ECET 430 FINAL PRESENTATION: SMART PLANT CARE PROJECT

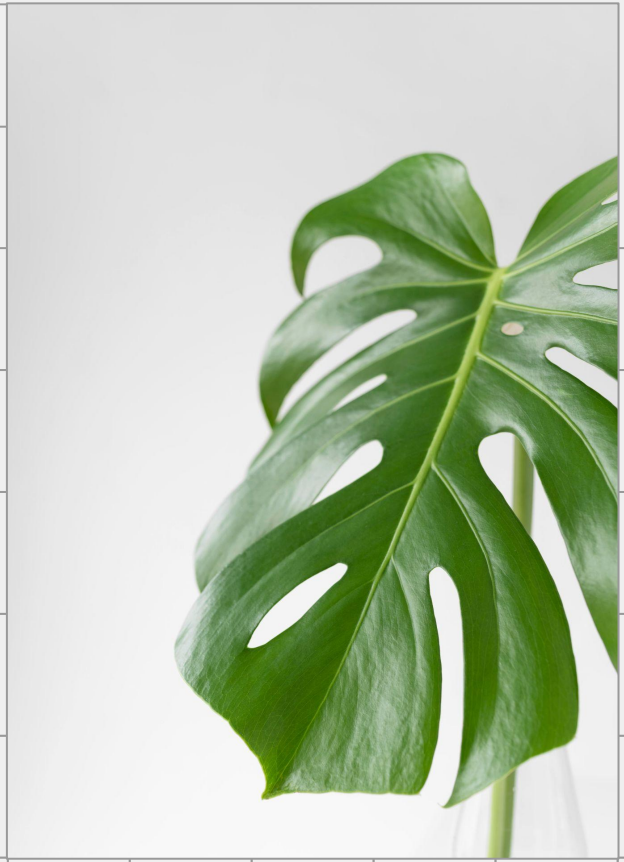
Natwarin Padtha, Jonathan Jou

INTRODUCTION & OVERVIEW OF THE PROJECT

The Smart Plant Care project is designed to help indoor plant owners monitor and maintain optimal growing conditions through real-time environmental data collection. This device integrates sensors to measure soil moisture, temperature, humidity, and light intensity, providing users with accurate insights into their plant's health.

EXPECTED OUTCOME

- Accurately measures soil moisture, temperature, humidity, and light intensity.
- Alerts users when watering is needed or conditions are extreme.
- Simple to use, making plant care easier for both beginners and experienced gardeners.



