

# Automated Irrigation System for Rooftop Gardening: Design, Implementation, and Sustainability Analysis

Md Selim Ahmed

Department of Computer Science and Engineering  
University of Liberal Arts Bangladesh  
Dhaka, Bangladesh  
selim.ahmed.cse@ulab.edu.bd

## Abstract

Rooftop gardening in densely populated urban areas presents a unique set of challenges, primarily revolving around the maintenance of optimal soil moisture amidst limited space, erratic weather patterns, and increasing water scarcity. In cities like Dhaka, where horizontal land is scarce, vertical and rooftop agriculture is becoming essential for food security and environmental cooling. However, manual irrigation is labor-intensive and prone to inefficiency. This project addresses these issues by developing a robust, low-cost Automated Irrigation System driven by embedded technology. The system architecture integrates an Arduino Uno microcontroller (ATmega328P) as the central processing unit, a submersible DC water pump, a 16x2 I2C LCD display for user feedback, a relay module with optical isolation for safe actuation, and a resistive soil moisture sensor for real-time environmental monitoring. The system operates on a continuous feedback loop, utilizing the microcontroller's 10-bit Analog-to-Digital Converter (ADC) to monitor soil resistivity. A specific control algorithm triggers the irrigation mechanism when moisture levels drop below a calibrated threshold and deactivates it once saturation is reached, effectively preventing both water stress (wilting) and root rot. To ensure environmental sustainability and reliability in off-grid scenarios, the system is powered by a 6V solar photovoltaic panel coupled with a TP4056 lithium-ion battery charging module, creating a self-sustaining energy cycle. This report provides a comprehensive, in-depth analysis of the hardware design, theoretical frameworks, control logic, power consumption metrics, and the system's strategic alignment with the United Nations Sustainable Development Goals (SDGs), establishing it as a scalable, viable solution for modern urban agriculture.

**Keywords:** Precision Agriculture, Arduino Uno, Soil Moisture Sensor, Solar Energy, Rooftop Gardening, Relay Module, Sustainable Development, Urban Farming, IoT.

## 1 Introduction

Agriculture has historically been the foundation of human civilization, enabling the growth of societies and economies. However, the rapid pace of modern urbanization has physically and conceptually distanced food production from population centers. In megacities like Dhaka, the "Concrete Jungle" effect—characterized by the replacement of vegetation with as-

phalt and concrete—has led to severe environmental degradation, including the Urban Heat Island (UHI) effect, reduced air quality, and a lack of connection to nature.

Rooftop gardening has emerged as a vital, multi-faceted solution to these urban problems. It offers a pathway to household food security by allowing families to grow fresh produce, contributes to temperature regulation by insulating buildings, and improves air quality through carbon sequestration. Despite these benefits, the adoption of rooftop gardening is often hindered by the maintenance burden. The primary challenge is irrigation; plants in containers on rooftops are exposed to harsher elements (wind, direct sun) and dry out faster than plants in the ground. Manual maintenance is labor-intensive, time-consuming, and often inefficient due to human error.

An automated irrigation system bridges the gap between the desire for urban green spaces and the practical constraints of urban living. By leveraging embedded systems and sensor technology, we can transition from traditional "schedule-based" irrigation—which waters plants regardless of need, often leading to wastage—to "need-based" precision irrigation. This system utilizes real-time soil analysis to deliver water only when necessary, optimizing resource usage and ensuring plant health.

Furthermore, the integration of renewable energy is a critical innovation in this project. By utilizing a solar panel to charge a Li-ion battery, the system operates independently of the municipal power grid. This not only reduces the operational carbon footprint but also ensures functionality during power outages or "load shedding," which is a common occurrence in many developing regions. This report details the design and implementation of such a system, proving that high-tech agricultural solutions can be accessible and affordable.

### 1.1 Background and Motivation

The motivation for this project stems from the intersection of two critical global crises: water scarcity and the urban environmental crisis.

**Water Scarcity:** Agriculture consumes approximately 70% of the world's accessible freshwater. Traditional irrigation methods, such as flood or hose watering, are estimated to waste up to 50% of this water due to runoff, evaporation, and over-watering. In water-stressed urban areas, where municipal water is costly and finite, this inefficiency is environmentally and economically unsustainable.

