LOVELY PROFESSIONAL UNIVERSITY

Smart Rain Water Harvesting and Irrigation System

Submitted by-

Name	**Registration Number**
Abhishek Maurya	12505398
Amogh Pandey	12505264
Ashish Chandrans	12505021
Ankit Kumar	12505189

1. Problem Statement

Water scarcity and inefficient water utilization have become pressing global challenges, particularly in agricultural and urban sectors. Traditional irrigation systems rely heavily on manual control or fixed scheduling, which often leads to over-irrigation, under-irrigation, and unnecessary water wastage. Similarly, conventional rainwater harvesting systems primarily focus on collection and storage but lack automation, monitoring, and intelligent overflow management.

The absence of an integrated, automated, and cost-effective system to simultaneously collect, store, monitor, and utilize rainwater for irrigation leads to inefficient water management. There is a need for a smart system that can autonomously detect rainfall, monitor soil moisture, measure tank water levels, and control water flow for both irrigation and recharge without human intervention.

Therefore, the problem addressed in this project is the lack of an affordable, automated, and sustainable solution that can:

1. Detect rainfall and automatically manage tank lid operation.
2. Monitor tank water levels and prevent overflow through groundwater recharge.
3. Supply water to crops automatically based on soil moisture conditions, independent of tank level.
4. Optimize water usage and support sustainable resource management using low-cost microcontroller-based technology.

2. Literature Survey

Abstract

This survey explores advancements in smart rainwater harvesting (RWH) and IoT-based sustainable water management. Ten key research papers (2019–2025) were analyzed on automation, real-time monitoring, and machine-assisted systems integrating sensors and microcontrollers. Modern RWH approaches now use Arduino or Raspberry Pi for automated control of water levels, soil moisture, and rainfall detection, addressing water scarcity and overuse in both urban and rural settings. The reviewed literature and a proposed model collectively demonstrate how IoT can revolutionize water harvesting through automation, real-time data exchange, and optimized storage–distribution cycles.

I. Introduction

Water scarcity is a global concern intensified by climatic unpredictability, deforestation, and rapid urbanization. Rainwater harvesting, historically vital in civilizations such as the Indus Valley, has evolved into a sustainable technology enhanced by IoT. By embedding microcontrollers, cloud platforms, and sensors, smart systems can autonomously detect rainfall, measure tank levels, regulate irrigation, and ensure overflow safety.

II. Conventional Concepts and Transition to Smart Systems

Early RWH design focused on manual collection and ground infiltration. Gannoju et al. (2019) introduced a servo-motor-based system that automatically opened tank lids during rainfall using Arduino input from water sensors, effectively minimizing overflow and maximizing catchment efficiency. Likewise, Ashish et al. (2019) designed a solar-powered system combining UPVC pipes, water-level sensors, and alarms to store rooftop water and recharge groundwater.
However, these early systems required local maintenance and lacked cloud connectivity or remote management, marking the transition phase toward IoT integration in the early 2020s.

III. IoT and Cloud Integrated Approaches

Mid-decade studies exhibit full IoT adoption:
• **Talari et al. (2021)** proposed the IIASRH model, which integrates Raspberry Pi 4, ultrasonic sensors, rainfall and pH sensors, and servo motors for redirection of water to separate tanks for filtration or treatment. The model achieved complete automation through Arduino Cloud IoT and MySQL connectivity, displaying tank levels and acidity remotely.
• **Gonsalves et al. (2023)** introduced an Arduino-UNO water management design featuring pH, flow, and water-level sensors transmitting data over ESP8266 Wi-Fi to a cloud interface for real-time visualization. The system activated pumps automatically, reducing overflow while providing filtration control.
• **Sachin et al. (2025)** extended this model using the Blynk IoT platform, allowing dashboards to visualize tank level, pump status, and overflow warnings on smartphones, merging traditional catchment storage with IoT alerting frameworks.

IV. Automation and Machine Learning Integration

By 2024, researchers incorporated analytics into water systems:
• **Nandini et al. (2024)** demonstrated an IoT smart water meter applying machine

learning (Random Forest and Gradient Boosting) for assessing water quality through pH and turbidity classifications, reaching ~89 % accuracy.
• Machine-assisted evaluation supports predictive maintenance, enabling real-time threshold validation and proactive control of valves or pumps.

V. Sustainable Practices and Policy Dimensions

Complementary studies emphasized socio-environmental impact. **Akash Kumar Singh et al. (2025)** analyzed India's rural and urban RWH policies, concluding that integrating indigenous Haveli and Johad systems with IoT sensors can enhance water security and reduce groundwater stress. **Hari Om et al. (2025)** contextualized traditional designs within modern infrastructures, highlighting RWH's role in flood control, erosion prevention, and aquifer recharge through advanced monitoring technology.

VI. Smart Monitoring and Home Integration

Shivkar et al. (2024) proposed a fully automated IoT-based rooftop conservation system using servo-driven lids that open upon rainfall detection and close afterward, monitored through a mobile app. Their system also automated pump operation based on upper and lower water thresholds, reducing human intervention and wastage.
Such works formed the foundation for autonomous home and agricultural water systems leveraging environmental sensors.

VII. Proposed Project Implementation

The proposed Arduino UNO-based Smart Rainwater Harvesting and Irrigation System builds directly upon the technologies described above.

