UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ANDRÉ MOACYR MARTINS LUCINI
MARLON DA SILVA CRUZ

SISTEMA DE MONITORIA DE RESERVATÓRIO D'ÁGUA COM ACESSO REMOTO UTILIZANDO ARDUINO

CURITIBA
2022

ANDRÉ MOACYR MARTINS LUCINI

MARLON DA SILVA CRUZ

SISTEMA DE MONITORIA DE RESERVATÓRIO D'ÁGUA COM ACESSO REMOTO UTILIZANDO ARDUINO

Arduino remote access in water tank monitoring system

Trabalho de conclusão de curso de graduação apresentada como requisito para obtenção do título de Tecnólogo em Sistemas de Telecomunicações da Universidade Tecnológica Federal do Paraná (UTFPR).
Orientador (a): MSc. Sergio Moribe

CURITIBA

2022

ANDRÉ MOACYR MARTINS LUCINI

MARLON DA SILVA CRUZ

SISTEMA DE MONITORIA DE RESERVATÓRIO D'ÁGUA COM ACESSO REMOTO UTILIZANDO ARDUINO

Trabalho de conclusão de curso de graduação apresentada como requisito para obtenção do título de Tecnólogo em Sistemas de Telecomunicações da Universidade Tecnológica Federal do Paraná (UTFPR).

Data de aprovação: 09/dezembro/2022

_____

Alexandre Jorge Miziara
Mestrado
Universidade Tecnológica Federal do Paraná

_____

Lincoln Herbert Teixeira
Mestrado
Universidade Tecnológica Federal do Paraná

_____

Sergio Moribe
Mestrado
Universidade Tecnológica Federal do Paraná

CURITIBA

2022

Dedicamos este trabalho a todo o curso de Sistemas de Telecomunicações da Universidade Tecnológica Federal do Paraná, corpo docente e discente, a quem ficamos lisonjeados por dele ter feito parte.

AGRADECIMENTOS

Agradecemos a todos os professores por nos proporcionarem o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, por tanto que se dedicaram a nós, não somente por terem nos ensinado, mas por terem nos feitos aprender. A palavra mestre, nunca fará justiça aos professores dedicados aos quais sem nominar terão os nossos eternos agradecimentos.

Porque difícil é errar!
(NABAS; KLEBER, 2019).

RESUMO

Água é um dos recursos mais importante para a vida no nosso planeta. O manejo inadequado e uso não sustentável contribuem para a escassez desse recurso no brasil e no mundo. Alguns fatores são causados pelos seres humanos como o desperdício de água, aumento de consumo devido ao crescimento da população, da indústria e da agricultura. Outros por fatores naturais como as ondas de calor que acontece na China, atualmente (2022), ou a falta de chuvas que a região sul do Brasil sofreu em 2020. Este trabalho tem como objetivo criar um sistema, por meio de telecomunicação, para monitorar o consumo de água.

Palavras-chaves: Telecomunicação; Monitoramento; Água; Servidor; Aplicativo.

ABSTRACT

Water is one of the most important resources for life on our planet. Inadequate management and unsustainable use contribute to the scarcity of this resource in Brazil and in the world. Some factors are caused by humans such as water wastage, increased consumption due to population growth, industry and agriculture. Others due to natural factors such as the heat waves that happen in China, currently (2022), or the lack of rains that the southern region of Brazil suffered in 2020. This work aims to create a system, through telecommunication, to monitor water consumption.

Keywords: Telecommunications; monitoring; Water; Server; Application.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

O mundo passa por uma transformação digital, precisamos de agilidade e meios que possamos aproveitar melhor o nosso tempo e maneiras de resolver problemas automaticamente. Segundo Spaccaquerche (2020), Internet das Coisas (*IoT, Internet of Things*) é o conjunto de tecnologias que coleta dados numa ponta e os transforma em algo que seja identificado pelo negócio, como uma coisa que melhore o negócio.

Neste trabalho foi construído um sistema de boias *micro switch* com três níveis diferentes, Figura 12, e ajustáveis de acordo com a necessidade do usuário, será enviado o status de volume para o micro controlador em intervalo regular de tempo que poderá ser acompanhado na internet. Os três níveis serão: alto, médio e baixo, sendo este último um aviso crítico para o usuário parar com o uso inadequado da água. O micro controlador escolhido foi o Arduino UNO R3 junto com o módulo ESP-01.

O acesso ao nível da água em um reservatório é dado pelo portal Adafruit que permite enviar os *feeds (*dados coletados e/ou processados que serão enviados para a plataforma) do circuito para a internet de forma gratuita. Foi criado um *dashboard* para facilitar a visualização das informações.

O projeto foi pensado para ter o menor custo possível, facilidade na instalação, fácil e rápido acesso as informações quando necessário. Nesse projeto foi utilizado Flange PVC, Canos PVC, cotovelo PVC, adaptadores impressos em 3D para acoplar as boias, sensor de fluxo, fonte, boia *micro switch*, Arduino UNO R3 e o módulo ESP-01.

O desenvolvimento deste trabalho teve duas partes. A primeira montar as partes mecânicas, ajustar os níveis escolhidos e alimentar o sistema. A segunda foi escrever o código no Arduino e configurar o servidor para ter acesso às informações.

1.1 Motivação

A estiagem sofrida no ano de 2020 a 2022 na cidade de Curitiba, capital do Paraná, fez com que a Companhia de Saneamento do Paraná (SANEPAR), junto com o governo do estado, aderisse ao sistema de rodízio de água. Ao todo, foram 649 dias de rodizio, onde os bairros atendidos pelo mesmo circuito hídrico ficariam durante 72 horas com o fornecimento interrompido, para que posteriormente ficasse 24 horas com abastecimento normal (JUSTINO, 2022). Neste momento seria muito importante ter um sistema de monitoria para melhor controle e uso de água.

1.2 Justificativa

Viabilizar um sistema de monitoria com acesso remoto para que o usuário possa saber, aproximadamente, quantos litros de água tem disponível em seu reservatório, podendo assim realizar as tarefas do lar ou de pequenas empresas sem ser surpreendido com a falta de água.

1.3 Objetivo

1.3.1 Geral

Construir um sistema de monitoria de caixa d'água utilizando Arduino, com possibilidade de acesso remoto para identificar qual nível está o armazenamento onde o sistema será instalado.

1.3.2 Específico

- Encontrar os melhores materiais com um custo benefício acessível para os periféricos a serem utilizados no sistema.
- Analisar qual será o melhor protocolo de comunicação a ser usado para comunicação entre os periféricos.
- Desenvolver um protótipo em pequena escala para demonstrar a utilização do sistema.
- Montar a parte mecânica do sistema.
- Instalar o sistema em uma caixa de água.
- Montar a parte eletrônica do sistema
- Configurar o servidor para acesso web das informações.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Produtos similares no mercado

No mercado não existem muitas opções de sistemas de monitoria de reservatórios residências. Geralmente são sistemas profissionais e de alto custo, que demandam de uma instalação profissional.

Devido ao uso de sensores ultrassônicos, o valor destes sistemas ultrapassa a casa dos mil reais, inviabilizando a instalação em residências.

A grande maioria dos monitores de nível do mercado, não possuem modulo WiFi e nem a possibilidade de acessar o nível a distância, sendo necessário a instalação cabeada.

2.2 Materiais utilizados

Com o intuito de criar um sistema de monitoria com valor acessível e com foco em instalação residencial, os materiais usados foram pensados para se ter um custo baixo e de fácil acesso. Como não há necessidade de precisão e não haverá contato com materiais corrosivos, foi utilizado sensores revestidos de plástico, específicos para contato com a água.

2.3 Adafruit

Adafruit é uma maneira mais fácil de colocar projetos na IoT. É um serviço de nuvem com a principal funcionalidade de armazenar e recuperar dados. Exibe dados em tempo real, conecta projetos a serviços da web como Twitter, e todos os itens acima pode ser acessado com o Adafruit IO.

