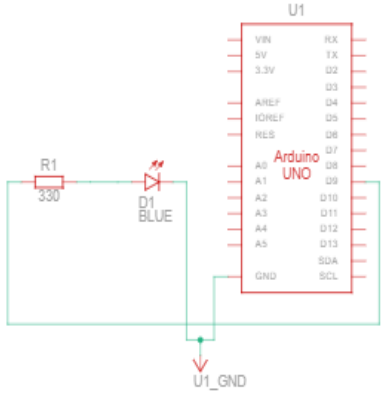
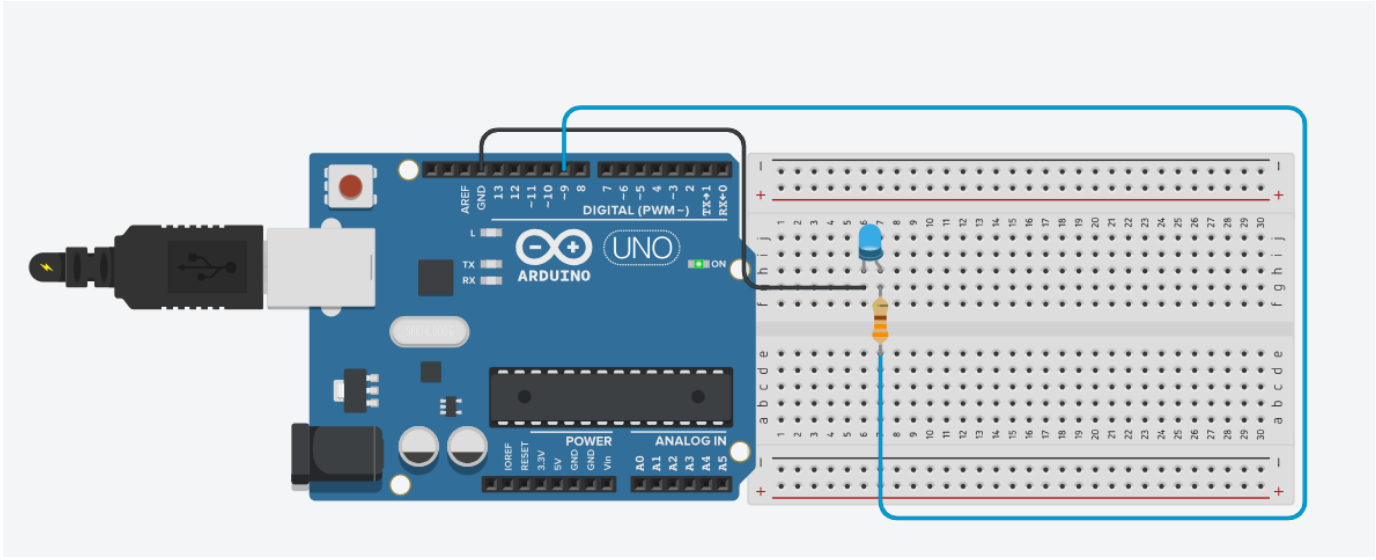
pro1:LED Blink

Text 1 (Arduino Uno R3)

```
1
2  void setup() {
3    pinMode(9, OUTPUT); // Initialize digital pin 9 as an output
4   }
5   void loop() {
6    digitalWrite(9, HIGH); // Turn the LED on
7    delay(1000); // Wait for a second
8    digitalWrite(9, LOW); // Turn the LED off
9    delay(1000); // Wait for a second
10  }
```

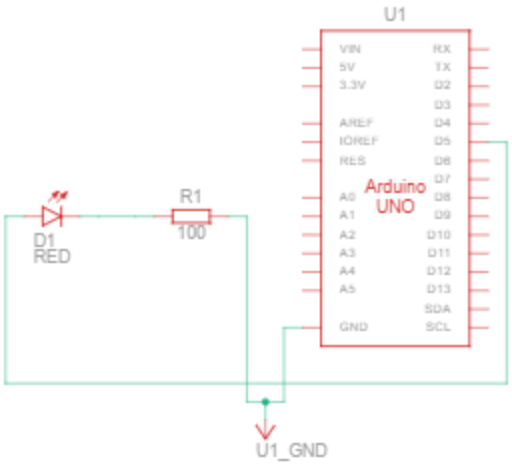


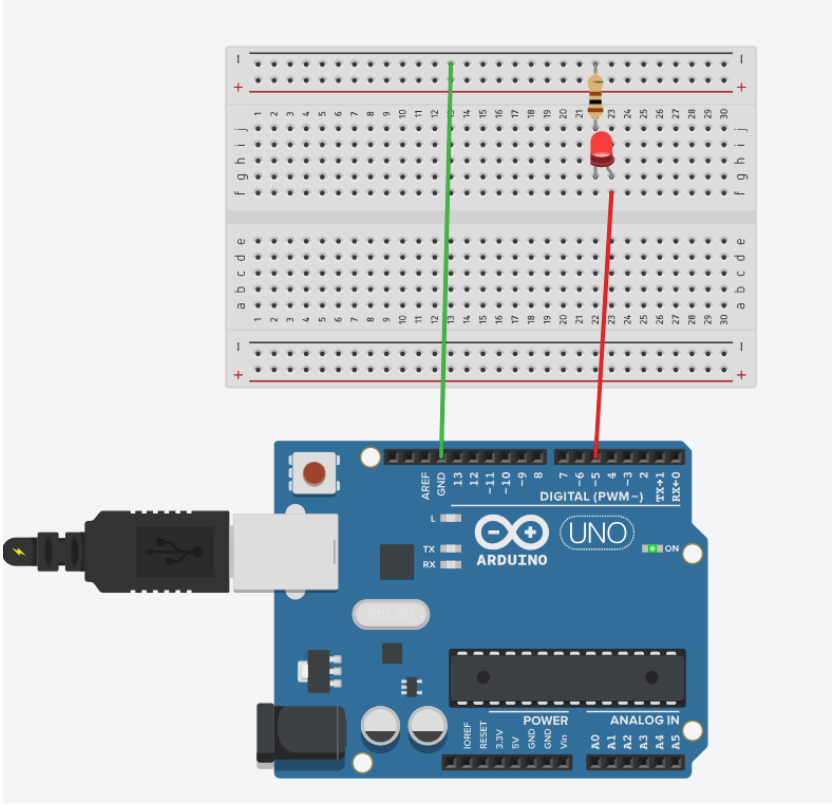


pro2: breathing light

Text | 1 (Arduino Uno R3)

```
int ledPin = 5; // Define the LED pin at D5
 void setup () {
  pinMode(ledPin, OUTPUT); // Initialize ledPin as an output
 }
 void loop () {
  for (int value = 0; value < 255; value++) {
    analogWrite(ledPin, value); // Gradually increase LED brightnes
    delay(5); // Delay 5ms
  }
  for (int value = 255; value > 0; value--) {
    analogWrite(ledPin, value); // Gradually decrease LED brightnes
    delay(5); // Delay 5ms
  }
 }
```

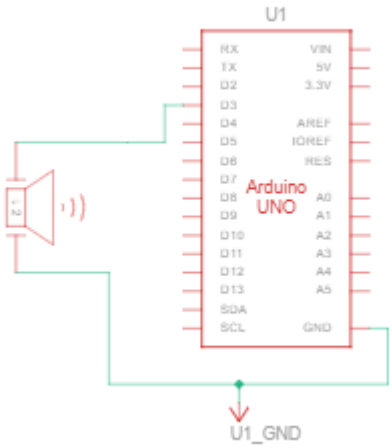


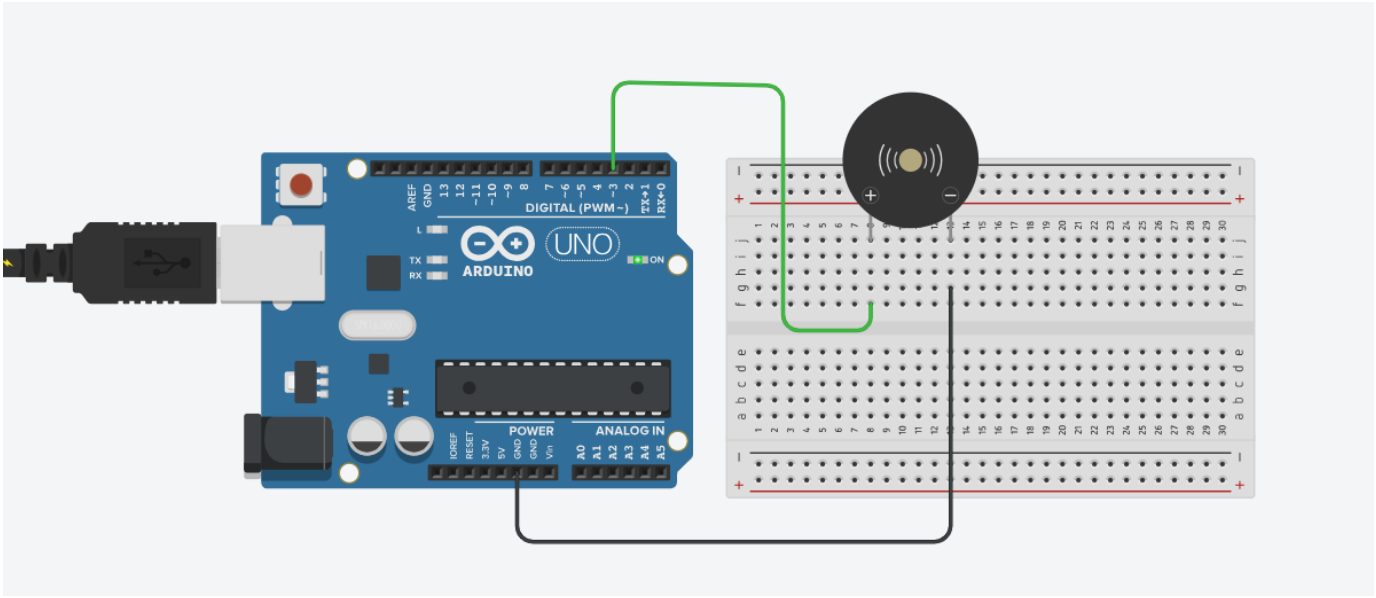


pro3: Passive Buzzer

Text | 1 (Arduino Uno R3)

```
int tonePin = 3; // Set the Pin of the buzzer to digital D3
void setup () {
  pinMode(tonePin, OUTPUT); // Set the digital IO pin mode to outpu
 }
 void loop () {
  for (unsigned char i = 0; i < 80; i++) {
    digitalWrite(tonePin, HIGH); // Sound
    delay(1); // Delay 1ms
    digitalWrite(tonePin, LOW); // No sound
    delay(1); // Delay 1ms
  }
  for (unsigned char i = 0; i < 100; i++) {
    digitalWrite(tonePin, HIGH); // Sound
    delay(2); // Delay 2ms
    digitalWrite(tonePin, LOW); // No sound
    delay(2); // Delay 2ms
  }
 }
```

