Smart Home Environment Monitor

FABLAB TECHUP SKILLS

GROUP 2: CRETECH MINDS

- **MARIE CLAIRE**
- **IGNACE**
- **GAGA ANDY**
- **DOLPHE**

1. Introduction

This project is a smart home monitor built with Arduino. It tracks temperature, light, and sound using sensors and shows the data on an LCD. Alerts are triggered with a buzzer and LED when values go beyond safe limits. The system is tested in Tinkercad using potentiometers to simulate sound input.

2. Project Overview

The Smart Home Environment Monitor is an Arduino-based system designed to track indoor temperature, light intensity, and sound levels. It uses analog and digital sensors to provide real-time feedback via an LCD display and triggers alerts using a buzzer and LED when environmental thresholds are exceeded. This project is ideal for DIY home automation and is fully testable in Tinkercad using simulation workarounds.

3. Project Objectives

- Monitor temperature, light, and sound levels in a home environment

- Display sensor data on an LCD screen in real-time
- Trigger alerts using a buzzer and LED when thresholds are crossed
- Simulate sound input in Tinkercad using analog substitutes
- Demonstrate practical application of Arduino programming and sensor integration

4. Components Used

Input Sensors

Component	Arduino Pin	Function
Potentiometer 1	A0	Simulates temperature sensor
Photoresistor	A1	Measures ambient light
Potentiometer 2	A2	Simulates microphone/sound level
Push Button	D11	Detects sound events (digital)

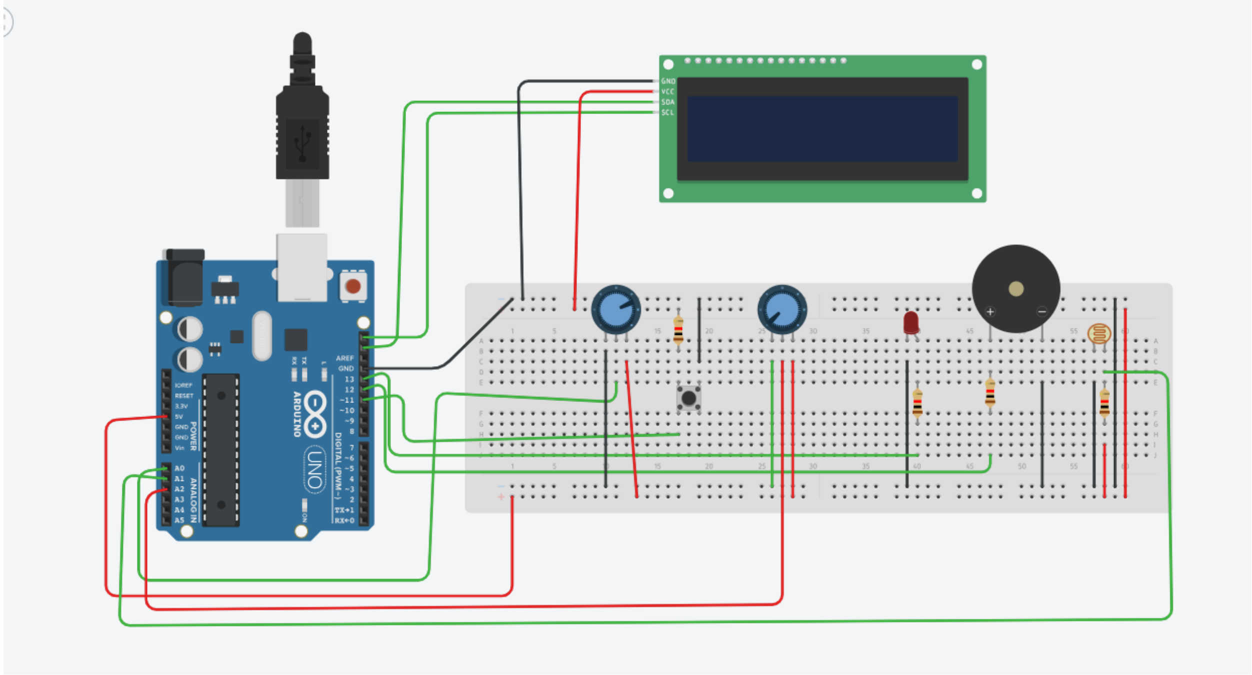
Output Devices

Component	Arduino Pin	Function
LED	D13	Visual alert
Buzzer	D12	Audible alert
LCD (I2C, 0x27)	SDA/SCL	Displays sensor data and alerts

