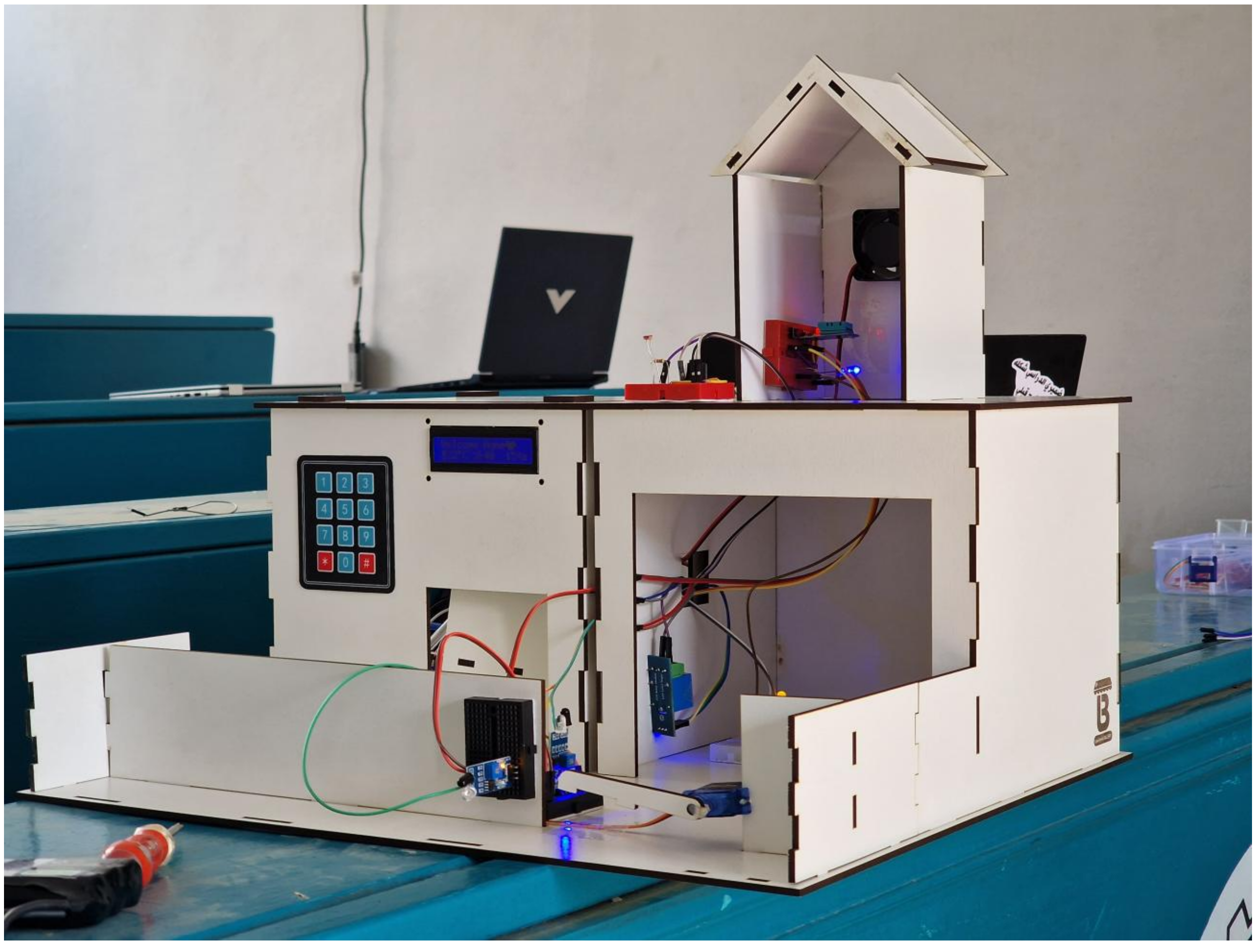
Project: Smart Home



Team Members:

- Abdalrhman Mahmoud Ibrahim Ibrahim Badawi
- Mohamed Fayez Mahmoud Elhabiby
- Khaled Ahmed Hamada Elsayed Saad
- Abdullh Osama Abdullh Zidan
- Omar Mohmed Mahros Amoud

1. Project Overview:

This project aims to create a smart home system that integrates door control, security measures, temperature management, and light adjustment. The system uses various sensors and actuators to automate these functions, making the home more secure and comfortable.