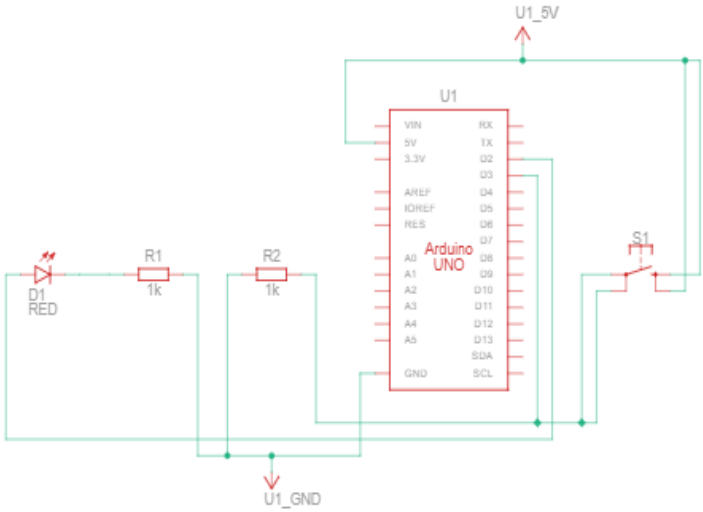


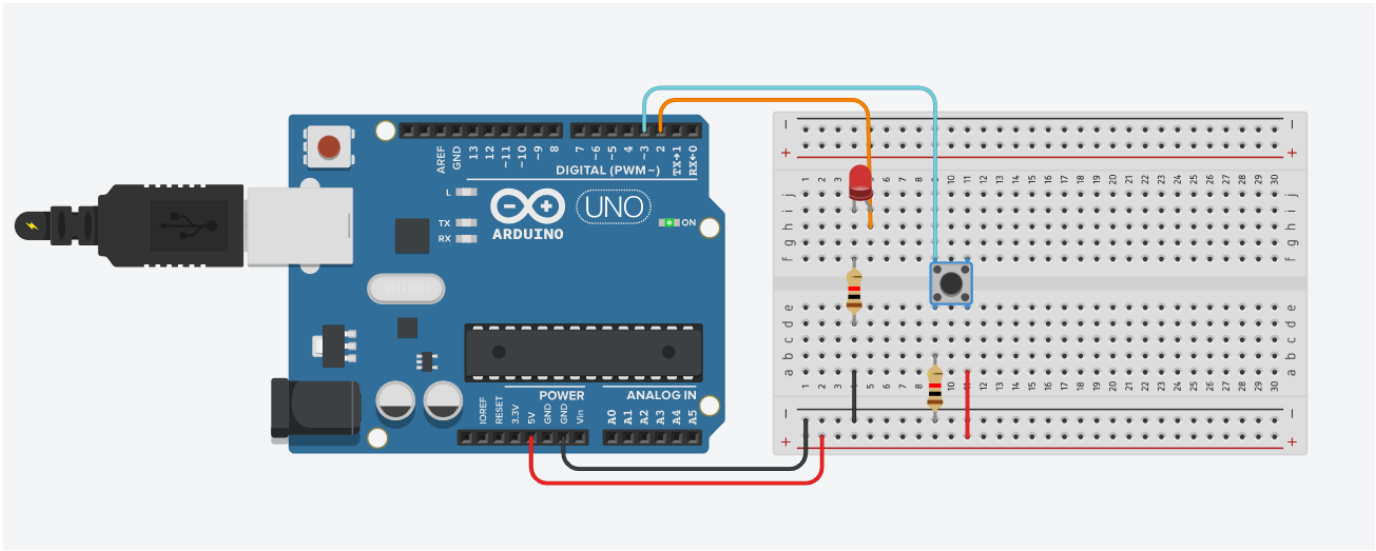


Marina Reda Abdullah 221101235 AIS

pro4:LED W/ Switch Button /Controlling LED By Button Module

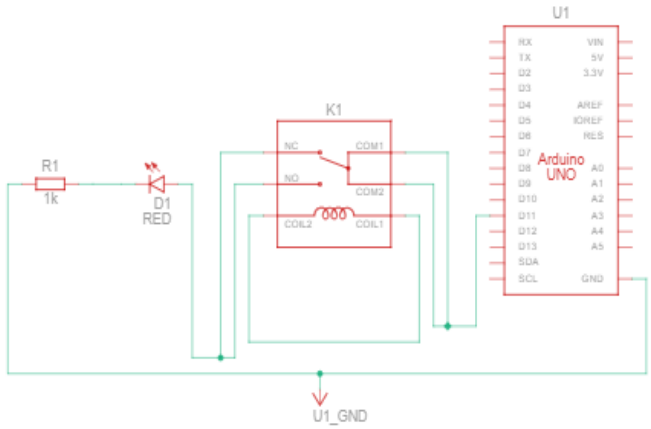
```
Text
1  void setup(){
2    pinMode(2, OUTPUT);
3    pinMode(3, INPUT);
4  }
5  void loop(){
6
7    if (digitalRead(3) == HIGH){
8    digitalWrite(2, HIGH);
9        delay(1000); // Wait for a second
10
11   }
12
13   else{
14
15     digitalWrite(2, LOW);
16
17 }
18  }
```

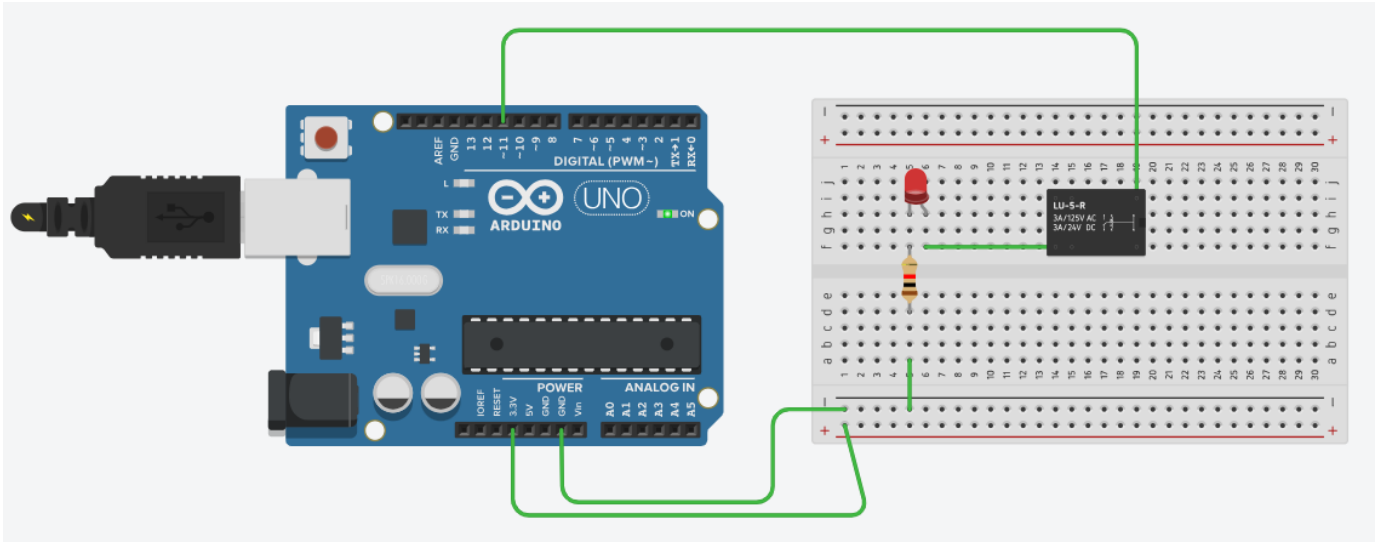




pro5: relay module

```
// C++ code
//
void setup()
{
  pinMode(11, OUTPUT);
}

void loop()
{
  digitalWrite(11,1);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(11,0);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

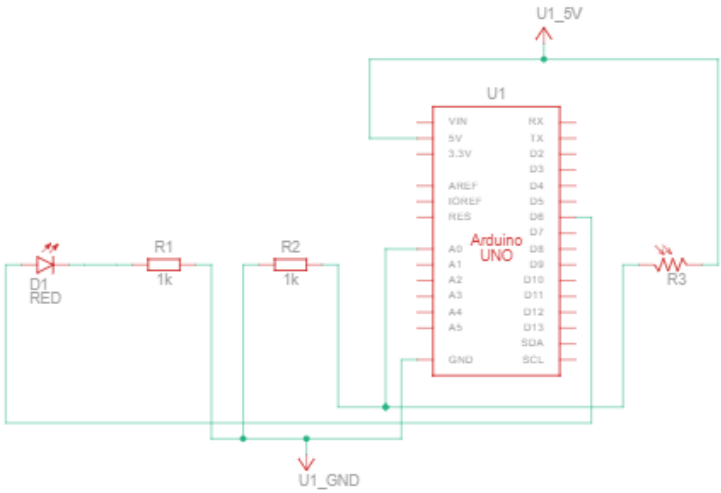


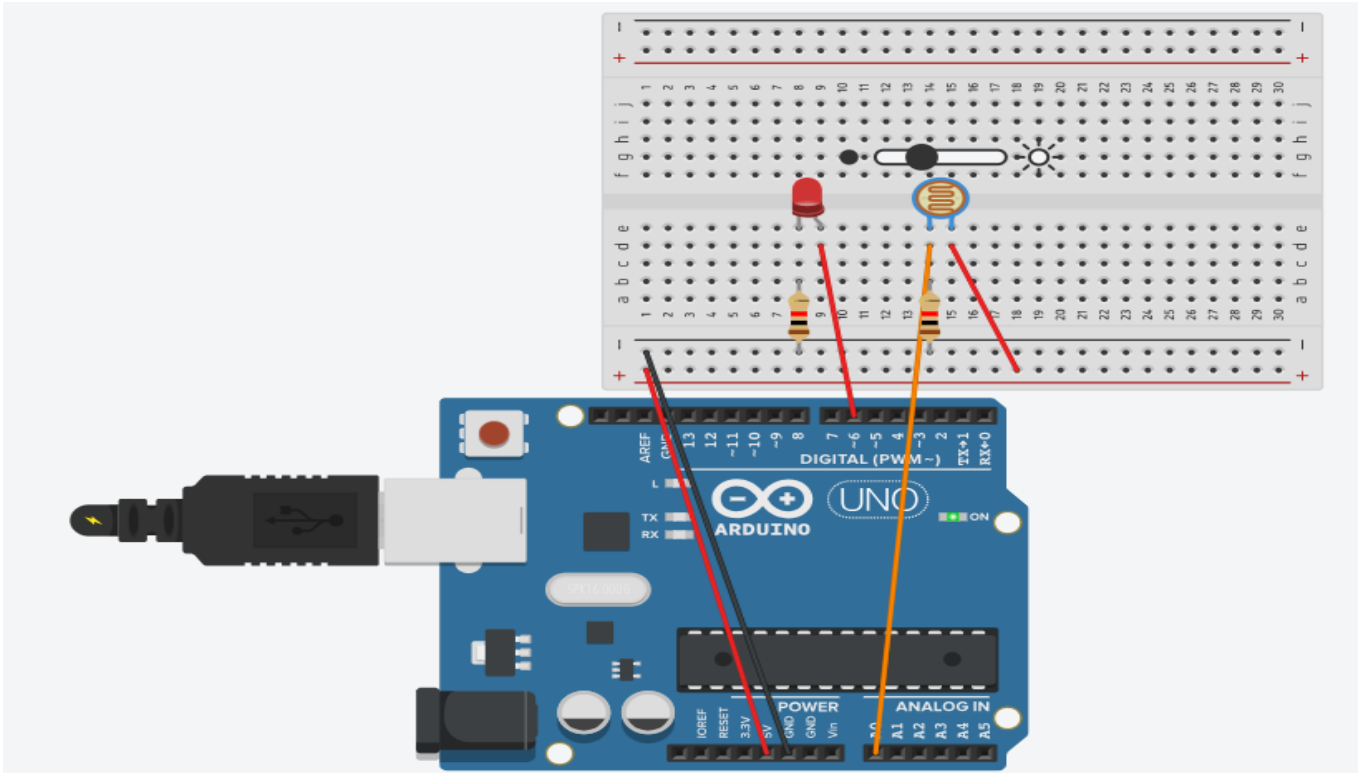


pro6: LED and Photocell

Text | 1 (Arduino Uno R3)

