

HAUNTED HOUSE

Muna Abdul
MariaMei Bellsolà
Cristina Molas

Jonathan Chacon
3r de GEDI
2020 - 2021

ELISAVA
Escola Universitària de Barcelona
Disseny i Enginyeria



INDEX

- 3** *Project*
- 5** *Materials*
- 6** *Schematic of electrical connections*
- 8** *Flow diagrams*
- 9** *Guide: how to build this project? Final assembly and code*
- 21** *Final assembly and code*
- 29** *Conclusion*
- 30** *Bibliography*

PROJECT_HAUNTED HOUSE

Description

Haunted House aims to put into practice all the knowledge learned during programming, codifying and Arduino's classes. To achieve this objective, jointly with the American's festive Halloween context, we have designed an enchanted house that interacts with users. Depending on the movements that the user does the house will perform one action or another.

1. Turn on the lights of the house when turning the sign.
2. Open and close the door when ringing the bell of the house and turning the hall's light on.
3. Wake up the lord of the house when touching his tomb and make the curtains move. If the actions take up during night terrific lights will be opened.



Illustration of the house finished without doing any action.



Storytelling

Storytelling of the Haunted House through illustrations, representation of the actions mentioned before in the description.

1



2



3



MATERIALS

Electronic components

Arduino UNO (x1)



Protoboard (x1)



Resistors 220 Ω (x9)



Resistors 10 k Ω (x4)



LEDs (x4)



LED RGB (x1)



Micro Servomotor (x1)



Motor CC (x1)



Photoresistor (x1)



Potentiometer (x1)



Transistor NPN (BJT) (x1)



Diode (x1)



Push Button (x2)



DFPlayer (x1)



Speaker (x1)



Cables



Components of the house

Cardboard to build the main structure of the house and other detail such as the door.

Moss paper to recreate the ground.

Scrapbook paper to simulate wood.

Halloween details, such as spiders, cobweb, ghosts or grass.

Fabric to make the curtains and ghosts

Foam to create shapes, like the tomb.

Tools used

Glue

Paints

Paint brush

Cutter

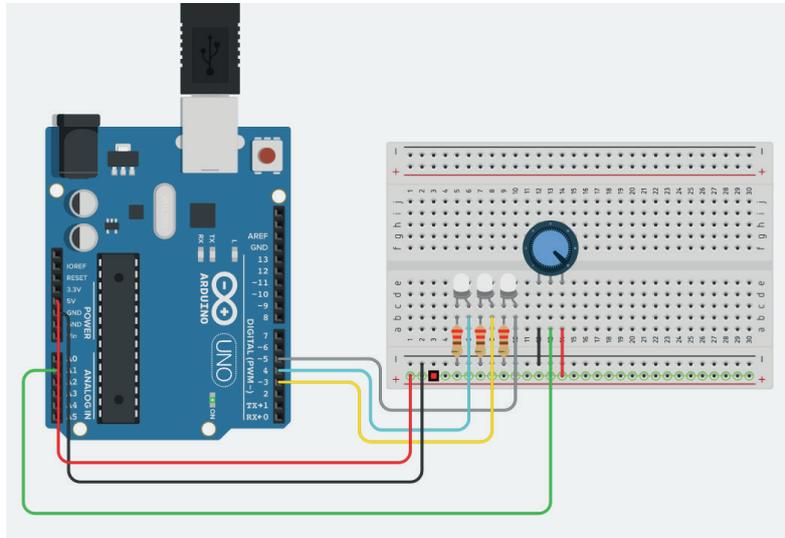
Pliers

Soldering iron

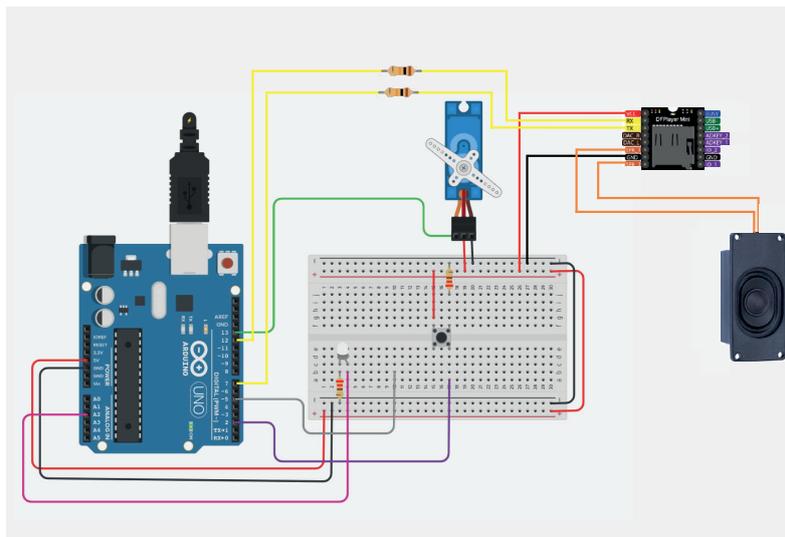
Ruler

SCHEMATIC OF ELECTRICAL CONNECTIONS

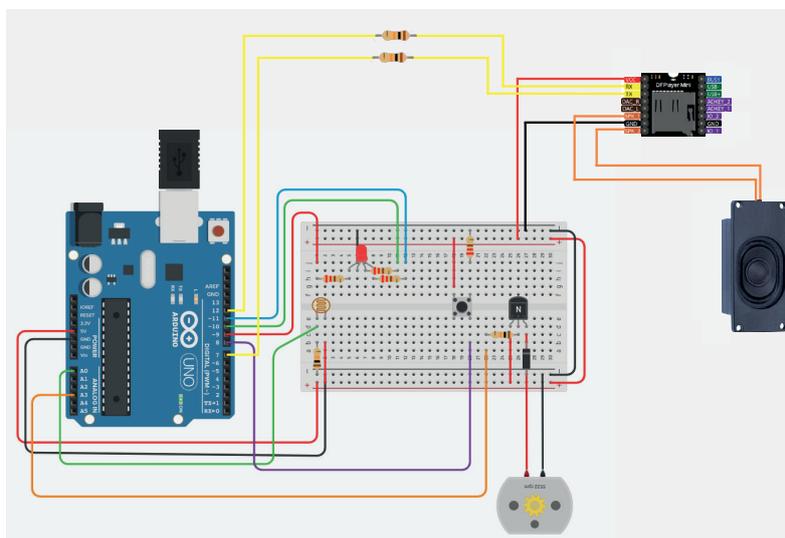
1



2



3



Explanation of the assembly

ACTION 1_Turn on the lights of the house when turning the sign.

