Islamic University of Technology
Organization of Islamic Cooperation (OIC)
Electrical and Electronic Engineering Department

Project Nema : An Automated Aquarium

A Project Report

Members:

Safwan Sami (ID: 200021221)

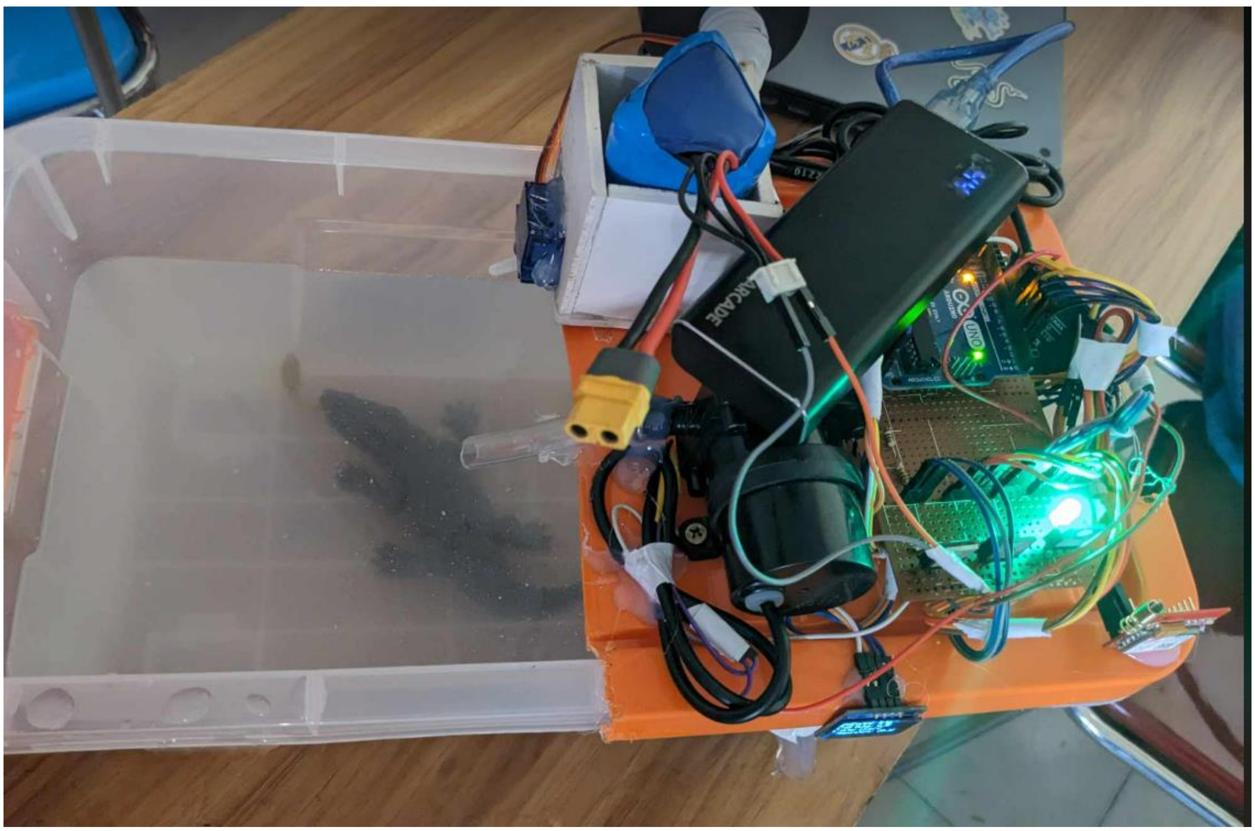
Ashfak Tashin (ID: 200021227)

Sayeed Hossain Sayor (ID: 200021236)

M. M. Sadman Shafi (ID: 200021240)

Shafayet Sadik Sowad (ID: 200021248)

Table of Contents:



Abstract:

This project report presents the development of an innovative and automated aquarium system leveraging Arduino technology. The primary objective is to create a comprehensive ecosystem within the aquarium through the integration of various sensors and actuators. The system incorporates pH sensors, a water level indicator, temperature control mechanisms, a GSM module for remote monitoring, and a servo motor for precise and automated feeding. The implementation allows for real-time monitoring and control, ensuring optimal environmental conditions for aquatic life while enabling remote accessibility for seamless management. This amalgamation of sensors, actuators, and Arduino-based automation heralds a new era in aquarium maintenance, enhancing the user experience and fostering an ecosystem conducive to thriving aquatic life.

Key Points:

pH sensors, Water level indicator, Temperature control, GSM module, Servo Motor, Automated Feeding, Arduino automation, real-time monitoring, remote accessibility, optimal aquatic environment, Iot implementation.

1.Introduction:

1.1 Purpose:

In the realm of aquatic habitat maintenance, the amalgamation of technology and ecological preservation has taken a pioneering leap. Our project endeavors to redefine the paradigm of aquarium management, where precision meets environmental sensitivity through the fusion of cutting-edge technology and intricate ecological understanding.

1.2 Objective:

Ecosystem Harmony: Forge an automated system that not only mimics but actively sustains a harmonious aquatic ecosystem by regulating critical environmental parameters such as pH levels and temperature.

