SMART WATER MANAGEMENT

Date	**31-10-2023**
Team ID	**666**
Team Name	**Proj_223434_Team_2**
Project Title	**Smart Water Management**

PROBLEM STATEMENT:

Overflowing water tanks contribute to the majority of wastage of water in residences. Most of the time, it is due to human error that tanks are allowed to overflow. So, by automating the process and by removing the human error from the picture, we can reduce the wastage of water by quite a bit. Another reason for wastage of water is due to leaks that occur in pipes. These leaks might not be detected immediately, and by the time the leak is detected, a lot of water will already be wasted. Sometimes the leak might not be detected until after a long time-wasting a lot of water in the process. By placing sensors in the pipes, we can detect the flow of the water in pipes and immediately detect any leaks in the pipe and immediately notify the user. By using electrically controlled valves, we can cut off the water supply to minimize the wastage of water. The utilization of water can be minimized by reducing the flow of water from the tank itself.

PROBLEM IDENTIFIED:

In smart water monitoring IoT projects, several challenges are often encountered. One significant problem is the integration of diverse sensor data from different sources and formats. Water monitoring systems typically rely on a variety of sensors, such as flow meters, water quality sensors, pressure sensors, and more. These sensors may use different communication protocols and data formats, making it challenging to aggregate and analyze data effectively. Inconsistent data can lead to inaccurate insights and decisions, undermining the project's goals of efficient water management and conservation. Furthermore, scalability is a common issue. As water monitoring projects expand to cover larger areas or more devices, managing the increasing volume of data becomes complex. This requires robust data storage, processing, and analytics solutions to handle the growing demands. Security is another critical concern. Ensuring the confidentiality and integrity of sensitive water data is essential to prevent unauthorized access and potential tampering. Implementing robust security measures is crucial to protect against data breaches and cyberattacks. Finally, there is a need for standardized communication protocols and data formats within the IoT ecosystem to enable seamless data exchange and interoperability among different devices and systems. Addressing these challenges is crucial for the successful implementation of smart water monitoring IoT projects and the sustainable management of water resources.

INTRODUCTION:

Water supply is the most important thing in daily home activity. We commonly supply the water by pumping the groundwater to fill a water tank. However, the utilization of non-automated switch used to turn on and turn off a pumping machine sometimes causes either the water spills or a wasteful electrical consumption. In this work, an automated water tank filling system will be proposed. By applying an ultrasonic sensor, an ultrasonic transmitter is mounted on the top of the tank and transmits an ultrasonic pulse down into the tank. The transmitter is programmed to automatically determine the liquid level and switch the pumping machine. The dynamics of water flow and liquid level during filling and draining the water tank will be reported. We hope to this system, people will enjoy supplying water without their worries related to water spills and a wasteful electrical consumption.

LITERATURE SURVEY

1. INTERNET OF THINGS (IOT)-BASED WASTEWATER MANAGEMENT IN SMART CITIES:

Abdullah I. A. Alzahrani, Sajjad Hussain Chauhdary, and Abdulrahman A. Alshdadi(2023) Pollutant extraction and refinement are accomplished through the use of wastewater management from waste water or sewage that is environmentally safe to be recycled into the water supply effects. Due to the current deteriorating environmental condition, new approaches and techniques are essential to ensure safe and intelligent wastewater management systems in smart cities. Mobile sensor Technologies for wastewater treatment that are in the horizon include networks and the Internet of Thing (IoT).The thorough literature review creates a theoretical basis for the Internet of Things. Block chain-based IoT- based wastewater management system (IoT-WMS) for smart cities. Currently, information is being stored on blockchain technology to create an incentive scheme for promoting the recycling of wastewater. The amount and quality of reclaimed wastewater are concerned; Tokens are distributed to businesses and households in.

2. AN IOT-BASED WATER MANAGEMENT SYSTEM FOR SMART CITIES:

Immanuel Savio and Udit Chakraborty(2021) Water conservation is one of the most pressing issues in today's world, as environmental conditions are worsening at an alarming rate. Smart cities, as opposed to traditional living systems, are at the forefront of innovation terms of both connectivity and technical growth. The key concept is to employ There is technology available to make life easier while causing less environmental damage. An Water management system based on the Internet of Things (IoT) and data analytics (DA).Will serve as a foundation for future study and deployment of data and IoT can be utilized to do this. The Internet of Things (IoT) is proposed in this study and a water distribution cum management system based on data analytics (DA) that would help with proper water distribution based on.