To perform the action used three LEDs, which represent the lights of the house, and a potentiometer to turn them on gradually. To make sure the potentiometer is coherent with the house and the context of Halloween we simulated the house entry sign.

ACTION 2_Open and close the door when ringing the bell of the house and turning the hall's light on.

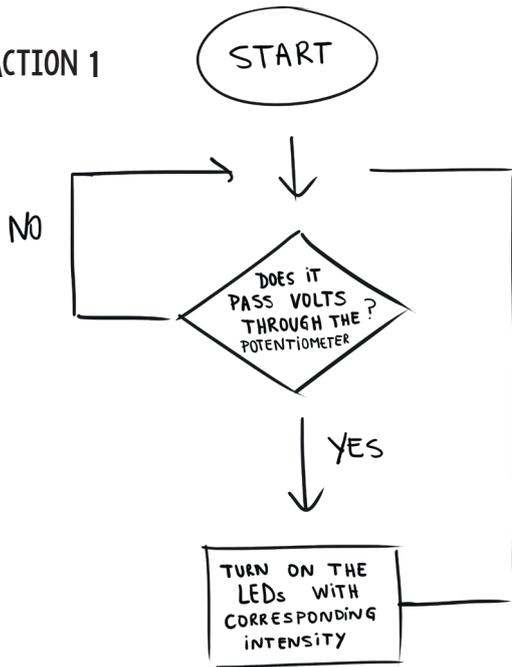
In that case, the idea is that when somebody rings the entry bell this one sounds while opening and closing the door of the house. This movement simulates ghosts and enchantments. At the same time, the same button, turns on the hall light of the house. The ring consists of a Push Button connected to the pin 2 of the Arduino board making the DFPlayer sound, the light of the hall a white LED and the door opens due to a Servo Motor.

ACTION 3_Wake up the lord of the house when touching his tomb and make the curtains move. If the actions take up during night terrific lights will be opened.

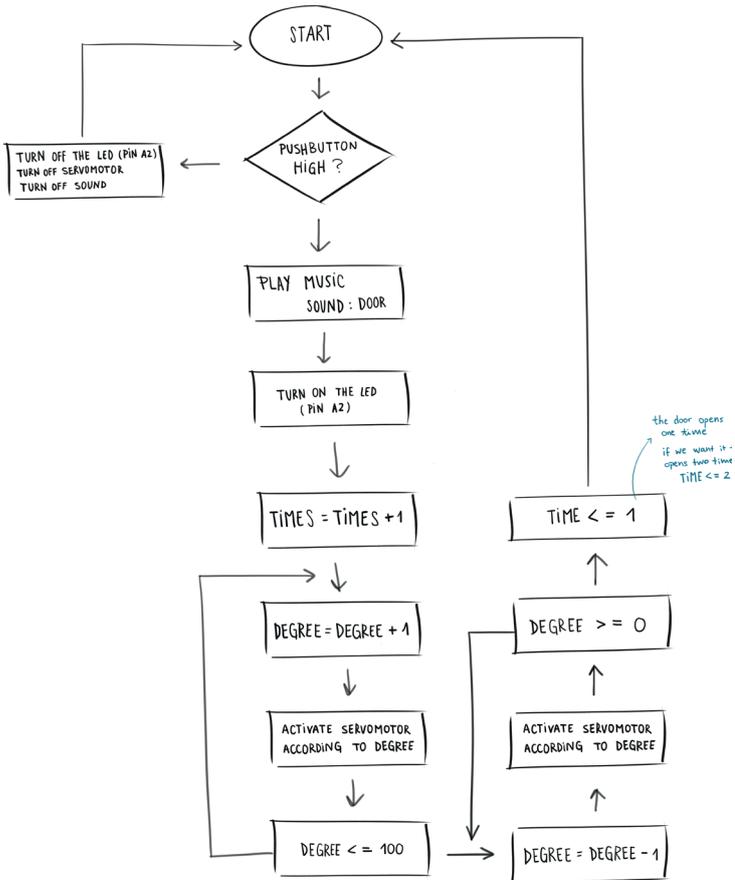
Finally if you touch the tomb located on the garden, that in reality is a Push Button, will activate a motor CC connected to a fan, making the curtains of the house move and fly while playing a terrific laugh played by the DFPlayer. When this happens, if the house is located in a dark room, will make a LED RGB blink with an orange hue. To achieve that we used a photosensor.

