



Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title
CS5068NI– Cloud Computing & IoT

Smart agriculture monitoring and Irrigation System

Assessment Type
50% Group Report
Semester
2024 Spring
Group Members

London Met ID	Student Name
23047483	Sulav Parajuli
23047492	Sworup Thapa
23047495	Purnika Khadka
23047576	Samyak Dhar Tuladhar
23048524	Dipesh Bhandari
23047560	Nishant Kasaudhan





Assignment Due Date: 06/05/2025
Assignment Submission Date: 06/05/2025
Submitted to: Mr. Sugat Man Shakya
Word Count:2849

*I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked.
I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*




22% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **55 Not Cited or Quoted 21%**
Matches with neither in-text citation nor quotation marks
-  **4 Missing Quotations 1%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 10%  Internet sources
- 6%  Publications
- 21%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Acknowledgement

We profoundly appreciate the opportunity given to us by our supervisor and module leader to work on the Smart Agriculture Monitoring and Irrigation System project. The supervisor's sustained backing along with their expert instruction and essential comments enabled the ultimate development of this project. We are also thankful towards our lecturers along with lab instructors who delivered key concepts and practical practice sessions enabling our understanding of IoT theoretical and practical aspects. We also thank our college for providing the necessary resources and equipment for the project. Finally, we want to express our appreciation to the team members for their cooperative efforts, as their dedication and teamwork contributed for the successful completion of this project.

Abstract

This project introduces a Smart Agriculture Monitoring and Irrigation System that promotes effective agricultural management for farmers through its real-time environmental monitoring capabilities. The system operates by using sensors to collect measurements of soil moisture together with temperature and humidity readings. The irrigation system controls its operations automatically using these measured parameters in order to minimize water consumption. The design enables farmers to obtain data-driven decisions through accurate information from real-time environmental data.

The document describes the operational stages of system development starting from the design process through to software and hardware analysis. The system depends on Arduino Uno as well as soil moisture sensor and DHT11 temperature and humidity sensor and relay module and a 5V water pump as primary hardware elements. Development of the system required detailed planning followed by component acquisition before merging hardware elements with the Arduino IDE for programming capabilities. The report describes the procedure that evaluated the system through real-world testing as well as an account of the evaluation process.

The developed system completed a successful testing phase which proved to provide affordable scalable technology for implementing irrigation optimization. The report shares investigation results about system performance as well as water efficiency outcomes while mentioning future development possibilities. The presented solution adds practical value to smart farming initiatives by providing a functional approach to enhance resource-efficient farming in limited resource zones.

Table of Contents

1. Introduction	1
1.1. Current Scenario.....	2
1.2. Problem statement and Project as a solution.....	2
1.3. Aim and Objective.....	3
1.3.1. Aim	3
1.3.2. Objectives	3
2. Background	4
2.1. System overview.....	4
2.2. Design Diagram	4
2.2.1. Hardware Architecture	4
2.2.2. Flowchart.....	6
2.2.3. Schematic view	7
2.2.4. Circuit Diagram	7
2.3. Requirement Analysis.....	8
2.3.1. Hardware.....	8
2.3.2 Software	13
3. Development	15
3.1. Planning and designing	15
3.2. Resource allocation	15
3.3. System Development.....	16
4. Results and findings	22
4.1. Result	22
4.2. Testing.....	22
4.1.1. Test 1	22
4.1.2. Test 2	24
5. Future works	32

6. Conclusion	33
7. References.....	34
8. Appendix	35
8.1. Source code.....	35
8.2. Code for testing Dht11 sensor	38
8.3. Code for testing soil moisture sensors.....	40

Table of figure

Figure 1Flowchart.....	6
Figure 2Schematic diagram of prototype.....	7
Figure 3Circuit diagram of prototype	8
Figure 4. Arduino Uno	9
Figure 5. Breadboard	9
Figure 6. Soil Moisture Sensor	10
Figure. 7DHT11 Temperature & Humidity Sensor.....	10
Figure 8. Relay Module	11
Figure 9. Jumper Wires	11
Figure 10. 9V Battery	11
Figure 11 Water Pump	12
Figure 12LCD board.....	12
Figure 13 I2C module.....	13
Figure 14. Logo of draw.io.....	14
Figure 15. Tinkercad logo.....	14
Figure 16Screenshot of the Arduino ide with code	16
Figure 17connction between arduino and breadboard	17
Figure 18 Connection of dht11 and soil moisture sensor	18
Figure 19Connection of relay module with Arduino	20
Figure 20 Connecting relay module to arduino.....	21
Figure 21Setup for soil moisture sensor testing	23
Figure 22 Upload success of soil moisture sensor for testing	23
Figure 23 Output of serial monitor	24
Figure 24Setup for dht11 sensor testing	24
Figure 25 Upload success of DHT11 sensor for testing	25
Figure 26 Output of serial monitor	25
Figure 27 Sucessfull compilation of source code	26
Figure 28soil moisture and motor status	27
Figure 29Live update of temperature and humidity	27
Figure 30 Water pump was turned on	28
Figure 31 The motor was turned off after reaching threshold value	29
Figure 32relay module was turned off	30

Figure 33 water pump was turned off	31
---	----

Table of Tables

Table 1Test 1	22
Table 2Test 2	24
Table 3Test 3	26
Table 4Test 4	28

1. Introduction

Connecting everyday items like vehicles and appliances to the internet is known as the Internet of Things, or IoT. To communicate with one another, these items use specialized technology like sensors and software. This information exchange improves system automation and efficiency. IoT is essentially the ability for objects to communicate with one another via the internet. Our lives are being made better by this technology in sectors like farming, healthcare, energy, and the way our houses and cities. It creates a network of computers, devices, and even person with person IDs and the ability to exchange data online, simplifying our everyday activities and communications. (yasar & Gillis, 2024)

IoT connects the digital and physical worlds via a variety of technologies. Real-world objects have sensors, such as those that monitor movement or temperature, and actuators, which act in response to sensor signals. These sensors and actuators communicate with computer systems via wireless (such as Wi-Fi or cellular networks) or wired (such as Ethernet) networks. These computer systems monitor and manage the condition and behavior of the equipment and objects that are connected. However, systems where all the sensors are there only to collect user input such as smartphone apps that primarily collect data via touchscreens or other computer software that uses ordinary keyboards and mouse as sensors are not included in our definition of the Internet of Things. (IBM.com, 2023)

The Smart Agricultural Monitoring and Irrigation System is an IoT-based system designed to help farmers monitor and manage essential environmental conditions such as soil moisture content, temperature, and humidity levels. By using sensors connected to an Arduino microcontroller, the system continuously collects data from the farming field and makes decisions based on that information. When the soil moisture drops below a certain threshold, the system automatically activates a water pump to irrigate the crops. Once the soil reaches an adequate moisture level, the pump is turned off, ensuring efficient use of water. This process helps provide crops with the right amount of water at the right time, improving plant health and crop yield. It also reduces the risk of overwatering or underwatering, which are common issues in traditional farming methods. In addition, the system minimizes the need for farmers to

constantly check the field manually, saving time and effort, especially in remote or large agricultural areas. By automating irrigation and monitoring, this IoT-based solution not only conserves water but also supports more sustainable and productive farming practices.

1.1. Current Scenario

Farming is an important part of life in Nepal, with many people depending on it for their income and daily needs. However, farmers often face problems such as lack of water, hard physical work, and the effects of changing weather. In many areas, rainfall is not regular, and irrigation systems are not well developed, which makes it difficult to grow healthy crops. To help with this, a smart farming system has been developed using simple tools like sensors and an Arduino board. This system checks the soil and automatically waters the crops when the soil becomes too dry, and it stops watering when enough moisture is present. This saves water, reduces the amount of work farmers have to do, and helps crops grow better. The system is affordable, easy to build, and can be very useful for farmers in rural areas. By using this smart system, farming can become more efficient, less tiring, and better suited for changing weather conditions.

1.2. Problem statement and Project as a solution

In Nepal, many farmers still depend on traditional farming methods, which often lead to problems such as water waste, poor crop growth, and low productivity. These challenges are made worse by irregular rainfall, water shortages, and the growing impact of climate change. Manual irrigation takes a lot of time and effort, and farmers in rural areas usually do not have access to modern tools to monitor soil conditions. This can result in either over-watering or under-watering, which harms the crops and reduces yield. To help solve this problem, we have created a Smart Agricultural Monitoring and Irrigation System using simple and low-cost components like Arduino Uno, soil moisture sensors, a temperature and humidity sensor (DHT11), a relay module, and a water pump. This system automatically checks the moisture level in the soil and turns the water pump on or off as needed. It helps farmers water their crops at the right time without having to be in the field constantly. By using this system, water can be used more efficiently, manual labour is reduced, and the overall health of the

crops improves. This project offers a practical and affordable solution for small and rural farmers in Nepal who want to make their farming easier, more efficient, and better prepared for changing weather conditions.

1.3. Aim and Objective

1.3.1. Aim

- To develop a low-cost, automated irrigation system that monitors environmental conditions like temperature, humidity, and soil moisture and adjust watering schedules accordingly such that the system will help optimize water use, ensure the health of crops, and increase farming efficiency.

1.3.2. Objectives

- To design and develop an IoT-based agricultural monitoring system.
- To create an automated irrigation system that operates based on real-time soil moisture data.
- To provide real-time monitoring capabilities through an alarm.
- To minimize water wastage and improve crop yield.

2. Background

2.1. System overview

The smart irrigation system functions as an automated water management system which activates pumps through sensor-based temperature and humidity and moisture information. A water pump operates automatically through sensors that transport environmental information before activating pump action only during soil dryness. An Arduino Uno serves as the system's main controlling component by reading sensor data for both input and output to the pump.

All components work together to let the DHT11 sensor measure both temperature levels and humidity percentages in the atmosphere. The system contains a moisture-sensing mechanism to check soil dryness. The Arduino receives current data from sensors before it transforms this information into commands that activate or deactivate the water pump. Motor driver functions as a protective interface for the water pump operation because it enables safe control of its activation. The system displays present temperature and humidity data along with real-time soil moisture information through its digital LCD screen. The system operates with battery power as its independent source but its components can be modified through jumper wires on a breadboard setup.

The system operates through an automatic feedback loop which keeps sensors active during condition evaluation and uses the Arduino processor to decide actions before the pump turns on automatically. The system operates using a compact design that enables portability while presenting easy-to-understand principles which makes it suitable for basic irrigation automation without human intervention requirements.

