Ahmedabad University

ENR 107 Digital Electronics and Microprocessors

Final Report

[Smart Compost Management System]

Submitted to faculty: Prof. Mariyam Kaveshgar

Date of Submission: 23 Nov 2024

Student Details

Roll No.	Name of the Student
AU2340080	Aneri Maniar
AU2340040	Prina Patel
AU2340114	Sakina Jambughodawala
AU2340196	Nidhee Adesara
AU2340264	Saumya Dholakia

2023-2024 (Monsoon Semester)

- **Abstract**

Problem:

The project aims to modernize traditional composting methods by integrating a digital monitoring system that helps optimize the composting process.

Methodology:

- Collecting information related to the project from Research Papers.
- Listing the features we wanted to incorporate in our project.
- Making a circuit in Tinkercad to check the connections.
- Arduino code
- Manual circuit and checking the working of components.
- Making a prototype by including a compost box, compost, and blades.

Results:

The smart compost system showed promise for increasing efficiency and reducing the amount of manual effort required for traditional composting through successful monitoring of important environmental elements and condition (like temperature, moisture, and pH) adjustments to maintain an ideal composting environment during testing. This initiative is a step in the direction of technically innovative and sustainable waste management systems.

- ## Table of Contents

Include section titles with corresponding page numbers.

- **Introduction**

Background: Why is it important or relevant?

There is a drastic increase in waste generation which has become a big problem in managing it. Managing biodegradable waste is critical for reducing greenhouse gasses. Composting manually is a very tedious job, whereas there should be a mechanism that should measure components such as temperature which would be much easier.

Objective:

The objective is to measure various compost conditions such as temperature, pH, and moisture.

- According to conditions, at regular intervals, the system should automatically rotate the fan and mix the compost.
- The system should also display the temperature, pH, and moisture levels on display and the buzzer should switch on when levels are not in optimum conditions.

Scope:

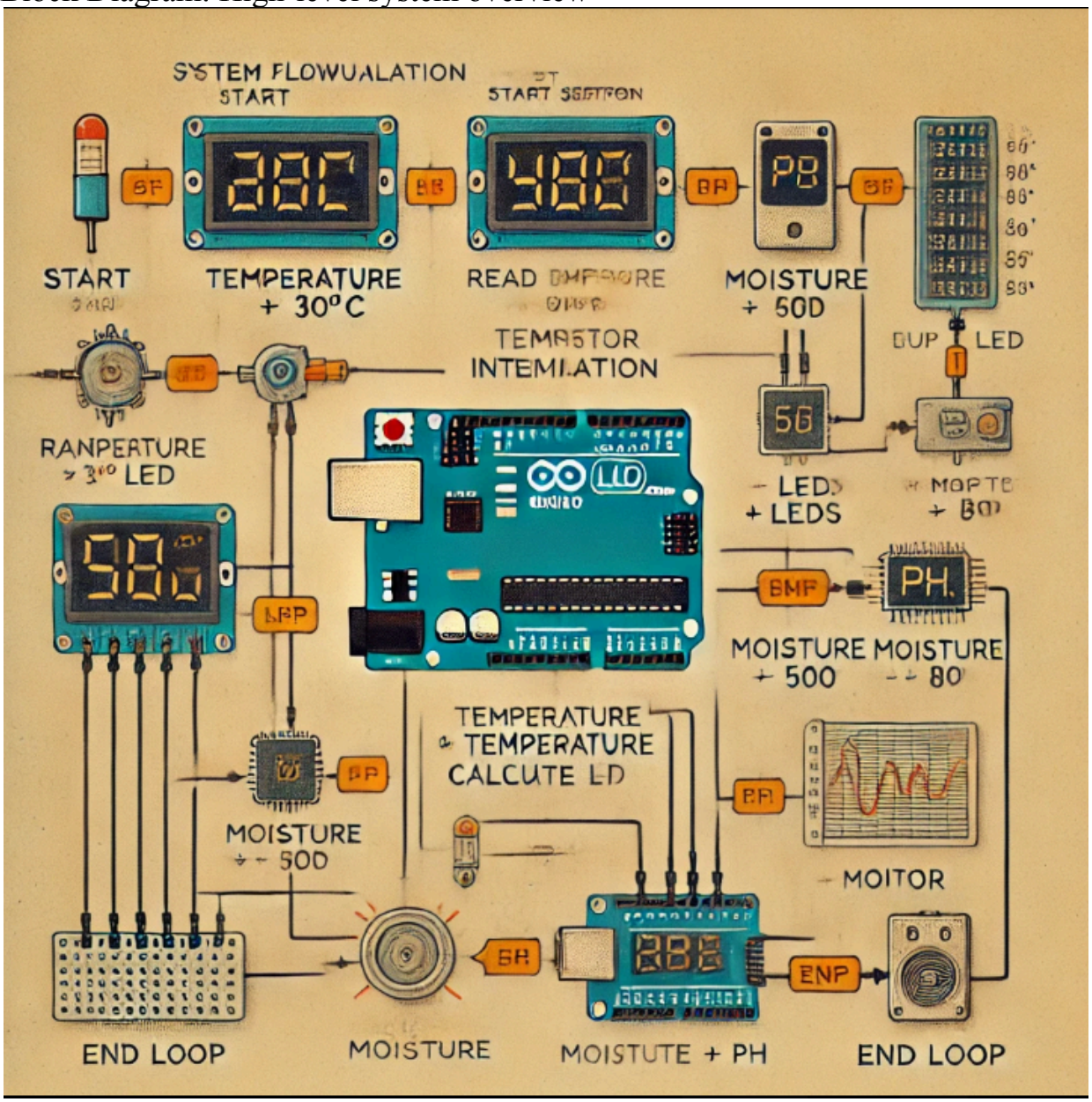
- Components: Incorporating Arduino UNO, buzzer, ph sensor, temperature sensor, moisture sensor, servo motor, motor driver, LCD Display,
- Automatically sensing conditions like temperature, moisture, and pH and displaying it on LCD and alerting via buzzer when necessary.
- Automatically mix up compost when necessary or at regular intervals.