A. Core Components

• **Arduino UNO:** Acts as the central microcontroller managing all sensors and actuators.
• **Rain Sensor:** Detects precipitation and signals the Arduino to trigger the servo motor.
• **Servo Motor:** Opens the tank lid automatically when rain begins and closes it after rainfall stops.
• **Ultrasonic Sensor:** Measures water level in storage. When approaching maximum capacity, it sends data to open the solenoid valve positioned at the outlet to discharge excess water into a ground recharge pit.
• **Soil Moisture Sensor:** Monitors soil humidity. When levels fall below threshold, the Arduino activates the solenoid valve connected to the tank outlet, watering the field automatically.
• **Solenoid Valves:** Control flow between tank, ground discharge, and irrigation pipelines using 12 V DC actuation.

B. Extra Components

• **Relay Module:** Functions as an electrically operated switch that allows the Arduino to control high-voltage or high-current devices such as solenoid valves and water pumps safely. It provides isolation between the low-voltage control circuit and high-power components.
• **Jumper Wires:** Used to establish electrical connections between the Arduino board, sensors, and modules on the breadboard or PCB. They enable flexible and non-permanent wiring during circuit prototyping.
• **Battery (DC Power Source):** Supplies backup or standalone power to the Arduino and sensors, ensuring uninterrupted operation during power outages or in remote field installations. Rechargeable or solar-charged batteries can enhance sustainability.

C. Working Principle

1. **Rain Detection:** The water sensor triggers the servo motor to open the tank inlet allowing rainwater collection.
2. **Storage & Monitoring:** The ultrasonic sensor continuously measures tank levels.
3. **Overflow Prevention:** Once the tank exceeds pre-set capacity, the Arduino activates an outlet solenoid to discharge excess water underground, contributing to groundwater recharge.
4. **Irrigation Automation:** The soil-moisture data ensure that when soil becomes dry, another solenoid valve opens to supply stored rainwater to the field.
5. **Power Source and Control:** A relay module manages switching between solenoid valves or pumps, while a battery or solar source provides sustainable, uninterrupted power supply for field deployment.

VIII. Research Gap

Although several studies and prototypes have explored rainwater harvesting and automated irrigation, most existing systems still operate in isolation, lack integrated intelligence, or require manual supervision.
The following key research gaps have been identified:

1. **Lack of integration between harvesting and irrigation systems:**
   Most rainwater harvesting setups are limited to water collection and storage. They rarely integrate real-time irrigation control, preventing optimal use of the stored rainwater.
2. **Limited automation and sensing capabilities:**
   Traditional irrigation systems often rely on fixed timers or manual switching, ignoring dynamic environmental conditions such as rainfall, soil moisture, and tank water level.

This leads to over-irrigation or under-utilization of water resources.

3. **Absence of intelligent overflow management:**
Existing RWH systems seldom address overflow control. Without automatic diversion, excess water is wasted instead of being redirected to groundwater recharge pits for sustainable reuse.

4. **Minimal use of low-cost microcontroller-based solutions:**
While some modern systems employ IoT platforms or industrial controllers, these are often expensive and complex. There is a gap for a low-cost, Arduino-based approach that can perform all major water management tasks effectively in rural and semi-urban contexts.

IX. Conclusion

Across the decade, RWH systems have evolved from purely mechanical storage units to autonomous IoT-enabled ecosystems. Automation using Arduino UNO, servo motors, solenoid valves, and sensors enables real-time monitoring, self-regulating control, and sustainable water redistribution.
The proposed system extends this evolution by integrating rain detection, overflow discharge, and irrigation automation into one framework powered by renewable energy. Such hybrid models can greatly assist rural and semi-urban agricultural regions by optimizing every drop of rainwater for long-term environmental balance.

References

1. Roshni Gannoju et al., “Automated Rainwater Harvesting System,” *Global Journal of Engineering Science and Researches*, 2019.
2. Ashish et al., “Smart Rainwater Harvesting,” *JETIR*, 2019.
3. Praveen Talari et al., “IIASRH: Integrative IoT Approach for Smart Rainwater Harvesting,” *IJITEE*, 2021.
4. Priyanca Gonsalves et al., “Smart Rainwater Harvesting System,” *IRJET*, 2023.
5. Amraja Shivkar et al., “Smart Water Conservation and RWH,” *Educational Administration*, 2024.
6. G.S. Nandini et al., “IoT Based Smart Water Management System with Machine Learning,” *IJERT*, 2024.
7. Akash Kumar Singh et al., “Performance and Policy Review of RWH in India,” *IJRASET*, 2025.
8. Hari Om et al., “Optimizing Rainwater Harvesting Systems,” *IJRCET*, 2025.