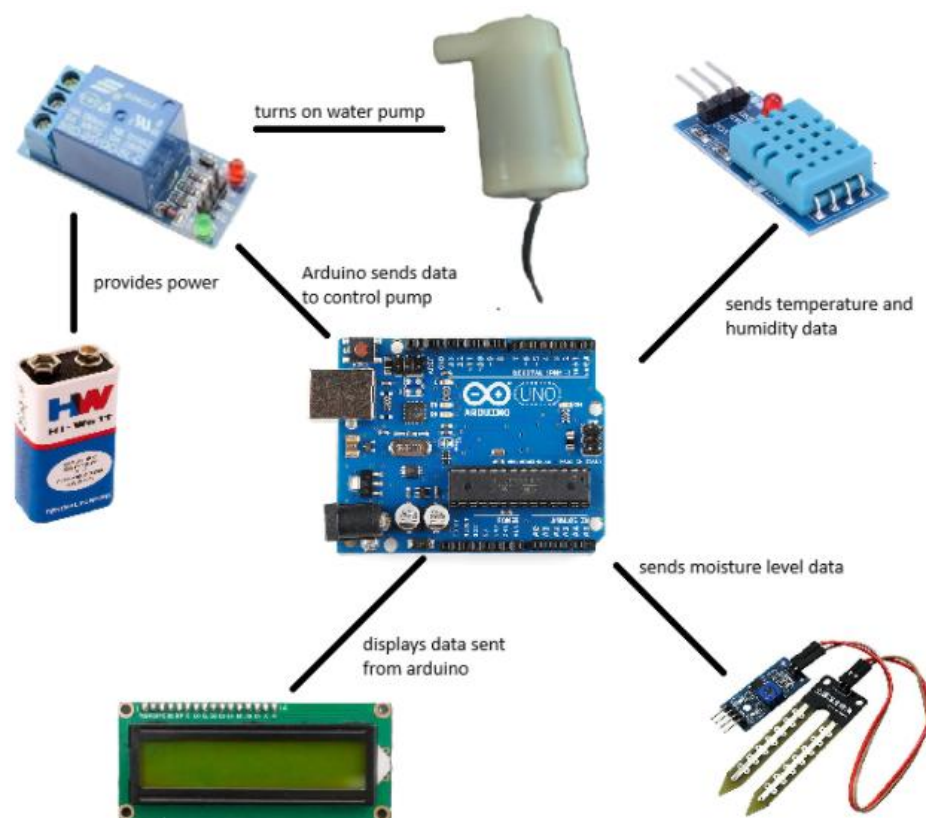
2.2. Design Diagram

2.2.1. Hardware Architecture

This diagram presents the entire hardware structure of the smart irrigation system while displaying the relationships between key components. The Arduino Uno operates as the main controller for the setup. The Arduino Uno obtains signals from both sensors including the DHT11 thermal and humid weather recorder and the soil moisture measuring sensors.

The Arduino computer receives this data before using established parameters to determine its course of action. The Arduino transmits activation signals to the relay module to trigger its function that activates the water pump when the soil remains dry. The watered plants receive automatic irrigation through the pump function. The 9V battery delivers power for running all system components to perform correctly. Through a connection to the Arduino system you can view in real-time the measured values of temperature humidity and soil moisture levels displayed on a digital LCD screen. Users can check environmental data live from the screen without needing digital devices to access.

The diagram presents an easy-to-understand structure of the system design. The labelled components follow a logical pattern to connect so viewers may track data flow and control signalling pathways. The foundational design of automatic irrigation relies on these connected hardware components to demonstrate how basic components construct an efficient intellectual watering system.



2.2.2. Flowchart

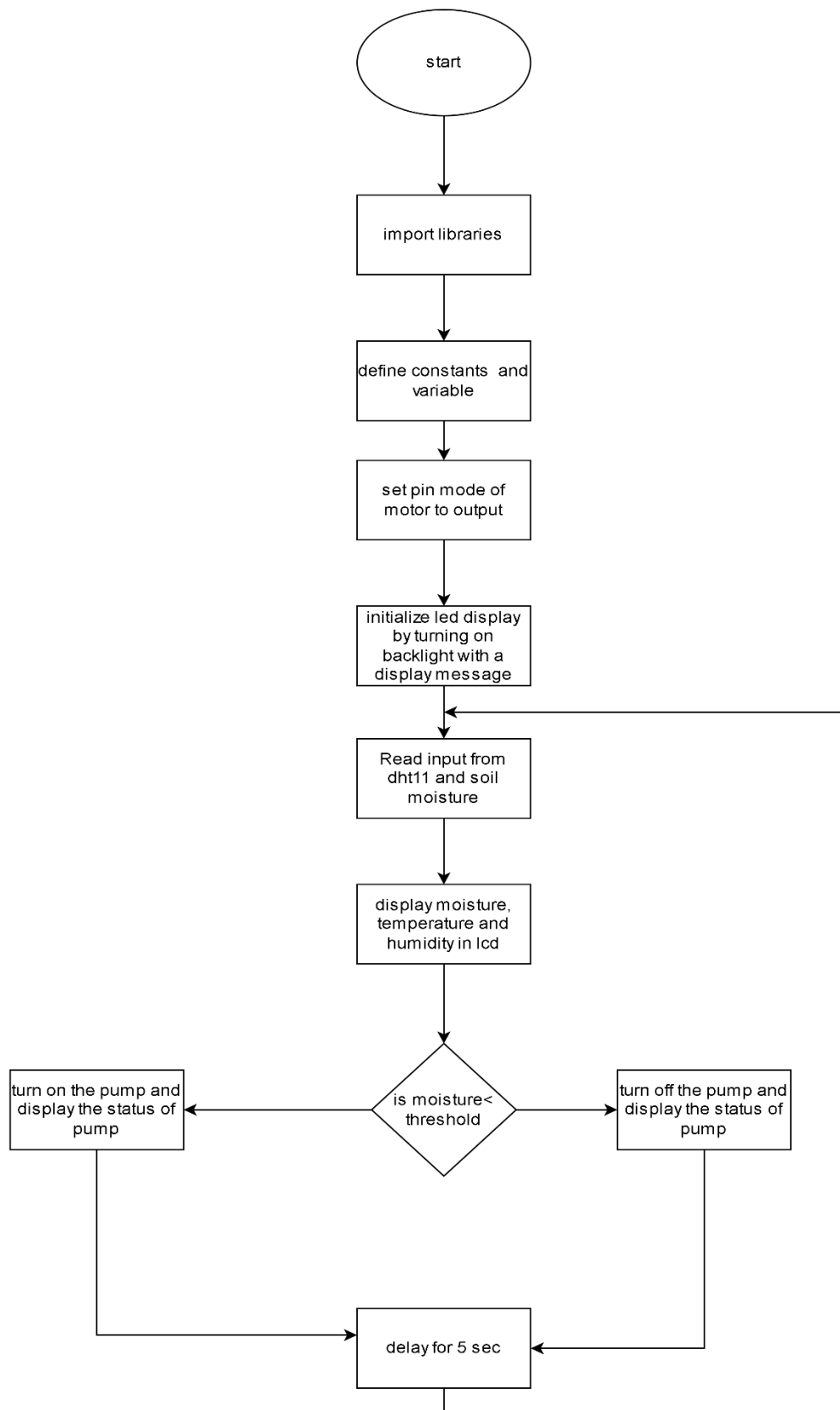


Figure 1Flowchart

2.2.3. Schematic view

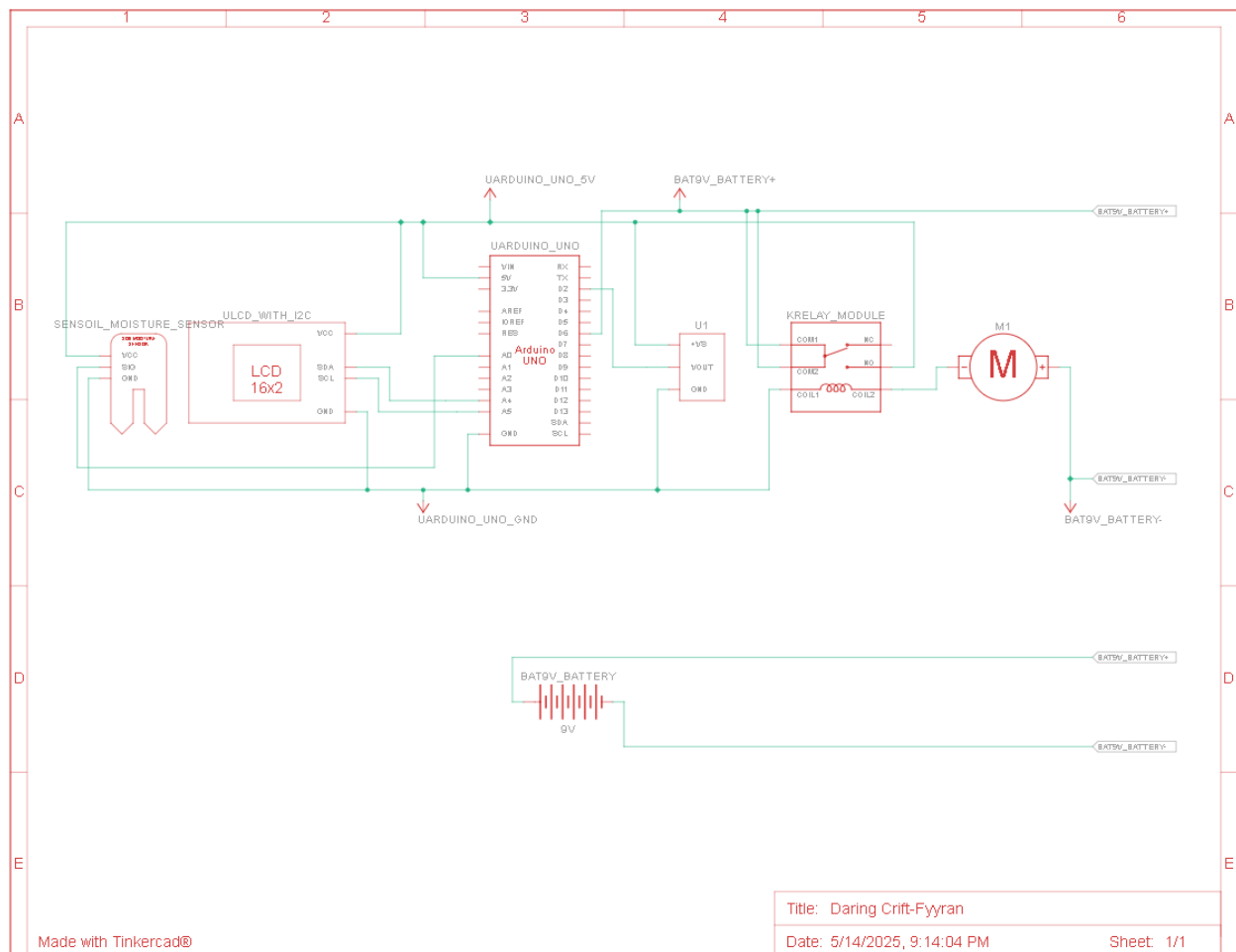


Figure 2 Schematic diagram of prototype

2.2.4. Circuit Diagram

A circuit diagram presents an electronic system through graphical representation which demonstrates component connectivity for functional achievement. Through this drawing engineers can observe both the wires and pin arrangements and project layout design. The design process and maintenance activities of Arduino-based automation tasks strictly require circuit diagrams to function well.

