**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS**
**ESCOLA POLITÉCNICA E DE ARTES / ENGENHARIA DE CONTROLE E AUTOMAÇÃO**
**Projeto Final de Curso**

**Ulisses Xavier Castilho**

# MONITORAMENTO DE QUALIDADE DA ÁGUA PARA BOVINOCULTURA

Projeto Final de Curso como parte dos requisitos para obtenção do título de bacharel em Engenharia de Controle e Automação apresentado à Pontifícia Universidade Católica de Goiás.

BANCA EXAMINADORA:

Prof. Dr. Antônio Marcos de Melo Medeiros – Orientador. PUC Goiás.
Prof(a). Mestre(a). Carlos Bezerra
Prof(a). Mestre(a). Pagotti

Goiânia, 15 de Junho de 2026.

# MONITORAMENTO DE QUALIDADE DA ÁGUA PARA BOVINOCULTURA UTILIZANDO IoT DE BAIXO CUSTO

*Castilho U. X., Medeiros A. M. M., Bezerra C., Pagotti*

*Escola Politécnica e de Artes, PUC-GOIÁS.*

**Abstract – The objective of this work was to develop a low-cost IoT prototype for water quality monitoring in cattle watering troughs. The proposed system employs an ESP32 microcontroller powered by solar energy, integrating sensors for turbidity, TDS, temperature, water level, and flow rate. Data are transmitted via WiFi and processed in a Node-RED supervisory interface, enabling real-time visualization and decision support for rural producers. Bench tests conducted over 24 hours demonstrated the system's effectiveness in capturing oscillations in water quality parameters, supporting the technical and economic feasibility of the solution for properties of varying sizes.**

**Keywords – IoT, water quality, cattle, ESP32, Node-RED, low cost, solar energy.**

**Resumo – O objetivo deste trabalho foi desenvolver um protótipo de sistema IoT de baixo custo para monitoramento da qualidade da água em bebedouros de bovinos. O sistema proposto emprega um microcontrolador ESP32 alimentado por energia solar, integrando sensores de turbidez, TDS, temperatura, nível e vazão. Os dados são transmitidos via WiFi e processados em uma interface supervisória Node-RED, possibilitando visualização em tempo real e suporte à tomada de decisão do produtor rural. Testes em bancada conduzidos por 24 horas demonstraram a eficácia do sistema na captura de oscilações nos parâmetros de qualidade da água, comprovando a viabilidade técnica e econômica da solução para propriedades de diferentes portes.**

**Palavras-Chave – IoT, qualidade da água, bovinocultura, ESP32, Node-RED, baixo custo, energia solar.**

## I. INTRODUÇÃO

A qualidade da água é um aspecto fundamental para garantir a saúde, a produtividade e o bem-estar dos animais em fazendas. A água é essencial para diversas funções fisiológicas, desde a hidratação até a manutenção de processos metabólicos vitais. No entanto, a contaminação da água por agentes patogênicos ou poluentes pode representar sérios riscos à saúde dos animais, comprometendo sua produção, reprodução e, em casos extremos, resultando em perdas significativas para os criadores.

Além disso, os bovinos não possuem reservas de água em seus corpos. Quando há escassez desse recurso, seus organismos tendem a buscar água nos órgãos para suprir a demanda, resultando em perda imediata de peso, especialmente nos músculos. Esse processo também pode causar sintomas como redução no crescimento e aumento do estresse nos animais. É relevante notar que o organismo dos bovinos é composto por uma proporção significativa de água, que pode variar entre 50% e 80% [1].

A automação do monitoramento da qualidade da água para animais tornou-se extremamente viável devido à acessibilidade dos sensores e à sua fácil integração. Com sensores cada vez mais avançados, os produtores agora têm a capacidade de monitorar continuamente, em tempo real, a qualidade da água, garantindo um ambiente saudável e seguro para seus animais [2].

## II. OBJETIVOS

### *A. Objetivo Geral*

Construir um protótipo de sistema IoT de baixo custo que utilize energia solar, sensores e comunicação WiFi para enviar dados advindos do bebedouro, tendo a capacidade de diagnosticar para o pecuarista o momento exato de realizar a troca de água e manutenção no bebedouro, garantindo maior otimização do tempo e manejo adequado da água.

### *B. Objetivos Específicos*

- Modelar uma placa de circuito para implementação de hardware robusto e otimizado;
- Implementar o hardware no bebedouro de forma segura e sem impactos ambientais;
- Coletar dados dos bebedouros;
- Transmitir os dados coletados de forma confiável para a sede da fazenda;
- Tratar os dados e gerar uma interface gráfica intuitiva para o colaborador, permitindo a otimização na tomada de decisão.

## III. FUNDAMENTAÇÃO TEÓRICA

### BOVINOCULTURA E REQUISITOS HÍDRICOS

O Brasil possui um dos maiores rebanhos bovinos do mundo, com mais de 210 milhões de cabeças. A água é o nutriente mais importante para o bovino, compondo entre 50% e 80% da massa corporal e participando de todas as funções fisiológicas essenciais — digestão, termorregulação, transporte de nutrientes e eliminação de metabólitos [1].

