Coding and solution

TEAM ID	PNT2022TMID52160
PROJECT NAME	REAL TIME RIVER WATER QUALITY MONITORING AND CONTROL SYSTEM

CODING:

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include <SoftwareSerial.h>
#include <NewPing.h>
#define SensorPin A2          //pH meter Analog output to Arduino Analog Input 0
#define Offset 0.00           //deviation compensate
unsigned long int avgValue;     //Store the average value of the sensor feedback

#define TRIGGER_PIN  23  // Arduino pin tied to trigger pin on ping sensor.
#define ECHO_PIN     22  // Arduino pin tied to echo pin on ping sensor.
#define MAX_DISTANCE 200 // Maximum distance we want to ping for (in centimeters). Maximum sensor distance is rated at 400-500cm.

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // NewPing setup of pins and maximum distance.

unsigned int pingSpeed = 50; // How frequently are we going to send out a ping (in milliseconds). 50ms would be 20 times a second.
unsigned long pingTimer;     // Holds the next ping time.

// Data wire is plugged into pin 2 on the Arduino
#define ONE_WIRE_BUS 6

SoftwareSerial mySerial(7, 8);
```

```
// Setup a oneWire instance to communicate with any OneWire devices (not
just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

//const int pingPin =22;
int sensorPin = A0;
int blueled = 13;
int redled = 24;
int greenled = 25;
int tempblueled = 32;
int tempredled = 33;
int tempgreenled = 34;
int levblueled = 35;
int levredled = 36;
int levgreenled = 37;
int turbblueled = 38;
int turbredled = 39;
int turbgreenled = 40;
int buzzer = 31;
float phValue;
float temperatureC;
long duration, cm;

void setup(void)
{
  // start serial port
  Serial.begin(9600);
  pingTimer = millis(); // Start now.
  // Start up the library
  sensors.begin(); // IC Default 9 bit. If you have troubles consider upping it 12.
Ups the delay giving the IC more time to process the temperature
measurement
  lcd.begin(16, 2); //initilise lcd with num of coloums 16 ,by row 2.
```

```
  lcd.clear(); //clears lcd just incase there is anytin been displayed
  pinMode(blueled, OUTPUT);
  pinMode(redled, OUTPUT);
  pinMode(greenled, OUTPUT);
  pinMode(tempblueled, OUTPUT);
  pinMode(tempredled, OUTPUT);
  pinMode(tempgreenled, OUTPUT);
  pinMode(levblueled, OUTPUT);
  pinMode(levredled, OUTPUT);
  pinMode(levgreenled, OUTPUT);
  pinMode(turbblueled, OUTPUT);
  pinMode(turbredled, OUTPUT);
  pinMode(turbgreenled, OUTPUT);
  pinMode(buzzer, OUTPUT);
    digitalWrite(buzzer, LOW);
  //initialization();
}

void loop() {
  sensors.requestTemperatures(); // Send the command to get temperature
  Serial.println(sensors.getTempCByIndex(0));
    int reading = analogRead(sensorPin);

  // converting that reading to voltage,
  float voltage = reading * 5.0;
  voltage /= 1024.0;

  // now print out the temperature
  float temperatureC = (voltage - 0.5) * 100 ;
  PH();
    digitalWrite(blueled, LOW);
  digitalWrite(redled, LOW);
  digitalWrite(greenled, LOW);
  temperature ();
  digitalWrite(tempblueled, LOW);
  digitalWrite(tempredled, LOW);
  digitalWrite(tempgreenled, LOW);
  Water_level();
     digitalWrite(levblueled, LOW);
  digitalWrite(levredled, LOW);
```

```
  digitalWrite(levgreenled, LOW);
  turbidity();
   digitalWrite(turbblueled, LOW);
  digitalWrite(turbredled, LOW);
  digitalWrite(turbgreenled, LOW);
  send_sms();
  delay(4000);
  send_sms1();
    digitalWrite(greenled, LOW);
  digitalWrite(tempgreenled, LOW);
  digitalWrite(levgreenled, LOW);
  digitalWrite(turbgreenled, LOW);
  delay(8000);
}



