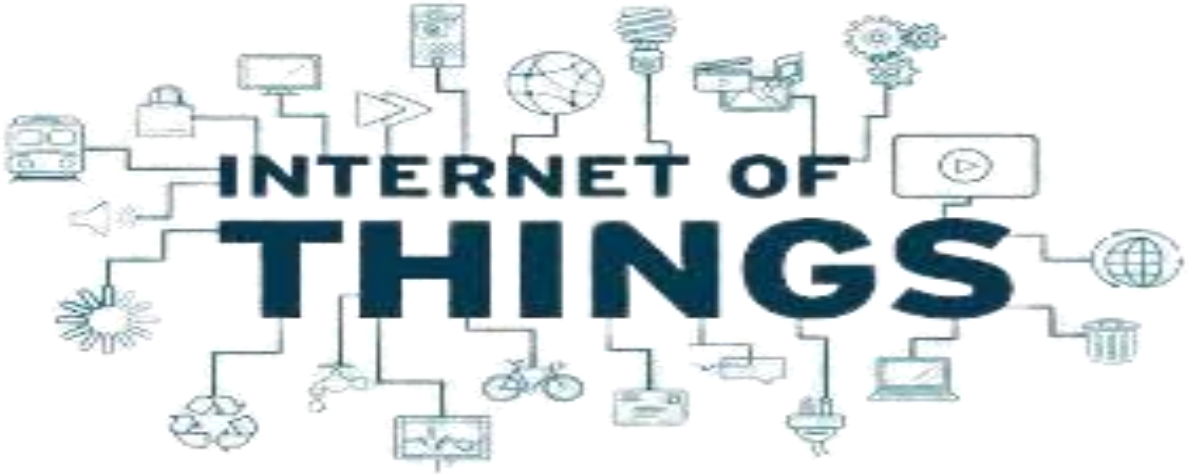
INTERNET OF THINGS (IOT)

SMART WATER MANAGEMENT:

WATER QUALITY MONITORING PUBLIC SWIMMING POOL:



PROJECT REPORT PHASE 3

SUBMITTED BY

TEAM LEADER: MOAMMED JAFFER K

TEAM MEMBERS: HEMA KUMAR S

TEAM MEMBERS: NAVEEN KUMAR V

TEAM MEMBERS: THUTHESH KUMAR R

INTRODUCTION:

Public swimming pools provide a refreshing escape for countless individuals seeking respite from the sweltering heat or an avenue for fitness and recreation. However, beneath the crystal-clear surface of these aquatic sanctuaries lies a complex world of chemical reactions, microbial life, and potential health risks. Ensuring the safety and well-being of swimmers in public pools demands the vigilant oversight of water quality.

Water quality monitoring in public pools is a paramount responsibility of pool operators, management, and regulatory authorities. It encompasses a multifaceted approach aimed at preserving the health, hygiene, and enjoyment of all who venture into the pool's waters. The essence of this monitoring lies in the systematic evaluation and maintenance of various water parameters that collectively define the pool's aquatic environment.

COMPONENTS REQUIRE:

- Arduino UNO
- NodeMCU
- Gravity Analog pH sensor
- DS18B20 temperature sensor
- Soil moisture sensor
- Power supply
- Jumpers
- Breadboard

COMPONENTS DESCRIPTION:

ARDUINO:

Arduino is an open-source electronics platform and a family of microcontroller boards designed for building and prototyping a wide range of digital devices and interactive objects. It provides both the hardware and software necessary

to create and control various electronic projects, from simple LED blinking experiments to complex robotics and automation applications.

Connect the sensors and provide power to the NodeMCU.

Connect the 5V pin from Arduino UNO to VCC (or power) on the NodeMCU.

Connect GND from Arduino UNO to GND on the NodeMCU.

Connect Arduino UNO's TX (Transmit) to NodeMCU's RX (Receive).

Connect Arduino UNO's RX (Receive) to NodeMCU's TX (Transmit).