Additional Components

- Arduino Uno R3
- Breadboard
- Jumper wires
- Resistors (10kΩ for pull-up, 220Ω for LED)

5. Circuit Design



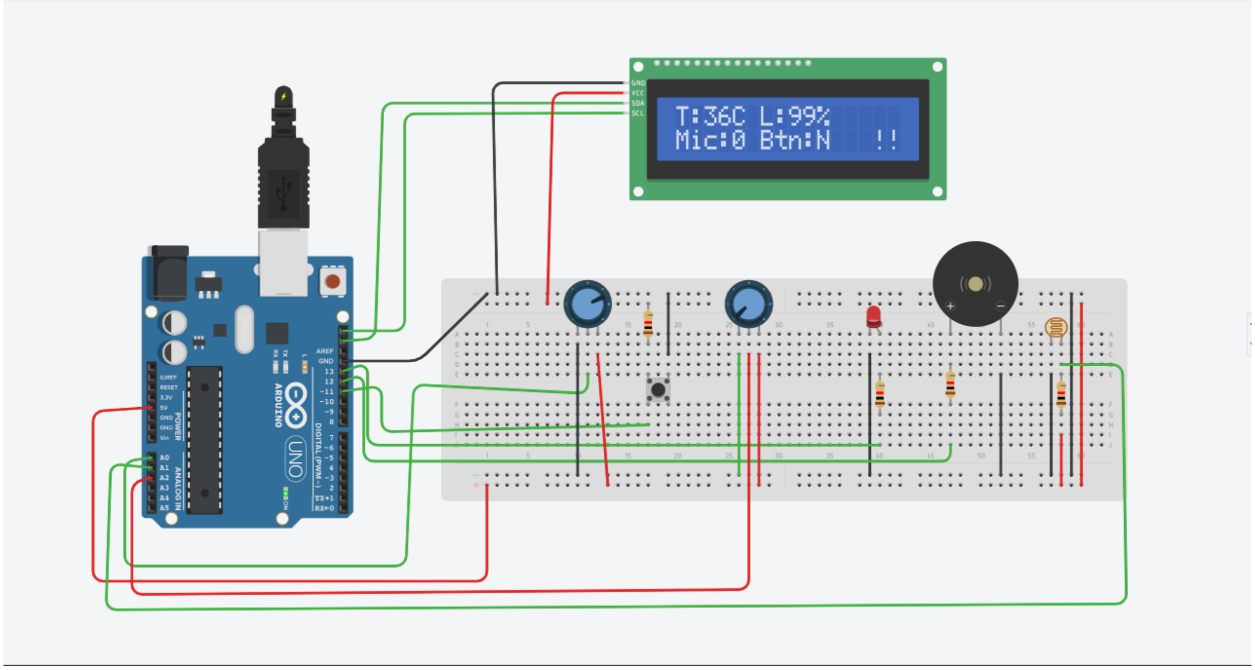
Pin Configuration Summary:

- **Analog Inputs:** A0 (Temperature), A1 (Light), A2 (Sound simulation)
- **Digital Inputs:** D11 (Sound button)
- **Digital Outputs:** D12 (Buzzer), D13 (LED)
- **I2C Communication:** SDA/SCL (LCD display)

Power Requirements:

- Operating voltage: 5V DC
- Current consumption: ~50mA (without alerts), ~80mA (with alerts active)

6. Simulation Notes (Tinkercad)



Tinkercad does not support microphone sensors, so this project uses creative simulation methods:

- **Temperature Simulation:** Potentiometer 1 on A0 mimics temperature sensor readings
- **Sound Simulation:** Potentiometer 2 on A2 simulates analog microphone input
- **Sound Detection:** Push button provides binary sound event detection
- **Testing Capability:** This setup allows full testing of alert logic and display behavior