////////////////////////////////////////////////////////// FUNCTIONS
//////////////////////////////////////////////////////////////////

void PH(){
  Serial.println(" ");
  lcd.clear( );
  digitalWrite(blueled, LOW);
  digitalWrite(greenled, LOW);
  digitalWrite(redled, LOW);
  digitalWrite(buzzer, LOW);
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("TAKING READINGS");
  lcd.setCursor(1,1);
  lcd.print("FROM PH SENSOR");
  Serial.println("Taking Readings from PH Sensor");
  PHblink();
  int buf[10];              //buffer for read analog
  for(int i=0;i<10;i++)       //Get 10 sample value from the sensor for smooth the
value
  {
    buf[i]=analogRead(SensorPin);
    delay(10);
  }
```

```
for(int i=0;i<9;i++)       //sort the analog from small to large
{
  for(int j=i+1;j<10;j++)
  {
    if(buf[i]>buf[j])
    {
      int temp=buf[i];
      buf[i]=buf[j];
      buf[j]=temp;
    }
  }
}
avgValue=0;
for(int i=2;i<8;i++)                    //take the average value of 6 center sample
  avgValue+=buf[i];
float phValue=(float)avgValue*3.8/1030/6; //convert the analog into millivolt
phValue=3.3*phValue+Offset;                    //convert the millivolt into pH value
Serial.print("pH:");
Serial.print(phValue,2);
Serial.println(" ");

if(phValue >= 7.30){
  lcd.clear();
  digitalWrite(blueled, LOW);
  digitalWrite(greenled, LOW);
  digitalWrite(redled, HIGH);
  digitalWrite(buzzer, HIGH);
  lcd.setCursor(1,0);//set cursor (colum by row) indexing from 0
  lcd.print("PH VALUE:");
  lcd.setCursor(10,0);
  lcd.print(phValue);
  lcd.setCursor(0,1);
  Serial.print("PH VALUE: ");
  Serial.println(phValue);
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("ALKALINITY HIGH");
  Serial.println("Water Alkalinity high");
  delay(3000);
 }
```

```
  if(phValue >= 6.90 && phValue <= 7.19){
    digitalWrite(blueled, HIGH);
    digitalWrite(greenled,LOW);
    digitalWrite(redled, LOW);
    digitalWrite(buzzer, LOW);
    lcd.clear();
    lcd.setCursor(1,0);//set cursor (colum by row) indexing from 0
    lcd.print("PH VALUE:");
    lcd.setCursor(10,0);
    lcd.print(phValue);
    lcd.setCursor(0,1);
    Serial.print("PH VALUE: ");
    Serial.println(phValue);
    lcd.setCursor(1,1);//set cursor (colum by row) indexing from 0
    lcd.print("WATER IS SAFE");
    Serial.println("Water Is  neutral (safe)");
  }

  if(phValue < 6.89){
    lcd.clear();
    digitalWrite(blueled, LOW);
    digitalWrite(greenled, LOW);
    digitalWrite(redled, HIGH);
    digitalWrite(buzzer, HIGH);
    lcd.setCursor(1,0);//set cursor (colum by row) indexing from 0
    lcd.print("PH VALUE:");
    lcd.setCursor(10,0);
    lcd.print(phValue);
    lcd.setCursor(0,1);
    Serial.print("PH VALUE: ");
    Serial.println(phValue);
    lcd.setCursor(2,1);//set cursor (colum by row) indexing from 0
    lcd.print("ACIDITY HIGH");
    Serial.println("Water Acidity High");
    delay(3000);
   }

  delay(8000);
}
```

```
void temperature (){
  Serial.println(" ");
  lcd.clear( );
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("TAKING READINGS");
  lcd.setCursor(0,1);
  lcd.print("FROM TEMP SENSOR");
  Serial.println("Taking Readings from Temperature Sensor");
  TEMPblink();
  temp_check_surr();
  delay(4000);
  temp_check_water();
}


void temp_check_surr(){
  digitalWrite(tempblueled, LOW);
  digitalWrite(tempgreenled, LOW);
  digitalWrite(tempredled, LOW);
  digitalWrite(buzzer, LOW);
  int reading = analogRead(sensorPin);