Arduino	PH Sensor board
5V	V+
GND	G
A0	Po

NODEMCU:

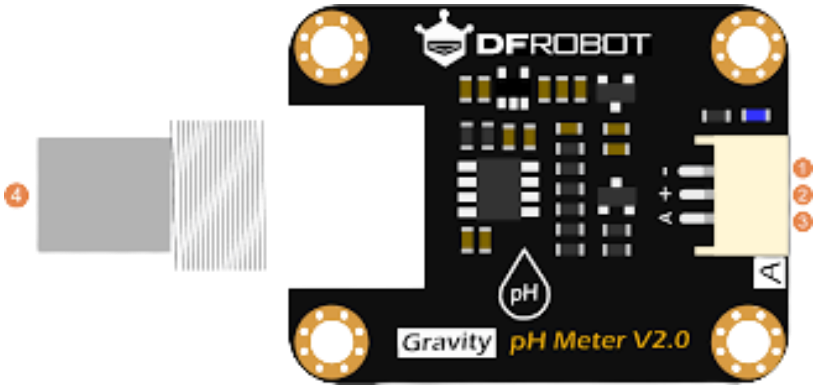
NodeMCU is an open-source development board that integrates the ESP8266 microcontroller and provides a platform for building IoT projects. It features built-in WiFi connectivity, a USB interface for easy programming, and GPIO pins for interfacing with various sensors and components. NodeMCU supports the Arduino IDE, Lua scripting language, and offers an accessible and versatile solution for developing WiFi-enabled applications and devices.

Connect the pH sensor to an analog pin on the NodeMCU

Power the pH sensor using 3.3V from the NodeMCU and connect its GND to NodeMCU's GND.

GRAVITY ANALOG PH SENSOR:

The "Gravity Analog pH Sensor" is a specific pH (acidity or alkalinity) measurement sensor designed for use in electronic projects and applications. It is part of the "Gravity" series, which is a line of electronic sensors and modules developed by DFRobot, a company specializing in open-source hardware and robotics. The Gravity series is known for its ease of use and compatibility with various development platforms, including Arduino and Raspberry Pi.



Pin Description:

V+: 5V DC input

G: Ground pin

Po: pH analog output

Do: 3.3V DC output

To: Temperature output

DS18B20 TEMPERATURE SENSOR:

The DS18B20 is a digital temperature sensor known for its precision and versatility. It is part of the Dallas Semiconductor 1-Wire family of sensors and is widely used in various applications for measuring temperature.

Pinout of DS18B20:

- VCC: Power input: (3.3 – 5) V DC
- Ground: Ground pin of the circuit
- Data: 1 Wire temperature value data output pin

SOIL MOISTURE SENSOR:

A soil moisture sensor is an electronic device designed to measure and monitor the moisture content or water level in the soil.

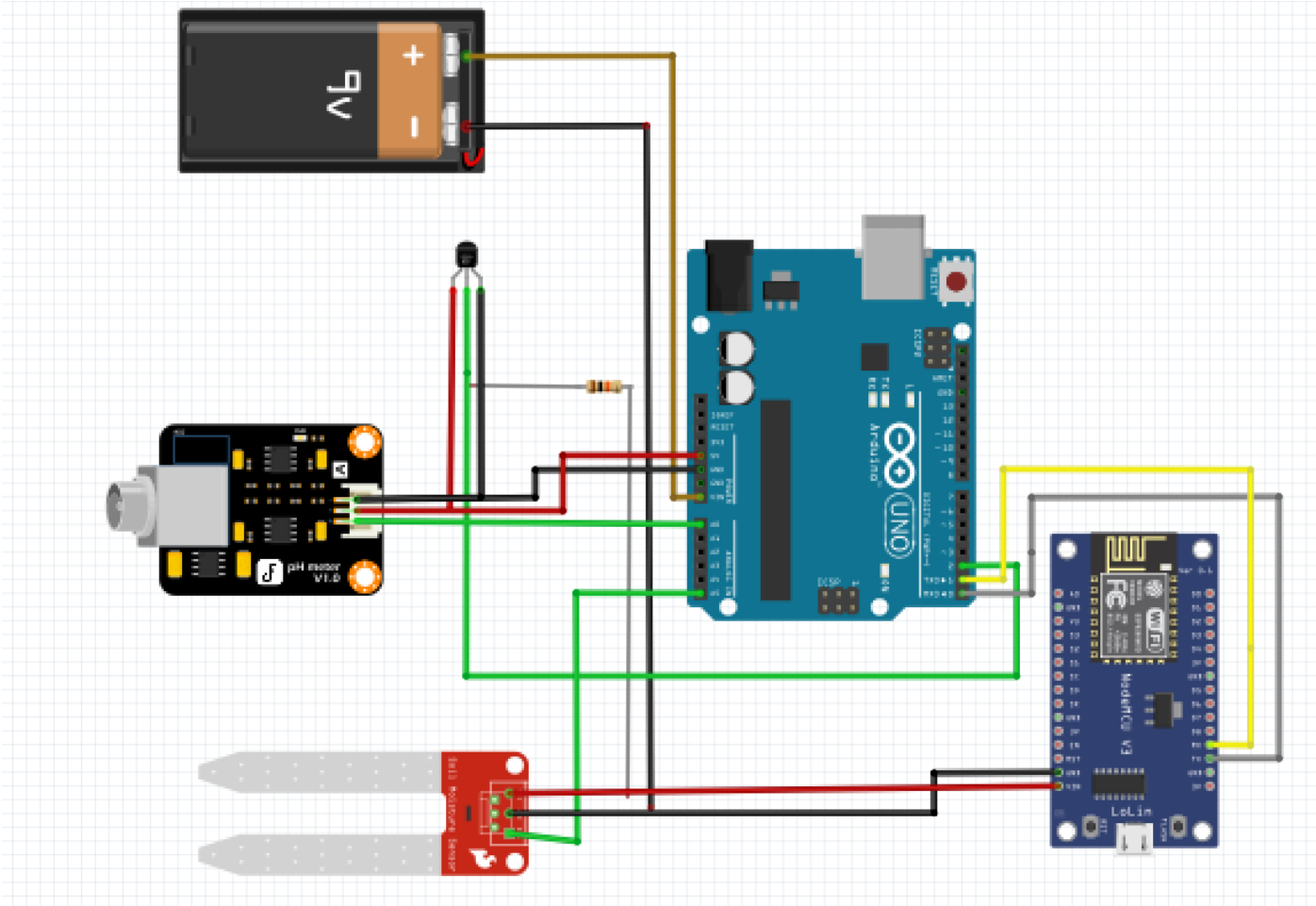
It provides data about the soil's ability to retain and supply water to plants, making it a valuable tool in agriculture, gardening, and environmental monitoring. These sensors help in making informed decisions related to irrigation, water management, and crop health.