3. SMART WATER MANAGEMENT SYSTEM USING CLOUD COMPUTING AND IOT:

Aniket Bhosale, Shashank Roy, Arjun Singh, Sureshkumar Natarajan(2017) We waste large amounts of water every day. As a result, we are suggesting a project that will make smart use of water. This system is simple to install in a variety of settings. The user is continuously informed of the water level through sensors that are installed in the tank at regular intervals. Users can see the water level on a computer system that is connected to the Internet, and this information will be updated in the cloud. The valve's operation will be automatically adjusted based on the amount of water in the tank. The valve will automatically open at low water levels and close when the tank is approximately to.

4. WATER MONITORING SYSTEM EMBEDDED WITH INTERNET OF THINGS (IOT) DEVICE:

N S Kamaruidzaman and Siti Nazahiyah Rahmat(2023) Urbanization and population growth are thought to increase water consumption, which may have an impact on water source. Consequently, it must be effectively managed. In order to manage and conserve the limited resources, this article examines how Internet of Things (IoT) devices are used to monitor water systems. The most prevalent wireless data system is most often associated with IoT technologies. Acquisition for continuous observation and monitoring. A wide range of monitoring sensors and networks, including computer systems, wireless sensor networks, and cloud computing, are used in real-time monitoring. Reading water quantity and quality has traditionally been a time-consuming and expensive task. Using IoT, human involvement will be reduced, and the majority of procedural decisions in a typical.

5. SMART WATER MANAGEMENT USING IOT TECHNOLOGY:

Prof. Priyanka Bagul, Medha Patil, Aarti Thakur, and Shubham Thakur(2022) The water is one of life's most important components from giving the human body nourishment to using it. Water is important for man daily functions. Water must therefore be conserved and used sensibly. A smart water management system is essential in the modern world, where the Internet of Things (IoT) is playing a significant role in every industry. One of the finest options for ensuring effectiveness and sustainable water use is a system. A system based on IoT will bringing clarity to the system's procedures, will enabling real-time monitoring and detection issues right away. The system will also be automated that will lessen.

6. SMART WATER MANAGEMENT USING IOT ENVIRONMENT:

A. Madhuraveni, G. Athithan, S. Thilagavathi, R. Vignesh(2018) Serves to govern the appropriate upkeep of water tank data for the monitoring section with proper record

updating.The issue impacts numerous water management procedures, including water dust was produced in the process of consumption, distribution, and water tank. These issues can be resolved by putting in place a suitable monitoring system and system for updating information. a group of sensors such as turbidity, pH sensors, water flow sensors, and salt sensors were used. This sensor communicates with the monitor section and provides information about the water level in the tank. To continue the tank free of microorganisms and bacteria, the chlorine powder if any variations in the PH value are discovered, is sprayed. In case the water level.

7. IOT-BASED SMART WATER MANAGEMENT SYSTEMS FOR RESIDENTIAL BUILDINGS IN SAUDI ARABIA:

Rayed Alghamdi and Sunil Kumar Sharma(2022) The water is a valuable resource that may be managed carefully. In a society where water tanks, motors, and pumps are commonplace, efficient water use necessitates computerized household water supply control. In nations like Saudi Arabia, both the government and the populace depend on effective water management. Providing a consistent, high-quality, affordable water supply is the problem. This study presents a smart water management (IoT-SWM) system that might be applied to buildings that do not have a continuous water supply but do have water stored beneath them in sizable tanks. Data on water use is gathered by the GSM module from every residence in a neighbourhood and sent to the cloud for analysis. A hybrid application that employs an inspection mode is a smart water grid.

8. WATER MANAGEMENT SYSTEM USING IOT:

P Kavya Nikeeta, N Kiranmayie, P Manishankar, Ch Krishna Nitej, V Murthy(2020) Manage the water supply, find leaks, communicate user statistics, and manage water flow. Leaks will be tracked by sensors that are installed in the pipes can use IOT to automatically shut off the supply which are positioned in the above-ground tanks, which will keep an eye on the water and transmit the information to the microcontroller. The stream of to reduce water use, the water is regulated. Information is delivered to the cloud, accessible by the user.

9. IOT-BASED WATER MANAGEMENT SYSTEMS: SURVEY AND FUTURE RESEARCH DIRECTION:

Shereen Ismail, Diana W. Dawoud, Nadhem Ismail, Ronald Marsh, and Ali S. Alshami(2022) The Internet of Things (IoT) and sustainable solutions have been presented as the next generation of techniques for controlling and keeping an eye on priceless natural resources like water. Smart IoT-based water management and monitoring system designs have received a lot of attention in research for a variety of applications, including those in the agricultural, industrial, residential, and crude oil

exploration sectors. This work includes an exhaustive review of 43 articles published between 2014 and 2022 for systems suggested to manage and monitor water in four different sectors: agricultural, industrial, residential, and oilfield sectors. This work differs from other surveys accessible in the literature in that it covers all four sectors. We also suggest a new optical water management method in this work to address a gap in.

