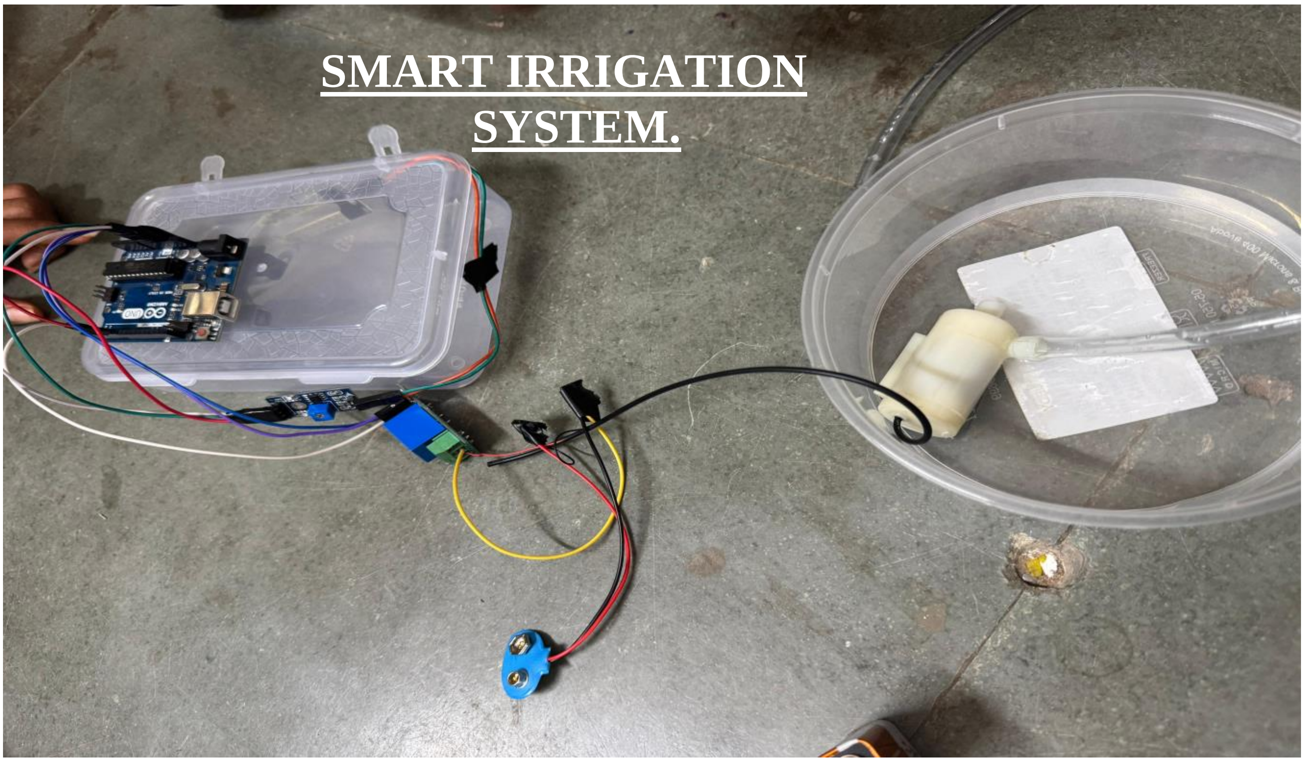
1. BUILDING A PROJECT PLAN FROM SCRATCH

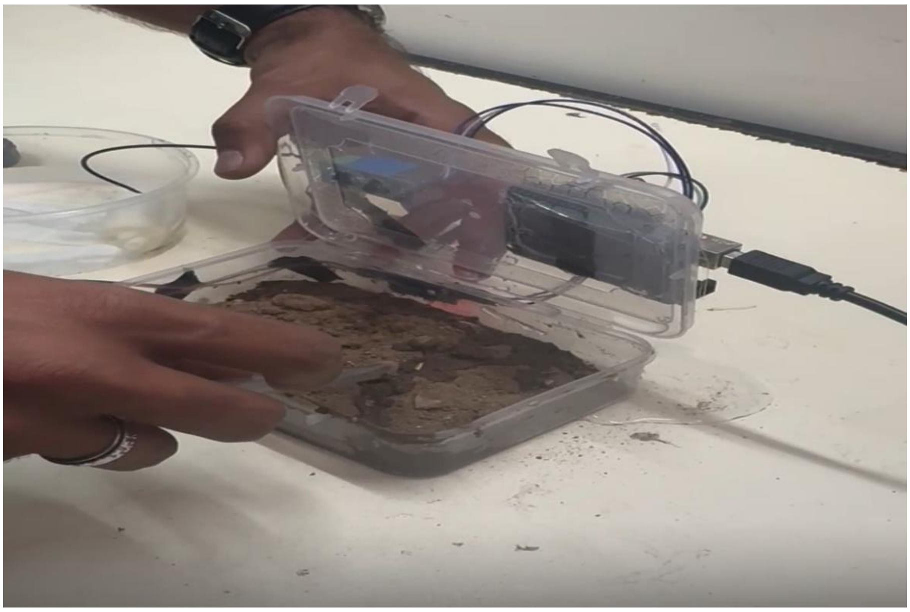
1.2) PROTOTYPE TITLE



1.2) PROJECT DEFINITION

- A smart irrigation system with Arduino refers to an automated watering system that utilizes Arduino, a microcontroller platform, to efficiently manage and control the irrigation process
- This system typically incorporates sensors to measure soil moisture levels, weather conditions, and other relevant parameters
- The Arduino processes this data to make informed decisions, activating or deactivating irrigation based on the specific needs of the plants
- The goal is to optimize water usage, conserve resources, and promote healthier plant growth
- The system can be integrated with IoT (Internet of Things) to enable remote monitoring and control via a mobile app or web interface.
- Solar power can be used to make the system energy-efficient and suitable for remote agricultural areas.
- The Arduino can be programmed to follow customizable schedules or respond dynamically to environmental changes.

- Additional sensors like temperature and humidity sensors can enhance decision-making for precise irrigation.



1.3) GROUP DETAILS

Team Members:-

1. Ayush Singh:

   Responsible for circuit design, component selection, and hardware assembly.

2. Parth Thakor:

   Responsible for circuit design, component selection, and hardware assembly.

3. Luhar Aastha:

   Responsible for circuit design, component selection, and hardware assembly.

4. Patel Keya:

   Developed the Arduino code for controlling soil moisture sensor.

5. Patel Aanshi:

   Developed the Arduino code for controlling soil moisture sensor.

1.4) SYSTEM REQUIREMENTS SPECIFICATION:

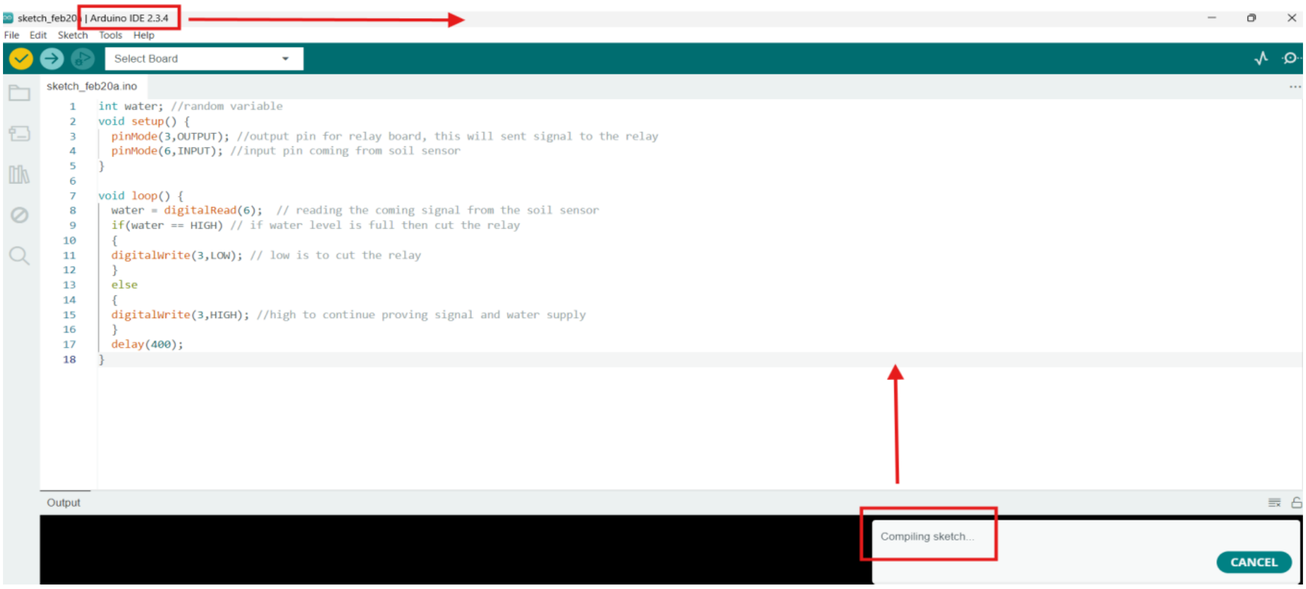
1. HARDWARE REQUIREMENTS

SR. NO	Category	Specification	Quantity
1.	Microcontroller (Control Unit)	Arduino Uno	1
2.	Sensor (Input Device)	Soil Moisture Sensor	1
3.	Switching Device (Control Device)	Relay Module	1
4.	Actuator (Output Device)	Water Pump	1
5.	Connecting Components (Wiring)	Jumper Wires	~10 - 12
6.	Power Device (Energy Source)	Battery	1
7.	Mechanical Component (Output Device)	Water Pipe	1

These Components Are Provided By Our College

2. SOFTWARE REQUIREMENTS

Used to WRITE AND UPLOAD code to the Arduino uno



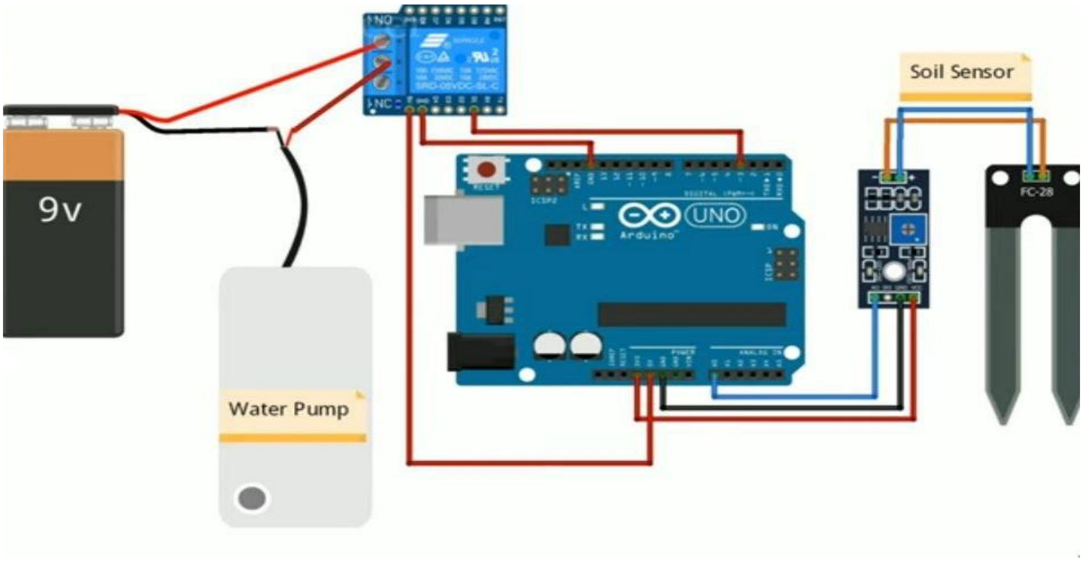
1.5) HARDWARE DEVICE COST SPECIFICATION

SR. NO	**Category**	**Specification**	**Quantity**	**Cost**
1.	Microcontroller (Control Unit)	Arduino Uno	1	400
2.	Sensor (Input Device)	Soil Moisture Sensor	1	150
3.	Switching Device (Control Device)	Relay Module	1	100
4.	Actuator (Output Device)	Water Pump	1	100
5.	Connecting Components (Wiring)	Jumper Wires	~10 – 12	100 (pack of 10)
6.	Power Device (Energy Source)	Battery	1	40
7.	Mechanical Component (Output Device)	Water Pipe	1	60