Limitations:

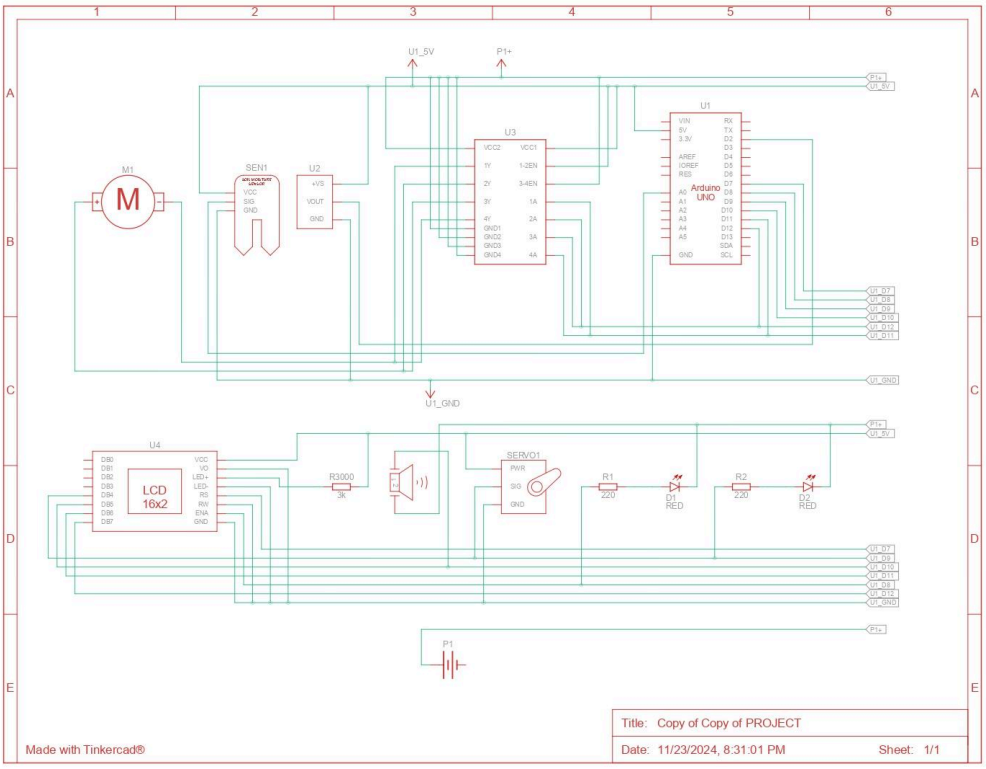
- This system can only be used on a small scale and cannot be used where agriculture takes place on a large scale.
- The accurate working of the model largely depends on the quality of the sensor used in the prototype.
- It requires a continuous power supply to work.