  // converting that reading to voltage,
  float voltage = reading * 5.0;
  voltage /= 1024.0;

  // now print out the temperature
  float temperatureC = (voltage - 0.5) * 100 ;  //converting from 10 mv per
degree wit 500 mV offset
  //to degrees ((voltage - 500mV) times 100)
  lcd.clear();
  Serial.print("Surrounding Temperature: ");
  Serial.println(temperatureC);

  if(temperatureC > 50){
    digitalWrite(tempblueled, LOW);
    digitalWrite(tempgreenled, LOW);
    digitalWrite(tempredled, HIGH);
    digitalWrite(buzzer, HIGH);
```

```
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("SUR TEMP:");
  lcd.setCursor(9,0);
  lcd.print(temperatureC);
  lcd.setCursor(14,0);
  lcd.print("*C");
  lcd.setCursor(0,1);
  Serial.print("Surrounding Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" degree C");
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("SURR TEMP HIGH");
  Serial.println("Surrounding Temperature high");
  delay(3000);
 }

if(temperatureC >= 10 && temperatureC <= 50){
  digitalWrite(tempblueled, HIGH);
  digitalWrite(tempgreenled,LOW);
  digitalWrite(tempredled, LOW);
  digitalWrite(buzzer, LOW);
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("SUR TEMP:");
  lcd.setCursor(9,0);
  lcd.print(temperatureC);
  lcd.setCursor(14,0);
  lcd.print("*C");
  lcd.setCursor(0,1);
  Serial.print("Surrounding Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" degree C");
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("SURR TEMP NORMAL");
  Serial.println("Surrounding Temperature normal");
}

if(temperatureC < 10){
  digitalWrite(tempblueled, LOW);
  digitalWrite(tempgreenled, LOW);
  digitalWrite(tempredled, HIGH);
```

```
    digitalWrite(buzzer, HIGH);
    lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
    lcd.print("SUR TEMP:");
    lcd.setCursor(9,0);
    lcd.print(temperatureC);
    lcd.setCursor(14,0);
    lcd.print("*C");
    lcd.setCursor(0,1);
    Serial.print("Surrounding Temperature: ");
    Serial.print(temperatureC);
    Serial.println(" degree C");
    lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
    lcd.print("SURR TEMP LOW");
    Serial.println("Surrounding Temperature low");
    delay(3000);
   }
  delay(8000);
}


void temp_check_water(){
  lcd.clear();
  digitalWrite(tempblueled, LOW);
  digitalWrite(tempgreenled, LOW);
  digitalWrite(tempredled, LOW);
  digitalWrite(buzzer, LOW);
  sensors.requestTemperatures(); // Send the command to get temperature
  Serial.print("Water Temperature: ");
  Serial.println(sensors.getTempCByIndex(0));

  if(sensors.getTempCByIndex(0) > 40){
    digitalWrite(tempblueled, LOW);
    digitalWrite(tempgreenled, LOW);
    digitalWrite(tempredled, HIGH);
    digitalWrite(buzzer, HIGH);
    lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
    lcd.print("WAT TEMP:");
    lcd.setCursor(9,0);
    lcd.print(sensors.getTempCByIndex(0));
    lcd.setCursor(14,0);
```

```
  lcd.print("*C");
  lcd.setCursor(0,1);
  Serial.print("Water Temperature: ");
  Serial.print(sensors.getTempCByIndex(0));
  Serial.println(" degree C");
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("WATER TEMP HIGH");
  Serial.println("Water Temperature high");
  delay(3000);
 }

 if(sensors.getTempCByIndex(0) >= 15 && sensors.getTempCByIndex(0) <=
40){
  digitalWrite(tempblueled, HIGH);
  digitalWrite(tempgreenled,LOW);
  digitalWrite(tempredled, LOW);
  digitalWrite(buzzer, LOW);
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("WAT TEMP:");
  lcd.setCursor(9,0);
  lcd.print(sensors.getTempCByIndex(0));
  lcd.setCursor(14,0);
  lcd.print("*C");
  lcd.setCursor(0,1);
  Serial.print("Water Temperature: ");
  Serial.print(sensors.getTempCByIndex(0));
  Serial.println(" degree C");
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("WATER TEMP NORMAL");
  Serial.println("Water Temperature normal");
 }

 if(sensors.getTempCByIndex(0) < 15){
  digitalWrite(tempblueled, LOW);
  digitalWrite(tempgreenled, LOW);
  digitalWrite(tempredled, HIGH);
  digitalWrite(buzzer, HIGH);
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("WAT TEMP:");
  lcd.setCursor(9,0);
```

