Project Title: Smart Water Quality Monitoring System in IoT

Development Overview:

Creating a smart water quality monitoring system involves various components, including sensors, data processing, and visualization. Below is a brief overview of the project, along with sample sensor code for commonly used components in such systems.

Project Overview:

The smart water quality monitoring system will utilize various sensors to measure parameters like water level, pH level, turbidity, temperature, and potentially other water quality indicators. The system will collect data from these sensors, process it, and display the results in a user-friendly format.

Hardware Components:

Microcontroller:

You can use popular microcontrollers like Arduino or Raspberry Pi to build your system.

Sensors:

a. Water Level Sensor:

Utilize sensors like ultrasonic or pressure transducers to measure water level.

b. pH Sensor:

Choose a pH sensor like the Atlas Scientific pH sensor for measuring the pH level of water.

c. Turbidity Sensor:

Use a turbidity sensor like the Turbidity Sensor for measuring water clarity.

d. Temperature Sensor:

The DS18B20 or similar temperature sensors can be used for water temperature measurement.

(You may also incorporate additional sensors based on specific water quality parameters of interest.)

Display:

A display unit, such as an LCD screen or an OLED display, can be used to show real-time water quality data.

Connectivity:

You can add Wi-Fi or GSM modules to enable data transmission to a remote server or for remote monitoring.

Program:

```
#include <EEPROM.h>


#include <LiquidCrystal.h>

LiquidCrystal lcd(2,3,4,5,6,7);


long duration, inches;

int set_val,percentage;

bool state,pump;
```

```
void setup() {

  lcd.begin(16, 2);
  lcd.print("WATER LEVEL:");
  lcd.setCursor(0, 1);
  lcd.print("PUMP:OFF MANUAL");

  pinMode(8, OUTPUT);
  pinMode(9, INPUT);
  pinMode(10, INPUT_PULLUP);
  pinMode(11, INPUT_PULLUP);
  pinMode(12, OUTPUT);

   set_val=EEPROM.read(0);
   if(set_val>150)set_val=150;
}
void loop() {

   digitalWrite(3, LOW);
   delayMicroseconds(2);
   digitalWrite(8, HIGH);
   delayMicroseconds(10);
   digitalWrite(8, LOW);
   duration = pulseIn(9, HIGH);
   inches = microsecondsToInches(duration);
```

```
percentage=(set_val-inches)*100/set_val;

lcd.setCursor(12, 0);
if(percentage<0)percentage=0;
lcd.print(percentage);
lcd.print("%   ");

if(percentage<30&digitalRead(11))pump=1;
if(percentage>99)pump=0;
digitalWrite(12,!pump);

lcd.setCursor(5, 1);
if(pump==1)lcd.print("ON ");
else if(pump==0) lcd.print("OFF");

 lcd.setCursor(9, 1);
 if(!digitalRead(11))lcd.print("MANUAL");
 else lcd.print("AUTO   ");

 if(!digitalRead(10)&!state&digitalRead(11)){
   state=1;
   set_val=inches;
   EEPROM.write(0, set_val);
   }

  if(!digitalRead(10)&!state&!digitalRead(11)){
        state=1;
        pump=!pump;

      }

    if(digitalRead(10))state=0;



    delay(500);
}
long microsecondsToInches(long microseconds) {
  return microseconds / 74 / 2;
}
```