Technical Ingenuity: Confront the complexities of integrating diverse technological components, including GSM modules for remote monitoring, servo motors for automated feeding, and precise control mechanisms for maintaining optimal water levels.

Environmental Precision: Attain a level of precision that caters to the nuanced needs of aquatic life, ensuring a stable and conducive environment that fosters their well-being and longevity.

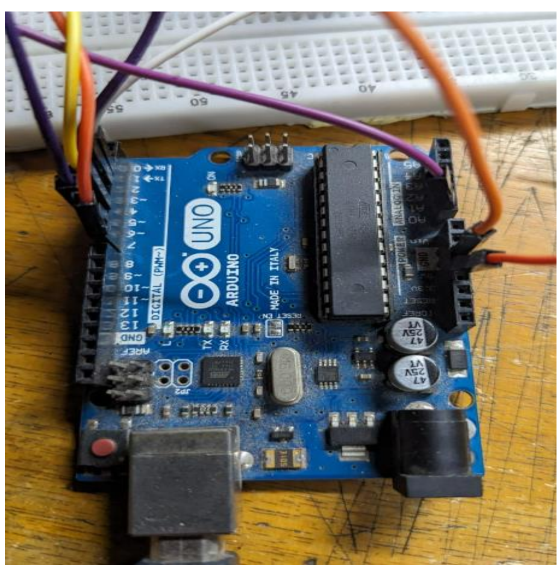
Symbiosis of Automation and Ecology: Bridge the gap between technological automation and ecological sensitivity, demonstrating how sophisticated systems can intricately coexist with, and enhance, the natural environment within an aquarium setting.

2. Overview of the components.

2.1 Principal Components:

2.1.1 Arduino Uno:

The Arduino Uno acts as the central intelligence in our automated aquarium project, orchestrating sensor data interpretation and precise control of actuators. It manages pH sensors, water level indicators, temperature controls, a GSM module for remote monitoring, and servo motors for automated feeding. Its scalability allows seamless integration of new sensors or features, ensuring adaptability and continuous evolution of the system to maintain an optimal aquatic environment.



2.1.2 GSM Module:

Integrated within our automated aquarium setup, the GSM module plays a pivotal role in enabling instant communication with the user. Coupled with the Arduino Uno, this module allows for the transmission of real-time updates and notifications directly to the user's mobile device. In case of any deviations in the aquarium's vital parameters, such as pH levels or temperature fluctuations beyond set thresholds, the GSM module triggers immediate alerts. This functionality empowers users with timely information, ensuring proactive intervention or adjustments to maintain an optimal aquatic environment, thus fostering a seamless connection between the aquarium and its caretaker regardless of physical proximity.

2.1.3 Temperature Sensor:

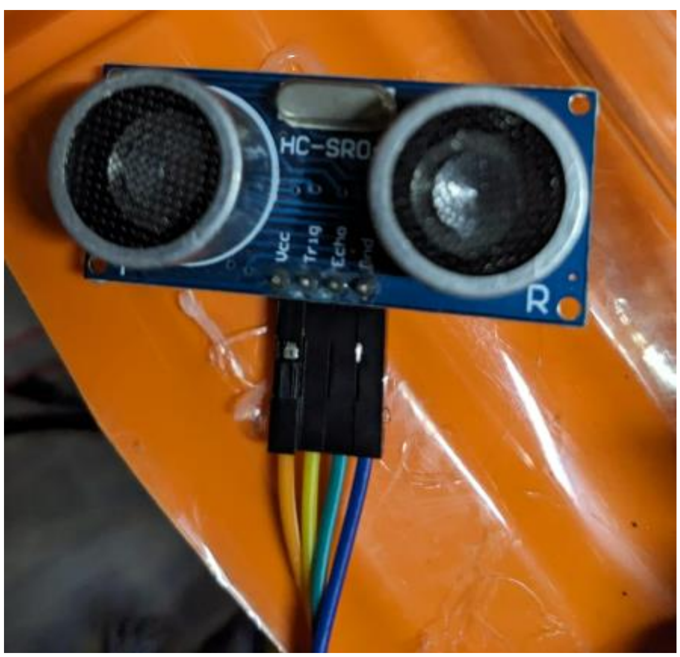
The temperature sensor within our automated aquarium system serves as the vigilant guardian of thermal stability. Linked to the Arduino Uno, this sensor continuously monitors the water temperature with precision. It ensures the aquatic environment remains within the predefined temperature range vital for the well-being of aquatic life. The sensor's role extends beyond mere measurement; it actively triggers necessary adjustments through the system, guaranteeing a consistent and comfortable habitat for the aquarium's inhabitants. Its seamless integration with the system underscores its significance in maintaining the delicate balance of the ecosystem.

2.1.4 pH Sensor:

At the core of our automated aquarium's chemical equilibrium lies the pH sensor, functioning as the custodian of water quality. Collaborating with the Arduino Uno, this sensor vigilantly scrutinizes the water's pH levels, a critical factor in sustaining a healthy aquatic environment. In the event of pH fluctuations beyond optimal ranges, the sensor communicates with the system to implement corrective measures. Its role goes beyond detection; it is an instrumental component in the system's ability to swiftly respond and restore the ideal pH balance, ensuring the well-being and longevity of the aquarium's inhabitants.

