**PROJECT TITTLE : SMART WATER MANAGEMENT**

**(PHASE -3 SUBMISSION)**

**1. Sensors and Data Acquisition:**

* **Water Quality Sensors:** pH sensors, turbidity sensors, dissolved oxygen sensors, and chemical sensors.
* **Flow Sensors:** Ultrasonic or electromagnetic flow sensors to measure the flow rate.
* **Level Indicators:** Ultrasonic or pressure-based sensors to measure water levels.
* **Microcontroller:** Arduino or Raspberry Pi for data processing and sensor interfacing.

**Water Quality Sensors:**

**Turbidity sensors:**

A turbidity sensor is an analytical sensor that measures turbidity. They are highly useful and effective instruments to identify the clarity and particle content in a solution, like water. Turbidity sensors are used to reduce waste, improve yields, and analyze water quality in a wide range of industries.

**Temperature sensors:**

Temperature sensors help to monitor water mixing, which is important in reservoirs, lakes, and other water systems. Temperature changes indicate changes in the distribution of water, which can be a sign of pollution, contamination, or other issues.

**Flow sensors:**

**Ultrasonic distance sensor:**

As they utilize high-frequency (ultrasonic) soundwaves to calculate the distance to a remote object without physically touching it, they can be used to create systems that reliably determine wave height and water levels at much lower installation and maintenance costs.



**2. Communication:**

* **Wireless Communication:** GSM, Wi-Fi, or LoRa modules for transmitting data to the central server.
* **Protocols:** MQTT or HTTP for secure data transmission.

**Wifi module:**

Through the Wi-Fi system, the sensor output data is sent to the concerned authority for further steps to supervise the water leakage. These sensor values are continuously uploaded into the cloud using wifi module.

**3. Centralized Server:**

* **Database:** MySQL or MongoDB to store real-time and historical data.
* **Backend:** Node.js, Django, or Flask for server-side scripting.
* **Data Analytics:** Python libraries like Pandas and Matplotlib for data analysis and visualization.
* **Web/Mobile Application:** HTML, CSS, JavaScript for the user interface. Charting libraries like Chart.js for graphical representation of data.

**4. User Interface:**

* **Web Application:** Allows authorities to monitor data, set thresholds, and receive alerts.
* **Mobile Application:** Provides real-time data access and alerts for consumers.

**5. Power Supply:**

* **Solar Panels:** To provide sustainable power to remote monitoring stations.
* **Battery Backup:** Lithium-ion batteries for continuous operation during power outages.

**6. Security:**

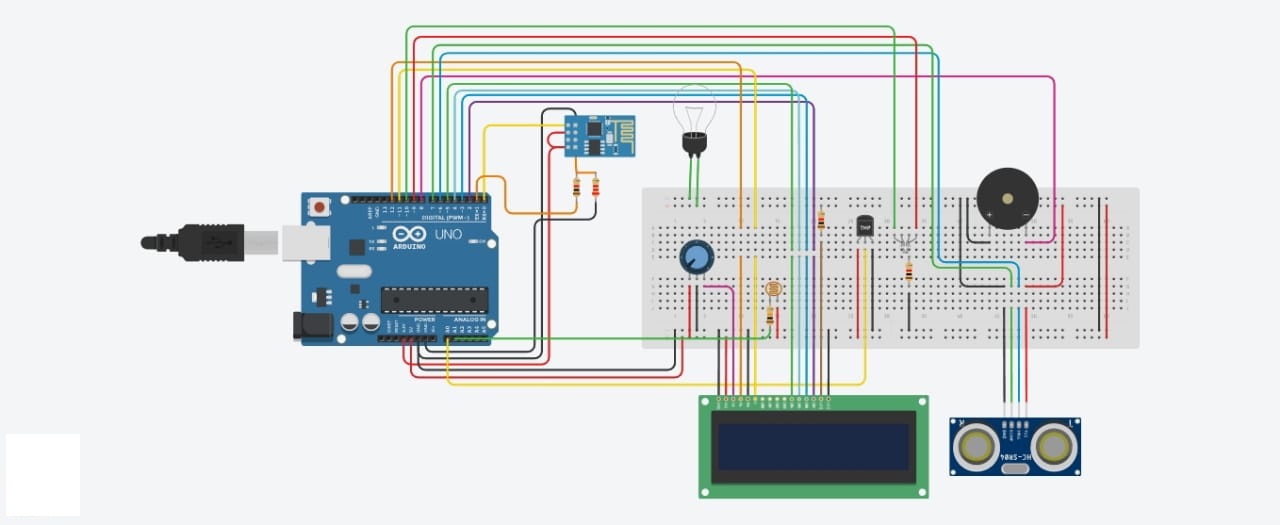
* **Data Encryption:** Use SSL/TLS protocols for secure data transmission.
* **Authentication:** Implement strong authentication mechanisms to prevent unauthorized access.