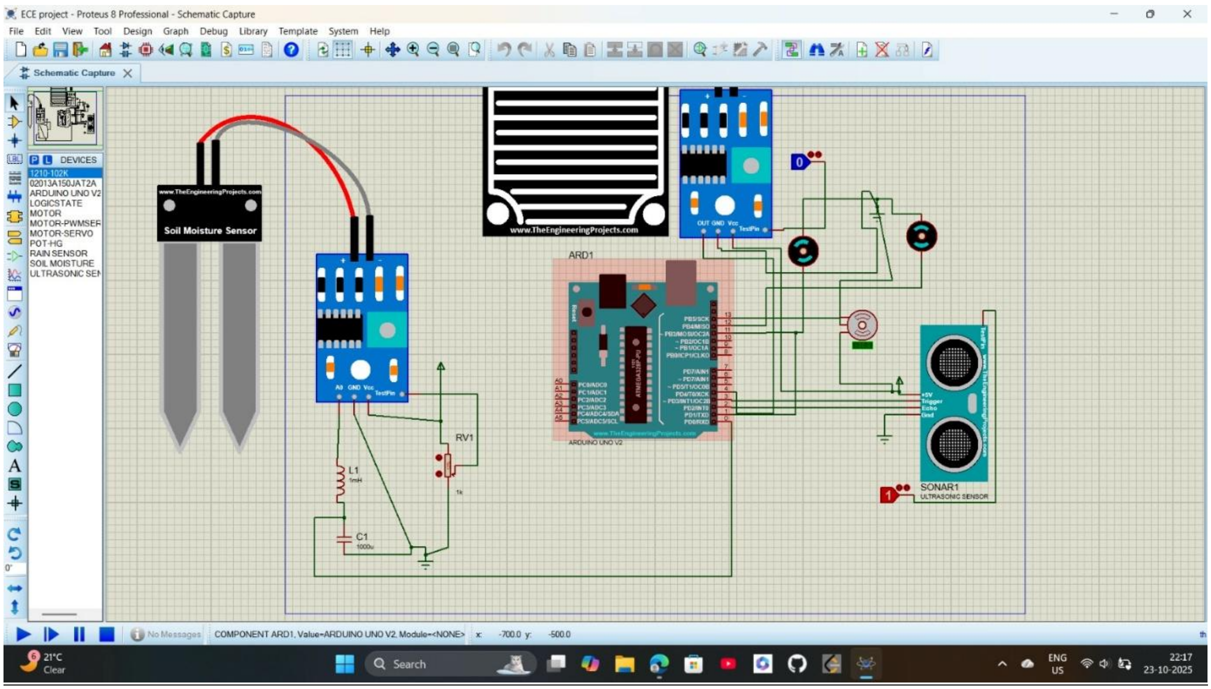
9. Sachin K. et al., “IoT-Enabled Smart RWH for Real-Time Monitoring,” *JETNR*, 2025.

10. Ayush Dixit et al., “Smart Rainwater Harvesting Using Real-Time IoT,” *IJCRT*, 2025.

3. Circuital Simulation

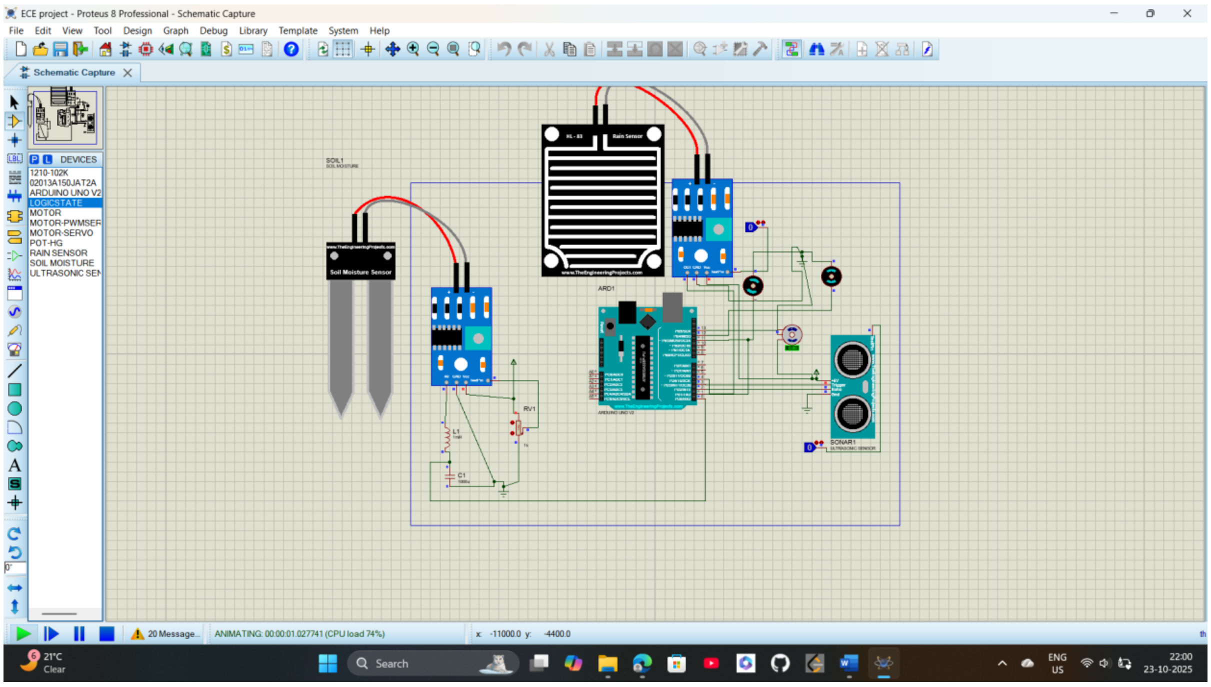
NOTE: We are using dc motors to represent solenoid valve in circuit diagram.