These Components Are Provided By Our College

2. ASSEMBLING AND PROGRAMMING IN VIRTUAL ENVIRONMENT

2.1) CIRCUIT DESIGN – DIAGRAM



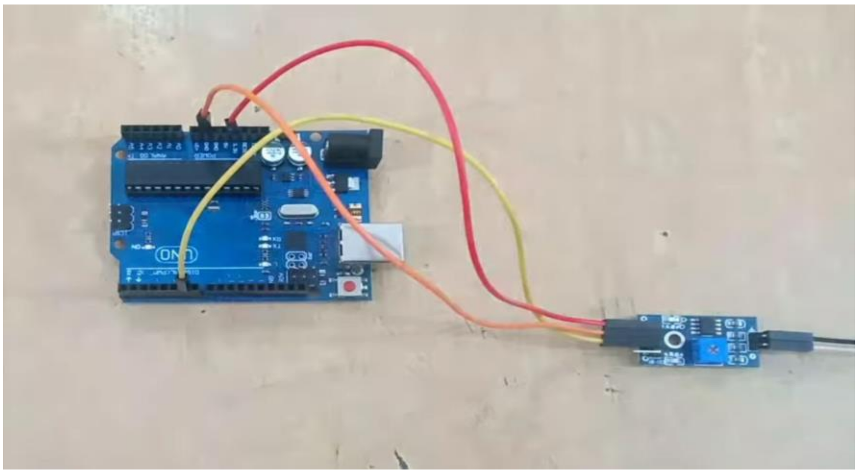
2.2) ASSEMBLING THE COMPONENTS – STEPS

Step:1) Connect the Soil Moisture Sensor to Arduino

VCC → Connect to 3.3V on Arduino

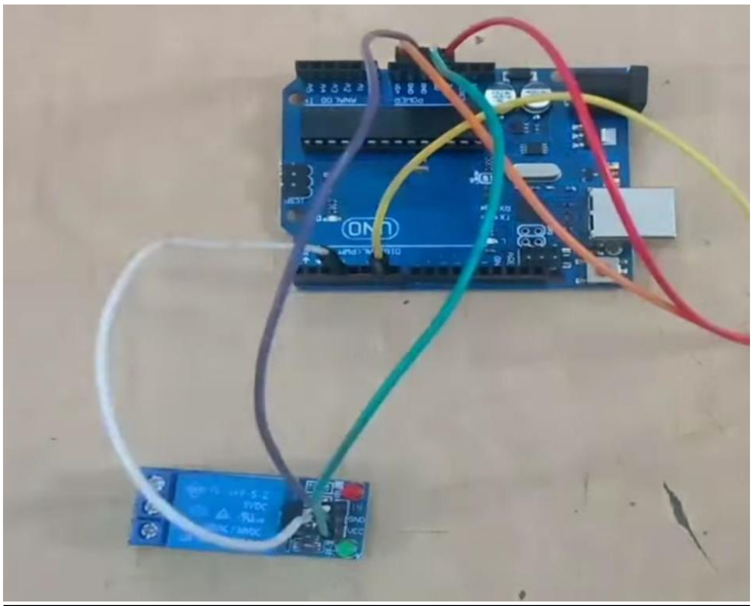
GND → Connect to GND on Arduino

A0 (Analog Output) → Connect to Digital 6 on Arduino



Step:2) Connect the Relay Module to Arduino

- The relay module acts as a switch for the water pump.
- VCC → Connect to 5V on Arduino
- GND → Connect to GND on Arduino
- D0 IN (Signal Pin) → Connect to Digital Pin 3 on Arduino



Step:3) Connect the Water Pump to the Relay Module

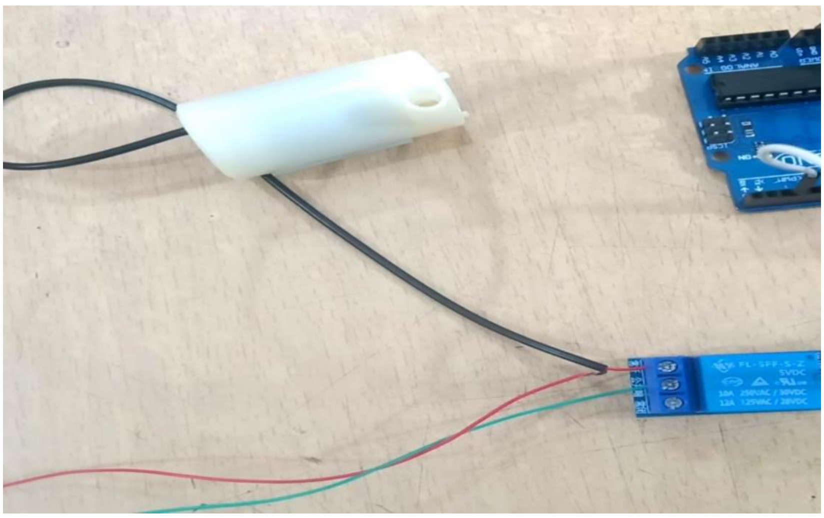
The relay module has three output terminals (NO, COM, NC).

