad
- 1x 1602 screen
- 1x ultrasonic sensor
- 1x rgb strip
- 1x ldr
- 2x white led (one for the porch and one for the room)
- 1x red led/1x green led (for the entrance)
- 1x buzzer
- 1x servo
- 1x bluetooth chip
- 1x pushbutton
- resistors



To start our build we need to pick a side and work on it from there. The smart features of the house will be done on the uno while the security features will be done on the mega. The reasoning we are doing this approach is since the bluetooth chip works much better with the uno than the mega in terms of controlling the led strips while the security aspect of the house requires a lot more pins that what the uno contains thus making the mega a better choice. This is going to be a two step procedure.

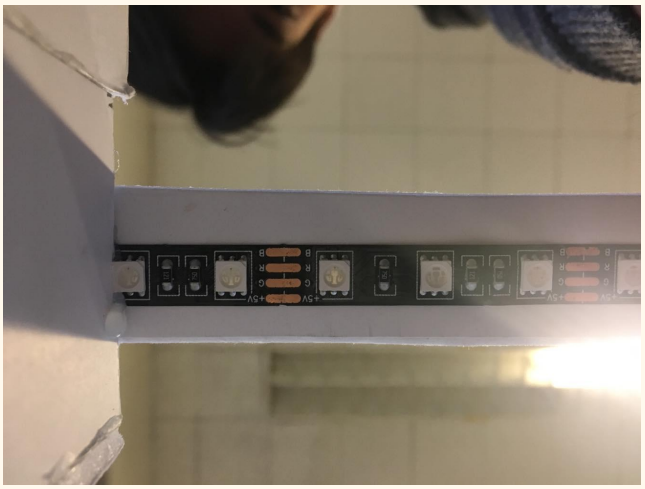
Arduino Uno



1. Smart Porch Light

The porch light of the house is all revolved around the ldr. It is important that we get the ldr wiring right as one pin of the ldr goes to 5v and the other pin goes two ways one is to gnd and the other is to an analog input with a 10k ohm resistor in between. Also having an led that has a reasonable resistor attached to it. For our case we used a 220 ohm resistor for it.

2. Mood lighting

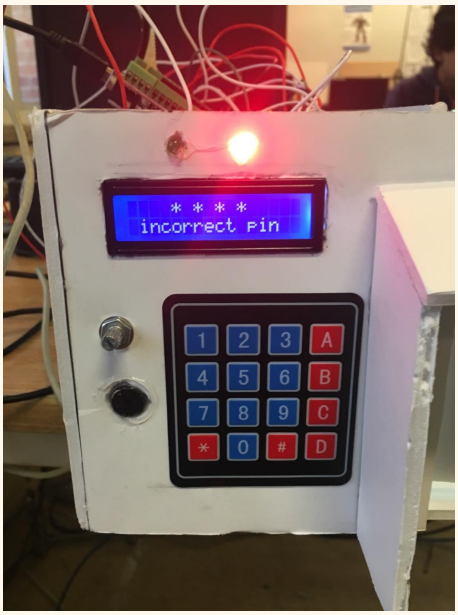


The rgb strip we used only required 5v. We were lucky for having that as most rgb strips are 12V or 9V. Having a 5v strip meant that we did not need transistors and mosfets to power the strips or need any external power as the arduino is only capable of generating 5v at the most. The rgb strips are connected to pwm pins on the arduino for fade control. The bluetooth chip is used to control the lighting of the mood light. This is done through a pre built app available on the play store. The coding aspect is built around the app and the code is based to work around the app. That is all for the mood lighting.