O consumo diário varia de 26 a 66 litros em condições normais, podendo ultrapassar 90 litros em clima quente. A privação hídrica por mais de 24 horas causa queda imediata no consumo de matéria seca, redução do ganho de peso e, em casos extremos, mortalidade. Estudos demonstram que água de qualidade inadequada pode reduzir o ganho de peso em até 20% [2]. O acesso contínuo a água de qualidade é reconhecido pela OIE (Organização Mundial de Saúde Animal) como requisito básico de bem-estar animal.

### PARÂMETROS DE QUALIDADE DA ÁGUA

#### *A. Turbidez*

A turbidez mede a resistência da água à passagem de luz causada por partículas em suspensão, expressa em NTU (Unidades Nefelométricas). Para bovinos recomenda-se turbidez inferior a 100 NTU; acima desse valor a palatabilidade e o consumo voluntário reduzem-se significativamente. Alta turbidez também indica risco microbiológico, pois as partículas protegem bactérias patogênicas contra desinfecção [3].

#### *B. Sólidos Totais Dissolvidos (TDS)*

O TDS representa a concentração total de sais iônicos dissolvidos (sódio, cálcio, magnésio, sulfatos, cloretos), expresso em mg/L. O NRC recomenda limite de 3.000 mg/L para bovinos de corte, com ideal abaixo de 1.000 mg/L. Concentrações elevadas de sulfatos causam diarreia osmótica e

desequilíbrio eletrolítico. O sensor TDS Meter V1.0 estima essa grandeza pela condutividade elétrica da solução [4].

*C. Temperatura, Nível e Vazão*

A temperatura ideal da água para bovinos está entre 10 °C e 20 °C; acima de 30 °C o consumo cai até 40%, favorecendo ainda a proliferação de algas e bactérias. O monitoramento do nível do reservatório permite detectar falhas de abastecimento — bomba defeituosa, válvula entupida — antes que o bebedouro esvazie. A vazão revela o padrão de consumo do rebanho ao longo do dia, possibilitando identificar anomalias como consumo excessivo (febre/diarreia) ou ausência de consumo (bloqueio/estresse) [5].

## INTERNET DAS COISAS E ESP32

A Internet das Coisas (IoT) é o paradigma pelo qual objetos físicos equipados com sensores, processadores e interfaces de comunicação interagem de forma autônoma com sistemas digitais. Sua arquitetura organiza-se em três camadas: percepção (sensores/atuadores), rede (protocolos de comunicação) e aplicação (processamento e visualização de dados) [6].

O ESP32, desenvolvido pela Espressif Systems, é um SoC (System on Chip) que integra processador dual-core Xtensa LX6 a 240 MHz, WiFi 802.11 b/g/n, Bluetooth 4.2/BLE, ADC de 12 bits e interfaces UART, SPI e I²C — tudo em um único componente de baixo custo. A compatibilidade com o framework Arduino e o modo deep sleep, que reduz o consumo a menos de 10 µA, tornam o ESP32 ideal para nós IoT alimentados por energia solar [7].

## ENERGIA SOLAR EM SISTEMAS EMBARCADOS

O Brasil recebe irradiação solar global entre 4,5 e 6,5 kWh/m²/dia; Goiás apresenta média de aproximadamente 5,5 kWh/m²/dia, favorável ao aproveitamento fotovoltaico [8]. Para sistemas IoT de campo, painéis de 5 a 20 W são suficientes para recarregar diariamente uma bateria chumbo-ácida de 12 V que garante autonomia nos períodos sem geração — noite e dias nublados.

O regulador buck LM2596 converte a tensão variável do painel (12–18 V) para os níveis estáveis exigidos pelo circuito (3,3 V / 5 V) com eficiência superior a 92%, minimizando dissipação de calor. O diodo Schottky entre painel e bateria impede a descarga reversa durante a noite, preservando a vida útil da bateria [9].

## HARDWARE EMBARCADO E PCB

A Placa de Circuito Impresso (PCB) é o substrato dielétrico — tipicamente FR4 (fibra de vidro/epóxi) — sobre o qual trilhas de cobre interligam os componentes eletrônicos. Em relação à fiação ponto a ponto, a PCB reduz dimensões, melhora a confiabilidade elétrica e facilita a replicação do protótipo [10].

Em sistemas IoT de campo, o projeto deve prever robustez contra umidade, temperatura e vibração. Capacitores de desacoplamento próximos a cada circuito integrado reduzem ruído eletromagnético; bornes de parafuso facilitam a substituição de sensores em campo sem necessidade de solda. O processo de fabricação artesanal por transferência de toner e corrosão em $FeCl_3$ é adequado para prototipagem acadêmica com trilhas ≥ 0,5 mm [10].

## NODE-RED, HTTP E JSON

Node-RED é uma plataforma de programação visual baseada em fluxo, construída sobre Node.js e mantida pela OpenJS Foundation. Nós funcionais são conectados graficamente para definir fluxos de aquisição, processamento e visualização de dados, dispensando o desenvolvimento de front-end convencional. O módulo node-red-dashboard gera painéis interativos diretamente no ambiente, com gauges, gráficos e caixas de texto configuráveis [11].

O protocolo HTTP (RFC 2616) opera no modelo cliente-servidor sem estado: o Node-RED (cliente) envia requisições GET periodicamente ao ESP32 (servidor embarcado), que responde com os dados dos sensores serializados em JSON. O JSON (RFC 8259) é o formato padrão para intercâmbio de dados em IoT por combinar leveza, legibilidade humana e suporte nativo em todas as linguagens modernas. Um único objeto JSON por requisição concentra todas as variáveis, reduzindo latência e o número de conexões TCP [12].