- **Project Design**

Block Diagram: High-level system overview



Circuit Diagram: Detailed Schematic



Components:

Servo motor	1	SG90
Blades	2-4	Stainless Steel
Moisture Sensor	1	Capacitive Soil Moisture Sensor
Temperature Sensor	1	DHT11
pH Sensor	1	Analog pH Sensor Module
LCD Display	1	16x2 LCD with I2C Interface
Buzzer	1	Buzzer module
Resistors	5	220Ω, 10kΩ
Capacitors	3	100µF, 10µF, 0.1µF
Power Supply	1	12V DC Adapter / Battery Pack
Connecting Wires	20+	Male-to-Male / Female-to-Male
Breadboard	1	
LED	2	5mm Red/Green LED
Arduino	1	

Description:

Step1:

DHT11 Sensor: This sensor detects humidity and temperature, two essential parameters for compost condition monitoring. To set it up, connect the DHT11's VCC pin to the Arduino's 5V output and its GND pin to the ground (GND) of the Arduino. For the Arduino to read sensor data, connect the Data pin to Digital Pin 2. Accurate temperature and humidity readings inside the compost bin are ensured by a 10kΩ pull-up resistor between the Data pin and VCC, which regulates the signal.

Step2:

Capacitive Soil Moisture Sensor: This sensor measures the amount of moisture in the soil, which is important for decomposition. Capacitive sensing, which is more dependable for soil-based moisture readings, is how it works. Attach the Arduino's GND to the ground and the VCC pin to 5V. Analog Pin A0 on the Arduino is connected to the sensor's analog output (AO) pin, enabling constant moisture level monitoring that may be modified as necessary for ideal compost conditions.

Step3:

The pH sensor keeps track of the compost's acidity or alkalinity, which is crucial for the best possible microbial activity. Attach the pH sensor's GND pin to the Arduino's ground and its VCC pin to the Arduino's 5V. The Arduino can read the pH levels by connecting the sensor's analog output pin to an analog pin, like A1. By showing whether changes are necessary to prevent excessively acidic or alkaline conditions, which can slow down decomposition, this data aids in maintaining a balanced compost environment.

Step4:

SG90 Servo Motor: By increasing aeration, the SG90 servo motor speeds up decomposition by automating the turning of compost material. Attach the red VCC wire to 5V, the brown GND wire to the ground, and the orange signal line to Digital Pin 9 on the Arduino. With this configuration, the servo angle may be controlled by the Arduino and set to turn the compost at predetermined times.

Step5:

Buzzer Module: A buzzer notifies the user when certain compost parameters, like high temperature or moisture content, are met. Attach the buzzer's positive terminal to Digital Pin 10 and its negative terminal to the ground. At certain levels, the Arduino may sound like a buzzer to alert users when compost needs to be adjusted.

Step6:

L293D Motor Driver: The L293D motor driver is responsible for controlling a DC motor that mixes the compost. Using Arduino signals, this part enables bidirectional motor control. Join Turn on Pin 1 (EN1) to 5V, pins 1 and 2 to digital pins 11 and 12, respectively. The motor

terminals are connected to the pins for outputs one and two. A 12V power source is connected to the driver's VCC pin (Pin 8), and the Arduino's ground is connected to GND. In order to mix the compost, which is essential for aeration and decomposition, this configuration offers power and control.

Step7:

16x2 LCD Display: The LCD display shows temperature, humidity, and moisture levels, providing a real-time compost status update. To simplify connections, use 4-bit mode, where only half of the data pins are required. Connect RS to Digital Pin 7, EN to Digital Pin 6, and D4–D7 to Digital Pins 5, 4, 3, and 2, respectively. Power the display by connecting VCC to 5V and GND to the ground. For contrast adjustment, connect a 10kΩ potentiometer between VCC, GND, and the V0 contrast pin on the LCD. This setup helps users easily view and monitor compost conditions.

Step8:

At last we placed the compost in one box and we placed our circuit in the same box at the side, putting all the sensors in the soil so that it detects the respected things.