2. Big Parts:

   A. Presenting Data
      Lcd is conttected to pins from 8 to 13 to show data of various tasks
   B. Password for house door (Keypad):
      A 2x2 Keybad connected to A5,A4,A3,A2 pins
      And a named variable with the password value is used to turn on a servo for opening the house door
   C. Automatic Garden Door (servo, 2 IR)
      A second servo controlled by to IRs on both sides connected to pins 1,2,3
   D. High Temperature Alram System (DHT11)
      A fan is turned on at 25 degrees connected to pin 6 but the fan only takes vcc and ground so for it to work and take signal from the arduino a relay will be used is the middle of the connection to control the fan state
      A buzzer turns on at temperatures 35 and turned off using a push button
      The sensor connected to A1 and Used in the code using the DHT library
   E. Light Control System
      Led is connected to manage the lighting of the house
      An ldr is connected via resistor and mapped from 0 to 1024 to 0 to 255 the 255 - brightness to flip it for the led too light at dark and turn off at bright
   F. Buzzer
      It is connected to pin 0 and used to alert the user is different situations such getting the password wrong or getting a temp above 35

G. The Arduino

One Arduino is used

Analog

A0:LDR

A1:DHT11

A2,A3,A4,A5:Key Pad

Digital

0:BUZZER

1:1ST IR

2:2ND IR

3:Car Servo

4:Door Servo

5:Fan

6:Push

7:Led

8,9,10,11,12,13:LCD

Code:

```
#include <Keypad.h>
#include <Servo.h>
#include <DHT.h>
#include <LiquidCrystal.h>

//Shapes
 byte Heart[] = {
  B00000,
  B00000,
  B01010,
  B11111,
  B11111,
  B01110,
  B00100,
  B00000
 };
```

```
byte Bell[] = {
 B00100,
 B01110,
 B01110,
 B01110,
 B11111,
 B00000,
 B00100,
 B00000
};
byte Alien[] = {
 B11111,
 B10101,
 B11111,
 B11111,
 B01110,
 B01010,
 B11011,
 B00000
};
byte Check[] = {
 B00001,
 B00001,
 B00011,
 B10110,
 B11100,
 B01000,
 B00000,
 B00000
};
byte Speaker[] = {
 B00010,
 B00110,
 B11110,
 B11110,
 B11110,
 B00110,
 B00010,
 B00000
};
byte Sound[] = {
 B00001,
 B00011,
 B00101,
 B01001,
 B01001,
 B01011,
 B11011,
 B11000
};
byte Skull[] = {
 B00000,
 B01110,
 B10101,
 B11011,
 B01110,
 B01110,
 B00000,
 B00000
};
byte Lock[] = {
 B01110,
 B10001,
 B10001,
 B11111,
 B11011,
 B11011,
 B11111,
 B00000
};
```

```
byte degreeSymbol[] = {
 B00000,
 B00100,
 B00100,
 B00000,
 B00100,
 B00100,
 B00000,
 B00000
};
byte sadFace[] = {
 B11111,
 B11111,
 B10101,
 B11011,
 B11111,
 B10001,
 B01110,
 B11111
};
byte Heart_1[] = {
 B00100,
 B01110,
 B11111,
 B11111,
 B01111,
 B00111,
 B00011,
 B00001
};
byte Heart_2[] = {
 B00100,
 B01110,
 B11111,
 B11111,
 B11110,
 B11100,
 B11000,
 B10000
};
byte Thermo[] = {
 B01110,
 B01010,
 B01010,
 B01010,
 B01010,
 B10001,
 B10001,
 B01110
};
byte Sun[] = {
 B00000,
 B11100,
 B00010,
 B00001,
 B00001,
 B00010,
 B11100,
 B00000
};
byte Sunray[] = {
 B01000,
 B10000,
 B00000,
 B11100,
 B00000,
 B10000,
 B01000,
 B00100
};
```

```
// LCD and Password
 const int rs = 8, en = 9, d4 = 10, d5 = 11, d6 = 12, d7 = 13;
 LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
 String correctPassword = "121";
 String enteredPassword = "";
 const int buzzerPin = 0;
 String newPassword = "";
 String changePassword = "111";

// States for changing password
 bool isChangingPassword = false;
 bool confirmNewPassword = false;
 unsigned long passwordChangeTimer = 0;   // Timer for password
change messages
 bool isPasswordChangeTimerActive = false;
 bool enter = false;
 bool nochange = false;
 unsigned long backtimerStart = 0;
 unsigned long passbuzzertimerStart = 0;

// Retry system
 int retryCount = 0;