FLOW DIAGRAMS

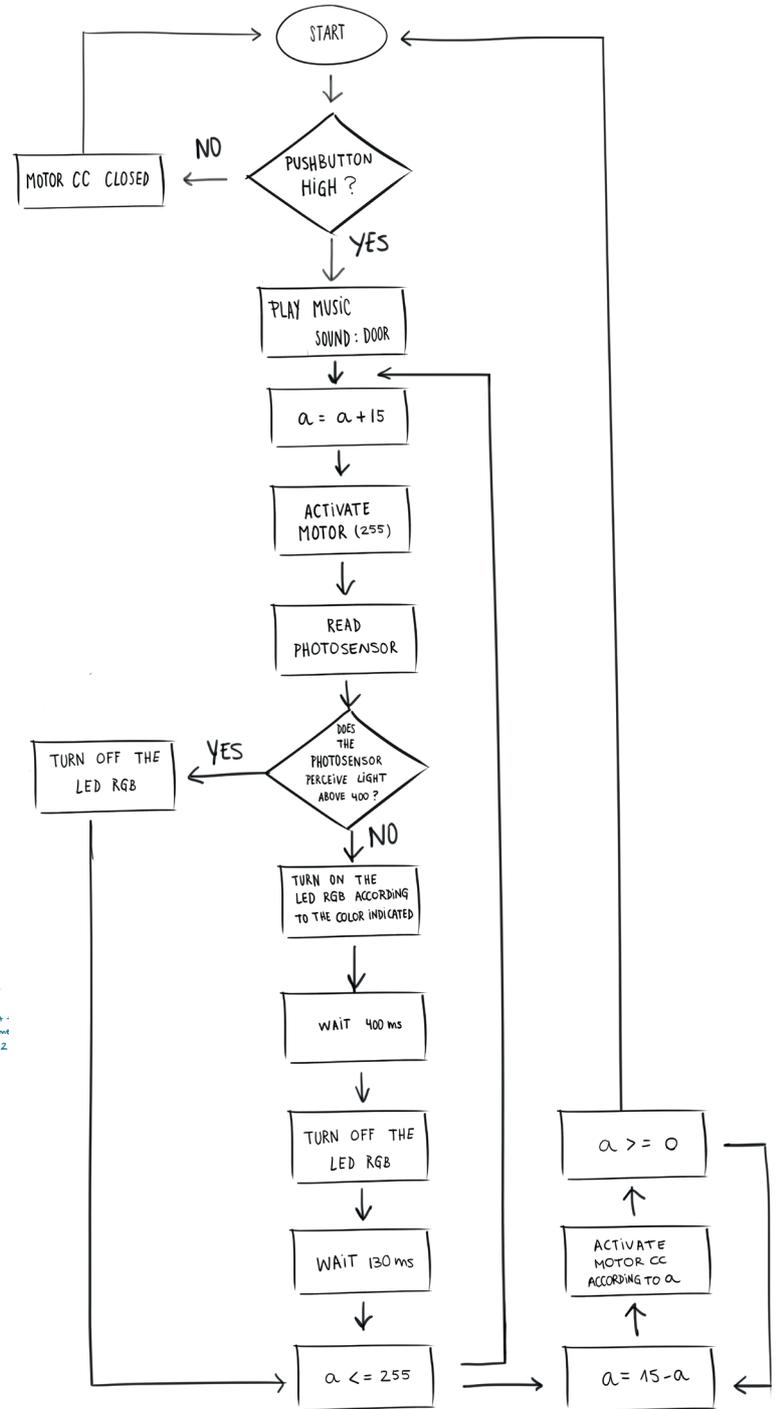
ACTION 1



ACTION 2



ACTION 3



GUIDE: HOW TO BUILD THIS PROJECT?

Before starting the project it is very important to prepare the essential materials (look headline 2) and to be clear with what you want the house to do. All this is basic to assemble the protoboard and write the code in a more effective way.

In this case we already know the following actions:

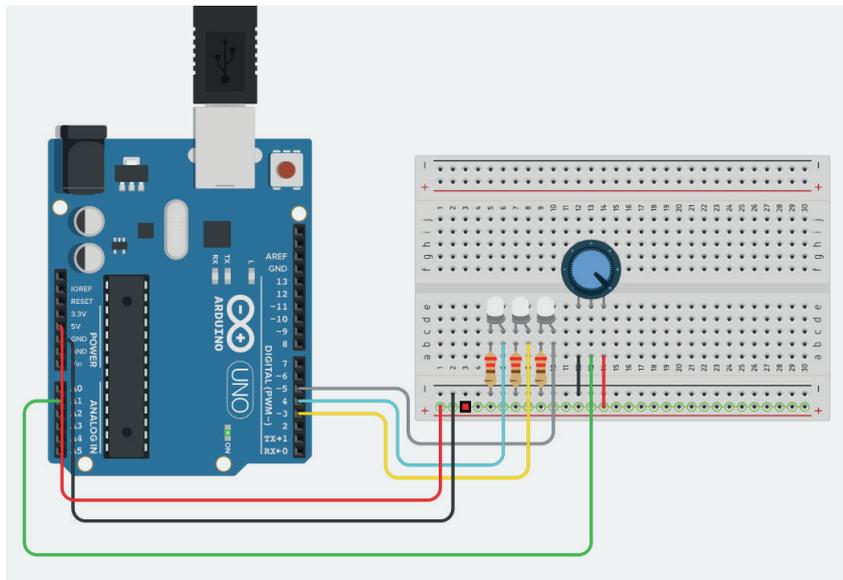
1. Turn on three white LEDs with a potentiometer in a gradual way.
2. Pressing a Push Button to activate a Servo Motor, to make a sound when moving the door and to open another white LED light.
3. The third action is to press another Push Button to make a malefic laugh, to activate a motorCC to move the blades of a fan and, in case the house is located in a dark space, to make a LED RGB blink with an orange hue with a photoresistor.

Arduino, connections and code

To be able to do the code and build the Arduino we can do the assembly of each action separately, and when they work as individuals, assemble all the parts in the same code and Arduino Board. This is due to the independence of each action because they don't depend on each other.

ACTION 1

Assembly



1. Place the three white LEDs to the protoboard.
2. Connect them to the pins 3, 4, 5. (Remember that the LED lights have a positive and negative leg. To make sure they work connect the positive and longest legs to the pins)
3. Connect the negative and short legs of the LEDs to the ground.
4. Remember to put resistances of 220Ω so the LEDs do not burn.

