Project: Smart Water System

Project Objectives

1. Real time water consumption monitoring
2. Public awareness
3. Water conservation
4. Sustainable Resource Management

Real Time Water Consumption Management

- Water is a precious resource, and efficient management is crucial. Real-time water consumption management allows us to monitor and control water usage effectively.

Why Real-Time Monitoring?

- Ensure efficient water use
- Detect leaks and abnormalities
- Save costs and conserve resources

Benefits:

- Water conservation
- Cost savings
- Early leak detection
- Improved sustainability

Conclusion:

- Real-time water consumption management is vital for resource conservation and cost savings.
- Embrace the technology for a sustainable future.

Public Awareness on Water

The Role of the Public:

- Reduce water waste
- Adopt water-efficient practices
- Advocate for policy changes

Reducing Water Footprint:

- Shorter showers
- Full loads in the dishwasher and laundry
- Turning off taps while brushing teeth

Certainly, here are key points to emphasize in a public awareness campaign on water conservation:

- Water's Vital Role: Highlight that water is essential for all life on Earth, emphasizing its significance for drinking, agriculture, industry, and ecosystems.
- Current Water Scarcity: Provide statistics and examples of regions facing water scarcity or drought to illustrate the pressing nature of the issue.
- Global Water Crisis: Explain that we are facing a global water crisis due to factors like population growth, climate change, and over-extraction of water resources.
- Individual Responsibility: Stress that everyone has a role to play in conserving water, from households to businesses and institutions.

Water Conservation

- Water is a Precious Resource:
  - Highlight the finite nature of freshwater resources on Earth.
  - Emphasize that only a small percentage of water is available for human use.
- Environmental Impact:
  - Discuss the ecological importance of water conservation.
  - Explain how water conservation can help protect ecosystems and wildlife.
- Reducing Water Waste:
  - Encourage individuals to fix leaks promptly to prevent water wastage.
  - Promote the use of water-efficient appliances and fixtures.
- Conscious Water Use:
  - Encourage shorter showers and turning off taps while brushing teeth.
  - Explain the concept of "if it's yellow, let it mellow; if it's brown, flush it down" for toilets.
- Rainwater Harvesting:
  - Explain the benefits of collecting rainwater for irrigation and non-potable uses.
  - Provide information on rain barrels and rain gardens.
- Community and School Involvement:
  - Highlight the role of communities and schools in promoting water conservation.
  - Mention community gardens and educational programs.

Sustainable Resource Management

Key Points on Water in Sustainable Resource Management:

Essential Resource: Water is a fundamental resource for life and is vital for ecosystems, human health, agriculture, industry, and energy production.

Finite Supply: Despite covering most of the Earth's surface, freshwater resources are limited and unevenly distributed, making sustainable management critical.

Integrated Approach: Sustainable water resource management requires an integrated approach that considers ecological, social, and economic factors.

IOT SENSOR DESIGN

Designing an Internet of Things (IoT) sensor involves several key steps and considerations. IoT sensors are devices that collect data from the physical world and transmit it to a central system or cloud for analysis and action.

DESIGN THE PLAN:

Designing an IoT sensor involves several key steps and considerations to ensure it functions reliably and efficiently. Below is a comprehensive plan for designing an IoT sensor:

1. DEFINE THE PURPOSE AND REQUIREMENTS:

Clearly define the purpose of the IoT sensor. What data do you want to collect, and what will you do with that data?

Determine the environmental conditions the sensor will be exposed to (e.g., temperature, humidity, dust, water resistance).

Specify the range and accuracy of measurements required.

2. SELECT SENSORS AND COMPONENTS:

Choose appropriate sensors based on the data you want to collect (e.g., temperature, humidity, motion, light).

Select microcontrollers or microprocessors (e.g., Arduino, Raspberry Pi) suitable for your application.

Decide on communication protocols (e.g., Wi-Fi, Bluetooth, LoRa, Zigbee) based on the range and data transfer requirements.

3. POWER SOURCE:

Determine the power source (e.g., batteries, solar, wired) and calculate power consumption for long-term operation.

Implement power-saving techniques to extend battery life if necessary

4. DATA PROCESSING AND STORAGE:

- *Decide whether data processing should be done on the sensor itself or on a centralized server.*
- *Select appropriate storage solutions (e.g., SD cards, cloud storage) for collected data.*

5. CONNECTIVITY:

- *Implement the chosen communication protocol to transmit data to a central hub or IoT platform.*
- *Ensure security measures (e.g., encryption) are in place to protect data during transmission.*

6. FIRMWARE AND SOFTWARE DEVELOPMENT:

- *Develop firmware for the sensor to handle data*

DEPLOYMENT OF IOT SENSORS TO MONITOR WATER CONSUMPTION IN PUBLIC PLACES:

Deploying IoT sensors to monitor water consumption in public places can help organizations and municipalities better manage their water resources, reduce waste, and improve overall efficiency. Here's a deployment plan for such a project:

1. DEFINE OBJECTIVES AND SCOPE:

- Clearly define the objectives of your water consumption monitoring project. Determine the specific public places you want to monitor (e.g., parks, public restrooms, government buildings).