CONTRIBUTORS

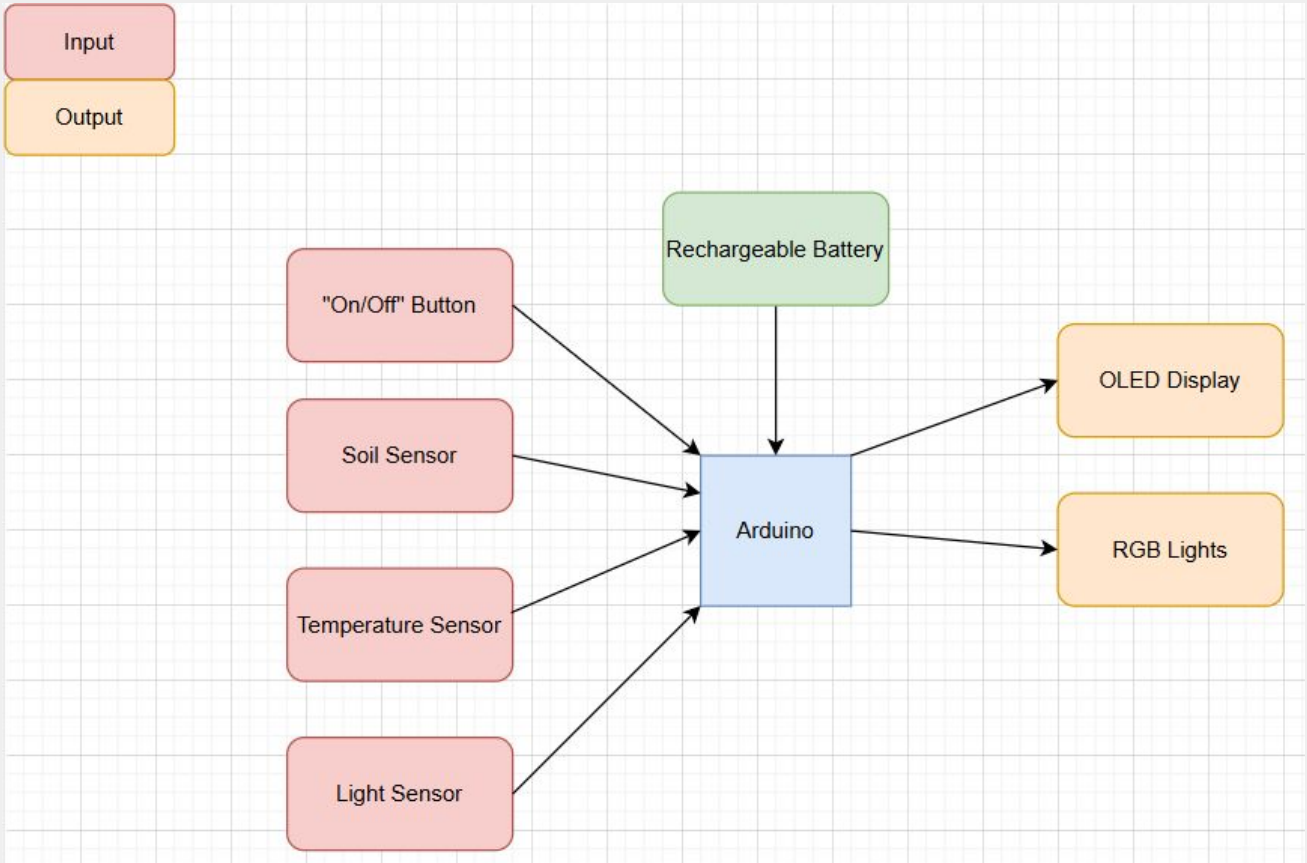
NATWARIN PADTHA

-Looks-Like Prototype
-Engineering Prototype
-KiCAD Schematics
-KiCAD PCB

JONATHAN JOU

-Works-Like Prototype
-Engineering Prototype
-Arduino Code
-Circuit

BLOCK DIAGRAM



ELECTRONIC COMPONENTS

- Microcontroller: Arduino UNO
- Temperature/Humidity Sensor: DHT11 Temperature and Humidity Module
- Soil Moisture Sensor: LM393 3.3V-5V Soil Moisture Sensor
- Light Sensor: APDS-9306 Digital LDR Photosensitive Light Sensor
- RGB lights
- Power Supply: 3.7V Li-ion 14500 battery
- Voltage regulator
- OLED Display
- Breadboard
- Wire
- Resistors

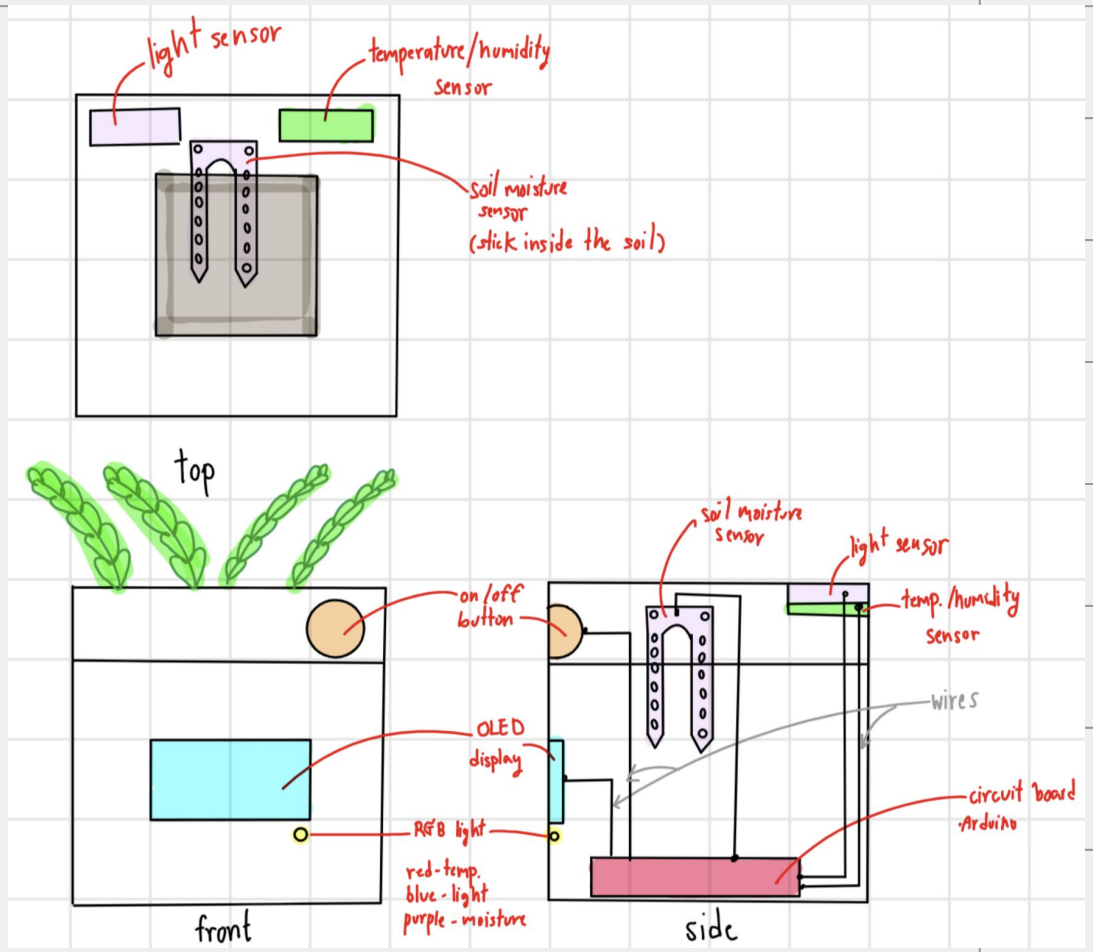
PROOF OF CONCEPT (POC): LOOKS-LIKE PROTOTYPE

Purpose:

- Demonstrates the physical appearance, size, and user interface layout.
- Focuses on aesthetics, ergonomics, and accessibility.

