Emily Harmon
MAE 3780 Final Report, Fall 2021

**Section I: Overview**

The AutoUnlocker is an automated door unlocking system. It is designed to add convenience to the everyday task of unlocking a door that is designed to only be unlocked with a key. It saves time and effort by using a continuous servo motor to pull the door lock open from the inside when the user enters a passcode. A 3D printed mechanism made out of PLA easily slides over the door lock. The mechanism features a hole near the top to allow a string to be tied from it to the micro servo. When someone wants to unlock the door all they need to do is press the button and then enter the passcode. A green LED or red LED will blink to let the user know whether they were right or wrong respectively. When the passcode entered is incorrect, the door won't be unlocked and the user is free to try again. If the code is correct, the micro servo will pull the door lock to its unlocked position and then rotate the opposite direction to release tension in the string so that the door can be locked again. The code can be easily edited and re uploaded to change the passcode for added security and customization.

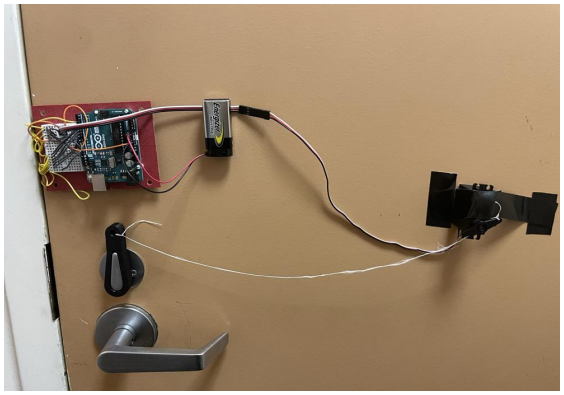

Figure 1: Completed Design Inside of Door

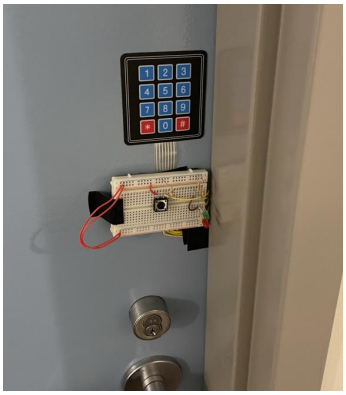

Figure 2: Completed Design Outside of Door

**Section II: Design Considerations**

If I was able to redo the project, I would do more precise hand calculations. I believe that the servo motor I used has a higher performance than it needs to. I could have reduced the length of my door lock fitting. I had made it longer to reduce the amount of torque needed to open the door. However if I had calculated the required length of the fitting using the motor specifications and an appropriate safety factor, I likely would have reduced my 3D printing expenses, which accounted for a major portion of my budget.

With added time, I would have liked to incorporate additional functionality into my code. For example, limiting the number of incorrect password attempts and locking the user out after that many tries for a set amount of time. Another useful thing would be creating a way to change the password using the keypad instead of having to edit the code to change it. Or adding a way to cancel the attempt if the user realized they accidentally hit the wrong key. Finally, it would be useful to designate a "backspace" key that would delete keypad entries.

With an increased budget I would purchase an LCD display. Currently, my project has user interactions with just the LEDs and the serial monitor. I included a 9V battery so that the system wouldn't have to solely rely on the computer to power the circuit. However, without the serial monitor, it's more difficult to know if the keys you pressed were correct. It would be better for the user to always be able to see outputs from an LCD display. I could also build an enclosure to mount to the door which would conceal the circuit and decrease the likelihood of it falling or a wire getting jostled and coming loose. It would also improve the overall aesthetics of the project.