2. SENSOR SELECTION:

- Choose appropriate water flow sensors or meters that can accurately measure water consumption.
- Select IoT devices with connectivity options (e.g., Wi-Fi, cellular, LoRa) compatible with the deployment locations.

3. DATA COLLECTION POINTS:

- Identify strategic locations for sensor placement within the selected public places. Ensure sensors are easily accessible for installation and maintenance.

4. CONNECTIVITY INFRASTRUCTURE:
Establish the necessary connectivity infrastructure, such as Wi-Fi access points or cellular signal boosters, to ensure reliable data transmission from the sensors to the central monitoring system.
5. POWER SUPPLY:
Determine the power source for the IoT sensors. Options include mains power, battery, or solar panels, depending on the availability and feasibility at each location.
6. CENTRALIZED DATA HUB:
Set up a centralized data hub or cloud-based platform to receive, store, and process data from the sensors. Implement security measures to protect data during transmission and storage.
7. SENSOR INSTALLATION:
Install the sensors at the predefined locations. Ensure they are securely mounted and properly calibrated. Configure the sensors to send data at regular intervals to the central hub.
8. DATA VISUALIZATION A ANALYSIS:
Develop a user-friendly dashboard or interface to visualize real-time and historical water consumption data. Implement analytics tools to identify trends, anomalies, and potential water leaks.
9. ALERTS AND NOTIFICATIONS:
Set up automated alerts and notifications for abnormal water consumption patterns, leaks, or sensor malfunctions. Define thresholds for triggering alerts.
10. USER ACCESS AND PERMISSIONS:
Provide access to the monitoring system for relevant stakeholders, such as facility managers, water conservation

REAL TIME TRANSIT INFORMATION PLATFORM

Real-time information, broadly defined, means any information available to transit providers or customers about the current status of vehicles, including approximate locations and predictive arrival times. Most real-time information relies on automatic vehicle location (AVL) and Global Positioning Systems (GPS) in order to estimate approximate arrival times for passengers and transit system operators. Passengers access real-time arrival and departure information through dynamic signs at stops and stations, or through the Internet at home or on smartphones. As smartphones become more prevalent, they have made access to third-party scheduling information and apps highly accessible for passengers.

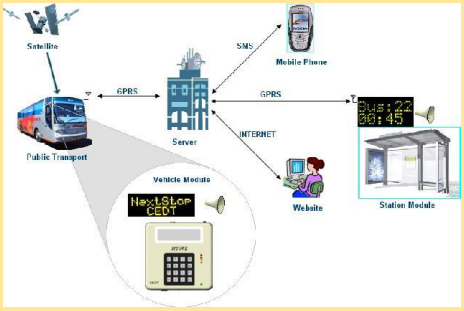
Mobile Technology

Because cellular phones and smartphones are so prevalent today, they can be very useful for disseminating real-time transit information. Mobile phones allow passengers to use SMS (or Short Message Service) to access schedule and real-time information via text message. This is a two-way method of communicating wherein the passenger can send a text message to an agency, usually with a code for the stop they want information about. The agency then automatically sends a response with the next bus' arrival times. These services do not necessarily always use real-time information, instead responding with the next scheduled bus arrival time. However, real time information makes texting more useful to customers.

- **Computer vision:** By utilizing data from parking lot cameras, computer vision technology identifies vacant parking spaces.
- **Deep learning:** Deep learning employs AI neural networks that have been exposed to comprehensive parking data. Similar to humans, these systems learn from experience, becoming more accurate as they collect and analyze more data.
- **Ground sensors:** Ground sensors utilize radar technology to detect parking space availability. These sensors are strategically placed across the parking lot floor.

- **Parking counter systems:** These legacy smart parking technologies count the number of vehicles entering or exiting a parking lot.
- **Automated parking lots:** Drivers leave their vehicles in a designated area, where sensors and lasers scan and measure the vehicle's dimensions. Subsequently, a moving platform lifts and transfers the vehicle to an available parking space.
- **Self-driving cars:** Many cars now come equipped with self-parking features, enabling drivers to park their vehicles automatically without being behind the wheel. This advancement streamlines the parking process and maximizes the utilization of parking spaces.

- **The Internet of Things (IoT):** The IoT refers to the interconnectedness of various devices via online communication. IoT systems enable drivers to access real-time information about available parking spaces through websites or dedicated apps. With this technology, drivers can plan their parking ahead of time, significantly reducing the time spent searching for a spot.
- The advantages of IoT in parking extend beyond assisting drivers in finding parking spaces. These systems also help businesses optimize parking lot occupancy, promote their services across multiple platforms, and contribute to a reduction in carbon emissions resulting from drivers aimlessly circling in search of parking.



INTEGRATION APPROACH

Using Iot Sensor

Determine how Iot sensor will send data to the data-sharing platform :

Iot sensors typically send data to a data sharing platform using the following steps:

1. **Sensor Data Acquisition:** IoT sensors collect data from the physical world, such as temperature, humidity, motion, or any other relevant parameters, depending on their purpose.