Looks-like prototype ideas

- The light sensor, soil moisture sensor, temperature sensor, display, and button will be placed on the outer surface for easy access.
- The electronics will be housed inside the vase for protection and a clean look.
- The prototype will be 3D-printed with plastic (PLA/ABS) for easy testing and changes.
- Size: 6" x 6" x 6"



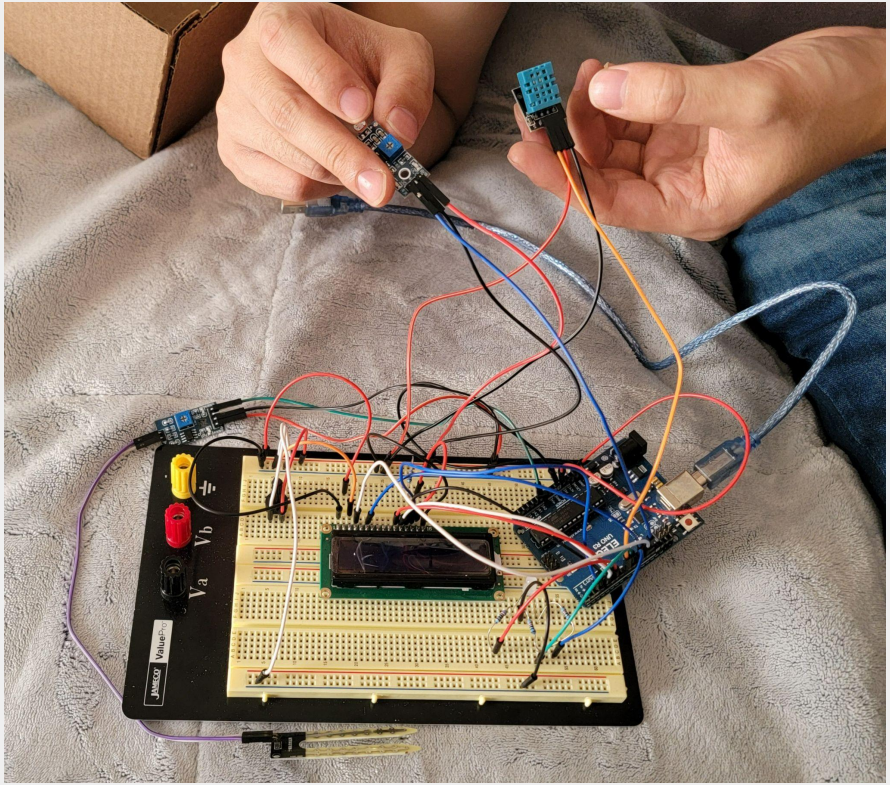
PROOF OF CONCEPT (POC): WORKS-LIKE PROTOTYPE

Purpose:

- Tests the core functionality and electronic components without focusing on appearance.
- Ensures sensors, power management, and display work as expected.

Works-Like Prototype:

- Microcontroller: Arduino Uno
- Sensors:
  -Soil moisture sensor to detect water levels
  -Temperature & humidity sensor for environmental conditions
  -Light sensor to measure light exposure
- Power Source: Li-ion or LiPo battery with a voltage regulator
- Output:
  -OLED display to show plant status
  -Button for user interaction