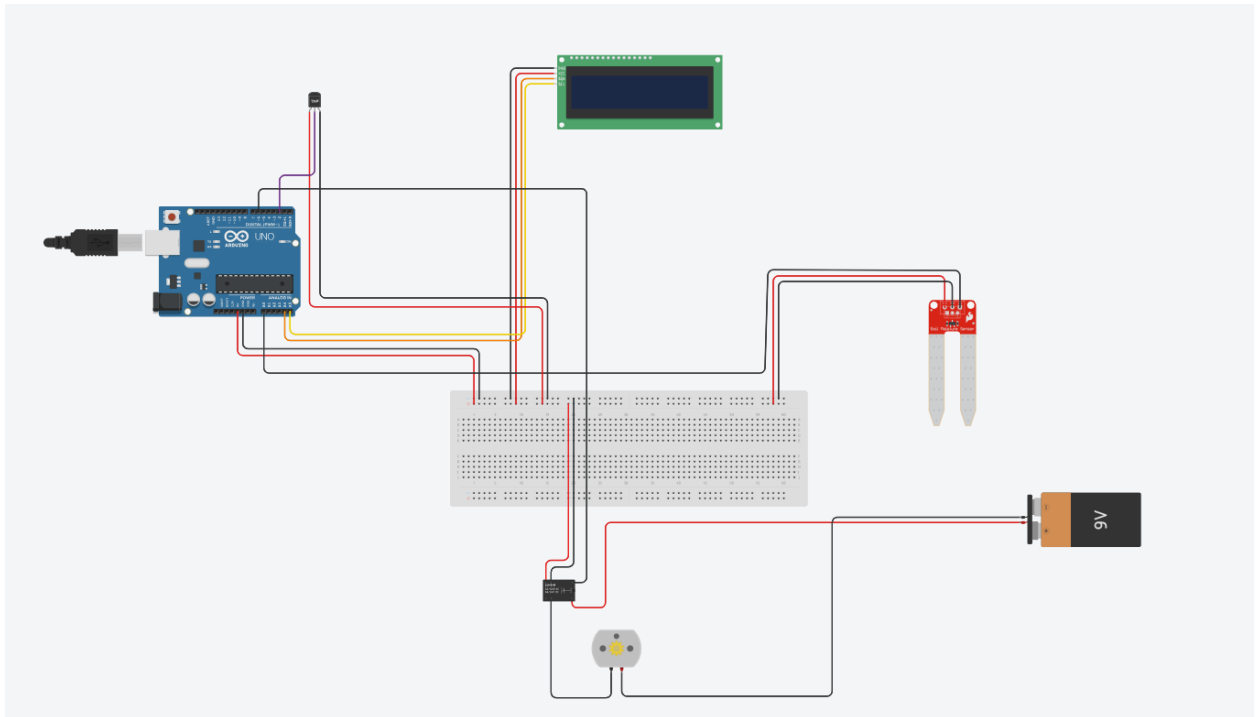


Figure 3Circuit diagram of prototype

A smart irrigation system that utilizes an Arduino Uno appears in the provided circuit diagram. The system incorporates the soil moisture sensor together with a DHT11 temperature and humidity sensor and a relay module and an LCD display and a water pump and a 9V battery. The Arduino system receives sensor data before using this information to operate the water pump by means of a relay. The LCD displays real-time environmental values. A breadboard connects all system components through jumper wires which facilitates communication and power distribution across the system

2.3. Requirement Analysis

2.3.1. Hardware

i. Arduino Uno:

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards can read inputs – such as signals from sensors – and turn them into outputs – such as activating a motor or displaying information. In this project, the Arduino Uno acts as the central controller, collecting sensor data and managing the irrigation system. (Arduino , 2022)

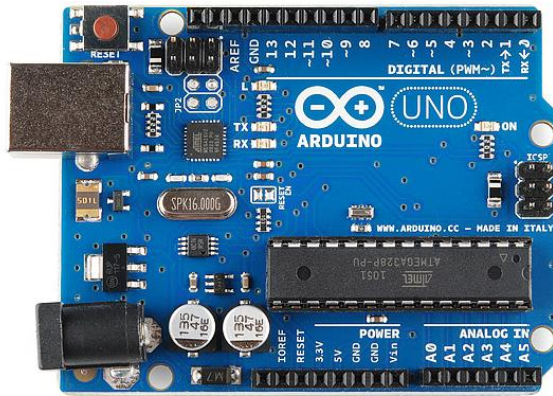


Figure 4. Arduino Uno

ii. Breadboard:

A breadboard is a construction base for prototyping electronics. It allows you to build circuits without soldering by plugging components and wires into the grid of holes. This is helpful for testing and modifying the project before making a permanent version.

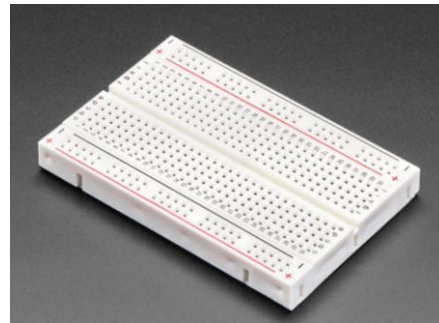


Figure 5. Breadboard

iii. Soil Moisture Sensor:

A soil moisture sensor detects the volumetric water content of the soil using conductive probes. It produces an analog value representing how wet or dry the soil is. In this system, it is used to determine whether the soil requires irrigation. (AgriTechTomorrow, 2019)

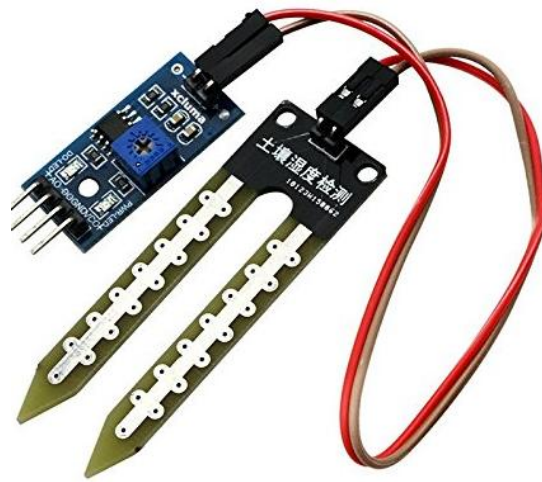


Figure 6. Soil Moisture Sensor

iv. DHT11 Temperature & Humidity Sensor:

DHT11 is a basic, ultra-low-cost digital sensor that measures temperature and humidity. It uses a capacitive humidity sensor and a thermistor to provide calibrated digital output. It helps in monitoring the weather conditions around the crops. (Adafruit, 2012)

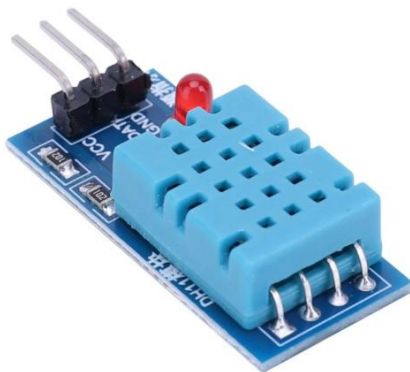


Figure. 7DHT11 Temperature & Humidity Sensor

v. Relay Module:

A relay is an electrically operated switch that allows low-voltage circuits (like Arduino) to control high-voltage devices (like a water pump). The relay used in this project helps switch the water pump ON or OFF safely.



Figure 8. Relay Module

vi. Jumper Wires:

Jumper wires are flexible wires that help to make connections between electronic components on a breadboard or between a breadboard and the Arduino. They are essential for circuit prototypes.



Figure 9. Jumper Wires

vii. 9V Battery:

A 9V battery provides the necessary power supply to the Arduino and its components. For field deployment, a battery is preferred, while USB is used during development.



Figure 10. 9V Battery

viii. Water Pump

A water pump is a mechanical device designed to move water from one location to another, utilizing pressure changes within a system (INECO, 2025).



Figure 11 Water Pump

ix. LCD 16x2

Liquid crystal display is a electronic display device that operates by applying a varying electric voltage to a layer of liquid crystal, thereby inducing changes in its optical properties. LCDs are commonly used for portable electronic games, as viewfinders for digital cameras and camcorders, in video projection systems, for electronic billboards, as monitors for computers, and in flat-panel televisions (Walton, 2025).

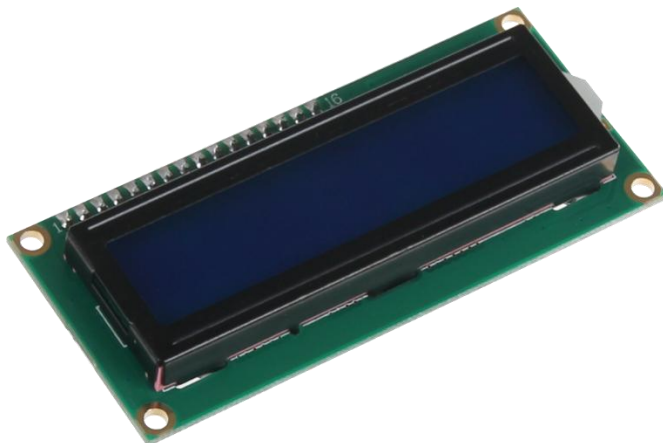


Figure 12LCD board

x. I2C

An I2C LCD module is one of such and is a LCD module that communicates with a microcontroller or another device with the help of the I2C communication

protocol. I2C is a model or a series of interactive display panels, especially constructed by Sharp within their AQUOS BOARD line.



Figure 13 I2C module

3.2.2 Software

i. Arduino IDE:

The Arduino Integrated Development Environment (IDE) is the official software used to write, compile, and upload code to Arduino boards. It provides a simple and user-friendly interface where users can write code using the Arduino programming language (a simplified version of C/C++). The IDE supports features like syntax highlighting, auto-formatting, serial communication, and built-in examples, which make it ideal for both beginners and professionals in embedded systems development. (Arduino, 2024)

ii. Draw.io

draw.io is a free, web-based diagramming tool that allows users to create a variety of visual representations, including flowcharts, UML diagrams, and more. It offers a comprehensive set of features, extensive shape libraries, and AI-powered smart templates, making it a versatile tool for various applications. (draw.io, 2025)



Figure 14. Logo of draw.io

iii. Tinkercad:

Tinkercad is a free, web-based 3D modelling program developed by Autodesk. It's known for its user-friendly interface and ease of use, making it a popular choice for beginners, educators, and hobbyists alike. Tinkercad allows users to create 3D designs, simulate electronic circuits, and even code using block-based programming. (tinkercad.com, 2025)



Figure 15. Tinkercad logo

3. Development

3.1. Planning and designing

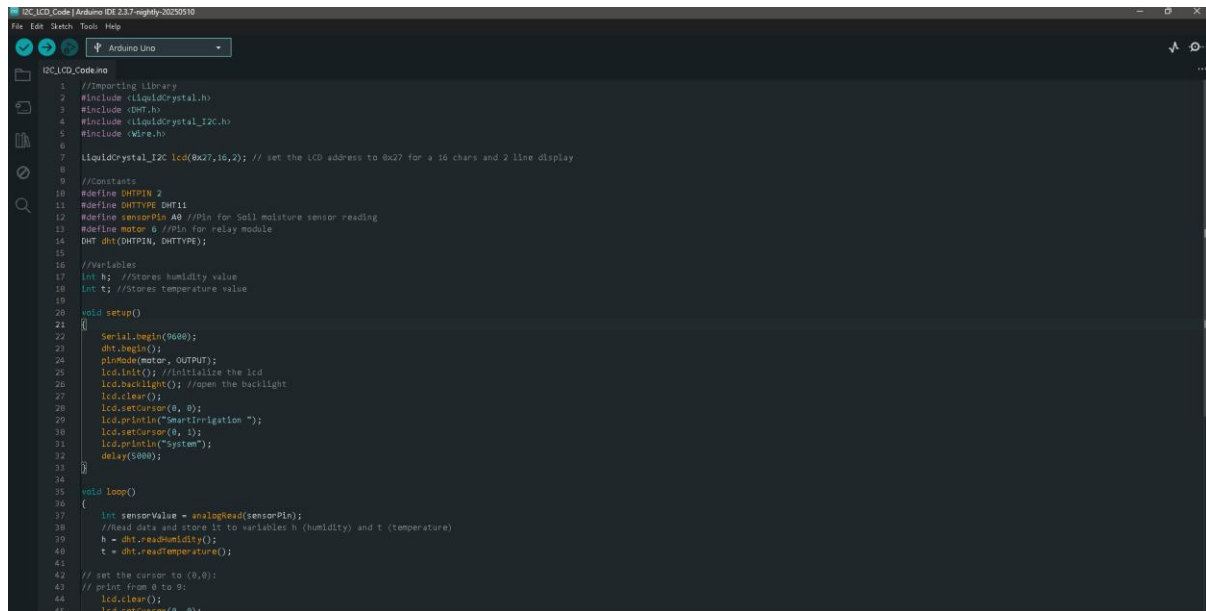
We formed a group on social media for our coursework and began by sharing different project ideas. Each member contributed suggestions, and after discussing their usefulness and feasibility, we decided to work on an automatic smart irrigation system. Choosing the right project and adjusting it to meet the coursework requirements was challenging at first. However, by working together, we successfully planned a solution that would be practical and easy to build. At the beginning, our design was basic and focused only on monitoring soil moisture. But after reviewing the requirements and exploring how to make the project more effective, we decided to include additional components such as a temperature and humidity sensor and a relay-operated water pump. Our main goal with this project was to help small-scale and rural farmers who often struggle with traditional irrigation methods. The system we designed helps reduce water waste and physical effort by automatically watering the crops when the soil becomes dry. It is especially helpful for farmers who cannot always be present in the field, offering a reliable, low-cost, and user-friendly solution for better crop care.

