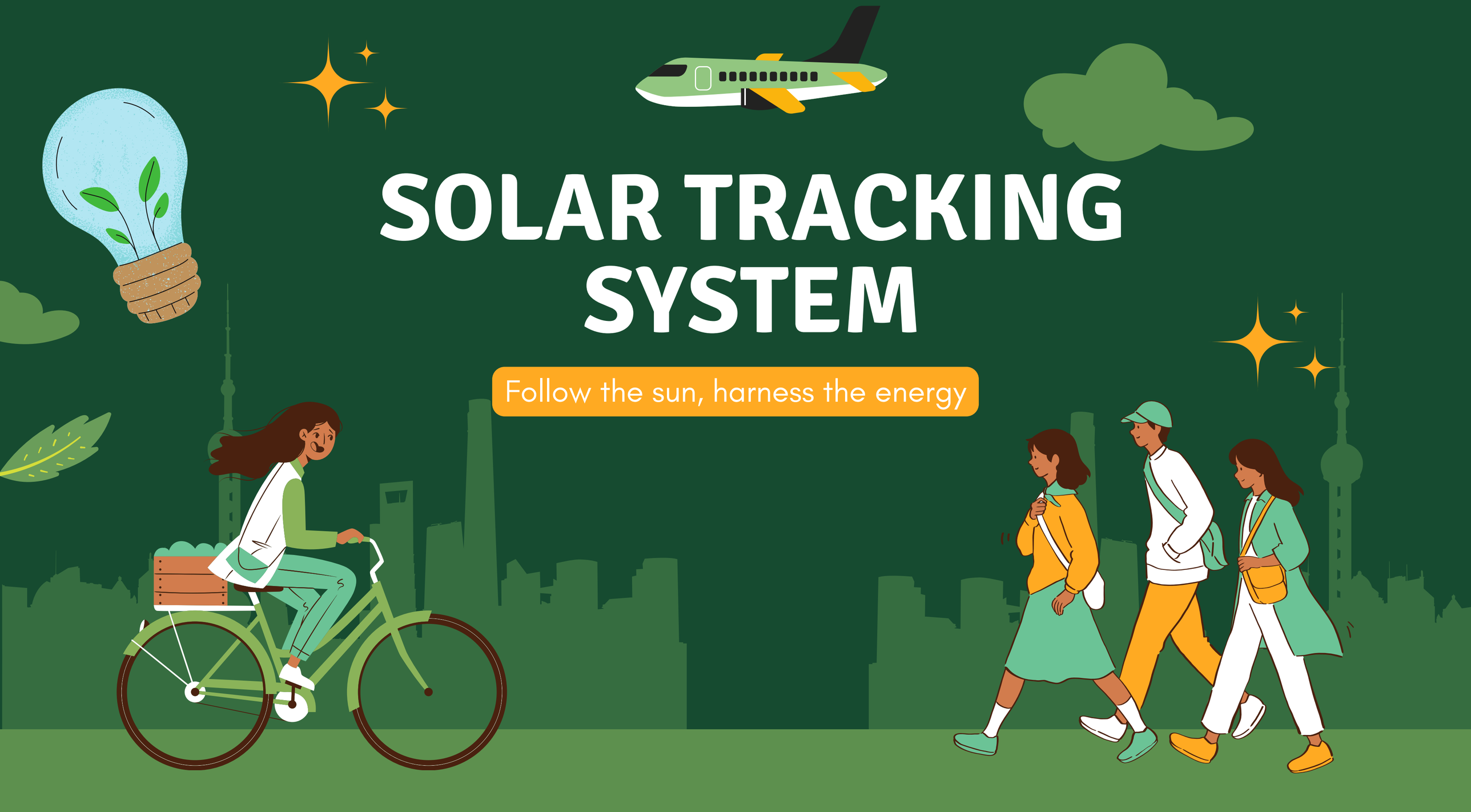
SOLAR TRACKING SYSTEM

Follow the sun, harness the energy



PROBLEM STATEMENT

Farmers in agricultural regions often struggle with fixed solar panel installations that limit energy efficiency. This project aims to develop a solar tracking system that automatically adjusts the position of solar panels to follow the sun's movement, maximizing sunlight capture and energy production. The system will utilize LDRs and a temperature sensor to optimize panel orientation, providing a reliable renewable energy source for irrigation pumps and reducing operational costs.