Code

```
//LIGHTS OF THE HOUSE
//LED DIGITAL:
int lecturaPotenciometer;
int ledPin3 = 3;
int ledPin4 = 4;
int ledPin5 = 5;

void setup(){

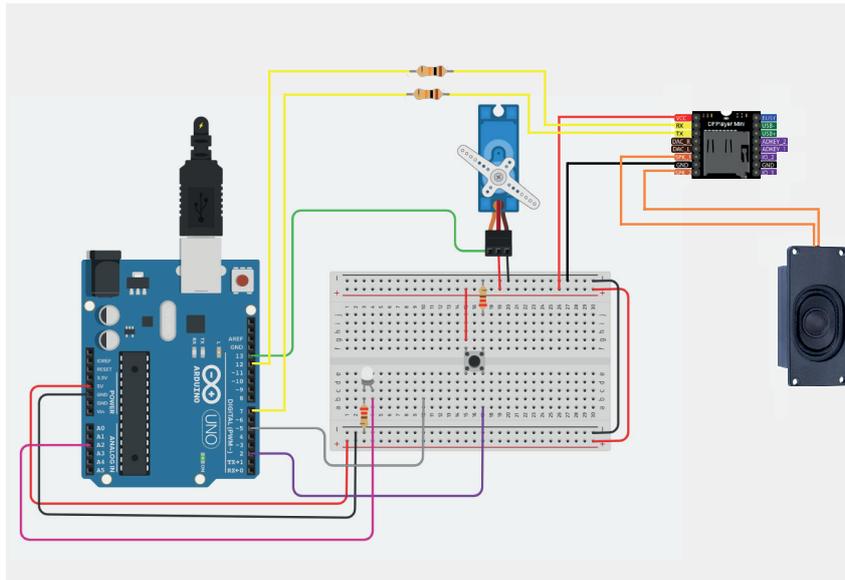
  //LIGHTS OF THE HOUSE
  //LED DIGITAL:
  pinMode(ledPin3, OUTPUT);
  pinMode(ledPin4, OUTPUT);
  pinMode(ledPin5, OUTPUT);
}

void loop(){
  //LIGHTS OF THE HOUSE
  //LED DIGITAL:
  lecturaPotenciometer = analogRead(A1); // lecture of the potentiometer
  int brillo = map(lecturaPotenciometer, 0, 1023, 0, 255); // value of the brightness of the
  leds
  analogWrite(ledPin3, brillo);
  analogWrite(ledPin4, brillo);
  analogWrite(ledPin5, brillo);
}
```

1. Declare the necessary variables with the int function, to name the pins that will be used in the code.
2. In the setup() we indicate the pins chosen to send information or outputs.
3. In the loop() we declare a variable (“lecturaPotenciometer”) to read the potentiometer. Afterwards we declare a new variable (“brillo”) to transform all the collected data of the potentiometer to values so the LED lights can turn on. Finally with this same variable, (“brillo”), we can open the LEDs depending on the potentiometer.

ACTION 2

Assembly



1. We place the ServoMotor and connect it to pin 13, 5V and the ground.
2. On the other hand place one PushButton in pull-down connected to pin 2.
3. Also place one white LED connected to the ground and pin A2.
4. Remember to put one resistance of 220Ω connected to the ground and the LED.

Code

```
//BELL, DOOR, LED
//LED vestibol
int ledPinA2 = A2;
//SERVOMOTOR
#include <Servo.h> //servo motor library
//for the servo motor
int degree = 0;
Servo servo_13;
//PUSHBUTTON
int buttonPin = 2;
int buttonStateDoor = 0;
int maxTimes=1;
//FOR THE TWO SOUNDS
//dfplayer libraries:
#include "SoftwareSerial.h"
#include "DFRobotDFPlayerMini.h"
//Use pins 7 and 12 to communicate with DFPlayer Mini
static const uint8_t PIN_MP3_TX = 7; // Connects to module's RX
```

```

static const uint8_t PIN_MP3_RX = 12; // Connects to module's TX
SoftwareSerial softwareSerial(PIN_MP3_RX, PIN_MP3_TX);
  // Create the Player object
DFRobotDFPlayerMini player;

void setup(){
  //BELL, DOOR, LED
  //SERVOMOTOR
  servo_13.attach(13);
  pinMode(buttonPin, INPUT); //Input because the information enters to the arduino
  pinMode(ledPinA2, OUTPUT);
  //FOR THE TWO SOUNDS / DFPLAYER
  // Init USB serial port for debugging
  Serial.begin(9600);
  // Init serial port for DFPlayer Mini
  softwareSerial.begin(9600);
  player.begin(softwareSerial);
  player.volume(30);
}

void loop(){
  //BELL, DOOR, LED
  buttonStateDoor=digitalRead(buttonPin);
  if(buttonStateDoor == HIGH){
    //open sound
    player.playMp3Folder(1);
    delay(800);
    //open LED of the looby
    digitalWrite(ledPinA2, HIGH);
    //open the door
    for(int times=1; times<=maxTimes; times++){
      // sweep the servo from 0 to "100" degrees in steps
      // of 1 degrees
      for (degree = 0; degree <= 100; degree += 1) {
        // tell servo to go to position in variable 'degree'
        servo_13.write(degree);
        // wait 50 ms for servo to reach the position
        delay(50); // Wait for 15 millisecond(s)
      }
    }
  }
}

```

```

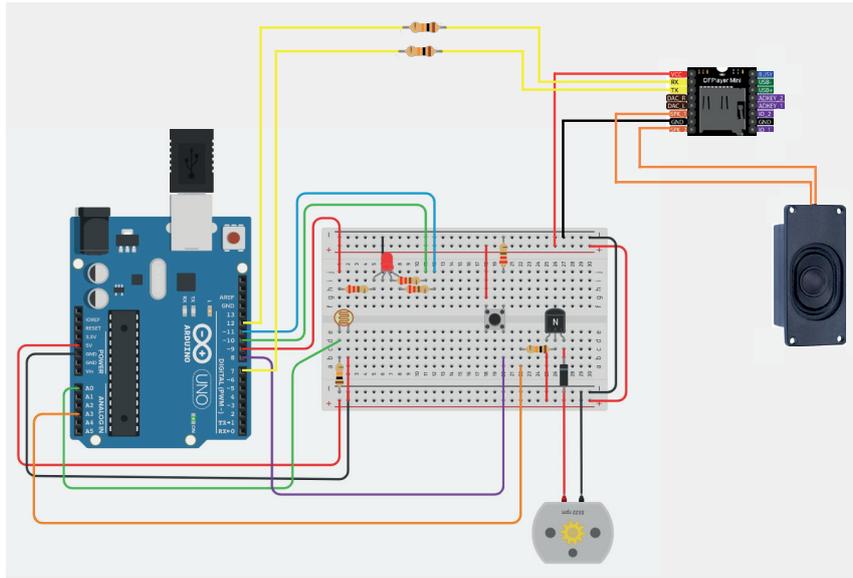
delay(200);
  for (degree = 100; degree >= 0; degree -= 1) {
    // tell servo to go to position in variable 'degree'
    servo_13.write(degree);
    // wait 50 ms for servo to reach the position
    delay(5); // Wait for 50 millisecond(s)
  }
}
}
else{
  servo_13.write(0);
  digitalWrite(ledPinA2,LOW);
}
delay(500);
player.stop();
}