Connections:

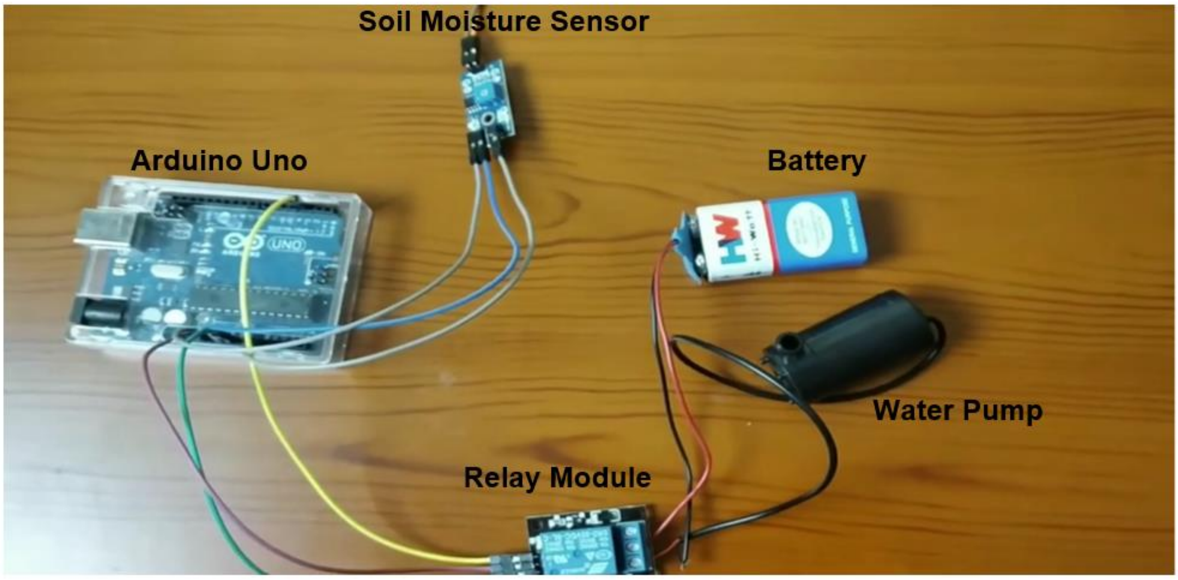
COM (Common Terminal) → Connect to the positive terminal of the water pump.

NO (Normally Open Terminal) → Connect to the positive terminal of the battery.

Negative terminal of the pump → Connect directly to the battery's negative terminal



2.3) LABELLED DIAGRAM OF PROTOTYPE



2.4) DEVELOPING THE LOGIC: COMPLETE CODE WITH EXPLAINATION

```
int water;      // Variable to store soil moisture sensor readings
void setup() {
        pinMode(3, OUTPUT);     // Pin 3 is connected to the relay (controls water pump)
        pinMode(6, INPUT);      // Pin 6 is connected to the soil moisture sensor
}
void loop() {
        water = digitalRead(6);       // Read the soil moisture sensor value (HIGH = Wet, LOW =
Dry)

                if (water == HIGH) {      // If soil is wet, turn OFF the pump
                    digitalWrite(3, LOW);      // Deactivate relay (Pump OFF)
 }
                else { // If soil is dry, turn ON the pump
                    digitalWrite(3, HIGH);      // Activate relay (Pump ON)
  }
      delay(400);      // Short delay to prevent rapid switching
}
```