Connect the soil moisture sensor's VCC to the NodeMCU's 3.3V.

Connect the soil moisture sensor's GND to the NodeMCU's GND.

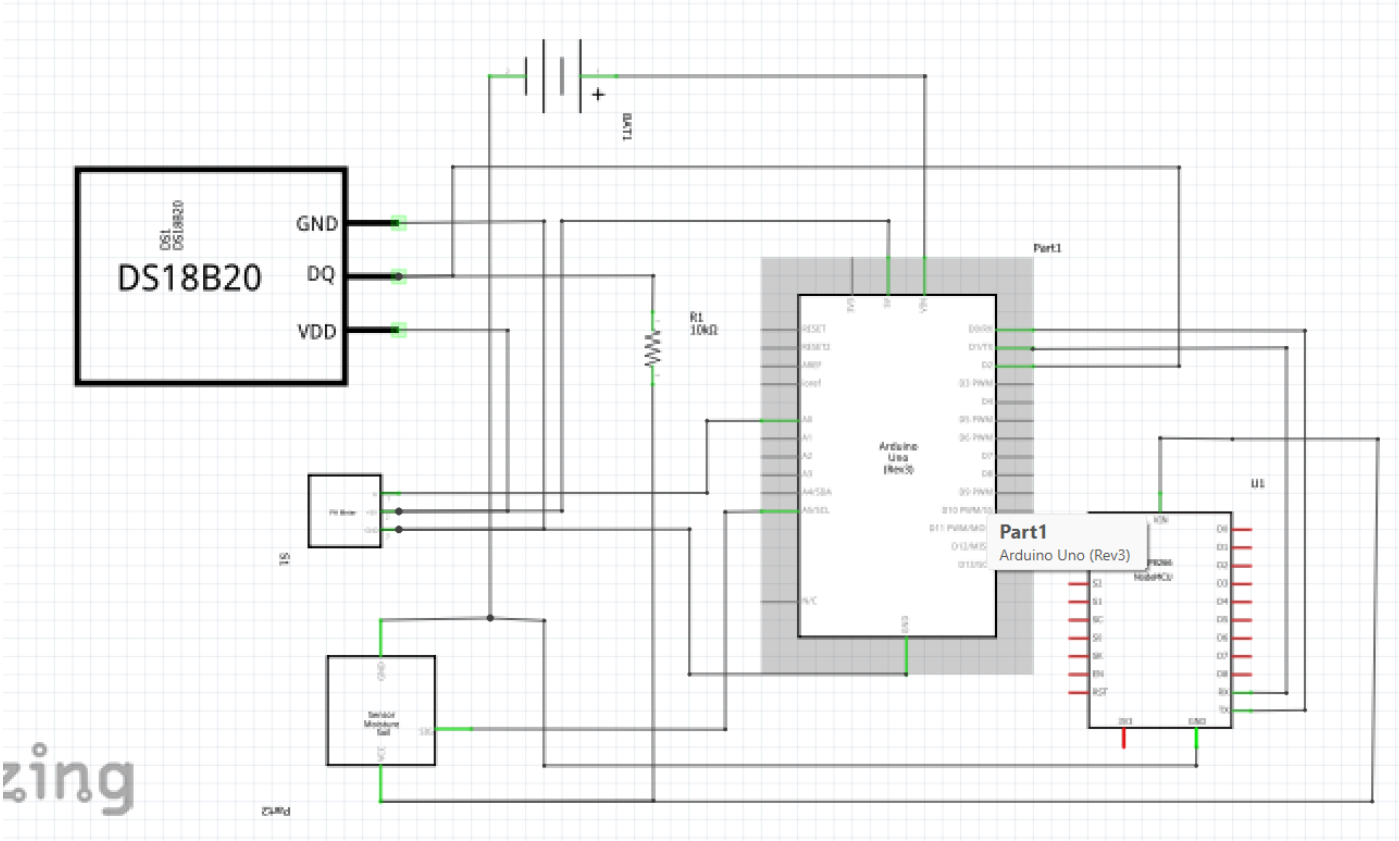
Connect the sensor's analog output to an analog pin on the Arduino UNO