```

1. Declare the variables needed for the LED, the PushButton and for the ServoMotor.
2. It is very important to remember that for use of the ServoMotor we have to add the library (`#include <Servo.h>`).
3. In the `setup()` we connect the ServoMotor to pin 13, indicating that the pin used for the LED sends information (OUTPUT) while the pin connected with the Push Button receives information (INPUT).
4. In the `loop()` we read the state of the PushButton. If this one has a HIGH state, when is pressed, the board will read and execute the code inside the `if`. So, because we want to turn on the LED and the motor to work, we write inside the key that the LED has a HIGH state and the motor turns 180° three times with three `for`. If the Push Button is not pressed, else, the ServoMotor and the LED will not be on (`servo_13.write(0)` and `digitalWrite(ledPinA2,LOW)`).
5. In the case of the DFPlayer first we have to include two libraries; “SoftwareSerial.h” and “DFRobotDFPlayerMini.h”. Then, we use pins 7 and 12 (TX and RX) of the arduino to communicate with DFPlayer. Inside the `setup()`, we have to initialize the serial ports for debugging and DFPlayer, and specify the amount of volume we want (0-30). Finally, in the `loop()` with the function `player.platMp3Folder(1)` we put the MP3 sound that we want, in this case the door sound that it is in the first place. It is important to mention that inside the SDcard it is created a folder named Mp3 and inside it we rename the files such as 0001.mp3, 0002.mp3. Order is important!

ACTION 3

Assembly



1. We connect the LED RGB to the pins 9, 10, 11 and to the ground. We recommend that the color of the wires used correspond to the colors of every leg of the RGB LED to avoid confusions.
2. Afterwards we connect the photoresistor to the A0 pin and to the ground.
3. Connect the motorCC to the 6 pin using one diode and one transistor NPN.
4. Finally we put the Push Button pull-down connected to the pin 8.
5. As mentioned before, it is important to remember to put resistances. In that case we use 220Ω and $10k\Omega$.

Code

```
//TOMB
//LED ANALOG:
int ledPin9 = 9;
int ledPin10 = 10;
int ledPin11 = 11;
int lecturaFotosensor = 0;
int analogPin0 = A0;
//MOTOR CC
int pinmotor = A3;
int buttonMotorClosed=0;
int buttonpinTomb = 8;
int buttonStateTomb = 0;
//FOR THE TWO SOUNDS
//dfplayer libraries:
#include "SoftwareSerial.h"
#include "DFRobotDFPlayerMini.h"
//Use pins 7 and 12 to communicate with DFPlayer Mini
```

```

static const uint8_t PIN_MP3_TX = 7; // Connects to module's RX
static const uint8_t PIN_MP3_RX = 12; // Connects to module's TX
SoftwareSerial softwareSerial(PIN_MP3_RX, PIN_MP3_TX);
  // Create the Player object
DFRobotDFPlayerMini player;

void setup(){
  //TOMB
  //LED ANALOG:
  pinMode(ledPin9, OUTPUT);
  pinMode(ledPin10, OUTPUT);
  pinMode(ledPin11, OUTPUT);
  Serial.begin (9600);
  //MOTORCC
  pinMode(pinmotor, OUTPUT);
  pinMode(buttonpinTomb, INPUT);
  //FOR THE TWO SOUNDS / DFPLAYER
  // Init USB serial port for debugging
  Serial.begin(9600);
  // Init serial port for DFPlayer Mini
  softwareSerial.begin(9600);
  player.begin(softwareSerial);
  player.volume(30);
}

void loop(){
  //TOMB
  //LED ANALOG:
  buttonStateTomb=digitalRead(buttonpinTomb);
  if(buttonStateTomb == HIGH){
    //open sound
    player.playMp3Folder(2);
    //MOTOR CC
    //incrementamos lentamente la señal PWM en el transistor
    for (int a = 0; a <= 255; a=15+a){
      analogWrite(pinmotor, 255);
      delay(10);
      lecturaFotosensor = analogRead(analogPin0);
      Serial.println(lecturaFotosensor);
    }
    if (lecturaFotosensor >= 400){
      analogWrite(ledPin9, 0);
      analogWrite(ledPin10, 0);
      analogWrite(ledPin11, 0); //Turn led off
    }
  }
}

```

```

else{ //led pampalluga
  analogWrite(ledPin9, 255);
  analogWrite(ledPin10, 30);
  analogWrite(ledPin11, 0);
  delay(400);
  analogWrite(ledPin9, 0);
  analogWrite(ledPin10, 0);
  analogWrite(ledPin11, 0);
  delay(130);
}
}
//disminuyendo lentamente la señal PWM en el transistor
for(int a = 255; a>= 0; a=15-a){
  analogWrite(pinmotor, a);
  delay(10);
}
}
else{
  analogWrite(pinmotor,buttonMotorClosed);
  delay(100);
}
}