```
#define LED 6  // LED connected to pin 6
int sensorValue;
double VoltageValue;
double ResValue;
int Intensity;
int MaxIntensity = 255;

// Define a threshold for light intensity (adjust as needed)
const int DaylightThreshold = 500; // Example value; you can adjust

void setup() {
  pinMode(LED, OUTPUT);  // Set pin 6 as output for LED
  Serial.begin(9600);    // Initialize serial communication
}

void loop() {
  // Read the analog value from the photocell (connected to pin A0)
  sensorValue = analogRead(A0);
  VoltageValue = sensorValue * 5.0 / 1023;
  ResValue = 1000 * (5 / VoltageValue - 1);

  // Print out the resistance value
  Serial.print("Resistance: ");
  Serial.print(ResValue);

  // Determine whether it's daytime or nighttime
  if (sensorValue < DaylightThreshold) {
    // It's nighttime (low light intensity)
    Intensity = MaxIntensity; // Turn on the LED
    Serial.println("    LED turned ON");
  } else {
    // It's daytime (high light intensity)
    Intensity = 0; // Turn off the LED
    Serial.println("    LED turned OFF");
  }

  // Set the LED intensity
  analogWrite(LED, Intensity);

  delay(1000); // Wait for 1 second before the next iteration
}
```

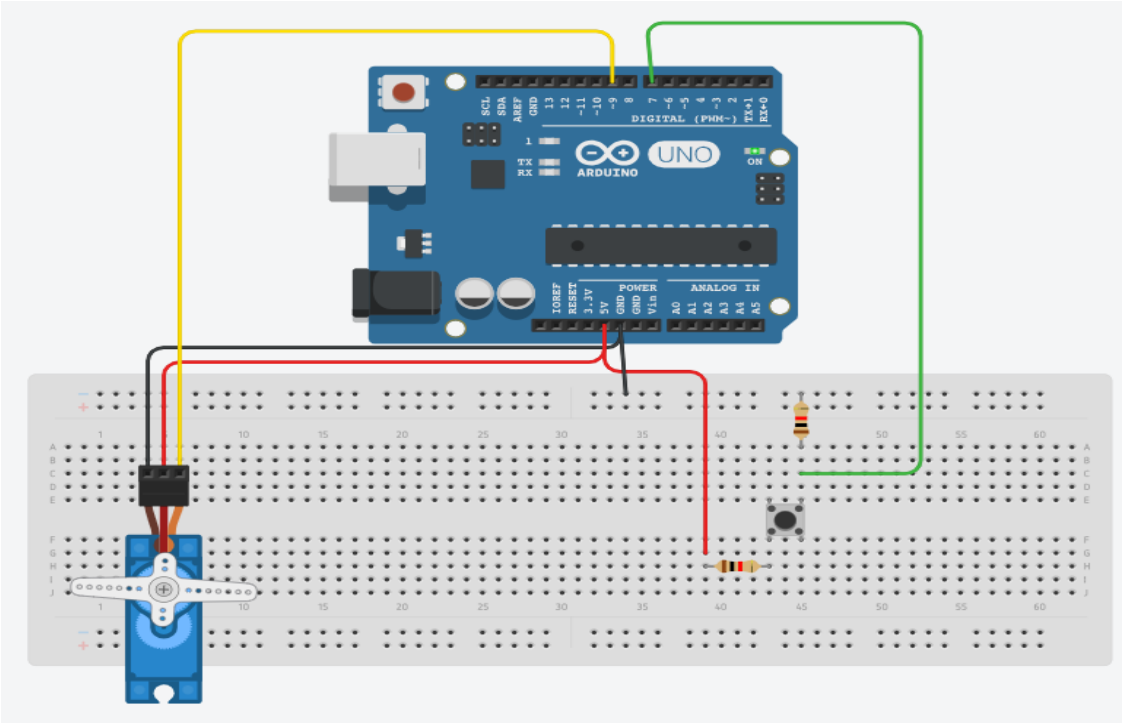




Pro7 : Adjusting Motor Servo Angle

Text 1 (Arduino uno)

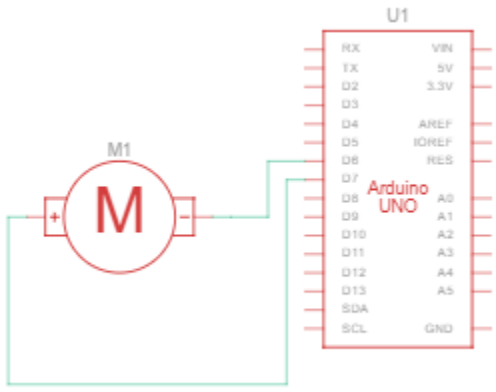
```
#include <Servo.h>

Servo myservo;
int value;
int angle = 0;

int inPin = 7;
void setup()
{
  pinMode(inPin, INPUT);
  Serial.begin(9600);
  myservo.attach(9);
}

void loop()
{


  int val = digitalRead(inPin);  // read input value
  //Serial.println(val);

  if (val == HIGH) {         // check if the input is HIGH (button

    angle = (angle+90) % 180;
    Serial.println(angle);
    myservo.write(angle);
    delay(500);
  }
}
```

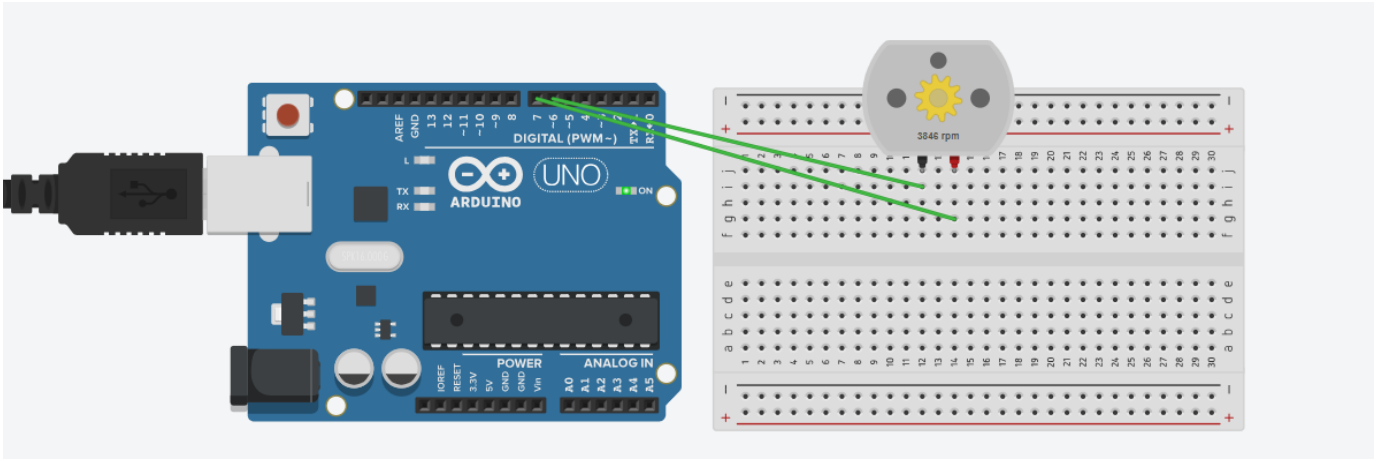


Pro8: Fan Module

Text 1 (Arduino Uno R3)

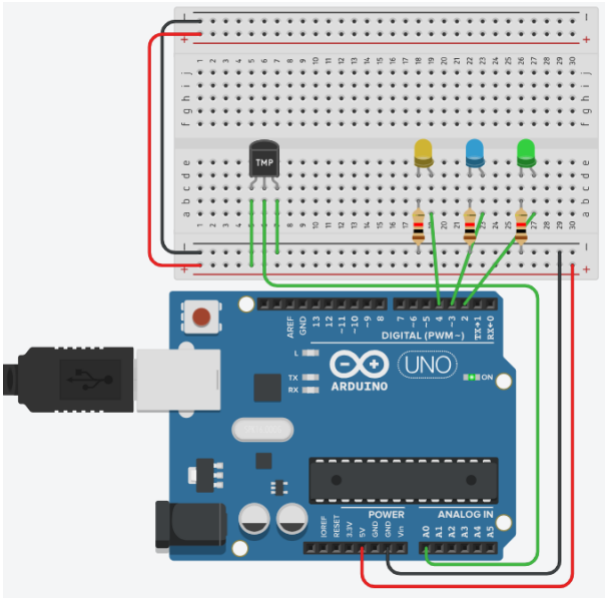
```
1  void setup () {
2     pinMode (7, OUTPUT); //define D7 pin as output
3     pinMode (6, OUTPUT); //define  D6 pin as output
4   }
5   void loop () {
6     digitalWrite (7, LOW);
7     digitalWrite (6, HIGH); // Reverse rotation of the motor
8     delay (3000); // delay 3S
9     digitalWrite (7, LOW);
10    digitalWrite (6, LOW); // The motor stops rotating
11    delay (1000); //delay 1S
12    digitalWrite (7, HIGH);
13    digitalWrite (6, LOW); // The motor rotates in the forward dire
14    delay (3000); // delay 3S
15    digitalWrite (7, LOW);
16    digitalWrite (6, LOW); // The motor stops rotating
17    delay (1000); //delay 1S
18  }
```

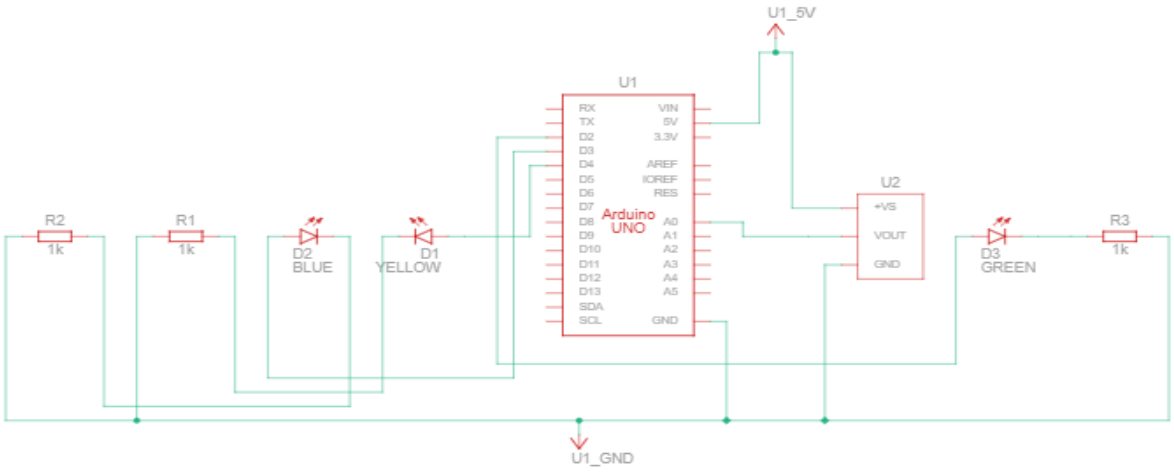




Pro9：Steam Sensor /TMP36 Temperature Sensor With Arduino

