Smart Aquarium System Guide

System Overview

The Smart Aquarium System is an automated monitoring and maintenance solution designed to keep your aquarium water clean and healthy with minimal effort. It uses a pH sensor to continuously check the water quality, and when the pH level goes beyond the safe range, the system automatically activates a water change using a water pump. The entire setup is controlled by the Arduino Mega and the data is sent to the cloud via ESP8266 it is then connected to a mobile app, where you can view real-time pH readings, water level percentage, receive alerts, and track your aquarium's water change history. With this system, you can ensure a safer environment for your fish without the hassle of manual testing and cleaning.

Hardware Components

- **Arduino Mega**
  Arduino Mega is the main controller that manages the entire system. It processes data from the pH sensor and ultrasonic sensor, makes decisions based on the readings, and controls the two water pumps through the relay module. With its large number of pins and memory, it’s perfect for handling multiple sensors and modules at once.

- **PH-4502C pH Sensor**
  A sensor that monitors the pH level of your aquarium water in real time. It detects if the water is too acidic or too alkaline, which could harm your fish. The sensor sends this data to the Arduino, which then decides whether to activate the water change process. Regular cleaning and calibration help maintain its accuracy.

- **LCD with I2C 1602**
  This is a small digital display that shows important information such as the current pH level and system status. The I2C interface simplifies wiring, using only two communication lines. This makes it easy to keep track of what's happening in your aquarium without needing to check the mobile app.

- **JSN-SR04M-2 Ultrasonic Sensor**
  A waterproof sensor is used to measure the water level in your aquarium. It sends out ultrasonic pulses and measures the time it takes for them to bounce back from the water

surface. This helps ensure the system doesn't overflow the tank.

- **ESP8266 Wi-Fi Module**
  The ESP8266 allows your system to connect to Wi-Fi and communicate with your mobile app. It sends real-time pH data and water level data to Firebase and then it’s being sent to your mobile app, allowing you to monitor the aquarium remotely.

- **Water Pump (x2)**
  There are two pumps in the system: one for draining dirty water and one for filling clean water into the aquarium. These pumps are controlled automatically by the Arduino via the relay module, based on the pH level and water level readings.

- **Buck Converter**
  This module adjusts and regulates voltage from the power supply to match the requirements of different components, such as stepping down 9V to 5V or 3.3V. It ensures each device gets safe and stable power, helping to prevent overheating or damage.

- **Relay Module**
  The relay module acts like a smart switch that allows the Arduino to control high-power components, such as the water pumps. When the Arduino sends a signal, the relay either turns a pump on or off. Each relay channel is connected to one of the two pumps, enabling separate control for draining and refilling.

- **Power Supply**
  This provides the necessary electricity to run all components in the system. It powers the Arduino, sensors, pumps, and the Wi-Fi module. A reliable power source is essential for keeping the system stable and operational 24/7.

Recommended Power-Up Sequence

1. **Plug the Adapter for the Water Pumps -** This ensures your high-power devices (the pumps) get stable power first.

2. **Power the LCD -** The adaptor for the LCD has a green electrical tape

3. **Power the Arduino Mega** - It will boot up, initialize I/O pins, and prepare for sensor reading/control.

4. **Turn on the ESP8266 -** After the Arduino is ready, switch on the ESP8266.

If you are turning it off, unplug in order 4-3-2-1.

Issues Encountered

The current setup uses time-based control (in seconds) for operating the pumps because the ultrasonic sensor isn't reliable at detecting the water level. If you want to adjust how long the pump drains or refills water, you'll need to modify the code accordingly to match your aquarium's requirements for your demo presentation. The code is provided below and the comments highlighted in red is what you need to edit to change the time.

