**ABSTRACT**

The degree of security is feeble. So there is a lot of robbery, theft going on in and around the world. So, people fear to keep any of their valuables in their homes. Henceforth, many people prefer to keep it in banks. However, in this insecure world even banks are not too safe enough to satisfy people needs. A common man feels his valuables are secured if there is efficiency in security. Hence this project can give effective security in minimal cost.

# Security is a main concern in our everyday life. Each and every individual needs to feel secure. An access control for doors forms an essential part in our security pattern. Doors locked using conventional locks are not as safe as they used to be, anyone can break in by breaking these locks. We n to make a framework that will give 24/7 benefit.

Password based door lock system allows only approved persons to access restricted areas. This system is fully controlled by Arduino. The password can be entered via a keypad. If the password is matched with the stored password in Arduino the door gets open.

This programmed password based bolt framework will give client more secure and minimal effort method for locking-opening framework. The security door lock automation system promises a bold step to the future where mechanical door locks will be substituted by electronic door locks.

**INTRODUCTION**

In this project we are providing enough security to satisfy the user’s needs. The user will be prompted to enter a password to unlock the door. On successful password entry, the door unlocks for a specified amount of time enabling him/her to store or restore his/her valuables. On the other hand, if the user enters an invalid password then corresponding equivalent message will be displayed. This project “Arduino based password protected locking system” can be used to provide enough security in various places like bank lockers, security doors, BIOS locking in computer etc. This project uses an arduino kit that consists of ATMega 328 which is one of the most popular microcontrollers that consists of 14 digital pins and 6 analog general purpose pins, EEPROM of capacity 1KB and a ram of 2KB.

**ADVANTAGES & DISADVANTAGES**

Advantages:-

* Simple to use
* Simple to deploy—since the operating system provides the user accounts and password, almost no extra configuration is needed.
* Generic password use

Disadvantages:-

* Security is entirely based on confidentiality and the strength of the password.
* Does not provide strong identity check (only based on password).

Future enhancements:-

* Addition of fingerprint module can be used for increasing security

**REQUIREMENTS**

ARDUINO UNO :

This microcontroller is based on the ATmega328P. There are total of 20 pins (0-19) out of which 6 are analog inputs, 14 are digital input output pins(6 pins provide PWM voltage) which can also be used as general purpose pins, a ceramic resonator of frequency 16 MHz, an USB connection, a power jack and a reset button. It has an operating voltage of 5V. It contains everything needed to support a microcontroller.



LCD :

Liquid Crystal Display, which we are using in our project is JHD 1602A. This display consists of 16 columns and 2 rows.

PIN SUMMARY OF LCD 1602A

Pin 1: VSS.

Pin 2: To VDD 5V input.

Pin 3: VL to adjust LCD contrast with the help of 10K potentiometer. Low VL indicates light contrast and high VL indicates dark contrast.

Pin 4: RS for register select. Data registers used for high RS. Similarly, instruction register for low RS. Pin 5: R/W signal stands for read/write. When R/W bit is high, it indicates a read operation. If R/W bit is low, it indicates write operation.

Pin 6: Clock Enable- Edge triggering.

Pin 7 to 14: Represents from Bit 0 to Bit 7.

Pin 15: back light Anode.

Pin 16: back light cathode.



Membrane Keypad :

In our project we used 4X4 matrix membrane keypad. This 16 button keypad provides user interface component for Arduino project. this is programmed using the library . It has the following features: 1. Easy interface to Arduino.

2. Ultra-thin design.

3. Cheap and economical

SUMMARY about Keypad pins:

1. Maximum operation rating: 24VDC

2. Insulation Resistance : 100M ohm , 30 mA.

3. Interface: 8 pins can be accessed in the form of 4X4 matrix.



Servo Motor:

The servo used in the project is SG90 Micro Servo weighing about 9g.It has the following operating conditions:

Modulation Analog Torque 25.0 oz-in (1.8kg/cm)

Speed 0.12 s/60 deg

Weight 0.32 oz (9.8g)

Motor type 3 pole

Gear type Plastic

Rotation/Support Bushing

Pulse Width 500-2400 micro-sec



**IMPLEMENTATION- CONNECTION WIRING SPOT DIAGRAM AND CIRCUIT DIAGRAM**

The following table shows the connection wiring spot diagram of our circuit. The entire circuit diagram can be tabulated as shown below. This tabulation is called connection wiring spot diagram. The entire circuit connection is being available in this table Labeling of the pins as per the above tabulation:

D’N’: D- Digital pins in arduino. N is the pin number

A’N’: A-Analog pins in arduino. N is the pin number

LCD’N’: Liquid Crystal Display pins. N is the pin number

DB0-DB7: Data byte pins from 0-7

R/W- Read/Write

VD0 and LED+- Positive connection of the LCD

VD1 and LED- - Negative connection of the LCD.