 const int maxRetries = 3;
 bool isLockedOut = false;
 bool lockoutActive = false;
 unsigned long lockoutTimer = 0;

// Define the pins for the keypad rows and columns
 const byte ROW_NUM = 2;     // Two rows
 const byte COLUMN_NUM = 2;  // Two columns

 char keys[ROW_NUM][COLUMN_NUM] = {
  {'1', '2'},  // Row 1
  {'4', '5'}   // Row 2
 };

 byte pin_rows[ROW_NUM] = {A5, A4};        // Row pins (4 and 5)
 byte pin_column[COLUMN_NUM] = {A3, A2};   // Column pins (6 and 7)

 Keypad keypad = Keypad(makeKeymap(keys), pin_rows, pin_column,
ROW_NUM, COLUMN_NUM);

// IR sensor and servos
 const int irSensorPin = 1;
 const int ir2SensorPin = 2;
 Servo carDoorServo;
 Servo houseDoorServo;
 int carDoorServoPin = 3;
 int houseDoorServoPin = 4;
 unsigned long carDoorTimer = 0;
 unsigned long houseDoorTimer = 0;
 bool carDoorOpen = false;
 bool houseDoorOpen = false;
 unsigned long doortimerStart = 0;      // Timer for handling delays
 unsigned long cardoortimerStart = 0;
 unsigned long lockoutStart = 0;   // Timer for lockout
 bool isTimerActive = false;       // Indicates if a timer is active
 bool isDoorOpen = false;

// Fan Control
 const int fanPin = 5;           // Pin for the DC motor (fan)
 unsigned long previousMillis = 0;

//DHT11
 #define DHTPIN A0       // Pin where the DHT11 is connected
 #define DHTTYPE DHT11    // Specify the sensor type (DHT11)
 DHT dht(DHTPIN, DHTTYPE);  // Initialize DHT sensor
 const int pushButtonPin = 6;     // Push button to turn off the alarm
 const float tempThreshold = 20.0;
 const float highTempThreshold = 35.0;
 bool alarmOn = false;
 bool buzzerOn = false;

//LDR
 const int ldrPin = A1;     // LDR connected to analog pin A0
 const int ledPin = 7;     // LED 1 connected to PWM pin 7

unsigned long UpdateTimer = 0;
const unsigned long UpdateInterval = 500;

bool inhouse = false;
unsigned long startTime;      // Variable to store the start time

void setup() {
  dht.begin();
  lcd.begin(16, 2);
  lcd.print("Enter Password:");

  lcd.createChar(0, Heart);
  lcd.createChar(1, Bell);
  lcd.createChar(2, Alien);
  lcd.createChar(3, Check);
  lcd.createChar(4, Speaker);
  lcd.createChar(5, Sound);
  lcd.createChar(6, Skull);
  lcd.createChar(7, Lock);
  lcd.createChar(8, degreeSymbol);
  lcd.createChar(9, sadFace);
  lcd.createChar(10, Heart_1);
  lcd.createChar(11, Heart_2);
  lcd.createChar(12, Thermo);
  lcd.createChar(13, Sun);
  lcd.createChar(14, Sunray);
  carDoorServo.attach(carDoorServoPin);
  houseDoorServo.attach(houseDoorServoPin);
  carDoorServo.write(180);
  houseDoorServo.write(95);

  digitalWrite(buzzerPin, LOW);
  digitalWrite(fanPin, LOW);

  pinMode(irSensorPin, INPUT);
  pinMode(ir2SensorPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(fanPin, OUTPUT);
  //pinMode(tempSensorPin, INPUT);
  pinMode(pushButtonPin, INPUT_PULLUP);
  pinMode(ledPin, OUTPUT); // Configure PWM pin as output
  pinMode(ldrPin, INPUT);     // Set LDR pin as input
  //pinMode(led1Pin, OUTPUT);   // Set LED 1 pin as output
  //pinMode(led2Pin, OUTPUT);   // Set LED 2 pin as output

  startTime = millis();       // Record the start time

}

void loop() {

  unsigned long currentMillis = millis();

    // Calculate and display elapsed time in seconds
  unsigned long elapsedTime = (millis() - startTime) / 1000; // Convert to
seconds
  lcd.setCursor(12, 1);
  lcd.print(elapsedTime);
  lcd.print("s"); // Display seconds

if (!inhouse) {
```

```
// Car detection
 if (digitalRead(irSensorPin) == LOW && !carDoorOpen) {
   lcd.clear();
   lcd.print("Car Detected!");
   carDoorServo.write(90);  // Open the car door
   carDoorOpen = true;
   carDoorTimer = currentMillis;