3.2. Resource allocation

The resources required for developing our IoT-based irrigation project were already available to our team leader. After finalizing the project idea, we listed out all the components needed for the system. We then submitted an application to Mr Shishir Subedi, requesting the necessary components from the college. After approval, we collected the components from the IT resource department. The materials needed for our project included an Arduino Uno, a soil moisture sensor, a DHT11 temperature and humidity sensor, a 1-channel relay module, a small 5V water pump, jumper wires, and a 9V battery with a connector. Among these, we received the Arduino Uno and the DHT11 sensor, relay module, female to female jumper wire, soil moisture from the college, while the rest of the components were arranged by the team. These resources helped us successfully build and test our smart irrigation system, which automatically controls the water supply based on soil moisture level.

3.3. System Development

Step 1: In the first step, we connected Arduino with a laptop using USB 2.0 Type A to Type B cable. Then we programmed a Arduino code for the project using many reference videos from YouTube. We debugged the code multiple times and after all the error were removed we uploaded the file to Arduino.



```
1 //Importing library
2 #include <LiquidCrystal.h>
3 #include <DHT.h>
4 #include <LiquidCrystal_I2C.h>
5 #include <Wire.h>
6
7 LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD address to 0x27 for a 16 chars and 2 line display
8
9 //Constants
10 #define DHTPIN 2
11 #define DHTTYPE DHT11
12 #define sensorPin A8 //Pin for Soil moisture sensor reading
13 #define motor 6 //Pin for relay module
14 DHT dht(DHTPIN, DHTTYPE);
15
16 //Variables
17 int h; //Stores humidity value
18 int t; //Stores temperature value
19
20 void setup()
21 {
22   Serial.begin(9600);
23   dht.begin();
24   pinMode(motor, OUTPUT);
25   lcd.init(); //Initialize the lcd
26   lcd.backlight(); //open the backlight
27   lcd.clear();
28   lcd.setCursor(0, 0);
29   lcd.println("SmartIrrigation ");
30   lcd.setCursor(0, 1);
31   lcd.println("System");
32   delay(1000);
33 }
34
35 void loop()
36 {
37   int sensorValue = analogRead(sensorPin);
38   //Read data and store it to variables h (humidity) and t (temperature)
39   h = dht.readHumidity();
40   t = dht.readTemperature();
41
42   // set the cursor to (0,0):
43   // print from 0 to 5:
44   lcd.clear();
45   lcd.setCursor(0, 0);
```

Figure 16 Screenshot of the Arduino ide with code

Step 2: In this step we connected 5v and GND pins of the Arduino to the breadboard. This helped us to extend the limited pins of Arduino as there are multiple devices requiring the 5v and GND pins and made the connection easier due to breadboard.

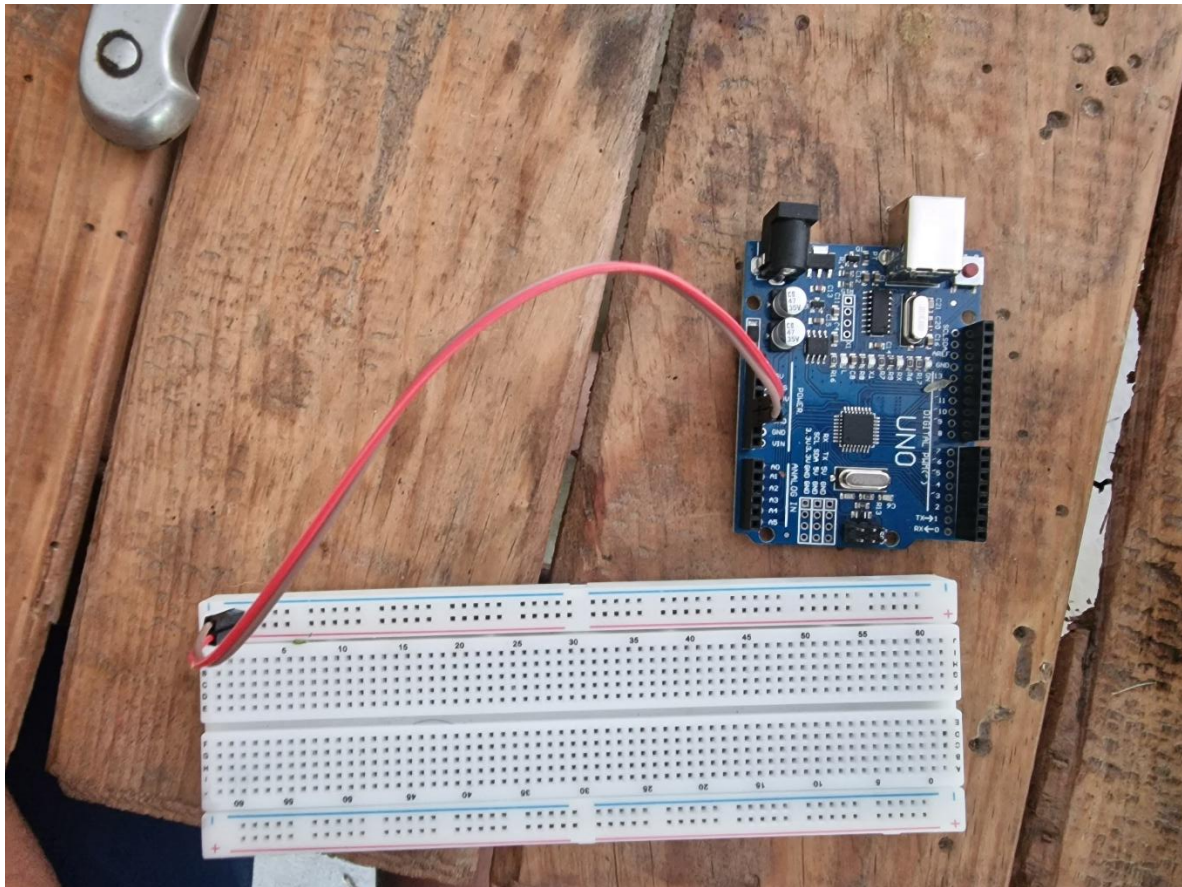


Figure 17 connection between arduino and breadboard

Step 3: In this step DHT11 sensor and moisture sensor were connected to the Arduino and breadboard. The VCC was connected to the positive and GND was connected to the negative pin in the breadboard. Data pin of DHT11 was connected to digital pin 2 and AO was connected to A0 of Arduino.