ROW’N’- Rows of the hex keypad. N is the pin number

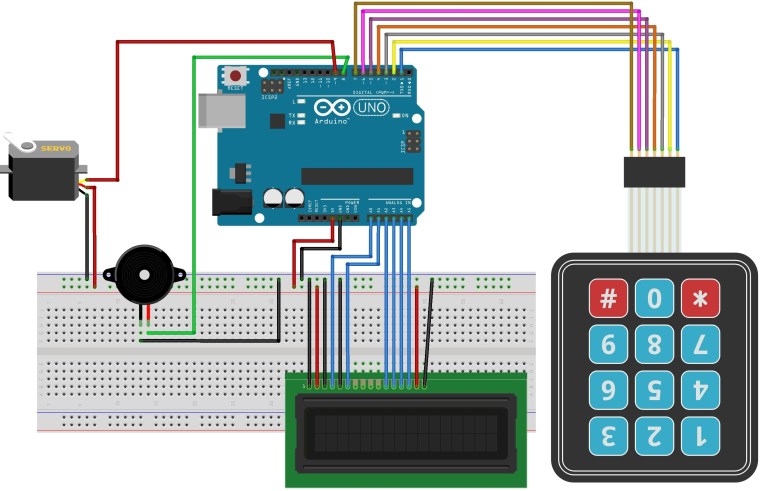
COL’N’- Columns of the hex keypad. N is the pin number

Motor- Servo motor Connection

MICRO- Arduino Connection

E- Enable

RS: Reset



**CODE :**

#include <Keypad.h>

#include <LiquidCrystal.h>

#include <Servo.h>

Servo myservo;

LiquidCrystal lcd(A0, A1, A2, A3, A4, A5);

#define Password\_Lenght 7 // Give enough room for six chars + NULL char

int pos = 0; // variable to store the servo position

char Data[Password\_Lenght]; // 6 is the number of chars it can hold + the null char = 7

char Master[Password\_Lenght] = "123456";

byte data\_count = 0, master\_count = 0;

bool Pass\_is\_good;

char customKey;

const byte ROWS = 4;

const byte COLS = 3;

char keys[ROWS][COLS] = {

{'1', '2', '3'},

{'4', '5', '6'},

{'7', '8', '9'},

{'\*', '0', '#'}

};

bool door = true;

byte rowPins[ROWS] = {1, 2, 3, 4}; //connect to the row pinouts of the keypad

byte colPins[COLS] = {5, 6, 7}; //connect to the column pinouts of the keypad

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

void setup()

{

myservo.attach(9);

ServoClose();

lcd.begin(16, 2);

lcd.print(" Arduino Door");

lcd.setCursor(0, 1);

lcd.print("--Look project--");

delay(3000);

lcd.clear();

}

void loop()

{

if (door == 0)

{

customKey = customKeypad.getKey();

if (customKey == '#')

{

lcd.clear();

ServoClose();

lcd.print(" Door is close");

delay(3000);

door = 1;

}

}

else Open();

}

void clearData()

{

while (data\_count != 0)

{ // This can be used for any array size,

Data[data\_count--] = 0; //clear array for new data

}

return;

}

void ServoOpen()

{

for (pos = 180; pos >= 0; pos -= 5) { // goes from 0 degrees to 180 degrees

// in steps of 1 degree

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

} }

void ServoClose()

{

for (pos = 0; pos <= 180; pos += 5) { // goes from 180 degrees to 0 degrees

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

}

void Open()

{

lcd.setCursor(0, 0);

lcd.print(" Enter Password")

customKey = customKeypad.getKey();

if (customKey) // makes sure a key is actually pressed, equal to (customKey != NO\_KEY)