PROOF OF CONCEPT (POC): ENGINEERING PROTOTYPE

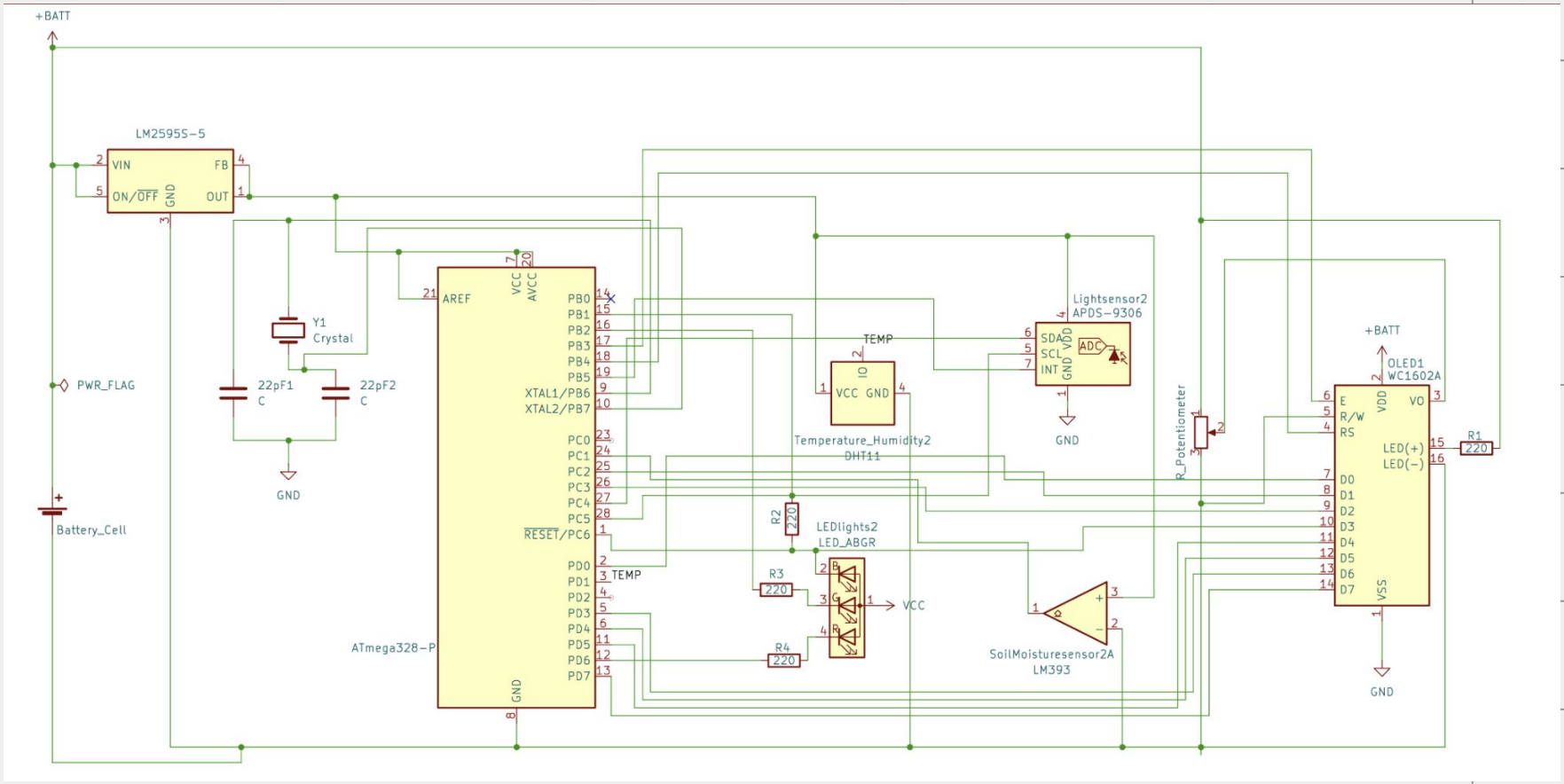
Purpose:

- Combines both the looks-like and works-like prototypes into a final integrated system.
- Ensures durability, reliability, and manufacturability.

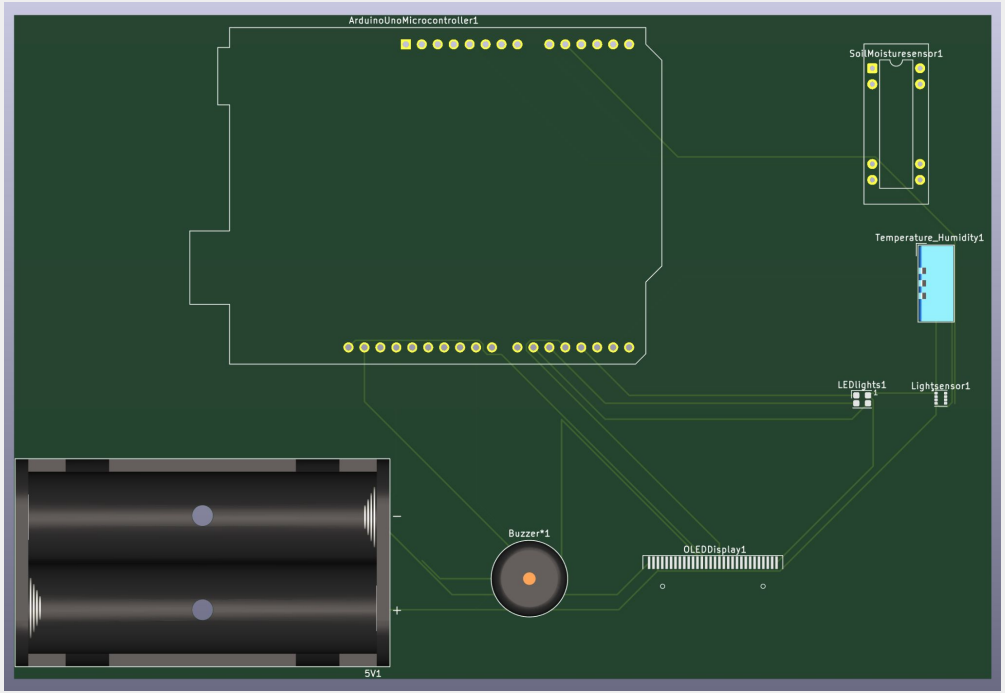
Design & Features:

- Fully assembled electronics housed inside the 3D-printed vase.
- Optimized circuit design using KiCad for PCB development.
- Sealed and durable casing to protect electronics from moisture.
- Testing and debugging to ensure all functions work correctly.