Before Simulation:



COMPONENT ARD1, Value=ARDUINO UNO V2, Module=<NONE>   x: -700.0 y: -500.0

After Simulation:

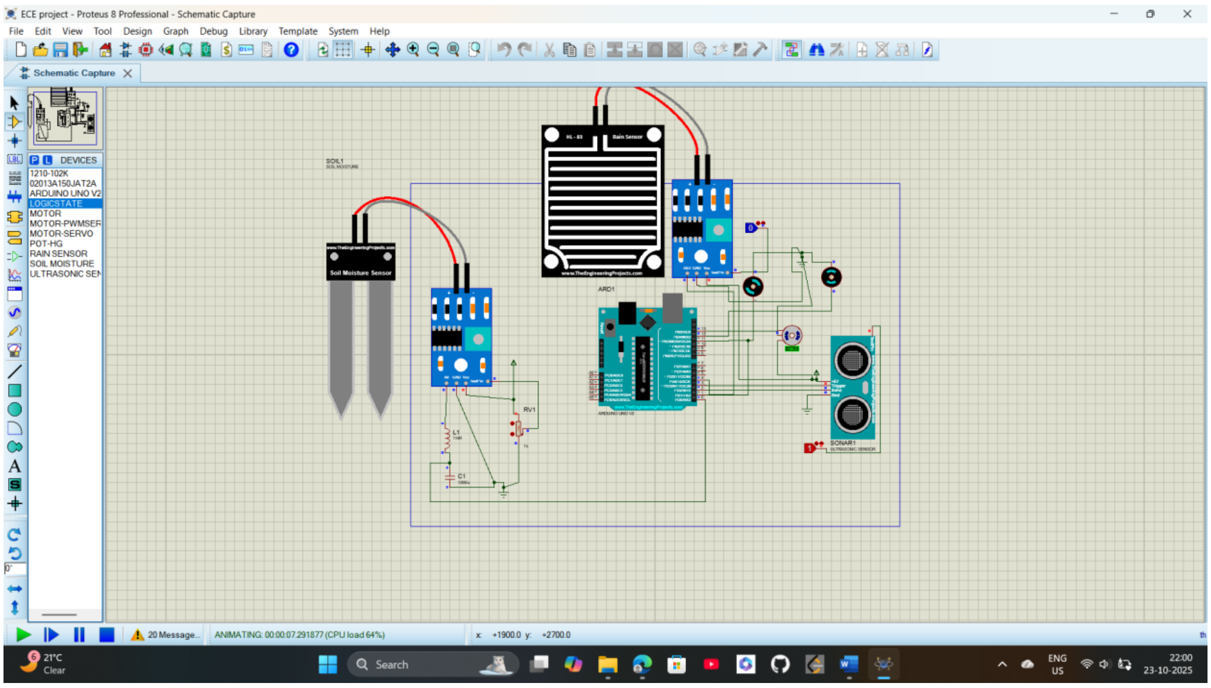


ECE project - Proteus 8 Professional - Schematic Capture

20 Message

ANIMATING: 00:00:01.027741 (CPU load 74%)