```
Text                                                    1 (Arduino Uno R3)

int baselineTemp = 0;
int celsius = 0;
int fahrenheit = 0;
void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}
void loop()
{
  baselineTemp = 40;
  celsius = map(((analogRead(A0) - 20) * 3.04), 0, 1023, -40, 125);
   fahrenheit = ((celsius * 9) / 5 + 32);
  Serial.print(celsius);
  Serial.print(" C, ");
  Serial.print(fahrenheit);
  Serial.println(" F");
   if (celsius < baselineTemp) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp && celsius < baselineTemp + 10) {
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 10 && celsius < baselineTemp + 20)
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 20 && celsius < baselineTemp + 30)
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  if (celsius >= baselineTemp + 30) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  delay(1000);
}
```

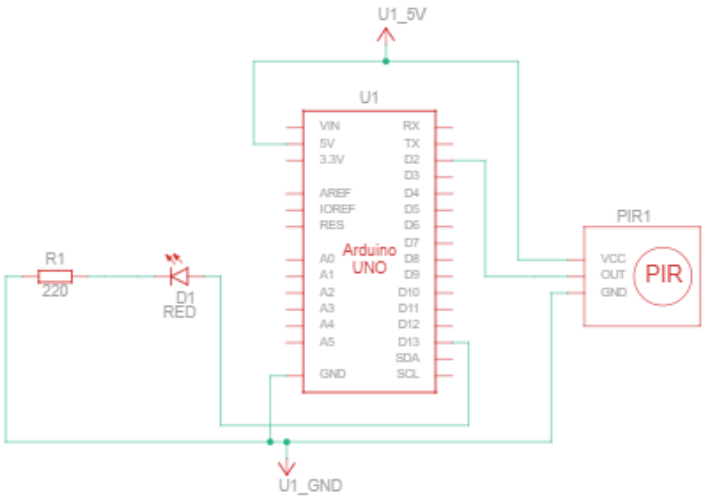


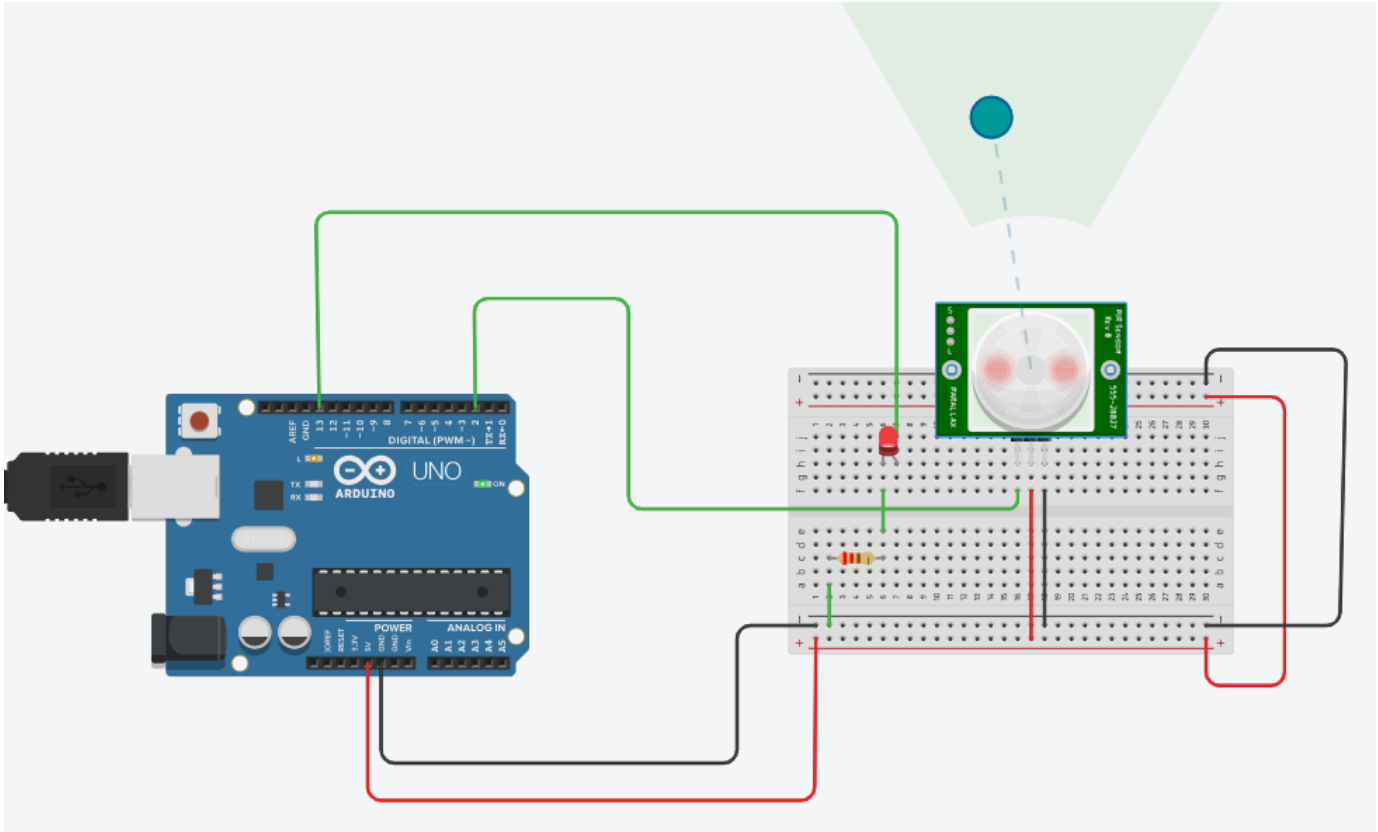


Pro10: PIR Motion Sensor

Text 1 (Arduino Uno R3)

```
int buttonState = 0;

void setup()
{
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
}

void loop()
{
  // read the state of the pushbutton
  buttonState = digitalRead(2);
  // check if pushbutton is pressed. if it is, the
  // button state is HIGH
  if (buttonState == HIGH)
  {
    digitalWrite(13, HIGH);
  }
  else
  {
    digitalWrite(13, LOW);
  }
  delay(10); // Delay a little bit to improve simulation performanc
}
```



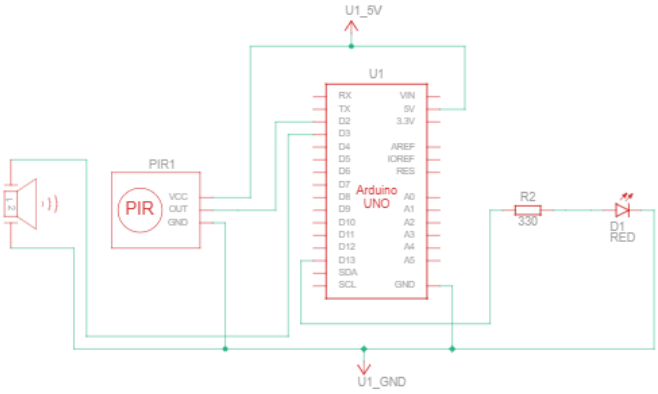


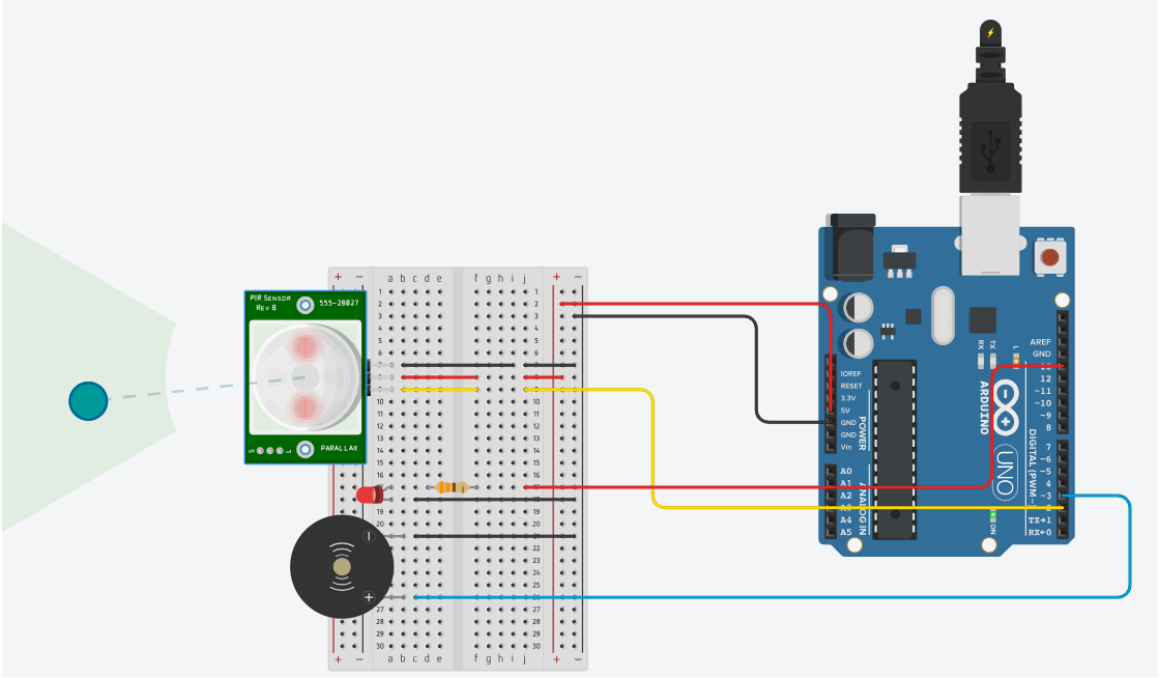
Marina Reda Abdullah 221101235 AIS

Pro11: PIR Motion Sensor and Piezo

Text | 1 (Arduino Uno R3)

```ino
int led = 13;
int sensor = 2;
int piezo = 3;

void setup() {
  pinMode(led, OUTPUT);
  pinMode(sensor, INPUT);
  pinMode(piezo, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int sensorval = digitalRead(sensor);
  Serial.println(sensorval);