Assim como todos os serviços online grátis tem suas limitações, o Adafruit não é diferente. Podemos mandar apenas 30 pontos de dados por minutos, apenas os dados dos últimos 30 dias podem ficar armazenados. Temos direito a quinze ações por minutos com no máximo 5 painéis em funcionamento no *dashboard*. Já na versão paga, esses serviços são ilimitados e tem um rápido tempo de resposta.

2.4 Inovação Tecnológica (IoT)

A internet das Coisas, ou IoT, é a fusão dos mundos digital e físico. É uma tendência significativa no mundo digital de hoje ter acesso a esse recurso para tomar decisões, ganhar eficiência, reduzir custos e ajudar na interação entre a marca e os

usuários. Segundo Spaccaquerche (2020), os empreendedores que dominarem as tendências inovadoras de IoT terão a oportunidade de liderar a inovação digital em seus negócios.

É fácil entender porque a internet das coisas é uma das tecnologias mais importantes do século XXI. Tudo que usamos se conecta a uma rede e estamos constantemente cercados por sinais de internet, seja WIFI, 4G, 5G, entre outras.

3 METODOLOGIA

Este trabalho teve como finalidade a criação de um protótipo com o objetivo de facilitar o monitoramento de água em uma caixa d'água.

A parte inicial do projeto é dividida em três grupos: mecânica, eletrônica e servidor. A parte mecânica é composta pelos canos, sensores e adaptador que recebem os dados do ambiente. A parte eletrônica com o Arduino Uno e ESP-01 se comunicando entre si utilizando a biblioteca A2A (BRINCANDO COM IDÉIAS, 2022), recebem os dados que são enviados para o servidor que guarda e apresenta os dados ao usuário.

Foi utilizado a IDE (*Integrated Development Environment*) do Arduino (ARDUINO, 2022) como ambiente de desenvolvimento, por ser gratuito e utilizada amplamente pelos desenvolvedores de projetos. A linguagem de programação é baseada em C/C++, linguagem bem conhecida, que facilitou na codificação.

Foi escolhido Adafruit IO como um serviço de nuvem, além de ser de licença livre e atender as necessidades do projeto, e os integrantes já possuíam experiência com a ferramenta, o portal possui ferramentas e bibliotecas (ADAFRUIT, 2022) de ensino de fácil acesso e entendimento.

O *dashboard* é um dos vários recursos disponíveis, e o mais importante para a visualização dos *feeds* dos sensores.

Na parte final do projeto envolve os testes com o protótipo, feito tanto na caixa d'água para testar como o produto final deve se comportar, quanto em um balde para simular os cenários de forma mais rápida. Sanar os problemas com os limitantes dos sensores e melhorias do código final.

4 DESENVOLVIMENTO DO PROTÓTIPO

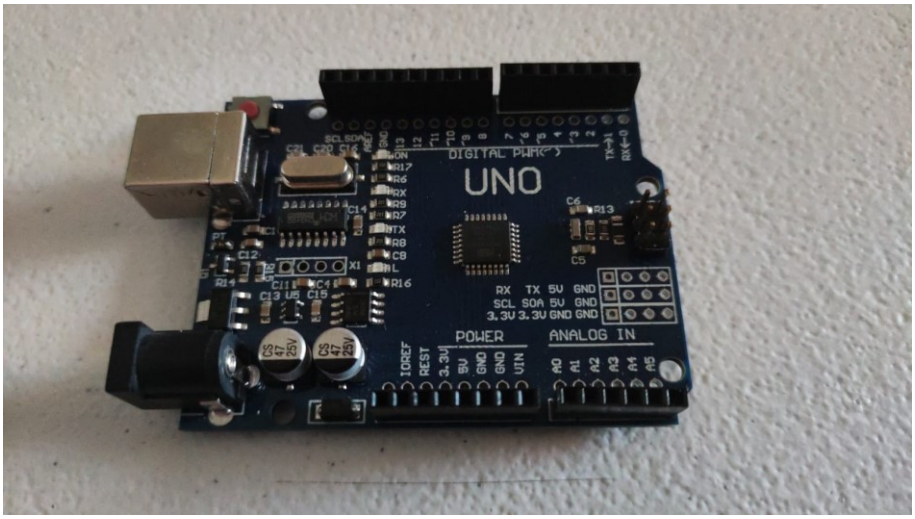
4.1 Arduino

Arduino Uno, Figura 1, é uma placa micro controladora baseada no ATmega328P. Possui 14 pinos de entrada/saída digital (dos quais 6 podem ser usados como saídas PWM), 6 entradas analógicas, um ressonador cerâmico de 16 MHz, uma conexão USB, um conector de alimentação, um conector ICSP e um botão de reset. Ele contém tudo o que é necessário para dar suporte ao micro controlador; basta conectá-lo a um computador com um cabo USB ou alimentá-lo com um adaptador *AC-to-DC* ou bateria. (ARDUINO, 2022).

O Arduino utiliza uma linguagem de programação própria que foi desenvolvida baseada nas linguagens C/C++, e uma IDE própria, fornecida gratuitamente.

Tem um custo muito acessível e uma plataforma *open source*, é muito utilizado para projetos de pequeno a grande porte.

Figura 1 – Arduino Uno

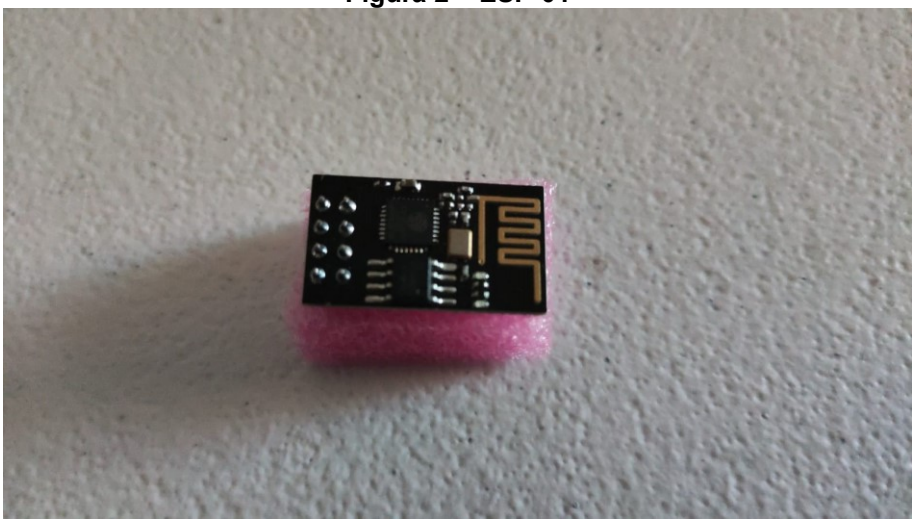


Fonte: Autoria Própria (2022)

4.2 ESP-01

ESP-01, Figura 2, é um modulo que permite ao Arduino acessar facilmente a rede WiFi. Com grande poder de processamento e interface I2C este modulo tem grande alcance a baixo consumo (ESP-01, 2022). Trabalha e uma tensão de 3.3V.