**7. Maintenance:**

* **Remote Diagnostics:** Include features for remote diagnosis and troubleshooting.
* **Regular Updates:** Ensure software and firmware updates for system efficiency and security.

By implementing this Smart Water Monitoring System, communities can make informed decisions about water usage, reduce waste, and ensure a sustainable and safe water supply for everyone.

**DESIGN**



**CODE**

// Allows communication with alphanumerical liquid crystal displays (LCDs).

#include <LiquidCrystal.h>

//Initialize the library with the numbers of the interface pins.

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

//This is the Arduino Pin that will read the sensor output.

int sensePin = A0;

//The variable we will use to store the sensor input.

int sensorInput;

//The variable we will use to store temperature in degrees.

double temp;

// Variable of the red led associated with the 9th pin

int redLed = 9 ;

// same, but for the green led with the 10th pin

int greenLed = 10 ;

int trigPin = 6;

int echoPin = 7;

long duration;

int distanceCm, dist;

int percentfull;

float speed,dist1,dist2,t1,t2;

int buck\_length=100;

int h=30;

const int piezoPin=8;

const int ledPinR=13;

const int ledPinG=6;

String ssid = "Simulator Wifi"; // SSID to connect to

String password = ""; // Our virtual wifi has no password (so dont do your banking stuff on this network)

String host = "api.thingspeak.com";

const int httpPort = 80;

String uri1 = "/update?api\_key=OR3V9VGVTTSWQMTZ&field1=";

String uri2 = "/update?api\_key=OR3V9VGVTTSWQMTZ&field2=";

String uri3 = "/apps/thinghttp/send\_request?api\_key=TM9724EPH078F0QC";

String uri4 = "/apps/thinghttp/send\_request?api\_key=S5RB4AGS933WXNDE";

String host1 = "maker.ifttt.com";

int setupESP8266(void) {

// Start our ESP8266 Serial Communication

Serial.begin(115200); // Serial connection over USB to computer

Serial.println("AT"); // Serial connection on Tx / Rx port to ESP8266

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("OK")) return 1;

// Connect to 123D Circuits Simulator Wifi

Serial.println("AT+CWJAP=\"" + ssid + "\",\"" + password + "\"");

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("OK")) return 2;

// Open TCP connection to the host:

Serial.println("AT+CIPSTART=\"TCP\",\"" + host + "\"," + httpPort);

delay(50); // Wait a little for the ESP to respond

if (!Serial.find("OK")) return 3;

return 0;

}

void anydata(float level) {

int temp = map(analogRead(A0),20,358,-40,125);

// Construct our HTTP call

String httpPacket = "GET " + uri1 + String(temp) + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";

int length = httpPacket.length();

// Send our message length

Serial.print("AT+CIPSEND=");

Serial.println(length);

delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

// Send our http request

Serial.print(httpPacket);

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("SEND OK\r\n")) return;

// 2

// Construct our HTTP call

httpPacket = "GET " + uri2 + String(level) + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";

length = httpPacket.length();

// Send our message length

Serial.print("AT+CIPSEND=");

Serial.println(length);

delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

// Send our http request

Serial.print(httpPacket);

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("SEND OK\r\n")) return;

}

void anydata1(){

// 3

// Construct our HTTP call

String httpPacket = "GET " + uri3 + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";

int length = httpPacket.length();

// Send our message length

Serial.print("AT+CIPSEND=");

Serial.println(length);

delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

// Send our http request

Serial.print(httpPacket);