10. OVERVIEW OF IOT BASED WATER MANAGEMENT SYSTEM:

Sanjay Kharat, Prof. Sumera Ali, Prof. Devendra L. Bhuyar(2020) In contrast, there is less usable water available in India relative to its population. India contains about 4% of the world's potable water. A human beings need for water is fundamental. On the other side, India's having problems with the uneven distribution of rainfall. It is certain that people would need water for drinking and other daily activities. Water is frequently squandered because it is not used in the right quantity when it is available. Additionally, it has been noted that the drainage water is contaminating clean water sources by combining with them.A result of inadequate drainage.

DESIGN THINKING

Design Thinking Approach

1. IoT Integration:

- Incorporate Internet of Things (IoT) technology to enable water overflow detection involves connecting sensors and communication mediums.
- Implement water level sensors at critical locations where overflow is a concern such as tank.
- Sensors capable of accurately measuring water levels and reduces the risk of water wastage through alerting.

2. Energy Efficiency:

- Choose energy-efficient sensors for water level monitoring.
- Low power sensors can operate for extended periods without consuming excessive energy.
- IoT can also monitor energy consumption in the water overflow detection system itself, helping identify areas where energy efficiency can be improved.

3. Interactive Features:

- Create interactive Display that shows real-time water level data using LCD Display.
- Allow users to set their own threshold levels for alerts.

4. Environmental Sustainability:

- The project helps conserve water resources by preventing overflow and reducing wastage, contributing to the sustainable use of a precious resource.

- ❖ Through IoT integration, the project can optimize energy usage, reducing the carbon footprint associated with water management operations**.**

5. Maintenance and Monitoring:

- ❖ Implement a predictive maintenance system that uses data from sensors to identify and address issues before they become major problems
- ❖ Set up remote monitoring and diagnostics capabilities to reduce downtime and maintenance costs.

6. Public Awareness:

- ❖ The project's interactive features can raise public awareness about water conservation, promoting responsible water usage.

7. Scalability:

- ❖ It can be scaled to accommodate growing water management needs, making it adaptable to changing circumstances.

8. Community Involvement:

- ❖ Involving the community in water conservation efforts through interactive features fosters a sense of ownership and sustainability.

THE PROJECT'S OVERVIEW

ESP32 Microcontroller:

Use an ESP32 development board as the central controller for data collection and transmission.

Sensor:

Ensure compatibility with the ESP32 voltage levels.

User Interface:

Create a web or mobile app that can communicate with the ESP32 to display real-time data to users.

Alerts and Notifications:

Implement alerting mechanisms through the LED to notify users of critical water quantity.

Power Management:

Optimize the ESP32's power consumption by using deep sleep modes and efficient coding practices, especially if your project is battery-powered.

Sensor-Specific LDs:

Assign LEDs to specific water quality or quantity sensors.

For example:

A blue LED to indicate water temperature within an acceptable range.

A yellow LED for water turbidity within a specified threshold.

A red LED for low water level.

Web Interface Control:

Enabling users to remotely start or stop the water flow via a web-based control panel. The HTML and JavaScript code provided constitute the foundation for this interaction. Users can interact with the control panel buttons labeled "Water Monitor" and "Stop Water Flow" to trigger these actions.

Serial Communication for Control:

Utilizing serial communication between the ESP 32 board and the web interface, enabling the transmission of control commands from the web panel to the ESP 32 for water monitoring operation.

IOT DEVICE SETUP:

COMPONENTS:

1.ESP 32

2.Ultrasonic sensor

3.Buttons

4.LED(for status indication)

5.OLED Display(to display the water level)

6.Power Supply

Steps:

Connect Ultrasonic Sensor:

VCC to Vin (ESP 32)

Trig to D27(ESP 32)

Echo to D26

GND to GND(ESP 32)

Connect 1st Button:

btn1:1.l to ESP 32 (D32)

btn1:2.l to Blue LED Cathode

btn1:2.r to btn2:2.l

Connect 2nd Button:

btn2:1.l to ESP 32(D33)

btn2:2.r to btn2:2.l

Connect 3rd Button:

btn3:1.l to ESP 32(D12)

btn3:2.r to GND(ESP 32)

Connect LED:

Blue LED: Anode(+ve) to ESP 32(D25) and Cathode(-ve) to btn1:2.l

Red LED: Anode(+ve) to ESP 32(D13) and Cathode(-ve) to btn1:2.l

Yellow LED: Anode(+ve) to ESP 32(D14) and Cathode(-ve) to btn1:2.l

Connect OLED Display:

GND to ESP 32(GND)

VCC to ESP 32(3.3V Power Supply)

SCL to ESP 32(D22)

SDA to ESP 32(D21)

Power Supply:

Power the ESP 32 board and ensure the power supply can handle the connected components requirements.

