**Título do Projeto**

**AquaSmart: Sistema IoT para Monitoramento Inteligente da Qualidade da Água na Piscicultura**

**Descrição e Objetivos**

O projeto **AquaSmart** tem como objetivo aplicar técnicas de Agricultura de Precisão na área de **Piscicultura**, utilizando sensores e microcontroladores para realizar o monitoramento automatizado dos principais parâmetros da água nos tanques de criação de peixes. Com isso, busca-se garantir o bem-estar animal, otimizar o uso de recursos e aumentar a produtividade da criação.

**Qual problema será resolvido?**

A piscicultura é uma atividade sensível às variações ambientais. Fatores como temperatura inadequada, baixa oxigenação ou pH fora do ideal podem comprometer a saúde dos peixes, acarretando perdas econômicas significativas. O sistema AquaSmart resolve esse problema por meio da coleta e análise automática de dados da água, permitindo ações corretivas em tempo real.

**Motivação para essa escolha**

Com o crescimento da produção aquícola no Brasil, especialmente em sistemas de cultivo intensivo, torna-se fundamental implementar tecnologias de monitoramento que permitam maior controle e sustentabilidade da produção. A automação com IoT é uma solução moderna, acessível e altamente eficaz nesse contexto.

**Impacto Esperado**

* Redução de perdas por mortalidade de peixes.
* Melhoria da qualidade da água e do manejo do tanque.
* Otimização no uso de recursos (água, energia, oxigenação).
* Aumento da produtividade e sustentabilidade ambiental.
* Apoio à tomada de decisão com base em dados históricos.

**Participantes do Projeto**

* Rafael
* Carlos E. Mohr Barreto
* Magno
* Carlos E. Peretiatko

**Tema**

**Piscicultura** – Implementação de um sistema automatizado para monitoramento da qualidade da água em tanques de criação de peixes, com foco em precisão, eficiência e bem-estar animal.

**Componentes Eletrônicos Utilizados**

| **Componente** | **Função** |
| --- | --- |
| **Placa ESP32** | Microcontrolador que coleta dados dos sensores e envia para a nuvem. |
| **Sensor de Temperatura DS18B20** | Mede a temperatura da água. |
| **Sensor de pH 4502C** | Mede o pH da água para controle químico. |
| **Sensor de Turbidez ST100** | Detecta partículas em suspensão na água. |
| **Sensor de Oxigenação SEN0237** | Mede o oxigênio dissolvido, essencial para os peixes. |
| **Display LCD 16x2(I2C Oled)** | Apresenta os dados para o usuário |
| **Led** | Apresenta ao usuário o alerta |
| **Jumpers, resistores, protoboard** | Para montagem do circuito. |

**Funcionalidades do Sistema**

* Aquisição automática de dados dos sensores.
* Envio de dados para plataforma central (local ou em nuvem).
* Emissão de alertas quando os parâmetros estiverem fora do ideal.
* Armazenamento de histórico para análises e relatórios.
* Possibilidade de integração com dispositivos atuadores (bombas, aeradores, etc.).

**Como Funciona**

1. Os sensores são conectados ao **ESP32**.
2. A cada intervalo de tempo, o ESP32 realiza a leitura dos dados de temperatura, pH, turbidez e oxigênio.
3. Os dados são transmitidos para um sistema de monitoramento (ex.: Blynk, MQTT ou dashboard web).
4. Caso algum parâmetro ultrapasse limites pré-estabelecidos, o sistema gera **alertas visuais, sonoros ou digitais**.
5. O histórico é armazenado para fins de **gestão e tomada de decisão** no manejo.