```

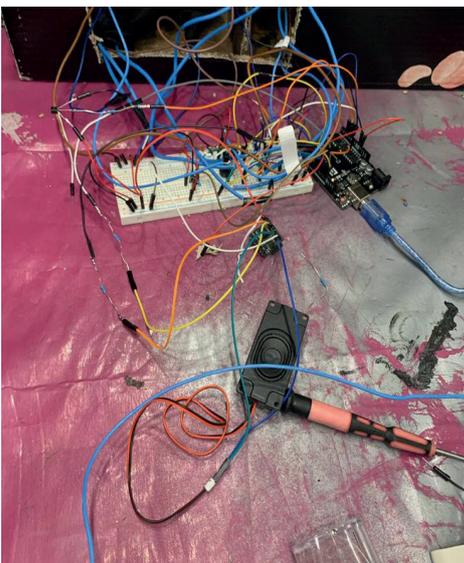
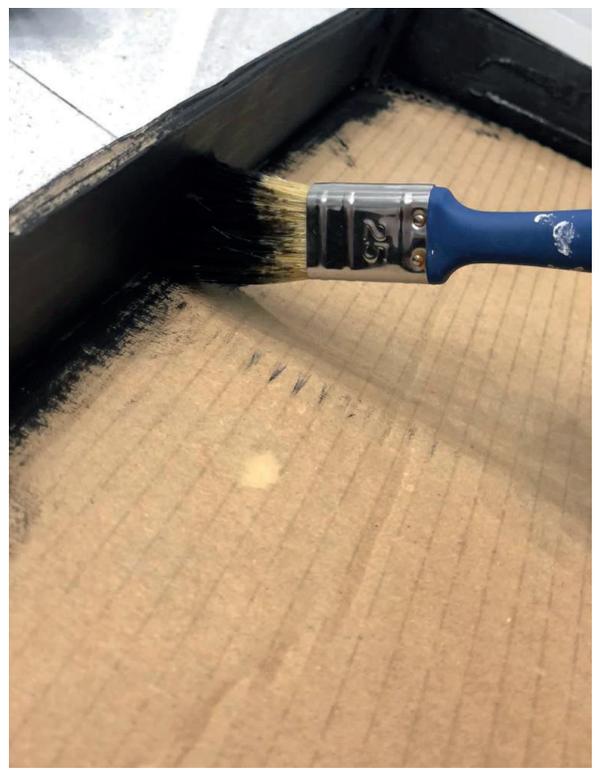
1. Declare the variables needed for the RGB LED, for the photosensor, for the Push Button and for the MotorCC.
2. In the setup() we indicate that the pins used for the LED and the motor send information (OUTPUT) while the pin connected to the Push Button receives information (INPUT).
3. In the loop() we read the state of the Push Button. If it has a HIGH state when pressed, the board will read and execute the code inside of the if. So if we want the motor to turn when pressing the button, we have to write with two fors to achieve the movement that we want. Inside the first for we add to the Arduino to read the fotosensor, so incase of a lot of light, the LED RGB will not open (if) and if not it will open with the colors declared with the three analogWrite() (else).
4. To include the scary laugh sound first we have to include two libraries; "SoftwareSerial.h" and "DFRobotDFPlayerMini.h". Then, we use pins 7 and 12 (TX and RX) of the arduino to communicate with DFPlayer. Inside the setup(), we have to initialize the serial ports for debugging and DFPlayer, and specify the amount of volume we want (0-30).Finally, in the loop() with the function player.platMp3Folder(2) we put the MP3 sound that we want, in this case the is the one that is in the order 2. It is important to mention that inside the SDcard it is created a folder named Mp3 and inside it we rename the files such as 0001.mp3, 0002.mp3. Order is important!