Using the following app:
https://play.google.com/store/apps/details?id=appinventor.ai_addmefaster1.BlueCore_Tech_RGB_LED_CONTROL

Arduino Mega

1. The lock



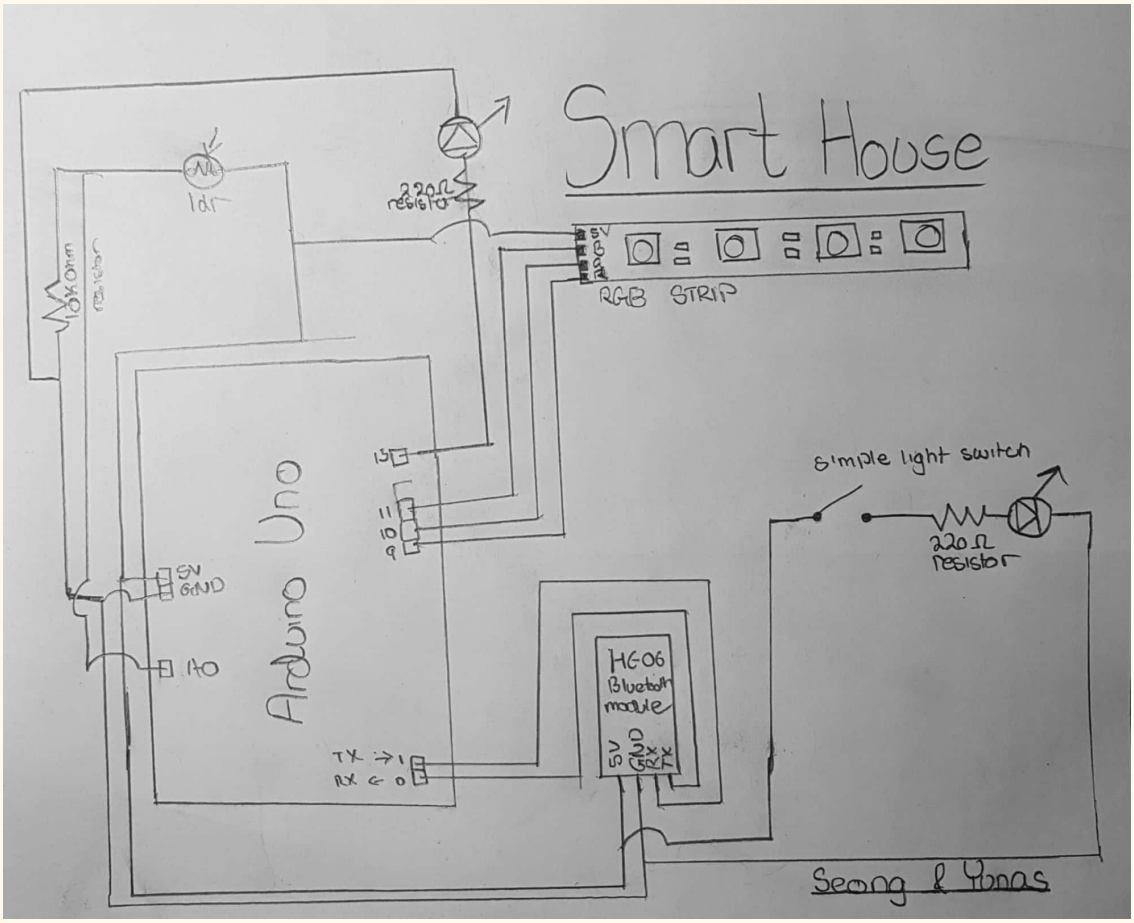
The lock has a few components to it. First is the lcd screen it's attached to the mega. This is used to display information like the correct pin. The keypad is also attached to the mega and is used to type the pin. There is a push button inside the house that is used that will open the door attached to the servo. After that button has been pushed the house goes into lockdown mode. When the house is in the lock down mode if there is something placed in the safe room and an intruder is detected there is a red led that blinks and the buzzer sounds an alarm. There are green led and red led in the lock system green is when the correct pin is entered and red is for when the wrong pin is incorrect. There is also a buzzer attached that is associated with the alarm system, one long buzzer for the correct pin and three short buzzes for the incorrect pin.