PLATFORM DEVELOPMENT:

Web Interface:

HTML: Create an HTML file for the user interface to control the water level and display sensor data.

JavaScript: Use JavaScript to handle button clicks, send commands to the ESP 32, and update the interface based on sensor readings.

HARDWARE INTEGRATION:

1. Communication:

Establish serial communication between the ESP 32 and the server using Eg.USB.

2. Sensor Data Transmission:

Send sensor data (water level readings) from the ESP 32 to the server for display on the web interface.

CODE IMPLEMENTATION

```
#define BLYNK_TEMPLATE_ID "TMPLlcLQu4bQ"
#define BLYNK_TEMPLATE_NAME "water monitor"
#define BLYNK_AUTH_TOKEN "OgvenxCWu9sG7-9deFGLFCLE4rWCGW7N"
char ssid[] = "Wokwi-GUEST";
char pass[] = "";
int emptyTankDistance = 150 ;
int fullTankDistance =  40 ;
int triggerPer =   10 ;
#include <Adafruit_SSD1306.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <AceButton.h>
```

```
using namespace ace_button;
#define TRIGPIN    27
#define ECHOPIN    26
#define wifiLed    2
#define BuzzerPin  13
#define RelayPin   14
#define ButtonPin1 12
#define ButtonPin2 33
#define ButtonPin3 32
#define fullpin    25
#define VPIN_BUTTON_1    V1
#define VPIN_BUTTON_2    V2
#define VPIN_BUTTON_3    V3
#define VPIN_BUTTON_4    V4
#define VPIN_BUTTON_5    V5
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32
#define OLED_RESET     -1
Adafruit_SSD1306
display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
float duration;
float distance;
int   waterLevelPer;
bool  toggleBuzzer = HIGH;
bool toggleRelay = false;
bool modeFlag = true;
bool conection = true;
String currMode;
char auth[] = BLYNK_AUTH_TOKEN;
ButtonConfig config1;
AceButton button1(&config1);
ButtonConfig config2;
```

```
AceButton button2(&config2);
ButtonConfig config3;
AceButton button3(&config3);
void handleEvent1(AceButton*, uint8_t, uint8_t);
void handleEvent2(AceButton*, uint8_t, uint8_t);
void handleEvent3(AceButton*, uint8_t, uint8_t);
BlynkTimer timer;
void checkBlynkStatus() {
  bool isconnected = Blynk.connected();
  if (isconnected == false) {
    digitalWrite(wifiLed, LOW);
    conection = true;
     }
  if (isconnected == true) {
    digitalWrite(wifiLed, HIGH);
    conection = false;
  }
}
BLYNK_WRITE(VPIN_BUTTON_3) {
  modeFlag = param.asInt();
  if(!modeFlag && toggleRelay){
      digitalWrite(RelayPin, LOW);
      toggleRelay = false;
    }
    controlBuzzer(500);
    currMode = modeFlag ? "AUTO" : "MANUAL";
}
BLYNK_WRITE(VPIN_BUTTON_4) {
  if(!modeFlag){
    toggleRelay = param.asInt();
    digitalWrite(RelayPin, toggleRelay);
    controlBuzzer(500);
```

```
  }
  else{
    Blynk.virtualWrite(VPIN_BUTTON_4, toggleRelay);
  }
}
BLYNK_WRITE(VPIN_BUTTON_5) {
  toggleBuzzer = param.asInt();
  digitalWrite(BuzzerPin, toggleBuzzer);
}
BLYNK_CONNECTED() {
  Blynk.syncVirtual(VPIN_BUTTON_1);
  Blynk.syncVirtual(VPIN_BUTTON_2);
  Blynk.virtualWrite(VPIN_BUTTON_3, modeFlag);
  Blynk.virtualWrite(VPIN_BUTTON_4, toggleRelay);
  Blynk.virtualWrite(VPIN_BUTTON_5, toggleBuzzer);
}
void displayData(){
  display.clearDisplay();
  display.setTextSize(3);
  display.setCursor(30,0);
  display.print(waterLevelPer);
  display.print(" ");
  display.print("%");
  display.setTextSize(1);
  display.setCursor(0,25);
  display.print(conection ? "OFFLINE" : "ONLINE");
  display.setCursor(60,25);
  display.print(currMode);
  display.setCursor(110,25);
  display.print(toggleRelay ? "! ON" : "OFF");
  display.display();
}
```

```
void measureDistance(){
  digitalWrite(TRIGPIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGPIN, HIGH);
  delayMicroseconds(20);
  digitalWrite(TRIGPIN, LOW);
  duration = pulseIn(ECHOPIN, HIGH);
  distance = ((duration / 2) * 0.343)/10;
  if (distance > (fullTankDistance - 10)  && distance < emptyTankDistance ){
    waterLevelPer = map((int)distance ,emptyTankDistance, fullTankDistance, 0, 100);
    Blynk.virtualWrite(VPIN_BUTTON_1, waterLevelPer);
    Blynk.virtualWrite(VPIN_BUTTON_2, (String(distance) + " cm"));
    if (waterLevelPer < triggerPer){
       if(modeFlag){
       if(!toggleRelay){
         controlBuzzer(500);
         digitalWrite(RelayPin, HIGH);
         toggleRelay = true;
         Blynk.virtualWrite(VPIN_BUTTON_4, toggleRelay);
       }
      }
      else{
       if (toggleBuzzer == HIGH){
         digitalWrite(BuzzerPin, HIGH);
         Serial.println(" BuzzerPin high");
       }
      }
    }
    if (distance < fullTankDistance){
      digitalWrite(fullpin, HIGH);
      if(modeFlag){
```