Figura 2 – ESP-01

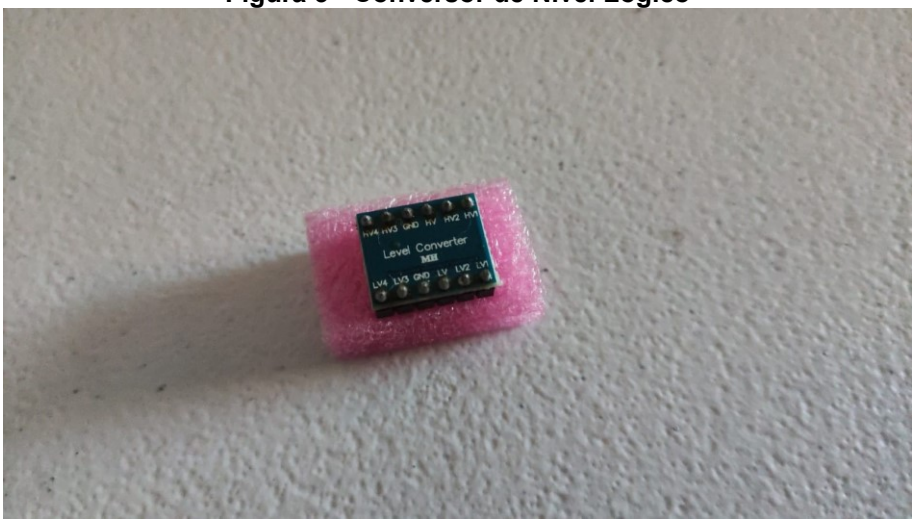


Fonte: Autoria Própria (2022)

4.3 Conversor de Nível Lógico

Devido ao Arduino Uno e o ESP-01 trabalharem em tensões nominais diferentes, houve a necessidade de integrar ao projeto, um conversor de nível lógico Figura 3, que, de maneira bidirecional, converte o nível lógico 0 e 1 de 3.3V para 5V.

Figura 3 - Conversor de Nível Lógico



Fonte: Autoria Própria (2022)

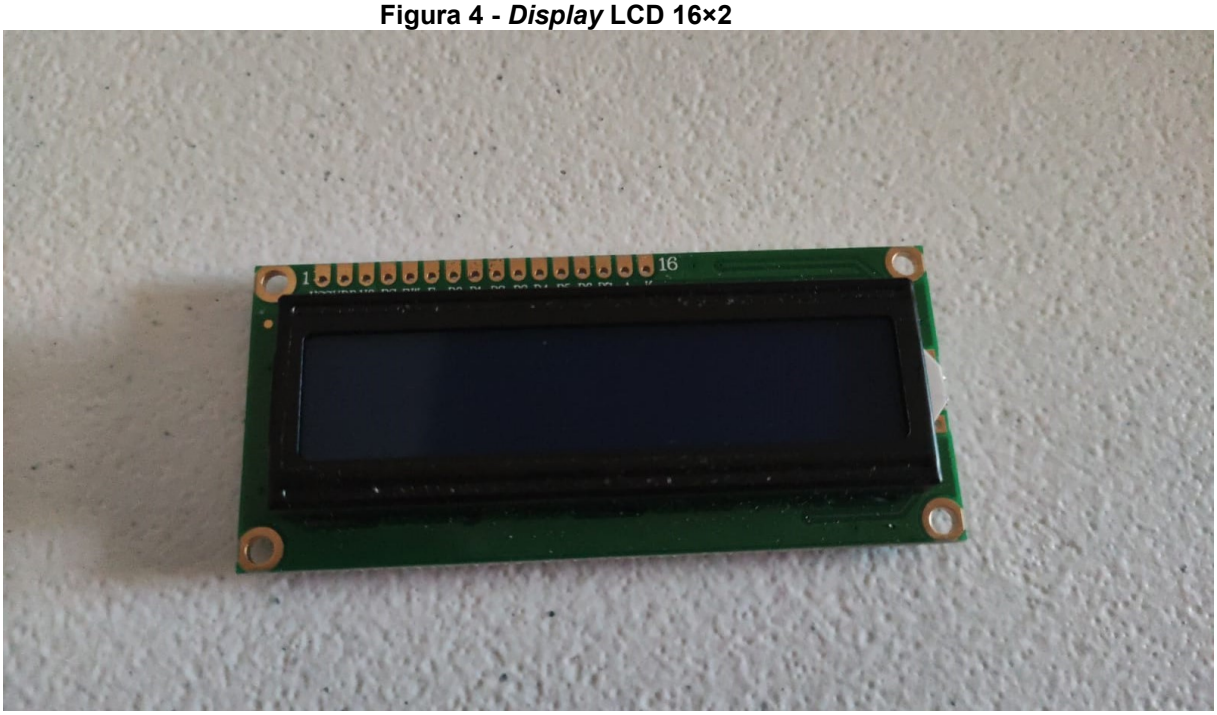
4.4 *Display* LCD 16×2

O LCD 16x2, Figura 4, possui 2 linhas com 16 caracteres, nele é possível escrever caracteres alfanuméricos e símbolos (RAJ, 2022). Para utilização é necessário incluir

na programação do Arduino, uma biblioteca capaz de interagir como chip controlador do *display*.

Essa comunicação se dá através de 16 pinos que funcionam para alimentação, comunicação de dados, ativação e controle de luminosidade.

Figura 4 - *Display* LCD 16×2

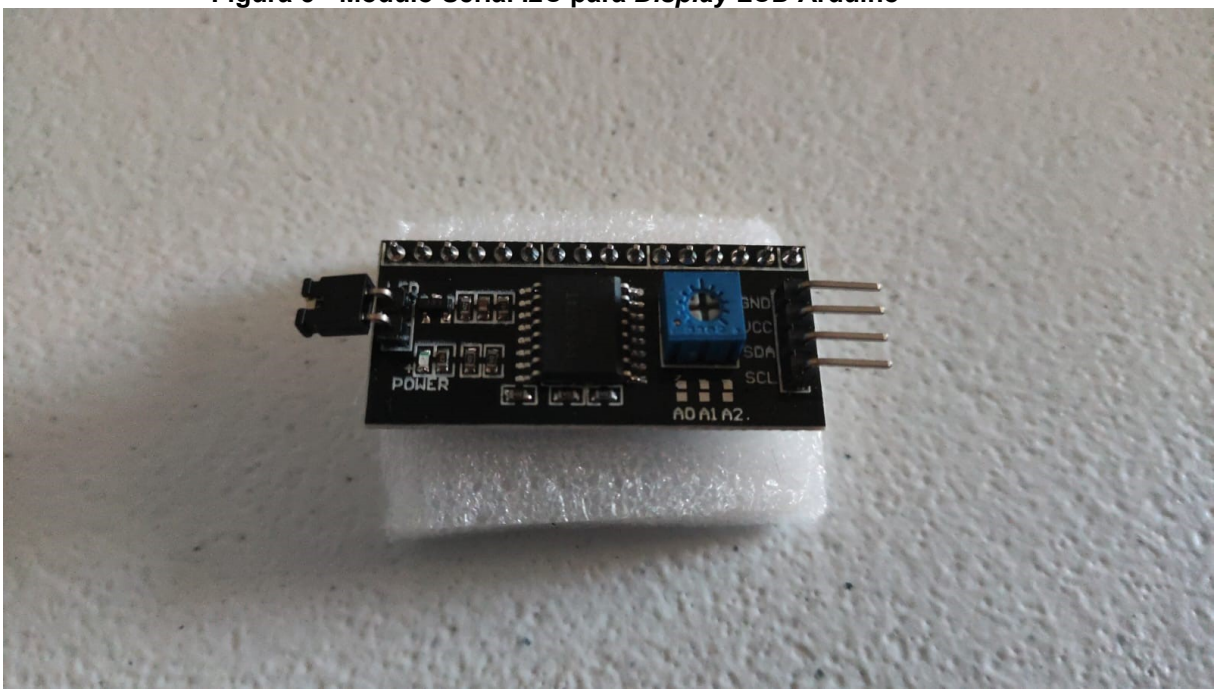


Fonte: Autoria Própria (2022)

4.5 Módulo Serial I2C para *Display* LCD Arduino

Usualmente o *display* LCD necessita de 6 pinos do Arduino para que se possa fazer o controle de caracteres. Com este modulo (HANDSONTEC, 2022), Figura 5, utilizando a comunicação I2C já implementada no Arduino, é possível fazer o mesmo controle com apenas 2 pinos.

Figura 5 - Módulo Serial I2C para *Display* LCD Arduino



Fonte: Autoria Própria (2022)

4.6 Fonte Ajustável Para *Protoboard*

Este modulo, Figura 6, é projetado para se encaixar no *protoboard* padrão, com entrada de 9V a 12V e saída ajustável de 5V e 3.3V necessário para alimentação dos sensores, Arduino uno, esp-01 e *display* LCD.