LIST OF MODULES

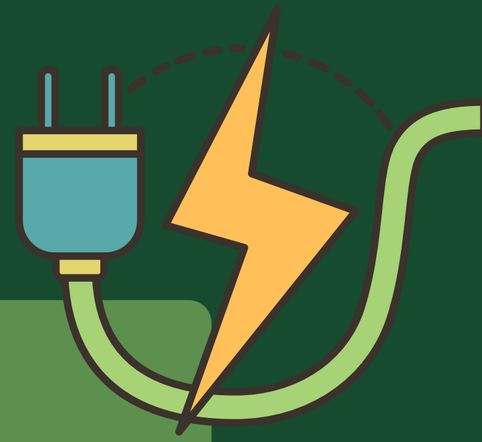
- Setup and Initialization
- Main Loop Logic
- Sensor Functions
- Night Mode Control
- Servo Control and Data Display

LIST OF COMPONENTS

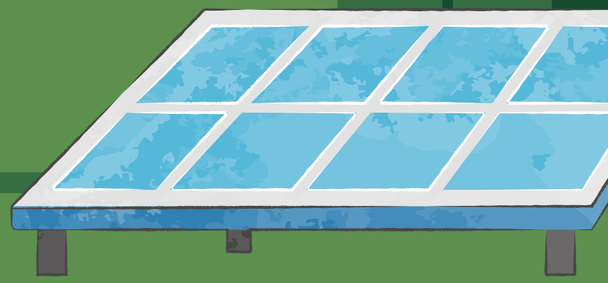
Name	Quantity	Component
U1	1	Arduino Uno R3
R1 R2	2	Photoresistor
SERVO1	1	Positional Micro Servo
R3 R4	2	10 kΩ Resistor
SC1	1	5 V, 100 mA Solar Cell
U2	1	Temperature Sensor [TMP36]
U3	1	PCF8574-based, 39 (0x27) LCD 16 x 2 (I2C)

SETUP AND INITIALIZATION

UHASHINI N (CB.SC.U4CSE23352)

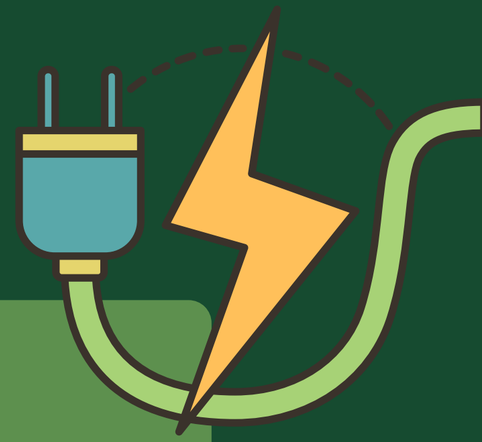


```
#include <Wire.h>                  // Library for I2C communication
#include <LiquidCrystal_I2C.h>     // Library for I2C LCD display
#include <Servo.h>                 // Library to control servo motors

Servo sg90;
int initial_position = 90;
int LDR1 = A0;
int LDR2 = A1;
int tempSensorPin = A2;
int servopin = 9;
int nightModeThreshold = 200;
int maxLightDifference = 200;
bool isNightModeActive = false;

LiquidCrystal_I2C lcd(0x27, 16, 2);
```

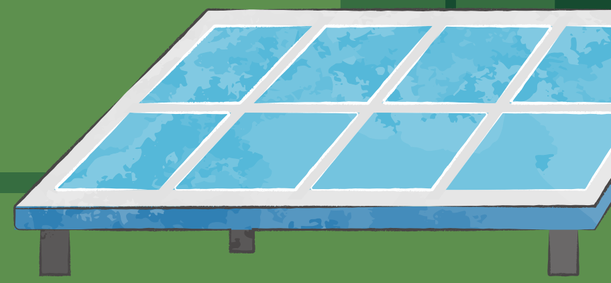


SETUP AND INITIALIZATION

UHASHINI N (CB.SC.U4CSE23352)



```
void setup() {
  sg90.attach(servopin);            // Attach the servo motor to the specified pin
  sg90.write(initial_position);     // Set the servo to its initial position (90 degrees)
  pinMode(LDR1, INPUT);             // Set LDR1 pin as an input
  pinMode(LDR2, INPUT);              // Set LDR2 pin as an input
  Serial.begin(9600);               // Start serial communication for debugging purposes

  lcd.init();                      // Initialize the LCD
  lcd.backlight();                  // Turn on the LCD backlight
  lcd.setCursor(0, 0);               // Set the cursor to the first row, first column
  lcd.print("Initializing...");     // Display an initial message on the LCD
  delay(2000);                      // Wait for 2 seconds
  lcd.clear();                     // Clear the screen to get ready for main loop display
}
```