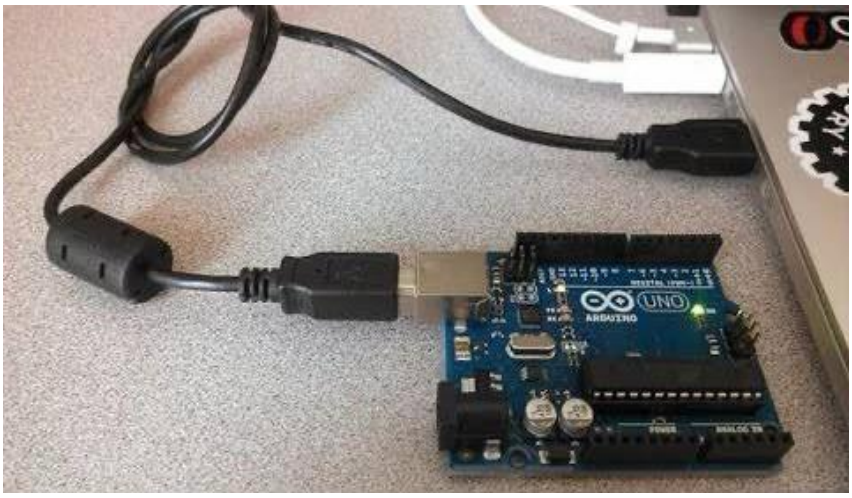
EXPLAINATION

- This Arduino-based **smart irrigation system** is designed to **automate the watering process**, ensuring efficient water usage and reducing manual intervention. The system utilizes a **soil moisture sensor** to continuously monitor the moisture levels in the soil and a **relay module** to control a **water pump** accordingly. The **Arduino Uno microcontroller** processes the sensor's readings and makes real-time decisions to turn the pump **ON or OFF** based on soil conditions.

- When the **sensor detects sufficient moisture** in the soil (**HIGH signal**), the Arduino **turns OFF** the relay, stopping the water pump to prevent overwatering. Conversely, when the **soil is dry** (**LOW signal**), the Arduino **activates the relay**, switching **ON** the water pump to irrigate the plants. The system includes a **400-millisecond delay** to avoid rapid relay switching, which helps maintain stable operation and prevents hardware wear.
- This automated system offers several advantages, including **water conservation, reduced human effort, and improved plant health** by providing the right amount of water at the right time. It is a **cost-effective and energy-efficient solution** for **small-scale farming, home gardens, and greenhouses**. Future enhancements could include **IoT integration for remote monitoring, weather-based automation, and solar power integration** to further improve sustainability. This **smart irrigation system** demonstrates how simple automation can contribute to **precision agriculture and resource optimization**.

2.5) BURNING THE CODE INTO THE PROTOTYPE: STEPS

- Connect Arduino to Computer using a USB cable (Type A to Type B).
- Open Arduino IDE and ensure it is installed.
- Select Board: Go to Tools > Board > Arduino Uno.
- Choose the Correct Port: Tools > Port > Select COM Port (e.g., COM3).
- Paste/Write the Code in Arduino IDE.
- Verify the Code (Click ✅ to check for errors).
- Upload the Code (Click ➞ to burn the code into Arduino).
- Now the Smart Irrigation System is programmed and ready to work!



3 DEMONSTRATING THE PROTOTYPE

3.1) PROTOTYPE ADVANTAGES

- **Water Conservation** – The system optimizes irrigation by using real-time soil moisture data, ensuring water is used efficiently. This prevents unnecessary watering and helps in areas facing water scarcity, leading to better resource management.
- **Automation** – The entire process is self-operating, eliminating the need for manual watering. This saves time, effort, and resources, making it ideal for busy farmers and gardeners while improving productivity.
- **Cost-Effective** – Uses affordable electronic components like Arduino, sensors, and relays, making it a budget-friendly solution for small-scale farms and home gardens. Compared to traditional irrigation, it reduces long-term water and labor costs.
- **Efficient Plant Growth** – Ensures optimal moisture levels, leading to healthier plants and higher crop yields. Prevents under-watering, which can stunt growth, and overwatering, which can damage roots, ensuring better soil conditions.
- **User-Friendly** – The system requires minimal technical knowledge to set up and operate. Once installed, it runs automatically, requiring only occasional maintenance, making it easy for farmers of all skill levels.
- **Reduces Labor Costs** – By automating irrigation, it eliminates the need for manual supervision, allowing farmers and gardeners to focus on other important tasks. This reduces the dependency on labor, lowering operational expenses.
- **Adaptable to Various Environments** – The system is suitable for home gardens, greenhouses, commercial farms, and even urban landscaping, making it a versatile irrigation solution. It can be modified to work with different soil and plant types.
- **Prevents Overwatering** – By supplying only the required amount of water, it reduces the risk of root rot, fungal infections, and soil erosion, leading to a healthier ecosystem. This prevents excess moisture that could attract pests or cause mold growth.
- **Energy Efficient** – Can be integrated with solar panels, making it a sustainable solution that reduces reliance on conventional electricity and lowers energy costs. This is ideal for remote areas where power supply is limited.
- **Customizable Settings** – Users can set specific irrigation schedules and adjust moisture thresholds based on different plant types and environmental conditions for better control. This ensures personalized irrigation for different crops and soil types.