Figura 6 - Fonte Ajustável Para *Protoboard*

Fonte: Autoria Própria (2022)

4.7 Sensor de Fluxo de água

O sensor de fluxo de água utilizado é o YF-S201, Figura 7, que é um dos mais baratos no mercado e já testado em vários projetos (NEWTON, 2022) com o Arduino Uno R3. Tem um alcance de medição de vazão que vai de 1L a 30L. Pode trabalhar em temperaturas de até 80º C, em caso de o sistema for instalado em lugares mais rigorosos. O diâmetro do sensor é de 36mm e o diâmetro de entrada e saída é de 20mm. A tensão de funcionamento é de DC 4,5V e corrente máxima de trabalho é de 15mA.

Figura 7 - Sensor de Fluxo de água

Fonte: Autoria Própria (2022)

4.8 Sensor de boia

O sensor de nível de água interruptor de boia, Figura 8, foi desenvolvido para automação e controle do nível de água em reservatórios. São utilizados três deles no projeto. Podem trabalhar em temperaturas de até 80º C, assim como o sensor de fluxo.

São chaves NA, que quando acionadas com a aproximação do ima na boia, fecham o contato.

Figura 8 - Sensor de boia

Fonte: Autoria Própria (2022)

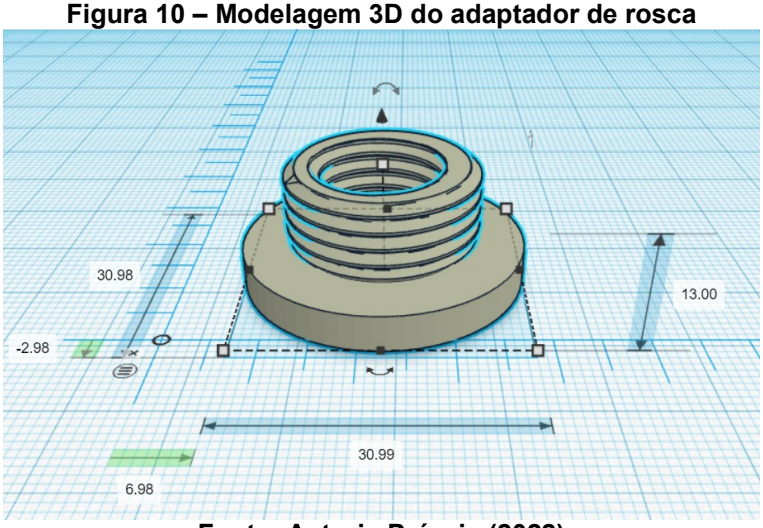
4.9 Adaptador de rosca

Devido a incompatibilidade de roscas do sensor (M12) de boia com os conectores (1/2”) de PVC houve a necessidade de projetar, Figura 10, um adaptador, Figura 9, impresso em 3D, pois não encontramos no mercado.

Figura 9 - Adaptador de rosca

Fonte: Autoria Própria (2022)

Figura 10 – Modelagem 3D do adaptador de rosca



Fonte: Autoria Própria (2022)

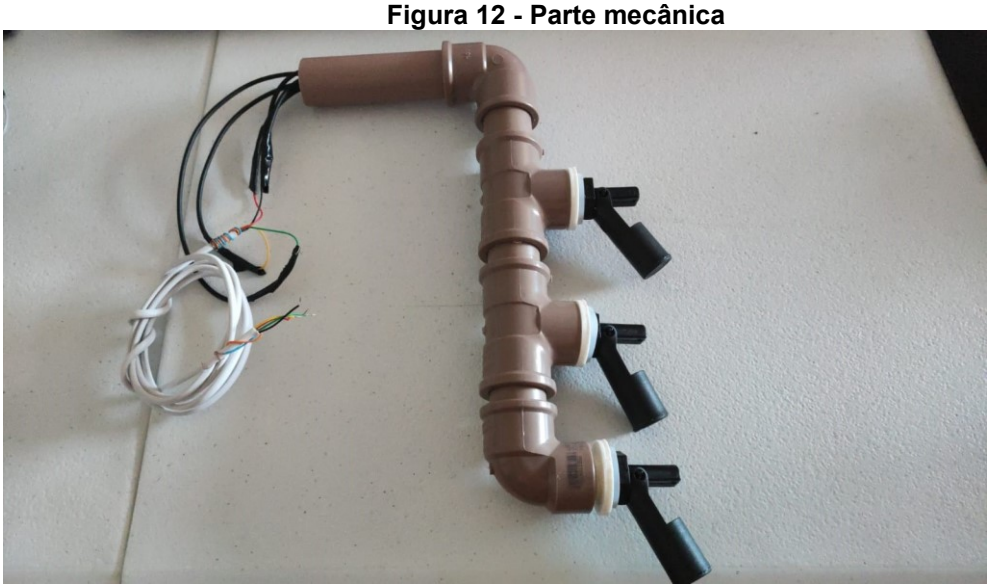
4.10 Parte mecânica

Visando a economicidade, a parte mecânica foi construída com peças de PVC, Figura 11. Além da facilidade de manuseio e resistência à água, essa seria a melhor opção para o projeto. Ela foi instalada na caixa de água e contém 1 flange PVC, 1 cano PVC cortado, 2 cotovelos PVC, 2 T PVC e 3 adaptadores impresso em 3D. Neste aparato, foram rosqueados os sensores de nível. Já o sensor de fluxo, foi rosqueado diretamente na entrada do reservatório.

Figura 11 – Conexões PVC

Fonte: Autoria Própria (2022)

Figura 12 - Parte mecânica

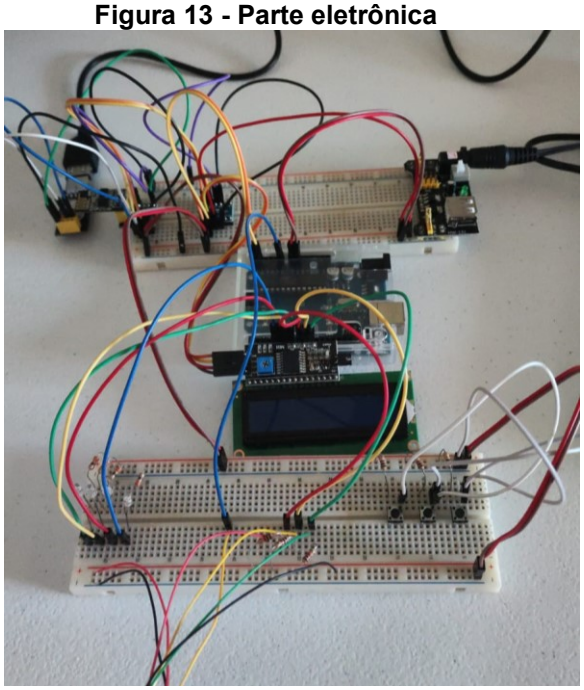


Fonte: Autoria Própria (2022)