MAIN LOOP LOGIC

SHREYA B (CB.SC.U4CSE23347)

```
void loop() {
  int R1 = analogRead(LDR1);       // Read the analog value from LDR1
  int R2 = analogRead(LDR2);       // Read the analog value from LDR2
  float temperature = readTemperature();  // Read temperature from sensor
  int diff = R1 - R2;              // Calculate the difference between the two LDR readings
  if (isNightMode(R1, R2)) {       // Determine if the system should activate night mode
    activateNightMode();           // If true, activate night mode
  } else {
    deactivateNightMode();         // Otherwise, ensure night mode is deactivated
    controlServo(diff);            // Control the servo based on the LDR difference
  }
  if (!isNightModeActive) {        // Display info only when not in night mode
    displayInfo(diff, temperature);   // Display the current information on the
  }
  plotData(R1, R2, temperature, initial_position); // Output data to Serial
  delay(500);                      // Add a delay before the next iteration
}
```

SENSOR FUNCTIONS

SWASTHIKA P S (CB.SC.U4CSE23350)

```
float readTemperature() {
  float voltage = analogRead(tempSensorPin) * (5.0 / 1023.0);
  // Convert analog reading to voltage
  float temperatureC = voltage * 100.0;
  // Convert voltage to temperature (assuming LM35 sensor)
  return temperatureC;
}

bool isNightMode(int R1, int R2) {
  return (R1 < nightModeThreshold && R2 < nightModeThreshold);
}
```

NIGHT MODE CONTROL

KANISHKA S (CB.SC.U4CSE23328)

```
bool isNightMode(int R1, int R2) {
  // Check if both LDRs report a value below the threshold, indicating low light conditions
  return (R1 < nightModeThreshold && R2 < nightModeThreshold);
}
void activateNightMode() {
  if (!isNightModeActive) {          // Only update if night mode status has changed
    lcd.clear();                     // Clear the LCD screen
    lcd.setCursor(0, 0);
    lcd.print("Night Mode ON ");     // Display "Night Mode ON" on the LCD
    sg90.detach();                   // Detach the servo motor to save power
    isNightModeActive = true;        // Update the status
    Serial.println("Night Mode Activated"); // Log message to Serial Monitor
  }
}
void deactivateNightMode() {
  if (isNightModeActive) {           // Only update if night mode status has changed
    lcd.clear();                     // Clear the LCD screen
    sg90.attach(servopin);           // Re-attach the servo motor when exiting night mode
    sg90.write(initial_position);    // Set servo back to initial position
    isNightModeActive = false;       // Update the status
    Serial.println("Night Mode Deactivated"); // Log message to Serial Monitor
  }  }
```

SERVO CONTROL AND DATA DISPLAY

ANAND PRIYADARSHINI (CB.SC.U4CSE23306)