**Section III: Assembly Instructions**

1. Attach the keypad to the outside of the door 6 inches above the lock.
    a. Peel back the paper covering the adhesive and press it on the door.
    b. Maintain light pressure for 30 seconds to ensure it remains attached
2. Use the circuit diagram (Figure 3 in Appendix B) to build the circuit
    a. See Figure 9 in Appendix C and Figures 1 and 2 in the Overview Section to understand the layout of the circuit with respect to inside and outside of the door.
    b. Use wires that are at least 6 inches long to connect the outside breadboard to the Arduino so that they can wrap around the door.
    c. I found that it was easier to wire the circuit using the mini breadboard to connect the sensors and actuators on the outside of the door to the inside.
    d. However, that is not necessary. And the wires could simply be connected straight to the Arduino
3. Tape the wires that wrap from the outside of the door to the inside of the door to make sure that the door will close easily and prevent any wires from coming undone
    a. See Appendix C Figures 6, 7, and 8 for a visual reference of how the wires should be taped
4. Tape the breadboard to the outside of the door immediately below the keypad using gorilla tape.
    a. Make sure to leave 2 inches between the edge of the door and the breadboard to allow the door to close properly.
    b. See Figure 8 in Appendix C for how the setup should look at this point
5. Tape the Arduino to the inside of the door at the same height as the breadboard on the opposite side of the door.
6. Tape the 9V battery on the inside of the door 3 inches to the right of the Arduino
    a. See Figure 11 in Appendix C for reference of how the inside of the door should look at this point
7. Tie a 1 ft. and 3 in. of string to the 3D printed door lock fitting and slide it over the door lock
    a. See Figure 10 in Appendix C to see the proper positioning of the door lock fitting
8. Tie the other side of the string to the Servo hook.
    a. Make sure that when the door is in the locked position, there is enough string length for the servo to be about a foot to the right of the door lock

    b. Figure 12 in Appendix C demonstrates how the string should look when the door is closed
9. Tape the servo to the door using Gorilla Tape.
    a. Use the string to ensure that the servo is level with door lock fitting. This will minimize the amount of torque the servo needs to output in order for it to be able to pull the lock open.
    b. Ensure that there is enough tape to keep the servo in place as it pulls the door unlocked.
    c. Figure 1 in the Overview Section and Figure 9 in Appendix C provides a reference for the layout on the inside of the door.
10. Upload the Design Project Code to the Arduino using the USB to USB B cable and the AutoUnlocker is now ready to use.

**Section IV: Operation instructions**

1. Begin on the outside of the locked door
2. Press the button located on the breadboard once. This will let the AutoUnlocker know that you are ready to start entering the password.
3. Then you can enter the password by pressing the correct buttons on the keypad.
4. Once you have finished entering the password, press the # key.
    a. This lets the AutoUnlocker know that you are done entering the password
    b. So for example of the password is 2023, you would press the button and then enter 2023#
5. If the computer is connected to the Arduino with the USB A to B cable, the serial monitor will print whether or not your password is correct. It will also show you all of the keys that you pressed excluding the # key
    a. If the password is correct, the green LED will blink three times and the servo will rotate to unlock the door
    b. If the password that was entered is incorrect, the red LED will light up for three seconds.
6. You are free to repeat the process again after a failed attempt. Begin again by following these instructions starting at step one
7. If you would like to change the passcode, you must edit the code provided in Appendix D
    a. The only line that needs to be changed is:
    const String password = "1234"; // this is where you can change the password if you want to!
    b. You can create a new password by changing the numbers within the quotation marks
    c. For example if I wanted the password to be 579402048284901948, my new line of code would be:
    const String password = "579402048284901948";  // this is where you can change the password if you want to!
    d. Then the revised version of the code must be uploaded to the Arduino by connecting your computer to the Arduino using the USB A to B cable and pressing upload

**Appendix A: BOM**

| Item | Part Number | Purchased From | Cost/ Item | Quantity | Split With Classmate? | Total Cost |
|---|---|---|---|---|---|---|
| Kookye 2PCS Mini Servo Motor 360 Degree Continuous Rotation w/Servo Horn Set | ASIN: B01HSX1IDE | Amazon | $14.97 | 2 motors | Yes: gmp66 | $7.49 |
| Gorilla Tape | ASIN: B001E5ZWT4 | Amazon | $2.97 | 1 | Yes: sep234 | $2.97 |
| 3x4 Matrix Keypad | 1528-1136-ND | Digi-key | $3.95 | 1 | No | $3.95 |
| PLA 3D printed lock fitting | N/A | Cornell RPL | $1/job+ $.40/g | 12.42 in^3 | No | $5.97 |
| Floss | N/A | Scavanged | $0.01 | 1 | No | $0.01 |
| 22AWG solid-core hookup wire | N/A | Lab Stock | $0.10/ft | 2ft | No | $0.20 |
| Green LED | 334086 | Jameco | $0.08 | 1 | No | $0.08 |
| Red LED | 697602 | Jameco | $0.05 | 1 | No | $0.05 |
| Tactile Switch Push Button | 155380 | Jameco | $0.35 | 1 | No | $0.35 |
| Arduino Board | 1050-1024-ND | Digi-key | $20.90 | 1 | No | $20.90 |
| Wire Kit | ASIN: B07PQKNQ22 | Amazon: Austor | $2.17 | 1 | No | $2.17 |
| 220 Ω Resistor | 220QBK-ND | Digi-key | $0.01 | 2 | No | $0.02 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1K Ω Resistor | 1.0kQBK-ND | Digi-key | $0.01 | 1 | No | $0.01 |
| 3 Wire Extension | 1568-1930-ND | Digi-key | $1.35 | 2 | No | $2.70 |
| USB Cable A to B | 39918 | Monoprice | $1.09 | 1 | No | $1.09 |
| Breadboard | 79X3922 | Newark | $2.71 | 1 | No | $2.71 |
| Mini Breadboard | 98AC7296 | Newark | $1.05 | 1 | No | $1.05 |