O cliente NTP (Network Time Protocol) implementado no ESP32 sincroniza o relógio interno com servidores de referência na internet, inserindo timestamp preciso em cada registro de leitura. Isso viabiliza a geração de relatórios históricos ordenados cronologicamente, a correlação temporal entre eventos e a exportação de dados para análise em planilhas [13].

## IV. MATERIAIS E MÉTODOS

Foi realizada uma lista de materiais necessários para conduzir testes em bancada e verificar o funcionamento do protótipo.

*A. Materiais Gerais*

- Painel solar;
- Bateria de 12 V;
- ESP-32;
- 2 sensores de temperatura DS18B20;
- Sensor de turbidez ST100;
- Sensor de TDS V1.0 Meter;
- Sensor de vazão YF-S201;
- Sensor de nível;
- Software Node-RED.

*B. Materiais para a Placa de Circuito*

- 1 capacitor eletrolítico de 100 µF / 63 V;
- 1 capacitor eletrolítico de 1 µF / 63 V;
- 1 capacitor cerâmico de 220 nF;
- 2 bornes de 2 vias; 5 bornes de 3 vias;
- 1 módulo regulador de tensão LM2596;
- 1 diodo Schottky;
- 2 resistores de 1 kΩ; 2 resistores de 10 kΩ;
- 1 resistor de 330 Ω e 1 de 150 Ω;
- 2 LEDs 5 mm;
- Conector barra 12 mm para circuito impresso 1×40.

A Figura 1 ilustra o fluxo de sinal do sistema, destacando a captação e o processamento de dados dos sensores. A energia é captada por um painel solar e regulada por um regulador de tensão, sendo armazenada em uma bateria de 12 V que alimenta o microcontrolador ESP-32.

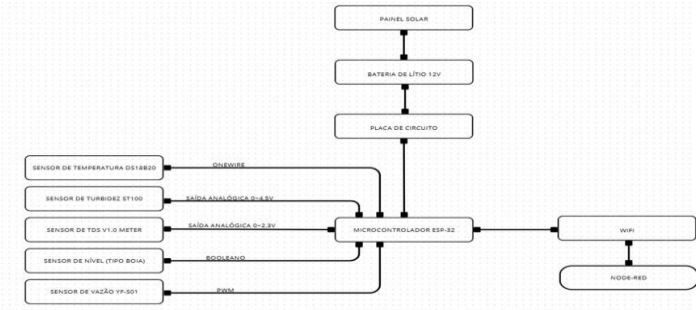


*Figura 1 – Diagrama de blocos do sistema.*

## V. DESENVOLVIMENTO DE HARDWARE

Para ter um sistema mais organizado e efetivo foi desenvolvida uma placa de circuito impressa, onde os sensores estão acoplados para enviar sinais até o microcontrolador. A Figura 2 mostra o layout das trilhas de cobre definido no software de roteamento, e a Figura 3 apresenta o projeto final da placa, já com a disposição dos componentes (regulador buck LM2596, bornes de conexão e microcontrolador). A Figura 4 ilustra a transparência utilizada no processo de transferência de toner para a corrosão da placa, e a Figura 5 mostra a placa de circuito já montada e energizada, com os LEDs indicadores acesos.