2.1.5 Water Level Indicator:

The SONAR-based water level indicator in our automated aquarium system uses ultrasonic waves to precisely measure water levels. Integrated with the Arduino Uno, it ensures the aquarium maintains an ideal water volume, triggering necessary actions for stability and the well-being of aquatic life.



2.1.6 Heater:

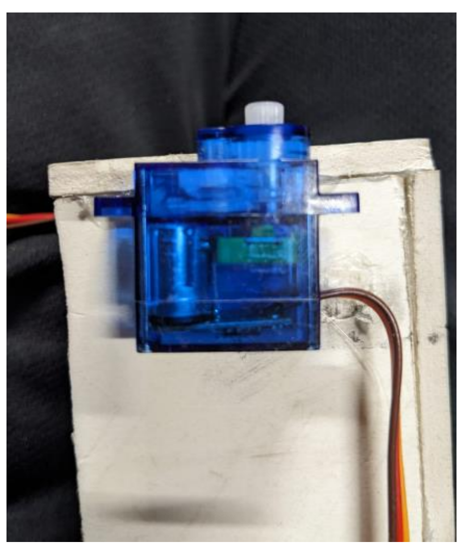
Paired with the Arduino Uno, the heater uses precise temperature control for maintaining optimal water conditions within the aquarium.

2.1.7 Water Pump for Changing Water:

Integrated with the Arduino Uno, the water pump facilitates periodic water changes, enhancing circulation and maintaining water quality in the aquarium.

2.1.8 Servo Motor:

Integrated with the Arduino Uno, the servo motor enables precise and automated fish feeding by controlling the dispensing mechanism, ensuring consistent and scheduled nourishment for aquarium inhabitants.



2.2 Basic Component:

Vero Board.

Bread Board.

Resistor.

MOSFET.

OLED Display.

Jumper Wire.

LED Bulb.

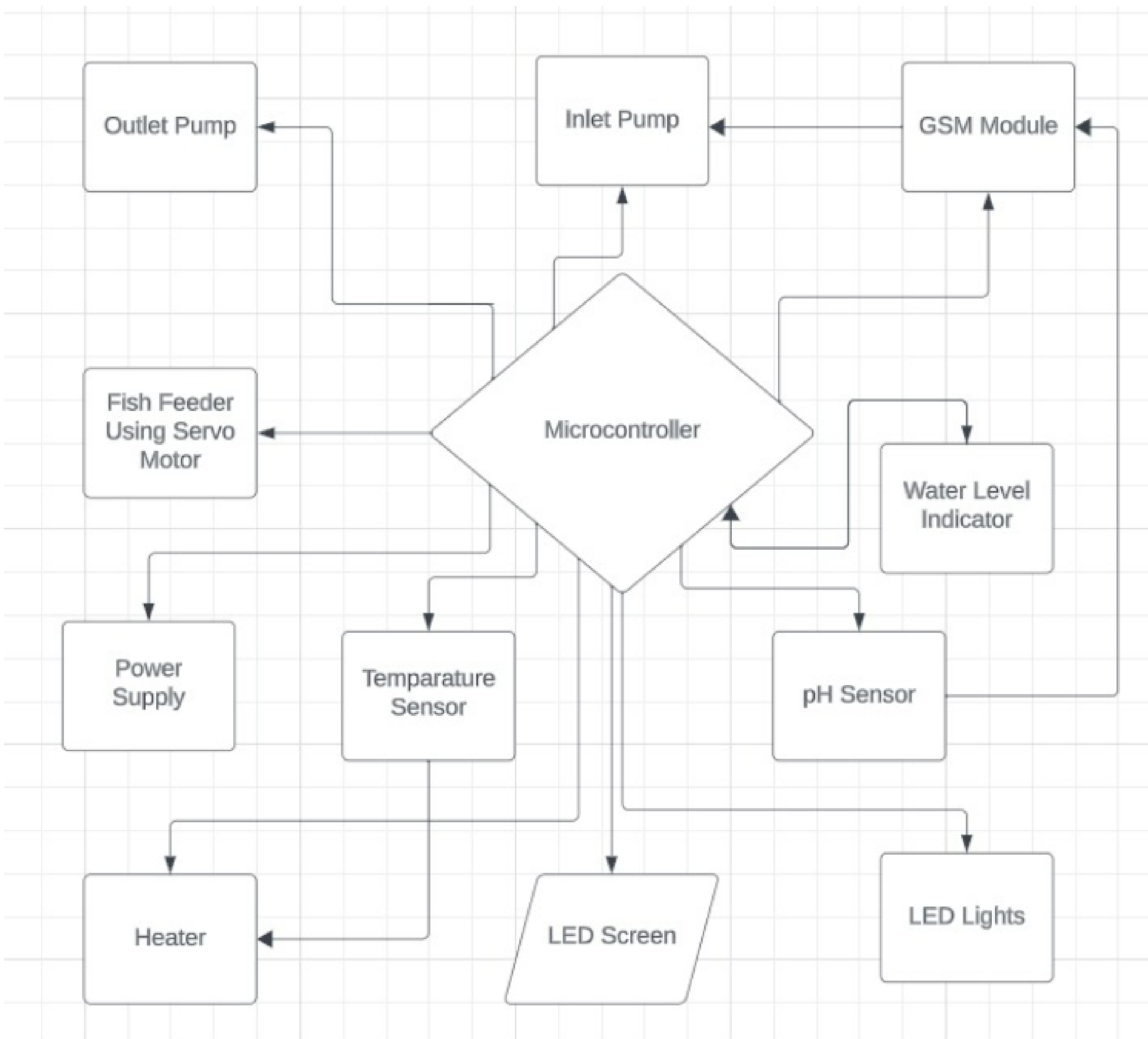
Pipes.