CIRCUIT DESIGN:



WATER QUALITY MONITORING PUBLIC SWIMMING POOL CIRCUIT DESIGN

PCB DESIGN:



WATER QUALITY MONITORING PUBLIC SWIMMING POOL PCB DESIGN

PYTHON SCRIPT:

```
# Arduino Code

#include <OneWire.h>

#include <DallasTemperature.h>

#include <ArduinoJson.h>

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>


OneWire oneWire(2);

DallasTemperature temp_sensor(&oneWire);

float calibration_value = 21.34;

int phval = 0;

unsigned long int avgval;

int buffer_arr[10], temp;


const char* ssid = "your_SSID";  // Replace with your network SSID

const char* password = "your_PASSWORD";  // Replace with your network password

ESP8266WebServer server(80);

String page = "";

int data1, data2, data3;


void setup() {

  Serial.begin(9600);
```

```
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {

    delay(500);

    Serial.print(".");

  }

  Serial.println(WiFi.localIP());

  server.on("/", []() {

    page = "<html><head><title>IoT Design</title></head><style type=\"text/css\">";

    page += "table{border-collapse: collapse;}th {background-color: green;color: white;}table,td {border: 4px solid black;font-size: x-large;";

    page += "text-align:center;border-style: groove;border-color: rgb(255,0,0);}</style><body><center>";

    page += "<h1>Smart Aquaculture Monitoring using IoT</h1><br><br><table style=\"width: 1200px;height: 450px;\"><tr>";

    page += "<th>Parameters</th><th>Value</th><th>Units</th></tr><tr><td>PH Value</td><td>" + String(data1) + "</td><td>N/A</td></tr>";

    page += "<tr><td>Temperature</td><td>" + String(data2) + "</td><td>Centigrade</td></tr><tr><td>Moisture</td><td>" + String(data3) + "</td><td>%</td>";

    page += "<meta http-equiv=\"refresh\" content=\"3\">";

    server.send(200, "text/html", page);

  });

  server.begin();

  temp_sensor.begin();

}



void loop() {
```

```
for (int i = 0; i < 10; i++) {
  buffer_arr[i] = analogRead(A0);
  delay(30);
}
for (int i = 0; i < 9; i++) {
  for (int j = i + 1; j < 10; j++) {
    if (buffer_arr[i] > buffer_arr[j]) {
      temp = buffer_arr[i];
      buffer_arr[i] = buffer_arr[j];
      buffer_arr[j] = temp;
    }
  }
}
avgval = 0;
for (int i = 2; i < 8; i++)
  avgval += buffer_arr[i];
float volt = (float)avgval * 5.0 / 1024 / 6;
float ph_act = -5.70 * volt + calibration_value;
temp_sensor.requestTemperatures();
int moisture_analog = analogRead(A1);
int moist_act = map(moisture_analog, 0, 1023, 100, 0);

StaticJsonBuffer<1000> jsonBuffer;
JsonObject& root = jsonBuffer.createObject();
```

```
  root["a1"] = ph_act;

  root["a2"] = temp_sensor.getTempCByIndex(0);

  root["a3"] = moist_act;


  // Send data as JSON over Serial

  root.printTo(Serial);


  // Update local variables

  data1 = root["a1"];

  data2 = root["a2"];

  data3 = root["a3"];


 zserver.handleClient();

}
```