2. **Data Preprocessing:** Raw sensor data may require preprocessing to clean, filter, or format it for transmission and analysis. This step can include data normalization and error correction.

3. **Data Transmission:** Sensors transmit data to the data sharing platform using various communication protocols, such as Wi-Fi, cellular networks (3G/4G/5G), Bluetooth, Zigbee, LoRaWAN, or MQTT. The choice of protocol depends on the application and the range of communication required.

4. Data Packaging: Data is often packaged into structured formats like JSON or XML before transmission. This ensures that the data is easily interpretable by the platform.

5. Security Measures: To protect data integrity and privacy, encryption and authentication mechanisms are often employed during data transmission. This helps prevent unauthorized access or tampering of sensor data.

6. Data Gateway (optional): In some cases, a data gateway or edge device may be used to collect data from multiple sensors and transmit it to the central platform. This can help reduce the load on individual sensors and optimize data transfer.

7. Data Reception: The data sharing platform receives incoming data from the sensors. This platform can be cloud-based, on-premises, or a hybrid solution.

8. Data Processing and Analysis: The platform may perform real-time or batch processing on the data to extract insights, detect anomalies, or trigger actions based on predefined rules and algorithms.

9. Data Sharing: Depending on the use case, the data can be shared with authorized users, other systems, or applications via APIs, dashboards, or reports.

10. Data Storage: The received data is typically stored in databases or data lakes for long-term retention and analysis

11. Data Retention and Compliance: Data retention policies and compliance with data privacy regulations must be followed. Data may be archived or deleted as necessary.

12. Monitoring and Maintenance: Continuous monitoring of sensors, data transmission, and the platform's health is essential to ensure the system's reliability and performance.

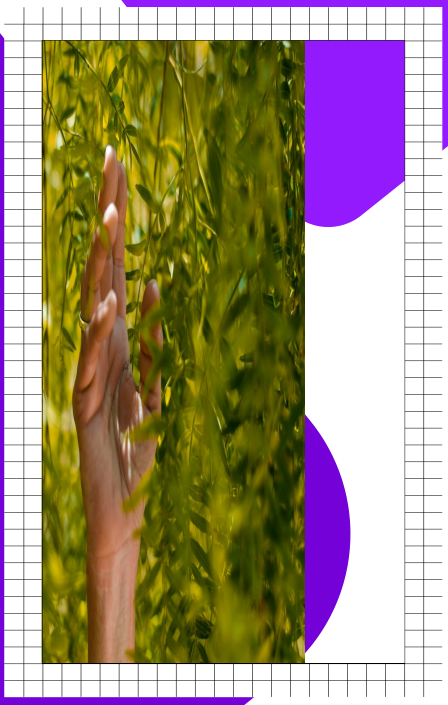
The specific implementation details may vary based on the IoT ecosystem, hardware, and software components chosen for your project. Proper planning and selection of appropriate technologies are crucial to ensure efficient and secure data transmission from IoT sensors to the data sharing platform.

Revolutionizing Water Management: Transforming Design into Innovation

1. Introduction: Importance of revolutionizing water management

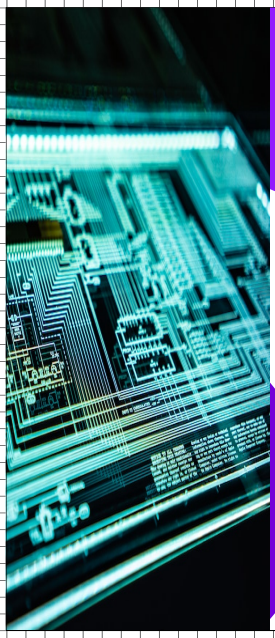
Water management is undergoing a groundbreaking transformation, as design and innovation converge to revolutionize this vital resource. With each passing day, new advancements in technology and research are reshaping the way we approach water conservation, treatment, and distribution. This shift towards innovative design holds the promise of more sustainable, efficient, and resilient water systems, ensuring a brighter future for generations to come.

2. Current challenges in water management



Despite the exciting advancements in water management, there are still pressing challenges that need to be addressed. Issues such as aging infrastructure, population growth, climate change, and water scarcity pose significant hurdles in achieving sustainable water systems. These challenges require innovative solutions and collective efforts from governments, industries, and communities to ensure the responsible and efficient management of this essential resource.