{

Data[data\_count] = customKey; // store char into data array

lcd.setCursor(data\_count, 1); // move cursor to show each new char

lcd.print(Data[data\_count]); // print char at said cursor

data\_count++; // increment data array by 1 to store new char, also keep track of the number of chars entered

}

if (data\_count == Password\_Lenght - 1) // if the array index is equal to the number of expected chars, compare data to master

{

if (!strcmp(Data, Master)) // equal to (strcmp(Data, Master) == 0)

{

lcd.clear();

ServoOpen();

lcd.print(" Door is Open");

door = 0;

}elsE{

lcd.clear();

lcd.print(" Wrong Password");

delay(1000);

door = 1;

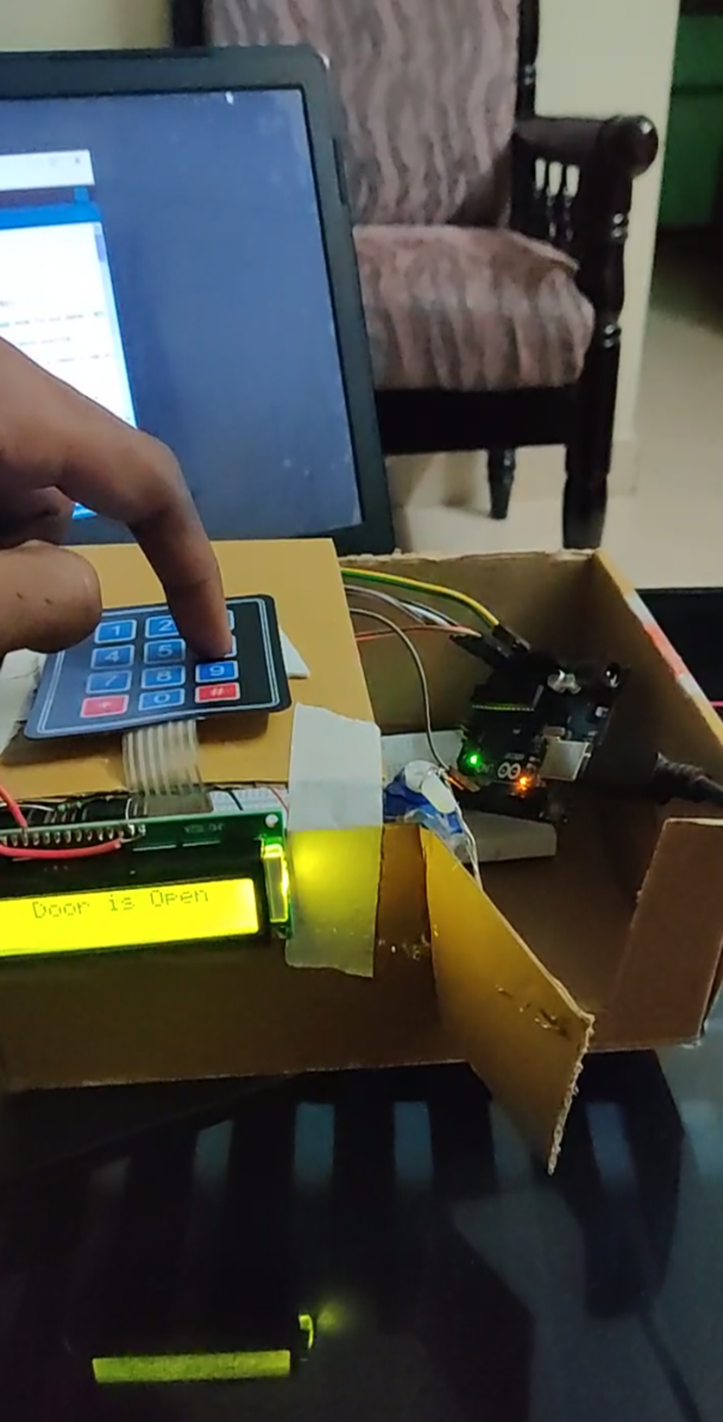
}

clearData();

}

}

**INTEGRATION AND SYSTEM TESTING**



**CONCLUSION**

This project is effective in providing enough security as long as the password is not shared. In future this “Arduino based password security locking system” can be provided maximum security by the above enhancements in order to completely satisfy user’s needs. Hence, a common man can afford to purchase such locking system in minimal cost to keep his valuables safely without any worries.