7. Alert System Configuration

The system activates the buzzer and LED alerts under the following conditions:

Parameter	Threshold	Alert Action
Temperature	> 35°C	LED + Buzzer activation
Light Level	> 80%	LED + Buzzer activation
Sound Button	Pressed	LED + Buzzer activation
Simulated Microphone	> 70 (analog value)	LED + Buzzer activation

8. Software Implementation

Key Programming Features:

- **Analog-to-Digital Conversion:** Sensor readings mapped to meaningful units
- **Real-time Display:** Continuous LCD updates every 500ms
- **Serial Communication:** Debug output via Serial Monitor
- **Threshold Monitoring:** Constant comparison against preset limits
- **Multi-sensor Integration:** Simultaneous monitoring of all inputs

Code Structure:

```
#include <LiquidCrystal_I2C.h>


// Setup LCD
LiquidCrystal_I2C lcd(0x27, 16, 2);


void setup() {
```

```
  // Start serial and LCD
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();

  // Setup pins
  pinMode(11, INPUT_PULLUP);  // Sound button
  pinMode(12, OUTPUT);       // Buzzer
  pinMode(13, OUTPUT);       // LED

  Serial.println("Smart Home Monitor Started!");
}

void loop() {
  // Read sensors
  int tempRaw = analogRead(A0);        // Potentiometer for temperature
  int lightRaw = analogRead(A1);       // LDR for light
  int micRaw = analogRead(A2);         // Second potentiometer simulating microphone
  bool soundPressed = !digitalRead(11);   // Sound button

  // Convert to useful values
  int temperature = map(tempRaw, 0, 1023, 0, 50);    // 0-50 degrees
  int lightLevel = map(lightRaw, 0, 1023, 0, 100);   // 0-100 percent
  int micLevel = map(micRaw, 0, 1023, 0, 100);      // Simulated sound level

  // Show on Serial Monitor
```

```
Serial.print("Temp: ");
Serial.print(temperature);
Serial.print("°C  Light: ");
Serial.print(lightLevel);
Serial.print("%  Sound Btn: ");
Serial.print(soundPressed ? "YES" : "NO");
Serial.print("  Mic Level: ");
Serial.println(micLevel);

// Show on LCD
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("T:");
lcd.print(temperature);
lcd.print("C L:");
lcd.print(lightLevel);
lcd.print("%");

lcd.setCursor(0, 1);
lcd.print("Mic:");
lcd.print(micLevel);
lcd.print(" Btn:");
lcd.print(soundPressed ? "Y" : "N");

// Check if alerts needed
bool alert = false;
```