3. Design thinking in water management innovation

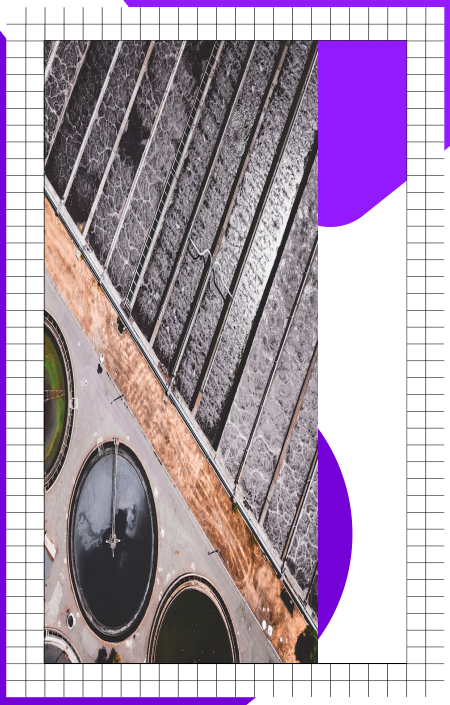


Design thinking is a crucial aspect of water management innovation. By adopting a human-centered approach, we can identify and understand the needs and challenges of various stakeholders, including communities, industries, and governments. This process enables us to create innovative and sustainable solutions that address the pressing issues in water management and ensure a more efficient and responsible use of this vital resource.

4. Understanding the role of technology in water management

Technology plays a pivotal role in revolutionizing water management. From advanced sensors and data analytics to remote monitoring systems, technology offers solutions for optimizing water distribution, detecting leaks, and managing water quality. By embracing technological advancements, we can enhance efficiency, reduce wastage, and ensure the sustainable management of this precious resource.

5. Exploring sustainable solutions for water conservation

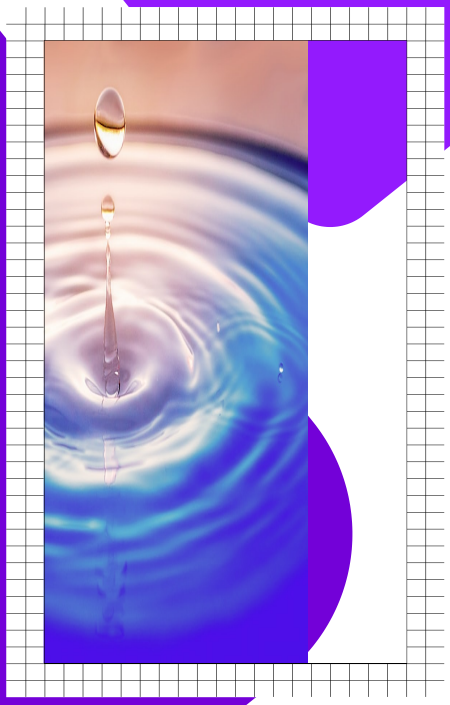


To address the growing water scarcity crisis, it is crucial to explore sustainable solutions for water conservation. This includes implementing rainwater harvesting systems, promoting water-efficient practices in agriculture and industry, investing in wastewater treatment and recycling, and adopting innovative irrigation techniques. By combining these strategies, we can achieve long-term water security and protect our environment for future generations.

6. Implementing innovative strategies in water treatment

To transform water management, it is essential to implement innovative strategies in water treatment. This includes adopting advanced water purification technologies, such as membrane filtration and reverse osmosis, to ensure the availability of clean and safe drinking water. Additionally, integrating smart water monitoring systems can help optimize water usage and identify leakage, contributing to efficient water management practices.

7. Case studies: Successful water management transformations



Case studies offer valuable insights into successful water management transformations. By examining real-world examples, we can learn how innovative design concepts and technologies have been applied to address specific challenges. These case studies highlight the benefits, outcomes, and replicable strategies that can be implemented in similar situations to revolutionize water management practices.

8. Overcoming obstacles to change in water management

Implementing innovative water management practices often faces obstacles that must be overcome. These obstacles may include resistance to change, lack of funding or resources, regulatory constraints, and the need for stakeholder collaboration. By identifying these challenges and finding effective solutions, we can transform the design into innovation and create a sustainable future for water management.

9. Future trends in water management innovation

As we look towards the future, several trends emerge in water management innovation. These include the use of advanced technologies like IoT and AI, the integration of renewable energy sources, the adoption of circular economy principles, and the emphasis on community engagement and education. By staying ahead of these trends, we can revolutionize water management and ensure a resilient and sustainable future for our global water resources.

10. Conclusion: Driving the future of water management

In conclusion, by embracing innovative design and incorporating advanced technologies, renewable energy, circular economy principles, and community engagement, we can drive the future of water management. Together, we can create a sustainable and resilient system that ensures the availability and quality of water for future generations. Let's revolutionize water management and make a positive impact on our global water resources.

Hardware Setup:

Choose suitable IoT hardware, like Raspberry Pi, Arduino, or specific IoT boards.
Connect sensors for measuring water parameters (e.g., water level, flow rate, quality).
Connect actuators (valves, pumps) for controlling water flow.

Software Dependencies:

Ensure your IoT device is running an operating system (e.g., Raspbian for Raspberry Pi).
Install necessary Python libraries for IoT, such as RPi.GPIO or Adafruit CircuitPython (for Raspberry Pi) or libraries specific to your hardware.

Sensor Data Acquisition:

Write Python code to read data from your sensors (e.g., ultrasonic sensor for water level, flow rate sensor, water quality sensor).