x: -11000.0 y: -4400.0

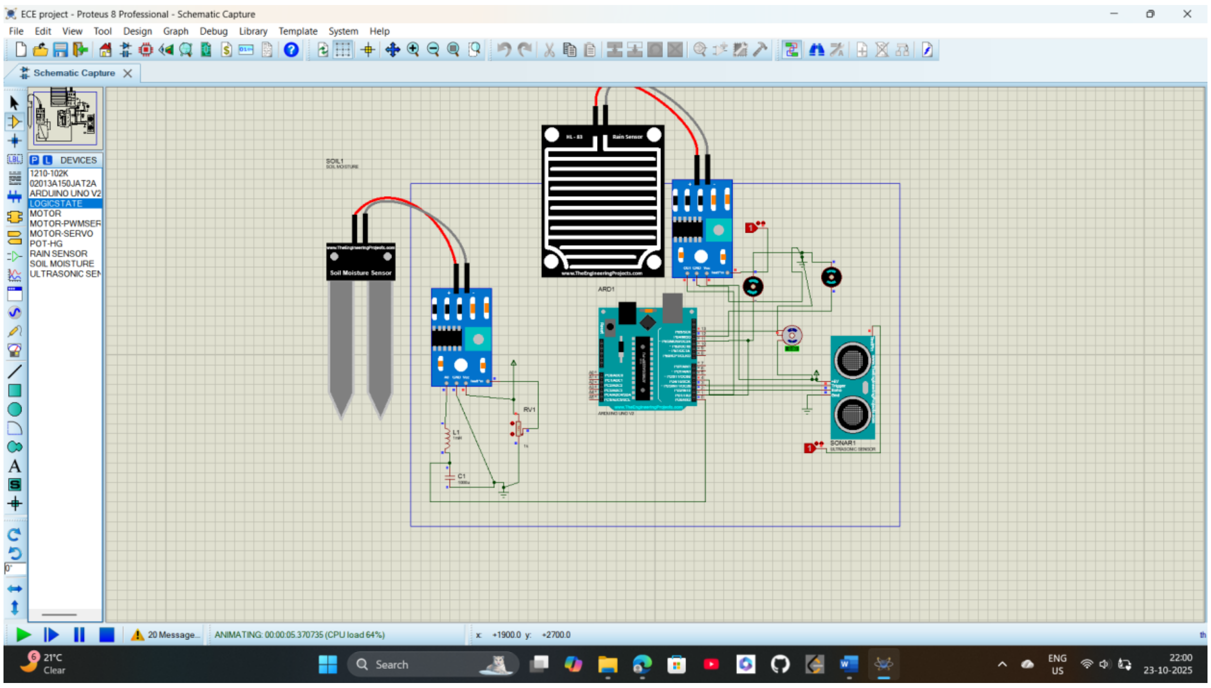


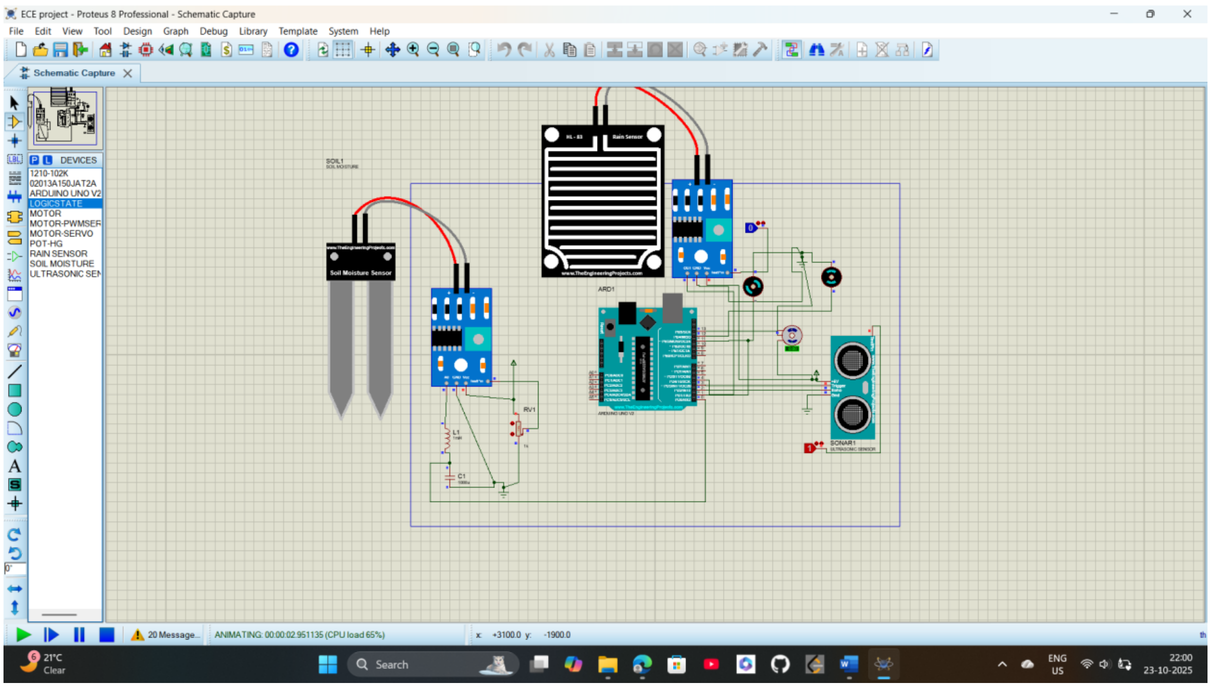
ECE project - Proteus 8 Professional - Schematic Capture

20 Message

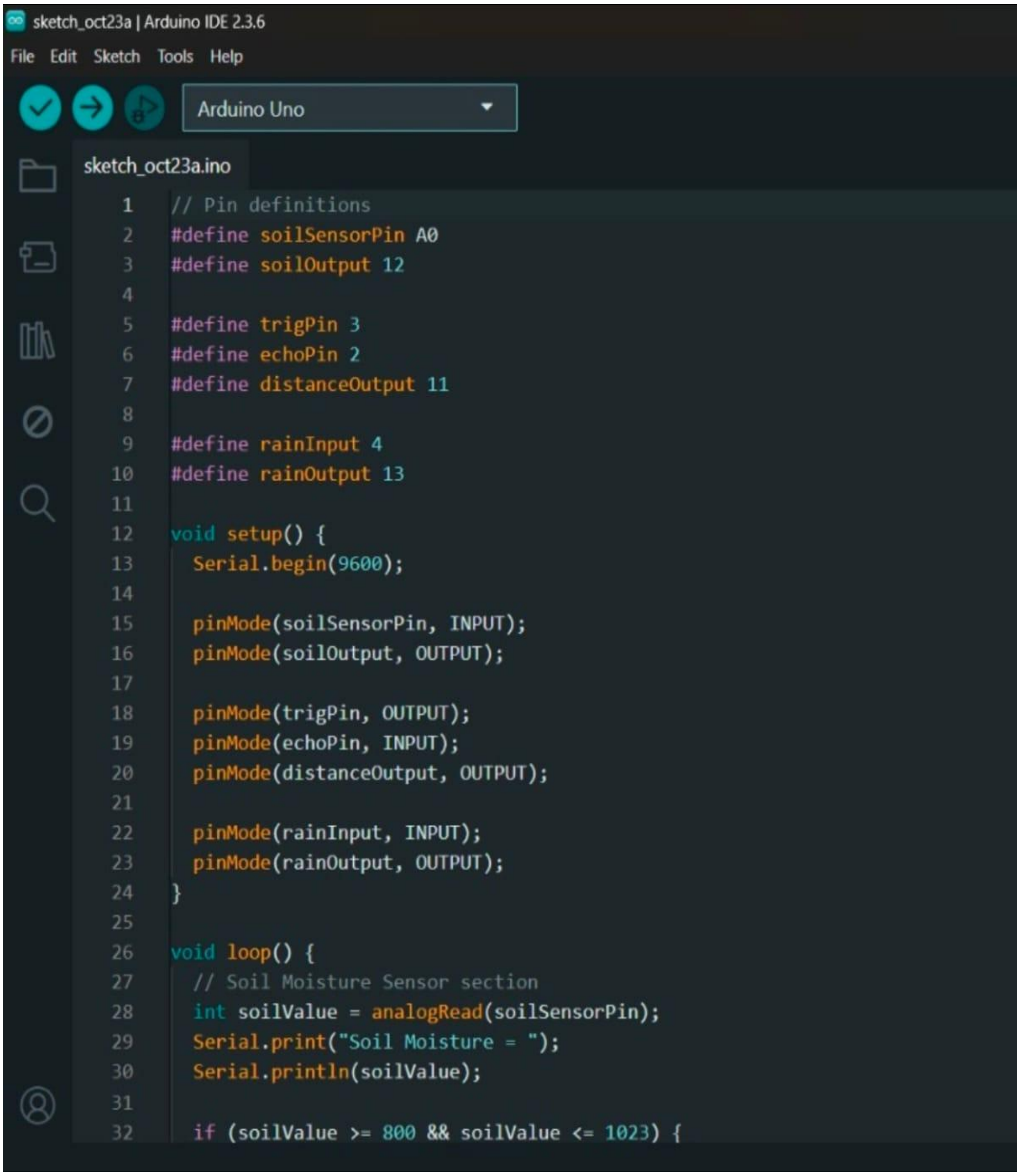
ANIMATING: 00:00:07.291877 (CPU load 64%)