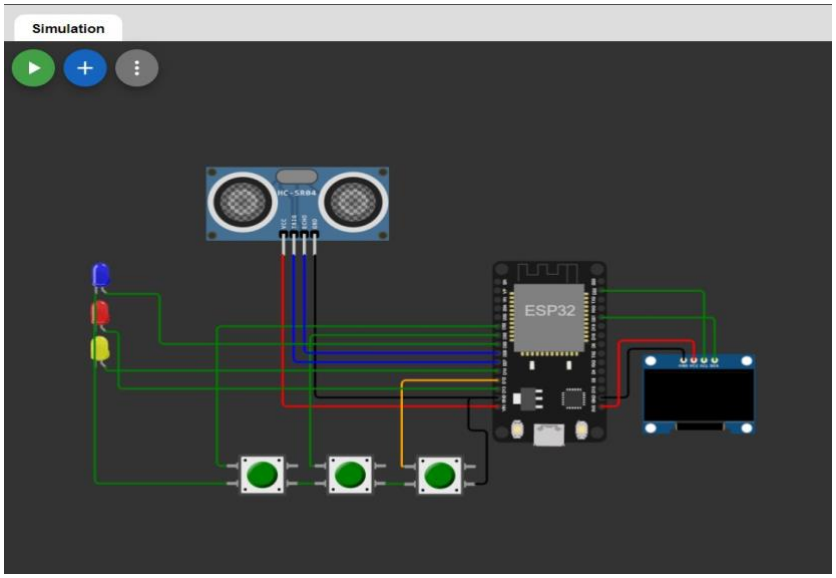
```
      if(toggleRelay){
        digitalWrite(RelayPin, LOW);
        toggleRelay = false;
        Blynk.virtualWrite(VPIN_BUTTON_4, toggleRelay);
        controlBuzzer(500);
      }
     }
     else{
      if (toggleBuzzer == HIGH){
      digitalWrite(BuzzerPin, HIGH);
      }
     }
   }
   if (distance > (fullTankDistance + 5) && waterLevelPer > (triggerPer + 5)){
    toggleBuzzer = HIGH;
    Blynk.virtualWrite(VPIN_BUTTON_5, toggleBuzzer);
    digitalWrite(BuzzerPin, LOW);
   }
   if (distance = fullTankDistance){
   Serial.println(" udh bang ");
 }
  }
  displayData();
  delay(100);
}
void controlBuzzer(int duration){
  digitalWrite(BuzzerPin, HIGH);
  Serial.println(" BuzzerPin HIT");
  delay(duration);
  digitalWrite(BuzzerPin, LOW);
}
```

```
void setup() {
  Serial.begin(9600);
  pinMode(ECHOPIN, INPUT);
  pinMode(TRIGPIN, OUTPUT);
  pinMode(wifiLed, OUTPUT);
  pinMode(RelayPin, OUTPUT);
  pinMode(BuzzerPin, OUTPUT);
  pinMode(fullpin, OUTPUT);
  pinMode(ButtonPin1, INPUT_PULLUP);
  pinMode(ButtonPin2, INPUT_PULLUP);
  pinMode(ButtonPin3, INPUT_PULLUP);
  digitalWrite(wifiLed, HIGH);
  digitalWrite(RelayPin, LOW);
  digitalWrite(BuzzerPin, LOW);
  config1.setEventHandler(button1Handler);
  config2.setEventHandler(button2Handler);
  config3.setEventHandler(button3Handler);
  button1.init(ButtonPin1);
  button2.init(ButtonPin2);
  button3.init(ButtonPin3);
  currMode = modeFlag ? "AUTO" : "MANUAL";
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
  delay(1000);
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.clearDisplay();
 WiFi.begin(ssid, pass);
  timer.setInterval(2000L, checkBlynkStatus);
  timer.setInterval(1000L,  measureDistance);
```