Data Processing:

Process the sensor data to extract relevant information.
Implement algorithms for anomaly detection and error handling.

Data Storage:

Store data in a database or a file for historical analysis.

Communication:

Implement a communication protocol (e.g., MQTT) to send data to a central server or cloud platform.
Use Python libraries like paho-mqtt for MQTT communication.

Remote Control:

Create functions to remotely control actuators like valves and pumps.

User Interface (Optional):

Develop a web-based dashboard or mobile app to monitor and control the system.
Utilize Python web frameworks like Flask or Django.

Automation and Logic:

Implement logic for smart water management, such as scheduling water flow, optimizing water usage, and responding to alarms or alerts.

Security:

Ensure data security by implementing encryption and authentication mechanisms.
Regularly update your device to patch security vulnerabilities.

Testing and Debugging:

Test your system thoroughly, simulate different scenarios, and debug issues.

Deployment:

Deploy your IoT device in the target environment.

Monitoring and Maintenance:

Set up monitoring for your IoT device's health and performance.
Establish a maintenance plan to address hardware and software issues.

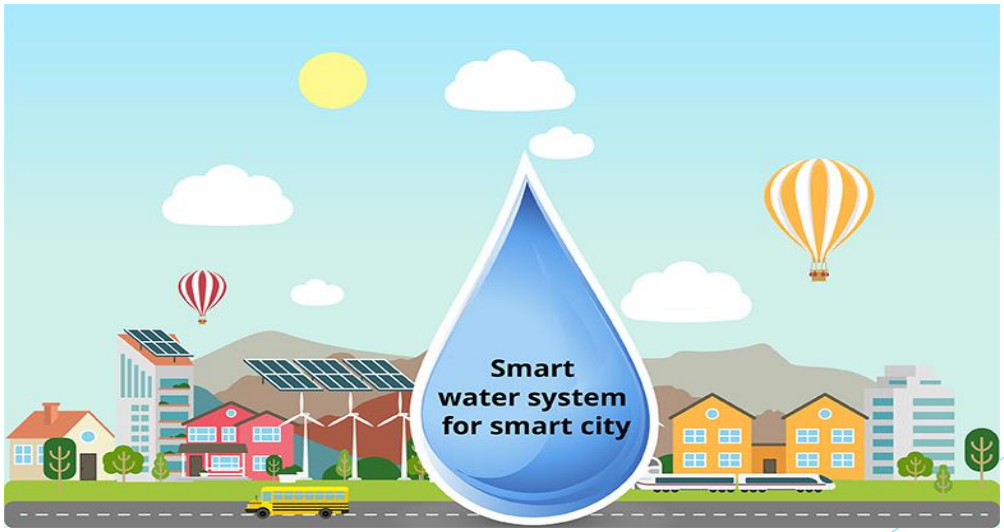
Write Python code to read data from your sensors (e.g., ultrasonic sensor for water level, flow rate sensor, water quality sensor).

To read data from sensors in a Python script, you need to interact with the sensors through their respective interfaces or GPIO pins. Below is a basic example of reading data from a hypothetical ultrasonic sensor, a flow rate sensor, and a water quality sensor. Please note that you'll need to use the libraries specific to your sensors and hardware.

```
# Import necessary libraries for sensor communication
import RPi.GPIO as GPIO  # For Raspberry Pi GPIO control
import time              # For time delays

# GPIO pins for the sensors
ultrasonic_trigger_pin = 23
ultrasonic_echo_pin = 24

flow_rate_sensor_pin = 17

# Initialize GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(ultrasonic_trigger_pin, GPIO.OUT)
```

```
GPIO.setup(ultrasonic_echo_pin, GPIO.IN)
GPIO.setup(flow_rate_sensor_pin, GPIO.IN)

# Function to read data from ultrasonic sensor
def read_ultrasonic_sensor():
    GPIO.output(ultrasonic_trigger_pin, True)
    time.sleep(0.00001)
    GPIO.output(ultrasonic_trigger_pin, False)

    while GPIO.input(ultrasonic_echo_pin) == 0:
        pulse_start_time = time.time()

    while GPIO.input(ultrasonic_echo_pin) == 1:
        pulse_end_time = time.time()

    pulse_duration = pulse_end_time - pulse_start_time
    distance = (pulse_duration * 34300) / 2  # Speed of sound = 34300 cm/s
    return distance

# Function to read data from flow rate sensor
def read_flow_rate_sensor():
    # Implement the code to read data from the flow rate sensor
    # This will depend on the specific sensor you are using

# Function to read data from water quality sensor
def read_water_quality_sensor():
    # Implement the code to read data from the water quality sensor
    # This will depend on the specific sensor you are using

try:
    while True:
        # Read data from sensors
        water_level = read_ultrasonic_sensor()
        flow_rate = read_flow_rate_sensor()
        water_quality = read_water_quality_sensor()

        # Print the sensor readings
        print(f"Water Level: {water_level} cm")
        print(f"Flow Rate: {flow_rate} L/min")
        print(f"Water Quality: {water_quality}")

except KeyboardInterrupt:
    GPIO.cleanup()
```

Smart water management using wokwi sensors



INTRODUCTION

Developing a complete smart water management system involving hardware like WOKWI sensors web technologies requires a substantial amount of code and configuration. Writing a comprehensive system involves multiple programming languages, databases, and frameworks.