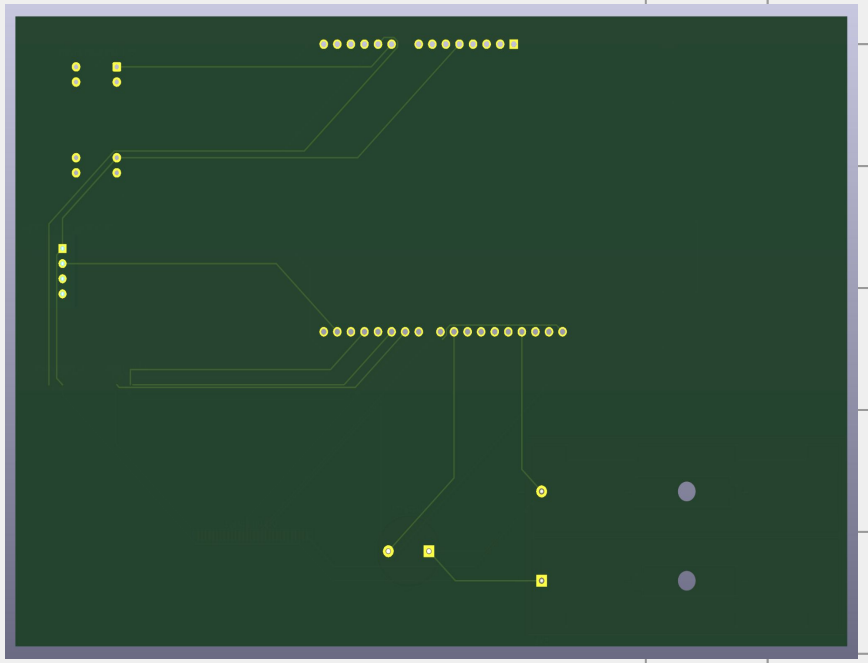
KICAD SCHEMATICS (WHOLE PROJECT)



KICAD PCB DESIGN (MIDTERM)