2. The safe room

The safe room, is designed to be a place where you keep your belonging. If the house is in the lock it activates the alarm system. When the ultrasonic sensor detects the distance is less than 10 cm, the alarm will sound and will ring the buzzer the red led will flash. This all can be accomplished through using the schematic for to follow the correct wiring pattern

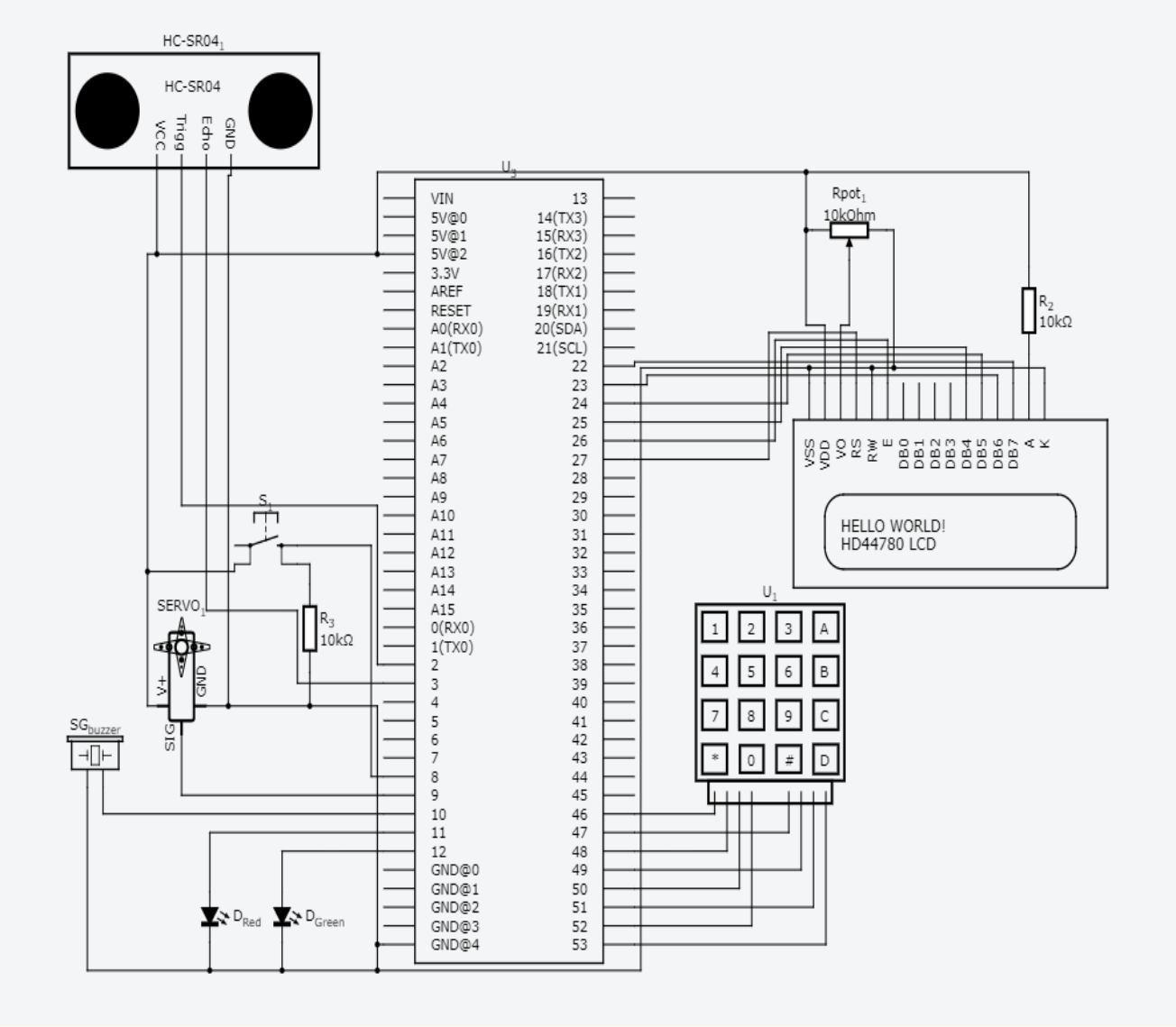
Schematic

The following is the schematic for the **Arduino Uno**:



The following is the schematic for the **Arduino Mega**:

The autodesk circuits tool was useful for the layout schematic of our project. However with the sophisticated wiring of our project and having alot of components on it, doing the sketch by hand would be a difficult unnecessary challenge. Therefore we used circuits.io for the build. Here is the URL for the schematic for a better view
https://circuits.io/circuits/5574732-the-unnamed-circuit/edit