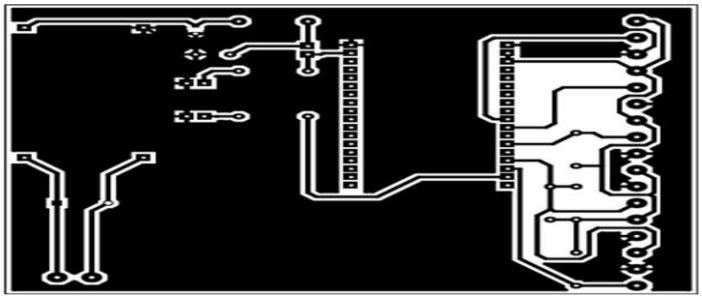

*Figura 2 – Layout das trilhas da placa de circuito.*

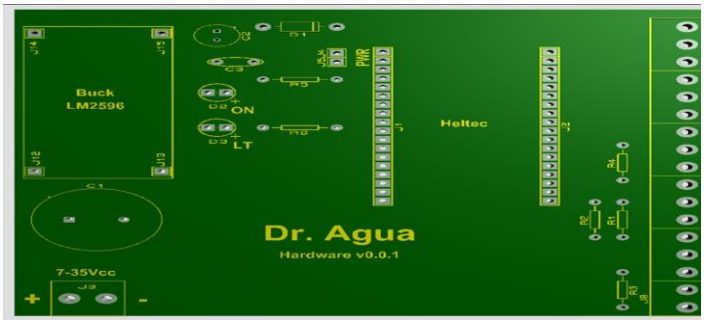


*Figura 3 – Projeto da Placa de Circuito Impressa.*

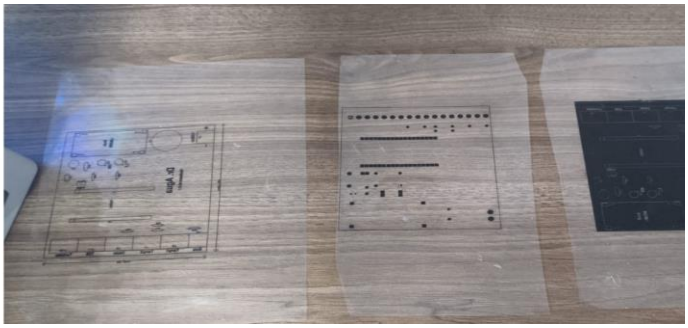

*Figura 4 – Transparência para confecção do PCB.*

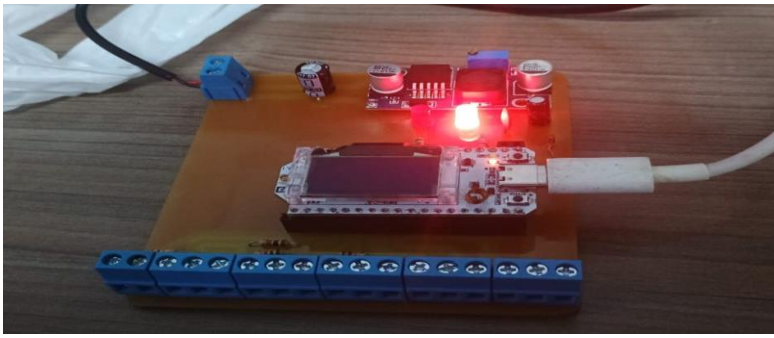

*Figura 5 – Placa de Circuito montada.*

## VI. DESENVOLVIMENTO DE SOFTWARE

Foi desenvolvido um sistema supervisório utilizando JavaScript na plataforma Node-RED, englobando o desenvolvimento do front-end e do back-end. O sistema tem como principal objetivo a análise de dados e a interpretação gráfica das informações coletadas, utilizando o protótipo ESP32 como servidor para requisições via protocolo HTTP.

### *A. WebServer*

Um WebServer é um software que processa requisições HTTP enviadas por clientes, como navegadores ou outros dispositivos conectados à rede. No contexto da ESP32, um WebServer permite que a placa atenda requisições para obter ou modificar estados de sensores e atuadores. A Figura 6 ilustra esse fluxo básico de comunicação, com o estabelecimento da conexão TCP e a troca de requisição (HTTP request) e resposta (HTTP response) entre cliente e servidor.

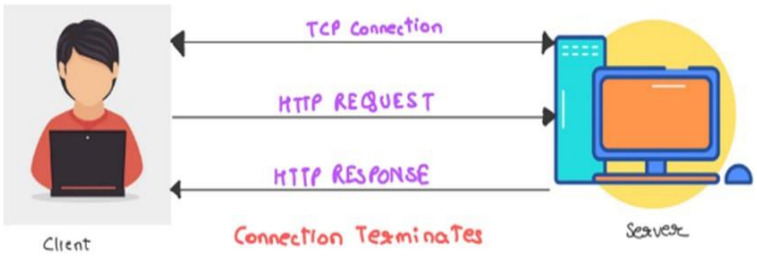


*Figura 6 – Arquitetura básica de um WebServer.*

### *B. Cliente e Servidor: ESP32 e Node-RED*

Na comunicação via WiFi entre a ESP32 e o Node-RED, a ESP32 atua como WebServer, processando requisições HTTP enviadas pelo Node-RED; enquanto o Node-RED atua como cliente HTTP, enviando requisições para a ESP32 para obter informações dos sensores ou enviar comandos de controle.

### *C. Métodos HTTP*

O protocolo HTTP suporta diversos métodos para a comunicação entre cliente e servidor. No contexto desta aplicação foi adotado o método GET, que melhor se encaixa nas diretrizes do projeto, utilizado para solicitar os dados dos sensores sem alteração de estado no servidor.

### *D. Arquivo JSON*

Nos sistemas IoT, o formato JSON (JavaScript Object Notation) tornou-se padrão para a troca de dados estruturados. JSON é leve, de fácil interpretação por máquinas e humanos, e suporta objetos, arrays e tipos primitivos. Sua utilização neste projeto permitiu transmitir todos os parâmetros dos sensores em uma única requisição HTTP, reduzindo latência e facilitando a expansão do sistema.

### *E. Back-end no Node-RED*

Para o desenvolvimento do back-end na plataforma Node-RED foram utilizados blocos de carimbo de data/hora, bloco de requisição HTTP, bloco de organização do arquivo JSON e

blocos de interface gráfica com caixas de texto e gauges para os sensores analógicos. A Figura 7 apresenta o fluxo completo montado no editor do Node-RED, encadeando os blocos de requisição, tratamento e distribuição dos dados para os elementos do dashboard. A Figura 8 detalha a configuração do bloco de requisição HTTP, com o método GET e a URL do servidor embarcado no ESP32. A Figura 9 mostra a configuração de um bloco do tipo "Gauge" para exibição do valor de TDS, a Figura 10 apresenta a configuração do bloco de requisição associado ao sensor de nível, e a Figura 11 ilustra a configuração da caixa de texto responsável por exibir a data e a hora da última leitura.