x: +1900.0 y: +2700.0

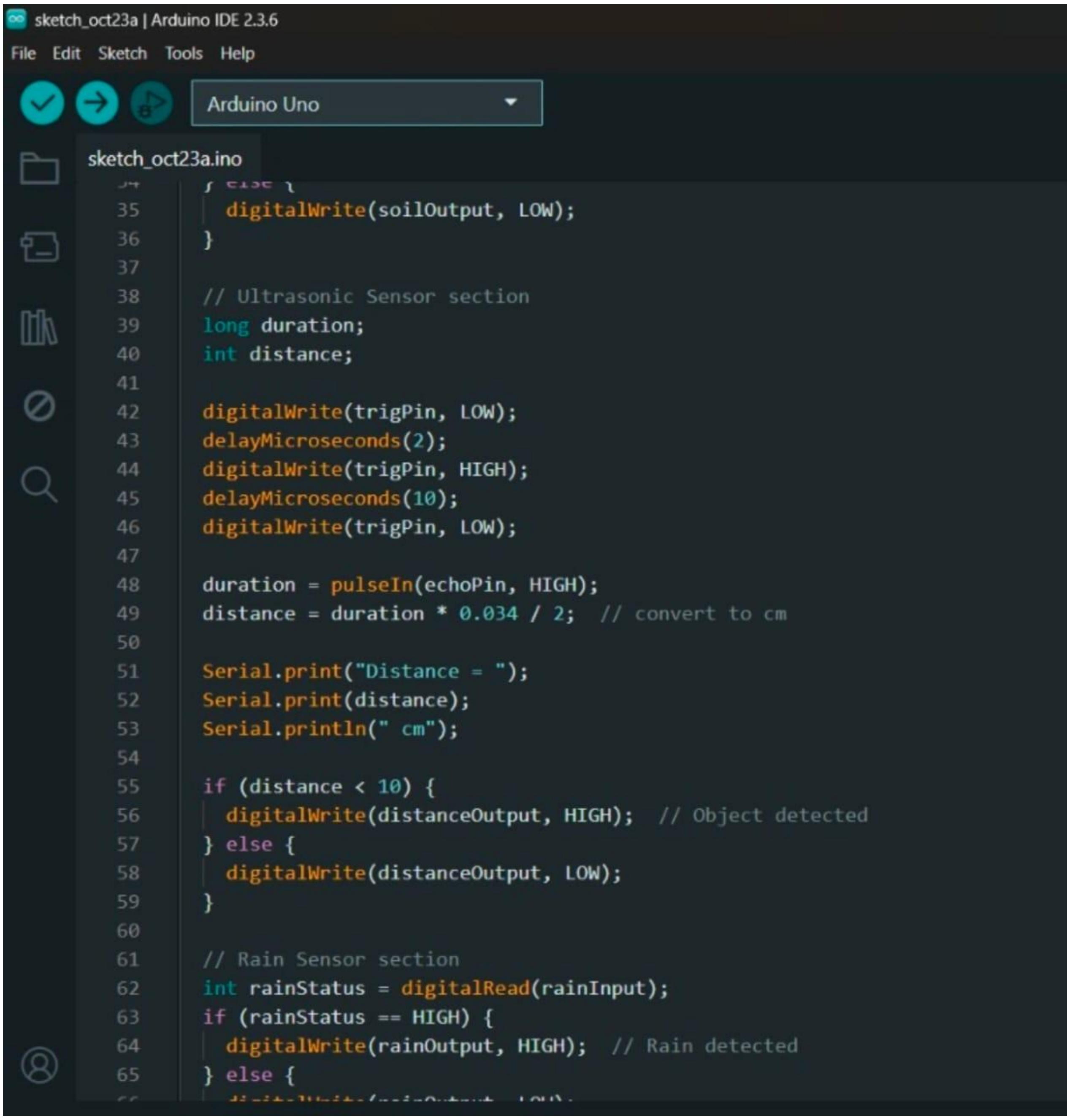

ECE project - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
Schematic Capture
DEVICES
1210-102K
02013A150JAT2A
ARDUINO UNO V2
LOGICSTATE
MOTOR
MOTOR-PWMSER
MOTOR-SERVO
POT-HG
RAIN SENSOR
SOIL MOISTURE
ULTRASONIC SEN
SOIL1
Soil Moisture Sensor
Rain Sensor
ARD1
RV1
L1
C1
SONAR1
ULTRASONIC SENSOR
20 Message
ANIMATING: 00:00:05.370735 (CPU load 64%)
x: +1900.0 y: +2700.0
21°C Clear
Search
ENG US
22:00 23-10-2025