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("SEND OK\r\n")) return;

}

void anydata2(){

// 3

// Construct our HTTP call

String httpPacket = "GET " + uri4 + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";

int length = httpPacket.length();

// Send our message length

Serial.print("AT+CIPSEND=");

Serial.println(length);

delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

// Send our http request

Serial.print(httpPacket);

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("SEND OK\r\n")) return;

}

void setup()

{

// pin of the red LED

pinMode(9, OUTPUT);

// pin of the green LED

pinMode(10, OUTPUT) ;

//Initialize the LCD's number of columns and rows.

lcd.begin(16, 2);

//Start the Serial Port at 9600 baud (default).

Serial.begin(9600);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

setupESP8266();

}

void loop()

{

//Set the cursor to column 0, line 0

lcd.setCursor(0, 0);

// LDR (Photoresistor)

int value = analogRead(A1); // Converting to Digital value

int aiv = map(value, 0, 1023, 0, 5000); // Converting back to Analog input voltage

float x = aiv / 1000; // Converting mV to V

int NTU = -111.25\*x + 506.67;

Serial.print(NTU);

Serial.println(" NTU");

if (NTU<=5) {

Serial.println("Water is Clear!");

}

else if (NTU>=6 && NTU<=99) {

Serial.println("Water is Turbid!");

}

else if (NTU>=100 && NTU<=500) {

Serial.println("Water is Dark!");

}

delay(250);

//Read the analog sensor and store it.

sensorInput = analogRead(A0);

float volt = sensorInput \* 5;

//Multiply by 5V to get voltage.

volt /= 1024;

Serial.print(volt);

Serial.println(" volts ");

//Subtract the offset.

temp = volt - 0.5;

//Convert to degrees.

temp = temp \* 100;

// recovering the differents result of

// the circuit in the serial monitor

delay(500);

Serial.print("Temperature ") ;

Serial.print(temp);

Serial.println(" Celsius");

// printing on the lcd screen the word "temperature"

lcd.print("Temperature: ");

//Set the cursor to column 0, line 1

lcd.setCursor(0, 1);

// Printing the temperature on the lcd screen

lcd.print(temp);

// printing the name of the scale used for temperature

lcd.print(" Celsius");

// if the temperature (var associated: temp) is superior

// to 40, then we enter the if loop

if (temp >= 40 || temp <= 23)

{

// turning on the red LED, stated as 'high'

digitalWrite(redLed, HIGH);

// turning off the green led, stated as 'low'

digitalWrite(greenLed, LOW);

anydata2();

//INPUT - FREQUENCY - TIME THAT LASTS

tone(piezoPin, 500, 500);

delay(200);

}

// otherwise, if the temperature is below 40

else if (temp < 40)

{

// turning on the green led

digitalWrite(greenLed, HIGH);

// and turning off the red led

digitalWrite(redLed, LOW);

noTone(8);

}

delay(5000);

lcd.clear();

dist=findDist();

percentfull=(h+buck\_length-dist)\*100/buck\_length;

//distanceInch = duration\*0.0133/2;

lcd.setCursor(0,0); // Sets the location at which subsequent text written to the LCD will be displayed

lcd.print("Level: "); // Prints string "Distance" on the LCD

lcd.print(h+buck\_length-dist); // Prints the distance value from the sensor

lcd.print(" cm");

delay(10);

lcd.setCursor(0,1);

lcd.print("%age full: ");

lcd.print(percentfull);

lcd.print("%");

delay(1000);

if(percentfull>=90)

{

digitalWrite(redLed,HIGH);

digitalWrite(greenLed,LOW);

tone(piezoPin,500,500);

delay(200);

//noTone(piezoPin);

anydata1();

}

else

{

digitalWrite(redLed,LOW);

digitalWrite(greenLed,HIGH);

delay(500);

digitalWrite(greenLed,LOW);

delay(200);

}

anydata(percentfull);

delay(5000);

lcd.clear();

}

int findDist(){

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

Serial.print("DURATION :");

Serial.print(duration);

distanceCm= duration\*0.034/4;

Serial.println("\nDISTANCE :");

Serial.print(distanceCm);

return distanceCm;

}