Code

Arduino Uno code

```
#include <SoftwareSerial.h> //library to create a serial connection
#include <Wire.h>
SoftwareSerial mySerial(0,1); //setting bluetooth connection to pins
int PIN_RED = 11;
int PIN_GREEN = 10;
int PIN_BLUE = 9;
String RGB = "";
String RGB_Previous = "255.255.255";
String ON = "ON";
String OFF = "OFF";
boolean RGB_Completed = false;
const int ledPin = 13;
const int ldrPin = A0;
 void setup() {
  pinMode (PIN_RED, OUTPUT);
  pinMode (PIN_GREEN, OUTPUT);
  pinMode (PIN_BLUE, OUTPUT);
  Serial.begin(9600);
  mySerial.begin(9600);
  RGB.reserve(30);
```

```
  pinMode(ledPin, OUTPUT);
  pinMode(ldrPin, INPUT);
  }
void loop() {

  rgb_main(); //function for bluetooth controlled RGB
  porchlight(); //function for porch light
  }
void rgb_main(){
  while(mySerial.available()){  //if bluetooth value detected
    char ReadChar = (char)mySerial.read();  //store character

    if(ReadChar == ')'){
      RGB_Completed = true;
    }else{
       RGB += ReadChar;
    }
  }

  if(RGB_Completed){

     Serial.print("RGB:");
     Serial.print(RGB);
```

```
    Serial.print("     PreRGB:");
    Serial.println(RGB_Previous);

    if(RGB==ON){
      RGB = RGB_Previous;
      Light_RGB_LED();
    }else if(RGB==OFF){
      RGB = "0.0.0";
      Light_RGB_LED();
    }else{
      Light_RGB_LED();
      RGB_Previous = RGB;
    }
    RGB = "";
    RGB_Completed = false;
  }
}
void Light_RGB_LED(){

  int SP1 = RGB.indexOf(' ');
  int SP2 = RGB.indexOf(' ', SP1+1);
  int SP3 = RGB.indexOf(' ', SP2+1);
  String R = RGB.substring(0, SP1);
```

```
  String G = RGB.substring(SP1+1, SP2);
  String B = RGB.substring(SP2+1, SP3);
  Serial.print("R=");
  Serial.println( constrain(R.toInt(),0,255));
  Serial.print("G=");
  Serial.println(constrain(G.toInt(),0,255));
  Serial.print("B=");
  Serial.println( constrain(B.toInt(),0,255));

  analogWrite(PIN_RED,  (255-R.toInt()));//comment if colors are inverted
  analogWrite(PIN_GREEN, (255-G.toInt()));//and uncomment above part.
  analogWrite(PIN_BLUE,  (255-B.toInt()));
}
void porchlight(){  //function for porch light
  int ldrStatus = analogRead(ldrPin); //read LDR value
  //Serial.println(ldrStatus);
  if(ldrStatus<=650){ //if dark turn on LED
    digitalWrite(ledPin, HIGH);
    //Serial.println("LDR is DARK, LED is ON");
   }
  else {  //if bright turn off LED
    digitalWrite(ledPin, LOW);
    //Serial.println("---------------");
```

```
 }
}
```