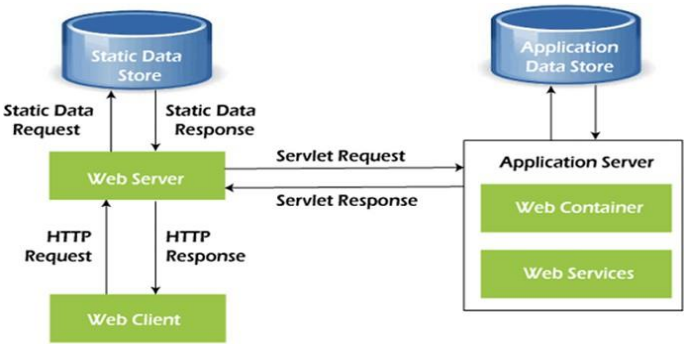


*Figura 7 – Programação em blocos com JavaScript.*

| Código | Significado |
|---|---|
| 200 OK | Requisição bem-sucedida |
| 400 Bad Request | Erro na requisição do cliente |
| 404 Not Found | Recurso não encontrado |
| 500 Internal Server Error | Erro no servidor |

*Figura 8 – Configurando o método GET e a URL.*

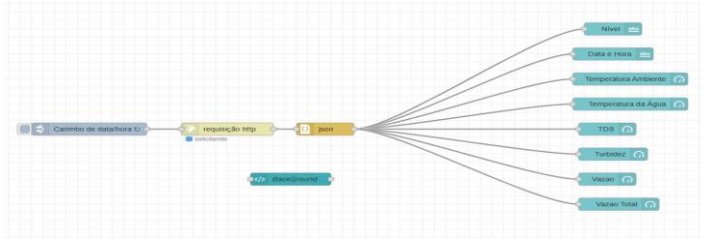

*Figura 9 – Configurando o bloco "Gauge" do sensor de TDS.*

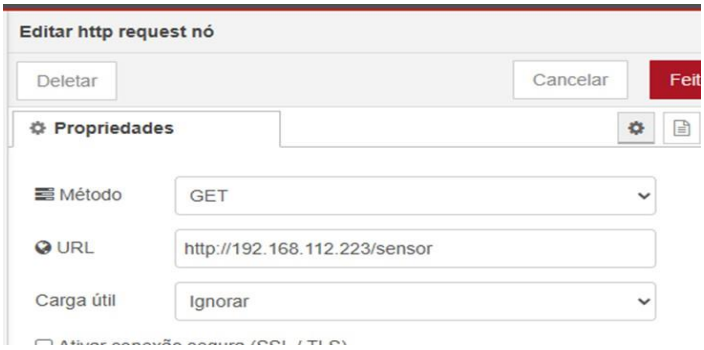


*Figura 10 – Configuração do bloco para leitura do sensor de nível.*

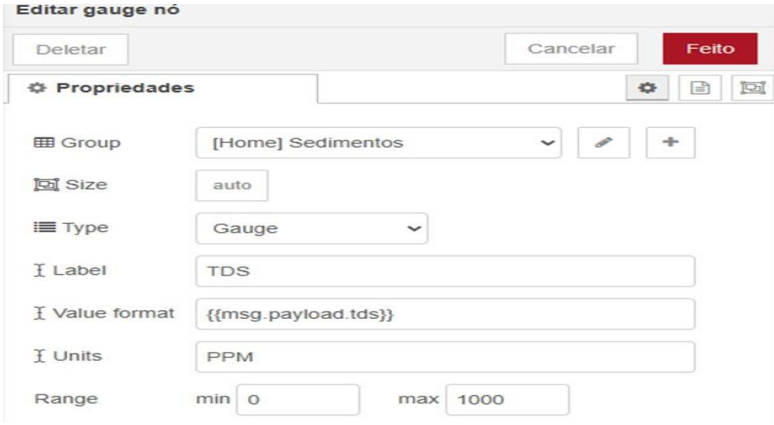


*Figura 11 – Configuração da caixa de texto para data e hora.*

### *F. Dashboard Final – Front-End*

Considerando as configurações aplicadas nos blocos e a comunicação eficiente entre cliente e servidor, o dashboard final apresentou os dados de todos os sensores de forma organizada e em tempo real, conforme ilustrado na Figura 12. Observa-se, no entanto, que o gauge de turbidez exibiu uma leitura negativa durante o teste registrado, resultado da curva de calibração polinomial utilizada na conversão da tensão analógica do sensor ST100 (detalhada na Figura 15), que não restringe a faixa de saída a valores fisicamente válidos (NTU ≥ 0). Esse comportamento é discutido como limitação do protótipo na seção de Discussão.