 }

 if (digitalRead(ir2SensorPin) == LOW && !carDoorOpen) {
   lcd.clear();
   lcd.print("Car Detected!");
   carDoorServo.write(90);  // Open the car door
   carDoorOpen = true;
   carDoorTimer = currentMillis;
 }

 if (carDoorOpen && currentMillis - carDoorTimer >= 5000) {
   carDoorServo.write(180); // Close the car door
   carDoorOpen = false;
   lcd.clear();
   lcd.print("Enter Password:");
}

  // Handle Door timer
if (isDoorOpen && (millis() - cardoortimerStart >= 5000)) {
  houseDoorServo.write(95);  // Close the house door
  isDoorOpen = false;
  isTimerActive = false;
  lcd.clear();
}

// Handle lockout timer
 if (isLockedOut && (currentMillis - lockoutStart >= 10000)) {
    isLockedOut = false;  // Unlock the system after 10 seconds
    retryCount = 0;       // Reset retry count
    lcd.clear();
    lcd.print("Enter Password:");
  }

  // Handle password change timer
    if (isPasswordChangeTimerActive && (currentMillis - 
passwordChangeTimer >= 2000)) {
     isPasswordChangeTimerActive = false;  // Stop the timer
     lcd.clear();
    if (isChangingPassword) {
     lcd.print("Enter New Pass:");
    } else {
     lcd.print("Enter Password:");
    }
  }

  // Handle Return to Default timer
   if (enter && (currentMillis - backtimerStart >= 5000)) {
    enter = false;
    noTone(buzzerPin);  // Stop the buzzer
    lcd.clear();
    lcd.print("Enter Password:");
  }

  // Handle Return to Default timer
   if (nochange && enter && (currentMillis - backtimerStart >= 5000)) {
    enter = false;
    nochange = false;
    lcd.clear();
    lcd.print("Enter Password:");
  }

  //Handling Password
```

```
    char key = keypad.getKey();
      if (key) {
       if (isChangingPassword) {
        handlePasswordChange(key);
       } else {
        handlePasswordEntry(key);
       }
      }
  }


if(inhouse) {

  static int lastTemperature = 0;
  static int lastBrightness = -1;    // Initial impossible value for brightness
  static bool screenUpdated = false;

      // Handle Door timer
  if (isDoorOpen && (millis() - cardoortimerStart >= 5000)) {
    houseDoorServo.write(90);  // Close the house door
    isDoorOpen = false;
    isTimerActive = false;
  }

  if (currentMillis - UpdateTimer >= UpdateInterval) {
   UpdateTimer = currentMillis;

   // Read the temperature
   int temperature = dht.readTemperature(); // Convert to Celsius (for
LM35)
   //lastTemperature = temperature;

   if (isnan(temperature)) {
     lcd.print("Error reading temp");
     return;
   }

  // Display the temperature on the LCD
    if (temperature != lastTemperature) {
      lastTemperature = temperature;

      lcd.setCursor(0, 1);
      lcd.write((byte)12);
      //lcd.print(" ");
      lcd.print(temperature);
      lcd.write(0b11011111); // Degree symbol
      lcd.print("C");
      lcd.print(" ");
      screenUpdated = true;

   }

  int ldrValue = analogRead(ldrPin);            // Read the LDR value (0-
1023)
  int brightness1 = map(ldrValue, 0, 1023, 0, 255); // Map LDR value to
LED brightness (0-255)
  int brightness2 = 255 - brightness1;         // Inverse brightness for LED 2
  int brightnessDisplay = map(brightness1, 0, 255, 0, 100); // Convert to
percentage

  analogWrite(ledPin, brightness2);           // Set brightness for LED 1

   if (brightness1 != lastBrightness) {
     lastBrightness = brightness1;
     lcd.setCursor(6, 1);
     lcd.write((byte)13);
     lcd.write((byte)14);
     //lcd.print(" ");
     lcd.print(brightnessDisplay);
     lcd.print(" ");     // Clear extra characters
     screenUpdated = true;
   }
  }

  //Screen 2
   if (!screenUpdated) {
   lcd.setCursor(0, 0);