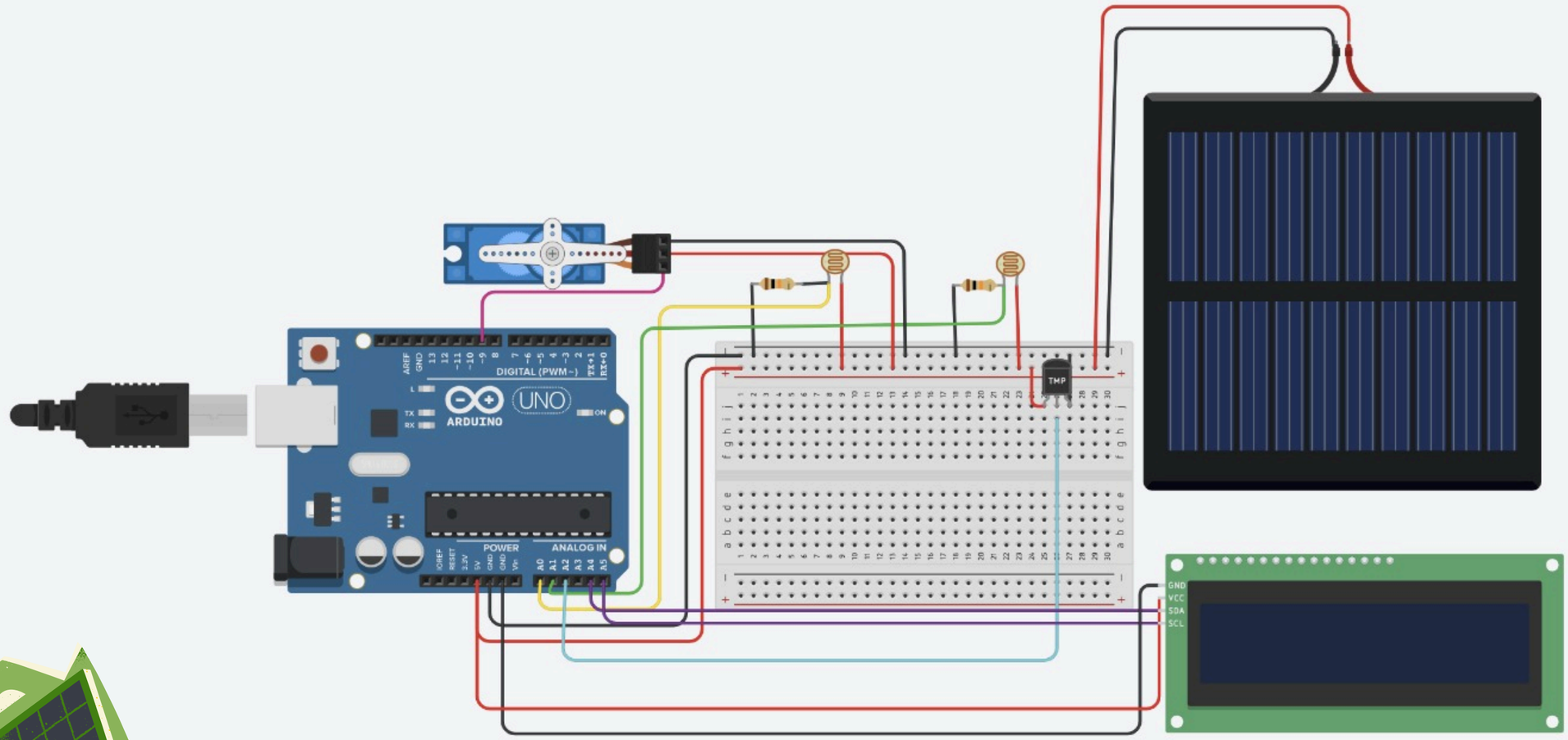
```
void controlServo(int diff) {
  int errorMargin = map(abs(diff), 0, maxLightDifference, 10, 2); // Map error margin
  if (abs(diff) > errorMargin) {   // If the difference is significant, adjust the servo
    int adjustment = map(diff, -maxLightDifference, maxLightDifference, -10, 10); // Calculate adjustment
    initial_position = constrain(initial_position + adjustment, 0, 180); // Ensure position stays within bounds
    sg90.write(initial_position);  // Move the servo to the new position
    delay(100); // Small delay to allow smooth movement
  }
}
void displayInfo(int diff, float temperature) {
  lcd.setCursor(0, 0);
  lcd.print("Angle: ");
  lcd.print(initial_position);
  lcd.print("   ");
  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print(" C ");
}
```

SERVO CONTROL AND DATA DISPLAY

ANAND PRIYADARSHINI (CB.SC.U4CSE23306)