```
    lcd.print(sensors.getTempCByIndex(0));
    lcd.setCursor(14,0);
    lcd.print("*C");
    lcd.setCursor(0,1);
    Serial.print("Water Temperature: ");
    Serial.print(sensors.getTempCByIndex(0));
    Serial.println(" degree C");
    lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
    lcd.print("WATER TEMP LOW");
    Serial.println("Water Temperature low");
    delay(3000);
  }
  delay(8000);
}


void Water_level() {
    Serial.println(" ");
  digitalWrite(levblueled, LOW);
  digitalWrite(levgreenled, LOW);
  digitalWrite(levredled, LOW);
  digitalWrite(buzzer, LOW);
  lcd.clear( );
  lcd.setCursor(2,0);//set cursor (colum by row) indexing from 0
  lcd.print("READINGS FROM");
  lcd.setCursor(0,1);
  lcd.print("WATER LEVEL SENS");
  Serial.println("Taking Readings from Water Level Sensor");
  LEVblink();
  level_check();
  delay(8000);
}


void level_check(){
  digitalWrite(levblueled, LOW);
  digitalWrite(levgreenled, LOW);
  digitalWrite(levredled, LOW);
  digitalWrite(buzzer, LOW);
```

```
   // Notice how there's no delays in this sketch to allow you to do other
processing in-line while doing distance pings.
  if (millis() >= pingTimer) {   // pingSpeed milliseconds since last ping, do
another ping.
    pingTimer += pingSpeed;      // Set the next ping time.
    sonar.ping_timer(echoCheck); // Send out the ping, calls "echoCheck"
function every 24uS where you can check the ping status.
//delay(1000);
}
}

void echoCheck() { // Timer2 interrupt calls this function every 24uS where you
can check the ping status.
  // Don't do anything here!
  if (sonar.check_timer()) { // This is how you check to see if the ping was
received.
   if(sonar.ping_result / US_ROUNDTRIP_CM > 7){
    digitalWrite(levblueled, LOW);
    digitalWrite(levgreenled, LOW);
    digitalWrite(levredled, HIGH);
    digitalWrite(buzzer, HIGH);
    lcd.clear( );
    lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
    lcd.print("WATER LEVEL:");
    lcd.setCursor(12,0);
    lcd.print(sonar.ping_result / US_ROUNDTRIP_CM);
    lcd.setCursor(14,0);
    lcd.print("cm");
    lcd.setCursor(0,1);
    Serial.print("Water Level: ");
    Serial.print(sonar.ping_result / US_ROUNDTRIP_CM);
    Serial.println("cm");
    lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
    lcd.print("WATER LEVEL LOW");
    Serial.println("Water Level low");
    delay(3000);

  }
```

```
  if(sonar.ping_result / US_ROUNDTRIP_CM >= 5 && sonar.ping_result /
US_ROUNDTRIP_CM <= 7){
     digitalWrite(levblueled, HIGH);
  digitalWrite(levgreenled, LOW);
  digitalWrite(levredled, LOW);
  digitalWrite(buzzer, LOW);
   lcd.clear( );
   lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
   lcd.print("WATER LEVEL:");
   lcd.setCursor(12,0);
   lcd.print(sonar.ping_result / US_ROUNDTRIP_CM);
   lcd.setCursor(14,0);
   lcd.print("cm");
   lcd.setCursor(0,1);
   Serial.print("Water Level: ");
   Serial.print(sonar.ping_result / US_ROUNDTRIP_CM);
   Serial.println("cm");
   lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
   lcd.print("WATER LEVEL NORMAL");
   Serial.println("Water Level normal");
  }