Arduino Mega code

```
#include <Keypad.h> //library for keypad
#include <LiquidCrystal.h>  //library for lcd display
#include <Servo.h>  //library for servo motor
const int servoPin = 9; //connect servo motor to pin 9
Servo servo;
const byte rows=4;
const byte cols=4;
char keys[rows][cols]={{'1','2','3','A'}, //setting up the keypad layout
                       {'4','5','6','B'},
                       {'7','8','9','C'},
                       {'*','0','#','D'}};

byte rowPins[rows]={46,48,50,52}; //keypad rows connected to these pins
byte colPins[cols]={47,49,51,53}; //keypad columns connected to these pins
char pw[4]; //array to store pin that you enter
int i=0;  //counter to determine how many pins you put in
char password[]={'1','2','3','4'};  //actual password
int a=0;  //counter to determine if pin entered is correct

int buttonPin=8;  //button connected to pin 8
```

```
int buttonState=0;  //variable to store button state

int buzzer=10;  //variable for buzzer

int red=11; //variable for led
int green=12;

const int trigPin = 2;  //variable for ultra sonic sensor
const int echoPin = 3;

bool lock=false;  //boolean to put the house in lock down mode

Keypad keypad=Keypad(makeKeymap(keys),rowPins,colPins,rows,cols); //making the keypad
LiquidCrystal lcd(32, 30, 28, 26, 24, 22);  //declaring which pins the lcd is connected to
void setup(){
  lcd.begin(16,2);
  servo.attach (servoPin); //attach servo to servoPin
  pinMode(buttonPin,INPUT); //declaring I/O
  pinMode(buzzer,OUTPUT);
  pinMode(red,OUTPUT);
  pinMode(green,OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
```

```
}
void loop(){
 buttonState=digitalRead(buttonPin); //read button state
 if(buttonState==HIGH){
  servo.write(180); //if button presses open the door
  lock=true;  //set to lock down mode
  delay(1500);
  servo.write(90);  //close door
 }


  char key=keypad.getKey();  //store the value of key pressed
  lcd.setCursor(4+2*i,0);  //set the cursor on the lcd
  lcd.blink();


  if(key){
   pw[i]=key; //if key pressed store in array
   lcd.setCursor(4+2*i,0);  //move cursor on LCD
   i+=1;  //counter to know how many pins you entered
   lcd.print("*");
   }
   if(i==4){ //if you've entered all four pins
```

```
for(int j=0; j<=3; j++){
  if(password[j]==pw[j]){ //compare password entered
  a+=1; //counter to check if all four pins are correct
  }
}
if(a==4){ //if password correct
  lcd.setCursor(2,1);
  lcd.print("correct pin");
  lock=false; //disalarm lock down mode
  lcd.noBlink();
  servo.write(180);
  tone(buzzer,1000);
  digitalWrite(green,HIGH);
  delay(1000);
  noTone(buzzer);
  delay(1000);
  servo.write(90);
  digitalWrite(green,LOW);
  lcd.clear();

}
else{ //if password is wrong
  lcd.setCursor(1,1);
```

```
    lcd.print("incorrect pin");
    lcd.noBlink();
    digitalWrite(red,HIGH);
    for(int j=0; j<=2; j++){
      tone(buzzer,200);
      delay(200);
      noTone(buzzer);
      delay(200);

    }
    delay(200);
    digitalWrite(red,LOW);
    lcd.clear();
  }

  i=0;
  a=0;
  pw[4];

}

if(lock){ //if in lock down mode
  int distance=sonic(); //get the distance from the ultrasonic sensor
```

```
    delay(100);
    if(distance<10){  //if distance is less then 10cm
      tone(buzzer,500); //buzz
      digitalWrite(red,HIGH);
      delay(200);
      noTone(buzzer);
      digitalWrite(red,LOW);
      delay(200);
    }
  }
}

int sonic(){  //function for ultrasonic sensor
  long duration;
  int distancecm;

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distancecm= duration*0.034/2;
```

```
 return(distancecm);
}
```

Code explanation

There are two main aspects of our code that we have in our final project. The first part is the bluetooth controlled RGB strip which was a huge portion of the arduino Uno code. The second part of the code is the keypad pin which was in the Arduino Mega. The keypad pin code was the underlying base that supported many of the smart home security features such as the servo controlled door, buzzer, LED and the alarm systems.

Bluetooth RGB strip

The main part of the bluetooth RGB strip code was the bluetooth connection and receiving input values from our phones to the Arduino. In order to hook up a bluetooth to the arduino, you had to use a serial communication port. As well, we used a library that would allow us to easily hook up the bluetooth to the Arduino.

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(0,1);
```

The library that we have is called SoftwareSerial.h. The second line of code is to declare that we are going to make a new serial communication via bluetooth called “mySerial” with the TX and RX pins on the Arduino.