Demo Aquarium.

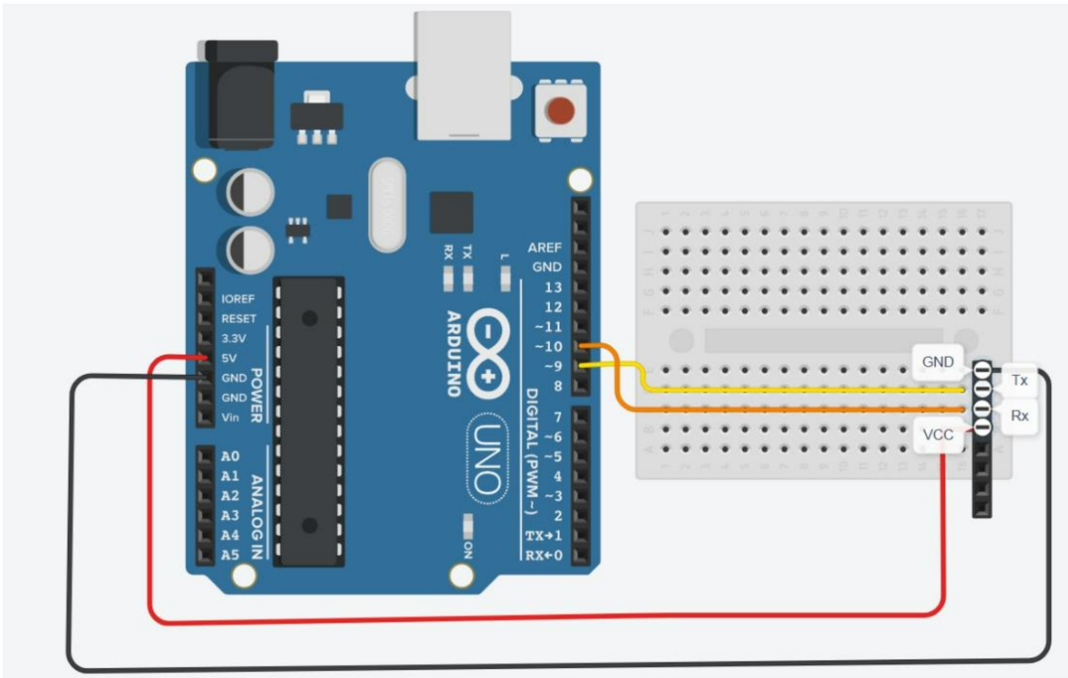
Pure Water.

3. System Operation Overview:

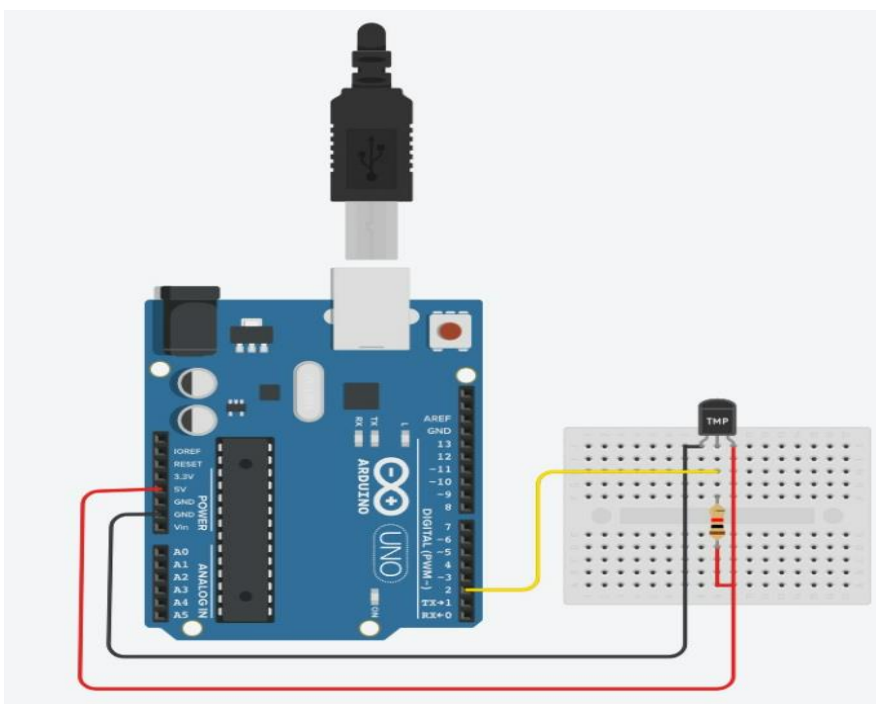
3.1 Methodology Flow:



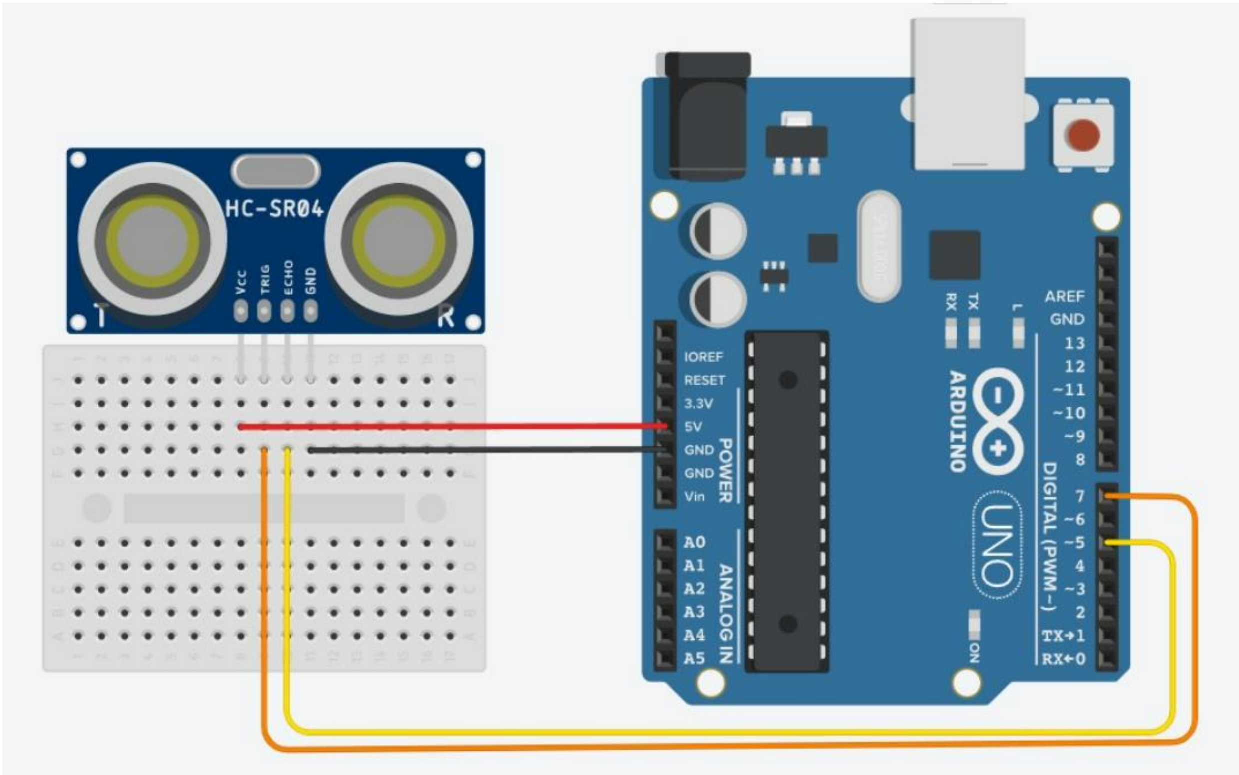
3.2 Circuit Diagram:
3.2.1 GSM:



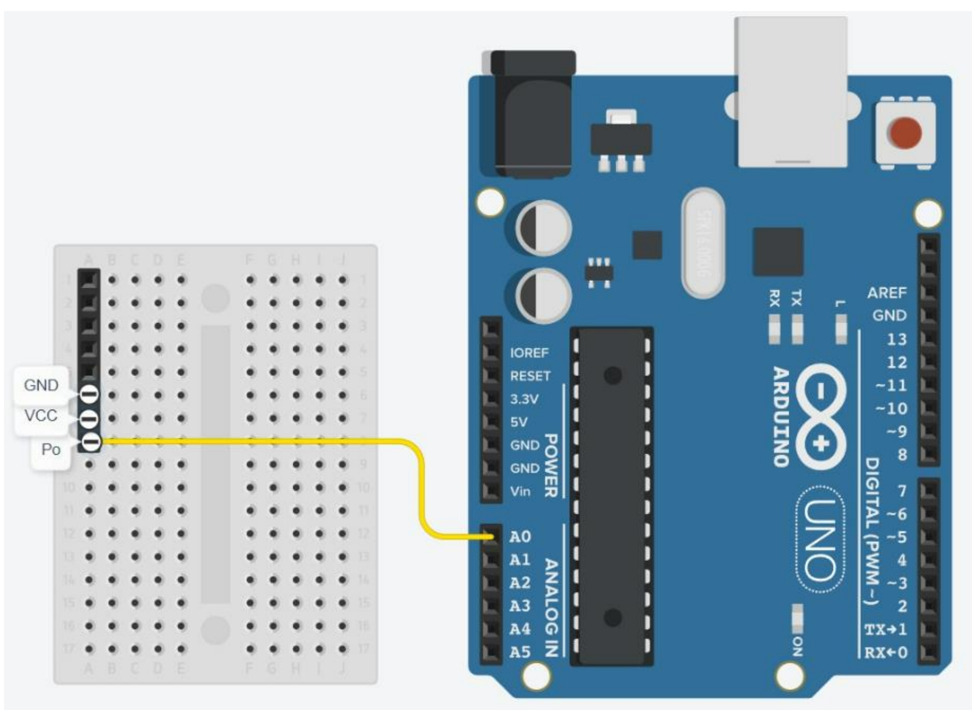
3.2.2 Temperature Sensor:



3.2.3 SONAR:



3.2.4 pH sensor:



3.3 Functionality Analysis:

3.3.1 Sensor Data Display on OLED

The system gathers sensor readings, including pH, temperature, and water level, and displays them on an OLED screen. This real-time data presentation on the OLED allows users to monitor crucial aquarium parameters conveniently and promptly.