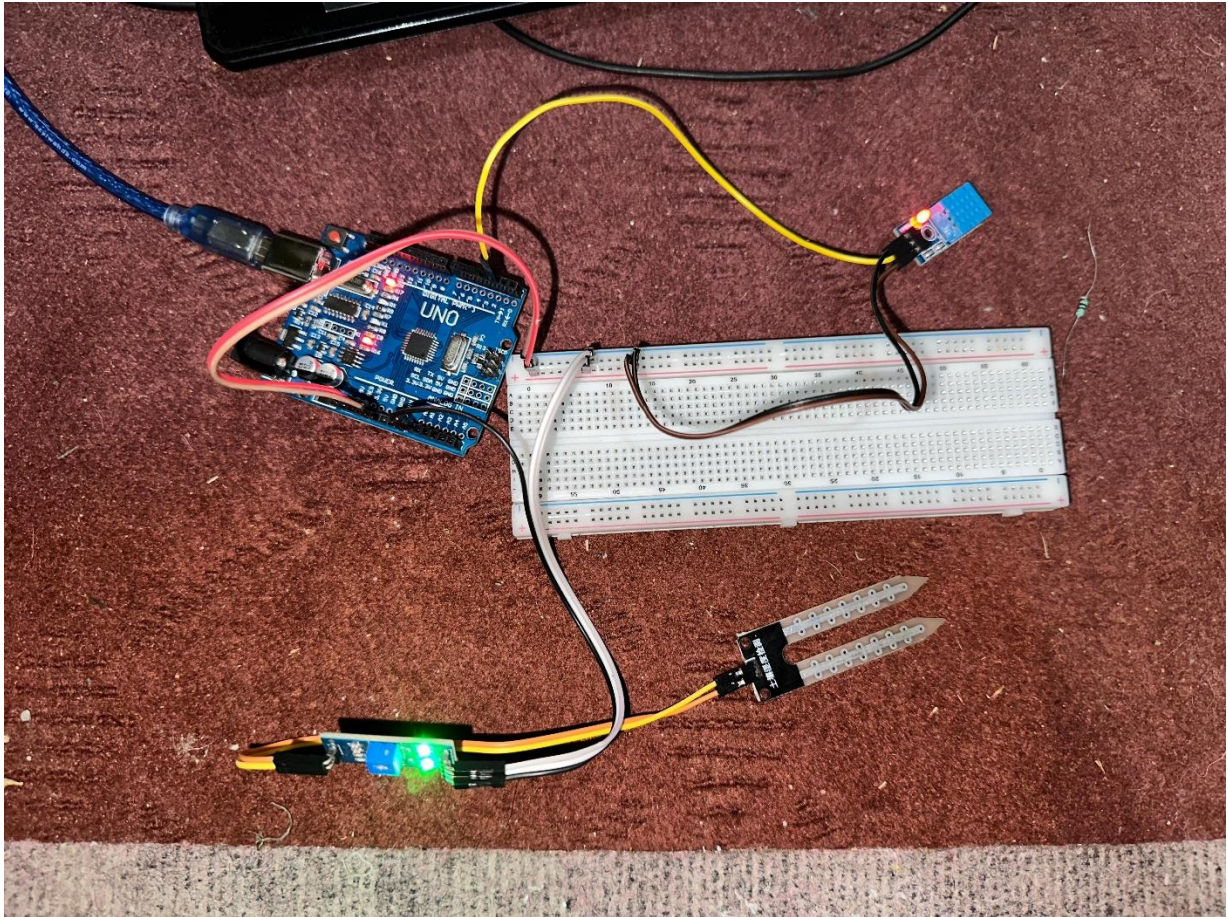
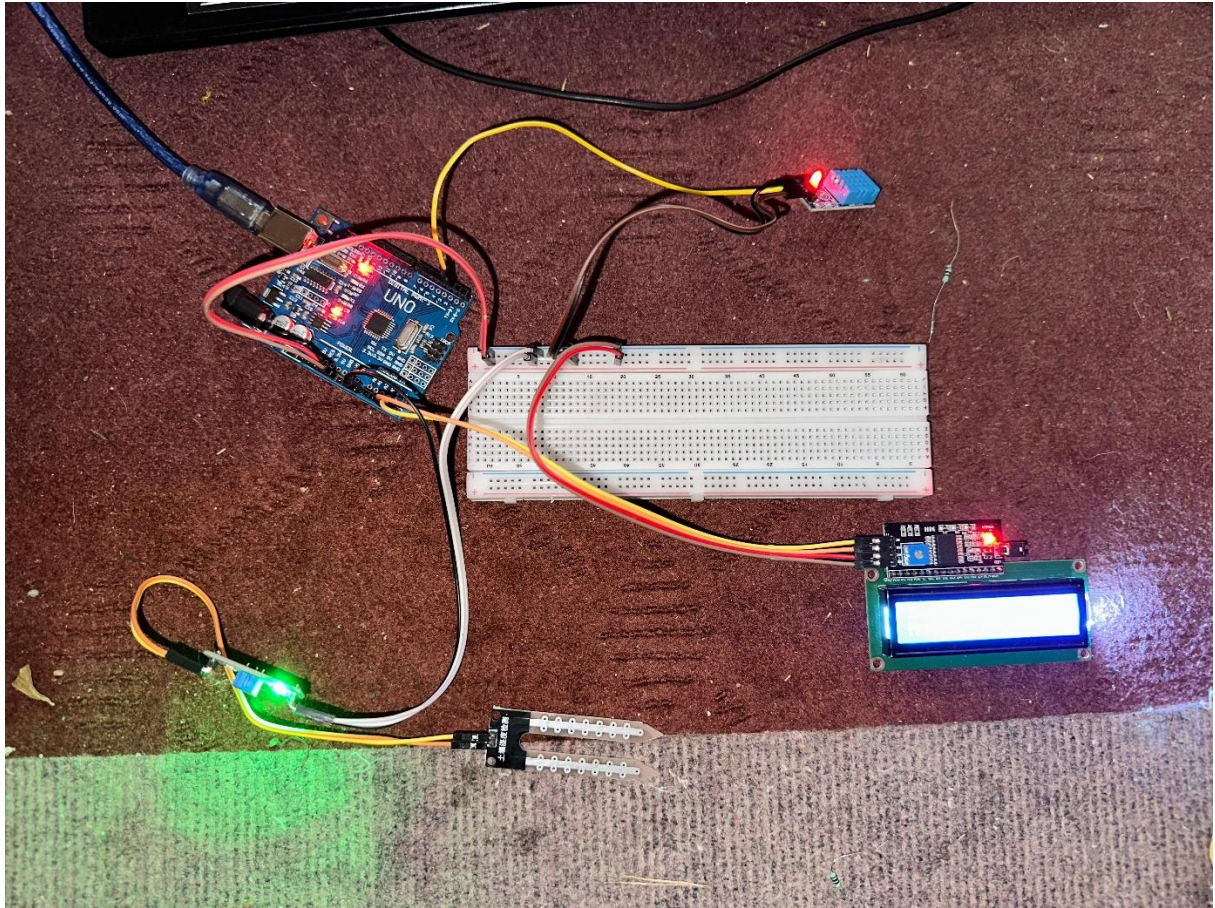
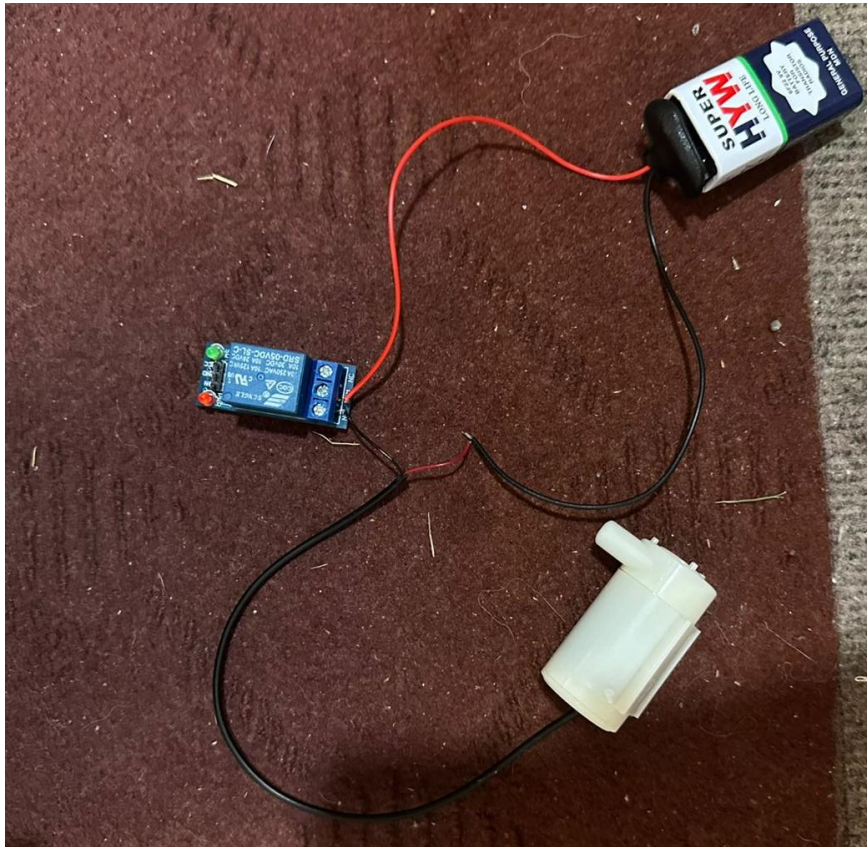


Figure 18 Connection of dht11 and soil moisture sensor

Step 4: LCD board was connected to the system in order to see the output of the sensors. To make the connection of LCD board easier we use I2C module while make the wiring easier. The VCC and GND was connected to the breadboard, SDA and SCL pin were connected to the A4 and A5 pin of Arduino respectfully.



Step 5: Relay module was connected to the water pump. And battery positive terminal of the battery was connected to COM port and negative terminal of the battery was connected to water pump. Another end of the Water pump is connected to NO port.



*Figure 19*Connection of relay module with Arduino

Step 6: In the final step, VCC and GND pins of the Relay module was connected to positive and negative terminal of breadboard and IN pin was connected to digital pin 6 of Arduino.

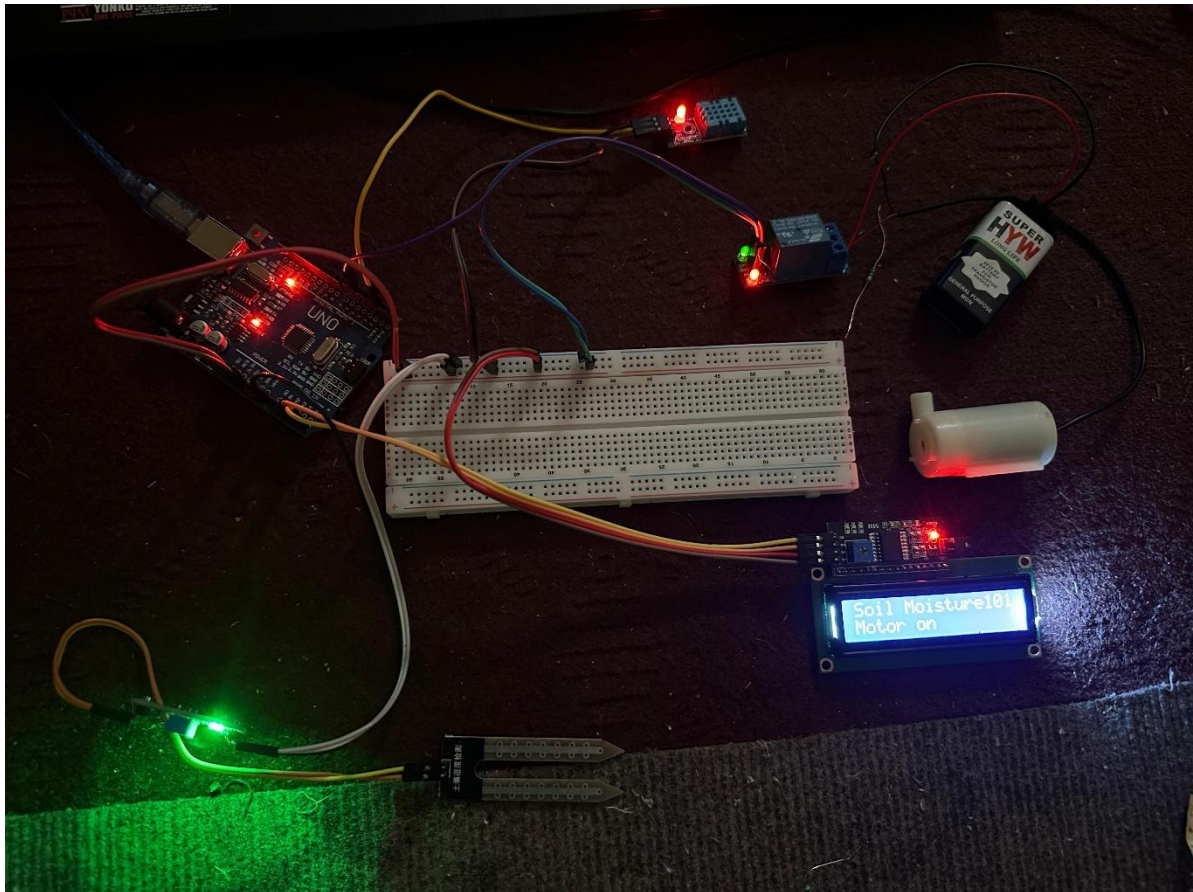


Figure 20 Connecting relay module to arduino

4. Results and findings

4.1. Result

Even before beginning the actual development, the idea of creating an automatic smart irrigation system gave us a clear picture of how helpful the final outcome would be for small-scale farmers. By the end of the project, we had successfully built an IoT-based irrigation system that could automatically manage watering based on the soil's moisture level. The system included a soil moisture sensor that detected when the soil was dry and triggered the water pump through a relay module. It also stopped watering once the soil reached the desired moisture level. The system was tested multiple times, and the results showed that it worked as expected. This project addressed two common problems in farming which are overwatering and water waste while also reducing the need for manual labour. The smart irrigation system we developed offered a practical and affordable solution for improving water management in agriculture, especially in areas where resources are limited.