  if(sonar.ping_result / US_ROUNDTRIP_CM < 5){
   digitalWrite(levblueled, LOW);
   digitalWrite(levgreenled, LOW);
   digitalWrite(levredled, HIGH);
   digitalWrite(buzzer, HIGH);
   lcd.clear( );
   lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
   lcd.print("WATER LEVEL:");
   lcd.setCursor(12,0);
   lcd.print(sonar.ping_result / US_ROUNDTRIP_CM);
   lcd.setCursor(14,0);
   lcd.print("cm");
   lcd.setCursor(0,1);
   Serial.print("Water Level: ");
   Serial.print(sonar.ping_result / US_ROUNDTRIP_CM);
   Serial.println("cm");
   lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
   lcd.print("WATER LEVEL HIGH");
```

```
   Serial.println("Water Level high");
   delay(2000);
     }
  }
 }

/*long microsecondsToCentimeters(long microseconds)
{
  return microseconds / 29 / 2;
}
*/

void turbidity() {
  Serial.println(" ");
  digitalWrite(turbblueled, LOW);
  digitalWrite(turbgreenled, LOW);
  digitalWrite(turbredled, LOW);
  digitalWrite(buzzer, LOW);
  lcd.clear( );
  lcd.setCursor(1,0);//set cursor (colum by row) indexing from 0
  lcd.print("READINGS FROM");
  lcd.setCursor(0,1);
  lcd.print("TURBIDITY SENSOR");
  Serial.println("Taking Readings from turbidity Sensor");
  TURBblink();
  int turbidityValue = analogRead(A1);
  float turbidityV = turbidityValue/100;
  Serial.print("Turbidity level: ");
  Serial.println(turbidityV);

  if( turbidityV > 9){
    digitalWrite(turbblueled, LOW);
    digitalWrite(turbgreenled, LOW);
    digitalWrite(turbredled, HIGH);
    digitalWrite(buzzer, HIGH);
    lcd.clear( );
    lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
    lcd.print("TURBI LEV:");
    lcd.setCursor(11,0);
    lcd.print(turbidityV);
```

```
  lcd.setCursor(14,0);
  lcd.print("NTU");
  lcd.setCursor(0,1);
  Serial.print("Turbidity Level: ");
  Serial.print(turbidityV);
  Serial.println("NTU");
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("WATER VERY CLEAN");
  Serial.println("Water Very Clean ");
  delay(3000);
 }

 if( turbidityV >= 6 && turbidityValue/100 <= 9 ){
  digitalWrite(turbblueled, HIGH);
  digitalWrite(turbgreenled, LOW);
  digitalWrite(turbredled, LOW);
  digitalWrite(buzzer, LOW);
  lcd.clear( );
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("TURBI LEV:");
  lcd.setCursor(10,0);
  lcd.print(turbidityV);
  lcd.setCursor(13,0);
  lcd.print("NTU");
  lcd.setCursor(0,1);
  Serial.print("Turbidity Level: ");
  Serial.print(turbidityV);
  Serial.println("NTU");
  lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
  lcd.print("WATER NORM CLEAN");
  Serial.println("Water Clean ");
 }

 if( turbidityV < 6){
  digitalWrite(turbblueled, LOW);
  digitalWrite(turbgreenled, LOW);
  digitalWrite(turbredled, HIGH);
  digitalWrite(buzzer, HIGH);
  lcd.clear( );
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
```

```
    lcd.print("TURBI LEV:");
    lcd.setCursor(11,0);
    lcd.print(turbidityV);
    lcd.setCursor(14,0);
    lcd.print("NTU");
    lcd.setCursor(0,1);
    Serial.print("Turbidity Level: ");
    Serial.print(turbidityV);
    Serial.println("NTU");
    lcd.setCursor(0,1);//set cursor (colum by row) indexing from 0
    lcd.print("WATER VERY DIRTY");
    Serial.println("Water Very Dirty ");
    delay(3000);
   }
  delay (8000);
}


void initialization(){
  lcd.setCursor(0,0);//set cursor (colum by row) indexing from 0
  lcd.print("INITIALIZING ALL");
  lcd.setCursor(0,1);
  lcd.print("PARAMETERS");
  delay(3000);
  lcd.setCursor(0,1);
  lcd.print("PARAMETERS.");
  delay(3000);
  lcd.setCursor(0,1);
  lcd.print("PARAMETERS..");
  delay(3000);
  lcd.setCursor(0,1);
  lcd.print("PARAMETERS...");
  delay(3000);
  lcd.setCursor(0,1);
  lcd.print("PARAMETERS....");
  delay(3000);
  lcd.setCursor(0,1);
  lcd.print("PARAMETERS.....");
  delay(3000);
  lcd.setCursor(0,1);
```