4.11 Parte eletrônica

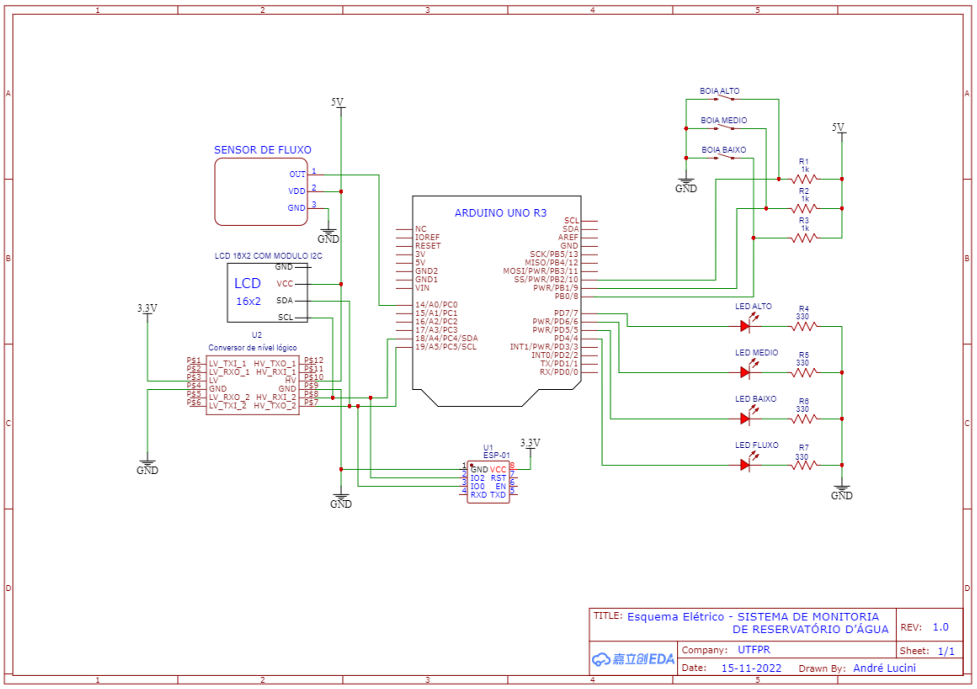
Na parte eletrônica, conforme visto na Figura 13, temos as conexões dos sensores, os periféricos visuais e o Arduino Uno, juntamente com o modulo de conexão WiFi. Utilizando o ESP-01, Figura 14, temos o esquemático elétrico, onde é possível entender melhor as conexões entre os componentes eletrônico utilizados.

Figura 13 - Parte eletrônica



Fonte: Autoria Própria (2022)

Figura 14 – Esquemático elétrico



Fonte: Autoria Própria (2022)

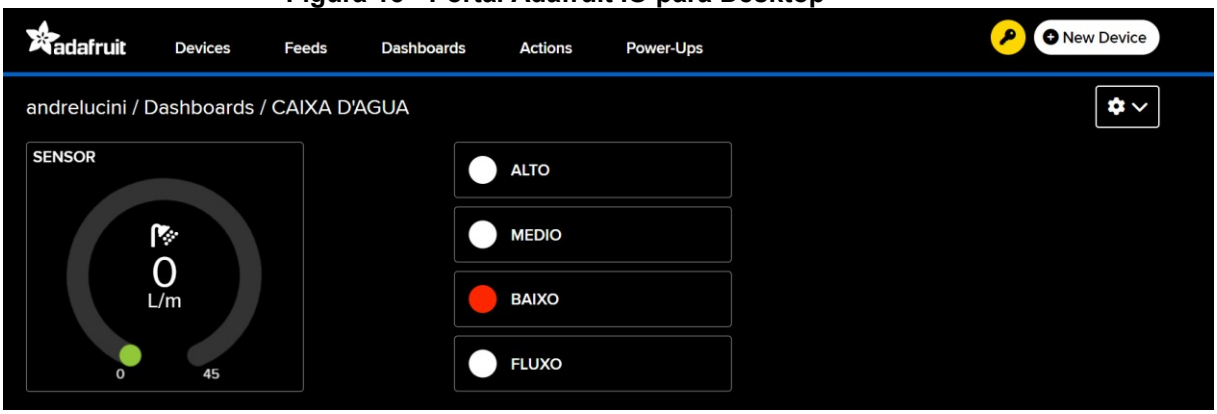
4.12 Portal Adafruit IO

O Adafruit é um serviço de nuvem (ADAFRUIT, 2022) com vários recursos e intuitiva configuração, um desses recursos é o *dashboard*. Nele será indicado o nível de água do reservatório, bem como o se o fluxo de água está presente e qual o seu valor.

Como é possível o acesso ao portal tanto por *desktop*, Figura 15, quanto por *smartphone,* Figura 16, foi realizada configuração de disposição dos elementos de maneira que fique simples a visualização por ambas os acessos.

Os sensores de níveis estão configurados da seguinte forma: alto, médio e baixo.

Figura 15 - Portal Adafruit IO para Desktop



Fonte: Autoria Própria (2022)

Figura 16 - Portal Adafruit IO para *Smartphone*

Fonte: Autoria Própria (2022)

4.13 Custo estimado do projeto

Na, Tabela 1, temos os valores dos materiais utilizados no protótipo, visto que o foco é utilizar os materiais mais baratos no mercado que satisfazem o projeto.

Tabela 1 – Tabela de valores gastos

Item	Quantidade	Custo Unitário	Custo Total
Arduino Uno R3	01	R$ 97,89	R$ 97,89
ESP-01	01	R$ 26,90	R$ 26,90
Modulo I2C	01	R$ 18,99	R$ 18,99
LCD 16X2	01	R$ 22,99	R$ 22,99
Conversor Lógico	01	R$ 16,99	R$ 16,99
LED	04	R$ 0,15	R$ 0,60
Resistores	07	R$ 0,01	R$ 0,07
Mini Bóia	03	R$ 27,50	R$ 82,50
Sensor de Fluxo	01	R$ 35,90	R$ 35,90
Cano PVC ½”	01	R$ 8,00	R$ 8,00

Flange PVC ½”	01	R$ 7,50	R$ 7,50
T PVC ½”	02	R$ 2,50	R$ 5,00
Cotovelo PVC ½”	02	R$ 1,50	R$ 3,00
Cabo 6x40	10	R$ 1,70	R$ 17,00
Total			R$ 343,33

Fonte: Autoria Própria (2022)

5 RESULTADOS

5.1 Protótipo

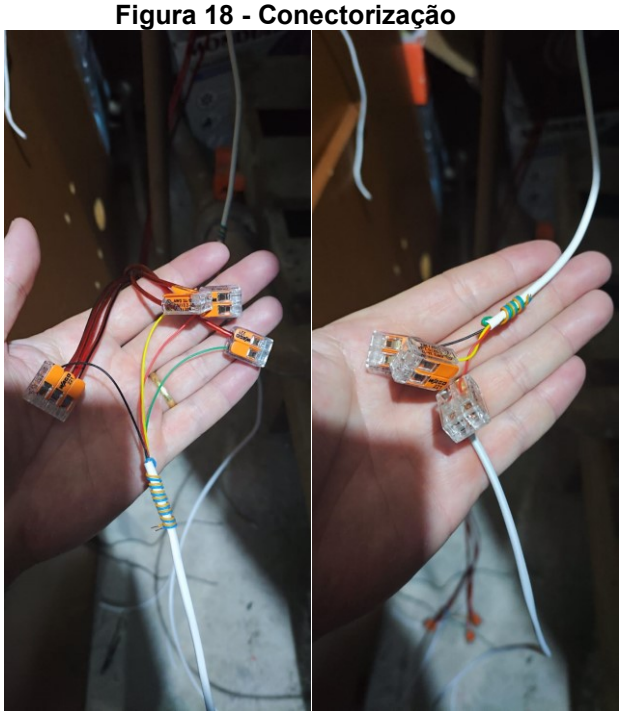
O protótipo foi pensado para instalação em um reservatório circular de 1.000L, de tamanho 1,16m de diâmetro e 97cm de altura. Tendo em vista que a altura interna utilizável para armazenamento é de 87 cm, os sensores “alto”, “médio” e “baixo” foram instalados respectivamente a 42cm, 34cm e 28cm, Figura 17, com volumes aproximadamente 450L, 350L e 220L, respectivamente. A altura do nível morto de água está a 20 cm de altura, esta diferença de 8cm é considerado como reservatório vazio, a fim de informar ao usuário antes que possa entrar ar na tubulação.

Figura 17 – Protótipo Instalado

Fonte: Autoria Própria (2022)

Para a fiação elétrica dos sensores, foi usado 2 lances de cabo 6x40mm multicores e conectores WAGO para realizar as emendas, Figura 18. Um lance para o conjunto de boias e outro para o sensor de fluxo, que foi instalado na entrada de agua do reservatório, antes da boia mecânica.