   lcd.print("Welcome Home!    "); // Static message
  }

  // Fan control logic
   if (lastTemperature > tempThreshold) {
    digitalWrite(fanPin, HIGH); // Full speed (always ON)
   } else {
    digitalWrite(fanPin, LOW); // Turn fan OFF
  }

  // High-temperature alarm
   if (lastTemperature > highTempThreshold && !alarmOn && !buzzerOn)
{
    digitalWrite(buzzerPin, HIGH); // Turn on the alarm
    buzzerOn = true;
    alarmOn = true;
  }

  // Turn off the alarm manually
   if (alarmOn && digitalRead(pushButtonPin) == LOW) {
    digitalWrite(buzzerPin, LOW);
    alarmOn = false;
    buzzerOn = false;
  }
  //delay(500);
 }

}
```

```
void handlePasswordEntry(char key) {
  if (key == '4') {  // Submit the password
    if (enteredPassword == correctPassword) {
      lcd.clear();
      lcd.print("Welcome Home");
      lcd.setCursor(12, 0);
      lcd.write((byte)10);  // Heart symbol
      lcd.write((byte)11);
      houseDoorServo.write(160);  // Open the house door
      cardoortimerStart = millis();     // Start the timer for door
      isDoorOpen = true;
      inhouse = true;
      //retrycount = 0;
    } else if (enteredPassword == changePassword) {  // Initiate password
change
      lcd.clear();
      lcd.print("Enter New Pass:");
      enteredPassword = "";
      isChangingPassword = true;
    } else { if (enteredPassword != changePassword){
      lcd.clear();
      lcd.print("Access Denied");
      lcd.setCursor(13, 0);
      lcd.write((byte)7);  // Lock symbol
      backtimerStart = millis();
      tone(buzzerPin, 1000);  // Start buzzer
      passbuzzertimerStart = millis();  // Start buzzer timer
      isTimerActive = true;
      enter = true;
      retryCount++;
    } if (retryCount >= maxRetries) {
        isLockedOut = true;
        lcd.clear();
        lcd.print("System Locked!");
        lockoutStart = millis();  // Start lockout timer
      }
    }
    enteredPassword = "";  // Clear the password
  } else if (key == '5') {  // Clear the entered password
    enteredPassword = "";
    isTimerActive = true;
    enter = true;
    lcd.setCursor(0,0);
    lcd.print("Password Cleared ");
    backtimerStart = millis();  // Start clear message timer
  } else {  // Add key to password
    if (enteredPassword.length() < 16) {  // Avoid overflow
      enteredPassword += key;
      enter = true;
      nochange = true;
      backtimerStart = millis();  // Start clear message timer
      lcd.clear();
      lcd.print("Password: ");
      lcd.print(enteredPassword);
    }
  }
}


void handlePasswordChange(char key) {
  if (key == '4') {  // Confirm new password
    if (!confirmNewPassword) {
      newPassword = enteredPassword;
      enteredPassword = "";
      confirmNewPassword = true;
      lcd.clear();
      lcd.print("Confirm New Pass:");
    } else {
      if (enteredPassword == newPassword) {
        correctPassword = newPassword;
        lcd.clear();
        lcd.print("Password Changed");
        passwordChangeTimer = millis();  // Start timer for success message
        enteredPassword = "";
        isPasswordChangeTimerActive = true;
        isChangingPassword = false;
        confirmNewPassword = false;
      } else {
        lcd.clear();
        lcd.print("Mismatch");
        passwordChangeTimer = millis();  // Start timer for mismatch
message
        isPasswordChangeTimerActive = true;
        enteredPassword = "";  // Reset entered password
        confirmNewPassword = false;  // Allow retry
      }
    }
  } else {  // Add key to new password
    enteredPassword += key;
    lcd.clear();
    lcd.print("Password: ");
    lcd.print(enteredPassword);
  }
}
```