- Budget Total: $19.11
- “From Scratch” Total: $49.95

**Appendix B: Circuit Diagram**

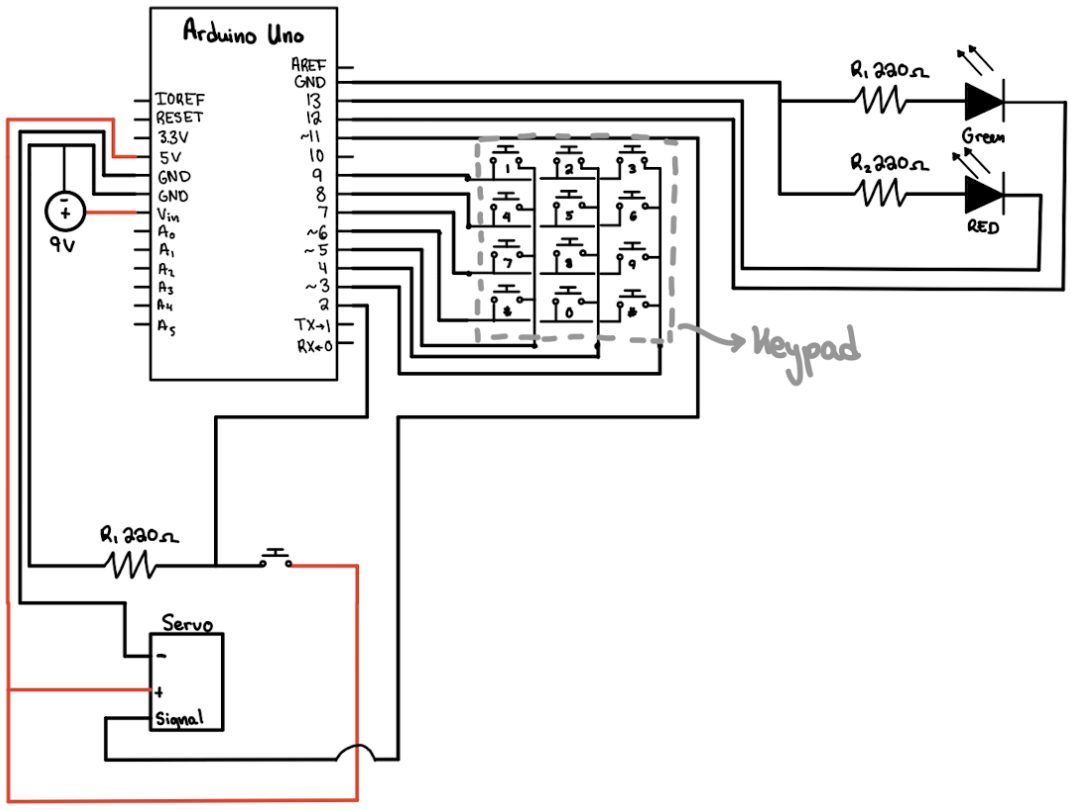


Figure 3: Circuit Diagram

## Appendix C: CAD Files and Drawings

Figure 4: Door Lock Fitting CAD

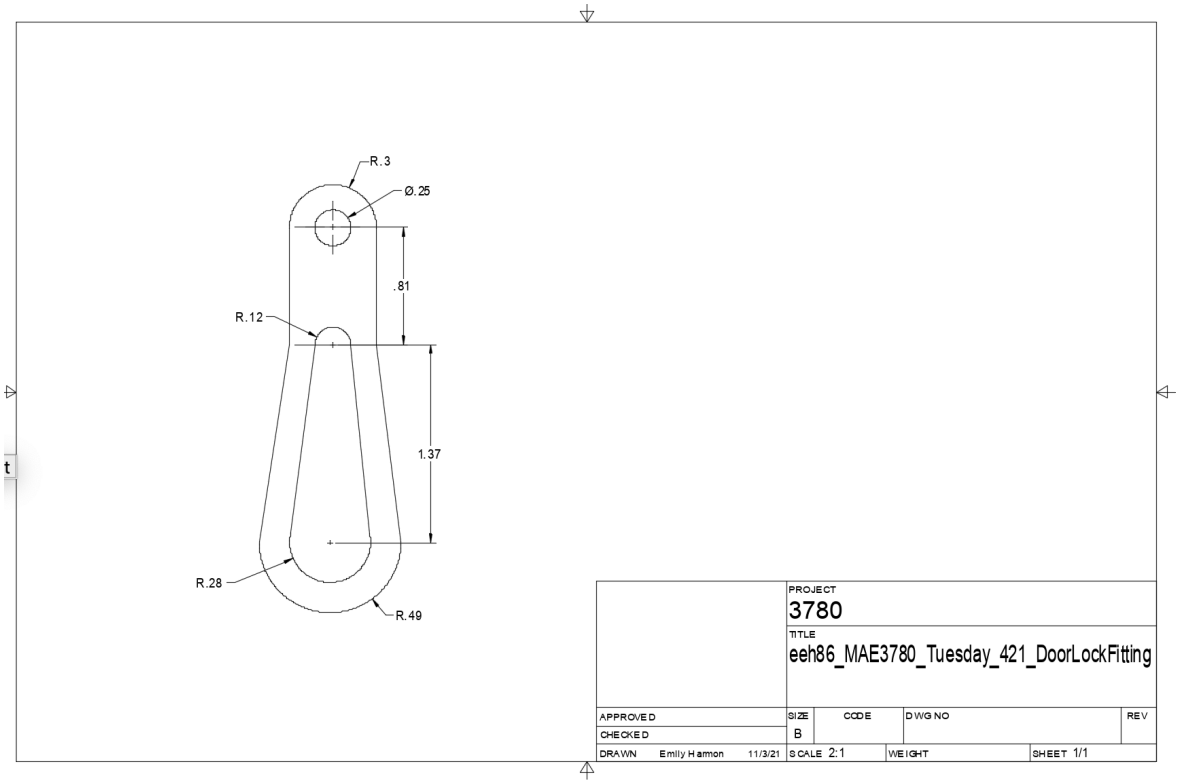


Figure 5: Dimensioned Part Drawing of Door Lock FItting


Figure 6: Taping wires down on the outside of the door

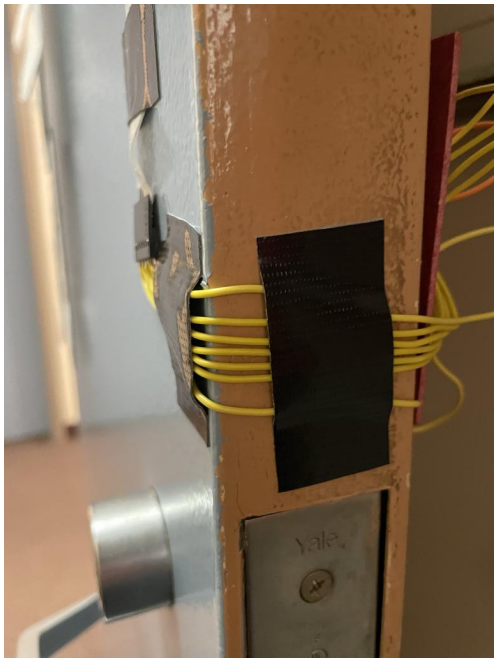


Figure 7: Taping wires on the side of the door

Figure 8: Completed outside circuit showing wires wrapped around the door

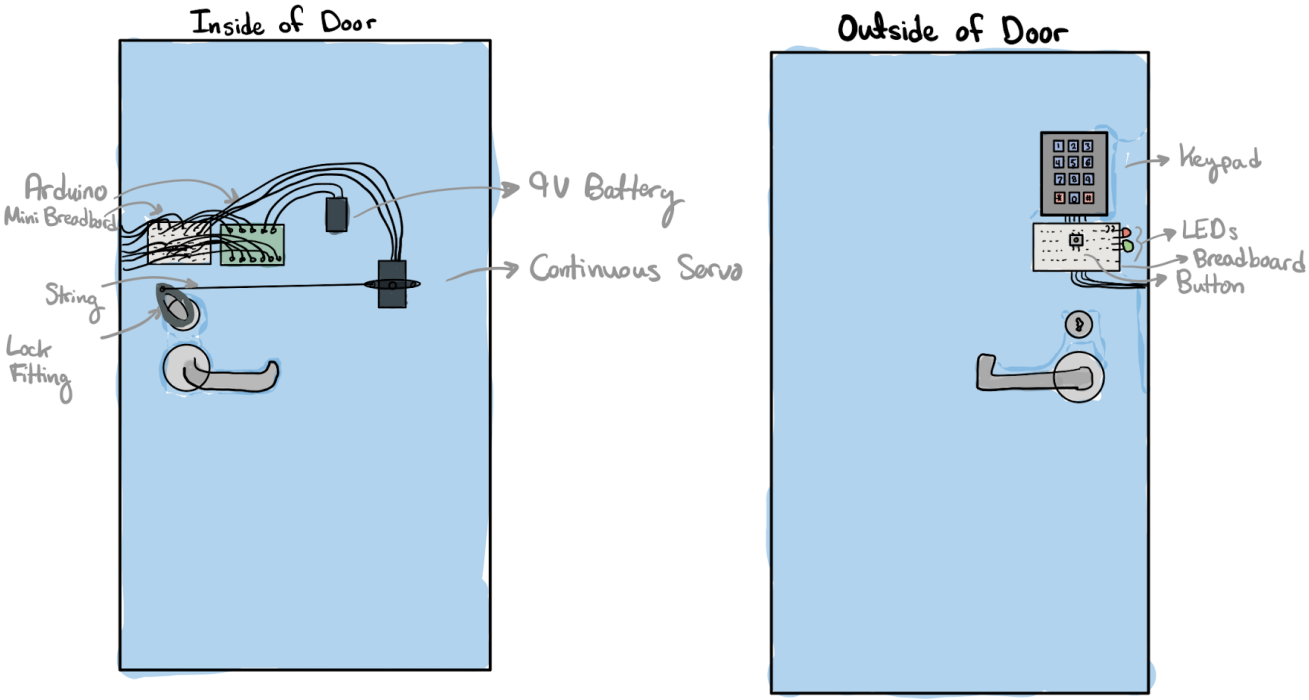


Figure 9: Assembly Sketch for Inside and Outside of Door

Figure 10: Door lock fitting placed on the door lock

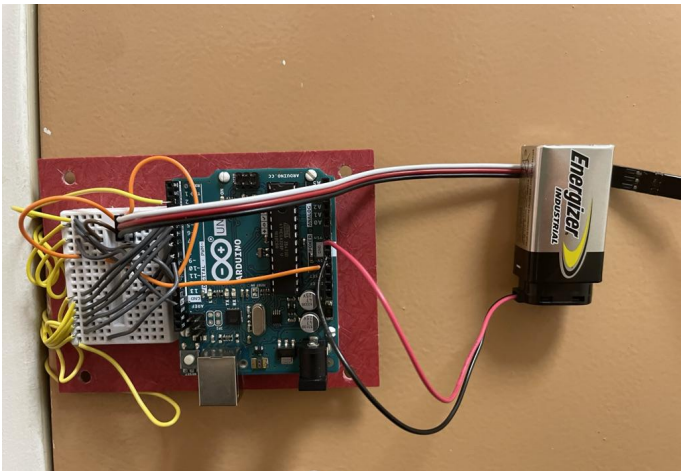


Figure 11: Arduino and 9V Battery Taped to Door

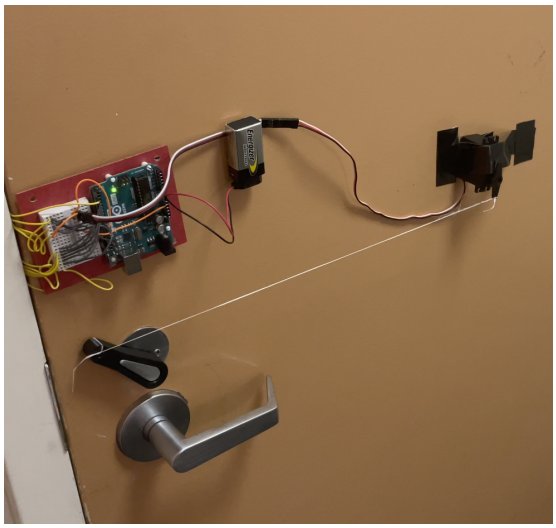

Figure 12: Door In Locked Position With Correct String Length

**Appendix D: Commented Arduino Code**

```
//This is the Arduino code for Emily Harmon's MAE 3780 Individual Project
//Referenced Code: https://arduinogetstarted.com/tutorials/arduino-keypad
        //
https://canvas.cornell.edu/courses/29449/files/4187703?module_item_id=1152867
//PROGRAM OVERVIEW
  //The program begins by setting a keyboard and a servo object, and a password