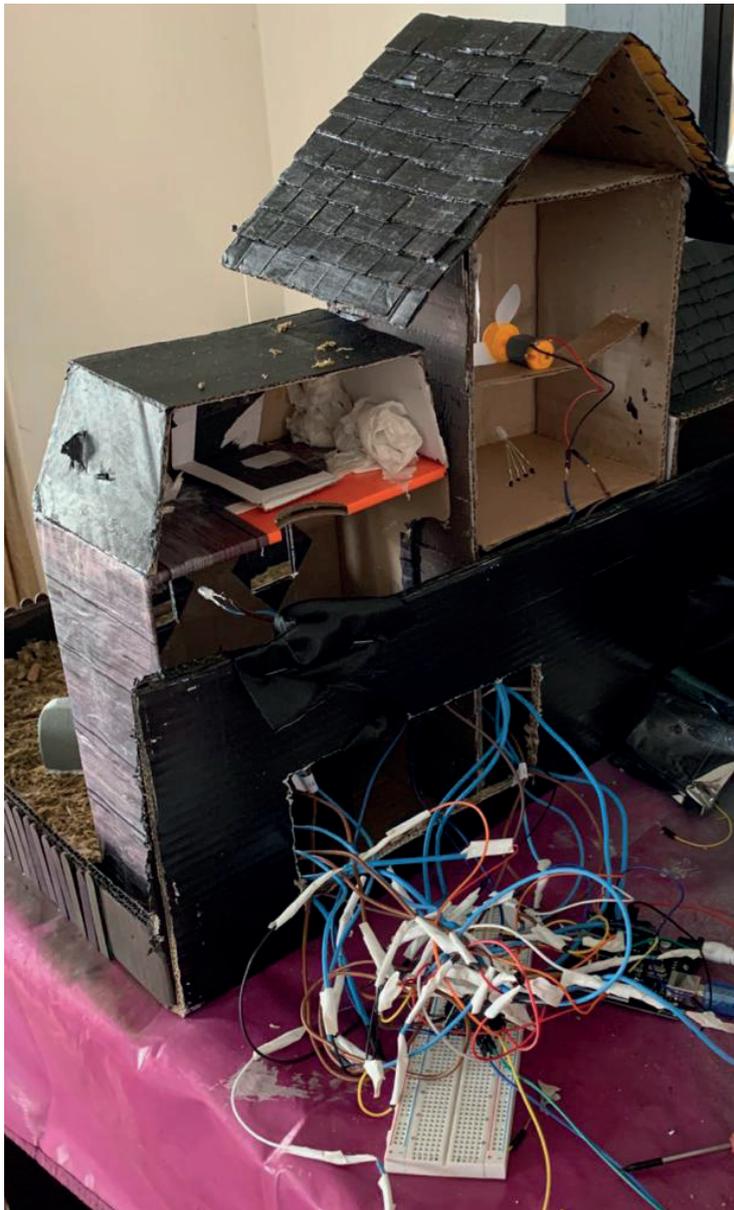
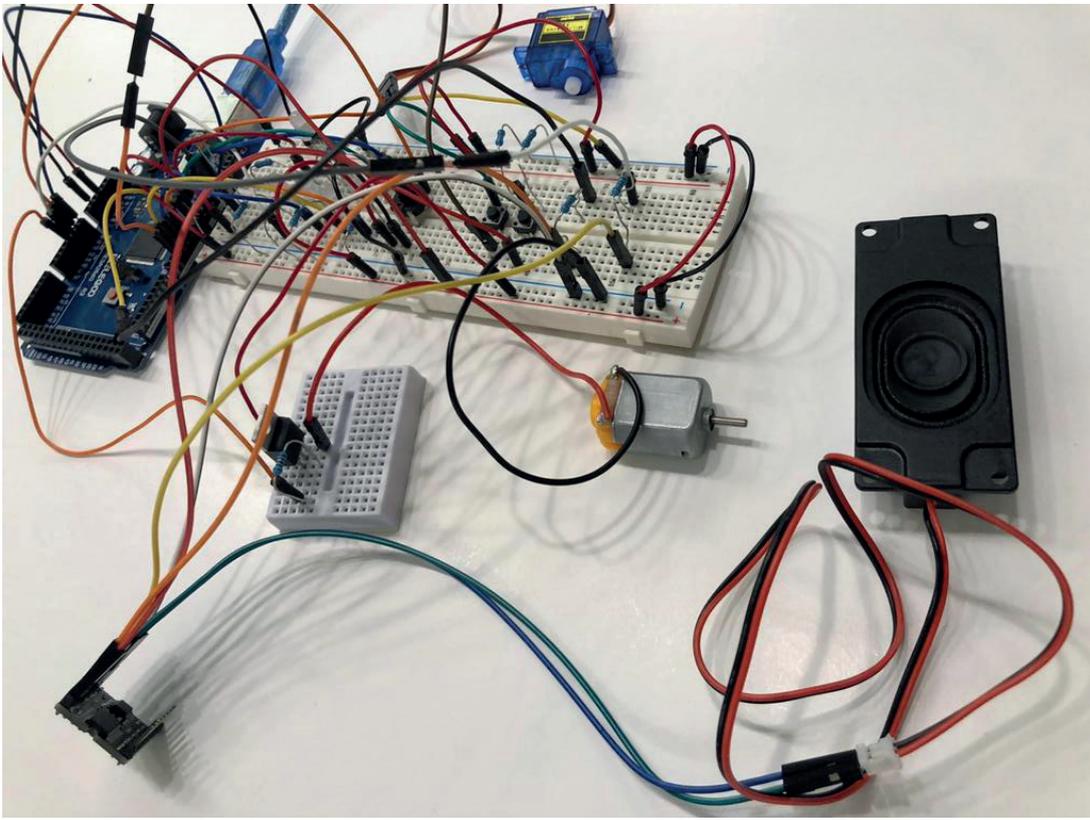
FINAL

When we have all the assemblies and codes ready it is time to join them in the same Arduino board and code. Make sure you have not repeated any pin.

Construction of the house

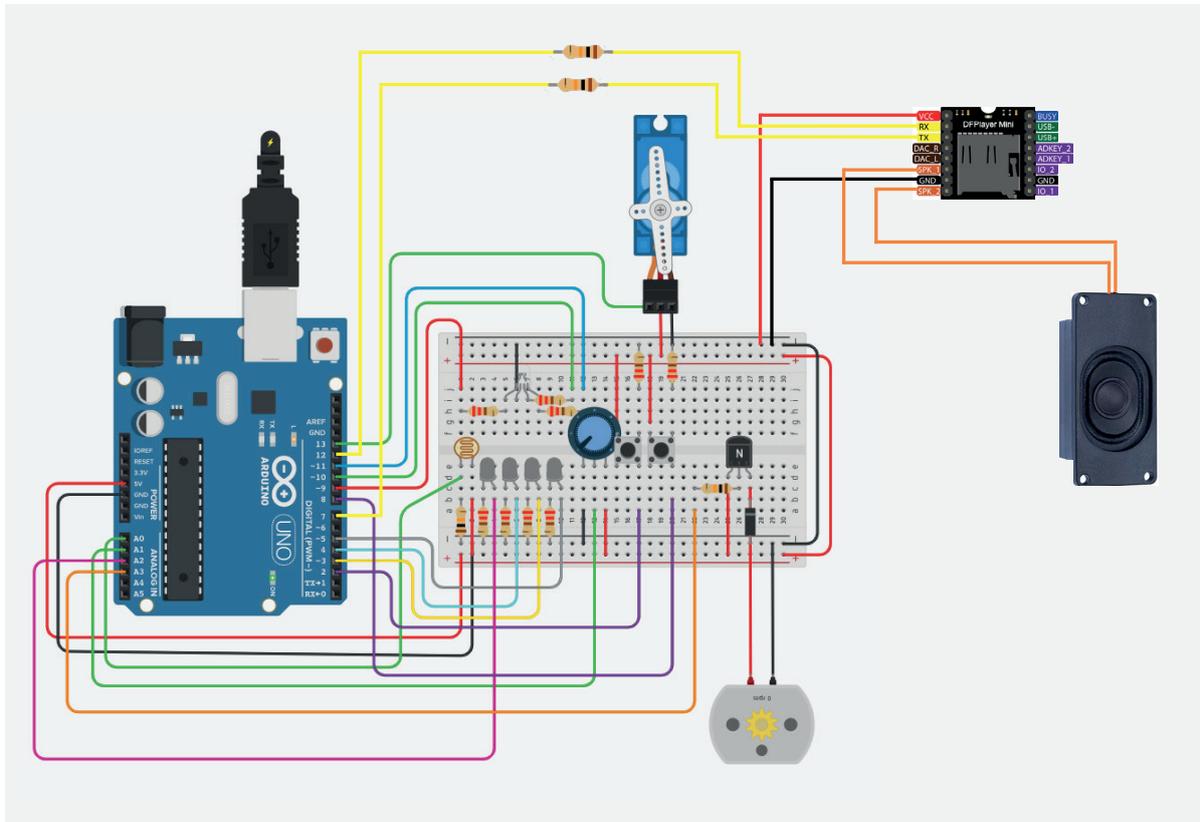
1. Cut all the cardboard with the needed measurements to create the main structure of the three houses. We recommend cutting out some windows, so that the light of the LEDs can be seen.
2. Stick all parts with glue and wait until everything is dry to continue. Make sure the unions are resistant enough to be able to work with the structure.
3. With paint, scrapbook decorative papers and cardboards decorate the houses. In our case, we used different methods for every house.
4. Afterwards add all the details. For instance, the door, the ghosts, some spiders, the grass or the curtains.
5. When the main structure is finished we can start adding the electronic part.
 - a. We install all the LED lighting and the MotorCC with the fan behind the respective windows.
 - b. Find the best position for the photosensor to receive the ambient light and add it there.
 - c. The PushButtons, the DFPlayer and the potentiometer are hidden with the decorations of the house, to make sure the mock-up looks more realistic.
 - d. Finally weld all the connections to avoid them from separating.