**Código da Aplicação**

*(Este trecho pode variar dependendo da biblioteca e plataforma utilizada, mas segue um exemplo ilustrativo para ESP32 com sensores fictícios):*

#include <OneWire.h>

#include <DallasTemperature.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

// Definição dos pinos

#define ONE\_WIRE\_BUS 4       // DS18B20 (GPIO4)

#define PH\_PIN 32            // Sensor de pH simulado

#define TURBIDEZ\_PIN 33      // Sensor de turbidez simulado

#define OXIGENIO\_PIN 34      // Sensor de oxigênio simulado (entrada somente)

#define ALERTA\_LED 2         // LED de alerta (GPIO2)

// Definições do display OLED

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

#define OLED\_RESET    -1

OneWire oneWire(ONE\_WIRE\_BUS);

DallasTemperature sensors(&oneWire);

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

// Variável para indicar se o display foi iniciado com sucesso

bool displayOk = false;

void setup() {

  Serial.begin(115200);

  pinMode(ALERTA\_LED, OUTPUT);

  sensors.begin();

  // Inicializa o display OLED

  if (!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

    Serial.println("Falha ao inicializar SSD1306");

    displayOk = false;

  } else {

    displayOk = true;

    display.clearDisplay();

    display.setTextSize(1);

    display.setTextColor(SSD1306\_WHITE);

    display.setCursor(0, 0);

    display.println("Piscicultura de Precisao");

    display.display();

    delay(1500);

  }

  Serial.println("=== Piscicultura de Precisão ===");

  Serial.println("Integrantes: Rafael, Carlos E. Mohr Barreto, Magno, Carlos E. Peretiatko");

  Serial.println("================================");

}

void loop() {

  sensors.requestTemperatures();

  float tempC = sensors.getTempCByIndex(0);

  int leituraPh = analogRead(PH\_PIN);

  int leituraTurbidez = analogRead(TURBIDEZ\_PIN);

  int leituraOxigenio = analogRead(OXIGENIO\_PIN);

  float ph = map(leituraPh, 0, 4095, 0, 14);

  bool alerta = (tempC < 22 || tempC > 30 || ph < 6.5 || ph > 8.5);

  digitalWrite(ALERTA\_LED, alerta ? HIGH : LOW);

  // Exibição no Serial Monitor

Serial.print("Temperatura: "); Serial.print(tempC); Serial.print(" °C | ");

Serial.print("pH: "); Serial.print(ph); Serial.print(" | ");

Serial.print("Turbidez: "); Serial.print(leituraTurbidez); Serial.print(" | ");

Serial.print("Oxigênio: "); Serial.println(leituraOxigenio);

}

*(As funções lerPH(), lerTurbidez() e lerOxigenio() devem ser definidas de acordo com os sensores e circuitos reais usados.)*

**Diagrama do Circuito**

*Inserir aqui o diagrama de montagem (gerado no Tinkercad ou Wokwi). Caso queira, posso te ajudar a montar esse circuito virtualmente.*

**Print ou Link da Simulação**

**Link da simulação:** [Inserir link Wokwi ou Tinkercad]  
*(Se quiser, posso criar a simulação no Wokwi e te enviar o link.)*

**Justificativa da Necessidade da Solução**

O controle manual da qualidade da água em sistemas de piscicultura é ineficiente e propenso a falhas. O uso de IoT permite automação, precisão e resposta rápida, reduzindo riscos e custos operacionais. Essa solução promove bem-estar animal, sustentabilidade ambiental e crescimento econômico para pequenos e médios produtores.

**Explicação da Utilização dos Dados Coletados**

* **Temperatura:** Garante que a água esteja em uma faixa segura para a espécie cultivada.
* **pH:** Mantém o ambiente químico adequado, evitando estresse e doenças.
* **Turbidez:** Identifica excesso de partículas ou resíduos no viveiro.
* **Oxigênio dissolvido:** Evita mortalidade por hipóxia.

Esses dados permitem ações preventivas, além de oferecer uma base para decisões futuras, ajustes no manejo e planejamento estratégico.

**Licença**

Este projeto está disponível sob a licença **MIT**.  
Sinta-se livre para colaborar, adaptar ou melhorar!

**Contato com os Participantes**

Para dúvidas, sugestões ou colaborações, entre em contato com:

* Rafael
* Carlos E. Mohr Barreto
* Magno
* Carlos E. Peretiatko