3.3.2 pH Sensor & GSM Integration

When the pH sensor detects values beyond defined thresholds, it triggers the GSM module to send immediate alerts or notifications to the user's mobile device. This integrated functionality serves as a proactive warning system, ensuring timely updates about critical changes in water pH.

3.3.3 User-initiated Water Change via GSM & Pump Control

Upon receiving a user-initiated message through the GSM module, the system activates the water pump to facilitate a water change within the aquarium. This interaction allows users remote control over the aquarium's water quality. By triggering the pumps, inlet and outlet, the system responds to user commands promptly, ensuring a dynamic and user-friendly approach to maintaining optimal water pH conditions.

3.3.4 Heater Activation and Temperature Monitoring with LED Indication

When the temperature sensor detects a drop below a predefined threshold, the Arduino activates the heater. Simultaneously, an LED indicator illuminates, signaling the heater's activation. This integrated functionality offers a dual-layered approach to maintaining optimal temperatures within the aquarium.

3.3.5 Automated Feeding Schedule with Servo Motor Activation

After a 12-hour interval, the Arduino triggers the servo motor, initiating the feeding mechanism within the aquarium. This automated feeding schedule ensures consistent nourishment for the aquatic inhabitants. By engaging the servo motor at specific intervals, the system optimizes the feeding process, providing sustenance without relying on manual intervention.

4. Features:

4.1 Automation Highlights

Intelligent Sensor Integration: Our project seamlessly combines various sensors—pH, temperature, and water level indicators—interfaced with the Arduino board. Through meticulously crafted Arduino code, these sensors provide real-time data insights crucial for maintaining an optimized aquatic environment. For instance, utilizing conditional statements in the Arduino code, we establish thresholds for sensor readings. When thresholds are breached, the system autonomously triggers corresponding actions, such as activating the heater when the temperature falls below a predefined value or sending immediate alerts via GSM upon pH irregularities.

Precise Actuation with Servo Motor Control: Leveraging the Arduino's capabilities, we've incorporated servo motor functionality for precise and scheduled feeding automation. Arduino's codebase orchestrates feeding schedules, activating the servo motor to dispense fish feed at predefined intervals. This functionality involves the creation of timed loops and servo motor control sequences within the Arduino code.

4.2 Advanced Monitoring Techniques

Real-Time Data Visualization on OLED Display: Through Arduino's code implementation, our system collects and displays crucial sensor data in real-time on an OLED screen. This feature is achieved by integrating sensor readings into the Arduino code and interfacing with the OLED display. The Arduino code utilizes libraries and functions to read sensor data and then formats and presents it on the display. This instant visual representation empowers users with immediate insights into vital parameters like pH, temperature, and water levels, allowing swift decision-making for aquarium maintenance.

GSM-Enabled Remote Monitoring and Alert System: Our system employs Arduino's GSM module integration to enable remote monitoring and alert functionalities. By incorporating GSM capabilities within the Arduino code, the system triggers alerts or notifications via SMS when sensor readings breach predefined thresholds. This feature involves conditional statements and GSM libraries in the Arduino code, ensuring prompt user notifications in case of critical changes in the aquarium environment. The remote accessibility provided by GSM connectivity enhances user convenience and enables proactive intervention, even from a distance, ensuring the aquarium's stability and health.

4.3 Algorithmic Control Precision

Conditional Logic for Sensor-Triggered Actions: The system's control algorithm, embedded within the Arduino code, employs conditional logic. These logical sequences interpret sensor data, enabling swift decision-making based on predefined thresholds. For instance, utilizing if-else statements, the code triggers specific actions—such as activating the heater or initiating pump operations—when sensor readings surpass established limits. This granular control ensures an immediate response to maintain optimal aquarium conditions.

Timed Sequences for Scheduled Automation: Incorporated within the Arduino code are timed sequences governing scheduled actions, such as feeding schedules. Employing loops and timing functions, the code activates the servo motor at defined intervals to dispense fish feed. This timed automation fosters consistency and precision, providing a dependable and predictable feeding regimen beneficial for the aquatic ecosystem.

5. Implementation:

5.1 Codes:

```
#include <Wire.h>  //libraries

#include <Adafruit_SSD1306.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include "SoftwareSerial.h"

#include <Servo.h>


Servo myservo;
```

```
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
#define TRIG_PIN 7
#define ECHO_PIN 5
#define ONE_WIRE_BUS 2
#define TEMP_ALERT_PIN 8


float calibration_value = 34.50; //for ph
int phval = 0;//ph
unsigned long int avgval;//ph
int buffer_arr[10], temp;//ph

OneWire oneWire(ONE_WIRE_BUS);//temp
DallasTemperature sensors(&oneWire);//temp

SoftwareSerial mySerial(9, 10);//for gsm
String cmd = "";//gsm

const int motor1 = 6; //for motor
const int motor2 = 3;//motor

bool msg_sent_ph = false; //boolean variable

void setup() {
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(TEMP_ALERT_PIN, OUTPUT);
```

```
pinMode(motor1, OUTPUT);
pinMode(motor2, OUTPUT);

myservo.attach(11); //servo
sensors.begin();

mySerial.begin(9600);
```