Figura 18 - Conectorização



Fonte: Autoria Própria (2022)

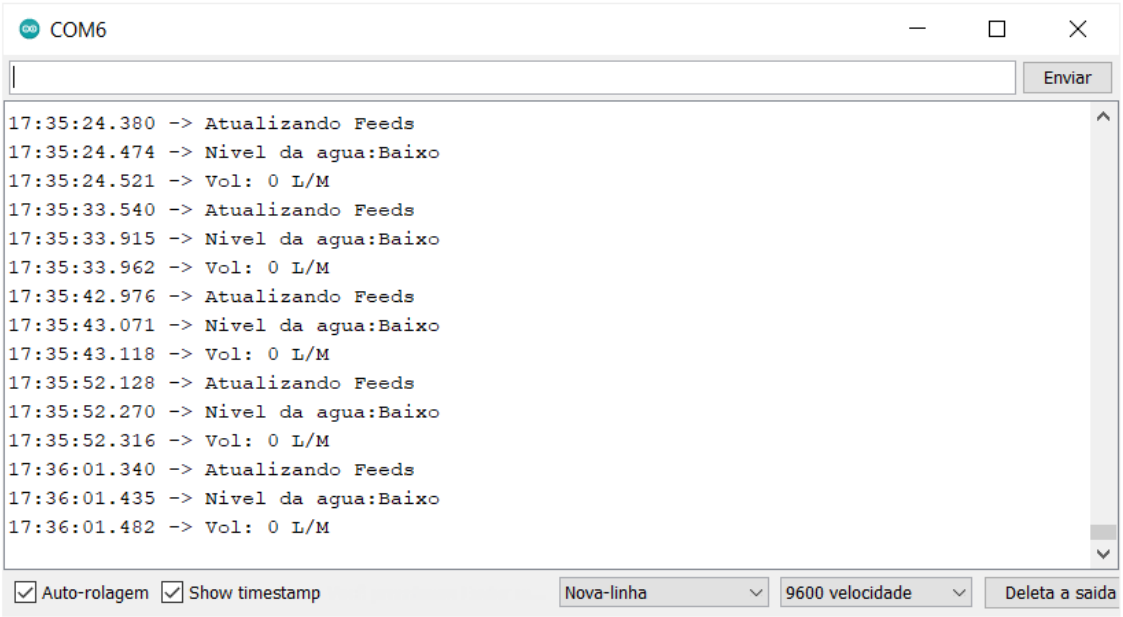
Ligados ao *protoboard* estão os *leds* indicadores de nível de água e fluxo presente. Também no *protoboard* estão ligados o *display* de cristal líquido, o Arduino Uno, o conversor logico de tensão, o ESP-01 e a fonte reguladora.

Para configuração do protótipo, há necessidade de um computador para realizar a gravação dos parâmetros direto no código fonte do Arduino. Os parâmetros devem ser inseridos na biblioteca “config.h”, onde há um campo para definir o usuário e chave do portal Adafruit IO, bem como o SSID e senha da rede WIFI a ser utilizada.

5.2 Testes

O protótipo foi testado durante 1 semana na residência do aluno André Lucini, que durante o uso, fez testes de interrupção de fornecimento de água, fazendo com que o protótipo passasse por todas as situações do cotidiano, tendo sua caixa d’água esvaziada pelo uso rotineiro das atividades residenciais.

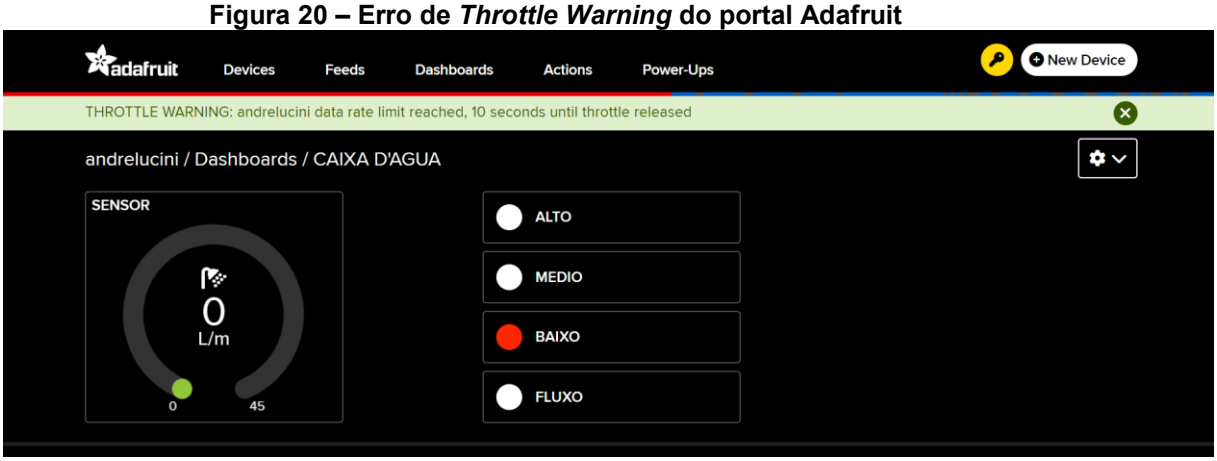
Figura 19 – Log serial do teste



Fonte: Autoria Própria (2022)

Devido ao tipo de conta registrada no portal utilizado, a limitação de atualizações dos *feed*, Figura 20, fez com que durante os testes, as configurações de atualização fossem alteradas para apenas 1 atualização por minuto.

Figura 20 – Erro de *Throttle Warning* do portal Adafruit



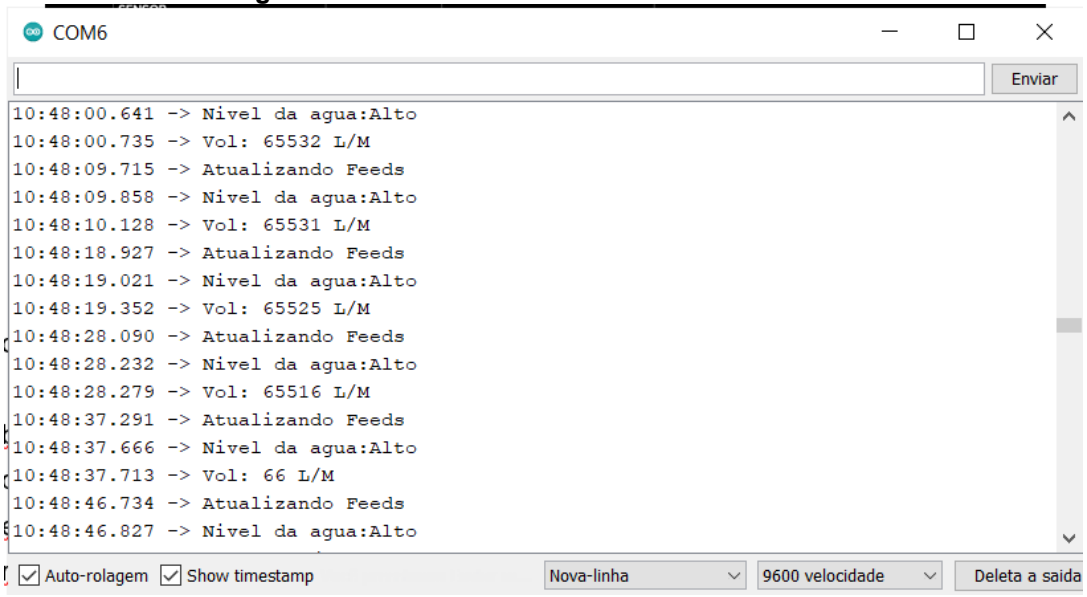
Fonte: Autoria Própria (2022)

A visualização do estado do reservatório ficou muito simples, tanto pelos leds e *display* quanto pelo portal, que poderia ser acessado por computador ou smartphone.

Tendo em vista que o foco do projeto era realizar a monitoração com itens de baixo custo, para tornar o produto acessível, foi usado materiais de baixa precisão, o que

acarretou em pequenos travamentos do sensor de fluxo. Por alguns momentos ele registrava o fluxo muito maior que seu limite máximo, Figura 21, havendo a necessidade de reinicialização do sistema, através da desenergização do mesmo.