Simulation using wokwi

CREATE A VIRTUAL CIRCUIT:

1. Visit the Wokwi website and create a new circuit.
2. Add components representing WOWKI sensors to your circuit. Wokwi provides a wide range of components that you can use in your virtual circuit.

1.CONNECT COMPONENTS:

1. Connect the WOWKI sensors to a microcontroller (like Arduino) virtually, simulating the actual hardware connections.

3. Create a circuit:

- Connect the sensors to the microcontroller according to their datasheets and specifications.
- Use Wokwi's online circuit simulation tool to create a circuit that includes the selected sensors, microcontrollers (such as Arduino), and any other components required for your project.

4. Write simulation code:

•Write code for the microcontroller to read data from sensors and process it. Use conditional statements and algorithms to manage water resources efficiently.

•Simulate sensor readings and test your code within the Wokwi simulation environment.

•For example, using Arduino code, you can read sensor values, process them, and control actuators (like pumps or valves) based on the readings.

To create a water management simulation using a sensor and the Wokwi platform, you'll need to use the Arduino platform and simulate it in the Wokwi environment. Here's a basic example using a virtual water level sensor:

1.Set up an Arduino project using the Arduino IDE or online Arduino editor.

2.Create a new Arduino sketch and use the following code to simulate a water management system with a virtual water level sensor:

```
const int sensorPin = A0;
// Analog pin for the water level sensor
const int pumpPin = 2;
// Digital pin for the water pump
int waterLevel = 0;
void setup() {
  pinMode(sensorPin, INPUT);
pinMode(pumpPin, OUTPUT);  Serial.begin(9600);
}void loop() {
// Read the water level from the virtual sensor
waterLevel = analogRead(sensorPin);
// Simulate a water management system based on
the water level
```

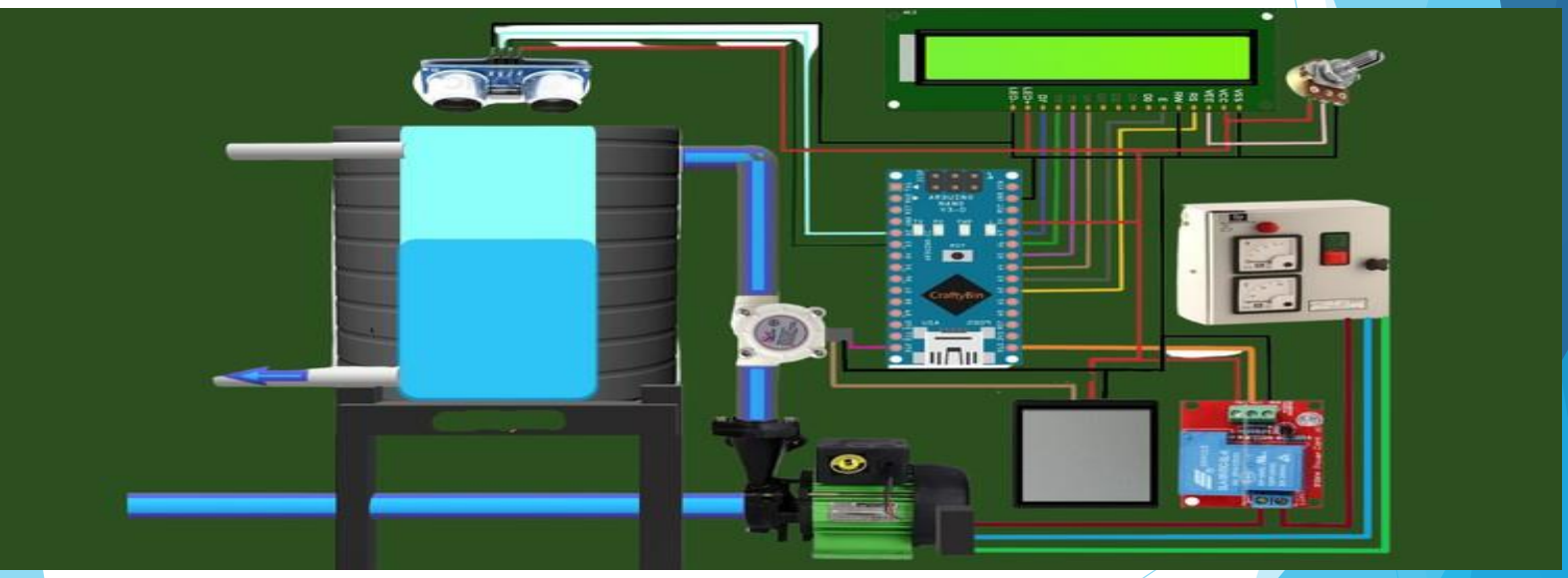
```
if (waterLevel < 300) {
  // If water level is low, turn on the water pump
digitalWrite(pumpPin, HIGH);
Serial.println("Water level is low. Pump is ON."); } else {
 // If water level is sufficient, turn off the water pump
digitalWrite(pumpPin, LOW);
Serial.println("Water level is sufficient. Pump is OFF."); }
delay(1000);
// Simulate readings every 1 second}
```