FINAL ASSEMBLY AND CODE

Assembly



Code

```
//Muna Abul, Maria Mei Bellsolà, Cristina Molas
//GEDI 101
//HAUNTED HOUSE CODE:

//BELL, DOOR, LED
//LED vestibol
int ledPinA2 = A2;
//SERVOMOTOR
#include <Servo.h> //servo motor library
//for the servo motor
int degree = 0;
Servo servo_13;
//DFPLAYER
//PUSHBUTTON
int buttonPin = 2;
int buttonStateDoor = 0;
int maxTimes=1;
```

```

//TOMB
//LED ANALOG:
int ledPin9 = 9;
int ledPin10 = 10;
int ledPin11 = 11;
int lecturaFotosensor = 0;
int analogPin0 = A0;
//MOTOR CC
int pinmotor = A3;
int buttonMotorClosed=0;
int buttonpinTomb = 8;
int buttonStateTomb = 0;

//LIGHTS OF THE HOUSE
//LED DIGITAL:
int lecturaPotenciometer;
int ledPin3 = 3;
int ledPin4 = 4;
int ledPin5 = 5;

//FOR THE TWO SOUNDS
//dfplayer libraries:
#include "SoftwareSerial.h"
#include "DFRobotDFPlayerMini.h"

//Use pins 7 and 12 to communicate with DFPlayer Mini
static const uint8_t PIN_MP3_TX = 7; // Connects to module's RX
static const uint8_t PIN_MP3_RX = 12; // Connects to module's TX
SoftwareSerial softwareSerial(PIN_MP3_RX, PIN_MP3_TX);
// Create the Player object
DFRobotDFPlayerMini player;

void setup(){

//LIGHTS OF THE HOUSE
//LED DIGITAL:
pinMode(ledPin3, OUTPUT);
pinMode(ledPin4, OUTPUT);
pinMode(ledPin5, OUTPUT);

```

```

//BELL, DOOR, LED
//SERVOMOTOR
servo_13.attach(13);
pinMode(buttonPin, INPUT); //Input because the information enters to the arduino
pinMode(ledPinA2, OUTPUT);

//TOMB
//LED ANALOG:
pinMode(ledPin9, OUTPUT);
pinMode(ledPin10, OUTPUT);
pinMode(ledPin11, OUTPUT);
Serial.begin(9600);
//MOTORCC
pinMode(pinmotor, OUTPUT);
pinMode(buttonpinTomb, INPUT);

//FOR THE TWO SOUNDS / DFPLAYER
// Init USB serial port for debugging
Serial.begin(9600);
// Init serial port for DFPlayer Mini
softwareSerial.begin(9600);
player.begin(softwareSerial);
player.volume(30);
}

void loop(){

//LIGHTS OF THE HOUSE
//LED DIGITAL:
lecturaPotenciometer = analogRead(A1); // lecture of the potentiometer
int brillo = map(lecturaPotenciometer, 0, 1023, 0, 255); // value of the brightness of the
leds
analogWrite(ledPin3, brillo);
analogWrite(ledPin4, brillo);
analogWrite(ledPin5, brillo);

//BELL, DOOR, LED
buttonStateDoor=digitalRead(buttonPin);
if(buttonStateDoor == HIGH){
//open sound
player.playMp3Folder(1);
}
}

```



```
lecturaFotosensor = analogRead(analogPin0);
  Serial.println(lecturaFotosensor);
  if (lecturaFotosensor >= 800){
    analogWrite(ledPin9, 0);
    analogWrite(ledPin10, 0);
    analogWrite(ledPin11, 0); //Turn led off
  }
  else{ //led pampalluga
    analogWrite(ledPin9, 255);
    analogWrite(ledPin10, 30);
```







CONCLUSION

As it is shown in the project we have been able to apply all the knowledge learned at class. All this theoretical knowledge has become a physical mock-up, we have been able to touch and experiment with electrical components such as LEDs, resistances, photoresistors, potentiometers or motors.

This fact makes us realize that only working with simulating Arduino websites, like Tinkercad, is not enough, because what you can create with a computer can be easier than reality. For instance, when we create an assembly with Arduino and physical components, there are a lot of external factors that can make the code or the assembly done before with Tinkercad not work. So it is important to be aware of this, and make up your mind to change these errors.

From this experience we conclude that even if you do a lot of prototypes with tools like Tinkercad, or 3D modelling programs, at the end what will tell us what works or if the system is functional will be the physical mock-up. This one helps us to understand shapes, structures, assemblies, etc, and lets you interact with the project.

Finally we would like to add that programming is one of the most useful tools nowadays and it is present in our lives everyday. So at the engineering and design level it is fundamental to have basic knowledge to adapt to new situations, changes, problems or necessities.