  if (sensorval == HIGH) {
    digitalWrite(led, HIGH);
    tone(piezo, 200);     // Start the tone
    delay(50);            // Play tone for 50 milliseconds
    noTone(piezo);        // Stop the tone
  } else {
    digitalWrite(led, LOW);
    noTone(piezo);        // Ensure the tone is stopped
  }
}
```

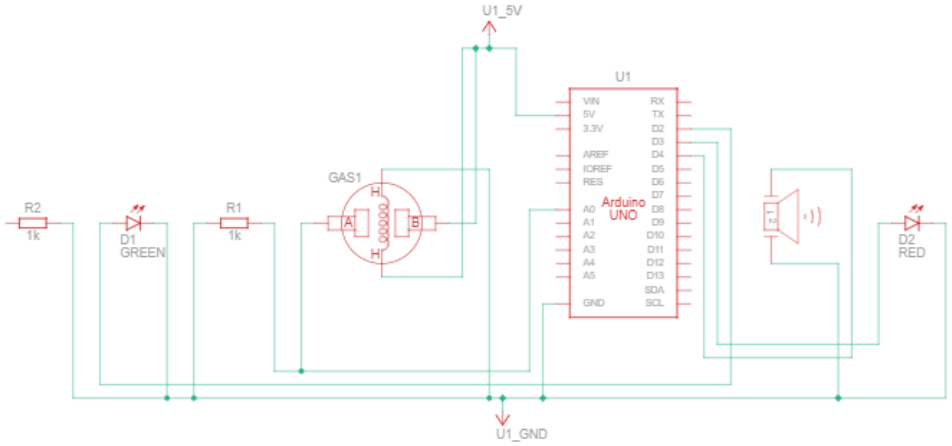


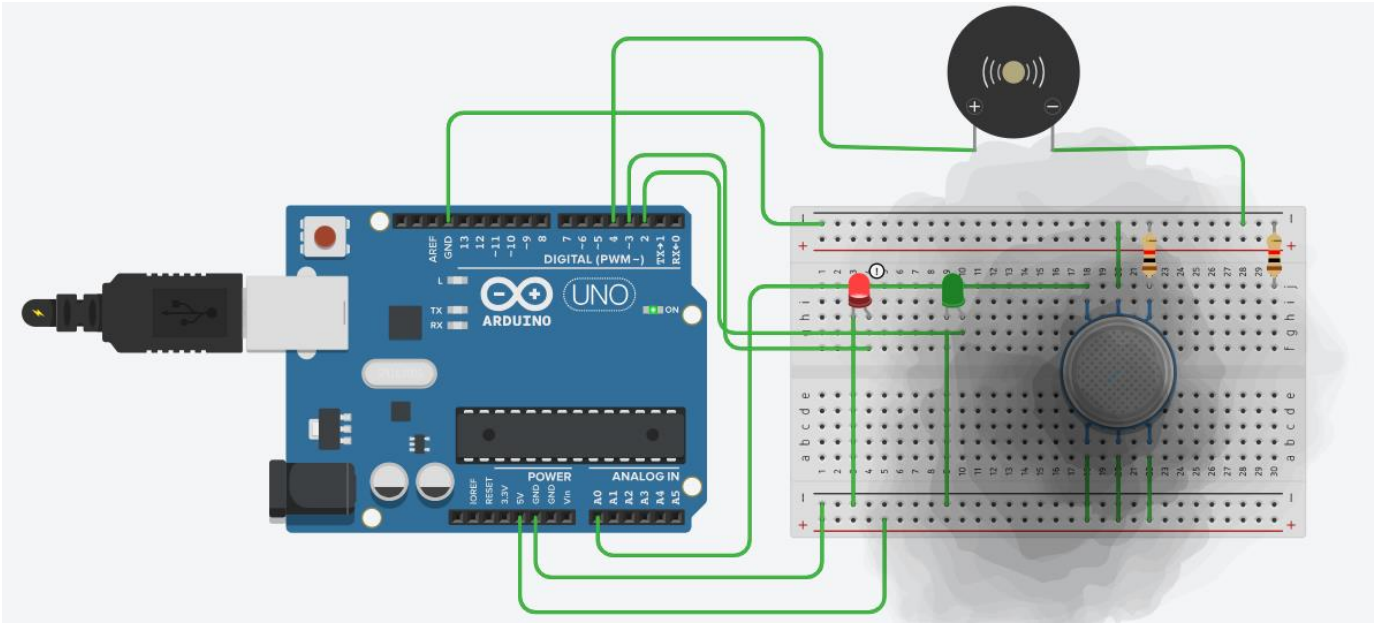


Pro12: Analog (MQ-2) Sensor

Text

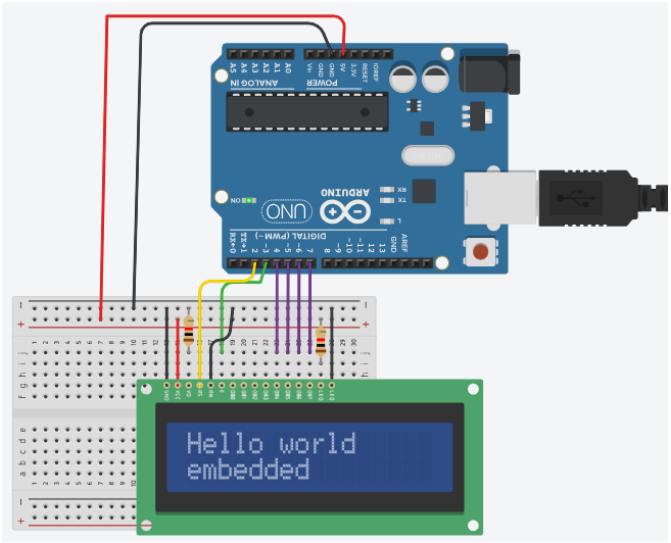
```
int redled=3;
int greenled=2;
int buz=4;
int sensor = A0;
int sensThre = 400;
void setup()
{
pinMode(redled, OUTPUT);
pinMode(greenled, OUTPUT);
pinMode(buz, OUTPUT);
pinMode(sensor, INPUT);
Serial.begin(9600);
}
void loop()
{
int sensValue=analogRead(sensor);
Serial.println(sensValue);
if (sensValue > 300)
{
digitalWrite(redled, HIGH);
tone(buz,10000,10);
digitalWrite(greenled, LOW);
}
else
{
digitalWrite(greenled, HIGH);
noTone(buz);
digitalWrite(redled, LOW);
}
}
```

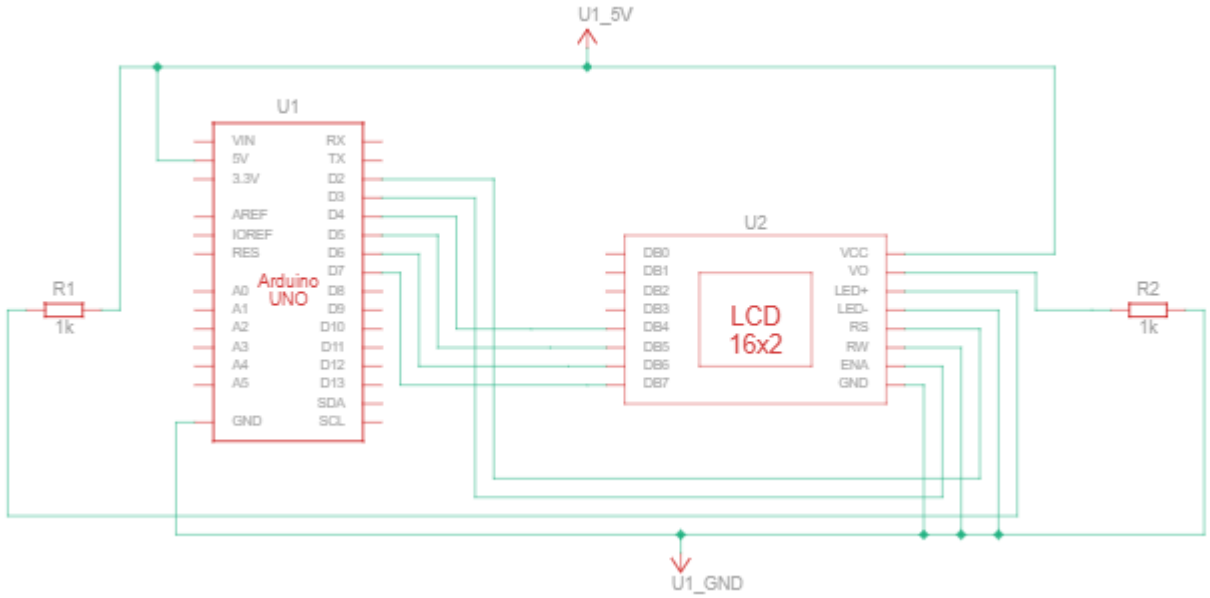




Pro13: 1602 LCD Display

```
#include <LiquidCrystal.h>

int RS = 2;
int E = 3;
int DB4 = 4;
int DB5 = 5;
int DB6 = 6;
int DB7 = 7;

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(RS, E, DB4, DB5, DB6, DB7);

void setup() {
  // The display has 16 columns and 2 rows
  lcd.begin(16, 2);

  // Print a message to the LCD
  lcd.print("Hello world");
  lcd.setCursor(0, 1);
  lcd.print("embedded");
}

void loop() {
  // No need to write code here as the message is static
}
```

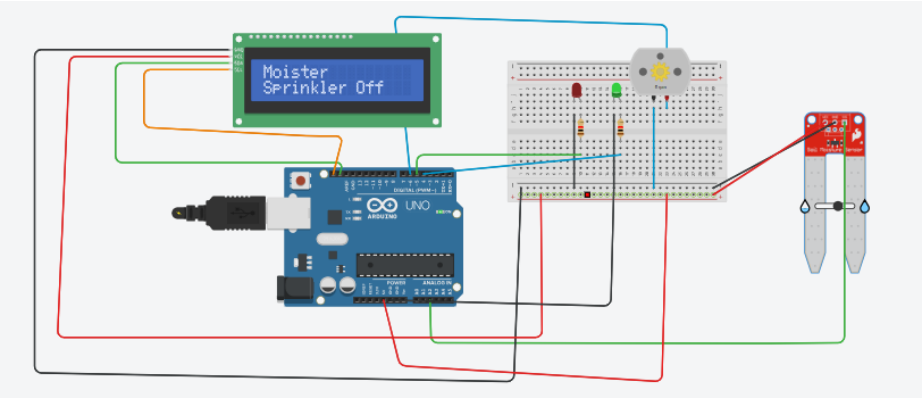


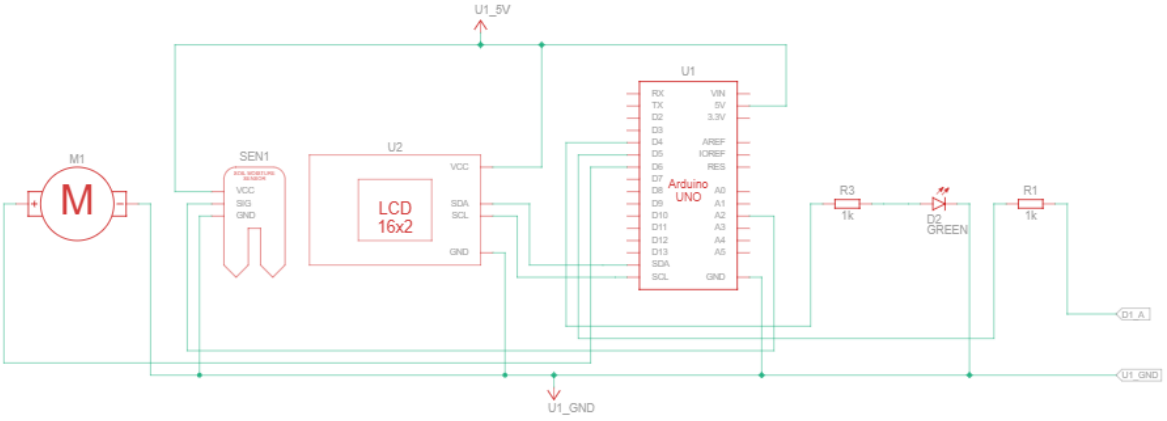


Pro14: Soil Humidity Sensor

Text | 1 (Arduino Uno R3)

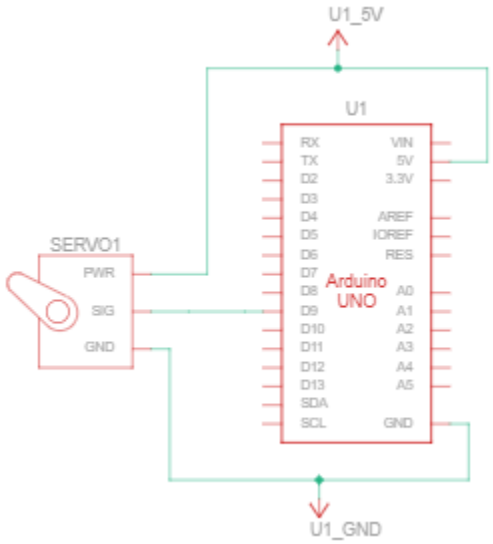
```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x20 , 16, 2);
int red=5;
int green=4;
int dc=6;
int sms=A2;
float x;
float moist;
void setup(){
  pinMode(red,OUTPUT);
  pinMode(green,OUTPUT);
  pinMode(dc,OUTPUT);
  pinMode(sms,INPUT);
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();
}
void loop()
{
  x=analogRead(sms);
  float moist=x*500.0/1023.0;
  if(moist<=300)
  {
     digitalWrite(red,HIGH);
     digitalWrite(green,LOW);
     digitalWrite(dc,HIGH);
       Serial.print(moist);
                 Serial.println("sprinkler on");
                 lcd.clear();
                 lcd.setCursor(0,0);
                 lcd.print("Moister");
                 Serial.println(moist);
                 lcd.setCursor(0,1);