**Urban Heat Island Effect:** As cities expand, green cover di-

minishes, leading to higher ambient temperatures. Rooftop gardens absorb solar radiation, significantly cooling buildings and reducing the energy load on air conditioning systems. However, maintaining these green spaces requires consistent water. Busy urbanites often fail to provide this consistency, leading to withered gardens that fail to provide cooling benefits.

This project aims to democratize "Precision Agriculture"—a concept usually reserved for large industrial farms—bringing it to the micro-scale of a household rooftop. By automating the most critical aspect of plant care, we motivate more citizens to participate in urban greening initiatives.

## 1.2 Problem Statement

Manual irrigation poses several distinct technical and practical problems that this project aims to solve:

1. **Inconsistency and Human Error:** Humans lack the ability to accurately judge soil moisture at root depth. Gardeners often water the surface while roots remain dry, or water too frequently, leading to fungal diseases and root rot.
2. **Resource Inefficiency:** "Blanket watering" or watering on a fixed schedule consumes significantly more water and energy than necessary.
3. **Plant Mortality:** Neglect during travel, holidays, or busy work periods frequently results in high plant mortality rates, discouraging people from gardening.
4. **Energy Dependency:** Most existing automated pumps rely on AC grid power. Rooftops often lack convenient outdoor power outlets, making extension cords a safety hazard.
5. **Labor Intensity:** The physical effort required to carry water to rooftops is a barrier for the elderly and differently-abled individuals.

## 1.3 Objectives

The primary objective is to develop a "Install-and-Forget" automated system that is robust, low-cost, and energy-independent. Specific technical objectives include:

- **Precision Sensing:** To utilize the Arduino's 10-bit ADC to detect minute changes in soil conductivity, allowing for precise control over soil moisture conditions.
- **Hysteresis Control:** To implement a robust software algorithm that prevents "pump chatter" (rapid switching) when moisture levels hover near the threshold.
- **Energy Independence:** To achieve a self-sustaining power cycle using solar energy, eliminating the need for battery replacement or grid connection.
- **User Feedback:** To provide real-time diagnostic data via an LCD interface, informing the user of the system status and soil health.
- **Scalability:** To design a system that can be easily upgraded with more sensors or larger pumps for bigger gardens.

## 2 Literature Review

To establish the context for this project, several existing studies and technologies were reviewed. The evolution of automated irrigation has moved from simple timers to complex IoT-based systems.

[1] **Krishnamoorthy et al. (2014):** This study proposed a microcontroller-based drip irrigation system. Their work focused on using sensors for both soil moisture and humidity to optimize crop yield. While effective in a farm setting, their system relied on standard AC power supplies and complex solenoid valves, limiting its application in remote or off-grid rooftop scenarios where plumbing pressure might be low. Our project addresses this by using a DC submersible pump and solar power.

[2] **Rangel et al. (2024):** This recent paper introduced irrigation systems utilizing "intelligent agents" for arid regions. Their work focused on complex algorithms and high-level computing to manage water resources. However, the complexity and cost of such systems make them inaccessible for the average urban hobbyist. Our approach prioritizes cost-effectiveness using standard, available components like the Arduino Uno to ensure accessibility for average households in developing nations.

[3] **Hamoodi et al.:** Discussed Photovoltaic (PV) pumping systems for irrigation. This study validated the feasibility of using solar power directly for water pumping. However, direct solar pumping stops when clouds pass. Our project improves on this by integrating a battery buffer (TP4056 + Li-ion), ensuring the system can operate during cloudy periods or even at night if necessary.

[4] **Singla et al. (2019):** Explored IoT-based irrigation using ESP8266/NodeMCU. While IoT offers remote monitoring via smartphones, it requires constant internet connectivity and consumes significantly more power for Wi-Fi transmission. For a basic rooftop garden, high reliability is preferred over connectivity. Our standalone embedded approach offers higher reliability and lower power drain, ensuring the garden survives even if the home Wi-Fi goes down.

[5] **Sy et al. (2020):** Developed a low-cost Arduino-based system that included pressure tank checks. Their work emphasized cost-reduction strategies, which we have adopted. However, their system lacked a visual interface for the user. We have integrated an I2C LCD to provide immediate visual feedback, which is crucial for user trust and system debugging.

## 3 Theoretical Framework

Before detailing the hardware, it is essential to understand the physical principles governing the system's operation.