- **Scalable System** – The system can be expanded with additional sensors and pumps to cover larger farms, multiple garden sections, or commercial agricultural fields. It can be integrated with IoT for real-time data monitoring and control.
- **Compatible with Different Water Sources** – Can be connected to borewells, storage tanks, rainwater harvesting systems, or direct water supply lines, ensuring flexibility in various setups. This allows efficient water management even in drought-prone areas.

3.2) PROTOTYPE CHALLENGES

- **Sensor Inaccuracy** :- Soil moisture sensors may sometimes provide inconsistent or incorrect readings due to environmental factors such as temperature fluctuations, soil composition, or sensor aging. Calibration and regular testing are required to maintain accuracy, but this can be time-consuming and requires technical expertise. Additionally, low-quality sensors may degrade over time, affecting performance.
- **Power Dependency** :- The system relies on a stable power supply to operate efficiently. In rural or remote areas where electricity supply is unreliable, maintaining continuous operation can be challenging. Alternative power sources, such as solar panels or backup batteries, may be required to ensure uninterrupted functionality. Without a consistent power source, the system may fail to activate the water pump when needed, leading to crop damage.
- **Weather Sensitivity** :- The system does not integrate real-time weather data, meaning it could still activate irrigation even if it rains. This results in unnecessary water usage and could lead to overwatering. A more advanced system could include rain sensors or fetch weather forecast data

from the internet to prevent irrigation when rainfall is expected.

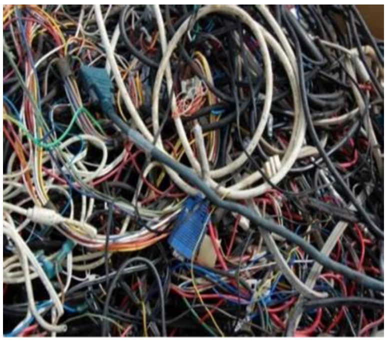
- **Scaling Costs** :- While affordable for small-scale gardens or farms, expanding the system to cover large agricultural fields increases costs significantly. Additional sensors, pumps, and long-range communication modules (such as LoRa or GSM) are required for large farms, making the system more expensive. The cost of high-quality sensors, durable pipes, and smart controllers can also add to the expenses.
- **Maintenance Needs** :- Regular cleaning, calibration, and troubleshooting are essential to keep the system running efficiently.
  1. Soil moisture sensors can accumulate dirt or mineral deposits, affecting accuracy.
  2. The water pump and pipes may clog due to dirt or algae buildup, leading to reduced water flow.
  3. The relay module or wiring connections may degrade over time, requiring replacements. A proper maintenance plan is necessary to ensure long-term reliability.
- **Initial Setup Complexity** :- Installing and configuring the system requires technical knowledge, especially for beginners.
  1. Understanding wiring connections, circuit diagrams, and Arduino programming can be challenging for non-experts.
  2. Setting the correct moisture thresholds in the code requires trial and error to match different soil types and crops.
  3. Integration with IoT or cloud services for remote monitoring demands additional expertise in networking and data management.
- **Limited Battery Life (for Wireless Sensors)** :- If the system includes wireless soil moisture sensors, they require a battery-powered setup.
  1. Batteries need frequent replacement or recharging, which can be inconvenient for remote areas.
  2. A solar-powered charging system could be a possible solution, but it adds extra cost and setup complexity.
- **Risk of System Failure** :- A malfunction in any key component (e.g., water pump, valve, Arduino, relay module, or sensor) can disrupt the entire irrigation process.
  1. If the pump fails, plants may not receive water, leading to crop damage.
  2. A faulty relay module might prevent the pump from switching ON/OFF at the right time.
  3. Wiring issues or Arduino crashes could cause system failure.