```
while(mySerial.available()){
   char ReadChar = (char)mySerial.read();
}
```

This segment of code reads the receiving signal input from the phone. When the Arduino receives a value, it stores it as a character. Then once this character is stored, the Arduino turns on the different colors of the RGB strip based on the value received.

Keypad Pin

The keypad pin code consists of three parts:

1. Entering the pin on the keypad
2. Determining if the pin that you have entered matches with the actual pin
3. Taking actions based on if the pin was correct or incorrect

```
char key=keypad.getKey();

if(key){

  pw[i]=key;

  i+=1;

 }
```

This is the first part of the code. The first line reads the of the key that you pressed and stores it as a character. Once a key is detected, it stores this character into an array and there is a counter for every key that you press because you can only enter four pins

```
if(i==4){

   for(int j=0; j<=3; j++){

    if(password[j]==pw[j]){

    a+=1;

    }

   }
```

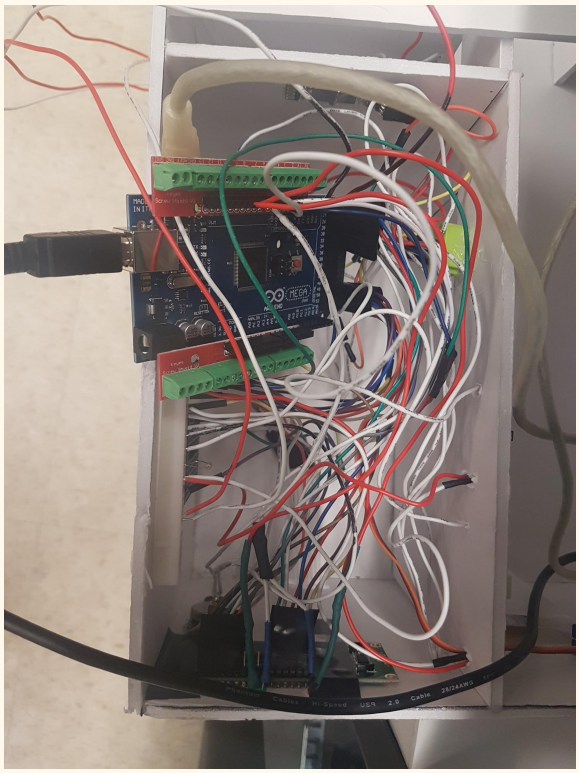
When you have entered all four pins, it goes through a for loop to compare the pin that you entered with the actual pin. There is a counter for every time each pin matches to determine if the pin is correct or not.

```
if(a==4){

    lcd.setCursor(2,1);

    lcd.print("correct pin");
```

```
    lock=false;
    lcd.clear();
  }
```

If the variable ‘a’ has counted up to four, that means that the pin was correct. Once it enters this if statement it disalarms the alarm system where lock is set to false. When lock is set to true, it activates the ultrasonic sensor and receives input values from it.

Reflective Questions



The first problem that we have encountered was the coding aspect of the project itself. When we got started on our smart home security system, we realized that our project required a very long code mainly because of the many features that must be put into it. We couldn’t tackle on the code all at once because it was too big and not structural so we didn’t know where to start. To solve this problem we broke down the code into small parts. We worked on different features of our project one at a time. For instance, we worked on the keypad pin code first, when we finished that, we worked on the porch lighting then the servo door and so on. Each segment of code was very plain as it only has one objective. When we had all of the code for each feature, we blended all the code together. We started to combine everything together so that each segment of code would work together and function as a smart home/ home security. The general insight that we have gained is that it will make it much more easier if you break it down and work from piece by piece.

The second problem that we have encountered was getting the bluetooth feature to function. When we found the right app on our phones that we wanted to use for the bluetooth, we attached it to the arduino Uno as a test and it worked smoothly with the RGB strip. At first, we wanted to have all the components attached to the Arduino Mega so we switched the

connection from Arduino Uno. However once we switch the connection to Arduino Mega, the bluetooth feature stopped working. After lots of time and tests, we realized that the particular app that we are using is only compatible with the Arduino Uno which is why we have two Arduinos in our project. The general insight that we have gained is that many of the apps used for the arduino is not compatible with bluetooth. This problem that we have encountered allowed us to gain deeper insight when working with bluetooth on the Arduino.

The Video:

Last but not least, here is the concluding video that will walk through the project:

https://drive.google.com/file/d/1s248UVr4okc0ghQQMSNjyceSQ_85fDbb/view?usp=sharing