### 3.1 Soil Conductivity Principle

The core detection mechanism relies on the electrical properties of soil. Dry soil is largely composed of silica and air spaces, both of which are electrical insulators. Water, especially tap water containing dissolved minerals (ions), is a conductor.

$$V_{out} = V_{in} \times \frac{R_{soil}}{R_{soil} + R_{pullup}} \quad (1)$$

The sensor acts as a variable resistor ( $R_{soil}$ ). As moisture content increases,  $R_{soil}$  decreases. The sensor module creates a

voltage divider circuit. The Arduino measures the voltage drop across this network. High moisture results in lower resistance, which the module converts to a lower analog voltage signal (inverted by the module's comparator logic in some versions, or read directly).

### 3.2 Photovoltaic Effect

The power system relies on the photovoltaic effect, where photons from sunlight strike the semiconductor material in the solar panel, knocking electrons loose and creating an electric current. The specific panel used (6V, 1W) produces voltage based on solar irradiance. Since the Arduino requires a stable 5V, and the battery operates at 3.7V-4.2V, a charging controller (TP4056) is necessary to regulate this fluctuating solar input into a safe charging current for the Lithium-Ion chemistry.

### 3.3 Electromagnetic Switching

The Arduino's GPIO pins can only supply 40mA of current at 5V. The water pump requires 300mA-500mA. Direct connection would destroy the microcontroller. A relay uses a small current to energize an electromagnet, which physically pulls a metal armature to close a high-current circuit. This provides "Galvanic Isolation," meaning there is no direct electrical path between the high-power pump loop and the sensitive logic loop, protecting the Arduino from voltage spikes (back-EMF) generated by the pump motor.

## 4 Methodology

The system design follows a modular architecture, ensuring that each subsystem (Sensing, Processing, Actuation, Power) can be tested and upgraded independently.

### 4.1 Hardware Requirements & Specifications

#### 4.1.1 1. Arduino Uno R3 (Microcontroller)

The brain of the system is the Arduino Uno.

- **Processor:** ATmega328P (8-bit AVR architecture).
- **Clock Speed:** 16 MHz, providing ample speed for sampling sensor data.
- **ADC Resolution:** 10-bit. This allows the system to read analog values from 0 to 1023, providing 1024 distinct levels of moisture resolution. This is far more precise than a simple digital (High/Low) sensor.
- **Operating Voltage:** 5V DC, compatible with standard USB power banks and solar regulators.

#### 4.1.2 2. Resistive Soil Moisture Sensor

- **Type:** YL-69 or similar fork-type resistive probe.
- **Mechanism:** Two exposed nickel-plated probes are inserted into the soil. A comparator chip (LM393) on the module allows for both digital (threshold) and analog (raw) output. We utilize the analog output for software-based thresholding.
- **Corrosion Note:** A known limitation of resistive sensors is electrolysis (corrosion of probes) when current flows

continuously. To mitigate this, future code iterations can power the sensor only during the reading instant.

#### 4.1.3 3. Relay Module (1-Channel, 5V)

- **Logic:** Active Low or Active High (configurable).
- **Isolation:** Features an optocoupler (e.g., PC817) which uses light to transmit the signal internally. This provides a second layer of safety alongside the electromagnetic isolation.
- **Switching Capacity:** Rated for 10A at 250V AC or 30V DC, which is more than sufficient for our 5V/0.5A DC pump.

#### 4.1.4 4. Power System (Solar & Battery)

- **Solar Panel:** 6V, 160mA (approx 1W). Polycrystalline cells were chosen for their cost-effectiveness.
- **Battery:** 18650 Li-ion cell (3.7V, 2000mAh). These batteries have high energy density and no memory effect.
- **TP4056 Module:** This is a complete constant-current/constant-voltage linear charger for single-cell lithium-ion batteries. It features:
  - Over-charge protection (cuts off at 4.2V).
  - Over-discharge protection (cuts off at 2.5V).
  - Short-circuit protection.

#### 4.1.5 5. Actuator (Mini Water Pump)

- **Type:** Submersible DC centrifugal pump.
- **Specs:** 3V-5V DC, low noise.
- **Flow Rate:** 100-120 Liters/Hour. Ideally suited for drip irrigation tubing in a small rooftop setup.