                lcd.print("Sprinkler On");
                 }
                 else{
                    digitalWrite(red,LOW);
                    digitalWrite(green,HIGH);

                    digitalWrite(dc,LOW);
                     Serial.println(moist);
                 Serial.println("sprinkler off");
                 lcd.clear();
                 lcd.setCursor(0,0);
                 lcd.print("Moister");
                 Serial.print(moist);
                    lcd.setCursor(0,1);

                lcd.print("Sprinkler Off");
                 }delay(1000);}
```

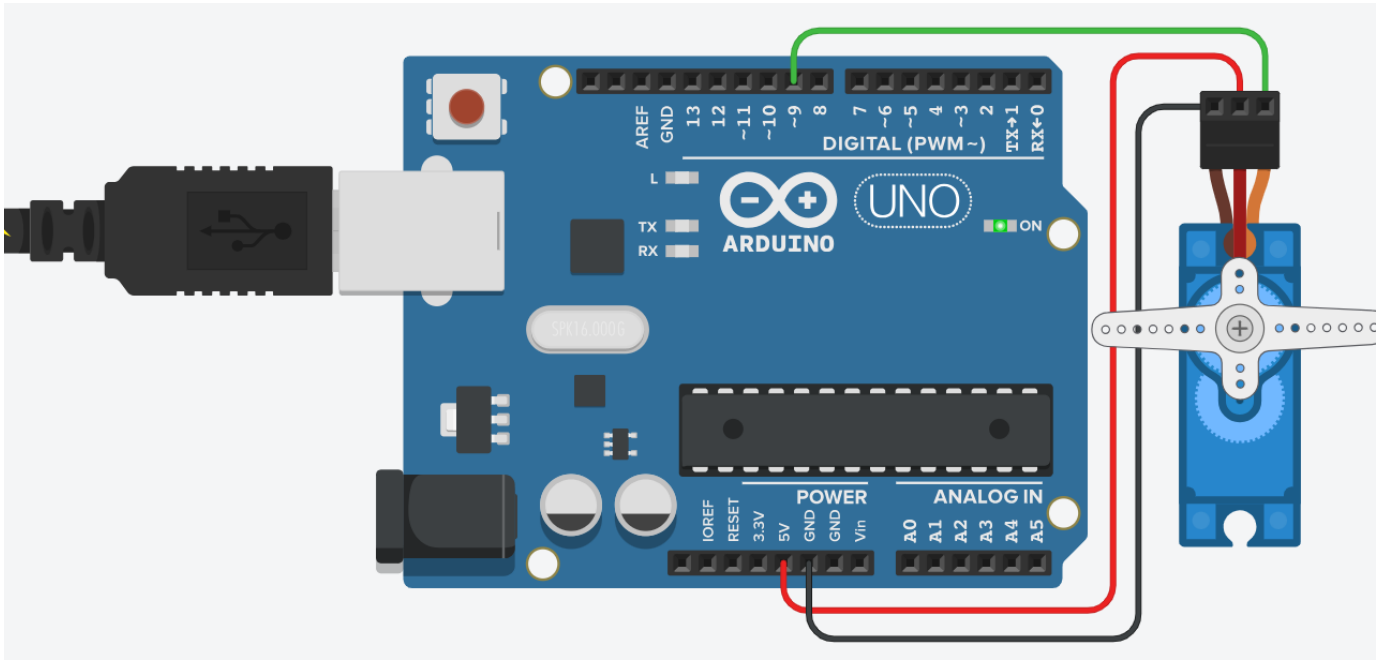




pro15: Bluetooth Test

```ino
// C++ code
//
#include <Servo.h>

int pos = 0;

int Bluetooth = 0;

Servo servo_9;

void setup()
{
  Serial.begin(9600);
  servo_9.attach(9, 500, 2500);
}

void loop()
{
  if (Bluetooth == 1) {
    for (pos = 0; pos <= 180; pos += 1) {
      Serial.println(pos);
      servo_9.write(pos);
      delay(15); // Wait for 15 millisecond(s)
    }
    for (pos = 180; pos >= 0; pos -= 1) {
      servo_9.write(pos);
      delay(15); // Wait for 15 millisecond(s)
    }
  }
  Serial.println(Bluetooth);
}
```

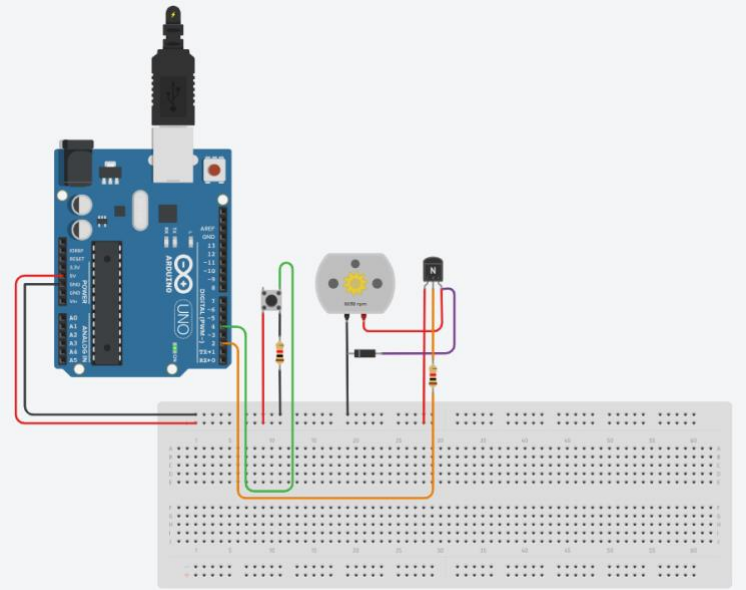


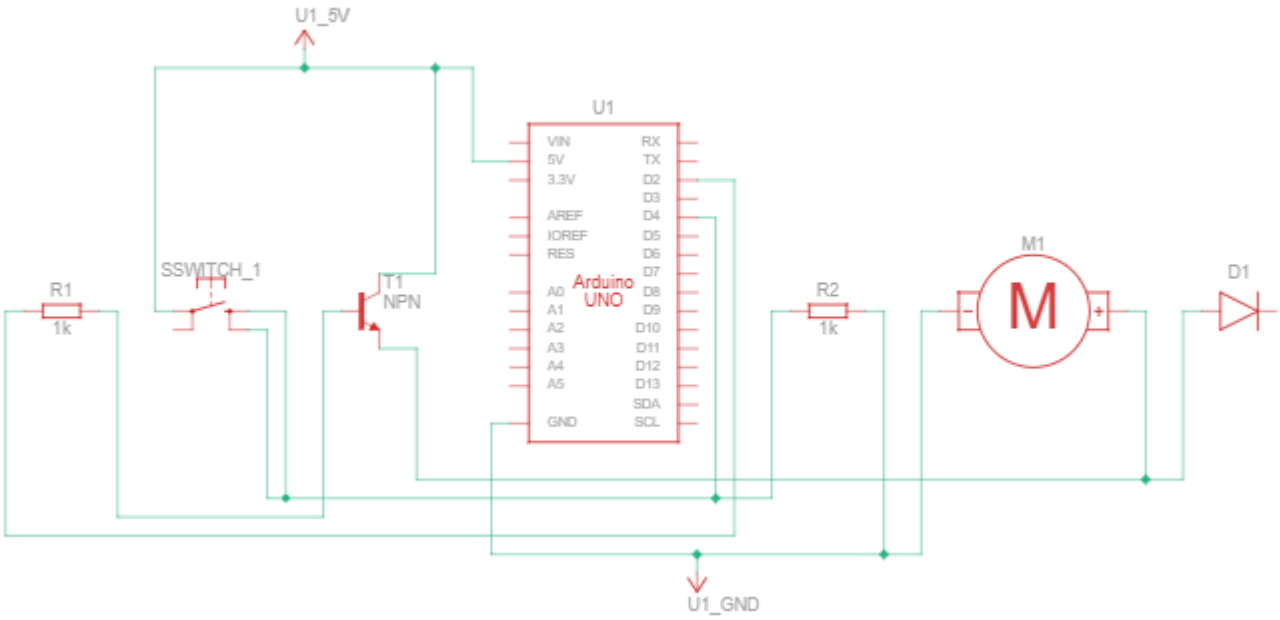


PRO 16:SWITCH AND FAN

Text | 1 (Arduino Uno R3)

```
int sw = 0;              // variable for the button
int anState = 0;         // variable for the anterior state of the
int swOff = 0;           // 0 = dc motor ON, 1 = dc motor OFF

void setup() {
  pinMode(4,INPUT);      // BUTTON as INPUT
  pinMode(2,OUTPUT);     // DC MOTOR as OUTPUT
}

void loop() {
  int sw = digitalRead(4);  // reading the state of the button

  if((sw == LOW) && (anState == HIGH)){
    swOff = 1 - swOff;
  }

  anState = sw;             // reads the actual value

  if(swOff == 1){
    digitalWrite(2,HIGH);
  }
  else {
    digitalWrite(2, LOW);
  }
}
```

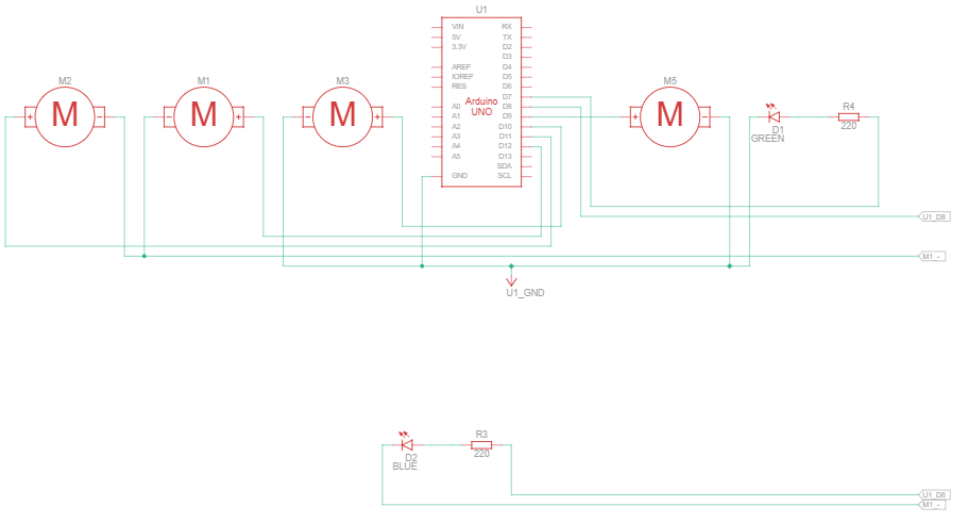


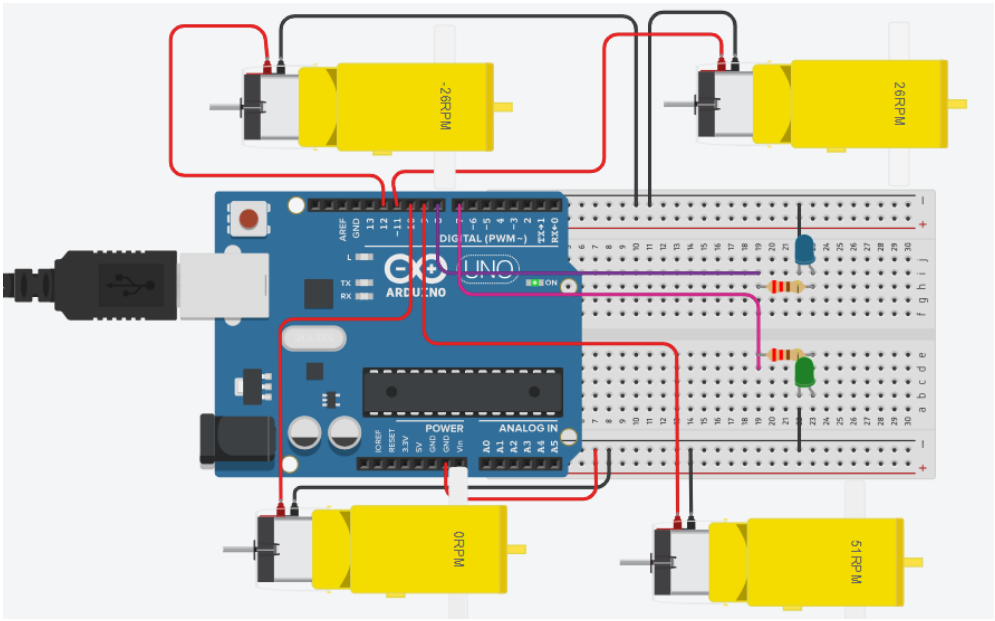


PRO 17:CIRCUIT CAR