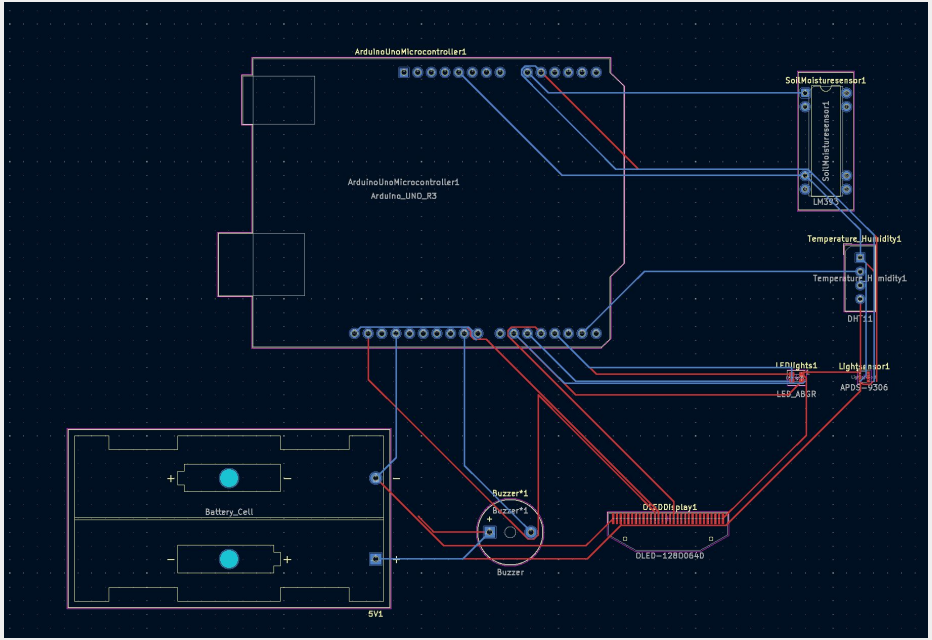


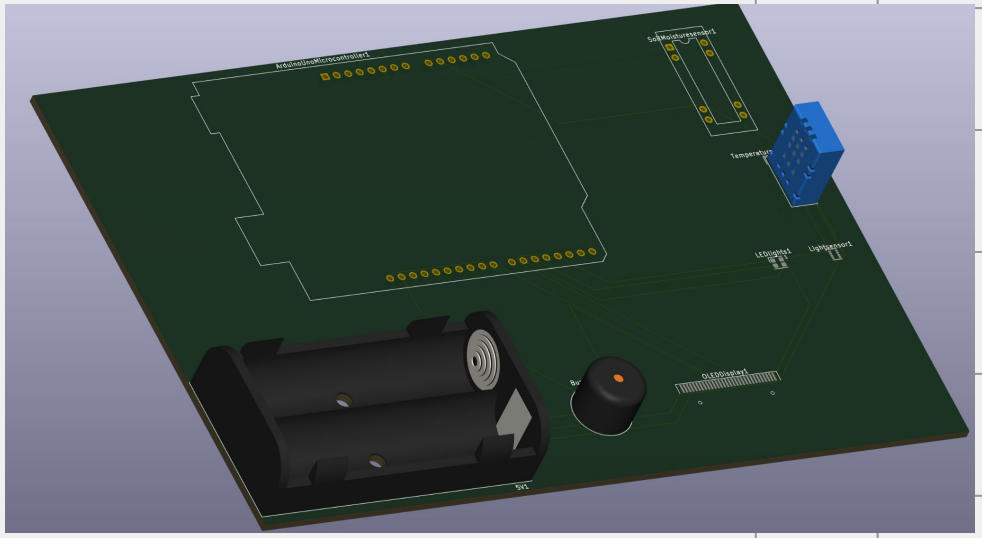
front



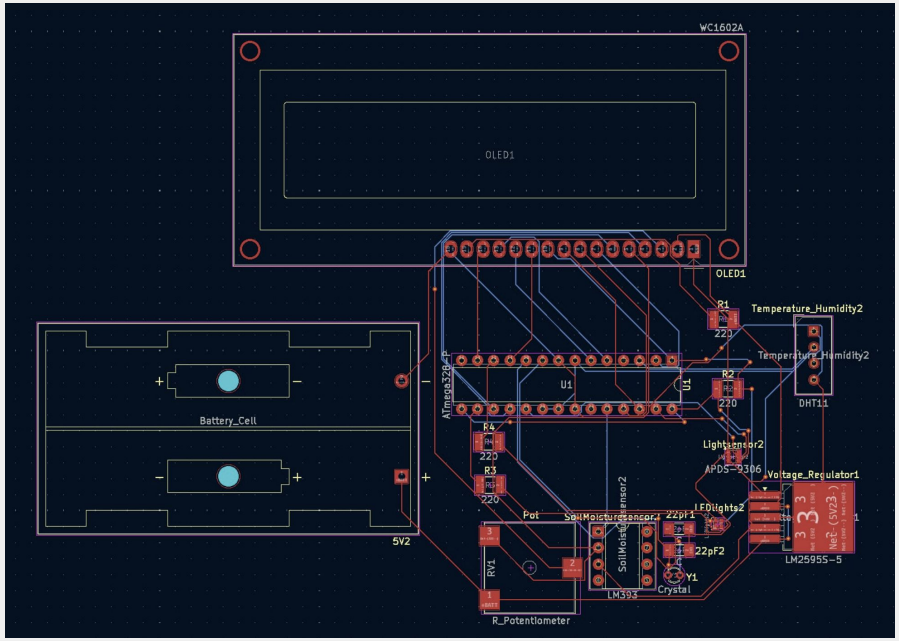
back

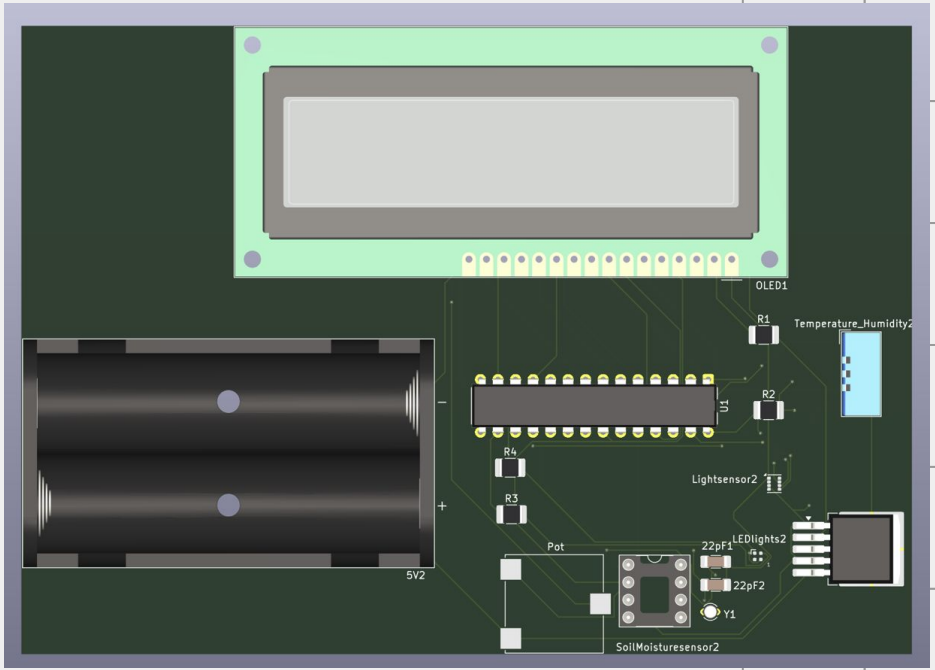
KICAD PCB DESIGN (MIDTERM)





KICAD PCB DESIGN (FINAL)