4.2. Testing

4.1.1. Test 1

Objective:	To check the working of soil moisture sensor.
Action:	Soi moisture sensor code was executed, the result was checked in serial monitor
Expected Result	The soil moisture sensor should send data periodically.
Actual Result	The soil moisture sensor sent data periodically.
Conclusion	The test was successful

Table 1Test 1

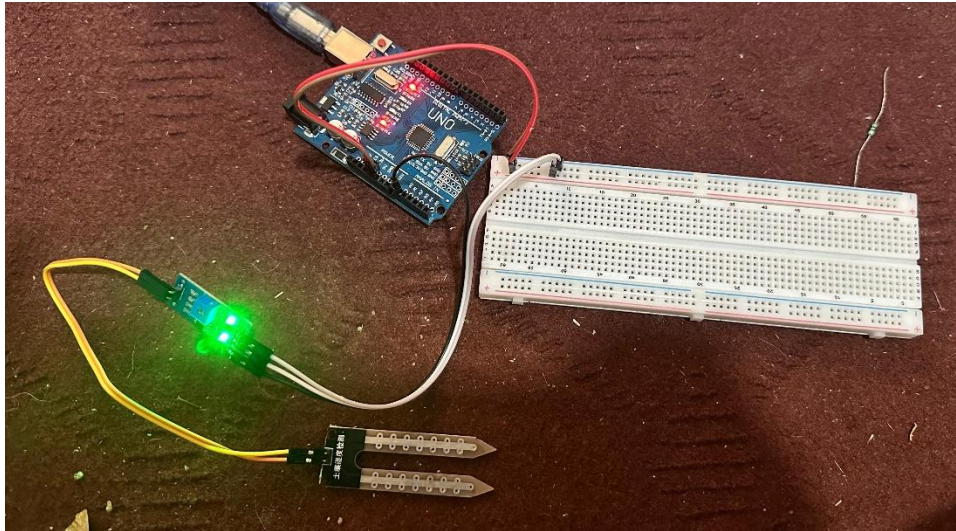


Figure 21 Setup for soil moisture sensor testing

A screenshot of the Arduino IDE interface. The main window displays a sketch named 'sketch_may15a.ino' with the following code:

```
1 const int sensorPin = A0; // Analog pin connected to the sensor
2 void setup() {
3   Serial.begin(9600); // Initialize serial communication
4 }
5 void loop() {
6   int sensorValue = analogRead(sensorPin); // Read the analog value from the sensor
7   Serial.print("Soil Moisture Value: ");
8   Serial.println(sensorValue); // Print the sensor value to the serial monitor
9   delay(1000); // Delay for 1 second
10 }
11
```

The 'Output' window at the bottom shows the following message:

```
Sketch uses 1922 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 210 bytes (10%) of dynamic memory, leaving 1838 bytes for local variables. Maximum is 2048 bytes.
```

A status bar at the bottom right indicates 'Ln 11, Col 1' and 'Arduino Uno on COM4'. A notification bubble at the bottom center says 'Done uploading.'.

Figure 22 Upload success of soil moisture sensor for testing

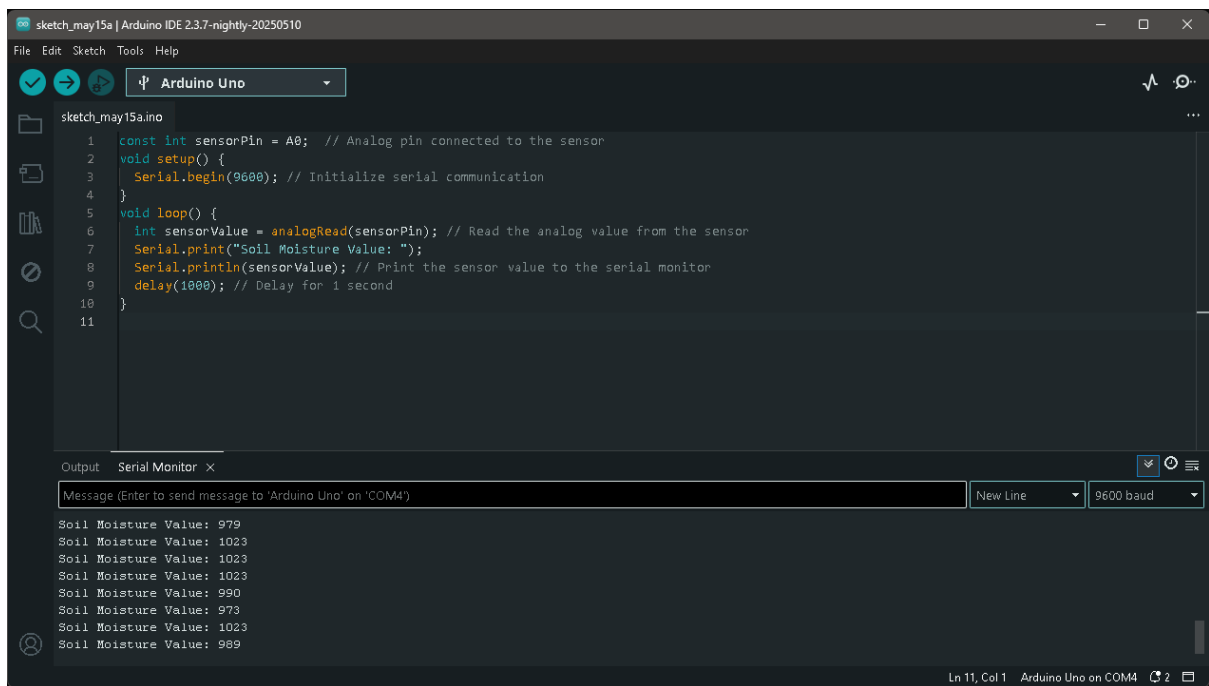


Figure 23 Output of serial monitor

4.1.2. Test 2

Objective:	To check the working of Dht11 sensor.
Action:	Dht11 sensor code was executed, the result was checked in serial monitor
Expected Result	The Dht11 sensor should send data periodically.
Actual Result	The Dht11 sensor sent data periodically.
Conclusion	The test was successful

Table 2 Test 2

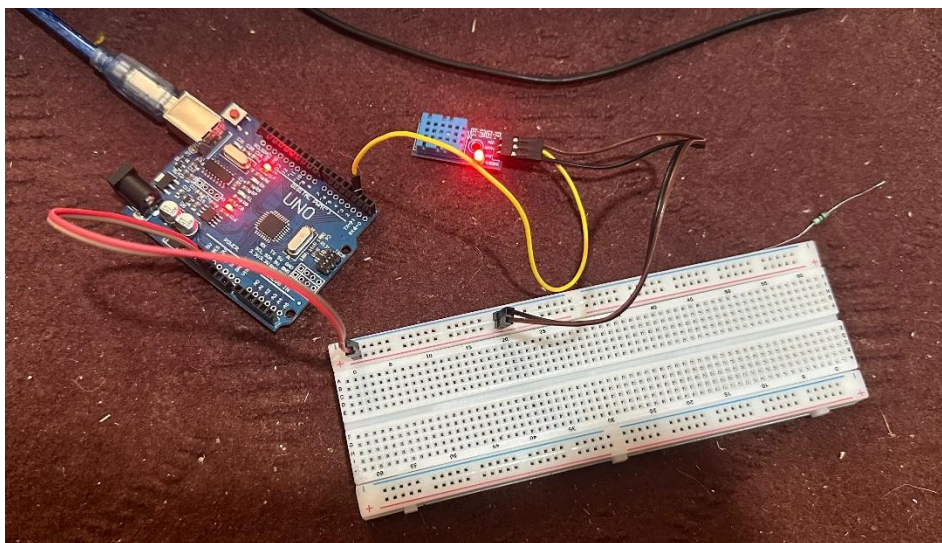


Figure 24 Setup for dht11 sensor testing

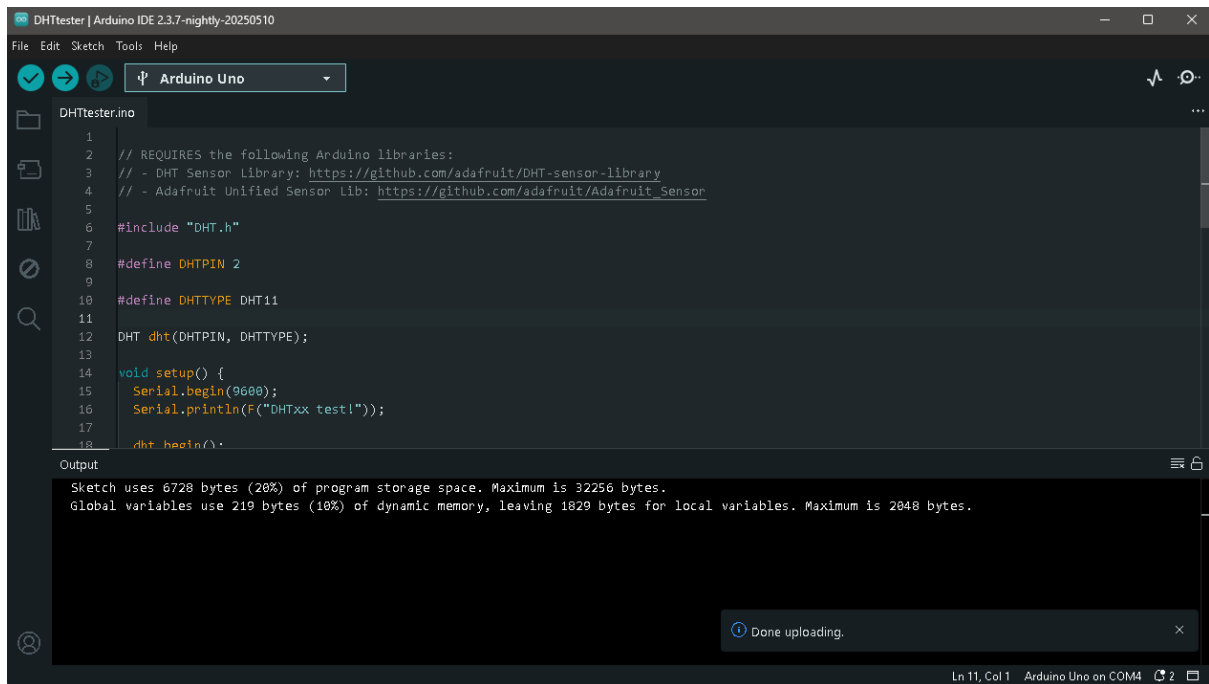


Figure 25 Upload success of DHT11 sensor for testing

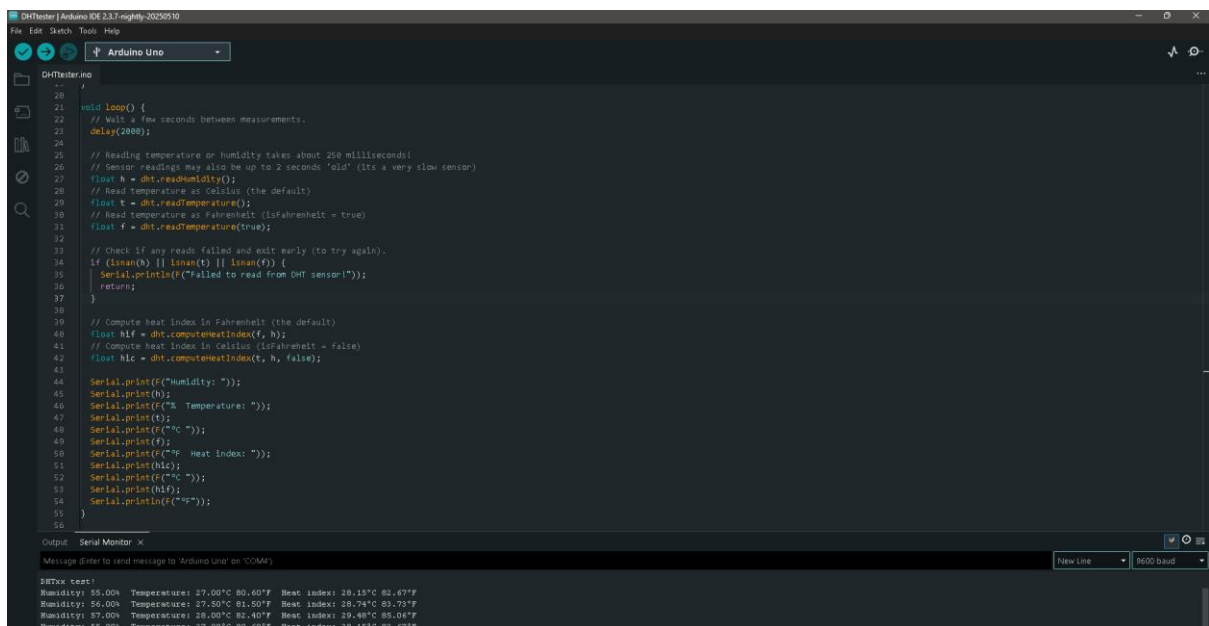


Figure 26 Output of serial monitor

4.1.3. Test 3

Objective:	To check if the water pump turns on when the soil moisture is below the threshold value.
Action:	Source code was executed and observed the results.
Expected Result	The water pump should turn on, with changes to the led display message.