⚠️If you are plugging the blue USB cord make sure that the Arduino adaptor is unplugged. If you plug it both at the same time it can damage the Arduino mega.

```
C/C++
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <NewPing.h>
#include <SoftwareSerial.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);
#define PUMP1_PIN 9
#define PUMP2_PIN 8

SoftwareSerial espSerial(5, 6);
float calibration_value = 21.7;
int phval = 0;
unsigned long int avgval;
int buffer_arr[10], temp;

#define TRIGGER_PIN  2
#define ECHO_PIN     3
#define MAX_DISTANCE 400
#define MAX_WATER_LEVEL_CM 35
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

float tempval1;
float tempval2;
int finalval;

unsigned long lastRefillTime = 0;
const unsigned long refillCooldown = 12UL * 60UL * 60UL * 1000UL;

void setup() {
```

```
  Serial.begin(9600);
  espSerial.begin(115200);

  lcd.init();
  lcd.begin(16, 2);
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("   Welcome to   ");
  lcd.setCursor(0, 1);
  lcd.print("  Smart Aqua    ");
  delay(2000);
  lcd.clear();

  pinMode(PUMP1_PIN, OUTPUT);
  pinMode(PUMP2_PIN, OUTPUT);
  digitalWrite(PUMP1_PIN, HIGH);
  digitalWrite(PUMP2_PIN, HIGH);
}

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;

  delay(20);
  int iterations = 10;
  tempval1 = ((sonar.ping_median(iterations) / 2) * 0.0343);
```

```
  if (tempval1 - tempval2 > 60 || tempval1 - tempval2 < -60) {
    tempval2 = (tempval1 * 0.02) + (tempval2 * 0.98);
  } else {
    tempval2 = (tempval1 * 0.4) + (tempval2 * 0.6);
  }

  finalval = tempval2;

  int water_percent = map(finalval, 0, MAX_WATER_LEVEL_CM, 100, 0);
  if (water_percent > 100) water_percent = 100;
  if (water_percent < 0) water_percent = 0;

  lcd.setCursor(0, 0);
  lcd.print("pH Value: ");
  lcd.setCursor(10, 0);
  lcd.print(ph_act, 2);

  lcd.setCursor(0, 1);
  lcd.print("Water Level: ");
  lcd.setCursor(12, 1);
  lcd.print(water_percent);
  lcd.print("%   ");

  Serial.print("pH: ");
  Serial.print(ph_act, 2);
  Serial.print(" | Water Level: ");
  Serial.print(water_percent);
  Serial.println("%");

  espSerial.println(String(ph_act, 2) + "," + String(water_percent));

  if (ph_act > 8.4 || ph_act < 6) {
    Serial.println("Abnormal pH detected. Activating Pump 1 for 10s...");
    digitalWrite(PUMP1_PIN, LOW);
    delay(30000); //Edit this for the pump out of the aquarium (currently 30
seconds if you want 1 min then 60000)
    digitalWrite(PUMP1_PIN, HIGH);

    while (true) {
      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];
      volt = (float)avgval * 5.0 / 1024 / 6;
      ph_act = -5.70 * volt + calibration_value;

      tempval1 = ((sonar.ping_median(iterations) / 2) * 0.0343);
      finalval = (tempval1 * 0.4) + (tempval2 * 0.6);
      tempval2 = finalval;

      int currentWaterPercent = map(finalval, 0, MAX_WATER_LEVEL_CM, 100, 0);
      currentWaterPercent = constrain(currentWaterPercent, 0, 100);

      lcd.setCursor(0, 0);
      lcd.print("pH Value: ");
      lcd.setCursor(10, 0);
      lcd.print("     ");
      lcd.setCursor(10, 0);
      lcd.print(ph_act, 2);

      lcd.setCursor(0, 1);
      lcd.print("Water Level: ");
      lcd.setCursor(12, 1);
      lcd.print("    ");
      lcd.setCursor(12, 1);
      lcd.print(currentWaterPercent);
      lcd.print("%");

      Serial.println("Waiting 10s before refilling...");
      delay(10000);

      digitalWrite(PUMP2_PIN, LOW);
      delay(30000);//Edit this for refill (currently 30 seconds if you want 1
min then 60000)
      digitalWrite(PUMP2_PIN, HIGH);
```

```
      Serial.print("Refilling... Water Level: ");
      Serial.println(ph_act, 2);
      Serial.println("Water level is 90% or more. Stopping Pump 2.");
      break;
    }
    Serial.println(" Exiting refill loop.");
  }
  delay(1000);
}
```

Smart Aquarium Wi-Fi Setup Guide

If your mobile app doesn't match the values on the aquarium's LCD display (like pH or water level), it might mean your device is not connected to your home Wi-Fi. When this happens, don't worry your Smart Aquarium will create its own Wi-Fi signal to help you set it up again.

Follow these steps:

1. **Open Wi-Fi Settings** on your phone, tablet, or laptop.

2. Look for a Wi-Fi network called:

```
Unset
SmartAquaSetup
```

3. and connect to it.
   Password: `smartpass123`

4. Once connected, your device may open a setup page automatically.
   If not, open a browser and go to:

```
Unset
http://192.168.4.1
```

5. On the setup page:
    - Choose your **home Wi-Fi network** from the list.
    - Enter your Wi-Fi **password**.
    - Click **Save** or **Connect**.
6. The Smart Aquarium will now try to connect to your Wi-Fi.
7. Once connected, the SmartAquaSetup signal will disappear.
8. Open the **Smart Aquarium app** and check if values like **pH** and **water level** are updating. If they are, you're all set!