ARDUINO CODE

```
#include <dht.h>
#include <LiquidCrystal.h>

// Sensor and LED Pins
#define outPin 8        // DHT11 sensor
#define DO_PIN 13       // Light sensor (digital)
int sensorPin = A0;     // Moisture sensor (analog)

// RGB LED Pins
const int redPin = 6;
const int greenPin = 9;
const int bluePin = 10;

// LCD Pins: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Temperature thresholds (in Fahrenheit)
const float minTempThresholdF = 70.0;
const float maxTempThresholdF = 80.0;

dht DHT;  // DHT11 sensor object

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);
  lcd.print("System Starting...");
  delay(2000);
  lcd.clear();

  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  pinMode(DO_PIN, INPUT);
  pinMode(sensorPin, INPUT);
}

void loop() {
  int readData = DHT.read11(outPin);
  float temperatureF = (DHT.temperature * 9.0) / 5.0 + 32.0;
  float humidity = DHT.humidity;
  bool lightGood = checkLightSensor();
  int moistureStatus = checkMoistureSensor();

  bool temperatureGood = (temperatureF >= minTempThresholdF &&
temperatureF <= maxTempThresholdF);
  bool moistureGood = (moistureStatus == 0);
  bool allGood = temperatureGood && lightGood && moistureGood;

  // Display sensor values
  Serial.println("============================");
  Serial.print("Temp = ");
  Serial.print(DHT.temperature);
  Serial.print("C | ");
  Serial.print(temperatureF);
  Serial.println("F");
  Serial.print("Humidity = ");
  Serial.print(humidity);
  Serial.println("%");
  Serial.println(lightGood ? "Light Present" : "No Light Detected");
  Serial.println(moistureStatus == -1 ? "Soil is Dry" : (moistureStatu
== 1 ? "Soil is Too Wet" : "Soil Moisture is Good"));
  Serial.println("============================");

  // LCD first row = Temp & Humidity
  lcd.clear();
```

ARDUINO CODE

```
  lcd.setCursor(0, 0);
  lcd.print("T:");
  lcd.print(temperatureF, 1);
  lcd.print("F H:");
  lcd.print(humidity, 0);
  lcd.print("%");

  if (allGood) {
    setGreen();
    lcd.setCursor(0, 1);
    lcd.print("All Good");
  } else {
    if (!temperatureGood) {
      flashRed();
    }
    if (!lightGood) {
      flashBlue();
    }
    if (!moistureGood && moistureStatus == -1) {
      flashPurple();
    }
  }

  delay(2000);
}
```

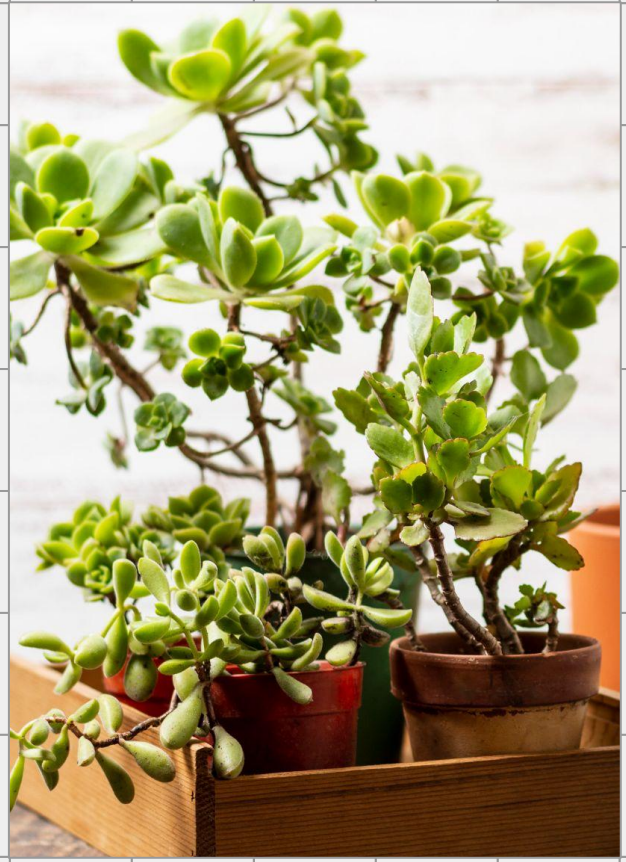
```
void flashRed() {
  lcd.setCursor(0, 1);
  lcd.print("Temp Low");
  for (int i = 0; i < 3; i++) {
    setRGB(HIGH, LOW, LOW);  // Red
    delay(300);
    setRGB(LOW, LOW, LOW);
    delay(300);
  }
}

void flashBlue() {
  lcd.setCursor(0, 1);
  lcd.print("No Light       ");  // Pad with spaces to clear previous
msg
  for (int i = 0; i < 3; i++) {
    setRGB(LOW, LOW, HIGH);  // Blue
    delay(300);
    setRGB(LOW, LOW, LOW);
    delay(300);
  }
}
```

```
bool checkLightSensor() {
  int lightState = digitalRead(DO_PIN);
  return (lightState == LOW);  // LOW = light detected
}

int checkMoistureSensor() {
  int sensorValue = analogRead(sensorPin);
  sensorValue = map(sensorValue, 0, 1024, 255, 0);
  if (sensorValue < 150) return -1;  // Dry
  else if (sensorValue > 200) return 1;  // Too wet
  else return 0;  // Good
}
```