3.In the Arduino IDE or online Arduino editor, select the "Tools" menu, and choose your target board and port.
4.Click the "Verify" button to compile the code, and if there are no errors, click the "Upload" button to upload the code to your virtual Arduino board in the Wokwi simulation.
5.In the Wokwi platform, you can add a virtual water level sensor and a pump to your simulated circuit. Interact with the circuit, and you'll see the water management simulation in action.

SIMULATION CODE FOR WATER MANAGEMENT

```
let waterLevel = 50; // Initial water level in percentage

function updateWaterLevel() {
  document.getElementById("water-level-value").textContent = waterLevel + "%";
}

function togglePump() {
  if (waterLevel < 100) {
    waterLevel += 10;
  } else {
     waterLevel = 0;
   }
   updateWaterLevel();
}

document.getElementById("pump-button").addEventListener("click", togglePump);
```

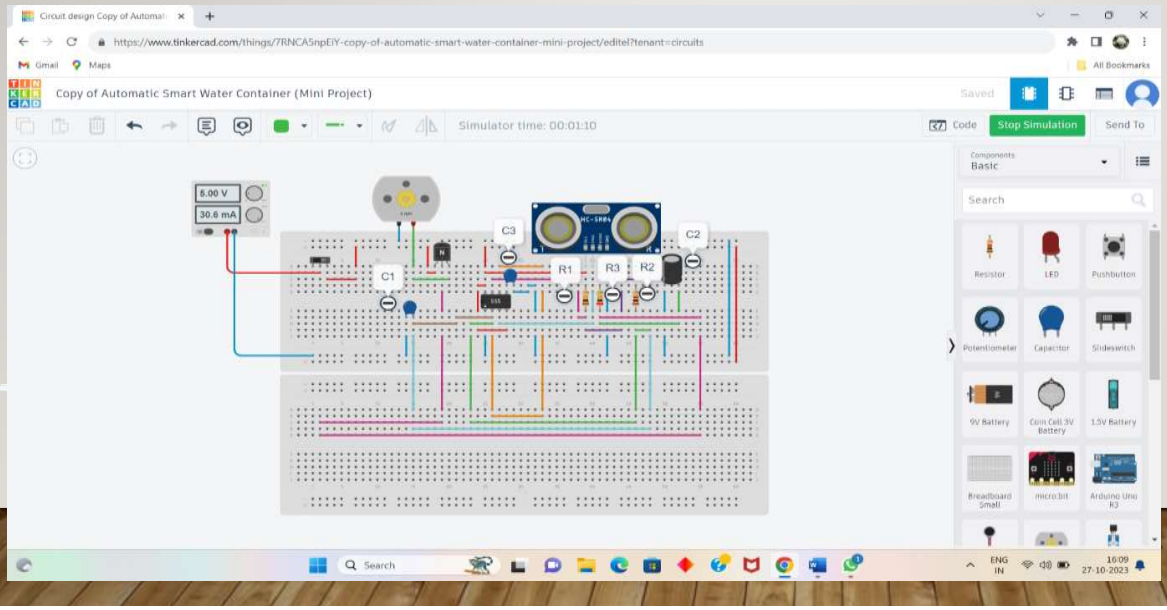
SIMULATON DIAGRAM



URL for Tinkercad Project of Smart water Foundation:

https://www.tinkercad.com/things/7RNCA5npEiY-copy-of-automatic-smart-water-container-mini-project/editel?tenant=circuits

Screenshot of Smart water water Foundation Circuit(Tinkercad):



Code for Running above Circuit:

```
# Import the necessary library
import tinkercad
import requests  # Add the requests library for HTTP requests
# Initialize the Tinkercad API
tc = tinkercad.Tinkercad(username="Kanimozhi01", password="Kanimozhi")
# Find the simulation you want to run (use the correct ID)
simulation_id = "2303774"
# Get the simulation
simulation = tc.get_simulation(simulation_id)
# Define the ThingSpeak API parameters
thingspeak_api_key = "G5YN4PEKH3VEQ0JA"
thingspeak_url = f"https://api.thingspeak.com/update?api_key={thingspeak_api_key}"
# Define the code to run in the Arduino
arduino_code = """
#include <Ultrasonic.h>
```

```
Ultrasonic ultrasonic(2, 3);  // Trigger (pin 2), Echo (pin 3)
void setup() {
  Serial.begin(9600);
}
void loop() {
  float distance = ultrasonic.Ranging(CM);
  Serial.println(distance);
  // Send data to the computer (Python script)
  Serial.print("D:");
  Serial.println(distance);

  delay(1000);
}
"""
```

```
# Upload and run the code in the simulation
simulation.run_code(arduino_code)
# Monitor the water level and send data to ThingSpeak
while True:
    data = simulation.get_serial_data()
    if data and data.startswith("D:"):
        distance = float(data[2:])
        print(f"Water level: {distance} cm")
        # Send data to ThingSpeak
        try:
            response = requests.get(f"{thingspeak_url}&field1={distance}")
            if response.status_code == 200:
                print("Data sent to ThingSpeak successfully.")
            else:
                print("Failed to send data to ThingSpeak.")
        except Exception as e:
print("Error sending data to ThingSpeak:", str(e))
```