ECE project - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
Schematic Capture
DEVICES
1210-102K
02013A150JAT2A
ARDUINO UNO V2
LOGICSTATE
MOTOR
MOTOR-PWMSER
MOTOR-SERVO
POT-HG
RAIN SENSOR
SOIL MOISTURE
ULTRASONIC SEN
SOIL1
Soil Moisture Sensor
Rain Sensor
ARD1
RV1
L1
C1
SONAR1
ULTRASONIC SENSOR
20 Message
ANIMATING: 00:00:02.951135 (CPU load 65%)
x: +3100.0 y: -1900.0
21°C Clear
Search
ENG US
22:00 23-10-2025

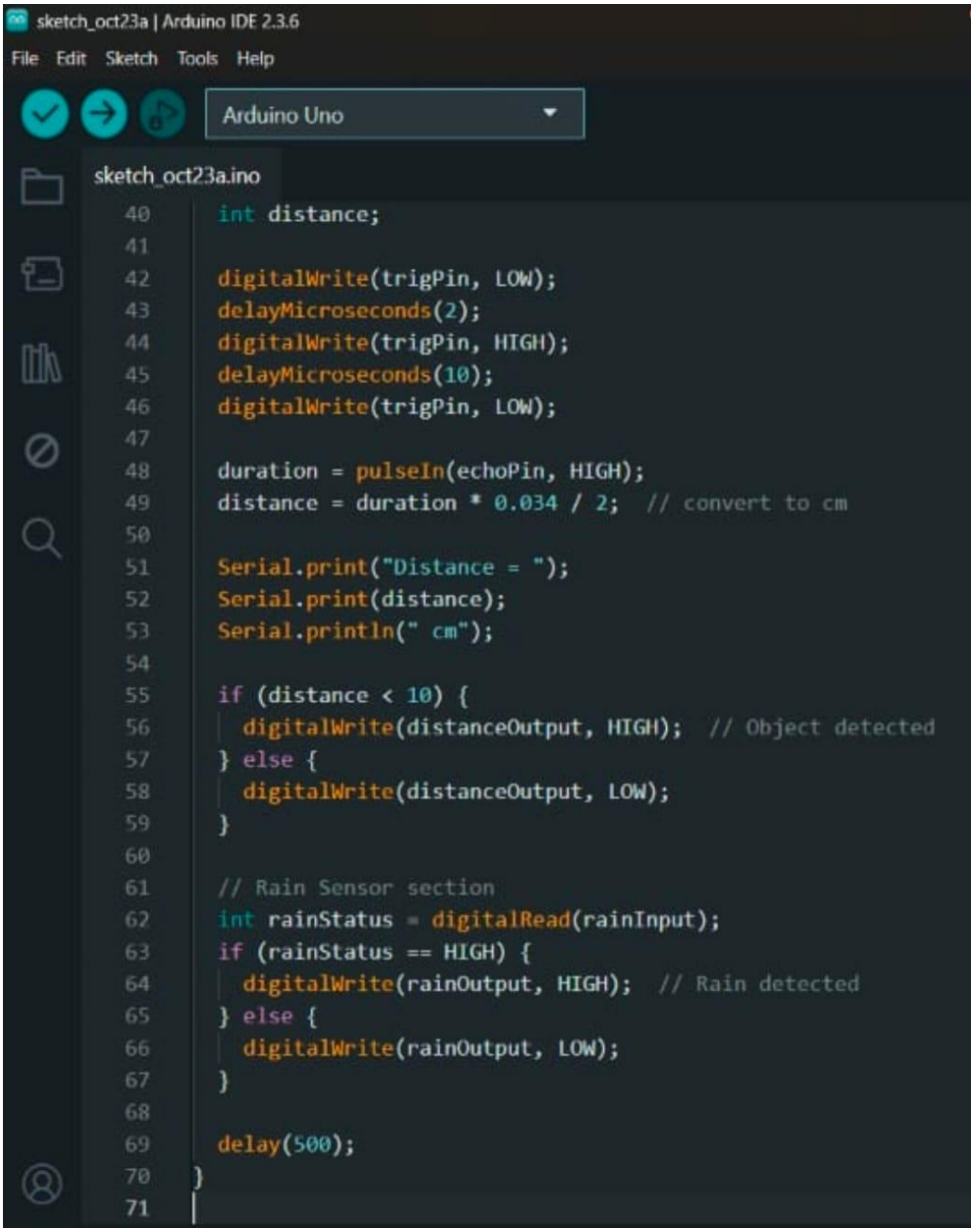
Arduino IDE Code


sketch_oct23a | Arduino IDE 2.3.6
File Edit Sketch Tools Help
Arduino Uno
sketch_oct23a.ino

```
// Pin definitions
#define soilSensorPin A0
#define soilOutput 12

#define trigPin 3
#define echoPin 2
#define distanceOutput 11

#define rainInput 4
#define rainOutput 13

void setup() {
  Serial.begin(9600);

  pinMode(soilSensorPin, INPUT);
  pinMode(soilOutput, OUTPUT);

  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(distanceOutput, OUTPUT);

  pinMode(rainInput, INPUT);
  pinMode(rainOutput, OUTPUT);
}

void loop() {
  // Soil Moisture Sensor section
  int soilValue = analogRead(soilSensorPin);
  Serial.print("Soil Moisture = ");
  Serial.println(soilValue);

  if (soilValue >= 800 && soilValue <= 1023) {
```