Figura 21 – Erro de leitura do sensor de fluxo



Fonte: Autoria Própria (2022)

6 CONSIDERAÇÕES FINAIS

Este estudo focou na monitoração de reservatórios residenciais, cujo foco é apenas o fornecimento de água para uso pessoal, não havendo a necessidade de precisão dos níveis e nem da necessidade de materiais especiais para produtos diferentes. Porém, devido aos materiais usados, fica limitado a instalação do sistema em reservatórios exteriores, com contato direto a intempéries do tempo.

Conforme demonstrado nos resultados, o projeto atendeu aos objetivos propostos, entregando um sistema de monitoria de fácil instalação, custo reduzido e precisão suficiente para atender as demandas de uma residência.

Durante o processo de estudo de caso e criação do protótipo, nos deparamos com a imprecisão dos sensores escolhidos. Tanto o sensor de nível de boia quando o o sensor de fluxo, demonstraram condizer com o baixo custo, entregando tanto falsos positivos quanto falsos negativos.

Estes problemas foram resolvidos na etapa de testes com o tratamento dos sinais recebidos pelo micro controlador. Incluímos na programação a releitura dos

níveis de boia, evitando debounce e laços de verificação para o sensor de fluxo, que em alguns momentos, tendia ao infinito a sua leitura.

Como melhoria e sugestão de atualizações do projeto, já foram pensados a criação de um serviço de hospedagem, diminuindo assim o tempo de atualização dos feeds, tornando o sistema mais preciso.

Tendo em vista que os componentes escolhidos eram de baixo custo e, portanto, de precisão pequena, sugerimos a troca dos sensores por equipamentos de melhor qualidade, evitando leituras erradas e até mesmo o travamento do sistema.

E por último, para facilitar a instalação ao usuário, seria interessante a implementação de um sistema onde fosse possível a configuração da rede WIFI do cliente sem a necessidade da reprogramação do Arduino.

REFERÊNCIAS

ADAFRUIT, **Adafruit IO HTTP API. 2022.** Disponível em: https://io.adafruit.com/api/docs/#adafruit-io-http-api. Acesso em: 15 nov. 2022.

ADAFRUIT, **Libraries. 2022.** Disponível em: https://learn.adafruit.com/welcome-to-adafruit-io/libraries. Acesso em: 15 nov. 2022.

ARDUINO, **Arduino Uno R3. 2022**. Disponível em: https://store-usa.arduino.cc/products/arduino-uno-rev3?selectedStore=us. Acesso em: 15 nov. 2022.

ARDUINO, **Arduino Software. 2022**. Disponível em: https://www.arduino.cc/en/software. Acesso em: 15 nov. 2022.

BRINCANDO COM IDÉIAS, **Biblioteca A2a. 2022.** Disponível em: https://github.com/canalBrincandoComIdeias/A2a. Acesso em: 15 nov. 2022.

ESP-01, **ESP-01 WiFi Module. 2022.** Disponível em: https://www.microchip.ua/wireless/esp01.pdf. Acesso em: 15 nov. 2022.

HANDSONTEC, **I2C Serial Interface 1602 LCD Module. 2022.** Disponível em: http://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf. Acesso em: 15 nov. 2022.

JUSTINO, Adriano. **Novo rodízio de água em Curitiba em 2022 é descartado após as chuvas de outono. 2022.** Disponível em: https://tribunapr.uol.com.br/noticias/curitiba-regiao/novo-rodizio-de-agua-em-curitiba-em-2022-e-descartado-apos-as-chuvas-de-outono. Acesso em: 27 mai. 2022.

NEWTON, Alex. **Arduino Water Flow Sensor to Measure Flow Rate & Volume. 2022.** Disponível em: https://how2electronics.com/arduino-water-flow-sensor-measure-flow-rate-volume/. Acesso em: 15 nov. 2022.

RAJ, Aswinth. **16x2 LCD Display Module. 2022.** Disponível em: https://circuitdigest.com/article/16x2-lcd-display-module-pinout-datasheet. Acesso em: 15 nov. 2022.

SPACCAQUERCHE, Paulo José, **A aplicação da Internet das Coisas nos negócios já é uma realidade. 2020.** Disponível em: https://itforum.com.br/colunas/a-aplicacao-da-internet-das-coisas-nos-negocios-ja-e-uma-realidade/. Acesso em: 15 nov. 2022.

APÊNDICE A - Código fonte do Arduino UNO

```
#include <A2a.h>
#define endereco 0x08
#include <Wire.h>

#define pinSensor 14

A2a arduinoMaster;

int x;
int y;
float tempo = 0;
float frequencia = 0;
int agua = 0;
float total = 0;
float LS = 0;

void setup() {
  arduinoMaster.begin(endereco);
  arduinoMaster.onReceive(receberDados);
  arduinoMaster.onRequest(enviarDados);

  pinMode(pinSensor, INPUT);

  Serial.begin(9600);
}

void loop() {

    x = pulseIn(pinSensor, HIGH);
    y = pulseIn(pinSensor, LOW);
    tempo = x+y;
    frequencia = 1000000/tempo;
    agua = frequencia/7.5;
    LS = agua/60;

   int valorSensor = agua;
   arduinoMaster.varWireWrite(0, highByte(valorSensor));
   arduinoMaster.varWireWrite(1, lowByte(valorSensor));
}

void receberDados() {
  arduinoMaster.receiveData();
}

void enviarDados() {
  arduinoMaster.sendData();
}
```

APÊNDICE B - Código fonte do ESP-01

```
// INCLUSÃO DE BIBLIOTECAS
#include <A2a.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include "config.h"

// DEFINIÇÕES DE PINOS
#define pinAlto 10
#define pinMedio 9
#define pinBaixo 8

#define ledAlto 7
#define ledMedio 6
#define ledBaixo 5
#define ledFluxo 4

// DEFINIÇÕES
#define endereco 0x08
#define tempoAtualizacao 1000 // original 1000
#define tempoConfirmacao 10
#define tempoFeed 9000

// INSTANCIANDO OBJETOS
AdafruitIO_Feed *alto = io.feed("alto");
AdafruitIO_Feed *medio = io.feed("medio");
AdafruitIO_Feed *baixo = io.feed("baixo");
AdafruitIO_Feed *fluxo = io.feed("fluxo");
AdafruitIO_Feed *sensor = io.feed("sensor");

A2a arduinoSlave;

LiquidCrystal_I2C lcd(0x27, 16, 2);

// DECLARAÇÃO DE FUNÇÕES
void configuraSlave();
void configuraMQTT();

bool monitoraAlto(byte pin);
bool monitoraMedio(byte pin);
bool monitoraBaixo(byte pin);
bool monitoraSensor();

// DECLARAÇÃO DE VARIÁVEIS
unsigned long controleTempo = 0;

bool estadoAlto = LOW;
bool estadoMedio = LOW;
bool estadoBaixo = LOW;
int estadoSensor = 0;
```

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

  configuraMQTT();

  arduinoSlave.begin(0, 2);
  configuraSlave();

  lcd.begin(0, 2);
  lcd.backlight();

  io.run();

  Serial.println("Fim do SETUP");
  lcd.setCursor(0,0);
  lcd.print("INICIANDO...");
  delay(2000);
}

void loop() {
  io.run();
//**LEITURA BOIAS*************************************************************
    if (millis() > controleTempo + tempoAtualizacao) {
    if (monitoraAlto(pinAlto)) {
      controleTempo = millis();
    }
  }

  if (millis() > controleTempo + tempoAtualizacao) {
    if (monitoraMedio(pinMedio)) {
      controleTempo = millis();
    }
  }

  if (millis() > controleTempo + tempoAtualizacao) {
    if (monitoraBaixo(pinBaixo)) {
      controleTempo = millis();
    }
  }