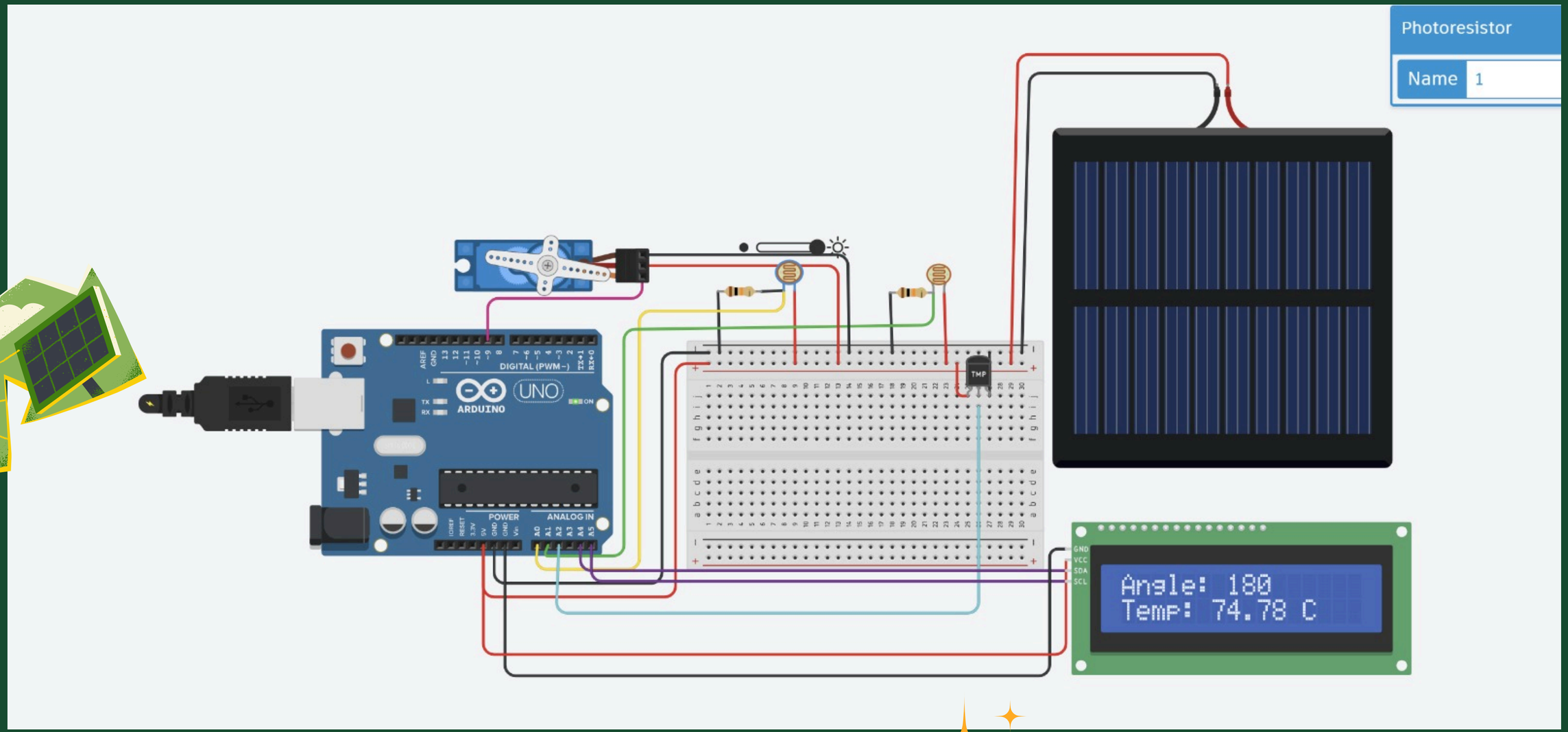
```
void plotData(int R1, int R2, float temperature, int angle) {
  Serial.print("LDR1:");
  Serial.print(R1);
  Serial.print("\tLDR2:");
  Serial.print(R2);
  Serial.print("\tTemperature:");
  Serial.print(temperature);
  Serial.print("\tAngle:");
  Serial.println(angle);
}
```