```
  lcd.print("PARAMETERS.....");
  Serial.println("Initializing All Parameters.....");
  delay(8000);
  Serial.println("Initializing Done.");
  lcd.clear();
  lcd.setCursor(1,0);
  lcd.print("INITIALIZATION");
  lcd.setCursor(6,1);
  lcd.print("DONE  ");
  delay(2000);
}

////////////////////////////////////////////////////// SMS FUNCTION
//////////////////////////////////////////////////////////////////


void send_sms(){
  lcd.clear();
  lcd.setCursor(2,0);//set cursor (colum by row) indexing from 0
  lcd.print("ATTENTION!!! ");
  lcd.setCursor(2,1);
  lcd.print("SENDING SMS ");
  String temp;
  String lev;
  String phm;
  String turb;
  int turbidityValue = analogRead(A1);
  float turbidityV = turbidityValue/100;

  int buf[10];                //buffer for read analog
  for(int i=0;i<10;i++)       //Get 10 sample value from the sensor for smooth the
value
  {
    buf[i]=analogRead(SensorPin);
    delay(10);
  }
  for(int i=0;i<9;i++)        //sort the analog from small to large
  {
    for(int j=i+1;j<10;j++)
    {
```

```
    if(buf[i]>buf[j])
    {
      int temp=buf[i];
      buf[i]=buf[j];
      buf[j]=temp;
    }
   }
  }
  avgValue=0;
  for(int i=2;i<8;i++)                    //take the average value of 6 center sample
   avgValue+=buf[i];
  float phValue=(float)avgValue*3.8/1030/6; //convert the analog into millivolt
  phValue=3.3*phValue+Offset;

   if(sensors.getTempCByIndex(0) > 40){
   temp = String("HIGH");
   }
   if(sensors.getTempCByIndex(0) >= 10 && sensors.getTempCByIndex(0) <=
40){
   temp = String("NORMAL");
   }
   if(sensors.getTempCByIndex(0) < 10){
    temp = String("LOW");
   }

   if(sonar.ping_result / US_ROUNDTRIP_CM > 8){
   lev = String("LOW");
   }
   if(sonar.ping_result / US_ROUNDTRIP_CM >= 5 && sonar.ping_result /
US_ROUNDTRIP_CM <= 8){
   lev = String("NORMAL");
   }
   if(sonar.ping_result / US_ROUNDTRIP_CM < 5){
   lev = String("HIGH");
   }

   if(phValue >= 7.30){
   phm = String("ALKALINE");
   }
   if(phValue >= 6.90 && phValue <= 7.19){
```

```
   phm = String("NORMAL");
   }
   if(phValue < 6.89){
    phm = String("ACIDIC");
   }

   if(turbidityV >= 6 && turbidityValue/100 <= 9){
   turb = String("CLEAN");
   }
   if(turbidityV < 6){
   turb = String("DIRTY");
   }

  mySerial.begin(19200);  //Default serial port setting for the GPRS modem is
19200bps 8-N-1
  mySerial.print("\r");
  digitalWrite(buzzer, LOW);
  digitalWrite(blueled, LOW);
  digitalWrite(greenled, LOW);
  digitalWrite(redled, LOW);
  delay(1000);                  //wait for a second while the modem sends an "OK"
  mySerial.print("AT+CMGF=1\r");    //Because we want to send the SMS in text
mode
  delay(1000);
  mySerial.print("AT+CMGS=\"+233540518223\"\r");    //Start accepting the
text for the message
  delay(1000);
  mySerial.print(temp);
  mySerial.print(" \r");
  mySerial.print("WATER TEMPERATURE= \r");   //The text for the message
  mySerial.print(sensors.getTempCByIndex(0));
  mySerial.print("*C\r");
  mySerial.println("\r");
  mySerial.print(lev);
  mySerial.print(" \r");
  mySerial.print("WATER LEVEL= \r");   //The text for the message
  mySerial.print(sonar.ping_result / US_ROUNDTRIP_CM);
  mySerial.print("cm\r");
  mySerial.println("\r");
  mySerial.print(phm);
```