### 4.2 System Architecture

The system architecture works as a linear control pipeline:

1. **Input Stage:** The solar panel charges the battery. The battery powers the Arduino. The Soil Sensor reads the environmental state.
2. **Processing Stage:** The Arduino converts the analog voltage to a digital integer (0-1023). It compares this integer against predefined constants defined in the software (the "Dry" and "Soaked" limits).
3. **Output Stage:** Based on the comparison:
  - **Display:** The LCD updates the text.
  - **Actuation:** The Digital GPIO pin triggers the relay to open or close the pump circuit.

## 5 Implementation

### 5.1 Circuit Connections

To ensure robust operation, the following pin mapping and connection strategy was implemented. Great care was taken to separate power lines from signal lines to reduce noise.

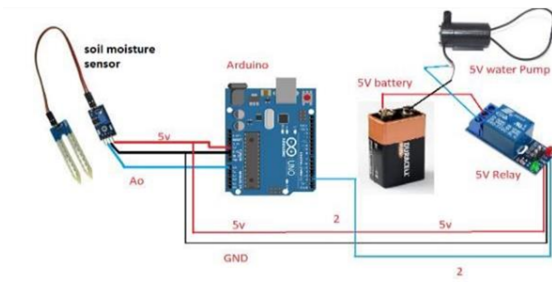


Figure 1: System Architecture: Interconnection of Sensors and Actuators

- **Sensor Data:** Connected to Analog Pin **A0**. VCC connected to Arduino 5V, GND to GND.
- **Relay Signal:** Connected to Digital Pin **D2**. The Relay VCC goes to 5V, GND to GND. The Pump's positive wire runs through the Relay's COM (Common) and NO (Normally Open) terminals. This ensures the pump is OFF if the system loses power (Fail-Safe).
- **LCD I2C:** Connected to **A4 (SDA)** and **A5 (SCL)**. This uses the standard I2C bus of the ATmega328P.
- **Power Distribution:**
  - **Solar:** Connected to the input pads of the TP4056.
  - **Battery:** Connected to the B+ and B- pads of the TP4056.
  - **Load:** The OUT+ and OUT- pads of the TP4056 power the Arduino via the 5V/GND pins or USB port. This ensures the load is disconnected if the battery voltage drops too low, protecting the battery health.

## 5.2 Software Logic and Hysteresis

The software logic is written in C++ using the Arduino IDE. A critical feature of the implementation is **Hysteresis**. If we used a single threshold (e.g., 50%), the moisture level would fluctuate rapidly around that point as the pump turns on, causing the pump to stutter on and off (oscillate). To prevent this, we implemented a "Dead Band":

- **Turn-ON Threshold:** The pump activates only when moisture drops below 30% (Significant dryness).
- **Turn-OFF Threshold:** The pump deactivates only when moisture exceeds 80% (Saturation).

The gap between 30% and 80% is the hysteresis band where the state does not change. This ensures the pump runs for a solid duration to deeply water the roots, then stays off for a long period until the soil is truly dry again.

## 5.3 Prototype States

The following figures illustrate the system in various operational states during the testing phase.

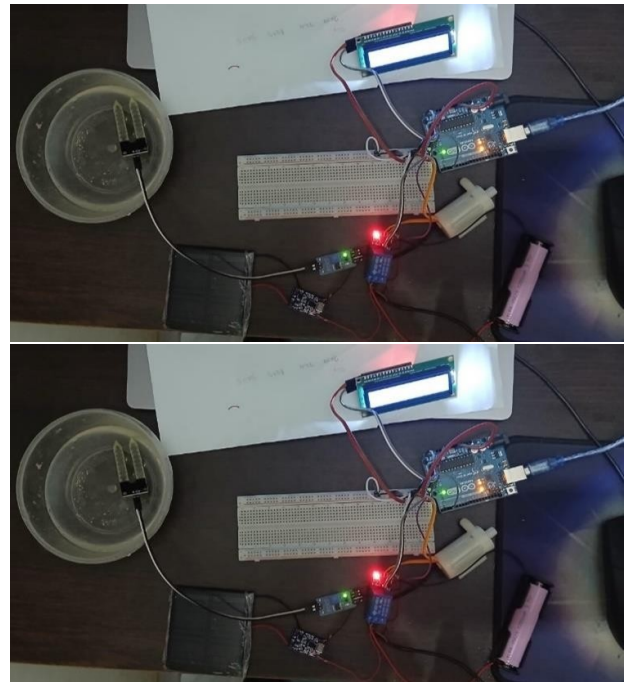


Figure 2: System Standby Mode (Pump OFF). The LCD indicates moisture is sufficient.