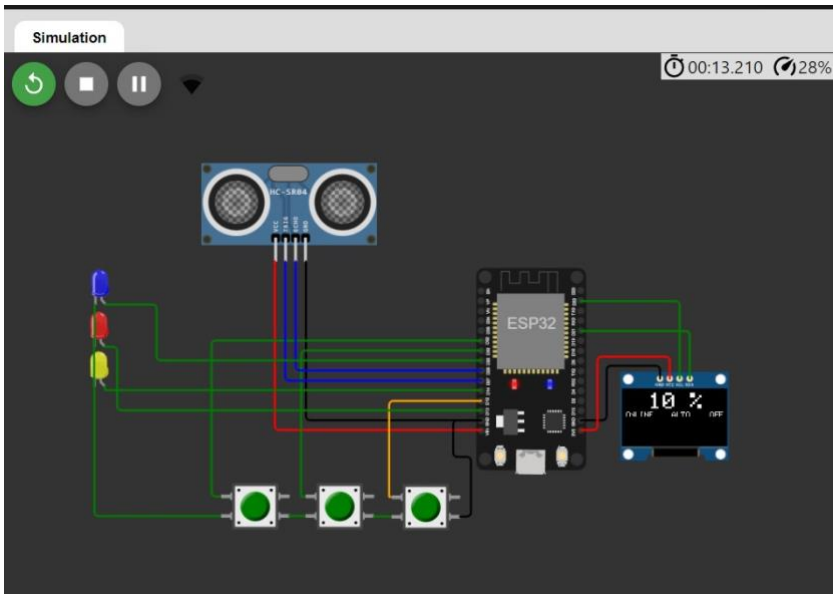
```
  Blynk.config(auth);
  delay(1000);
  Blynk.virtualWrite(VPIN_BUTTON_3, modeFlag);
  Blynk.virtualWrite(VPIN_BUTTON_4, toggleRelay);
  Blynk.virtualWrite(VPIN_BUTTON_5, toggleBuzzer);
  delay(500);
}
 void loop() {
  Blynk.run();
  timer.run();
  button1.check();
  button3.check();
if(!modeFlag){
    button2.check();
  }
}
void button1Handler(AceButton* button, uint8_t eventType, uint8_t buttonState) {
  Serial.println("EVENT1");
  switch (eventType) {
    case AceButton::kEventReleased:
      if(modeFlag && toggleRelay){
        digitalWrite(RelayPin, LOW);
        toggleRelay = false;
        controlBuzzer(500);
      }
      modeFlag = !modeFlag;
      currMode = modeFlag ? "AUTO" : "MANUAL";
      Blynk.virtualWrite(VPIN_BUTTON_3, modeFlag);
      controlBuzzer(200);
      break;
  }
}
```

```
void button2Handler(AceButton* button, uint8_t eventType, uint8_t buttonState) {
  Serial.println("EVENT2");
  switch (eventType) {
    case AceButton::kEventReleased:
      if(toggleRelay){
        digitalWrite(RelayPin, LOW);
        toggleRelay = false;
      }
      else{
        digitalWrite(RelayPin, HIGH);
        toggleRelay = true;
      }
      Blynk.virtualWrite(VPIN_BUTTON_4, toggleRelay);
      controlBuzzer(500);
      delay(1000);
      break;
  }
}
void button3Handler(AceButton* button, uint8_t eventType, uint8_t buttonState) {
  Serial.println("EVENT3");
  switch (eventType) {
    case AceButton::kEventReleased:
      digitalWrite(BuzzerPin, LOW);
      toggleBuzzer = LOW;
      Blynk.virtualWrite(VPIN_BUTTON_5, toggleBuzzer);
     break;
  }
}
```

CIRCUIT DIAGRAM:



OUTPUT:



WEB DEVELOPMENT CODE

SOURCE CODE:

HTML:

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Water Monitor</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="container">
        <h1>Water Monitor</h1>
        <div class="status">
            <h2>Water Level: <span id="waterLevel">0%</span></h2>
            <h2>Status: <span id="status">Offline</span></h2>
        </div>
        <div class="controls">
            <button id="toggleMode">Toggle Mode</button>
            <button id="toggleRelay">Toggle Relay</button>
            <button id="toggleBuzzer">Toggle Buzzer</button>
        </div>
    </div>
    <script src="script.js"></script>
</body>
</html>
```

CSS:

```
body {
    font-family: Arial, sans-serif;
}
.container {
    text-align: center;
    margin-top: 50px;
}
.status {
```

```
    margin-bottom: 20px;
}
.controls {
    display: flex;
    justify-content: center;
}
button {
    margin: 10px;
    padding: 10px 20px;
font-size: 16px;
}
```

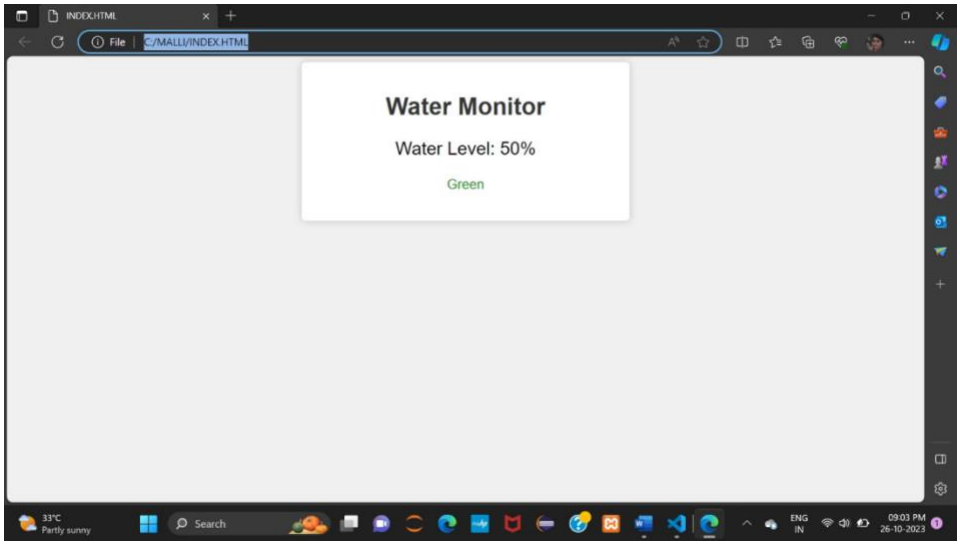
JAVASCRIPT:

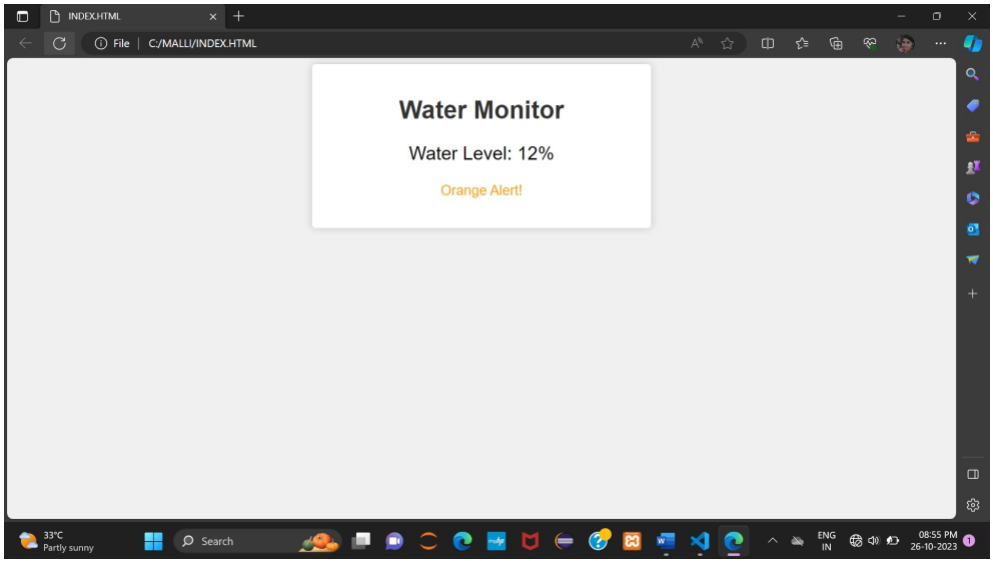
```
document.addEventListener("DOMContentLoaded", function () {
    var waterLevelSpan = document.getElementById("waterLevel");
    var statusSpan = document.getElementById("status");
    var toggleModeBtn = document.getElementById("toggleMode");
    var toggleRelayBtn = document.getElementById("toggleRelay");
    var toggleBuzzerBtn = document.getElementById("toggleBuzzer");
    var socket = new WebSocket("ws://localhost:3000");
       socket.onopen = function () {
        statusSpan.textContent = "Online";
        statusSpan.style.color = "green";
    };
       socket.onmessage = function (event) {
        var data = JSON.parse(event.data);
        waterLevelSpan.textContent = data.waterLevel + "%";
    };
        socket.onclose = function () {
        statusSpan.textContent = "Offline";
        statusSpan.style.color = "red";
    };
```

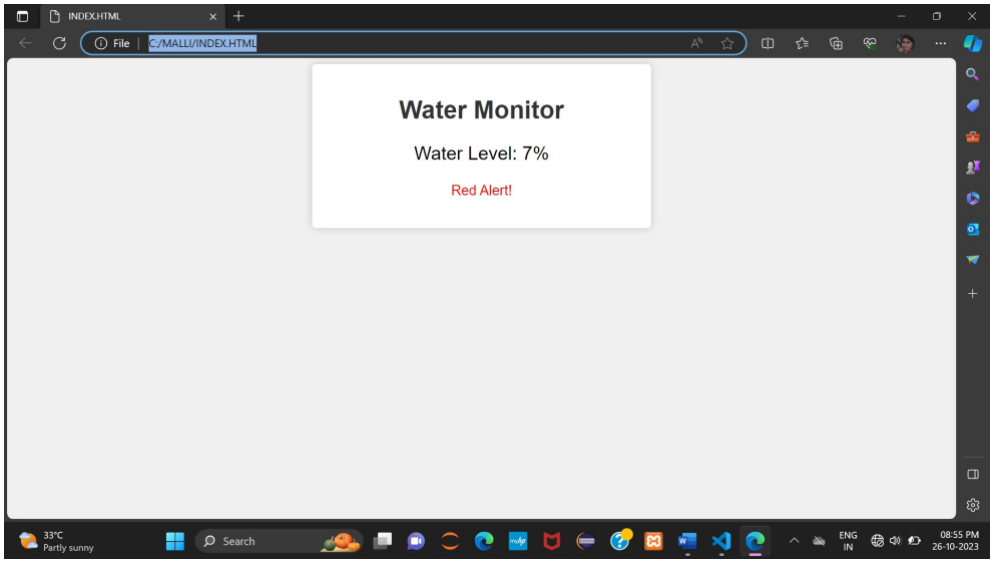
```
    toggleModeBtn.addEventListener("click", function () {
      // Send a message to the server to toggle the mode
      socket.send("toggleMode");
  });
      toggleRelayBtn.addEventListener("click", function () {
      // Send a message to the server to toggle the relay
      socket.send("toggleRelay");
  });