CONNECTION DIAGRAM IN TINKERCAD

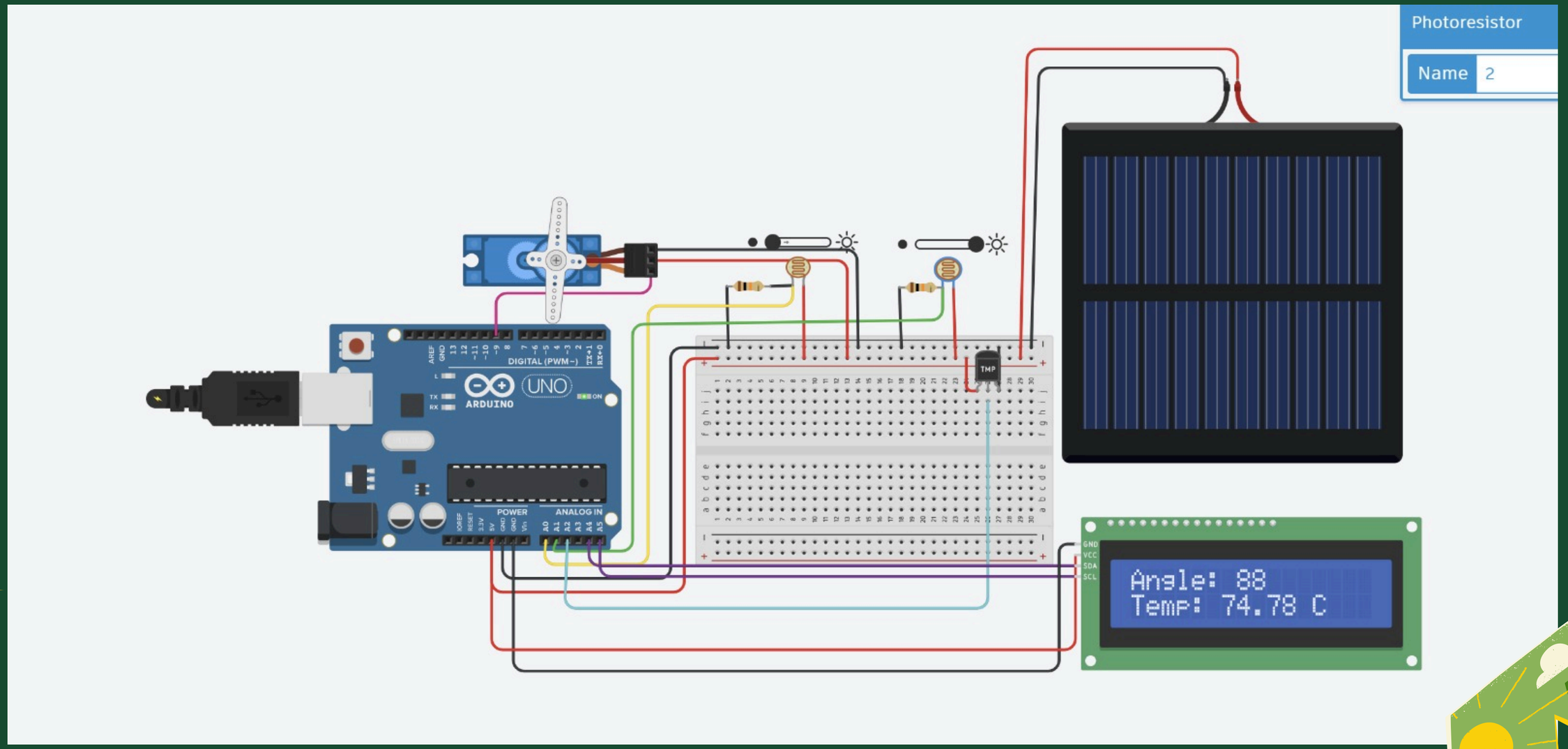


TINKERCAD SIMULATION DONE BY UHASHINI N

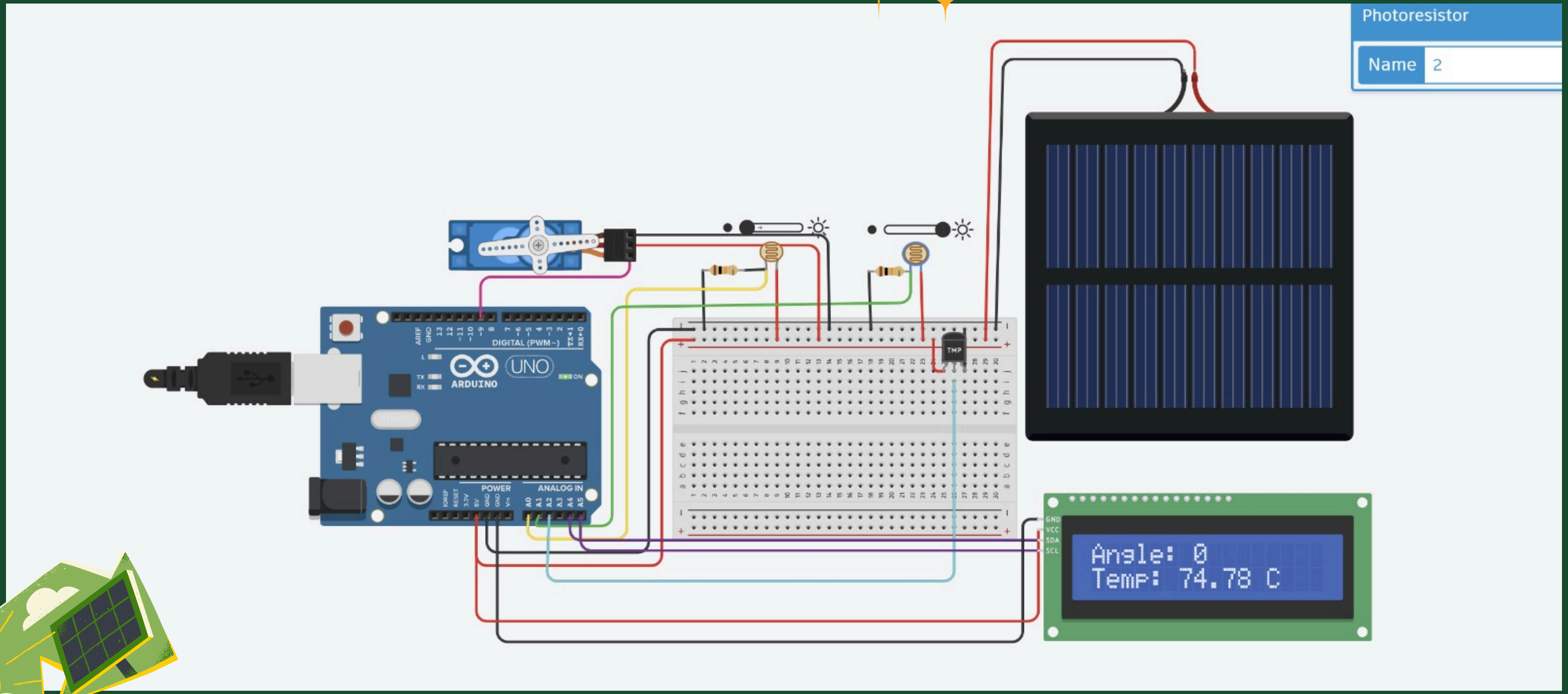
AT ANGLE 180


Photoresistor
Name 1
DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
TMP
GND
VCC
SDA
SCL
Angle: 180
Temp: 74.78 C