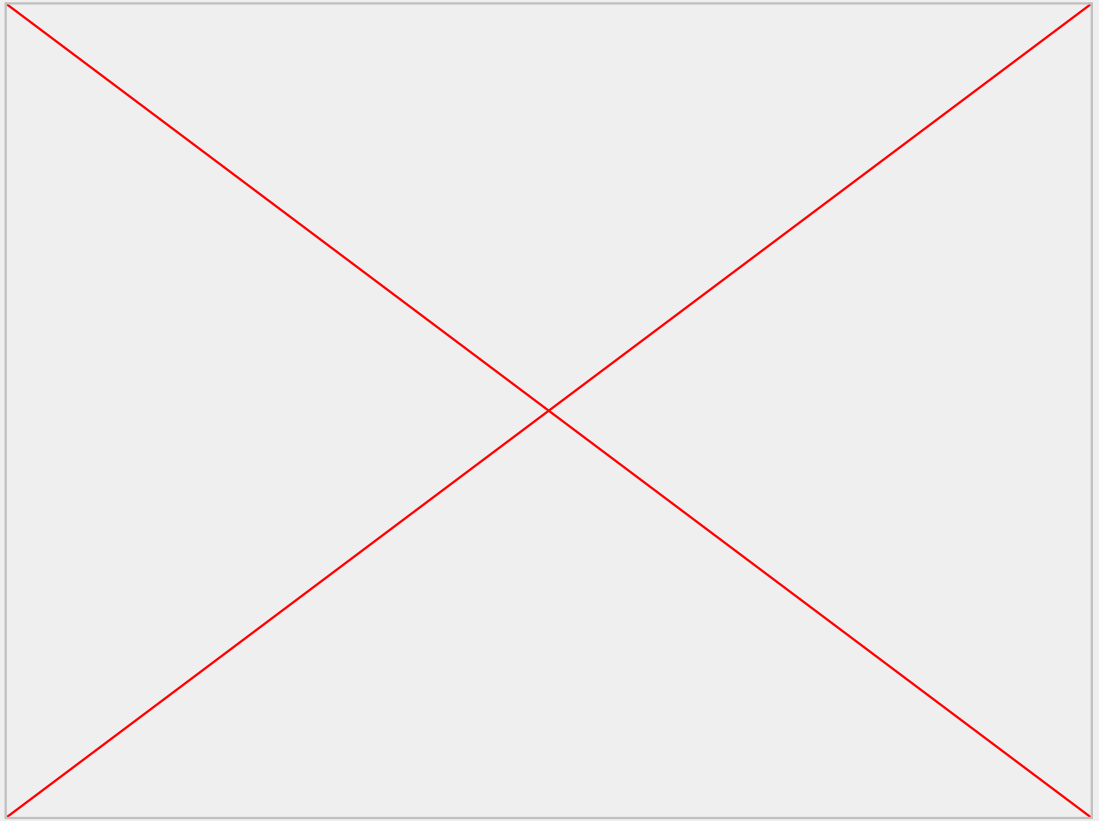
```
void flashPurple() {
  lcd.setCursor(0, 1);
  lcd.print("Soil Dry");
  for (int i = 0; i < 3; i++) {
    setRGB(HIGH, LOW, HIGH);  // Purple
    delay(300);
    setRGB(LOW, LOW, LOW);
    delay(300);
  }
}
```

```
void setGreen() {
  setRGB(LOW, HIGH, LOW);  // Green
  Serial.println("Everything is good! Green Light ON.");
}

void setRGB(int r, int g, int b) {
  digitalWrite(redPin, r);
  digitalWrite(greenPin, g);
  digitalWrite(bluePin, b);
}
```

PROTOTYPE DEMONSTRATION



DEMONSTRATION OF CORE FUNCTION



TESTING AND RESULTS

Test Case	Test Method	Expected Result	Actual Result	Pass/Fail
Temperature Sensor Accuracy*	Compare with a thermometer	The result matched the thermometer	The system displays real-time temperature and humidity levels, providing accurate environmental monitoring.	Pass
Soil Moisture Sensor Calibration*	Measure values in dry, moist, and wet soil	Readings match expected thresholds	The system detects soil values and outputs and determines whether the conditions are classified as "wet" or "dry" based on the measured data.	Pass
Light Sensor Functionality*	Test under dark and bright environment	Readings correspond to light or no light	The system outputs a "light" or "no light" result based on the detected environmental conditions.	Pass
OLED Display Output	Display text messages & check readability	Text and alerts are clearly visible	It did not show any text on the display.	Pass
Data Logging and Communication	Connect to the computer and check synchronization	Data updates accurately with minimal delay	It displays real-time data from each sensor on the computer.	Pass

THANK YOU