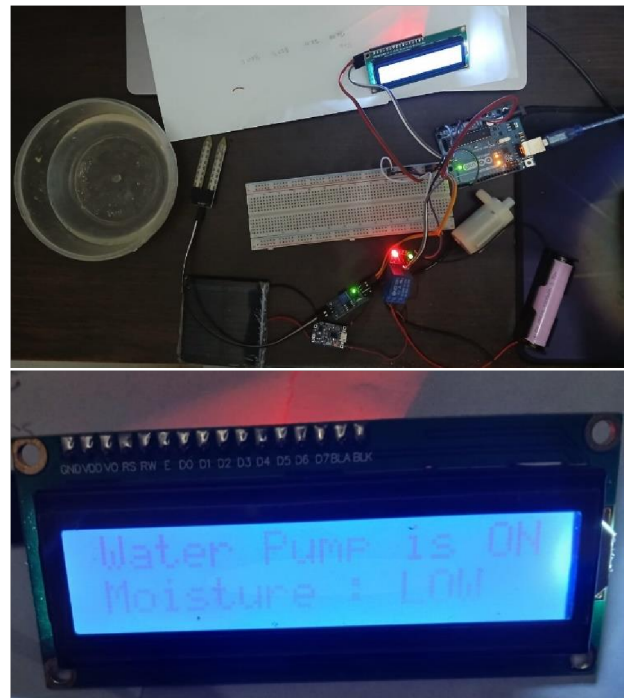


Figure 3: Active Irrigation Mode (Pump ON). The LCD indicates low moisture levels.

# 6 System Performance Analysis

To validate the design, we performed a theoretical power consumption analysis and budget review.

## 6.1 Power Consumption Profile

Understanding the energy budget is crucial for a solar-powered system.

- **Idle State (Monitoring):**
  - Arduino Uno: 45mA
  - LCD (Backlight ON): 20mA
  - Sensor: 5mA
  - **Total Idle Current:**  $\approx 70\text{mA}$
- **Active State (Pumping):**
  - Water Pump: 300mA
  - Relay Coil: 70mA
  - System Overhead: 70mA
  - **Total Active Current:**  $\approx 440\text{mA}$

## 6.2 Battery Life Calculation

Given a single 18650 Li-ion Battery with a capacity of **2000mAh**:

- **Theoretical Idle Runtime:**

$$\frac{2000\text{mAh}}{70\text{mA}} \approx 28.5 \text{ Hours}$$

This means even with zero sunlight, the system can monitor the plant for over a full day.

- **Theoretical Active Runtime:**

$$\frac{2000\text{mAh}}{440\text{mA}} \approx 4.5 \text{ Hours}$$

Since the pump typically requires only 5 minutes to water a pot, the daily active energy consumption is very low ( $\approx 36\text{mAh}$ ).

- **Solar Recharge:** The 160mA solar panel can generate  $\approx 800 - 900\text{mAh}$  effectively on a sunny day (5-6 peak hours). This is more than sufficient to replenish the daily consumption of the system, ensuring perpetual operation without external charging.

## 6.3 Budget Analysis

A key constraint was affordability for urban households. The following table breaks down the cost of the prototype.

Table 1: Detailed Component Budget

Component	Specs	BDT
Arduino Uno R3	ATmega328P	1050
I2C LCD Display	16x2 Char	360
Mini Water Pump	3-6V DC	160
Relay Module	1-Channel 5V	75
Soil Sensor	Resistive	120
Solar Panel	6V 1W	350
TP4056 Module	1A Charging	40
18650 Battery	2000mAh	300
Misc (Wires/Chassis)	-	100
<b>Total</b>		<b>2,555</b>

With a total cost of approximately 2,555 BDT ( $\approx \$23$  USD), the system is highly affordable compared to commercial automated irrigation solutions which can cost upwards of \$100.

## 7 Testing and Validation

The system was tested under controlled conditions to verify its reliability.

### 7.1 Calibration

The sensor values were calibrated by taking readings in two extremes:

- **Air (Dry):** Sensor reading  $\approx 1023$  (Max resistance).
- **Water (Wet):** Sensor reading  $\approx 250$  (Min resistance).

These values were mapped to a 0-100% scale in the software to provide intuitive user feedback.