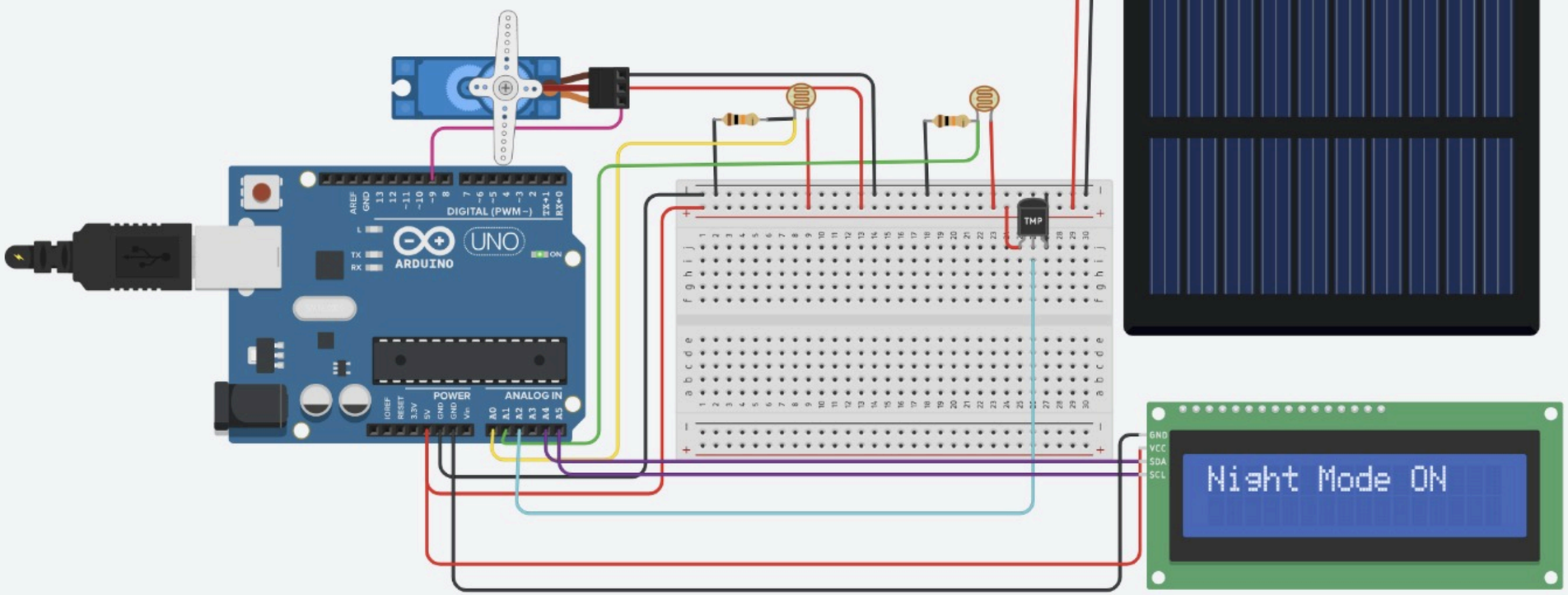
AT ANGLE 88


Photoresistor
Name 2
DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
TMP
GND
VCC
SDA
SCL
Angle: 88
Temp: 74.78 C

AT ANGLE 0


Photoresistor
Name 2
DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
TMP
GND
VCC
SDA
SCL
Angle: 0
Temp: 74.78 C