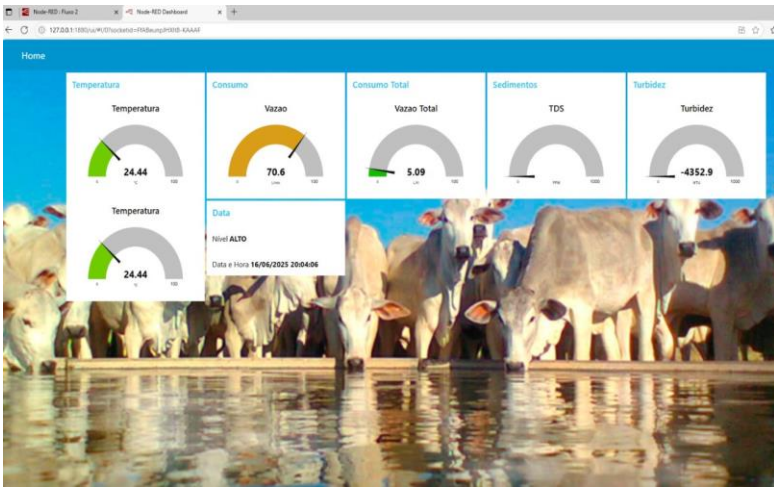


*Figura 12 – Dashboard Final.*

### *G. Métodos de Implementação do Firmware*

Para a fundamentação do desenvolvimento do software, os sensores foram implementados individualmente, seguidos por testes de comunicação com o sistema supervisório. Bibliotecas específicas foram empregadas para cada sensor, resultando em estrutura modular e concisa. A Figura 13 apresenta as bibliotecas incluídas no firmware e a inicialização do objeto WebServer.

```
#include <Arduino.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <time.h>
#include <WebServer.h>

WebServer servidor(80);

// WiFi
const char* ssid = "Gabriel";
const char* password = "12345678";
```

*Figura 13 – Bibliotecas e framework Arduino incluídos no firmware.*

Após a leitura dos sensores, os valores são organizados em uma única estrutura JSON antes do envio ao Node-RED. A Figura 14 mostra a montagem dessa string, concatenando data/hora, turbidez, TDS, temperatura, vazão, volume acumulado e o estado do sensor de nível.

```
// JSON
String json = "{";
json += "\"dataHora\":\"" + String(dataHora) + "\",";
json += "\"turbidez\":" + String(Turbidez, 2) + ",";
json += "\"tds\":" + String(TDS, 2) + ",";
json += "\"Temperatura\":" + String(temp, 2) + ",";
json += "\"vazao\":" + String(CalculoVazao, 2) + ",";
json += "\"acumulado\":" + String(VazaoAcumulada, 2) + ",";
json += "\"nivel\":\"" + String(lerNivel() ? "BAIXO" : "ALTO") + "\"";
json += "}";
return json;
}
```

*Figura 14 – Estruturação dos dados coletados no arquivo JSON.*

*Função enviarParaNodeRED*

Esta função transmite os dados coletados pelo ESP32 para o Node-RED por meio do protocolo HTTP (método GET), inicializando uma conexão com o servidor, definindo o cabeçalho da requisição como application/json e enviando o pacote de dados. O código de status HTTP retornado é registrado para monitoramento da comunicação.

*Função responderRequisicaoLocal()*

Esta função fornece acesso local aos dados coletados pelo ESP32 sem encaminhamento ao Node-RED, organizando os valores dos sensores em um objeto JSON e enviando-os com código de status 200 (OK) em resposta às requisições diretas ao servidor web embutido. As Figuras 16 e 17 apresentam, respectivamente, a função setup(), responsável pela inicialização dos pinos, da conexão WiFi e do servidor HTTP, e a função loop(), que controla o intervalo de envio periódico dos dados ao Node-RED.

```
String coletarJSON() {
  // Hora
  struct tm timeinfo;
  char dataHora[25] = "00/00/0000 00:00:00";
  if (getLocalTime(&timeinfo)) {
    strftime(dataHora, sizeof(dataHora), "%d/%m/%Y %H:%M:%S", &timeinfo);
  }

  // Turbidez
  TensaoTurbidez = analogRead(ST100) / 4095.0 * 3.3;
  Turbidez = -1120.4 * TensaoTurbidez * TensaoTurbidez + 5742.3 * TensaoTurbidez - 4352.9;

  // TDS
  TensaoTDS = analogRead(Condutividade) / 4095.0 * 3.3;
  TDS = (133.42 * pow(TensaoTDS, 3)) - (255.86 * pow(TensaoTDS, 2)) + (857.39 * TensaoTDS);
  TDS *= 0.5;

  // Temperatura
  sensortemp.requestTemperatures();
  temp = sensortemp.getTempCByIndex(0);

  // Vazão (não bloqueante)
  unsigned long inicio = millis();
  Contador = 0;
  while (millis() - inicio < 1000) {
    yield();  // evita WDT
  }
  int pulsos = Contador;

  CalculoVazao = pulsos * 2.25;
  VazaoAcumulada += (CalculoVazao / 1000.0);
  CalculoVazao = (CalculoVazao * 60) / 1000.0;
```

*Figura 15 – Leitura e cálculo dos valores dos sensores na função coletarJSON().*

```
void setup() {
  Serial.begin(9600);

  pinMode(ST100, INPUT);
  pinMode(Condutividade, INPUT);
  pinMode(nivel, INPUT_PULLUP);
  pinMode(SensorVazao, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(SensorVazao), Vazao, RISING);

  WiFi.begin(ssid, password);
  Serial.print("Conectando ao WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi conectado.");
  Serial.print("IP local: ");
  Serial.println(WiFi.localIP());

  configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
  sensortemp.begin();

  servidor.on("/sensor", responderRequisicaoLocal);
  servidor.begin();
  Serial.println("Servidor HTTP iniciado.");
}
```

*Figura 16 – Função Setup.*

```
void loop() {
  servidor.handleClient();

  if (millis() - tempoAnterior >= intervaloEnvio) {
    tempoAnterior = millis();
    String json = coletarJSON();
    enviarParaNodeRED(json);
  }
}
```