//*****Sensor*********************************************
   if (millis() > controleTempo + tempoAtualizacao) {
    if (monitoraSensor()) {
      controleTempo = millis();
      sensor->save(estadoSensor);
    }
  }

//******LOGICA*********************************************************
```

```
    if (!estadoBaixo && !estadoMedio && !estadoAlto){
    Serial.println("Nivel da agua:Alto");
    arduinoSlave.digitalWireWrite(endereco, ledAlto, HIGH);
    arduinoSlave.digitalWireWrite(endereco, ledMedio, LOW);
    arduinoSlave.digitalWireWrite(endereco, ledBaixo, LOW);
    lcd.setCursor(0,0);
    lcd.print("Nivel:Alto   ");
    alto->save(HIGH);
    medio->save(LOW);
    baixo->save(LOW);
    }

    if (!estadoBaixo && !estadoMedio && estadoAlto){
    Serial.println("Nivel da agua:Medio");
    arduinoSlave.digitalWireWrite(endereco, ledAlto, LOW);
    arduinoSlave.digitalWireWrite(endereco, ledMedio, HIGH);
    arduinoSlave.digitalWireWrite(endereco, ledBaixo, LOW);
    lcd.setCursor(0,0);
    lcd.print("Nivel:Medio   ");
    alto->save(LOW);
    medio->save(HIGH);
    baixo->save(LOW);
    }

    if (!estadoBaixo && estadoMedio && estadoAlto){
    Serial.println("Nivel da agua:Baixo");
    arduinoSlave.digitalWireWrite(endereco, ledAlto, LOW);
    arduinoSlave.digitalWireWrite(endereco, ledMedio, LOW);
    arduinoSlave.digitalWireWrite(endereco, ledBaixo, HIGH);
    lcd.setCursor(0,0);
    lcd.print("Nivel:Baixo   ");
    alto->save(LOW);
    medio->save(LOW);
    baixo->save(HIGH);
    }

    if (estadoBaixo && estadoMedio && estadoAlto){
    Serial.println("Nivel da agua:VAZIO");
    arduinoSlave.digitalWireWrite(endereco, ledAlto, LOW);
    arduinoSlave.digitalWireWrite(endereco, ledMedio, LOW);
    arduinoSlave.digitalWireWrite(endereco, ledBaixo, LOW);
    lcd.setCursor(0,0);
    lcd.print("Nivel:Vazio   ");
    alto->save(LOW);
    medio->save(LOW);
    baixo->save(LOW);
    }

if(estadoSensor <= 0){
```

```
  lcd.setCursor(0,1);
  lcd.print("Vol:0.00 L/M     ");
  Serial.print("Vol: ");
  Serial.print(estadoSensor);
  Serial.println(" L/M");
  arduinoSlave.digitalWireWrite(endereco, ledFluxo, LOW);
  sensor->save(estadoSensor);
  fluxo->save(LOW);
  }
  if(estadoSensor > 0){
  lcd.setCursor(0,1);
  lcd.print("Vol:");
  lcd.print(estadoSensor);
  lcd.print(" L/M       ");
  Serial.print("Vol: ");
  Serial.print(estadoSensor);
  Serial.println(" L/M");
  lcd.setCursor(0,1);
  arduinoSlave.digitalWireWrite(endereco, ledFluxo, HIGH);
  fluxo->save(HIGH);
  }
  delay(tempoFeed);
  Serial.println("Atualizando Feeds");
}

//******ALTO*************************************************************************************
*****
  bool monitoraAlto(byte pin) {
  static bool leituraAnt;
  static bool mediaLeitura[5];

  bool leitura = arduinoSlave.digitalWireRead(endereco, pin);
  estadoAlto = leitura;

  if (leitura != leituraAnt) {

     for ( byte i = 0; i < 5 ; i++) {
     mediaLeitura[i] = arduinoSlave.digitalWireRead(endereco, pin);
     delay(tempoConfirmacao);
    }

    byte qtdAnterior = 0;
    byte qtdAtual = 0;

    for ( byte i = 0; i < 5 ; i++) {
      if (mediaLeitura[i] == leituraAnt) {
        qtdAnterior++;
      }
      else {
        qtdAtual++;
```

```
      }

    }

    if ( qtdAtual > qtdAnterior) {
      leituraAnt = leitura;
      return 1 ;
    } else {
      return 0;
    }

  } else {
    return 0;
  }

}

//******MEDIO*************************************************************************************
******
bool monitoraMedio(byte pin) {
  static bool leituraAnt;
  static bool mediaLeitura[5];

  bool leitura = arduinoSlave.digitalWireRead(endereco, pin);
  estadoMedio = leitura;

  if (leitura != leituraAnt) {

      for ( byte i = 0; i < 5 ; i++) {
      mediaLeitura[i] = arduinoSlave.digitalWireRead(endereco, pin);
      delay(tempoConfirmacao);
    }

    byte qtdAnterior = 0;
    byte qtdAtual = 0;

    for ( byte i = 0; i < 5 ; i++) {
      if (mediaLeitura[i] == leituraAnt) {
        qtdAnterior++;
      }
      else {
        qtdAtual++;
      }

    }

    if ( qtdAtual > qtdAnterior) {
      leituraAnt = leitura;
      return 1 ;
    } else {
```

```
      return 0;
    }

  } else {
    return 0;
  }

}

//******BAIXO*****************************************************************************
******
bool monitoraBaixo(byte pin) {
  static bool leituraAnt;
  static bool mediaLeitura[5];

  bool leitura = arduinoSlave.digitalWireRead(endereco, pin);
  estadoBaixo = leitura;

  if (leitura != leituraAnt) {

    for ( byte i = 0; i < 5 ; i++) {
    mediaLeitura[i] = arduinoSlave.digitalWireRead(endereco, pin);
    delay(tempoConfirmacao);
    }

    byte qtdAnterior = 0;
    byte qtdAtual = 0;

    for ( byte i = 0; i < 5 ; i++) {
      if (mediaLeitura[i] == leituraAnt) {
        qtdAnterior++;
      }
      else {
        qtdAtual++;
      }

    }

    if ( qtdAtual > qtdAnterior) {
      leituraAnt = leitura;
      return 1 ;
    } else {
      return 0;
    }

  } else {
    return 0;
  }

}
```

```
//***SENSOR*****************************************************
bool monitoraSensor() {
  static int leituraAnt;

  byte byte1 = arduinoSlave.varWireRead(endereco, 0);
  byte byte2 = arduinoSlave.varWireRead(endereco, 1);
  unsigned int leitura = byte1 << 8 | byte2;

  if (leitura != leituraAnt) {

    estadoSensor = leitura;

    leituraAnt = leitura;
    return true;
  } else {
    return false;
  }
}

//*****************************************************************************

void configuraSlave() {
  Serial.println("configurando pinMode do Arduino");
  arduinoSlave.pinWireMode(endereco, pinAlto, INPUT);
  arduinoSlave.pinWireMode(endereco, pinMedio, INPUT);
  arduinoSlave.pinWireMode(endereco, pinBaixo, INPUT);
  arduinoSlave.pinWireMode(endereco, pinBaixo, INPUT);
  arduinoSlave.pinWireMode(endereco, ledAlto, OUTPUT);
  arduinoSlave.pinWireMode(endereco, ledMedio, OUTPUT);
  arduinoSlave.pinWireMode(endereco, ledBaixo, OUTPUT);
  arduinoSlave.pinWireMode(endereco, ledFluxo, OUTPUT);

  arduinoSlave.digitalWireWrite(endereco, ledAlto, LOW);
  arduinoSlave.digitalWireWrite(endereco, ledMedio, LOW);
  arduinoSlave.digitalWireWrite(endereco, ledBaixo, LOW);
  arduinoSlave.digitalWireWrite(endereco, ledFluxo, LOW);
  Serial.println("Slave Configurado!");
}

void configuraMQTT() {
  Serial.print("Conectando ao Adafruit IO");
  io.connect();

  while (io.status() < AIO_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println();
```

```
  Serial.println(io.statusText());
}
```