3.3) PROTOTYPE REAL LIFE APPLICATIONS

- **Agriculture:** Efficient water management for crops, reducing water waste, and improving yield through precise irrigation based on soil moisture and weather conditions.
- **Gardens and Lawns:** Automating watering schedules for home gardens or public parks, ensuring plants are watered as needed, preventing over- or under-watering.
- **Greenhouses:** Monitoring and controlling irrigation in controlled environments to maintain optimal moisture levels for plant growth.
- **Golf Courses:** Managing large green spaces efficiently by using smart irrigation to maintain turf health while conserving water.
- **Commercial Landscaping:** Optimizing water usage for landscapes around office buildings, hotels, or other commercial properties, ensuring greenery stays healthy with minimal water use.
- **Smart Cities:** Part of broader smart city initiatives, where public parks and urban green spaces use IoT-based smart irrigation to conserve water and maintain urban ecosystems.

3.4) ROBOPEDIA PROJECT WORK: I TO IV FIVE LEARNINGS

- **Understanding Automation & Robotics** – Learned how Arduino and sensors automate processes.
- **Circuit Design & Hardware Integration** – Gained hands-on experience in connecting and troubleshooting electronic components.
- **Programming & Code Implementation** – Developed coding skills in C/C++ for Arduino-based projects.
- **Problem-Solving & Troubleshooting** – Identified and resolved errors in hardware connections and software logic.
- **Project Planning & Execution** – Learned to design, test, and optimize a working prototype.



3.5) PROJECT WORK CONCLUSION

- **Successful Implementation :-** The smart irrigation system prototype was successfully developed and tested. The integration of **Arduino Uno, soil moisture sensor, relay module, and water pump** demonstrated effective automation in managing irrigation. The system responded accurately to moisture levels, ensuring water was provided when needed and stopped when the soil was adequately moist.
- **Automation Achieved :-**The project successfully implemented **automation** by utilizing sensors and a microcontroller to eliminate manual intervention. The system intelligently detects **soil moisture levels** and activates the **water pump** accordingly, ensuring plants receive adequate water without human input. This highlights the potential for **precision agriculture**, reducing human effort and optimizing irrigation schedules.
- **Water Conservation :-**One of the key advantages of this project is its ability to **conserve water** by using real-time moisture level detection. Traditional irrigation methods often lead to **overwatering**, resulting in unnecessary water wastage. With the **sensor-based automation**, the system only waters plants when necessary, **minimizing waste** and contributing to sustainable agricultural practices.
- **Improved Efficiency:-** By eliminating **manual irrigation**, this system significantly improves efficiency. Farmers and gardeners no longer need to **monitor soil moisture manually** or operate water pumps manually. The system ensures that crops receive the right amount of water at the right time, reducing **human error and labor costs**. Additionally, the system's quick response time allows for **precise irrigation**, leading to **better plant growth and higher yields**.
- **Future Scope:-** While this prototype successfully automates irrigation, future enhancements could make it even more effective:

1. **IoT Integration:** Adding **Wi-Fi or GSM modules** would allow for **remote monitoring and control** through a smartphone or web application.
2. **Weather-Based Automation:** Integrating a **rain sensor or weather API** could further optimize water usage by preventing irrigation during rainfall.
3. **Mobile Control:** Developing a **mobile app** to manually override or adjust watering schedules based on real-time conditions.
4. **Solar Power Integration:** Making the system **self-sustainable** by incorporating **solar panels** for power supply.
5. **Nutrient Monitoring:** Expanding the system to include **soil nutrient sensors**, enabling **smart fertigation** (automated fertilization with irrigation).

3.6) REFERENCES

https://www.raypcb.com/smart-irrigation-system-using-arduino/#:~:text=Smart%20irrigation%20systems%20save%20about,amount%20of%20water%20it%20saves.

https://circuitdigest.com/microcontroller-projects/interfacing-soil-moisture-sensor-with-arduino-uno

https://www.renkeer.com/smart-irrigation-technology-and-system/

https://dipslab.com/smart-irrigation-system/

https://www.linkedin.com/pulse/future-smart-irrigation-mobitech-wireless-solution