//display

```
Serial.begin(9600);
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(50, 0);
display.println("EEE");
display.setCursor(50, 10);
display.println("4518");
display.display();
delay(1000);
```

//gsm initiating

```
Serial.println("Initializing...");
delay(1000);
mySerial.println("AT");
updateSerial();
mySerial.println("AT+CMGF=1");
updateSerial();
```

```
  mySerial.println("AT+CNMI=1,2,0,0,0");
  updateSerial();
 }

void loop() {
 //ph
  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;

//ph logic to gsm
```

```
  if (ph_act > 10 && msg_sent_ph == false) {
    mySerial.println("AT+CMGS=\"+8801601313077\"");
    updateSerial();
    mySerial.print("Boss I'm dying.");
    updateSerial();
    mySerial.write(26);

    msg_sent_ph = true;
  }

  if(ph_act <= 5)
    msg_sent_ph = false;

//distance
  long duration, distance;
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  duration = pulseIn(ECHO_PIN, HIGH);
  distance = (duration / 2) / 29.1;

//Check water level value
  if (distance >=12) {
    digitalWrite(motor1, LOW);
    digitalWrite(motor2, HIGH);
  } else if (distance <= 4) {
```

```
    digitalWrite(motor2, LOW);
  }
```

//temparature depended heater mechanism

```
  sensors.requestTemperatures();
  float tempC = sensors.getTempCByIndex(0) - 5;

  if (tempC < 25) {
    digitalWrite(TEMP_ALERT_PIN, HIGH);
  } else {
    digitalWrite(TEMP_ALERT_PIN, LOW);
  }
```

//OLED display output

```
  display.clearDisplay();
  display.setCursor(0, 0);
  display.println("pH Value:");
  display.setCursor(60, 0);
  display.println(ph_act);
  display.setCursor(0, 10);
  display.println("Water Level:");
  display.setCursor(70, 10);
  display.println(distance);
  display.setCursor(0, 20);
  display.println("Temperature:");
  display.setCursor(80, 20);
  display.println(tempC);
```

```
  display.display();

//for serial monitor
  Serial.print("pH Value: ");
  Serial.println(ph_act);
  Serial.print("Distance: ");
  Serial.println(distance);



  updateSerial();

  delay(1000);

 //feeder mechanism
  myservo.write(45);  // move servo to 45 degrees
  delay(43200000);  // wait for 2 minutes

   myservo.write(0);  // move servo to 0 degrees (anticlockwise)
  delay(1000);  // wait for 1 second

}

void updateSerial() {
  cmd = "";
  while (mySerial.available()) {
    cmd += (char)mySerial.read();
    if (cmd != "") {
```

```
    cmd.trim();
   }
  }
  Serial.print(cmd);
  Serial.println("");
  if (cmd != "") {
   char message = cmd.charAt(cmd.length() - 1);
   Serial.println(message);
   if (message == 'm') {
    digitalWrite(motor1, HIGH);  // Set digital pin 6 to 5V
   }
  }
  delay(500);

  while (Serial.available()) {
   mySerial.write(Serial.read());
  }

  while (mySerial.available()) {
   Serial.write(mySerial.read());
  }
}
```

5.2 Hardware Implementation Overview:

5.2.1 Sensors Integration:

pH Sensor: Connected to Arduino's analog input pin using appropriate voltage dividers or amplifiers for accurate readings. The sensor outputs pH values proportional to the solution's acidity, which are then interpreted by the Arduino code.

Temperature Sensor: Interfaced with Arduino via digital or analog pins, transmitting temperature values in digital or analog format, which the Arduino code interprets to regulate the heater and display temperature data.

Water Level Indicator (SONAR): SONAR sensor connected to Arduino's digital pins for distance measurement. The sensor determines water levels through ultrasonic waves, transmitting data to Arduino for processing and water level control.

5.2.2 Actuators and Modules:

Heater: Directly connected to Arduino's digital or analog pins through appropriate relay or transistor modules for controlling the heating element. Arduino code governs the heater based on temperature sensor readings.

Servo Motor (Feeding Mechanism): Linked to Arduino's PWM pins, the servo motor's control wire is connected to generate precise movements for automated fish feeding. Arduino code activates the servo motor at scheduled intervals.

GSM Module: Interfaced with Arduino using serial communication (RX, TX pins) or software serial library for data transmission. The module enables communication for remote monitoring and alerts based on sensor readings.

OLED Display: Connected to Arduino's I2C pins for data transmission, displaying real-time sensor data. Arduino code formats and sends sensor readings to the display for user visualization.

5.2.3 Control and Interfacing:

Arduino Uno: Serves as the central processing unit, interfacing with all sensors, actuators, and modules. It executes the programmed code, interpreting sensor data and controlling actions based on predefined thresholds.