  //The loop begins by waiting for the button to be pressed
  //Then the program loops to detect the keys that are pressed
  //Once the # key has been pressed, the program compares the entered password to
the real password
  //Then the program will tell the user if they were correct and then use the appropriate
actuators

#include <Servo.h>
#include <Keypad.h>

const byte ROWS = 4; // rows
const byte COLS = 4; // columns
//define the symbols on the buttons of the keypads
char keys[ROWS][COLS] = {
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};
byte rowPins[ROWS] = {9, 8, 7, 6}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {5, 4, 3}; //connect to the column pinouts of the keypad
```

```
//initialize an instance of class NewKeypad
Keypad keypad = Keypad( makeKeymap(keys), rowPins,colPins, ROWS, COLS );

//Create Servo objects
Servo myservo;
int angle = 0;  // declare variable for servo angle
int dt = 50;    // short delay time


//Make the password to unlock the door
const String password = "1234"; // this is where you can change the password if you
want to!
String input_password;
void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT); //set pin to control green LED
  pinMode(12, OUTPUT); //set pin to control red LED
  pinMode(2, INPUT); //set pin to observe button

}
void loop() {
  //Begin evaluating sensor readings from keypad and button
  char key = keypad.getKey();  //variable that stores which key the user has pressed

  if (digitalRead(2) == HIGH) { //if the button is pressed
      input_password = ""; // clear input password
      Serial.println("Begin entering password");
       while (key != '#') {     //loop until the pound key has been pressed
           key = keypad.getKey();
           if (key != NO_KEY && key != '#'){  //Check to see if a key is pressed and make
sure that it isn't the pound key
              input_password += key; // append new character to input password string
(each time a key is pressed)
            Serial.println(key); //print the character of the keys pressed to the serial monitor
so you know which keys you pressed
          }
          }
      //the while loop ending signifies that the user is done entering the password
      //now we need to compare the entered password to the actual password
        if(password == input_password) {  //if the user is correct
          //Let the user know their password attempt was correct
          Serial.println("Password is correct, door will now unlock");
          delay(1000);
          //Blink green LED three times
          for (int i=1; i <= 3; i++) {
             digitalWrite(13, HIGH);
```

```
            delay(300);
            digitalWrite(13, LOW);
            delay(300);
          }
          delay(1000); //one more delay so that the process is easier to observe

        //Use the arduino to unlock the door
        myservo.attach(11); // tell the Arduino which pin will drive the servo- I am doing this later than setup in case the user opens the door multiple times
        // these values can be tuned
        long rotateTime = 2000;
        double rotateSpeed = 1; // from 0 to 1
        long timeStop = 200;

        // control servo
        myservo.write(90 - rotateSpeed * 90);
        delay(rotateTime);
        myservo.write(90);
        delay(timeStop);
        myservo.write(90 + rotateSpeed * 90);
        delay(rotateTime);

        myservo.detach(); //makes the servo stop by detaching servo from arduino to prevent “creeping” effect

      } else { //What happens when the entered password is incorrect
        //Let the user know their password was wrong
        Serial.println("Password is incorrect, try again");
        //Turn red LED on
        delay(300);
        digitalWrite(12, HIGH);
        delay(1000);
        digitalWrite(12, LOW);
      }
      input_password = ""; // clear input password
    }
}
```