Actual Result	The water pump turned on, the led display message displayed the pump on and continuous monitoring of the temperature, humidity and moisture.
Conclusion	The test was successful

Table 3 Test 3

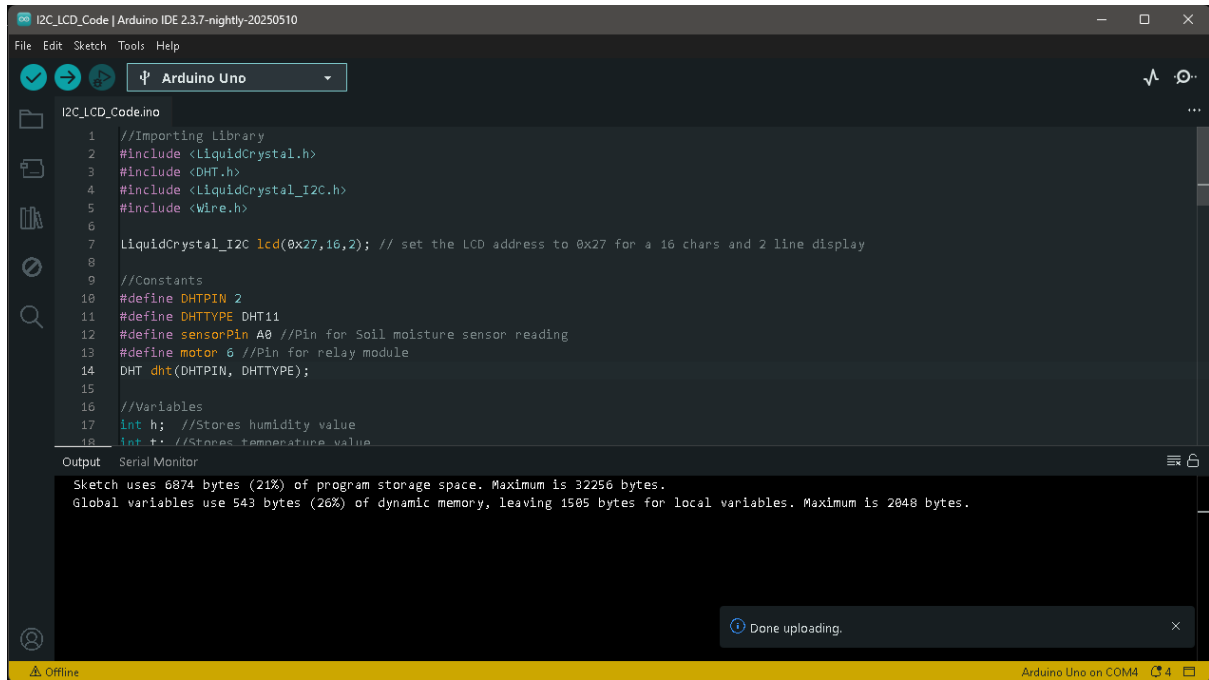


Figure 27 Successfull compilation of source code

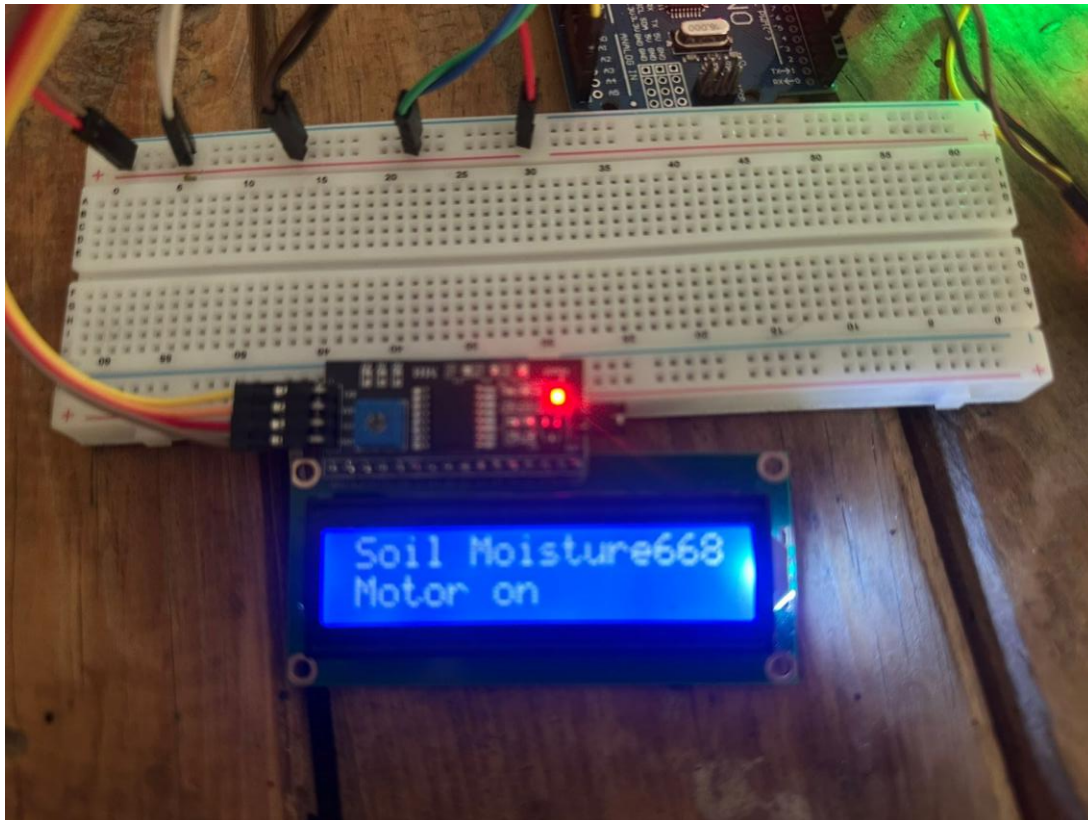


Figure 28soil moisture and motor status

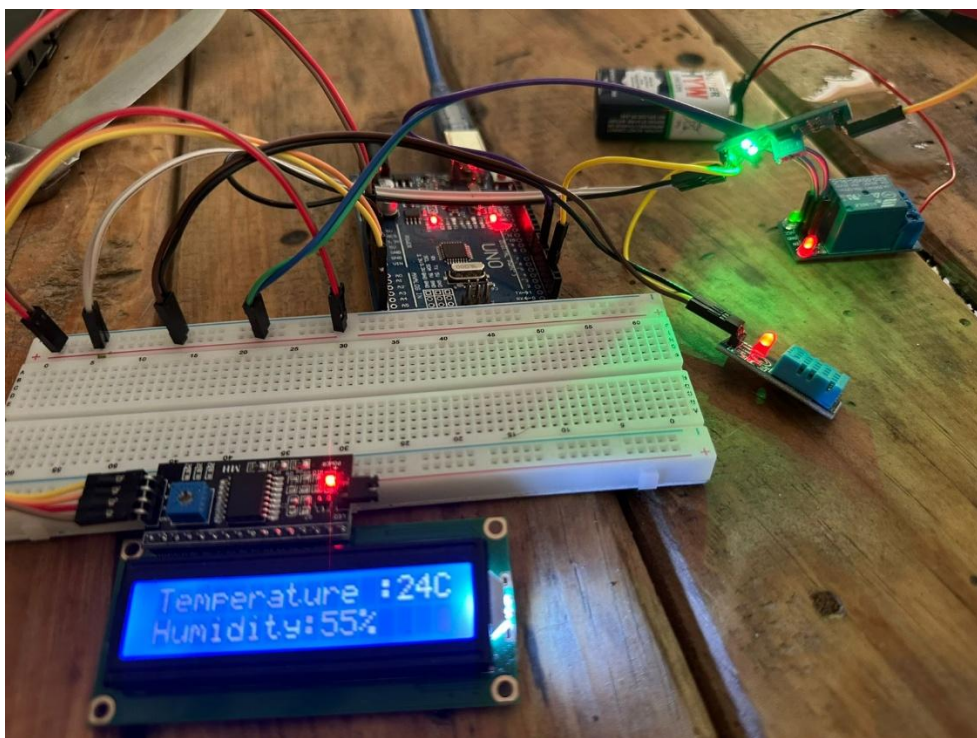


Figure 29Live update of temperature and humidity



Figure 30 Water pump was turned on

4.1.4. Test 4

Objective:	To check if the water pump turns off when the soil moisture is above the threshold value.
Action:	Source code was executed and observed the results.
Expected Result	The water pump should turn off, with changes to the led display message.
Actual Result	The water pump turned off, the led display message displayed the pump off and continuous monitoring of the temperature, humidity and moisture.
Conclusion	The test was successful