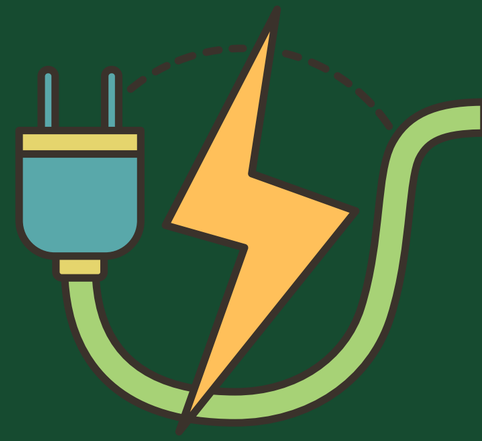
NIGHT MODE ON



```
#include <stdio.h>
#include <stdbool.h>
#include <stdlib.h>
// Definitions for pin numbers and constants
#define LDR1_PIN 0          // A0 pin for LDR1
#define LDR2_PIN 1          // A1 pin for LDR2
#define TEMP_SENSOR_PIN 2   // A2 pin for temperature sensor
#define SERVO_PIN 9         // PWM pin for servo motor
#define NIGHT_MODE_THRESHOLD 200
#define MAX_LIGHT_DIFFERENCE 200
// Variables
int initial_position = 90;
bool isNightModeActive = false;
// Simulated LCD and Servo functions (these would need actual libraries on embedded
hardware)
void lcd_init() {
    printf("LCD Initialized\n");
}
void lcd_backlight_on() {
    printf("LCD Backlight ON\n");
}
void lcd_set_cursor(int row, int col) {
    printf("LCD Cursor set to row %d, col %d\n", row, col);
}
```

```
void lcd_print(const char* message) {
    printf("LCD Display: %s\n", message);
}
void lcd_clear() {
    printf("LCD Cleared\n");
}
void servo_attach(int pin) {
    printf("Servo attached to pin %d\n", pin);
}
void servo_write(int position) {
    printf("Servo moved to %d degrees\n", position);
}
void setup() {
    // Servo setup
    servo_attach(SERVO_PIN);
    servo_write(initial_position);
    printf("LDR1 (A0) and LDR2 (A1) set as INPUT\n");
    printf("Serial Communication started at 9600 baud\n");
    lcd_init();
    lcd_backlight_on();
    lcd_set_cursor(0, 0);
    lcd_print("Initializing...");
    printf("Delay for 2 seconds\n");
    lcd_clear();
}
```

```
void loop() {
  int lightLevel1, lightLevel2, tempValue;
  // Prompt user for input
  printf("Enter LDR1 light level (0-1023): ");
  scanf("%d", &lightLevel1);
  printf("Enter LDR2 light level (0-1023): ");
  scanf("%d", &lightLevel2);
  printf("Enter temperature sensor value (0-1023): ");
  scanf("%d", &tempValue);
  // Calculate light difference and temperature in Celsius
  int lightDifference = abs(lightLevel1 - lightLevel2);
  float temperature = (tempValue / 1024.0) * 500.0;
  int servoPosition;
  lcd_set_cursor(0, 0);
  printf("Temp: %.1f C\n", temperature);
  if (lightLevel1 < NIGHT_MODE_THRESHOLD && lightLevel2 < NIGHT_MODE_THRESHOLD) {
    isNightModeActive = true;
    lcd_set_cursor(0, 1);
    lcd_print("Night Mode Active ");
    servoPosition = 90; // Neutral position in night mode
  } else {
    isNightModeActive = false;
    lcd_set_cursor(0, 1);
    lcd_print("Adjusting Servo   ");
```



```
    if (lightDifference > MAX_LIGHT_DIFFERENCE) {
      // Determine servo position based on which LDR has more light
      if (lightLevel1 > lightLevel2) {
        servoPosition = 0;  // Turn servo fully left
      } else {
        servoPosition = 180; // Turn servo fully right }
    } else {
      // Calculate servo angle based on LDR difference
      servoPosition = 90 + (lightDifference * 90 / MAX_LIGHT_DIFFERENCE) * (lightLevel1 >
lightLevel2 ? -1 : 1);
      if (servoPosition < 0) servoPosition = 0;
      if (servoPosition > 180) servoPosition = 180;
    }
  }
  // Move servo to calculated position and print angle
  servo_write(servoPosition);
  printf("Servo Position set to %d degrees based on LDR inputs\n", servoPosition);
  printf("LDR1: %d | LDR2: %d | Temperature: %.1f C | Servo Position: %d\n",
      lightLevel1, lightLevel2, temperature, servoPosition);
    char cont;
  printf("Do you want to continue? (y/n): ");
  scanf(" %c", &cont);
  if (cont == 'n' || cont == 'N') {
    printf("Stopping program.\n");
    exit(0); // Stop the program
  }
```

```
  // Simulate delay
   printf("Delay for 1 second\n");
}

int main() {
   setup();

   while (1) {
      loop();
   }

   return 0;
}
```


SAVE OUR PLANET!
ACT NOW
THE FUTURE OF ENERGY IS IN OUR HANDS; LET'S TURN TO THE SUN WITH INNOVATION AND DETERMINATION!

THANK YOU

ANAND PRIYADARSHINI (CB.SC.U4CSE23306)
KANISHKA S (CB.SC.U4CSE23328)
SHREYA B (CB.SC.U4CSE23347)
SWASTHIKA P S (CB.SC.U4CSE23350)
UHASHINI N (CB.SC.U4CSE23352)