```
Text
```

```
int mr1 = 12;
int mr2 = 11;
int mr3 = 10;
int mr4 = 9;
int LED1 = 8;
int LED2 = 7;

void setup() {
  pinMode(mr1, OUTPUT);
  pinMode(mr2, OUTPUT);
  pinMode(mr3, OUTPUT);
  pinMode(mr4, OUTPUT);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
}

void loop() {
  // Forward motion
  digitalWrite(mr1, HIGH);
  digitalWrite(mr2, LOW);
  digitalWrite(mr3, HIGH);
  digitalWrite(mr4, LOW);
  delay(1000);

  // Blink LED1
  digitalWrite(LED1, HIGH);
  delay(500);
  digitalWrite(LED1, LOW);
  delay(500);

  // Stop
  digitalWrite(mr1, LOW);
  digitalWrite(mr2, LOW);
  digitalWrite(mr3, LOW);
  digitalWrite(mr4, LOW);
  delay(500);

  // Backward motion
  digitalWrite(mr1, LOW);
  digitalWrite(mr2, HIGH);
  digitalWrite(mr3, LOW);
  digitalWrite(mr4, HIGH);
  delay(1000);

  // Blink LED2
  digitalWrite(LED2, HIGH);
  delay(500);
  digitalWrite(LED2, LOW);
  delay(500);

  // Stop
  digitalWrite(mr1, LOW);
  digitalWrite(mr2, LOW);
  digitalWrite(mr3, LOW);
  digitalWrite(mr4, LOW);
  delay(500);

  // Left motion
  digitalWrite(mr1, LOW);
  digitalWrite(mr2, HIGH);
  digitalWrite(mr3, HIGH);
  digitalWrite(mr4, LOW);
  delay(1000);

  // Right motion
  digitalWrite(mr1, HIGH);
  digitalWrite(mr2, LOW);
  digitalWrite(mr3, LOW);
  digitalWrite(mr4, HIGH);
  delay(1000);
}
```





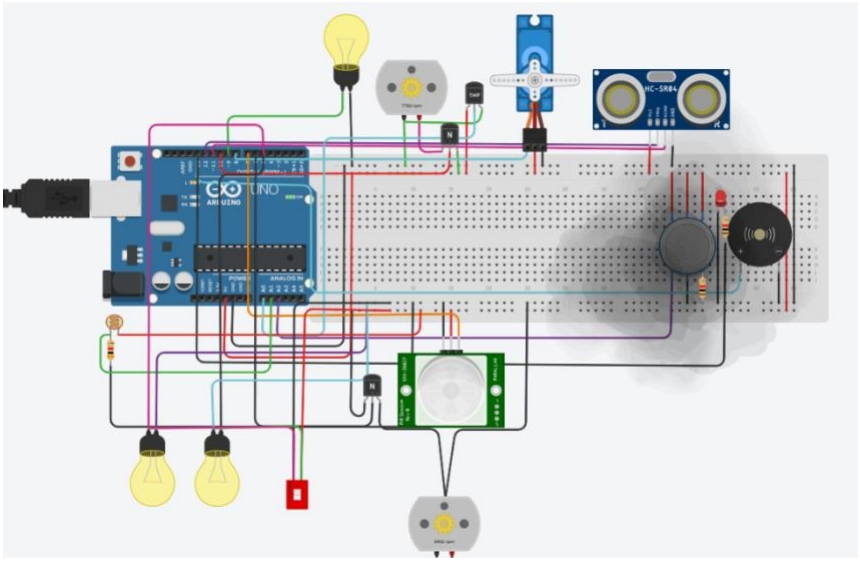
PRO 18: SMART HOME USING ARDUINO

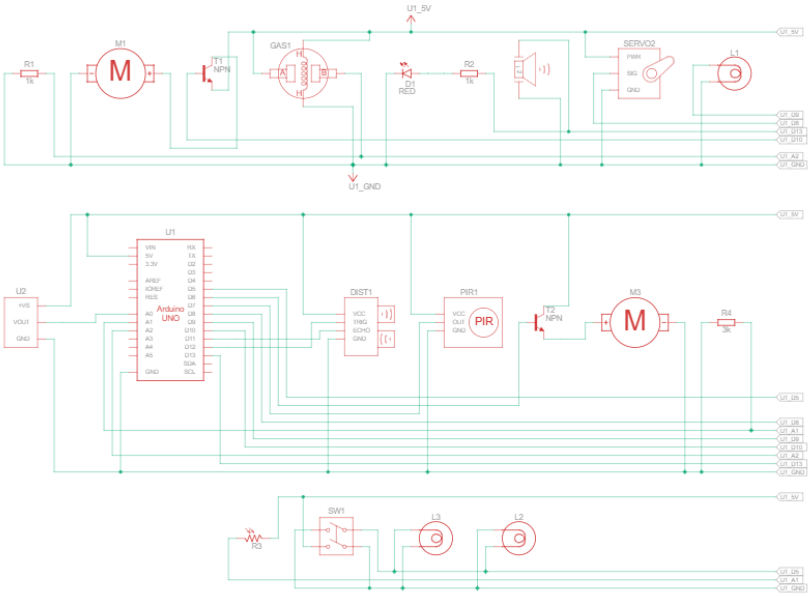
Text | 1 (Arduino Uno R3)

```ino
// C++ code
//
#include <Servo.h>

int dist = 0;

int temp = 0;

int tempconversion = 0;

int gas = 0;

int PR = 0;

int photoresistor = 0;

int button = 0;

long readUltrasonicDistance(int triggerPin, int echoPin)
{
  pinMode(triggerPin, OUTPUT);  // Clear the trigger
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  // Sets the trigger pin to HIGH state for 10 microseconds
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  pinMode(echoPin, INPUT);
  // Reads the echo pin, and returns the sound wave travel time
  return pulseIn(echoPin, HIGH);
}

Servo servo_8;

void setup()
{
  pinMode(A0, INPUT);
  pinMode(7, INPUT);
  pinMode(A2, INPUT);
  pinMode(A1, INPUT);
  pinMode(10, OUTPUT);
  servo_8.attach(8, 500, 2500);
  pinMode(9, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}
```

Text | 1 (Arduino Uno R3)

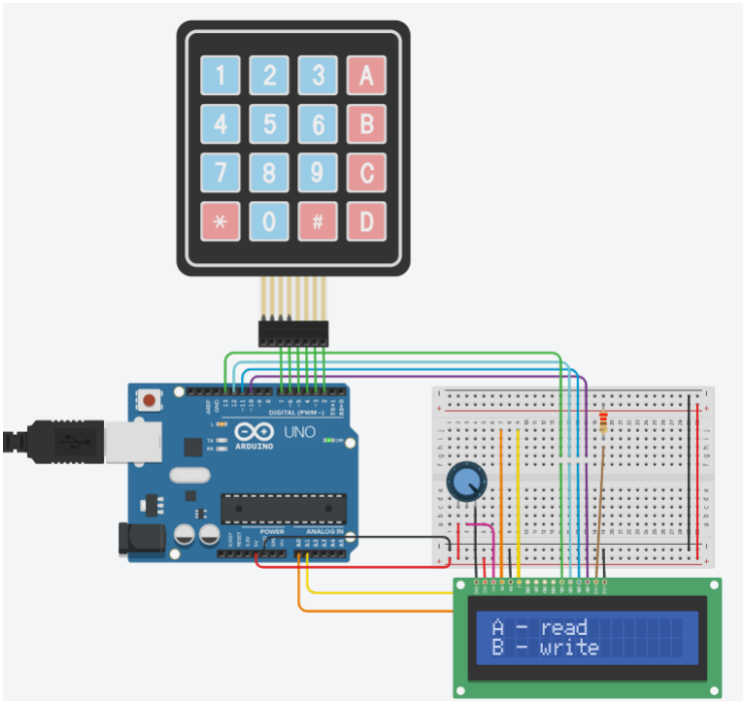
```ino
void loop()
{
  dist = 0.01723 * readUltrasonicDistance(12, 11);
  temp = (-40 + 0.488155 * (analogRead(A0) - 20));
  PR = digitalRead(7);
  gas = analogRead(A2);
  photoresistor = analogRead(A1);
  if (dist <= 100) {
    if ((-40 + 0.488155 * (analogRead(A0) - 20)) > 16) {
      digitalWrite(10, HIGH);
    } else {
      digitalWrite(10, LOW);
    }
    servo_8.write(90);
    if (photoresistor < 800) {
      if (dist < 100) {
        digitalWrite(9, HIGH);
      }
    } else {
      digitalWrite(9, LOW);
    }
  } else {
    digitalWrite(10, LOW);
    servo_8.write(0);
    digitalWrite(9, LOW);
  }
  if (PR == HIGH) {
    digitalWrite(6, LOW);
  } else {
    digitalWrite(6, HIGH);
  }
  if (photoresistor < 800) {
    digitalWrite(5, HIGH);
  } else {
    digitalWrite(5, LOW);
  }
  if (gas > 180) {
    if (temp > 45) {
      digitalWrite(13, HIGH);
      Serial.println("Fire Alarm");
    } else {
      digitalWrite(13, HIGH);
      Serial.println("Gass Leakage");
    }
  } else {
    digitalWrite(13, LOW);
  }
  delay(10); // Delay a little bit to improve simulation perform
}
```

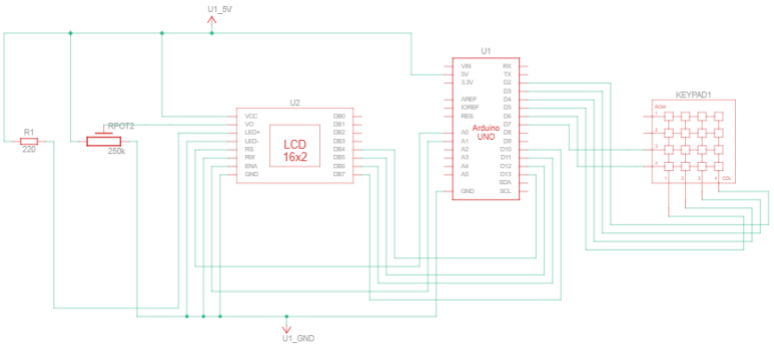




PRO 19:EEPROM