Table 4Test 4

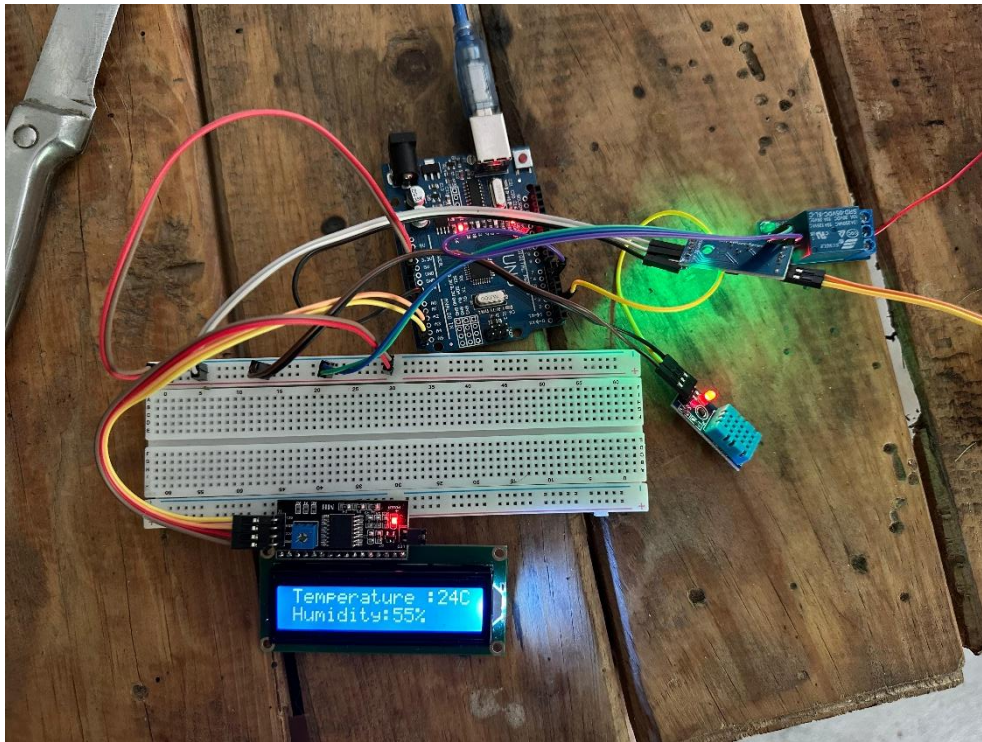


Figure 31 The motor was turned off after reaching threshold value

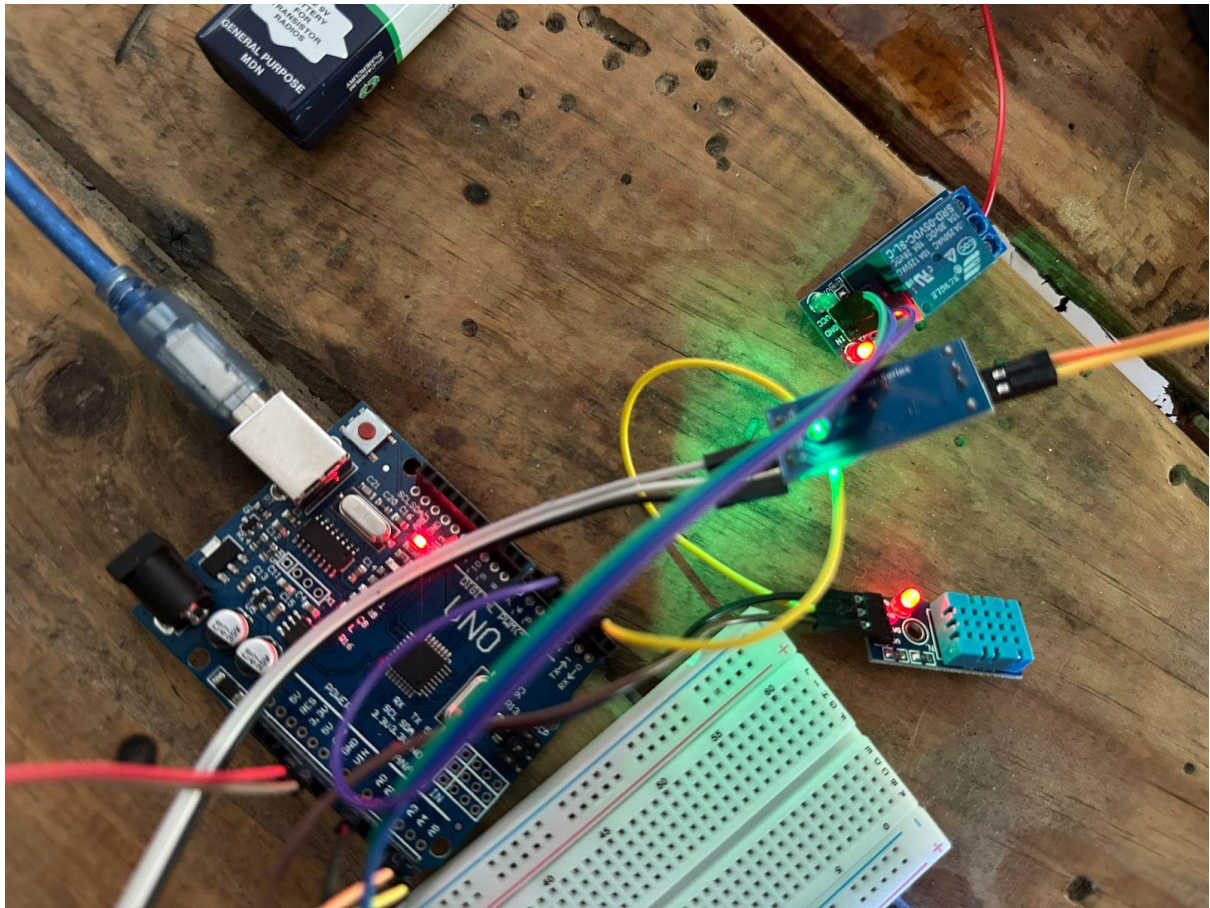


Figure 32 relay module was turned off



Figure 33 water pump was turned off

5. Future works

The smart irrigation system monitors temperature and soil moisture levels to operate automated watering which exclusively provides water to plants at needed times. The system functions individually at different locations but it does not connect wirelessly. The system can become more reachable and usable by adding wireless functionality through ESP8266 modules. A wireless connection added to the system via ESP8266 modules would enable real-time monitoring as well as control through either mobile apps or web-based dashboards. Users can access live alerts concerning moisture levels in addition to temperature or pump status from any location thus creating a more efficient interactive system. Continuous long-term operation of the current system faces potential problems because it depends solely on battery power. The implementation of a solar-powered version solves this issue since it supplies a sustainable and maintenance-free energy source. The system becomes more dependable after this upgrade particularly when operating in outdoor or power-independent areas. Weather forecasting through online APIs allows the system to automatically schedule watering to occur after rainfall. A forecast integration system with online APIs helps cut down excessive water consumption and maximizes irrigation system efficiency. The present design operates with small gardens in mind yet future iterations will be able to work with numerous sensors and water pumps for broader field applications. Through this update the system enables farmers to deliver water specifically to different areas of their land according to what their crops need and how their soils hold water.

6. Conclusion

The Smart Agriculture Monitoring and Irrigation System project has proven that it is possible to use IoT technologies to address real-life agriculture problems. By using components such as soil moisture sensors, temperature and humidity sensors (DHT11), a relay module, and an Arduino Uno, we developed a functional, low-cost, and automated irrigation system that can operate independently in small-scale agricultural environments. The system not only detects the levels of soil moisture, and turns on water pump only when necessary, it also optimizes consumption of water, saves human labor and ensures sustainable farming practices.

During the course of the testing phase, the system has reliably responded to environmental inputs as well as performed according to our expectation, thereby confirming that our designing and development approach worked. In addition to fulfilling its technical objectives, the project set a wide scope for implementation in the rural and resource-poor areas. In addition, it provides the base for further improvements such as wireless control, solar power usage, and multi-zone support of larger fields. Such solution complements current undertakings in precision agriculture, hence making smart farming attainable and affordable

.

7. References

Adafruit, 2012. *DHT11, DHT22 and AM2302 Sensors*. [Online]
Available at: <https://learn.adafruit.com/dht/overview>

AgriTechTomorrow, 2019. *Soil Moisture Sensors in Agriculture*. [Online]
Available at: <https://www.agritechtomorrow.com/article/2019/07/soil-moisture-sensors-in-agriculture/11534/>

Arduino, 2022. *What is Arduino?*. [Online]
Available at: <https://www.arduino.cc/en/Guide/Introduction>
[Accessed 6 April 2025].

Arduino, 2024. *Arduino IDE – Learn the Basics*. [Online]
Available at: <https://www.arduino.cc/en/software>

draw.io, 2025. *draw.io*. [Online]
Available at: <https://www.drawio.com/about>
[Accessed 2025].

IBM.com, 2023. *IBM*. [Online]
Available at: [https://www.ibm.com/think/topics/internet-of-things#:~:text=The%20Internet%20of%20Things%20\(IoT\)%20refers%20to%20a%20network%20of,to%20collect%20and%20share%20data.](https://www.ibm.com/think/topics/internet-of-things#:~:text=The%20Internet%20of%20Things%20(IoT)%20refers%20to%20a%20network%20of,to%20collect%20and%20share%20data.)
[Accessed 2025].

INECO, 2025. [Online]
Available at: <https://www.ingco.com/blog/what-is-water-pump#:~:text=A%20water%20pump%20works%20by,outlet%20to%20push%20it%20out.>

tinkercad.com, 2025. *tinkercad*. [Online]
Available at: <https://www.tinkercad.com/>
[Accessed 2025].

Walton, H. G., 2025. [Online]
Available at: <https://www.britannica.com/technology/liquid-crystal-display>

yasar, K. & Gillis, A. S., 2024. *techtarget*. [Online]
Available at: <https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT>
[Accessed 2025].

8. Appendix

8.1. Source code

```
//Importing Library

#include <LiquidCrystal.h>

#include <DHT.h>

#include <LiquidCrystal_I2C.h>

#include <Wire.h>


LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD address to 0x27 for a 16 chars and 2
line display


//Constants

#define DHTPIN 7

#define DHTTYPE DHT11

#define sensorPin A0 //Pin for Soil moisture sensor reading

#define motor 6 //Pin for relay module

DHT dht(DHTPIN, DHTTYPE);


//Variables

int h; //Stores humidity value
```

```
int t; //Stores temperature value
```

```
void setup()
```

```
{  
    Serial.begin(9600);  
    dht.begin();  
    pinMode(motor, OUTPUT);  
    digitalWrite(motor, HIGH);  
    lcd.init(); //initialize the lcd  
    lcd.backlight(); //open the backlight  
    lcd.clear();  
    lcd.setCursor(0, 0);  
    lcd.println("SmartIrrigation ");  
    lcd.setCursor(0, 1);  
    lcd.println("System");  
    delay(5000);  
}
```

```
void loop()
```

```
{  
    int sensorValue = analogRead(sensorPin);  
    //Read data and store it to variables h (humidity) and t (temperature)  
    h = dht.readHumidity();  
    t = dht.readTemperature();
```

```
// set the cursor to (0,0):

// print from 0 to 9:

lcd.clear();

lcd.setCursor(0, 0);

lcd.println("Temperature :");

lcd.setCursor(13, 0);

lcd.print(t);

lcd.print("C");

lcd.setCursor(0, 1);

lcd.print("Humidity:");

lcd.print(h);

lcd.print("%");

delay(2000); //Delay 1 sec.

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Soil Moisture");

lcd.print(sensorValue);

lcd.setCursor(0, 1);

if(sensorValue <= 500) // if Moisture level is Enough then cut the relay
{
    digitalWrite(motor,LOW); // low is to cut the relay

    lcd.print("Motor off");
}
```

```
else

{

    digitalWrite(motor,HIGH); //high to continue proving signal and water supply

    lcd.print("Motor on");

}

delay(2000); //Delay 2 sec.

}
```

8.2. Code for testing Dht11 sensor

// REQUIRES the following Arduino libraries:

// - DHT Sensor Library: <https://github.com/adafruit/DHT-sensor-library>

// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit_Sensor

```
#include "DHT.h"
```

```
#define DHTPIN 2 .
```

```
#define DHTTYPE DHT11 // DHT 11
```

```
DHT dht(DHTPIN, DHTTYPE);
```

```
void setup() {
```

```
    Serial.begin(9600);
```

```
    Serial.println(F("DHTxx test!"));
```

```
    dht.begin();
```

```
}
```

```
void loop() {
```

```
    // Wait a few seconds between measurements.
```

```
    delay(2000);
```

```
    // Reading temperature or humidity takes about 250 milliseconds!
```

```
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
```

```
    float h = dht.readHumidity();
```

```
    // Read temperature as Celsius (the default)
```

```
    float t = dht.readTemperature();
```

```
    // Check if any reads failed and exit early (to try again).
```

```
    if (isnan(h) || isnan(t) || isnan(f)) {
```

```
        Serial.println(F("Failed to read from DHT sensor!"));
```

```
        return;
```

```
    }
```

```
    Serial.print(F("Humidity: "));
```

```
    Serial.print(h);
```

```
    Serial.print(F("% Temperature: "));
```

```
    Serial.print(t);
```

```
    Serial.print(F("°C "));
```

```
    Serial.print(f);
```

```
}
```


8.3. Code for testing soil moisture sensors

```
const int sensorPin = A0; // Analog pin connected to the sensor

void setup() {

  Serial.begin(9600); // Initialize serial communication

}

void loop() {

  int sensorValue = analogRead(sensorPin); // Read the analog value from the sensor

  Serial.print("Soil Moisture Value: ");

  Serial.println(sensorValue); // Print the sensor value to the serial monitor

  delay(1000); // Delay for 1 second

}
```