Thingspeak Channel stats of Smart Water foundation:

Platform UI code for Smart Water Foundation:

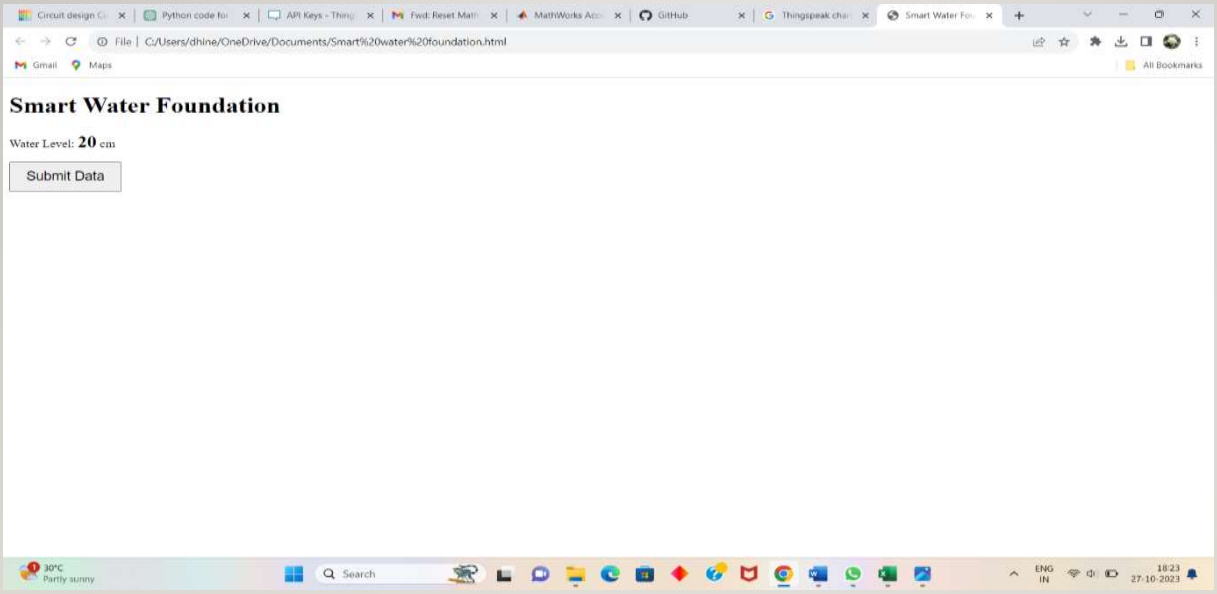
```
<!DOCTYPE html>
<html>
<head>
  <title>Smart Water Foundation</title>
  <style>
    /* Style for the water level display */
    #water-level {
      font-size: 24px;
      font-weight: bold; }
    /* Style for the submit button */
    #submit-button {
      padding: 10px 20px;
      font-size: 18px;
    }
  </style>
</head>
```

```
<body>
    <h1>Smart Water Foundation</h1>
    <p>Water Level: <span id="water-level">Loading...</span> cm</p>
    <button id="submit-button" onclick="sendDataToServer()">Submit Data</button>
    <script>
        // Function to update water level data from the server
        function updateWaterLevel() {
            // You can use AJAX or fetch to get data from your server
            // Replace the URL with the actual endpoint that provides water level data
            fetch('/getWaterLevelData')
                .then(response => response.json())
                .then(data => {
                    document.getElementById('water-level').textContent = data.waterLevel + " cm";
                })
                .catch(error => {
                    console.error('Error fetching water level data:', error);}).}
```

```
// Function to send data to the server (e.g., to trigger data collection)
    function sendDataToServer() {
      // You can use AJAX or fetch to send data to your server
      // Replace the URL with the actual endpoint that handles data submission
      fetch('/submitData', { method: 'POST' })
        .then(response => {
          if (response.status === 200) {
            console.log('Data submitted successfully');
          } else {
            console.error('Data submission failed with status:', response.status);
          } .
      catch(error => {
          console.error('Error submitting data:', error);
        });
    }
```

```
 // Update water level initially and then at regular intervals
      updateWaterLevel();
      setInterval(updateWaterLevel, 10000);  // Update every 10
seconds
   </script>
</body>
</html>
```

Output for above Program:




THANK
YOU