```ino
#include <EEPROM.h>
#include <LiquidCrystal.h>
#include <Keypad.h>

#define regselect A0
#define enable A1

String add, num;
const byte numRows = 4;
const byte numCols = 4;

char keymap[numRows][numCols] = {
    {'1', '2', '3', 'A'},
    {'4', '5', '6', 'B'},
    {'7', '8', '9', 'C'},
    {'*', '0', '#', 'D'}};

byte rowPins[numRows] = {9, 8, 7, 6};
byte colPins[numCols] = {5, 4, 3, 2};

LiquidCrystal lcd(regselect, enable, 13, 12, 11, 10);
Keypad myKeypad = Keypad(makeKeymap(keymap), rowPins, colPins,
                         numRows, numCols);


char temp;

void setup()
{
    Serial.begin(115200);
    lcd.begin(16, 2);
    lcd.setCursor(0, 0);
    lcd.print("A - read");
    lcd.setCursor(0, 1);
    lcd.print("B - write");
}

void loop()
{
    char keypressed = myKeypad.waitForKey();

    if (keypressed == 'A')
    {
        // read
        lcd.clear();
        lcd.print("Address: ");
        lcd.setCursor(0, 1);
        lcd.print("'#' to read");
        lcd.setCursor(9, 0);
        add = "";
        temp = ' ';
        while (true)
        {
            temp = myKeypad.waitForKey();
            if (temp == '#')
                break;
            add += temp;
            lcd.print(temp);
        }

        int _add = add.toInt();

        lcd.clear();
        if (_add < 0 || _add > 255)
            lcd.print("Only 0-255");
        else
        {
            lcd.print("Value: ");
            lcd.print(EEPROM.read(_add));
            lcd.setCursor(0, 1);
            lcd.print("'#' to confirm");
        }

        myKeypad.waitForKey();
        lcd.clear();
        lcd.print("A - read");
        lcd.setCursor(0, 1);
        lcd.print("B - write");
    }
    else if (keypressed == 'B')
    {
        // write
        lcd.clear();
        lcd.print("Address: ");
        lcd.setCursor(9, 0);

        add = "";
        num = "";
        temp = ' ';
        while (true)
        {
            temp = myKeypad.waitForKey();
            if (temp == '#')
                break;
            add += temp;
            lcd.print(temp);
            lcd.setCursor(1,1);
            lcd.print("'#' to select");
        }
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Value: ");

        temp = ' ';
        while (true)
        {
            temp = myKeypad.waitForKey();
            if (temp == '#')
                break;
            num += temp;
            lcd.print(temp);
            lcd.setCursor(1,1);
            lcd.print("'#' to write");
        }

        int _add = add.toInt();
        int _num = num.toInt();

        if (_add < 0 || _add > 255)
            lcd.print("Only 0-255");
        else
        {
            EEPROM.write(_add, _num);
        }

        lcd.clear();
        lcd.print("A - read");
        lcd.setCursor(0, 1);
        lcd.print("B - write");
    }
}
```





PRO 20: WIFI

```
#include <WiFi.h>
#include <WiFiClient.h>

const int DHTPin = 10;
const int LEDPin = 13;

const int MINHumidity = 25;

char ssid[] = "yourNetwork"; // Your network SSID (name)
char pass[] = "secretPassword"; // Your network WPA2 password
char server[] = "smtp.yourdomain.com"; // Your SMTP server

boolean firstEmail = true;

int status = WL_IDLE_STATUS;

WiFiClient client; // Set up the wireless client

void setup() {
  Serial.begin(9600);

  Serial.println("Plant monitor");

  // Configure the LED pin, set as output, high
  pinMode(LEDPin, OUTPUT);
  digitalWrite(LEDPin, HIGH);

  // Is there a WiFi shield installed?
```

```
 if (WiFi.status() == WL_NO_SHIELD) {
  Serial.println("ERR: WiFi shield not found");
  // No point continuing with the sketch
  while (true);
 }

 // Attempt to connect to the WiFi network
 while (status != WL_CONNECTED) {
  Serial.print("Attempting to connect to WPA SSID: ");
  Serial.println(ssid);
  // Connect to WPA/WPA2 network:
  status = WiFi.begin(ssid, pass);

  // Wait 10 seconds for connection:
  delay(10000);
 }

 // If we got here, then the connection is good. Set LED pin low and display information on serial
 digitalWrite(LEDPin, LOW);
 Serial.println("Connected!");
}

void loop() {
 // Get a humidity reading
 int val = getDht11Humidity();

 // Print it out to the serial port
 Serial.print("Current humidity: ");
 Serial.print(val);
```

```
  Serial.println("");

  if (val < MINHumidity) {
    // Below minimum humidity. Warn!
    Serial.println("Plant is thirsty!");
    sendEmail();
    firstEmail = false;
  }
  else {
    // All OK
    Serial.println("Humidity OK");
    firstEmail = true;
  }

  // Wait for half an hour
  delay(1800000);
}

int getDht11Humidity() {
  byte data[6] = {0};

  // Set up variables
  byte mask = 128;
  byte idx = 0;

  // Request a sample from the DHT11
  pinMode(DHTPin, OUTPUT);
  digitalWrite(DHTPin, LOW);
  delay(20);
```

```
digitalWrite(DHTPin, HIGH);
delayMicroseconds(40);
pinMode(DHTPin, INPUT);

// Will we get an ACK?
unsigned int loopCnt = 255;
while (digitalRead(DHTPin) == LOW) {
  if (--loopCnt == 0) return NAN;
}

loopCnt = 255;
while (digitalRead(DHTPin) == HIGH) {
  if (--loopCnt == 0) return NAN;
}

// Acknowledged, read in 40 bits
for (unsigned int i = 0; i < 40; i++) {
  // Pin will go low. Wait until it goes high
  loopCnt = 255;
  while (digitalRead(DHTPin) == LOW) {
    if (--loopCnt == 0) return NAN;
  }

  // What is the current time?
  unsigned long t = micros();

  // Pin will go high. Calculate how long it is high.
  loopCnt = 255;
  while (digitalRead(DHTPin) == HIGH) {
```

```
    if (--loopCnt == 0) return NAN;
   }

   // Is this a logical one, or a logical zero?
   if ((micros() - t) > 40) data[idx] |= mask;
   mask >>= 1;
   if (mask == 0) { // next byte?
    mask = 128;
    idx++;
   }
  }

  // Get the data, and return it
  float f = data[0];
  return (int)f;
}

boolean sendEmail() {
  // Attempt to connect
  if (!client.connect(server, 25))
   return false;

  // Change this to your IP
  client.write("helo 1.2.3.4\r\n");

  // change to your email address (sender)
  client.write("MAIL From: <plant@yourdomain.com>\r\n");

  // change to recipient address
```

```
client.write("RCPT To: <you@yourdomain.com>\r\n");

client.write("DATA\r\n");

// change to recipient address
client.write("To: You <you@yourdomain.com>\r\n");

// change to your address
client.write("From: Plant <plant@yourdomain.com>\r\n");

client.write("Subject: I need water!\r\n");

if (firstEmail == true) { // First email
  client.write("I'm thirsty!\r\n");
}
else {
  int i = random(4);
  if (i == 0)
    client.write("You don't love me any more, do you?\r\n");
  if (i == 1)
    client.write("All I know is pain...\r\n");
  if (i == 2)
    client.write("I would have watered you by now...\r\n");
  if (i == 3)
    client.write("My suffering will soon be over...\r\n");
}

client.write(".\r\n");
```

```
  client.write("QUIT\r\n");
  client.stop();

  return true;
}
```

PRO 21: ETHERNET

```
#include <SPI.h>
#include <Ethernet.h>

// Enter a MAC address and IP address for your controller below.
// The IP address will be dependent on your local network:
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,0,177);

int lightPin = A3;

// Initialize the Ethernet server to listen for connections on port 80
EthernetServer server(80);

void setup() {
  // Open serial communications
  Serial.begin(9600);

  // Start the Ethernet connection and the server
  Ethernet.begin(mac, ip);
  server.begin();
```

```
  Serial.print("Server up on ");
  Serial.println(Ethernet.localIP());
}

void loop() {
  // Listen for incoming clients
  EthernetClient client = server.available();

  if (client) {
    Serial.println("New connection");

    // An HTTP request ends with a blank line, wait until the request has finished
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.write(c);

        // Check for the end of the HTTP request
        if (c == '\n' && currentLineIsBlank) {
          // Send a standard HTTP response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");
          client.println("Refresh: 5");
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html>");
```

```
        // Get a light level reading
        int light = analogRead(lightPin);

        // Send light level data as a webpage
        client.print("Current light level is ");
        client.print(light);
        client.println("<br />");

        client.println("</html>");
        break;
      }
      if (c == '\n') {
        // Start of a new line
        currentLineIsBlank = true;
      }
      else if (c != '\r') {
        // Character on the current line
        currentLineIsBlank = false;
      }
    }
  }

  // Wait a bit for the client to receive data
  delay(1);

  // Close the connection
  client.stop();
  Serial.println("Client disconnected");
}}
```

PRO 22: I2C

```
#include <Wire.h>

#define SLAVE_ADDRESS 0x08

int data = 0;

int state = 0;


void setup() {

  pinMode(13, OUTPUT); // Internal LED

  Serial.begin(9600);

  Wire.begin(SLAVE_ADDRESS); // Initialize as I2C slave


  // Register I2C callbacks

  Wire.onReceive(receiveData);

  Wire.onRequest(sendData);

}


void loop() {

  // Nothing to do

  delay(100);

}


// Callback for data reception

void receiveData(int byteCount) {

  while(Wire.available()) {

    data = Wire.read();

    Serial.print("Data received: ");

    Serial.println(data);


    if (data == 1) {
```

```
    digitalWrite(13, HIGH); // Turn the LED on
    state = 1;
   } else {
    digitalWrite(13, LOW); // Turn the LED off
    state = 0;
   }
 }
}

// Callback for sending data
void sendData() {
 Wire.write(state); // Send the LED state
}
```

PRO 23:SPI

```
#include <SPI.h>
const int slaveSelectPin = 10;
void setup() {
 Serial.begin(9600);
 // Initialize the bus for a device on pin 10
 SPI.begin();
}

void loop() {
 // Read in 4 bytes of data
 byte data1 = SPI.transfer(0);
 byte data2 = SPI.transfer(0);
 byte data3 = SPI.transfer(0);
 byte data4 = SPI.transfer(0); // Stop
```

```
// Create two 16-bit variables
word temp1 = word(data1, data2);
word temp2 = word(data3, data4);

// Is the reading negative?
bool neg = false;
if (temp1 & 0x8000) {
  neg = true;
}

// Is the MAX31855 reporting an error?
if (temp1 & 0x1) {
  Serial.println("Thermocouple error!");
  if (temp2 & 0x1)
    Serial.println("Open circuit");
  if (temp2 & 0x2)
    Serial.println("VCC Short");
  if (temp2 & 0x4)
    Serial.println("GND short");
}

// Keep only the bits that interest us
temp1 &= 0x7FFC;

// Shift the data
temp1 >>= 2;

// Create a celsius variable, the value of the thermocouple temp
```

```
  double celsius = temp1;

  // The thermocouple returns values in 0.25 degrees Celsius
  celsius *= 0.25;
  if (neg == true)
    celsius *= -1;

  // Now print out the data
  Serial.print("Temperature: ");
  Serial.print(celsius);
  Serial.println();

  // Sleep for two seconds
  delay(2000);
}
```