### 7.2 Test Scenarios

**Case 1: Dry Soil.** The probe was placed in dry potting mix. The LCD displayed "Moisture: 10%". The Relay clicked ON, and the pump dispensed water. **Case 2: Wet Soil.** Water was added until the probe read "Moisture: 85%". The Relay clicked OFF immediately. **Case 3: Solar Charging.** The system was left in sunlight. The TP4056 red LED lit up, indicating charging. After 4 hours, the blue LED lit up, indicating a full charge.

## 8 Alignment with SDGs

This project is not merely a technical exercise but a social intervention aligned with the UN Sustainable Development Goals (SDGs):

- **SDG 2 (Zero Hunger):** Target 2.4 calls for sustainable food production systems. By automating rooftop gardening, this project empowers urban families to grow their own food, enhancing local food security and reducing reliance on expensive, transported produce.
- **SDG 6 (Clean Water and Sanitation):** Target 6.4 focuses on water-use efficiency. Manual hose watering often leads to significant runoff. By using sensor data to irrigate only when necessary, this system eliminates waste, embodying the principle of "More crop per drop."
- **SDG 7 (Affordable and Clean Energy):** Target 7.2 aims to increase the share of renewable energy. This project demonstrates that small-scale solar integration is viable and cost-effective, serving as an educational tool for renewable energy adoption.
- **SDG 11 (Sustainable Cities and Communities):** Target 11.6 aims to reduce the adverse environmental impact of cities. Rooftop gardens are a primary defense against the Urban Heat Island effect. Automation ensures these gardens survive, contributing to cooler, greener, and more breathable cities.
- **SDG 12 (Responsible Consumption and Production):** The system promotes responsible resource usage. It prevents the waste of water, electricity, and seeds (by preventing plant death), ensuring a sustainable consumption pattern.

## 9 Conclusion

The Automated Irrigation System for Rooftop Gardening successfully demonstrates that accessible, low-cost technology can solve critical urban problems. By automating the task of watering, we have removed the primary barrier to entry for potential urban gardeners: consistency and time commitment. The system is effective, environmentally friendly, and scalable. It proves that sustainability does not require expensive industrial equipment; it can be achieved with a \$23 device, open-source code, and the sun.

### 9.1 Future Scope

While the current prototype is fully functional, several enhancements are proposed for future iterations:

- **IoT Integration:** Adding an ESP8266 Wi-Fi module would allow the system to upload data to cloud platforms like ThingSpeak. This would enable users to monitor their garden from anywhere in the world and receive SMS alerts if the water tank is empty.
- **Weather API Integration:** By connecting to the internet, the system could check local weather forecasts. If rain is predicted in the next 2 hours, the system could delay watering, saving even more water.
- **Capacitive Sensing:** Replacing the resistive sensor with a capacitive one would eliminate the issue of probe corrosion, extending the hardware lifespan significantly.

## References

- [1] Krishnamoorthy, P., et al. (2014). "Automatic Irrigation System Using Microcontroller." *International Journal of Engineering Research & Management Technology*, 1(2), 141.
- [2] Salazar, J. C. R., et al. (2024). "Irrigation System through Intelligent Agents Implemented with Arduino Technology," *Technological University of Panamá*.
- [3] Hamoodi, A., et al. "Automated irrigation system based on soil moisture using Arduino board." *Bulletin of Electrical Engineering and Informatics*, Vol 9, Issue 3.
- [4] Singla, B., et al. (2019). "A study on smart irrigation system using IoT." *International Journal of Advance Research, Ideas and Innovations in Technology*, 5(2), 1416.
- [5] Sy, J. B., et al. (2020). "A Low-Cost Arduino-based Smart Irrigation System," *Int. J. Emerg. Trends Eng. Res.*, 8(9), 5645–5650.
- [6] Ahmed, M. S., et al. (2020). "Automated watering and irrigation system using Arduino UNO," *International Journal of Scientific & Engineering Research*, 11(8), 15–20.
- [7] Patil, S. S., et al. (2021). "Fabrication of solar powered automatic irrigation system," *IRJET*, 11(1), 58–65.
- [8] Adekunle, A. S., et al. (2020). "Design and implementation of solar powered automatic irrigation system," *American Journal of Electrical and Computer Engineering*, 4(1), 1–8.
- [9] Babaa, S. E., et al. (2020). "Smart irrigation system using Arduino with solar power," *IJERT*, 9(5), 1–6.