```
  mySerial.print(" \r");
  mySerial.print("WATER PH VALUE= \r");   //The text for the message
  mySerial.print(phValue);
  mySerial.println("\r");
  mySerial.print(turb);
  mySerial.print(" \r");
  mySerial.print("WATER TURBIDITY= \r");   //The text for the message
  mySerial.print(turbidityV);
  mySerial.print("NBT\r");

  digitalWrite(greenled, HIGH);
  digitalWrite(tempgreenled, HIGH);
  digitalWrite(levgreenled, HIGH);
  digitalWrite(turbgreenled, HIGH);
  delay(3000);
  /*lcd.clear();
  lcd.setCursor(5,0);//set cursor (colum by row) indexing from 0
  lcd.print("SMS SENT ");
  lcd.setCursor(2,1);
  lcd.print("SUCCESSFULLY ");*/
  mySerial.write(0x1A);  //Equivalent to sending Ctrl+Z
}

void send_sms1(){
  lcd.clear();
  lcd.setCursor(2,0);//set cursor (colum by row) indexing from 0
  lcd.print("ATTENTION!!! ");
  lcd.setCursor(2,1);
  lcd.print("SENDING SMS ");
  String temp;
  String lev;
  String phm;
  String turb;
  int turbidityValue = analogRead(A1);
  float turbidityV = turbidityValue/100;

  int buf[10];              //buffer for read analog
  for(int i=0;i<10;i++)      //Get 10 sample value from the sensor for smooth the
value
  {
```

```
   buf[i]=analogRead(SensorPin);
   delay(10);
  }
  for(int i=0;i<9;i++)       //sort the analog from small to large
  {
   for(int j=i+1;j<10;j++)
   {
    if(buf[i]>buf[j])
    {
     int temp=buf[i];
     buf[i]=buf[j];
     buf[j]=temp;
    }
   }
  }
  avgValue=0;
  for(int i=2;i<8;i++)                    //take the average value of 6 center sample
   avgValue+=buf[i];
  float phValue=(float)avgValue*3.8/1030/6; //convert the analog into millivolt
  phValue=3.3*phValue+Offset;

   if(sensors.getTempCByIndex(0) > 40){
   temp = String("HIGH");
   }
   if(sensors.getTempCByIndex(0) >= 10 && sensors.getTempCByIndex(0) <=
40){
   temp = String("NORMAL");
   }
   if(sensors.getTempCByIndex(0) < 10){
    temp = String("LOW");
   }

   if(sonar.ping_result / US_ROUNDTRIP_CM > 8){
   lev = String("LOW");
   }
   if(sonar.ping_result / US_ROUNDTRIP_CM >= 5 && sonar.ping_result /
US_ROUNDTRIP_CM <= 8){
   lev = String("NORMAL");
   }
   if(sonar.ping_result / US_ROUNDTRIP_CM < 5){
```

```
   lev = String("HIGH");
   }

   if(phValue >= 7.30){
   phm = String("ALKALINE");
   }
   if(phValue >= 6.90 && phValue <= 7.19){
   phm = String("NORMAL");
   }
   if(phValue < 6.89){
    phm = String("ACIDIC");
   }

   if(turbidityV >= 6 && turbidityValue/100 <= 9){
   turb = String("CLEAN");
   }
   if(turbidityV < 6){
   turb = String("DIRTY");
   }

  mySerial.begin(19200);  //Default serial port setting for the GPRS modem is
19200bps 8-N-1
  mySerial.print("\r");
  digitalWrite(buzzer, LOW);
  digitalWrite(blueled, LOW);
  digitalWrite(greenled, LOW);
  digitalWrite(redled, LOW);
  delay(1000);                  //wait for a second while the modem sends an "OK"
  mySerial.print("AT+CMGF=1\r");    //Because we want to send the SMS in text
mode
  delay(1000);
  mySerial.print("AT+CMGS=\"+233265188849\"\r");    //Start accepting the
text for the message
  delay(1000);
  mySerial.print(temp);
  mySerial.print(" \r");
  mySerial.print("WATER TEMPERATURE= \r");   //The text for the message
  mySerial.print(sensors.getTempCByIndex(0));
  mySerial.print("*C\r");
  mySerial.println("\r");
```