- **Methodology**

Development Approach:

1. **Research and Feature Selection**: Begin by reviewing research papers on compost management to understand the key parameters (temperature, humidity, pH, and moisture) and technologies used. Based on this, list the features you wish to include, such as automated mixing, real-time monitoring, and alert notifications, defining the project's scope.
2. **Circuit Design in Tinkercad**: Use Tinkercad to create a virtual circuit with components like the Arduino, LCD, sensors, motor driver, servo motor, and buzzer. This simulation allows you to arrange and test the connections, identifying and solving potential issues before building the physical circuit.
3. **Arduino Coding**: Develop Arduino code to control each component, ensuring correct readings from sensors and proper operation of actuators. Program conditions to activate the buzzer when thresholds (e.g., moisture or pH levels) are met, and test the integrated system within the code.
4. **Manual Circuit Assembly and Testing**: Build the circuit manually, following the Tinkercad layout. Test each component individually, troubleshooting any issues to confirm all parts function as expected before final assembly.
5. **Prototype Assembly and Testing**: Combine the circuit with a compost box that includes compost and blades for mixing. Arrange sensors to monitor conditions within the compost, testing the system's functionality under real-world conditions to confirm effective automation and monitoring.

Programming:

TinkerCad: This is the link where the video of the working of the circuit in Tinkercad is shown:

https://drive.google.com/file/d/1u-_bfAmxaNwTIX_I2-Q-mNRSAb3BYB-b/view?usp=drive_link

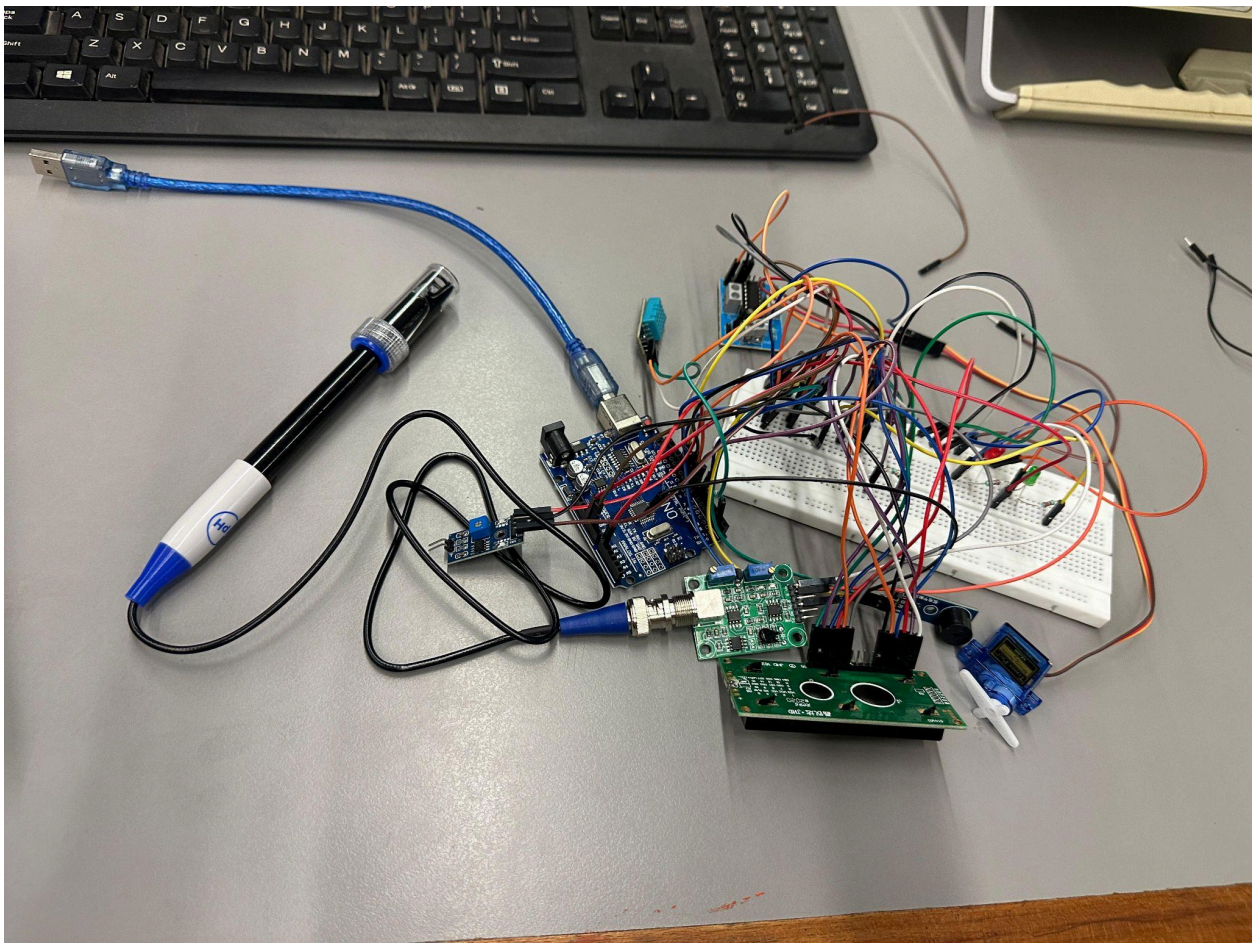
Arduino:

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

// Initialize the LCD (RS, E, DB4, DB5, DB6, DB7)
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

// Defining pins
#define MOISTURE_SENSOR A0
#define TEMP_SENSOR A1
#define PH_SENSOR A2
#define BUZZER_PIN 13
#define TEMP_LED_PIN 2
#define MOIST_LED_PIN 3
#define MOTOR_EN 5
#define MOTOR_IN1 4
```

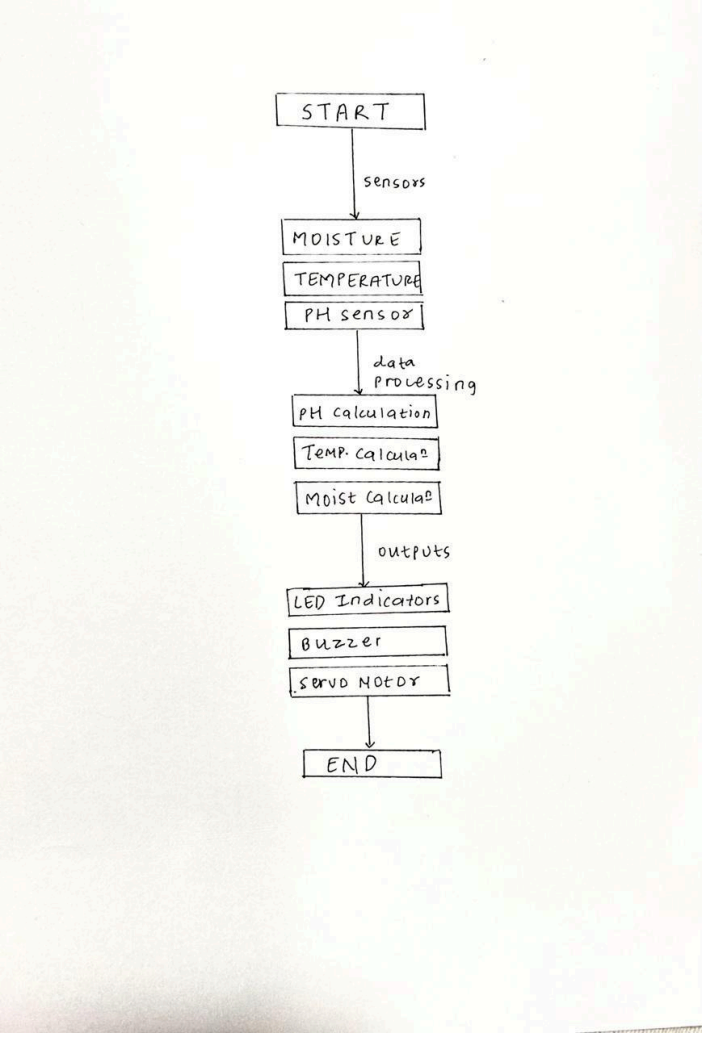
```
#define MOTOR_IN2 6
#define SERVO_PIN 9

Servo servoMotor;

void setup() {
  // Initialize LCD
  lcd.begin(16, 2); // 16x2 LCD
  lcd.print("System Initializing");
  delay(2000); // Display initialization message for 2 seconds
  lcd.clear();

  // Setup pins
  pinMode(MOISTURE_SENSOR, INPUT);
  pinMode(TEMP_SENSOR, INPUT);
  pinMode(PH_SENSOR, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(TEMP_LED_PIN, OUTPUT);
  pinMode(MOIST_LED_PIN, OUTPUT);
  pinMode(MOTOR_EN, OUTPUT);
  pinMode(MOTOR_IN1, OUTPUT);
  pinMode(MOTOR_IN2, OUTPUT);

  // Initialize Servo
  servoMotor.attach(SERVO_PIN);
  servoMotor.write(0); // Move servo to initial position

  // Initialize LCD display
  lcd.print("System Ready!");
  delay(2000);
  lcd.clear();

  Serial.begin(9600);
}

void loop() {
  // Read sensors
  int moistureLevel = analogRead(MOISTURE_SENSOR);
  int tempLevel = analogRead(TEMP_SENSOR); // Calculated temperature sensor
  int pHLevel = analogRead(PH_SENSOR);

  // Calculate temperature conversion
  int temperature = map(tempLevel, 0, 1023, -10, 50); // Example: -10°C to 50°C
  float pH = (pHLevel / 1023.0) * 14.0; // Example: Map analog value to 0-14 pH scale


  if (temperature > 30) {
```

```
  digitalWrite(TEMP_LED_PIN, HIGH);
} else {
  digitalWrite(TEMP_LED_PIN, LOW);
}

if (moistureLevel < 500) { Setting moisture sensor led
  digitalWrite(MOIST_LED_PIN, HIGH)
} else {
  digitalWrite(MOIST_LED_PIN, LOW);
}

// Display values on LCD
lcd.setCursor(0, 0);
lcd.print("Temp: ");
lcd.print(temperature);
lcd.print("C");

lcd.setCursor(0, 1);
lcd.print("Moist: ");
lcd.print(moistureLevel);

delay(2000); // Pause for 2 seconds
lcd.clear();

lcd.setCursor(0, 0);
lcd.print("pH: ");
lcd.print(pH);

lcd.setCursor(0, 1);
lcd.print("Moist: ");
lcd.print(moistureLevel);

delay(2000);
lcd.clear();

// Control Buzzer for alerts
if (temperature > 40 || pH < 5 || pH > 8) {
  digitalWrite(BUZZER_PIN, HIGH);
  delay(1000);
  digitalWrite(BUZZER_PIN, LOW);
}

// Control Servo Motor
if (moistureLevel < 300) {
  servoMotor.write(90);
  delay(2000);
  servoMotor.write(0);
```

```
  }

  // Control Motor
  if (pH < 6 || pH > 8) {
    digitalWrite(MOTOR_EN, HIGH); // Enable motor
    digitalWrite(MOTOR_IN1, HIGH); // Rotate motor in one direction
    digitalWrite(MOTOR_IN2, LOW);
    delay(5000);
    digitalWrite(MOTOR_EN, LOW);
  }
}
```



Flowchart and Pseudocode:



Initialize system:
- Start LCD display and show "System Initializing"
- Set up sensor pins (moisture, temperature, pH, and motor pins)
- Initialize the Servo motor and set it to 0 degrees
- Set up LED pins and Buzzer
- Set serial communication for debugging

Main Loop:
Read sensors:
- Read moisture level from the moisture sensor
- Read the temperature level from the temperature sensor
- Read the pH level from the pH sensor

Convert sensor values:
- Convert temperature from analog to a specific temperature range (e.g., -10°C to

```
50°C)
    - Convert pH value from analog input to 0-14 scale

  Check compost readiness conditions:
    - If moisture > 700, temperature between 25°C and 35°C, pH between 6 and 7.5:
      - Set compostReady to true
      - Turn on Temp and Moisture LEDs
    - Else:
      - Set compostReady to false
      - Turn off Temp and Moisture LEDs

  Display values on LCD:
    - Display temperature and moisture values
    - Clear LCD after 5 seconds
    - Display pH and moisture values

  Control Buzzer:
    - If compost is ready (compostReady is true):
      - Display "Ready" on LCD
      - Turn on Buzzer
      - Wait for 10 seconds
      - Turn off Buzzer
      - Reset compostReady flag to avoid repeated buzzing

  Control Servo Motor (for mixing):
    - If moisture level is low (e.g., < 300):
      - Rotate Servo motor to 90 degrees to simulate compost mixing
      - Wait for 2 seconds
      - Return the Servo to 0 degrees (initial position)

  Control Motor (for mixing if pH is not ideal):
    - If pH is outside the ideal range (less than 6 or greater than 8):
      - Activate motor to mix compost
      - Wait for 10 seconds
      - Turn off motor
```

Testing: Procedures and tools for validation.

Unit Testing:

- Testing individual components (e.g., sensors, servo motor, LCD, etc.) to ensure they function as expected.

Integration Testing:

- Validating the interaction between different components.
- For example Testing the LCD to confirm it correctly displays all the data processed by the Arduino.

Functional Testing**:**

- Testing the system’s overall functionality based on requirements.
- For example ensuring the buzzer sounds when the compost is ready.

Challenges:

- The main challenges included the connecting of multiple sensors and the proper working of them.
- The other challenge was connecting the circuit with the LCD which took much time as there were some errors by us while connecting wires which weren't at the place they should be.
- The other challenge was to connect multiple sensors like temperature sensor, moisture sensor, ph sensor which should work simultaneously.

- **Applications**

1. Efficient Composting in Urban Areas

- It helps to manage organic waste in urban areas, households and apartments.
- It promotes a localized compost management system thus it helps to reduce the dependency on large-scale composting plants.

2. Agricultural Benefits

- It helps farmers to produce high-quality compost by using waste-generated house, thus it also helps reduce the waste
- As compost is generated by waste and garbage it gives a nutrient-rich compost which helps to enrich the soil fertility.

3. Smart Gardening Systems

- For homes and parks, this set up helps to reduce waste and also provides compost which can be used in their gardens.
- Supports rooftop farms as it provides a steady supply of compost, as compost will continue to be generated from daily waste.

4. Environmental Impact

- It helps to reduce landfill waste by converting the biodegradable waste into compost.
- It also decreases methane emissions from organic waste decomposition in landfills.

- **Future Work**

1. IoT Integration for Remote Monitoring

- To enable real-time monitoring we can add a Wi-Fi or Bluetooth module and control via a mobile app.
- Composting progress, sensor data, and system alerts can be viewed by the users.

2. Automated Compost Quality Assessment

- By adding a gas sensor to this (e.g., CO_2 or ammonia sensor) to monitor the decomposition process and ensure optimal compost quality.

3. Renewable Energy Integration

- By including a solar panel to power the system, we can make it energy-efficient and sustainable for remote or off-grid locations.
- Also, this can store excess energy in a battery for continuous operation of the system

\

- # Conclusion

Findings:

1. A smart compost system would automatically sense the conditions like temperature, pH, and moisture of the soil and will regulate the mixing of it.
2. At regular time intervals, it would display the readings of all the sensors.
3. The buzzer would alert the user if there are any unusual readings measured by sensors.

Outcomes:

System Initialization	-	LCD displays "System Initializing" and "System Ready". Sensors and components set up.
Sensor Readings	-	Reads moisture, temperature, and pH values from respective sensors.
Temperature Conversion	-	Converts the analog temperature reading to a range of -10°C to 50°C.
pH Conversion	-	Maps pH analog value to a scale from 0 to 14.
Compost Readiness Check	Moisture > 700, Temp between 25°C-35°C, pH between 6-7.5	Compost is ready. Temp and Moisture LEDs ar turned on.
	Otherwise	Compost is not ready. LEDs remain off.
LCD Display (Sensor Values)	-	Displays current temperature, moisture, and pH values every 5 seconds.

Servo Motor Control	Moisture level < 300	Servo motor rotates to 90 degrees, then returns to 0 degrees after 2 seconds.
Motor Control (pH)	pH < 6 or pH > 8	Motor turns on to mix compost for 10 seconds before turning off.
Continuous Monitoring	-	System loops to read sensors and control components continuously.

Learnings:

1. We gained knowledge on how a compost system works.
2. We learned to integrate multiple sensors with Arduino and make them work.
3. Experienced making a complex circuit including LCD display and a buzzer.

- **References**

  https://ieeexplore.ieee.org/document/9298219

- **Appendix**

  Full source code:

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

// Initialize the LCD (RS, E, DB4, DB5, DB6, DB7)
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

// Define pins for components
#define MOISTURE_SENSOR A0
#define TEMP_SENSOR A1       // Simulated analog pin for temperature sensor
#define PH_SENSOR A2         // Simulated analog pin for pH sensor
#define BUZZER_PIN 13
#define TEMP_LED_PIN 2
#define MOIST_LED_PIN 3
#define MOTOR_EN 5           // Motor Driver Enable Pin
#define MOTOR_IN1 4          // Motor Driver Input 1
#define MOTOR_IN2 6          // Motor Driver Input 2
#define SERVO_PIN 9          // Servo Motor Pin

Servo servoMotor;            // Servo motor object

bool compostReady = false;   // Flag to track compost readiness

void setup() {
  // Initialize LCD
  lcd.begin(16, 2); // 16x2 LCD
  lcd.print("System Initializing");
  delay(2000); // Display initialization message for 2 seconds
  lcd.clear();

  // Setup pins
  pinMode(MOISTURE_SENSOR, INPUT);
  pinMode(TEMP_SENSOR, INPUT);
  pinMode(PH_SENSOR, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(TEMP_LED_PIN, OUTPUT);
  pinMode(MOIST_LED_PIN, OUTPUT);
  pinMode(MOTOR_EN, OUTPUT);
```

```
  pinMode(MOTOR_IN1, OUTPUT);
  pinMode(MOTOR_IN2, OUTPUT);

  // Initialize Servo
  servoMotor.attach(SERVO_PIN);
  servoMotor.write(0); // Move servo to initial position

  // Initialize LCD display
  lcd.print("System Ready!");
  delay(2000);
  lcd.clear();

  Serial.begin(9600); // For debugging
}

void loop() {
  // Read sensors
  int moistureLevel = analogRead(MOISTURE_SENSOR);
  int tempLevel =analogRead(TEMP_SENSOR); // Simulated temperature sensor
  int pHLevel = analogRead(PH_SENSOR);

  // Simulated temperature conversion (adjust as needed)
  int temperature = map(tempLevel, 0, 1023, -10, 50); // Example: -10°C to 50°C
  float pH = (pHLevel / 1023.0) * 14.0; // Example: Map analog value to 0-14 pH scale
  digitalWrite(BUZZER_PIN, LOW);
  // Determine compost readiness condition
  if (moistureLevel > 700 && temperature > 25 && temperature < 35 && pH > 6 &&
pH < 7.5) {
    compostReady = true; // Compost is ready
    digitalWrite(TEMP_LED_PIN, HIGH); // Turn on LED to indicate readiness
    digitalWrite(MOIST_LED_PIN, HIGH);
  } else {
    compostReady = false; // Compost is not ready
    digitalWrite(TEMP_LED_PIN, LOW);
    digitalWrite(MOIST_LED_PIN, LOW);
  }

  // Display sensor values on LCD
  lcd.setCursor(0, 0);
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print("C");

  lcd.setCursor(0, 1);
  lcd.print("Moist: ");
  lcd.print(moistureLevel);
```

```
  delay(5000);
  lcd.clear();

  lcd.setCursor(0, 0);
  lcd.print("pH: ");
  lcd.print(pH);

  lcd.setCursor(0, 1);
  lcd.print("Moist: ");
  lcd.print(moistureLevel);

  delay(2000);
  lcd.clear();

  // Control Buzzer
  if (compostReady) {
    lcd.print("ready");
    digitalWrite(BUZZER_PIN, HIGH); // Turn on buzzer
    delay(10000); // Keep buzzer on for 30 seconds
    digitalWrite(BUZZER_PIN, LOW); // Turn off buzzer
    compostReady = false; // Reset compostReady flag to avoid repeated buzzing
  }

  // Control Servo Motor (Simulate blade movement)
  if (moistureLevel < 300) { // Example: Blade activation condition
    servoMotor.write(90); // Rotate servo to 90 degrees
    delay(2000);
    servoMotor.write(0); // Return to initial position
  }

  // Control Motor (Simulate a motor-driven component, e.g., compost mixing)
  if (pH < 6 || pH > 8) { // Example: Motor activation condition
    digitalWrite(MOTOR_EN, HIGH); // Enable motor
    digitalWrite(MOTOR_IN1, HIGH); // Rotate motor in one direction
    digitalWrite(MOTOR_IN2, LOW);
    delay(10000); // Run motor for 5 seconds
    digitalWrite(MOTOR_EN, LOW); // Turn off motor
  }
}
```

Prototype images:

This is the video:

https://drive.google.com/file/d/1DtlUZWUN6bKeMEVNcFq-xtt35lTvwNOQ/view?usp=drive_link