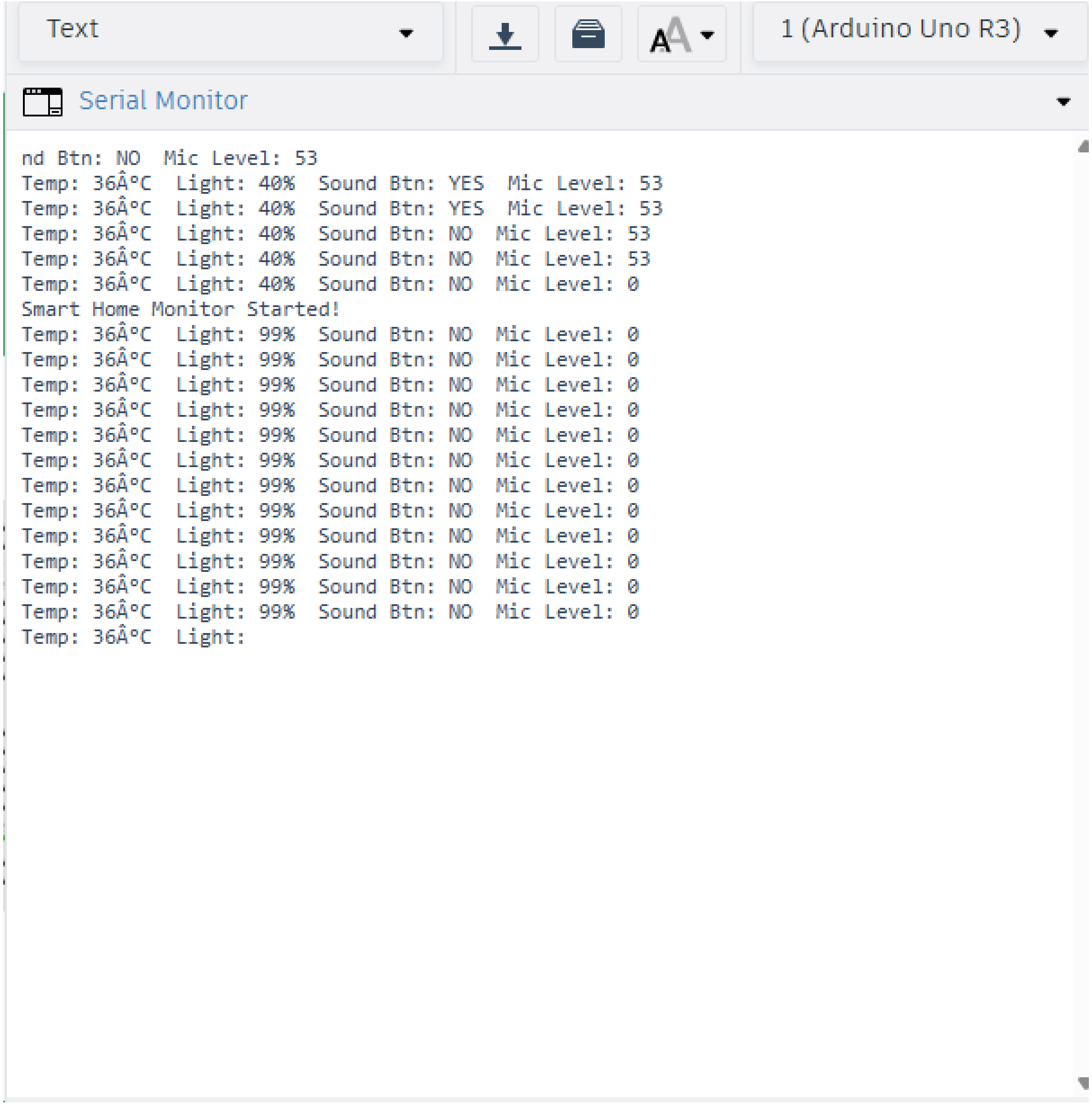
```
  if (temperature > 35) alert = true;     // Too hot
  if (lightLevel > 80) alert = true;      // Too bright
  if (soundPressed) alert = true;         // Sound button pressed
  if (micLevel > 70) alert = true;        // Loud simulated sound

  // Turn on buzzer and LED if alert
  if (alert) {
    digitalWrite(13, HIGH);   // LED on
    tone(12, 1000);           // Buzzer on
    lcd.setCursor(14, 1);
    lcd.print("!!");          // Alert sign on LCD
  } else {
    digitalWrite(13, LOW);    // LED off
    noTone(12);               // Buzzer off
  }

  delay(1000);  // Wait 1 second before checking again
}
```

9. Sample Output Examples

Serial Monitor Output:



```
nd Btn: NO  Mic Level: 53
Temp: 36Â°C  Light: 40%  Sound Btn: YES  Mic Level: 53
Temp: 36Â°C  Light: 40%  Sound Btn: YES  Mic Level: 53
Temp: 36Â°C  Light: 40%  Sound Btn: NO  Mic Level: 53
Temp: 36Â°C  Light: 40%  Sound Btn: NO  Mic Level: 53
Temp: 36Â°C  Light: 40%  Sound Btn: NO  Mic Level: 0
Smart Home Monitor Started!
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light: 99%  Sound Btn: NO  Mic Level: 0
Temp: 36Â°C  Light:
```

10. conclusion

The Smart Home Environment Monitor successfully demonstrates how Arduino can be used to track and respond to environmental changes in real time. By combining sensors, output devices, and creative simulation techniques in Tinkercad, the project offers a practical and educational solution for smart home automation. It’s easy to build, test, and expand it, making it a great foundation for future innovations in home monitoring systems.