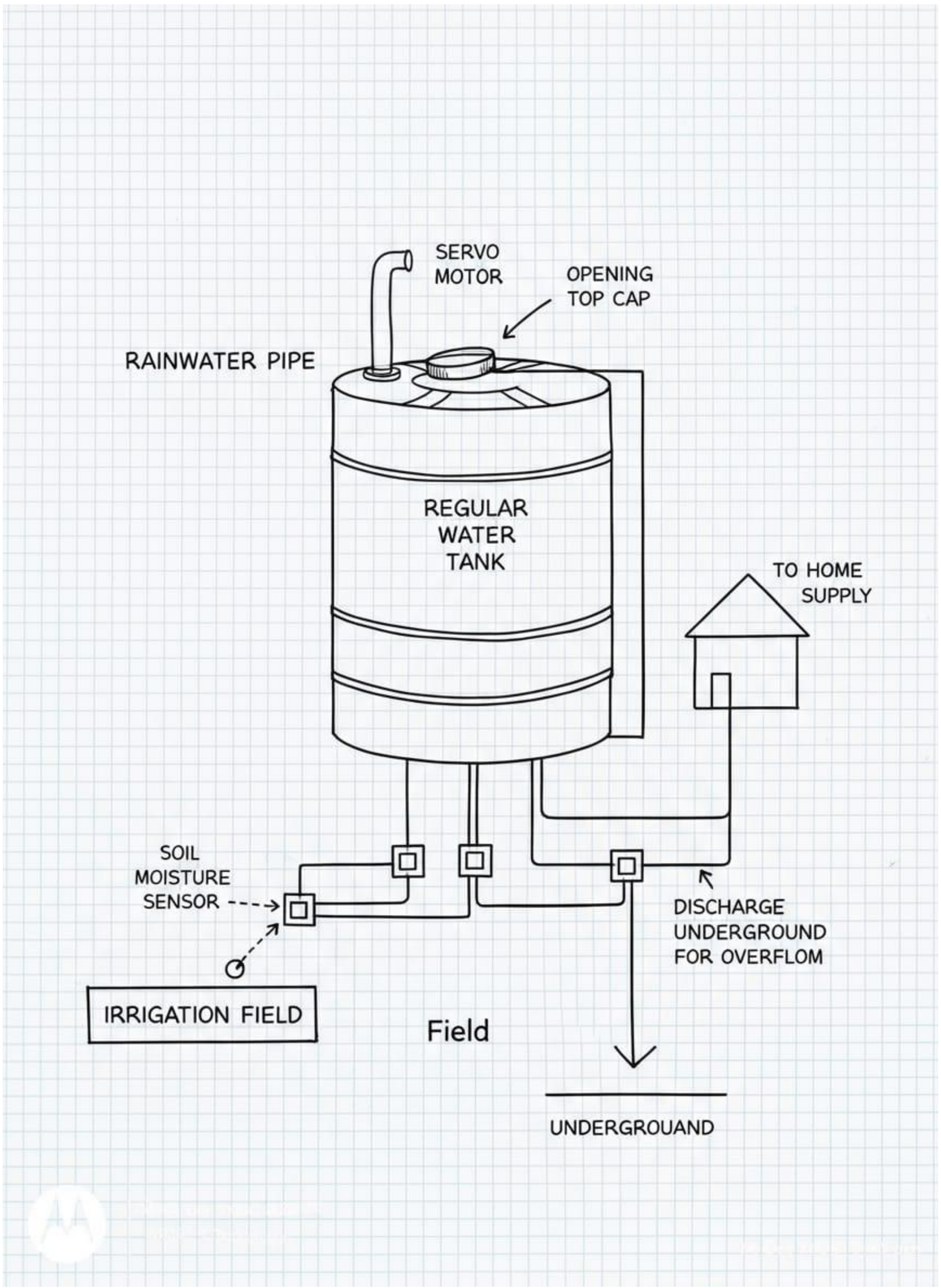

sketch_oct23a | Arduino IDE 2.3.6
File Edit Sketch Tools Help
Arduino Uno
sketch_oct23a.ino

```
} else {
  digitalWrite(soilOutput, LOW);
}

// Ultrasonic Sensor section
long duration;
int distance;

digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);
distance = duration * 0.034 / 2;  // convert to cm

Serial.print("Distance = ");
Serial.print(distance);
Serial.println(" cm");

if (distance < 10) {
  digitalWrite(distanceOutput, HIGH);  // Object detected
} else {
  digitalWrite(distanceOutput, LOW);
}

// Rain Sensor section
int rainStatus = digitalRead(rainInput);
if (rainStatus == HIGH) {
  digitalWrite(rainOutput, HIGH);  // Rain detected
} else {
  digitalWrite(rainOutput, LOW);
```


sketch_oct23a | Arduino IDE 2.3.6
File Edit Sketch Tools Help
Arduino Uno
sketch_oct23a.ino

```
  int distance;

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;  // convert to cm

  Serial.print("Distance = ");
  Serial.print(distance);
  Serial.println(" cm");

  if (distance < 10) {
    digitalWrite(distanceOutput, HIGH);  // Object detected
  } else {
    digitalWrite(distanceOutput, LOW);
  }

  // Rain Sensor section
  int rainStatus = digitalRead(rainInput);
  if (rainStatus == HIGH) {
    digitalWrite(rainOutput, HIGH);  // Rain detected
  } else {
    digitalWrite(rainOutput, LOW);
  }

  delay(500);
}
```

MODEL DIAGRAM


SERVO MOTOR
OPENING TOP CAP
RAINWATER PIPE
REGULAR WATER TANK
TO HOME SUPPLY
SOIL MOISTURE SENSOR
DISCHARGE UNDERGROUND FOR OVERFLOM
IRRIGATION FIELD
Field
UNDERGROAUND