5.3 Connections with Arduino:

5.3.1 SONAR-Arduino

SONAR Pin	Arduino Pin
GND	GND
Echo	Digital - 6
Trigg	Digital - 7
Vcc	5V

5.3.2 pH-Arduino

Sensor Pin	Arduino Pin
V+	5V
G	GND
P0	Analog A0

5.3.3 OLED-Arduino

SOLED Pin	Arduino Pin
GND	GND
Vcc	5V
SCL	Analog A5
SDA	Analog A4

5.3.4 Temperature-Arduino

Sensor Wire	Arduino Pin
Yellow	Digital 2
Red	5V
Black	GND

5.3.5 GSM- Arduino

GSM Pin	Arduino Pin
Vcc	5V
GND	GND
Tx	Digital 9
Rx	Digital 10

5.3.6 Servo Motor- Arduino

Motor Pin	Arduino Pin
Signal	Digital 11
Vcc	5V
GND	GND

6. Evaluation of Automated Aquarium System:

6.1 Performance Analysis:

The performance analysis of our automated aquarium system reveals commendable reliability in maintaining the aquarium's environmental parameters. However, critical evaluation highlights occasional latency in sensor responsiveness, particularly observed during rapid environmental fluctuations. The system showcases robust stability in maintaining preset conditions; yet, minor inaccuracies in certain sensor readings demand fine-tuning or calibration. These findings underscore the necessity for meticulous calibration procedures and potential sensor upgrades to optimize system responsiveness and accuracy, thereby fortifying its reliability in dynamic aquatic settings.

6.2 Impact:

The implementation of our automated aquarium system has exhibited profound positive impacts on aquatic habitat maintenance. Notably, it has significantly reduced the manual workload associated with monitoring and adjusting aquarium parameters, freeing users from mundane tasks. The system's automated functionalities have led to a discernible improvement in water quality, promoting a healthier ecosystem conducive to thriving aquatic life. Its integration of remote monitoring via the GSM module has empowered users with timely alerts, enabling swift intervention to preserve the aquarium's stability. Overall, the system's impact has redefined aquarium management, fostering an environment where technology harmonizes with ecology for sustainable aquatic preservation.

6.3 User Experience:

The user experience with our automated aquarium system reflects a blend of convenience and empowerment. Users appreciate the system's intuitive OLED display, providing real-time insights into crucial parameters. However, while the GSM module's remote accessibility has been lauded, the interface's simplicity warrants enhancement for seamless user interaction. Feedback regarding automated feeding via the servo motor has been overwhelmingly positive, streamlining feeding schedules and promoting consistent nourishment. However, there's a growing desire for an expanded user interface allowing greater customization and control. Despite minor areas for improvement, the overall user experience portrays a promising intersection of technological sophistication and user-friendly functionality, promising a brighter future for aquarium enthusiasts.

7. Future Enhancements:

7.1 Integration of Oxygen Sensor:

Addition of an oxygen sensor for real-time monitoring of dissolved oxygen levels. Enables precise adjustments to maintain optimal oxygen levels crucial for aquatic life. Strengthens the system's capability to sustain a healthy aquatic environment.

7.2 Implementation of AI Technology:

Incorporation of AI for predictive analysis and adaptive control.

AI-driven learning from data patterns to predict changes in aquarium parameters.

Facilitates preemptive adjustments, optimizing the environment proactively.

Evolves the system into an intelligent, self-regulating platform for enhanced management strategies and a resilient aquatic ecosystem.

8. Challenges:

Delay while Sync:

Synchronization delays between sensor readings and actuator responses caused intermittent disruptions in maintaining optimal conditions, demanding fine-tuning for real-time responsiveness.

GSM Module Code/Function:

Challenges in GSM module coding involved ensuring seamless data transmission and reception, requiring meticulous code development and testing for reliable remote communication.

Traditional Water Level Sensor Issues:

Initial setbacks with traditional water level sensors prompted the adoption of SONAR technology for accurate and consistent water level measurements, necessitating an abrupt shift to a more reliable alternative.

Code Optimization and Performance:

Sluggish code execution led to system inefficiencies; therefore, dedicated efforts were invested in optimizing the code for faster and more streamlined performance.

Code Syntax Errors:

Frequent code syntax errors posed challenges during development, demanding meticulous debugging and thorough syntax validation to ensure smooth code execution.

Code Debugging and Module Integration:

Integrating diverse modules demanded rigorous debugging efforts to harmonize their functionalities, ensuring seamless interaction and coordinated operation within the system.

9. Estimated cost:

Item	Cost
Push Button	20 BDT
Arduino Uno R3	1100 BDT
Vero Board	30 BDT

Resistor	10 BDT
Bread Board	160 BDT
OLED Display	700 BDT
Jumper Wire	100 BDT
GSM Module	2500 BDT
Servo Motor	1500 BDT
Pump	450 BDT
Heater	500 BDT
Temperature Sensor	800 BDT
pH Sensor	2500 BDT
Total	**10,370 BDT**

10. Conclusion:

As we come to the grand finale of our aquarium project, it's been an exhilarating journey merging cutting-edge technology with the marvels of nature. Picture this: sensors and Arduinos teaming up underwater, orchestrating a synchronized ballet of data and commands. We've essentially built a high-tech underwater hub where these gadgets party together, creating an innovative space for our aquatic buddies. Think of fish tanks not merely as glass boxes, but as intricate ecosystems designed for the ultimate comfort and joy of our aquatic pals. So, let's keep riding this wave of tech and nature fusion, where innovation and nature's brilliance collide to craft the most extraordinary underwater realms!

END