*Figura 17 – Função loop.*

## VII. RESULTADOS

Foi conduzido um teste em ambiente controlado por um período de 24 horas e, com base nos dados coletados pelos sensores, elaborou-se uma planilha contendo gráficos para análise crítica das informações obtidas. A Tabela 1 apresenta os dados brutos coletados hora a hora, e o Gráfico 1 ilustra a evolução desses parâmetros ao longo do período observado.

| Dr. Água | | Dados Coletados | | | | |
|---|---|---|---|---|---|---|
| | | Sensores: | | | | |
| Dia: 04/03/25 | Hora do dia: | TDS (mg/L) | Turbidez (NTU) | Temperatura (C°) | Vazão (L/min) | Nível (Bool) |
| | 00:00 | 324 | 249 | 21 | 5 | 1 |
| | 01:00 | 320 | 242 | 22 | 8 | 1 |
| | 02:00 | 316 | 236 | 22 | 10 | 1 |
| | 03:00 | 313 | 230 | 23 | 12 | 1 |
| | 04:00 | 310 | 225 | 23 | 18 | 1 |
| | 05:00 | 308 | 220 | 24 | 20 | 1 |
| | 06:00 | 306 | 216 | 24 | 22 | 0 |
| | 07:00 | 304 | 212 | 25 | 26 | 1 |
| | 08:00 | 302 | 209 | 25 | 30 | 1 |
| | 09:00 | 301 | 206 | 26 | 30 | 1 |
| | 10:00 | 300 | 204 | 26 | 20 | 1 |
| | 11:00 | 300 | 202 | 27 | 15 | 1 |
| | 12:00 | 300 | 201 | 27 | 12 | 1 |
| | 13:00 | 300 | 200 | 28 | 10 | 1 |
| | 14:00 | 300 | 200 | 28 | 6 | 1 |
| | 15:00 | 301 | 200 | 28 | 4 | 1 |
| | 16:00 | 302 | 201 | 27 | 2 | 1 |
| | 17:00 | 304 | 202 | 27 | 2 | 1 |
| | 18:00 | 306 | 204 | 26 | 3 | 1 |
| | 19:00 | 308 | 206 | 26 | 0 | 1 |
| | 20:00 | 310 | 209 | 25 | 5 | 1 |
| | 21:00 | 313 | 212 | 25 | 6 | 1 |
| | 22:00 | 316 | 216 | 24 | 7 | 1 |
| | 23:00 | 320 | 220 | 24 | 8 | 1 |
| | Média Total: | 307,6666667 | 213,4166667 | 25,125 | 11,71 | 0,958333333 |

*Tabela 1 – Coleta de dados experimentais nas 24 horas.*

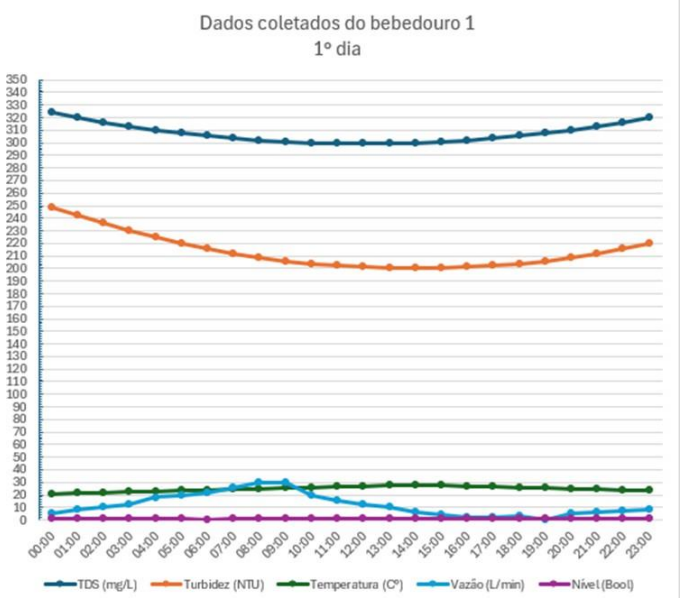


*Gráfico 1 – Medidas das variáveis ao longo das 24 horas.*

### *A. Análise Crítica dos Resultados*

Os dados coletados durante 24 horas evidenciam oscilações relevantes entre os parâmetros monitorados. Os sólidos totais dissolvidos (TDS) mantiveram-se relativamente estáveis, com média de 307,7 mg/L. A turbidez apresentou média de 213,4 NTU, com picos nas primeiras horas da madrugada (249 NTU às 00h00) e valores menores à tarde (200 NTU entre 13h00 e 15h00), indicando maior clareza nos horários de maior consumo.

A temperatura variou de 21 °C a 28 °C, acompanhando o ciclo térmico ambiental, com tendência de aquecimento entre 12h00 e 15h00. A vazão média foi de 11,7 L/min, com picos significativos às 09h00 (30 L/min) e às 07h00 (26 L/min). O nível do reservatório manteve-se estável na maior parte do tempo, com exceção de uma queda às 06h00, que pode indicar falha temporária no abastecimento [3].

## VIII. DISCUSSÃO

A implementação do protótipo IoT para monitoramento da qualidade da água em bebedouros de bovinos demonstrou resultados promissores em ambiente controlado. O sistema mostrou-se eficiente na captação, processamento e transmissão das informações dos principais parâmetros de qualidade da água – turbidez, TDS, temperatura, nível e vazão –, viabilizando acompanhamento em tempo real pelo produtor rural.