```
  mySerial.print(lev);
  mySerial.print(" \r");
  mySerial.print("WATER LEVEL= \r");   //The text for the message
  mySerial.print(sonar.ping_result / US_ROUNDTRIP_CM);
  mySerial.print("cm\r");
  mySerial.println("\r");
  mySerial.print(phm);
  mySerial.print(" \r");
  mySerial.print("WATER PH VALUE= \r");   //The text for the message
  mySerial.print(phValue);
  mySerial.println("\r");
  mySerial.print(turb);
  mySerial.print(" \r");
  mySerial.print("WATER TURBIDITY= \r");   //The text for the message
  mySerial.print(turbidityV);
  mySerial.print("NBT\r");

  digitalWrite(greenled, HIGH);
  digitalWrite(tempgreenled, HIGH);
  digitalWrite(levgreenled, HIGH);
  digitalWrite(turbgreenled, HIGH);
  delay(3000);
  lcd.clear();
  lcd.setCursor(5,0);//set cursor (colum by row) indexing from 0
  lcd.print("SMS SENT ");
  lcd.setCursor(2,1);
  lcd.print("SUCCESSFULLY ");
  mySerial.write(0x1A);  //Equivalent to sending Ctrl+Z
}

void PHblink() {
  digitalWrite(blueled, HIGH);
  delay(1000);
  digitalWrite(blueled, LOW);
  delay(1000);
  digitalWrite(blueled, HIGH);
  delay(1000);
  digitalWrite(blueled, LOW);
  delay(1000);
  digitalWrite(blueled, HIGH);
```

```
  delay(1000);
  digitalWrite(blueled, LOW);
  delay(1000);
   digitalWrite(blueled, HIGH);
  delay(1000);
  digitalWrite(blueled, LOW);
  delay(1000);
}

void TEMPblink() {
  digitalWrite(tempblueled, HIGH);
  delay(1000);
  digitalWrite(tempblueled, LOW);
  delay(1000);
  digitalWrite(tempblueled, HIGH);
  delay(1000);
  digitalWrite(tempblueled, LOW);
  delay(1000);
  digitalWrite(tempblueled, HIGH);
  delay(1000);
  digitalWrite(tempblueled, LOW);
  delay(1000);
   digitalWrite(tempblueled, HIGH);
  delay(1000);
  digitalWrite(tempblueled, LOW);
  delay(1000);
}

void LEVblink() {
  digitalWrite(levblueled, HIGH);
  delay(1000);
  digitalWrite(levblueled, LOW);
  delay(1000);
  digitalWrite(levblueled, HIGH);
  delay(1000);
  digitalWrite(levblueled, LOW);
  delay(1000);
  digitalWrite(levblueled, HIGH);
  delay(1000);
  digitalWrite(levblueled, LOW);
```

```
  delay(1000);
   digitalWrite(levblueled, HIGH);
  delay(1000);
  digitalWrite(levblueled, LOW);
  delay(1000);
}

void TURBblink() {
  digitalWrite(turbblueled, HIGH);
  delay(1000);
  digitalWrite(turbblueled, LOW);
  delay(1000);
  digitalWrite(turbblueled, HIGH);
  delay(1000);
  digitalWrite(turbblueled, LOW);
  delay(1000);
  digitalWrite(turbblueled, HIGH);
  delay(1000);
  digitalWrite(turbblueled, LOW);
  delay(1000);
   digitalWrite(turbblueled, HIGH);
  delay(1000);
  digitalWrite(turbblueled, LOW);
  delay(1000);
}
```