  toggleBuzzerBtn.addEventListener("click", function () {
      // Send a message to the server to toggle the buzzer
      socket.send("toggleBuzzer");
  });
});
```

OUTPUT:







PROJECT EXPLANATION IN DETAIL

Components:

Ultrasonic Sensor:

Ultrasonic water-level sensors operate on the same basic principles as general-purpose ultrasonic sensors. They emit ultrasonic waves and measure the time it takes for the waves to bounce back from the liquid surface.

ESP 32:

The ESP32 is a key component in a water monitoring system as it serves as the central processing unit and communication hub.

OLED Display:

An OLED (Organic Light-Emitting Diode) display can be a useful component in a water monitoring system to provide a local visual interface for real-time data.

LED:

Provides visual feedback to indicate the operational status of the water level.

Code Overview:

1.Setup Function:

Initializes pins for input and output, setting the initial state of components and initiating serial communication for debugging.

2.Loop Function:

- Continuously reads the water level from the ultrasonic sensor.
- If the water level surpasses a predetermined threshold (200 cm) , turns on the LED to indicate the operational.
- If the water level is higher than the range, red LED act as an indicator which glows. Whereas green LED glow if water level is lesser than the given range.

HTML Interface:

1.Status Display:

Provides information on the current status of the water level, whether it's actively running or stopped, allowing users to visualize its condition.

2.Distance Measurement:

Displays the measured water level detected by the ultrasonic sensor.

3.Control Buttons:

Offers users the ability to start or stop the water flow via the web interface by clicking on the corresponding buttons.

CONCLUSION:

Overall, a smart water management project using IoT technology is a powerful tool for addressing water scarcity, quality control, and sustainable resource management, benefiting both communities and the environment. The Smart Water Management System IoT project offers a promising solution to address water resource management and conservation. By leveraging Internet of Things (IoT) technology, this system provides real-time data collection and analysis, enabling more efficient water usage and early detection of issues. It has the potential to reduce water wastage, improve resource allocation, and contribute to environmental sustainability. However, the successful implementation of this project depends

on factors such as scalability, data security, and user accessibility. With proper planning and execution, it can play a vital role in ensuring a more sustainable and responsible use of our precious water resources.