A utilização de sensores de baixo custo e energia solar demonstrou viabilidade técnica e econômica, tornando o sistema acessível para propriedades de diferentes portes. O uso do Node-RED facilitou a visualização dos resultados e a tomada de decisão, comprovando a importância de interfaces intuitivas para a adoção efetiva da tecnologia no campo.

A confiabilidade da transmissão de dados via WiFi mostrou-se adequada para ambientes com cobertura de rede local, mas pode demandar adaptações em fazendas com maior extensão ou áreas remotas. A modularidade do sistema, no entanto, permite ajustes na infraestrutura de comunicação conforme a necessidade de cada propriedade [4].

Identificou-se também uma limitação na curva de calibração do sensor de turbidez (ST100), cujo polinômio de conversão tensão-NTU permitiu, em determinadas condições de leitura, a geração de valores negativos sem significado físico, como evidenciado na Figura 12. Esse comportamento não compromete a arquitetura geral do sistema, mas indica a necessidade de recalibração do sensor com pontos de referência adicionais e de tratamento de saturação (clamping) no firmware antes de uma futura validação em campo.

## IX. CONCLUSÃO

Conclui-se, por meio deste trabalho, que foi possível integrar um hardware viável e autossuficiente a um software de supervisão e a uma planilha para análise crítica de dados, proporcionando uma solução relevante tanto para os animais quanto para o produtor rural. A disponibilidade de um sistema de coleta de dados é fundamental para que o produtor compreenda a influência da qualidade da água na produtividade e no bem-estar animal.

Para trabalhos futuros, vislumbra-se a validação do sistema em campo com rebanhos em diferentes condições ambientais, a integração com comunicação LPWAN para áreas remotas e o desenvolvimento de alertas automáticos baseados em limiares definidos pelos parâmetros normativos de qualidade da água para bovinos.

## X. REFERÊNCIAS

**[1]** BOI SAÚDE. A importância da água para bovinos. 2020. Disponível em: <https://boisaude.com.br>. Acesso em: 30 mai. 2026.

**[2]** IMPACTOS DA QUALIDADE DA ÁGUA NA PRODUÇÃO DE BOVINOS DE CORTE CRIADOS A PASTO. 2021.

**[3]** MINHO, A. P.; GASPAR, E. B. Água na pecuária: requerimento animal e gerenciamento das fontes. In: SILVEIRA, M. C. T.; TRENTIN, G. (org.). Manejo da água na pecuária. Brasília: Embrapa, 2023. p. 57–74.

**[4]** GROUT, A. S. et al. Differential effects of sodium and magnesium sulfate on water consumption by beef cattle. Journal of Animal Science, v. 84, n. 5, p. 1252–1258, May 2006.

**[5]** ESTADOS UNIDOS. National Research Council. Nutrient requirements of beef cattle. 7th rev. ed. Washington: National Academic Press, 2000.

**[6]** PALHARES, J. C. P. Qualidade da água na produção animal. Comunicado Técnico 103. Embrapa, 2011.

**[7]** MELO, T. V. Água Na Nutrição Animal. Agência Embrapa de Informação Tecnológica, 2012.

**[8]** MARINO, C. T.; MEDEIROS, T. M. S. R. Minerais e vitaminas na nutrição de bovinos. São Paulo: Roca, 2010.

**[9]** LIMA, G. J. M. M.; PIOCZCOVSKI, G. D. Água: principal alimento na produção animal. Porto Alegre: Embrapa Suínos e Aves, 2014.

**[10]** OLIVEIRA, S. Internet das Coisas com ESP8266, Arduino e Raspberry Pi. São Paulo: Novatec, 2017.

**[11]** STEVAN JR., S. L. Internet das coisas: fundamentos e aplicações em Arduino e Node-RED. Curitiba: Intersaberes, 2020.

**[12]** COMER, D. E. Redes de computadores e internet. 6. ed. Porto Alegre: Bookman, 2016.

**[13]** BELSHE, M.; PEON, R.; THOMSON, M. Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540, 2015.

Pontifícia Universidade Católica de Goiás
*Pontifical Catholic University of Goias*
Av. Universitária, 1069, Setor Universitário
Caixa Postal 86 - CEP 74.605-010
Goiânia – Goiás – Brasil

**RESOLUÇÃO n° 038/2020 – CEPE**

**ANEXO I**

APÊNDICE ao TCC

**Termo de autorização de publicação de produção acadêmica**

O estudante Ulisses Xavier Castilho do Curso de Engenharia de Controle e Automação matrícula 2023.201180003-8 telefone: (62)993415011 e-mail ulissesxca@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado Monitoramento de qualidade d'água para bovinocultura gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 26 de março de 2026.

gov.br Documento assinado digitalmente
ULISSES XAVIER CASTILHO
Data: 26/03/2026 00:22:13-0300
Verifique em https://validar.iti.gov.br

Assinatura do autor: _____

Nome completo do autor: Ulisses Xavier Castilho

gov.br Documento assinado digitalmente
ANTONIO MARCOS DE MELO MEDEIROS
Data: 26/03/2026 17:37:12-0300
Verifique em https://validar.iti.gov.br

Assinatura do professor-orientador: _____
Nome completo do professor-orientador: