UNIVERSIDADE FEDERAL DO TOCANTINS
CAMPUS UNIVERSITÁRIO DE PALMAS
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

WECLESON BRANDÃO DA SILVA

MONITORAMENTO DO NÍVEL DE COMBUSTÍVEL DO TANQUE DO GRUPO GERADOR DO DATACENTER DA UFT UTILIZANDO MICROCONTROLADOR ESP8266.

PAMAS/TO
2019

WECLESON BRANDÃO DA SILVA

MONITORAMENTO DO NÍVEL DE COMBUSTÍVEL DO TANQUE DO GRUPO GERADOR DO DATACENTER DA UFT UTILIZANDO MICROCONTROLADOR ESP8266.

Trabalho de Conclusão de Curso apresentado à UFT – Universidade Federal do Tocantins – Campus Universitário de Palmas, Curso de Engenharia Elétrica, como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica.

Orientadora: Dra. Kathy Camila Cardozo Osinski Senhorini.

PALMAS/TO
2019

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da Universidade Federal do Tocantins

S586m Silva, Wecleson Brandão da .

Monitoramento do Nível de Combustível do Tanque do Grupo Gerador do Data Center da UFT utilizando Microcontrolador ESP8266. / Wecleson Brandão da Silva. – Palmas, TO, 2019.

70 f.

Monografia Graduação - Universidade Federal do Tocantins – Câmpus Universitário de Palmas - Curso de Engenharia Elétrica, 2019.

Orientadora : Kathy Camila Cardozo Osinski Senhorini

Coorientador: Arabutan Marques Neres

1. Internet da Coisas. 2. Automatização e Controle. 3. Sistemas Embarcados. 4. Rede Computadores. I. Título

CDD 621.3

Elaborado pelo sistema de geração automatica de ficha catalográfica da UFT com os dados fornecidos pelo(a) autor(a).

WECLESON BRANDÃO DA SILVA

MONITORAMENTO DO NÍVEL DE COMBUSTÍVEL DO TANQUE DO GRUPO GERADOR DO DATACENTER DA UFT UTILIZANDO O MICROCONTROLADOR ESP8266

Trabalho de Conclusão de Curso apresentado à UFT – Universidade Federal do Tocantins – Campus Universitário de Palmas, Curso de Engenharia Elétrica, como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Data de aprovação: 21 / 02 / 2020

Banca Examinadora

Profa. Dra. Kathy Camila Cardozo Osinski Senhorini – Orientadora, UFT

Téc. Lab. Arabutan Marques Neres – Coorientador, UFT

Prof. Me. Alcy Monteiro Júnior – Examinador, UFT

Profa. Dra. Priscila da Silva Oliveira – Examinadora, UFT

PALMAS
2019

AGRADECIMENTOS

Os meus agradecimentos vão primeiramente a minha família pela colaboração, apoio e incentivo em todo o caminho trilhado até esse momento. Vale ressaltá-los aqui: minha mãe, Edineuza Brandão da Silva, pessoa que significa quem sou e nunca deixa de acreditar no meu potencial, meus irmãos que tenho muito apreço: Adriana Brandão da Silva, Wanderson Brandão da Silva e Yara Brandão da Silva Dias. Minha eterna gratidão.

Agradecimento especial a minha orientadora, Profª. Kathy Camila, fundamental com seu apoio para desenvolvimento desse projeto.

Agradecimento especial aos professores Maxwell Diógenes Bandeira de Melo e Priscila da Silva Oliveira, coordenadores anterior e atual, respectivamente, do Curso de Engenharia Elétrica, pelo apoio e compreensão prestados nos últimos semestres do curso.

Agradecimento especial aos amigos pelo apoio, incentivo e camaradagem durante toda essa jornada árdua e grandiosa construída em nossas vidas.

RESUMO

A necessidade de monitoramento e controle nos processos industriais, remotamente e em tempo real, têm crescido de maneira categórica nos últimos anos, levando à conectorização de diversos equipamentos a rede mundial de computares que proporcionam o sensoriamento e atuação no ambiente que estão inseridos. Sendo assim, em ambientes com elevada exigência de disponibilidade dos serviços fornecidos, como é o caso dos *data centers* com seus serviços de Tecnologia da Informação e Comunicação (TIC), não é diferente, faz-se necessário o controle das condições ambientes para manutenção do seu adequado funcionamento e confiabilidade. Em *data centers*, falhas no fornecimento de energia elétrica é uma das situações que se deve evitar ao máximo para não afetar os serviços prestados, por isso são construídos sistemas próprios de suprimento energético em caso de interrupção do fornecimento de eletricidade pela concessionária de energia elétrica. Nesse contexto, propõe-se o monitoramento em tempo real do nível de combustível do tanque diesel que abastece o grupo gerador de eletricidade que atende o *data center* da Universidade Federal do Tocantins. Para isso, foi desenvolvido uma solução com uso do microcontrolador ESP8266 conectado à internet, banco de dados *rela time* e aplicação *Web* para devida visualização do nível de combustível.

Palavras-chaves: Monitoramento. Tanque Diesel. Grupo Gerador. Data Center.

ABSTRACT

The need for monitoring and control in industrial processes remotely and in real time has grown categorically in recent years, leading to the connectorization of various equipment to the worldwide computer network that provide the sensing and performance in the environment that are inserted. Thus, in environments with high demand for the availability of the services provided, such as data centers with their Information and Communication Technology (ICT) services, it is no different, it is necessary to control the environmental conditions to maintain the service. its proper functioning and reliability. In data centers, power supply failures are one of the situations that should be avoided as much as possible so as not to affect the services provided. Therefore, own power supply systems are built in case of interruption of electricity supply by the electric utility. In this context, the real-time monitoring of the fuel level of the diesel tank that supplies the electricity generator group that serves the data center of the Federal University of Tocantins is proposed. For this, a solution was developed using the ESP8266 microcontroller connected to the internet, real time database and *Web* application for proper fuel level visualization.

Key-words: Monitoring. Diesel tank. Generator set. Data center.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O monitoramento e controle de processos industriais, visando o aumento de eficiência e escala na produção de bens, e economia de recursos empregado nesses processos tem crescido bastante, facilitado com o advento da conectorização de objetos à rede mundial de computadores, chamada de Internet das Coisas, que possibilita o acompanhamento em tempo real de tudo o que está acontecendo em uma planta industrial (FIRJAN, 2019).

Destaca-se no desenvolvimento da Internet das Coisas, o avanço conseguido na microeletrônica e comunicação, o que possibilitou miniaturização de circuitos e com isso a criação de microcomputadores que desempenham uma função específica, com menor capacidade de armazenamento e processamento comparado a um computador multifuncional (TOCCI, 2011). Estes circuitos são embutidos em um único chip e recebem o nome de circuitos embarcados. Isso junto às novas tecnologias em comunicações, como exemplos, as diversas concepções de conexão wireless (bluetooth, Wi-Fi, ZigBee, etc.), e também o conceito de *Big Data* e computação em nuvem, conceberam assim a criação de uma rede das redes de alcance maior que a internet, chamada *Internet of Things* (Internet da Coisas ou, do inglês, IoT), (SANTOS et al, 2016).

Assim, de acordo com o exposto nos parágrafos anteriores, procurou-se, nesse trabalho elaborar uma solução de sistema embarcado conectado à Internet, baseado na plataforma Arduino e no microcontrolador ESP8266, para monitoramento do nível de combustível do tanque abastecedor do gerador do *data center* da Universidade Federal do Tocantins em tempo real através de uma página *Web* e com o auxílio do banco de dados Google Firebase Real Time.

1.1 Justificativa

O gerenciamento do nível de combustível diesel para abastecimento de gerador elétrico é uma questão imprescindível ao seu funcionamento adequado e de segurança, qualquer que seja a aplicação dessa geração de energia elétrica. Como é caso quando se trata de *data centers*, que deve ter disponibilidade de funcionamento 24/7, ou seja, 24 horas por dia em 7 dias por semana (VERAS, 2009). Com relação a isso, propôs-se uma solução em IoT para medir o nível de combustível do tanque de abastecimento do *data center* da Universidade Federal do Tocantins, tornando-o mais seguro. Essa medição será realizada através de um sistema microcontrolador conectado à internet para acompanhamento em tempo real do nível de combustível do tanque de abastecimento do grupo gerador.

1.2 Objetivos

1.2.1 Objetivo Geral

Aplicação do sistema microcontrolador ESP8266 conectado à internet para monitoramento do nível de combustível do tanque de abastecimento do data-center da Universidade Federal do Tocantins, utilizando-se uma aplicação *Web* para visualização do volume.

1.2.2 Objetivos Específicos

- Demonstrar a importância de funcionamento do *data center*;
- Demonstrar as características operacionais do grupo gerador a diesel;
- Definição do método de medição de nível adequado;
- Desenvolver código para medição do volume;
- Realizar a integração do módulo ESP8266 e FIREBASE;
- Criar aplicação *Web* para acompanhamento do nível de diesel.

1.3 Metodologia

Nesta seção, são apresentados a classificação e procedimentos metodológicos empregados para o desenvolvimento do presente trabalho.

Método:

Aplicou-se o método hipotético-dedutivo, em que o problema se baseia na verificação do nível de combustível que abastece o *data center* da UFT, e a solução pretendida é monitoração do volume de combustível através de um sistema microcontrolador conectado à internet.

Abordagem:

Empregou-se uma abordagem qualitativa, onde ao final do projeto são verificados o alcance dos objetivos propostos.

Objetivos:

Utilizou-se pesquisa descritiva, que teve como base a pesquisa biográfica e análise documental dos componentes que integram o ambiente de aplicação do projeto.

Natureza:

A natureza empregada foi a pesquisa aplicada para resolução do problema específico, que consiste no monitoramento do nível do tanque.

Procedimento:

Inicialmente para desenvolvimento do projeto aqui proposto, houve uma pesquisa bibliográfica juntamente com pesquisa de campo, para análise do ambiente data center da UFT e embasamento dos objetos que compõem o ambiente da solução, através de visita in loco e consultas a livros, artigos, trabalhos de conclusão de curso, dissertações, teses etc.; essa etapa corresponde ao Capítulo 2, onde foi possível delimitar os componentes empregados no projeto e perceber o grande potencial de desenvolvimento da Internet da Coisas e sua capacidade de transformação em diversas áreas do cotidiano atual.

Após a etapa de embasamento teórico, realizou-se a etapa de implementação da construção do projeto proposto, Capítulo 3 .Então, após aquisição dos componentes efetuou-se a elaboração da montagem do circuito na protoboard, programa em linguagem Arduino, linguagens HTML, CSS e *Java Script*, na criação da página *Web*, e protótipo de tanque para validar o monitoramento do nível de diesel em tempo real do tanque que abastece o grupo gerador para fornecimento de energia elétrica ao data center da Universidade Federal do Tocantins.

O Capítulo 4 é reservado aos resultados e discussão, nele são apresentados a maneira de aferimento das medições do volume de uma jarra, que simula o tanque, para confirmação protótipo desenvolvido. E por fim o Capítulo 5, são realizados as considerações finais e sugestões para trabalhos futuros.

1.4 Estrutura Do Trabalho

Atualmente, o presente trabalho, está organizado em cinco capítulos conforme descrito a seguir.

Capítulo 1 – Introdução:

No primeiro capítulo é realizada uma breve contextualização da área de aplicação em que se encontra o projeto e apresentados as dificuldades e os caminhos a princípio adotados para atendimento dos objetivos declarados.

Capítulo 2 – Revisão Bibliográfica:

O Capítulo 2 trata do referencial teórico adotado para embasamento da área fim que compreende o projeto a ser criado, através de livros, artigos, trabalhos de conclusão de curso, dissertações e teses etc.

Capítulo 3 – Projeto Proposto:

É feito uma descrição conceitual das várias partes que se integram para implementação do projeto em questão.

Capítulo 4 – Resultados e Discussão:

Nesse quarto Capítulo, são apresentados os resultados obtidos na validação do protótipo e a apresentação das medidas na página *Web* criada.

Capítulo 5 – Conclusão:

O último capítulo é realizado os comentários finais sobre os resultados obtidos e feito sugestões de melhorias para trabalhos futuros.

2 REFERENCIAL BIBLIOGRÁFICO

Neste capítulo são presentados os principais objetos de estudo que constituem o ambiente de aplicação da solução proposta por este projeto, detalhando suas funcionalidades e principais características.

2.1 Internet das Coisas

A Internet das Coisas ou *Internet of Things* (IoT) pode ser vislumbrada como extensão da Internet atual, possibilitando a integração à rede mundial de computadores de qualquer tipo de objeto do cotidiano (lâmpadas, veículos, portas, eletrodomésticos, entre outros), que possua capacidade computacional e comunicação. Essa conexão à Internet permitirá o controle remotamente e acesso como provedor de serviços aos objetos, que se convertem assim, em objetos inteligentes (SANTOS et al,2016).

À medida que a tecnologia de Internet das Coisas cresce, se desenvolve e amadurece, novos conceitos sobre sua definição surgem e outros se perdem. Logo, não há um consenso unânime e pronto sobre o que é a Internet das Coisas. Porém, de maneira geral:

> "pode ser entendido como um ambiente de objetos físicos interconectados com a internet por meio de sensores pequenos e embutidos, criando um ecossistema de computação onipresente, voltado para a facilitação do cotidiano das pessoas, introduzindo soluções funcionais nos processos do dia a dia"
> (MAGRANI 2018, p. 20).

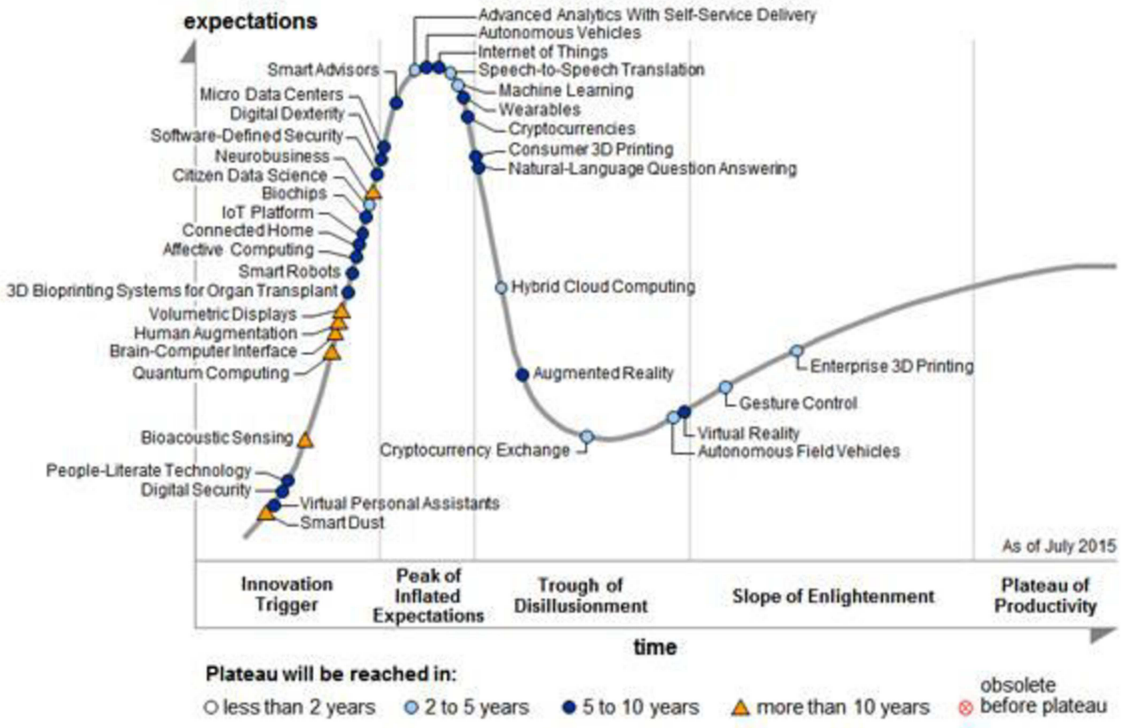
E ainda, segundo o mesmo autor, o que existe de comum entre todas as definições "é que elas se concentram em como computadores, sensores e objetos interagem uns com os outros e processam informações/dados em um contexto de hiperconectividade." (MAGRANI, 2018, p.20).

O surgimento do universo da Internet das Coisas foi possível graças aos avanços de diversas áreas como sistemas embarcados, microeletrônica, comunicação e sensoriamento (SANTOS *et al*, 2016). O termo *Internet of Things* foi cunhado por Kevin Ashton, cofundador e diretor executivo do *Auto-ID Center*, no ano de 1999 durante uma palestra para a *Protect & Gamble*, onde apresentou uma nova proposta de uso de sistemas RFID (*Radio Frequency Identification* ou Identificação por Radiofrequência*)* para rastreabilidade de produtos na cadeia de suprimentos. Ashton já concebia as vantagens de ter objetos do mundo físico conectados à internet. Naquele momento a Internet das Coisas estava baseada principalmente na tecnologia RFID (MANCINI, 2018).

Ao se utilizar os recursos de objetos inteligentes é possível detectar seu contexto, controlá-los, viabilizar troca de informações uns com os outros, acessar serviços da Internet e interagir com pessoas. E com isso, surge uma gama de novas possibilidades de aplicações, como exemplo, cidades inteligentes (*Smart Cities*), Saúde (*Healthcare*), casas inteligentes (*Smart Home*) e também novos desafios emergem, como é o caso da regulamentação, segurança e padronização da IoT (SANTOS *et al*, 2016).

Estudos realizados apontam que mais de 50 bilhões de dispositivos estarão conectados a rede mundial de computadores até o ano de 2020, com uma taxa de 6,58 dispositivos conectados por pessoas (EVANS, 2011). Em 2012, previu-se que a IoT levaria entre cinco a dez anos para ser adotada pelo mercado, e no ano de 2015 se atingiu o pico de expectativas no âmbito acadêmico e industrial, conforme pode ser visto na Figura 1, que é uma maneira de representar o surgimento, adoção, maturidade e impacto de diversas tecnologias emergentes (SANTOS *et al*, 2016).

Figura 1 - Hype Cycle para tecnologias emergentes



Fonte: Santos, 2016.

2.2 Data Center

Os *data centers* ou centro de operação de dados, são ambientes considerados de missão crítica devido sua elevada exigência de disponibilidade dos serviços neles contidos. São espaços que concentram um conjunto de equipamentos integrados de alta tecnologia que permitem o fornecimento de serviços de infraestrutura de TI e valor agregado, predominantemente processamento e armazenamento de dados, em larga escala para qualquer tipo de organização de médio e grande porte, como empresas, órgãos governamentais, instituições de ensino, hospitais, indústrias etc. Assim, as operações diárias dessas instituições estão intimamente relacionadas com a disponibilidade dos *data centers*, devido aos seus sistemas informatizados (VERAS, 2009; e MARIN, 2011).

E ainda de acordo com Marin (2011, p.19), o conceito de *datas centers* não se limita apenas a sala que comporta os diversos equipamentos de TI, trata-se também dos seguintes componentes:

- Sistema de ar condicionado e controle ambiental;
- Distribuidor elétrico e Nobreaks (UPS);
- Automação do edifício;
- Sistema de detecção e supressão de incêndio;
- Segurança e controle;
- Espaço de suporte etc.

O *data center* da UFT é classificado como *enterprise*, em que sua locação, montagem e operação é realizado pela própria instituição. E quanto a classificação de disponibilidade e redundância possui nível *tier* III, pelo fato de ter manutenção e operação simultâneos sem que haja parada de funcionamento. (MARIN, 2011).

2.3 Gerador

Os geradores ou alternadores, são máquinas síncronas utilizadas para converter potência mecânica em potência elétrica CA, através da rotação do eixo em velocidade. Tem por características construtivas um rotor, em eletroímã ou ímã permanente, onde é aplicado uma corrente CC a um enrolamento que ao ser acionado por uma máquina motriz primária (motor de combustão à gasolina, diesel etc.) entra em giro produzindo um campo magnético girante que por sua vez induz tensões trifásicas no estator do alternador. (CHAPMAN, 2013, p.192). A

construção dessa máquina acontece com polos salientes ou não - salientes, como representada na Figura 2.

As tensões por fase geradas podem ser calculadas pela Equação 1:

$$E_{gf} = 4{,}44\Phi N_f f K_n K_p \times 10^{-8}\, V \quad (1)$$

Onde:

Φ - fluxo por polo em linhas ou maxwells
N_f - número total de espiras por fase
f – frequência em Hertz
k_p – fator de passo
k_d – fator de distribuição

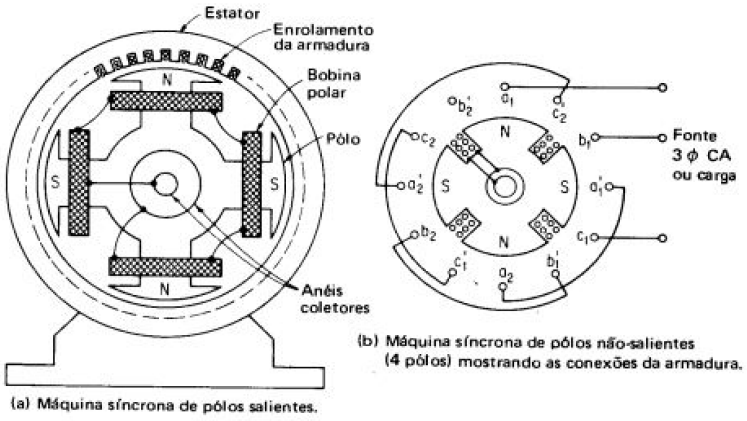
A frequência é relacionada à fatores construtivos, como número de polos e enrolamentos, ela é calculada pela Equação 2:

$$f = \frac{PxN}{120} \quad (2)$$

Onde:

f – frequência em Hertz
P – fator de passo
N – fator de distribuição
(KOSOVO, 200.p. 1982)

Figura 2- Gerador síncrono



Fonte: Kosovo, (1982, p.42)

Tratando-se sobre geradores ou grupos motor-gerador para fornecimento de energia à *data centers*, estes são dimensionados a prover energia em carga plena aos *data centers*, o que

inclui banco nobreak, climatização e iluminação. E segundo Anexo F da ABNT 14565 (2013, p.?), os geradores devem:

- Ser abastecidos por tanques reservatórios de combustíveis com capacidade mínima de 24h de alimentação. Considerações devem ser feitas para permitir maior tempo de alimentação. Contratos para abastecimento de combustível com fornecedores locais devem ser previstos;
- Entrar em operação por chaves de transferência automática, ensaiadas previamente e com rotinas de ensaios periódicos (recomenda-se quinzenalmente).

O grupo gerador instalado para alimentação do *data center*, em casos de falha no fornecimento de energia elétrica pela concessionária, da Universidade Federal do Tocantins é o modelo Cummins C90D6, como visto na Figura 3, sendo fabricado pela empresa Cummins Power Generation.

Figura 3: Gerador Cummins Power Generation C90D6

Fonte: Manual Gerador Cummins Power Generation C90D6 (2010).

Este grupo gerador possuí as seguintes característica, adaptado do manual de funcionamento Cummins Power Generation C90D6 (2010):

- Motor a diesel arrefecido a água;
- Motor de partida elétrico e alternador em 12 Vcc;

- Alternador com enrolamento único;
- Modo de operação *standby* (alimenta a carca apenas em falta de energia elétrica);
- Rolamento único, campo rotativo, 4 polos;
- Excitatriz de CA e unidade retificadora rotativa;
- Conexão 380/220;
- Rotação de 1800 rpm.

No Quadro 1, são apresentados alguns dados de operação desse grupo gerador em modo *standby*.

Quadro 1: Especificações Técnicas do Gerador Cummins C90D6

Potência Nominal	116 kVA/ 93 kW				
Potência Bruta do Motor	108 kW				
Modelo Motor	4BTA 3.9-G4				
Cilindros	4 cilindros				
Cilindrada	3.92 litros				
Consumo de Combustível	Potência Nominal	116 kVA		93 kW	
	Carga Aplicada	Full	3/4	1/2	1/4
	Litros/ Horas	27	21	16	8
Capacidade do tanque da base	200 litros				

Fonte: Adaptado do manual de uso Cummins Power Generation C90D6 (2010).

2.4 Medição de Nível

A altura do nível de uma substância, que pode ser sólido ou líquido, contido em um reservatório é uma das principais variáveis empregadas no controle de processos industriais contínuos, pois a medição de nível proporciona (GONÇALVES, 2003):

- Avaliação do volume ou massa estocado de materiais em tanque de armazenamento;
- Distribuição equilibrada de materiais de processos contínuos onde existam volumes líquidos ou sólidos de acumulação temporária, reações, mistura etc.;
- Manutenção da segurança e controle de alguns processos onde o nível do produto não pode ultrapassar determinados limites.

Um bom controle do processo passa pela escolha do sensor de nível adequado que vai depender das características do processo ser aferido e dos resultados desejados como precisão, exatidão e repetibilidade (INSTRUMENTAÇÃO E CONTROLE, 2017?a).

Existem três métodos para obtenção de sensoriamento de nível: medição direta, indireta e descontínua, que serão apresentados a seguir juntamente com algumas técnicas referentes a cada um desses métodos.

2.4.1 Medição de Nível Direta

Este método segundo Gonçalves (2003, p. 61) "é a tomada como referência a posição do plano superior da substância medida. Neste tipo de medição podemos utilizar réguas ou gabaritos, visores de nível, boia ou flutuador". São técnicas desse método:

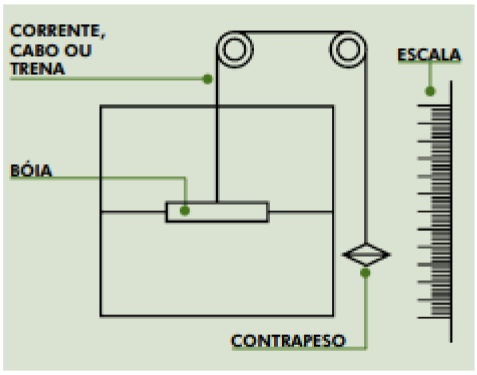
- Régua ou gabarito;
- Visores de nível;
- Boia ou flutuador;

Medição de Nível por Boia ou Flutuador

Uma das maneiras mais tradicionais de medição de nível em líquidos, baseia-se nos princípios da flutuação de Arquimedes, ainda é amplamente adotado devido seu baixo custo de aplicação, simplicidade e confiabilidade proporcionada (THOMAZINE; ALBUQUERQUE, 2011).

Nesse tipo de medição, a boia fica suspensa sobre o líquido variando sua posição de acordo com o nível do fluido. São aplicados geralmente em tanques abertos, com a boia acoplada a um contrapeso na parte externa do tanque através de cabo e polia, que indicará o nível do líquido em uma escala, como representada na Figura 4 (RIBEIRO, 1999).

Figura 4 – Sensor boia com escala linear

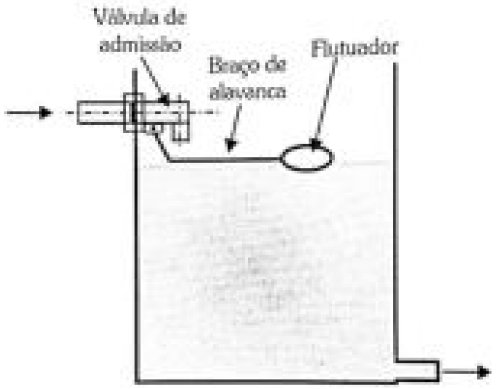


Fonte: Instrumentação e Controle (2019).

É possível ainda aplicação de sensores boias em processos que exigem o armazenamento em ambientes pressurizados, graças ao uso de selo entre o processo e o indicar de escala. Na maior parte dos casos isso ocorre com uso de acoplamento magnético, foles pneumáticos e conexões mecânicas (RIBEIRO, 1999).

Outro emprego usual de medidores boia, apresentada por Ribeiro (1999, p.306), acontece com acionamento direto do controle de nível pela posição da boia nas superfícies da substância. São conhecidos por chave boia, bastante aplicadas em caixas d'água residencial como visto na Figura 5.

Figura 5 - Controle de nível por chave boia



Fonte: Fialho, 2010 apud Silva; Silva (2017).

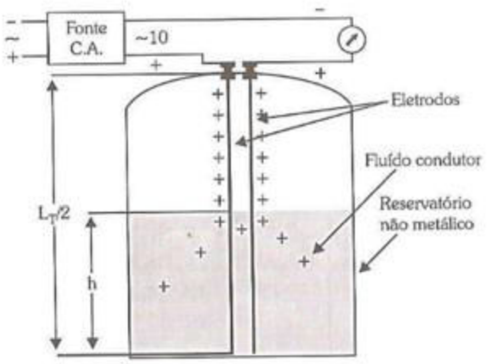
Medição de Nível por Eletrodos

São sondas condutoras (eletrodos) usadas para medições de nível pontuais, valores pré-estabelecidos, de líquidos condutores não voláteis. Podem ser uasados dois ou mais eletrodos

para indicar os níveis definidos. Caso as paredes do tanque sejam de metal, podem ser usadas como eletrodo comum, dessa maneira é necessário inserir apenas um eletrodo no tanque. (DUNN, 2013, p.91).

Quando o líquido está em contato com duas sondas, a tensão entre as sondas faz com que uma corrente flua indicando que um nível definido foi atingido e então um contato elétrico é fechado. Então, sondas podem ser usadas para indicar quando o nível do líquido está baixo ou acima do estabelecido, e assim acionar uma bomba para preencher o recipiente ou bloquear o enchimento (DUNN, 2013, p.91). A Figura 6, ilustra esse método de medição.

Figura 6 – Tanque com par de elétrodos



Fonte: Fialho, 2010 apud Silva, Silva (2017?, pag. 4)

2.4.2 Medição Nível Indireta

Neste tipo de medição o nível é aferido indiretamente em função de uma segunda variável que representa grandezas físicas associadas a substância em observação como pressão, empuxo, radiação e propriedades elétricas (GONÇALVES, 2003).

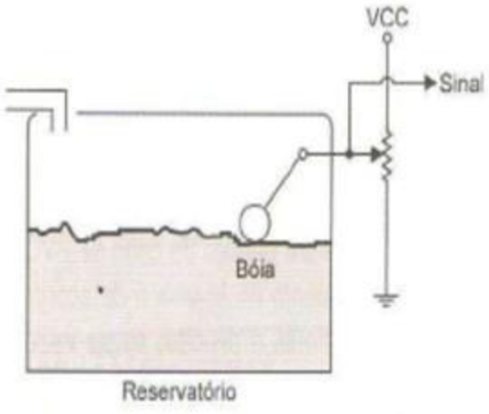
Na maioria dos casos os sensores desse tipo convertem a grandeza mensurada em sinal elétrico proporcional para serem tratados e enviados a dispositivos que iram realizar o controle do processo em questão. A seguir são apresentados algumas dentre as diversas técnicas ou métodos que podem ser utilizadas para medição de maneira indireta do nível do fluido.

Medição por Boias ou Flutuadores Potenciométricos

Nesta técnica de medição ao invés de um contrapeso acoplado a boia, como citado anteriormente no caso de medição direta, é acoplado potenciômetro à boia cuja resistência

elétrica varia a medida que o nível da substância se desloca, fornecendo uma tensão de saída proporcional ao nível do líquido. Trata-se de uma técnica bastante consolidada na medição de combustível em tanques de automóveis e grupos geradores de eletricidade (THOMAZINE; ALBUQUERQUE, 2011). A Figura 7 demostra essa aplicação.

Figura 7 - Sensor flutuador potenciômetro



Fonte: Thomazine; Albuquerque (2011).

Medição de Nível por Capacitância

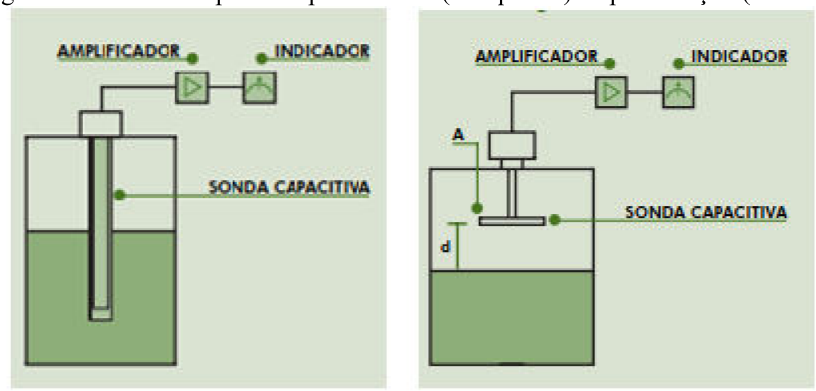
O medidor por capacitância consiste em um eletrodo vertical, haste ou cabo flexível, inserida no recipiente no qual se deseja acompanhar o nível da substância. Essa sonda pode ser isolada (em geral usa-se teflon) ou não, dependendo se o líquido é condutivo ou não, e serve como uma das placas do capacitor, sendo a outra placa a própria parede do tanque. Uma outra maneira de se realizar essa medição ocorre sem contato com o processo, por meio de uma sonda de aproximação (em formato de disco), que constitui umas das placas do capacitor, e a outra placa é a própria superfície do produto ou a base do tanque (GONÇALVES, 2003). Conforme pode ser visto na Figura 8.

À medida que o nível do fluido entre a sonda e a parede do reservatório varia, o valor da capacitância da sonda muda também, isso ocorre devido as diferentes constantes dielétricas do ar e do fluido, resultando em uma capacitância equivalente na sonda que é enviada a um circuito eletrônico para calcular a altura do fluido (THOMAZINE; ALBUQUERQUE, 2011).

O uso da técnica de medição por detectores capacitivos é adequado para tomada de nível contínuo. Possui instalação simples e baixo custo de aplicação. Tem ampla aplicação em líquidos condutivos, porém são especialmente usados para água, solventes, óleos, combustível,

amônia, plásticos líquidos, plásticos granulados, cimentos, alimentos etc. A Figura 8, ilustra essa técnica de medição.

Figura 8 - Sensores capacitivo por imersão (a esquerda) e aproximação (a direita)



Fonte: Gonçalves (2003).

Medição de Nível por Pressão

A medição de nível pela aferição da variável pressão é uma das formas de controle e instrumentação mais empregada em diversos tipos de processos industriais. Os sensores de nível por pressão hidrostática são construídos para serem submersos ou instalados direto no processo e operam de acordo com os seguintes princípios:

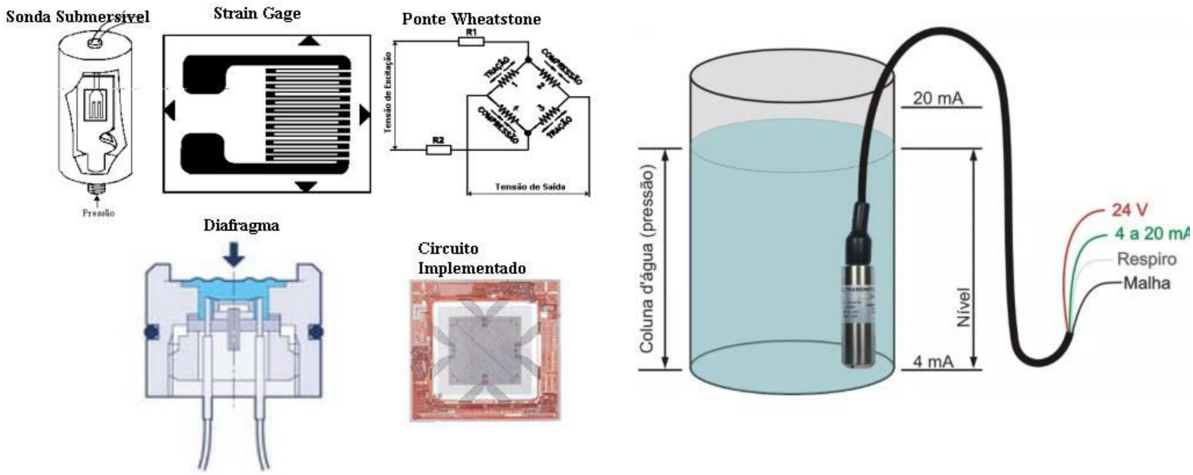
- Capacitivo: capacitor montado atrás de um diafragma, mede a deformação das placas do capacitor;
- Piezorresistivo (extensômetros): baseia-se na variação da resistência como resultado da pressão aplicada;
- Piezoelétrico: produzidos em cristais (quartzo, turmalina ou titanato) que acumulam cargas elétricas em certas áreas da estrutura cristalina quando sofrem uma deformação pela ação pressão.

São constituídos quase sempre por uma membrana que é conectada mecanicamente ou hidraulicamente a um transdutor (converte mecanicamente a pressão em um sinal eletrônico proporcional) baseado nas tecnologias citadas anteriormente (THOMAZINE; ALBUQUERQUE, 2011).

A pressão aferida pelos transdutores são proporcionais à altura da coluna do fluído no tanque e é relacionada pela seguinte equação, de acordo com o Teorema de Stevin: $\boldsymbol{P = h \,.\, \delta}$, onde $\mathbf{h}$ é a altura do volume e $\boldsymbol{\delta}$ é a densidade do líquido à temperatura ambiente (GONÇALVES, 2003).

Na Figura 9 é apresentado uma sonda piezorresistiva de submersão, que tem como elemento sensor um diafragma em ponte de Wheatstone constituído por elementos *strain gauge* (estrutura cristalina que se deforma com aplicação da pressão).

Figura 9 - Sensor de nível piezorresistivo hidrostático por submersão



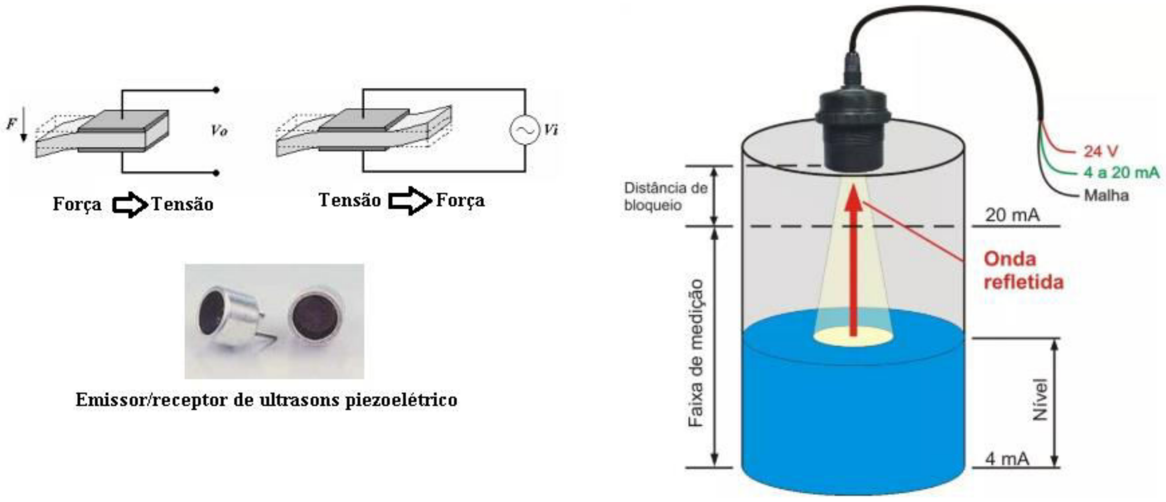
Fonte: Adaptado de Cassiolato (2018) e Alfacomp (2019).

Medição de Nível por Ultrassom

O ultrassom é a onda sonora com níveis de frequência de oscilação superior ao que o ouvido humano pode escutar, ou seja, acima de 20 KHz. Os detectores ultrassônicos baseiam-se no princípio da reflexão das ondas sonoras geradas pelo transdutor, quando encontra a interface da substância a ser medida (THOMAZINE; ALBUQUERQUE, 2011). O transdutor é formado por material piezoelétrico, que é capaz tanto de gerar um sinal sonoro ao ser excitado eletricamente, dito transmissor, como também converte a onda sonora em sinal elétrico, dito receptor (GONÇALVES, 2003).

O nível da substância é medido em relação ao tempo decorrido entre a emissão e o retorno do sinal sonoro pelo transdutor em direção a superfície da substância. Os sensores ultrassônicos de nível são indicados quando não há necessidade de contato com a substância a ser medida e não existe gases, espuma ou poeira entre a superfície e o transdutor (THOMAZINE; ALBUQUERQUE, 2011). Esse tipo de sensor é representado na Figura 10.

Figura 10 – Componentes construtivos e operação de um sensor ultrassônico



Fonte: Adaptado de Coelho (2004) e Alfacomp (2019).

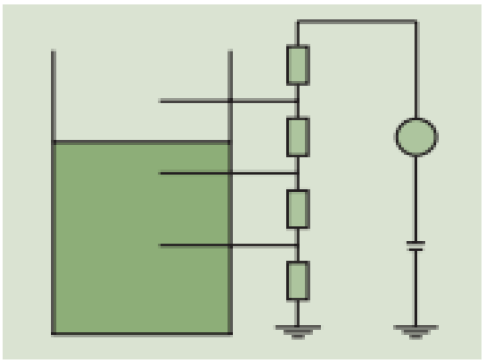
2.4.3 Medição de Nível Descontínua

Os medidores deste tipo são indicados para aplicações onde deseja-se a obtenção de níveis pontuais do fluido, como é caso de sistemas de alarmes e segurança de nível alto ou baixo (GONÇALVES, 2003). As principais formas de medição descontinuada são por uso de condutores elétricos e por sistemas boias como apresentado a seguir.

Medição de Nível Descontínua por Condutividade

Este método é realizado em fluidos condutivos, onde são empregados eletrodos metálicos de comprimentos diferentes. Assim, quando houver condução entre dois eletrodos o sistema terá a indicação que o nível do fluido alcançou o mais elevado eletrodo entre os dois. A Figura 11, a seguir, representa esse método de medição.

Figura 11 - Medição de nível descontínua por condutividade

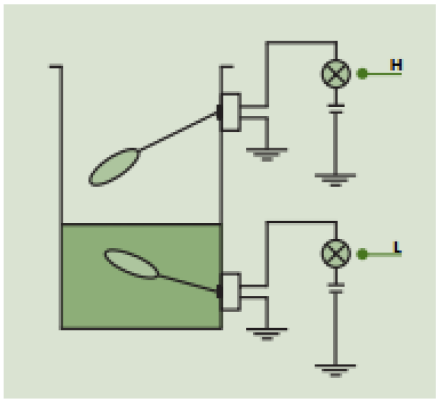


Fonte: Gonçalves, (2003).

Medição de Nível Descontínua por Boia

Segundo Gonçalves (2003, pag. 70) “diversas técnicas podem ser utilizadas para medição descontínua, desde uma simples boia acoplada a contatos elétricos, até sensores eletrônicos do tipo capacitivo ou ultrassônico, que se diferenciam pela sensibilidade, tipo de fluido, características operacionais de instalação e custo”. Veja sua representação na Figura 12.

Figura 12 - Medição de nível descontínua por boia



Fonte: Gonçalves (2003).

2.5 Tanque de Armazenamento do Combustível Diesel

O tanque externo para abastecimento do grupo gerador instalado no data center da UFT tem capacidade de 1000L, com vazão de 40L/min, vem acompanhado com uma bomba 12V, mostrador de nível mecânico (3 dígitos), fabricado em material polietileno e porta palete em aço. Suas dimensões são: (A) 950 mm x (L) 1000 mm x (C) 1150 mm, com peso de 63Kg. Esse tanque é conectado ao grupo gerador através de uma mangueira própria de abastecimento,

conforme pode ser visto na Figura 13. A UFT mantém contrato com posto de abastecimento de combustível próximo para facilitar a logística de abastecimento do grupo gerador, a fim de evitar pane por falta de diesel, que podem comprometer não só o grupo gerador como os serviços informatizados de toda a universidade.

Figura 13 Tanque combustível de abastecimento

Fonte: Arquivo pessoal, 2019

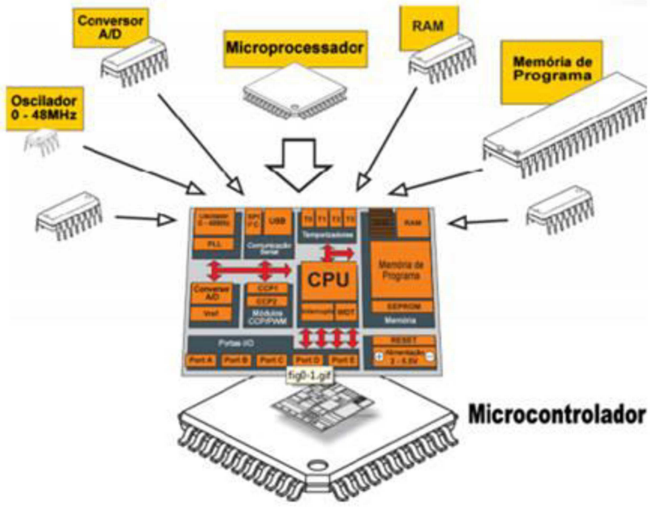
2.6 Microcontrolador e Microprocessador

Os microcontroladores estão presentes em nosso cotidiano executando diversas tarefas específicas como em aparelhos telefônicos, vídeo games, carros, geladeiras, máquinas lavadoras e diversos outros equipamentos eletrônicos que se pode imaginar. São construídos em um único circuito integrado (*chip*) com as mesmas funcionalidades de um computador, porém mais simples, devido serem empregados em tarefas dedicadas que exigem menor capacidade de processamento, e possuí integrado ao seu circuito portas de entrada e saída de dados (TOCCI, WIDMER; MOSS,2011, p.18-19).

Já os microprocessadores são a unidade central de processamento integrado aos microcontroladores com função de processar todas a instruções descritas pelo programa (desenvolvido e implementada pelo projetista), sua construção é condensada em um único *chip* (TOCCI, WIDMER e MOSS,2011, p.18). Os programas, em geral, são desenvolvidos em linguagem de alto nível como a C, C++ etc., e após compilação pelos microprocessadores são convertidos em linguagem de máquina (“0s” e “1s”) para processamento, e após isso, disponibilizar os resultados ao microcontrolador para a devida execução de instruções do programa que irá acionar um periférico conectado a ele (PATTERSON; HENNESSY, 2005).

Assim, os sistemas microcontrolados pelo baixo poder de processamento são considerados microcomputadores, comparados aos sistemas computacionais de uso geral. Ou seja, em um único *chip* são embutidos microprocessador mais periféricos. A Figura 14, ilustra os componentes que formam um microcontrolador em geral.

Figura 14 - Componentes de um microcontrolador



Fonte: Bastos (2013)

2.7 Módulo ESP8266

O módulo ESP8266 é uma plataforma *open source* criada para ser utilizada em desenvolvimento de projetos IoT, a empresa criadora do módulo é a Espressif Systems. Foi construído com um processador Tensilica L106 de 32 bits em arquitetura RISC[1], com frequência de 80/160 MHz e memória flash[2] de até 16MB. Já vem com certificação Wi-Fi Aliance, ou seja, possui um *chip* integrado para conexão em redes Wi-Fi de 2.4 GHz (padrão 8002.11 b/g/n). A interface de periféricos do microcontrolador se dá pelos conectores GPIO (General Purpose Input/Output Interface), tem barramento I2C, SPI, UART, IR, conversor analógico digital (ADC), saída PWM. Sua tensão de operação varia entre 2,5 V a 3,6 V, cada GPIO tem corrente máxima de 12mA. E seu consumo geral fica em torno de 200mA e no modo *deep sleep* 20 µA. (ESPRESSIF SYSTEMS, 2018).

O módulo ESP8266 tem como principal destaque a integração do seu microcontrolador ao *chip* Wi-Fi, chamado de System-On-Chip (SoC), que assim permite maior eficiência energética, circuito compacto e conexão confiável a internet. Existe uma enorme variedade de módulos ESP8266, alguns exemplos são mostrados na Tabela 1.

Tabela 1 - Características de algumas séries do módulo ESP8266

Série	Pinos	GPIO	LED	Antena	Memória Flash
ESP-01	8	2	Sim	Impressa na placa	512KB
ESP-03	14	7	Não	Cerâmica	512KB
ESP-07	16	9	Sim	Cerâmica	1MB
ESP-07S	16	9	Não	Somente conector	4MB
ESP-12	16	11	Sim	Impressa na placa	4MB
ESP-12F	22	11	Sim	Impressa na placa	512KB
ESP-201	26	11	Sim	Impressa na placa	512KB

Fonte: OLIVEIRA (2018, p. 20)

No desenvolvimento do presente projeto foi definido o modelo ESP – 12E, com 4MB de memória flash, antena microfita acoplada na placa e com interface conector USB, como visto na Figura 15.

[1] RISC é o acrônimo de Reduced Instruction Set Computer; em português, "Computador com conjunto reduzido de instruções (CRISP,2004,p.158)"

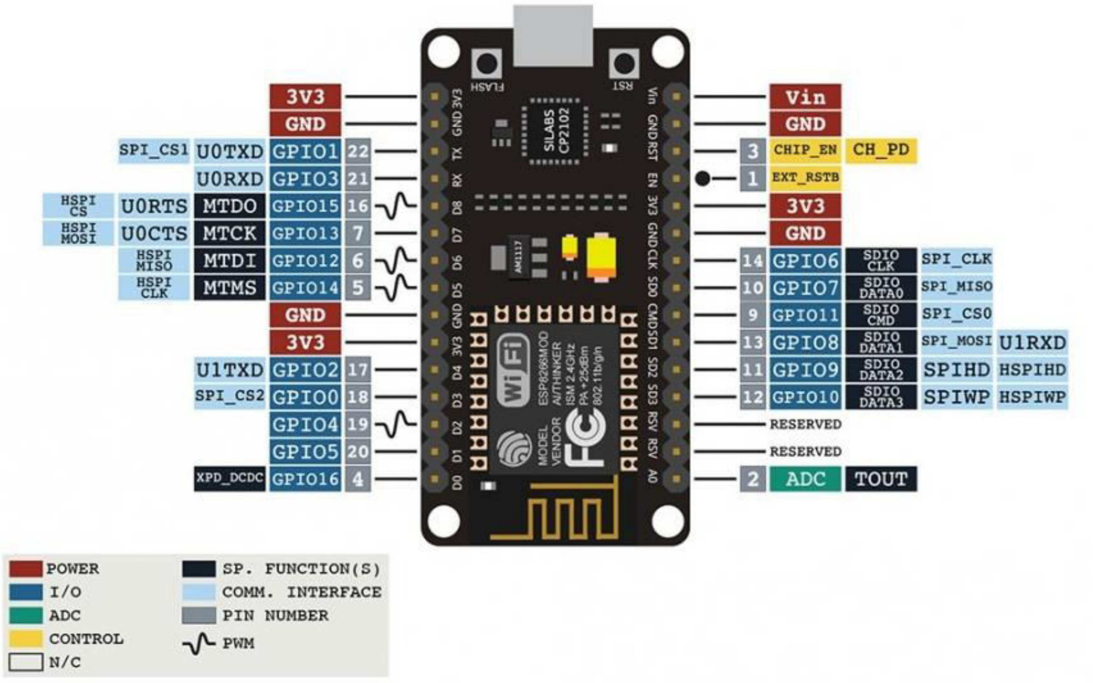
[2] As memórias do tipo *Flash* são não-voláteis, podem ser apagadas eletronicamente e tem esse nome porque possuem curtos tempos de apagamento e escrita (TOCCI, WIDMER e MOSS, 2007, p. 436).

Figura 15 – Módulo ESP8266 modelo ESP – 12E V3

Fonte: Arquivo pessoal (2019)

A alimentação do módulo ESP 8266 pela USB passa através de um regulador de tensão que fornece 3,3 V. Os pinos de acesso às GPIOs têm encaixe padrão compatível com *Protoboards*, e pode-se gravar os programas pela interface USB-Serial. A Figura 16, apresenta o esquema de pino do modelo ESP – 12E.

Figura 16 – Esquema dos pinos módulo ESP8266 modelo ESP -12E



Fonte: Adaptado de Fablab (2018)

2.8 Redes Wi-Fi

A comunicação sem fio, conhecida também por *wireless*, é uma maneira de estabelecer conexão entre dispositivos através de ondas eletromagnéticas, evitando assim, o uso de conexão física por cabos de cobres ou ópticos. As redes *wireless*, geralmente são empregadas onde o uso de cabo torna-se inviável economicamente, como a conexão de dispositivos móveis ou instalação onde deve-se manter a estrutura do prédio intacta (PANSANATO, pág. 83). Leva-se em conta à disseminação dessas redes o fato que atualmente a maioria dos aparelhos eletrônicos são fabricados com adaptadores para conexão sem fio (PINTO; ABRANTES, 2013).

Atualmente o padrão tecnológico *wireless* mais difundindo em nossa sociedade é o Wi-Fi[3] (*Wireless Fidelity* ou Fidelidade sem Fio), que se trata de um conjunto de normas para redes locais sem fio conhecido por WLAN (*Wireless Local Area Network*), que não possui a necessidade de licença dos órgãos governamentais de telecomunicações para instalação e/ou operação. O Wi-Fi baseia-se na norma IEEE 802.11, criada pelo Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) com o objetivo de padronizar a criação e uso de redes locais sem fio, e assim, proporcionar a conexão entre diferentes marcas de equipamentos (PINTO; ABRANTES, 2013).

De acordo com Stangarlin, (2012, p. 19) desde o surgimento o padrão IEEE 802.11, em 1997, diversas melhorias vêm sendo feitas com relação às taxas de transmissão e segurança, e novos padrões são criados. Na Tabela 2, apresenta-se a evolução do padrão IEEE 802.11 com alguns exemplos à medida que novas tecnologias surgiram.

Tabela 2: Evolução padrão IEEE 802.11

Versão do Protocolo	Data de Publicação	Frequência (GHz)	Largura de Banda (MHz)	Velocidade (Mbit/s)
802.11	Jun. 1997	2,4	22	1 -2
802.11b	Set.1999	2,4	22	1-11
802.11g	Jun. 2003	2,4	20	6-54
802.11n	Out. 2009	2,4/5,0	20/40	7,2-150
802.11ac	Dez.2013	5,0	20/40/80/160	Até 866,7
802.11ad	Dez 2012	60	2.160	Até 6912

Fonte: Oliveira (2018, p.15)

[3] A tecnologia Wi-Fi é uma marca de propriedade da organização sem fins lucrativos Wi-Fi Aliance.

O padrão IEEE 802.11 define duas formas de autenticação de clientes *wireless* para acesso à rede: Open System Authentication (sistema de autenticação aberto) e Shared Key Authentication (autenticação com chave compartilhada). No caso do Open System, qualquer requisição de autenticação é aceita para conexão à rede, e para o Shared Key, somente clientes com senha de acesso pode se conectar à rede (INTEL, 2019).

Os principais protocolos de segurança utilizados, desde o surgimento na década de 1990 até hoje, pelas redes sem fio são WEP (Wired Equivalent Privacy), WAP (Wi-Fi Protected Access) e WAP2. Além de autenticação de acesso, esses protocolos garantem a criptografia dos dados que trafegam nas redes *wireless*. O modelo WEP foi adotado como padrão pela IEEE 802.11 em 1999 e tornou-se obsoleto devido sua vulnerabilidade em meados de 2003. Como aprimoramento, ainda em 2003, a Wi-Fi Alliance desenvolveu o WAP que manteve deficiências do seu antecessor e logo foi descartado. Então, em 2004 foi introduzido o WAP2, tendo como diferencial o método de criptografia AES (Advanced Encryption Standard), ainda bastante confiável atualmente. (NETSPOT, 2019).

2.9 Sensor Ultrassônico HC-SR04

O sensor ultrassônico HC – SR04 é bastante difundido na prototipagem de projetos com Arduino para medição de distâncias, veja a Figura 17. E segundo o fabricante Elecfreaks (2011), permite medições de distâncias entre 2 cm a 400 cm, com precisão de 3mm. Abaixo são apresentadas as especificações técnicas desse sensor:

- Alimentação: 5 Vcc;
- Corrente de alimentação: 15mA;
- Faixa de medição: 2 cm a 400cm;
- Resolução: 3 mm;
- Ângulo de detecção: 15º;
- Frequência ultrassônica: 40 kHz;
- Possui 4 pinos: Vcc, GND, *Trigger* (gatilho) e *Echo* (sinal).

(ELECFREAKS, 2011).

Figura 17 – Sensor HC – SR04

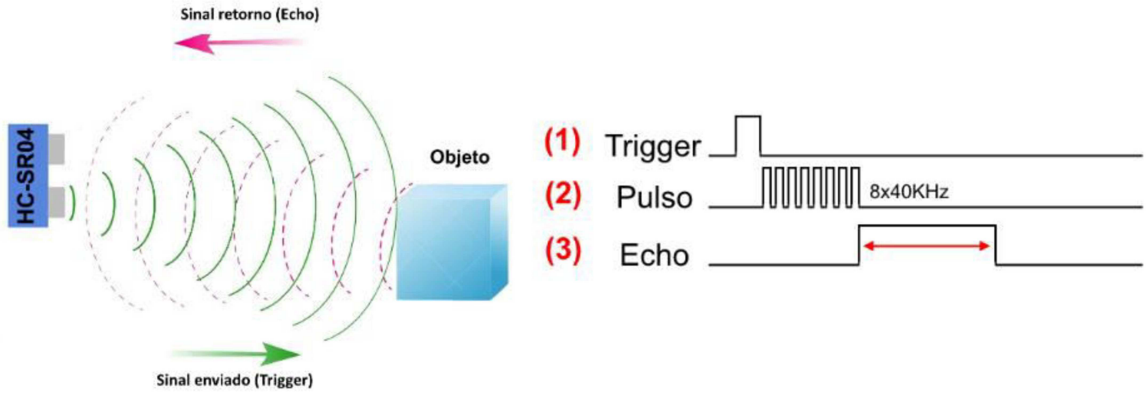
Fonte Mota (2018?)

Esse sensor usa o método de medição do tempo de voo de um pulso, em inglês *pulsed time of flight (TOF)*, ou seja, refere-se ao cálculo do tempo que um pulso de sinal ultrassônico gasta, de ida e volta a um objeto observado, desde a emissão pelo transmissor até a detecção de retorno pelo receptor. Então, com base nesse intervalo de tempo, determina-se a distância entre o sensor e objetor detectado (NIDEJELSKI, 2018, p. 34).

A Figura 17 apresenta o princípio de funcionamento do sensor. Incialmente é efetuado o envio, *trigger* em 5V, de um pulso de 10 µs, indicando o início da transmissão de dados. Em seguida acontece disparo pelo *trigger* de 8 pulsos de 40 KHz, então o sensor aguarda o retorno do sinal, em nível alto, ser detectado pelo *echo* para determinar a distância entre o sensor e o objeto, através da Equação 2 (ELECFREAKS, 2011):

$$Distância = [\,Tempo\ echo\ em\ HIGH\,] \times \frac{\left[velocidade\ som\left(304\frac{m}{s}\right)\right]}{2} \qquad (2)$$

Figura 18 – Princípio de funcionamento do HC – SR04



Fonte Thomsen, (2011)

2.10 Plataforma Arduino

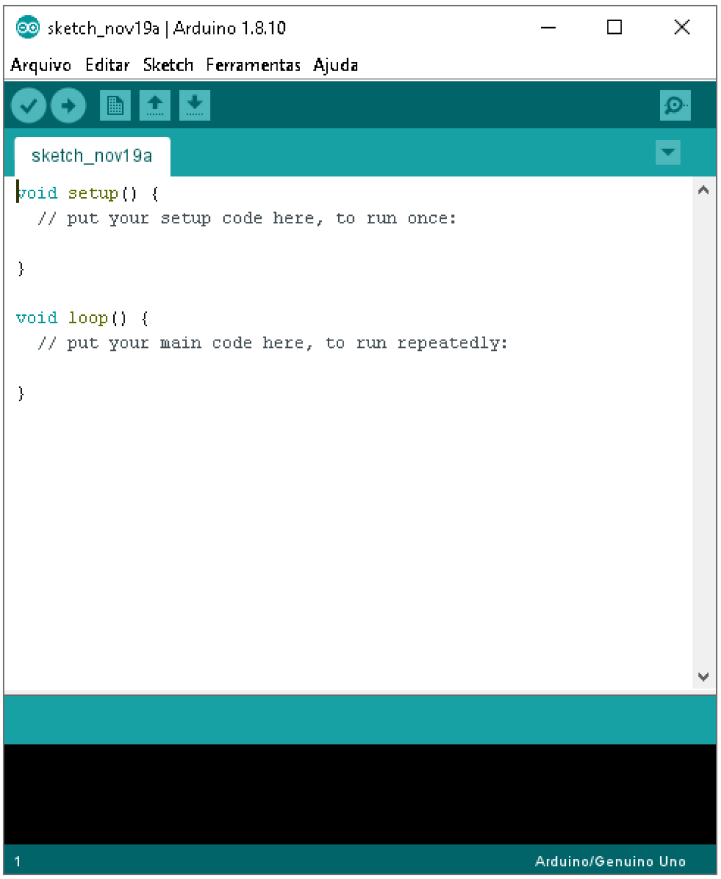
O Arduino é uma plataforma de hardware e software livre desenvolvida especialmente para facilitar a criação de projetos microcontrolados sem a necessidade de grandes conhecimentos em eletrônica e programação, foi lançado em 2005 a fim de ajudar estudantes e amadores a projetarem seus protótipos para controle de sensoriamento do ambiente, de lâmpadas, motores, robôs etc. (ARDUINO,2019).

A placa de prototipagem é composta por um microcontrolador Atmel e circuitos de entradas e saídas. Sendo facilmente programável via Ambiente de Desenvolvimento Integrado (IDE), utilizando linguagem de programação Arduino, baseada em C/C++.

2.10.1 Ambiente de Desenvolvimento Integrado - Arduino IDE

O IDE, sigla para *Integrated Development Environment,* é o software de criação dos programas na linguagem Arduino, chamados de *sketches*, que serão embarcados no microcontrolador, o habilitando a interagir com ambiente. Esse software possui dupla função, pois além de editor de código, realiza a tarefa de compilação dos mesmos em código de máquina. Assim, a IDE permite ao programador escrever o código, corrigir erros de sintaxe, compilar e gravar no microcontrolador via interface USB – Serial (OLIVEIRA, 2018).

Figura 19 – Tela inicial do Arduino IDE, versão 1.8.10



Fonte: Elaboração própria (2019)

Na Figura 19, destacam-se alguns itens e informações importantes, como:

- **Sketch_nov19a**: nome do arquivo dado automaticamente pelo Arduino IDE;
- **Arduino 1.8.10**: versão do software;
-  : botão **Verificar**, usado para verificar erros no código;
-  : botão ***Upload***, usado para transferir o programa para o microcontrolador;
-  : botão **Novo**, usado para criar um novo código;
-  : botão **Abrir**, usado para abrir arquivos criados pelo Arduino IDE;
-  : botão **Salva**, usado para salvar o código atual;
-  : botão **Monitor Serial**, janela que apresenta todas as informações recebidas e transmitidas pela porta serial;
- **Arduino/Genuino Uno**: modelo de placa selecionada.

Salienta-se que qualquer sketch criado deve ser salvo em arquivo com extensão “. ino” dentro de uma pasta de mesmo nome. Em alguns casos, pode ocorrer o não reconhecimento do *driver* da interface USB – Serial pelo sistema operacional, principalmente nos casos de “placas clones”, inviabilizando a gravação do microcontrolador (OLIVEIRA, 2018, p. 24).

2.10.2 Estrutura do código de programação do Arduino IDE

Os programas desenvolvidos no Arduino IDE, em linguagem C/C++, são formados por duas funções principais: *setup ()* e *loop (),* como visto no corpo de edição do *sketch* na Figura 19. A função *setup (),* executada uma única vez durante a inicialização da placa Arduino, serve para definir quais portas serão de entrada e quais serão de saída e para inicializar variáveis e bibliotecas. Em seguida, encontra-se a função *loop ()* contendo o código do programa que será executado continuamente. Todas as funções, variáveis e estruturas nativas da plataforma Arduino podem ser conferidas no site oficial do Arduino, na seção “*Language Reference*” MARGOLIS, 2011, p. 21).

2.10.3 Bibliotecas

As bibliotecas são blocos de instruções, embutidos em outros arquivos, que visam facilitar o desenvolvimento da programação dos *sketches*, tornando o código mais simples e organizado, e por consequência diminuindo a probabilidade de erros. Elas são criadas para habilitarem a execução de tarefas específicas relacionadas a um determinado dispositivo, como a conexão WiFi, sensores, displays, módulos ou a manipulação de dados. Devem ser declaradas no início dos *sketches*, dessa maneira podem ser “chamadas” a partir de qualquer ponto do código do programa (ARDUINO, 2017).

O Arduino IDE vem pré-carregado com várias bibliotecas, como *Ethernet*, *LiquidCristal*, Servo etc. Para ver a lista de bibliotecas disponíveis no menu do IDE, basta clicar em “Sketch” > “Incluir Biblioteca” > “Gerenciador Bibliotecas...” e na caixa “Tipo” podem ser escolhidas várias opções, dentre as instaladas, atualizadas ou recomendadas. No campo de busca pode-se digitar o nome da biblioteca, palavras chaves ou ainda o nome do módulo ou sensor. Uma lista complementar delas pode ser encontrada e baixada na seção *Reference* > *Library* do site oficial do Arduino. Outras centenas de novas bibliotecas podem ser encontrados em diversos sites na internet, principalmente na plataforma de hospedagem e compartilhamento colaborativo de códigos GitHub (ARDUINO, 2017).

2.10.4 Configuração Arduino IDE para ESP8266

Uma das vantagens de se utilizar a Arduino IDE é que ela, além de suportar nativamente diversos modelos de placas da família Arduino e ser bastante consolidada no movimento *maker* (movimento faça você mesmo), ainda suporta outros modelos de placas como é o caso do módulo ESP8266, da família Espressif, sendo necessário apenas instalação dos *cores* (núcleos) para adaptar o software e as bibliotecas com a arquitetura da placa.

Essa adaptação segue os seguintes passos: acessar “Arquivos”> “Preferências”> “URLs adicionais para Gerenciadores de Placas”, e nesse último adicionar o endereço http://arduino.esp8266.com/stable/package_esp8266com_index.json. Em seguida deve-se acessar “Ferramentas”> “Placas”> “Gerenciador de Placas”, e instalar o *core* para o módulo ESP8266 (ARDUINO, 201?).

Deve-se instalar manualmente o drive da interface USB – Serial, pois como dito anteriormente essa placa não pertence à família Arduino. E no caso da versão ESP-12E, usada neste projeto, o modelo do chip da interface USB-Serial é CH340G. E por fim, após a instalação do driver da interface USB-Serial e do *core* do módulo ESP8266, faz-se a seleção da placa em “Ferramentas”> “Placas” > “NodeMCU 1.0 (ESP-12E Module)”.

2.11 Conexão Módulo ESP8266 com o Google Firebase

Os dados gerados em dispositivos IoT na realização de sensoriamento, controle ou monitoramento de certas variáveis do ambiente, para serem visualizadas em uma página *Web* ou aplicativo em tempo real, requerem um elemento de integração entre o dispositivo IoT e página *Web* ou aplicativo *mobile*. Tal integração pode ser feita por plataformas computacionais específicas capazes de armazenar e permitir acesso remoto posteriormente para a devida análise e processamento dos dados gerados. Essas plataformas são bancos de dados que oferecem soluções de gerenciamento e compartilhamento de dados em nuvem[4] (SANTOS et al, *2015*?).

2.11.1 Banco de Dados

Conceitualmente banco de dados pode ser entendido como um armário de arquivamento de dados, sendo os arquivos organizados por gavetas, pastas e tipos. Podendo a qualquer

[4] Serviços computacionais disponibilizados na internet como armazenamento em banco de dados, aplicativos, análises e outros recursos de TI. (AMAZON AWS, 2019a)

momento ser atualizados, acrescentados, excluídos e acessados sempre que necessário com segurança e integridade dos dados (ALECRIM, 2018b). Existem diversos tipos de bancos de dados, porém dois se destacam pela sua grande difusão, são eles, os relacionais e não-relacionais.

Os bancos de dados relacionais como o próprio nome diz é uma coleção de dados inter-relacionados entre si, ou seja, trata-se de uma mesma categoria de informação que são armazenados na forma de tabelas, em linhas e colunas, dentro do banco de dados para representar um objeto. Onde cada coluna contém um tipo de dado (inteiro, números reais, *strings* de caracteres etc.) e campo, e as linhas na tabela representam uma coleção de valores relacionais de um objeto ou entidade que os identifiquem. Utilizam como interface padrão a linguagem SQL (*Structured Query Language*, ou Linguagem de Consulta Estruturada), responsável por adicionar, atualizar excluir linhas de dados, recuperar subconjuntos de dados para processamento de transações e aplicações de análise, além de gerenciar todos os aspectos do banco de dados (AMAZON AWS, 2019b).

Agora os bancos de dados não-relacionais, conhecidos também por NoSQL (Not Only SQL), caracterizam-se por realizar o gerenciamento de imensos volumes de dados e diversos tipos de arquivos, como gráficos, mapas, tabelas, chaves valores, de maneira flexível com alto tráfego e baixo tempo de resposta (AMAZON AWS, 2019c). Sendo por isso o preferível em aplicações *real time.*

Portanto, na escolha da plataforma IoT para o desenvolvimento do projeto em internet das coisas, deve-se levar em consideração o tipo de banco de dados empregado por ela, buscando-se uma que melhor se adequa as necessidades da solução pretendida com a implementação do sistema IoT.

2.11.2 Plataforma IoT

Uma plataforma IoT ou *middleware*[5], são recursos de gerência e infraestrutura de TI disponíveis em nuvem que fornecem serviços de conectividade, armazenamento, processamento e análise às aplicações de sistemas IoT. Tais plataformas consistem em uma camada de *software* inseridas entre as aplicações e a infraestrutura suporte. A adoção de uma plataforma IoT contribui para facilitar a construção de aplicações em IoT (PIRES *et al* 2015).

[5] Middleware é o software de comunicação e gerenciamento de dados que intermedia a interação entre a aplicação final e as fontes de informações, os bancos de dados, em tempo real. (4LINUX).

Atualmente existem diversas plataformas *middleware* voltadas ao universo de internet das coisas disponíveis no mercado (PIRES et al). Então, após análise quanto ao tipo de banco de dados adequado a elaboração deste projeto, em que a organização estrutural dos dados armazenados não é tão importante e o baixo tempo de resposta é primordial, optou-se pelo uso do banco de dados *Firebase Realtime Database* do Google Firebase. Algumas de suas características que levaram a essa escolha são:

- banco de dados NoSQL;
- sincronização em tempo real, sempre que os dados são alterados;
- armazenamento dos dados acontece no formato *Json*[6] (JavaScript Object Notation);
- os dados podem ser acessados e apresentados diretamente em dispositivo móvel ou navegador *Web*, se um servidor de aplicativos.

2.11.3 Plataforma Google Firebase

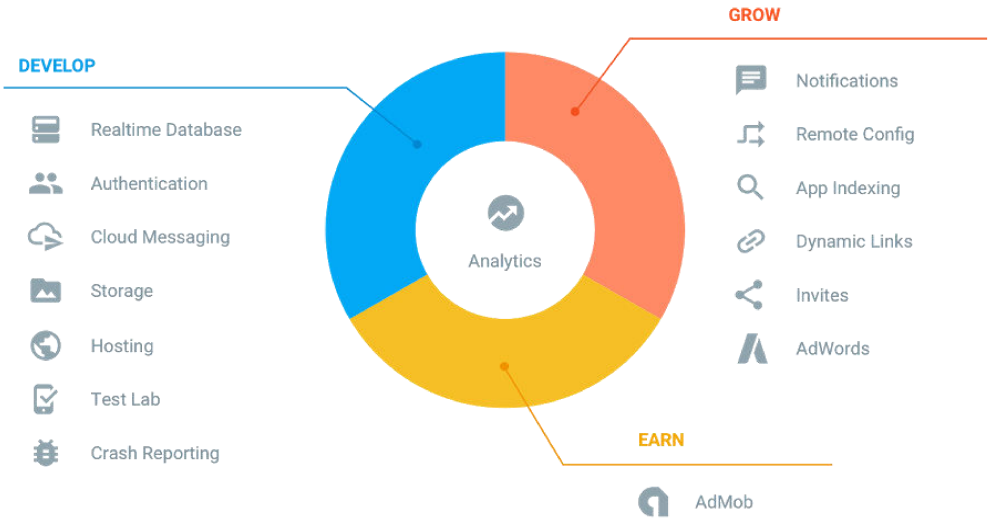
É uma plataforma *middleware*, adquirida pelo Google em 2004, destina-se a fornecer diversos serviços em nuvem do tipo *Backend as a Service*[7] (BaaS) para criação de aplicações de maneira fácil e rápida nos ambientes *Web* e *Mobile* e IoT. Alguns exemplos de serviços oferecidos são análise de dados, autenticação com redes sociais, hospedagem de aplicativos *Web*, envio de notificações, armazenamento de mídia, etc., a Figura 20 dá uma visão geral sobre a gama de serviços disponíveis no Google Firebase (ORLANDI, 2019). Tem grande facilidade de uso pois, apenas com algumas linhas de comando é possível efetivar a integração de determinada aplicação com seu banco do Firebase. Ainda na sua versão gratuita, encontra-se disponíveis a maioria desses serviços, com destaque para possibilidade de se ter 100 acessos simultâneos e até 10 GB de tráfego de dados por mês.

---

[6] JSON é o formato de representação de dados em um par chave-valor baseado na linguagem de programação *Javascript* (JASON,2019).

[7] Backend as a Service é um serviço de computação na nuvem que visa fornecer módulo de códigos de alguns recursos complexos comuns em aplicações móveis e para *Web*, como autenticação de integração com redes sociais, notificações, armazenamento na nuvem, dentre outros serviços (BATSCHINSKI, 2016).

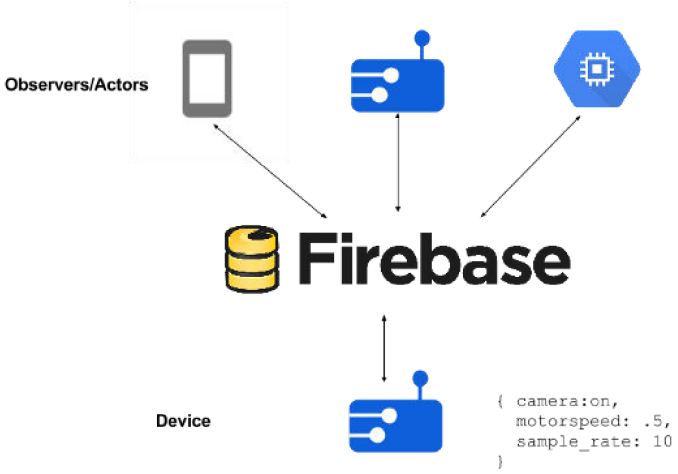
Figura 20 - Serviços disponíveis no Google Firabase



Fonte: Google Developers, 2019.

Como dito anteriormente, no desenvolvimento do presente trabalho foram utilizados os serviços do *Realtime Database* (Banco de dados em tempo real) e o *Firebase Hosting*, para a devida visualização em tempo real do nível de combustível através de uma página *Web*. Na Figura 21, é mostrado a arquitetura de integração do ambiente IoT utilizando a plataforma *Google Firebase*.

Figura 21 - Arquitetura IoT utilizando o Firebase

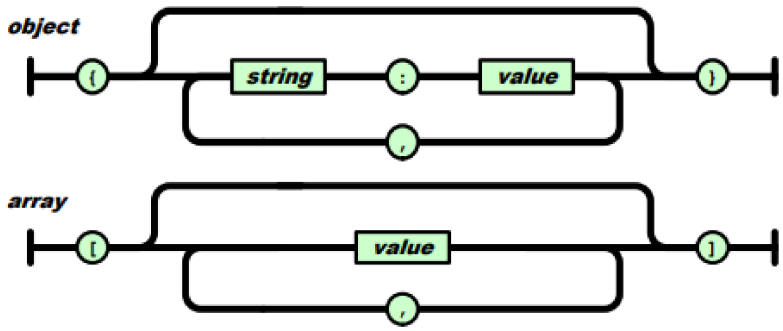


Fonte: FIREBASE (2017)

2.12 Formato JSON

O Firebase Realtime Database armazena os dados no formato JSON (JavaScript Object Notation - Notação de Objetos JavaScript), em que cada objeto é representado por chave/valor, onde as chaves representam os nomes dos atributos da classe e cada atributo possui o valor do objeto. Essa forma de representação de dados, mostrada na Figura 22, é baseada na linguagem de programação Javascript.

Figura 22 – Formato da entidade objeto em JSON



Fonte: EMCA INTERNATIONAL (2017)

O JSON se estrutura em *objects* (objetos) e/ou arrays (listas). Os objetos são representados por {} com uma coleção de pares chave–valor separados por dois pontos; e os *arrays* por [] com uma lista ordenada de valores separados por vírgula. Além desses dois tipos principais de dados, existem quatro tipos de dados básicos que podem constituí-los, que são: string, número, booleano e nulo. Salienta-se ainda que, um objeto pode ter atributos do tipo *array* e um array pode ser do tipo objeto ou *array* (ALVES, 2018).O Firebase armazena os dados como objetos JSON, a Figura 23 demonstra como o objeto carro pode ser representado nessa linguagem (FIREBASE, 20-?).

Figura 23 – Objeto carro representado em JSON

```
1 {"Carro":"[id=1, modelo=Celta, placa=AAA1234]"}
```

(GAMA, 2011).

2.13 Aplicação *Web*

Uma maneira de se compartilhar informação através da internet se dá por meio da *Web,* ou melhor *World Wide Web* (rede de alcance mundial), onde hiperligações entre as páginas

permitem acesso às informações armazenadas em servidores de hospedagem no formato digital de hipertexto. Assim, a *Web* pode ser entendida como um sistema de documentos em hipermídia (vídeos, sons, hipertextos e figuras) que são interligados e compartilhados com todo o mundo através da Internet (TAVARES, 2012).

O acesso a uma aplicação *Web* é realizada utilizando-se os chamados *browser* (navegadores), como alguns exemplos desses, pode-se citar: o Mozilla Firefox e Google Chrome. Esses facilitam a busca, ou requisição de acesso, na internet por meio de links URL (localização de recurso uniforme) que funcionam como uma ponte entre os diversos sites disponíveis. As aplicações *Web* são implementadas com base em três principais ferramentas (MDN WEB DOCS, 200?):

- protocolo HTTP, responsável pela transmissão dos dados;
- sistemas de endereçamento próprio, o URL;
- e a linguagem HTML, que trata da formatação dos documentos transmitidos.

2.13.1 Linguagens HTML, CSS e Java Script

A construção de uma página *Web* baseia-se nas linguagens HTML, CSS e Java Script, que são usadas para estruturar e formatar o conteúdo apresentado em um website. A seguir são descritas a função dessas “três camadas”.

A linguagem HTML (Hypertext Markup Languge - Linguagem de Marcação de Hypertexto), é a primeira camada em um *website*, e encarrega-se pela marcação da estrutura do conteúdo em seções como títulos, parágrafos, cabeçalhos, imagens, vídeos e links etc. Assim, o HTML define o que é cada informação e à estrutura no seu devido lugar dentro da página *Web* (EIS; FERREIRA, 2012).

Enquanto o CSS (Cascading Style Sheets), é a segunda camada de uma página *Web*, responsável pela apresentação visual do conteúdo estruturado pelo HTML. Essa ferramenta em suma trata da formatação do conteúdo para exibição de maneira adequada em diversos tipos de dispositivos como tablets, smartphones etc. (EIS; FERREIRA, 2012).

E por último vem a linguagem Java Script, que fica responsável pelo controle comportamental dos elementos CSS, o que significa se esses elementos serão arrastados, dimensionados, rotacionados, reformados etc (EIS; FERREIRA, 2012).

Essas ferramentas foram necessárias para implementação da página *Web* onde o nível de combustível do tanque será apresentado.

2.13.2 Firebase Hosting

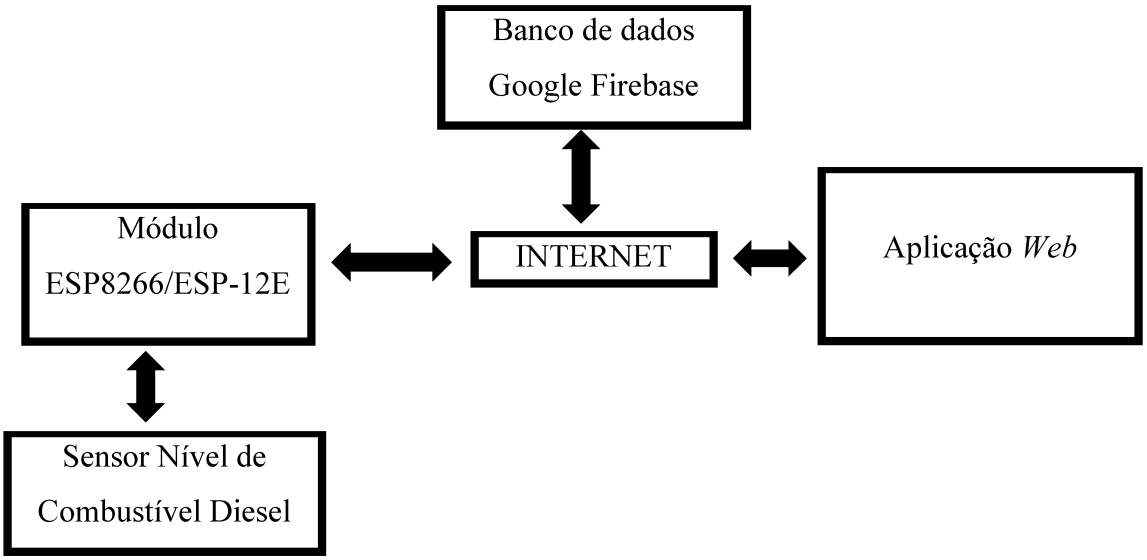
O Firebase Hosting é um serviço de hospedagem para diversos tipos de aplicações *Web* e *mobile* em HTML, CSS e JavaSrip, oferecido pelo Google em planos que vão desde acesso gratuito, para iniciantes e *hobystas*, com 1GB de armazenamento e 10GB/mês de transferência de dados, até planos para profissionais com centenas de GB. Esse serviço provê automaticamente certificação SSL (Secure Sockets Layer) ao *website* para acesso seguro, por criptografia de sessão, com qualquer navegador em qualquer parte do mudo.

Esse foi o servidor hosting escolhido para hospedagem da página *Web* desenvolvida, já que pertence aos serviços disponibilizados pelo Google na criação projeto de monitoramento do nível de combustível no banco de dados Firebase Real Time. E após efetuar *deploy* da página criada no Firebase Hosting é gerado um endereço URL no subdomínio do Firebase , que permite o acesso a partir de qualquer local no mundo pela internet.

3 PROJETO PROPOSTO

Ao longo da seção anterior foram delimitados todos os componentes que constituem o projeto proposto. Neste Capítulo serão apresentadas as etapas de desenvolvimento e implementação, como mostrado na Figura 24.

Figura 24 – Diagrama de blocos do projeto proposto



Fonte: Elaboração própria (2019)

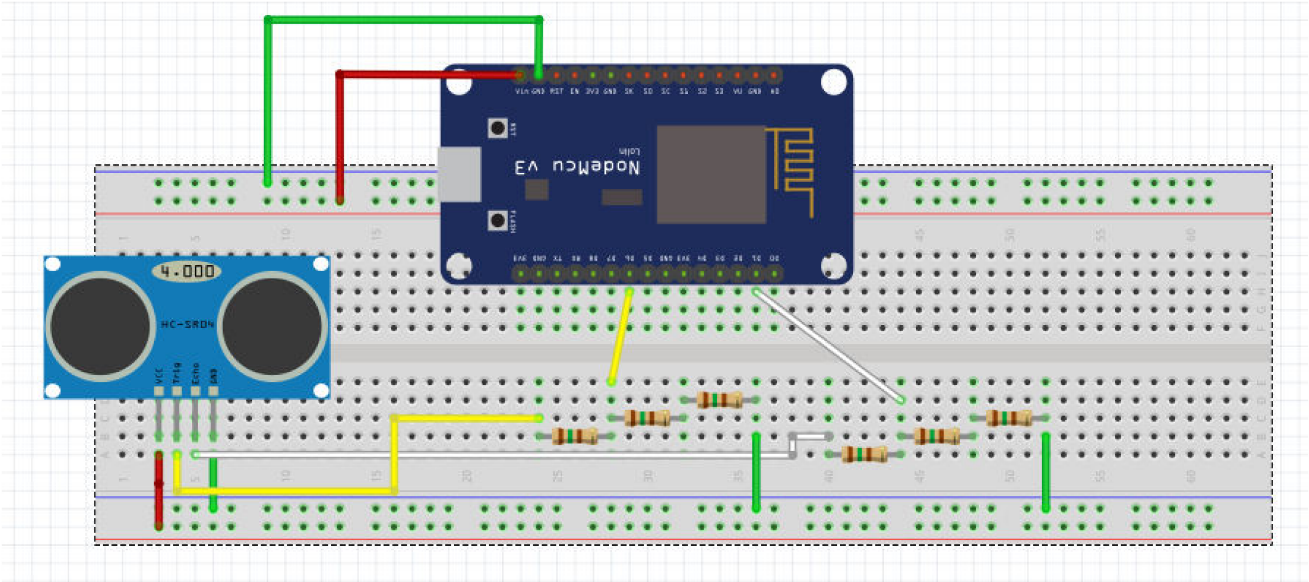
Esse sistema terá instalado no tanque de combustível do grupo gerador do *data center* um sensor ultrassônico que irá aferir as distâncias do limite de capacidade máxima de armazenamento do tanque até a altura do nível que se encontra o diesel. O sinal gerado será enviado ao microcontrolador ESP8266 que executará o cálculo do volume e todos esses dados gerados serão armazenados no banco de dado do Google Firebase Real Time que ficará disponível para serem capturadas e apresentadas em qualquer navegador *Web* por meio do App *Web* desenvolvido, e este estará armazenado no Firebase Hosting.

3.1 Montagem do Circuito

O circuito montado pode ser visto na Figura 25, nela apresenta-se as conexões entre o módulo ESP8266 Node MCU ESP-12E e o sensor ultrassônico HC – SR04. Além desses dois componentes, dois circuitos divisores de tensão, cada um com 3 resistores de 150 Ω a fim de adequar a tensão de operação do sensor ultrassônico, 5V, com a tensão do módulo ESP8266

que é de 3,3 V. Alimentação do módulo ESP8266 e sensor ultrassônico foram efetuadas por uma fonte de celular , 5V e 1 A, adaptada.

Figura 25 – Circuito do projeto montado na *Protoboard*

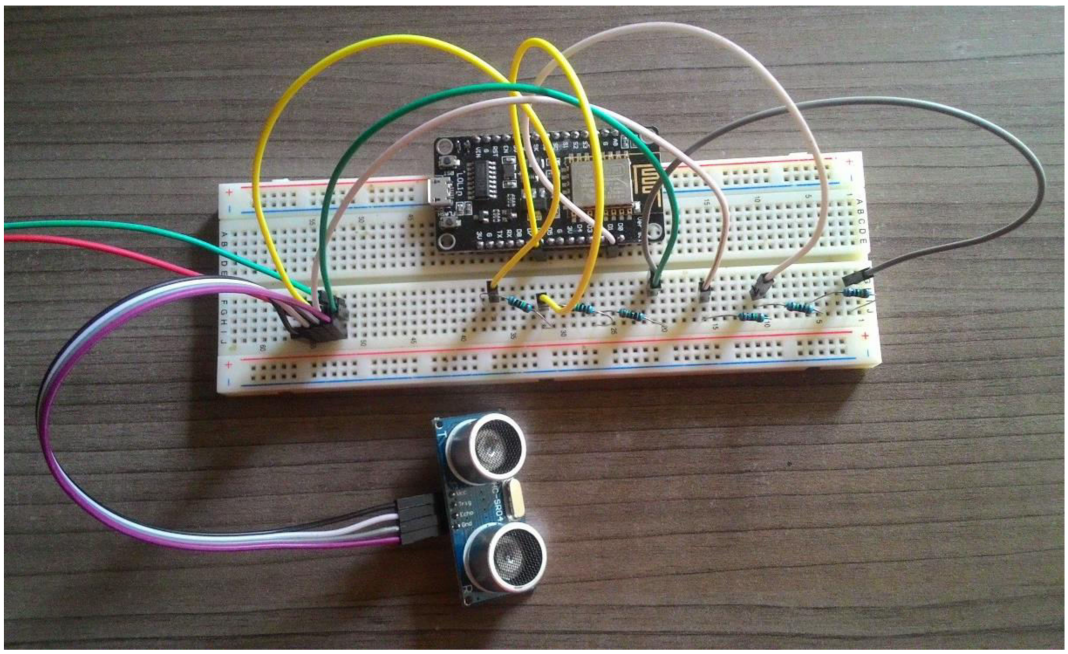


Fonte: Elaboração própria (2019)

Os pinos GPIO's utilizados do Módulo ESP8266 para controle do sensor ultrassônico foram: 05 e 16. Sendo o pino ECHO do sensor ultrassônico conectado à porta GPIO-05 do microcontrolador ESP8266, através dos *jumpers* brancos. E o pino TRIGER foi conectado à porta GPIO-16 com *jumpers* amarelo.

A Figura 26 demonstra a implementação desse circuito. Foram utilizados *jumpers* do tipo macho-macho para construção dos dois circuitos divisores de tensão e sua conexão com o Módulo ESP8266, e para alimentação dos componentes ativos. Já a conexão do módulo a *Protoboard* foi feita através de *jumpers* macho-fêmea.

Figura 26 – Implementação do circuito do projeto



Fonte: Autoria própria (2019)

3.2 Análise programação do Módulo ESP8266

A programação no Arduino IDE se estrutura fundamentalmente em duas funções: *setup ()* e *loop (),* como explicado na seção 2.10.2 .Na função *setup ()* inicializa-se as configurações dos pinos GPIO (*input* ou *output*), das variáveis globais, das bibliotecas etc. Essa função é executada uma única vez no código ao energizar-se o circuito ou quando acontece um *reset*. Em seguida ao *setup ()*, ocorre a função *loop ()*, que executa indefinidamente uma sequência de comandos determinada, ou até que aconteça algum comando de interrupção externo.

Na primeira parte do programa são definidos os parâmetros de inicialização do código, que são as inclusões das bibliotecas[8] FirebaseESP8266, ESP8266WiFi e Ticker, em seguida são declaradas as credenciais de autenticação junto a rede WiFi e ao banco de dados Firebase Real Time e definido o tempo de envio de dados, e por último definem-se as variáveis globais e seus tipos como pinos trigger (GPIO16) e echo (GPIO05), distância medida, volume, tempo e dimensões do tanque de combustível. A Figura 27 apresenta o trecho do código correspondente a essa inicialização.

[8] Essas bibliotecas estão disponíveis no Gerenciador de Bibliotecas do programa Arduino IDE.

Figura 27 – Inicialização das bibliotecas utilizadas

```
#include <FirebaseESP8266.h>
#include <ESP8266WiFi.h>
#include <Ticker.h>
FirebaseData firebaseData;
String path = "/Tanque";

const char* ssid = "ewway";
const char* password = "copa20182638";

#define FIREBASE_HOST "monitoramentocombustivel.firebaseio.com"
#define FIREBASE_AUTH "Ueg2FauUYoox0pu2R8sYugfm2Filr7yBGXPzw5kx"
#define PUBLISH_INTERVAL 1000*60*3
Ticker ticker;
bool publishNewState = true;

void publish(){
  publishNewState=true;
}

int echoPino = 05;
int trigPino = 16;
float tempo = 0;
float distancia = 0;
float altTanq = 100;
float compTanq = 115;
float largTanq = 95;
float altNivel = 0;
float volume = 0;
```

Fonte: Autoria própria (2019)

Posteriormente, o programa executa a função *setup ()*, nela realiza-se a configuração da taxa de transmissão de dados em bits por segundos para comunicação serial necessária a realização de testes através do Serial Monitor da Arduino IDE, como leitura dos valores medidos, conexão à rede WiFi ou banco de dados Firebase Real Time. Na função *setup ()* é dado início a conexão com a WiFi através da função WiFi.begin (ssid, password), contendo respectivamente o nome e a senha da WiFi. A conexão com o Firebase é realizada pela função Firebase.begin (FIREBASE_HOST, FIREBASE_AUTH), em que “FIREBASE_HOST” e “FIREBASE_AUTH” são o endereço *Web* e chave de autenticação do banco de dados, respectivamente.

Ainda dentro do *setup ()* é estabelecido uma rotina de verificação se a conexão com a rede WiFi e o banco de dados Firebase foram bem sucedidas. Em caso afirmativo, são apresentados no Serial Monitor a confirmação de acesso para ambos. Já em caso negativo, o programa entra em um *loop* de reconexão, finalizando apenas quando são reestabelecidas as conexões como mostra a Figura 28.

Figura 28 – Código da função *setup ()*

```
void setup()
{
 Serial.begin(115200);
 WiFi.begin(ssid, password);
 pinMode(echoPino, INPUT);
 pinMode(trigPino, OUTPUT);

 while (WiFi.status() != WL_CONNECTED){
 delay(500);
 Serial.println("Conectando ao WiFi...");
 }

 Serial.println("Endereço IP conexão:");
 Serial.println(WiFi.localIP());

 Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
 if(Firebase.failed())
   firebasereconecta();

 ticker.attach_ms(PUBLISH_INTERVAL, publish);
 delay(100);
}
```

Fonte: Autoria própria (2019)

Ao finalizar a execução da função *setup()*, o programa entra na função *loop(),* que será executada indefinidamente. De início é feito uma nova verificação de conectividade com o banco de dados Firebase, caso apresente problema o fluxo do programa é redirecionado para a função *firebasereconecta ()*, e o fluxo do programa só é retornado após reestabelecida comunicação com o banco de dados. Em seguida a essa verificação inicia-se todo processo de medição de distância entre o sensor e combustível, que segue o processo explicado anteriormente na seção 2.9, é valido observar que para uma leitura adequada dessa distância, o sensor HC-SR04 deve ser instalado, em relação ao nível máximo do combustível, em no mínimo dois 2 cm acima, pois em menos desse valor as leituras tornam-se imprecisas como especificado no *datasheet* do fabricante. Dessa maneira, fica claro que se deve então desconsiderar esse espaçamento entre sensor e a superfície do combustível da leitura realizada. Finalmente, com a leitura da distância se calcula o volume em cm^3, deve ser convertido para litros dividindo por 1000.

Após obtido o valor do volume, deve-se armazenar essa informação no banco de dados para posterior requisição de acesso pelo *App Web* desenvolvido para visualização do nível de combustível. O armazenamento acontece de maneira periodizada para que não haja um

excesso de armazenamento desnecessário. Isso é controlado pelo tempo atribuído a variável PUBLISH_INTERVAL e a função *publish ()*. Veja o trecho referente a *função loop ()* na Figura 29.

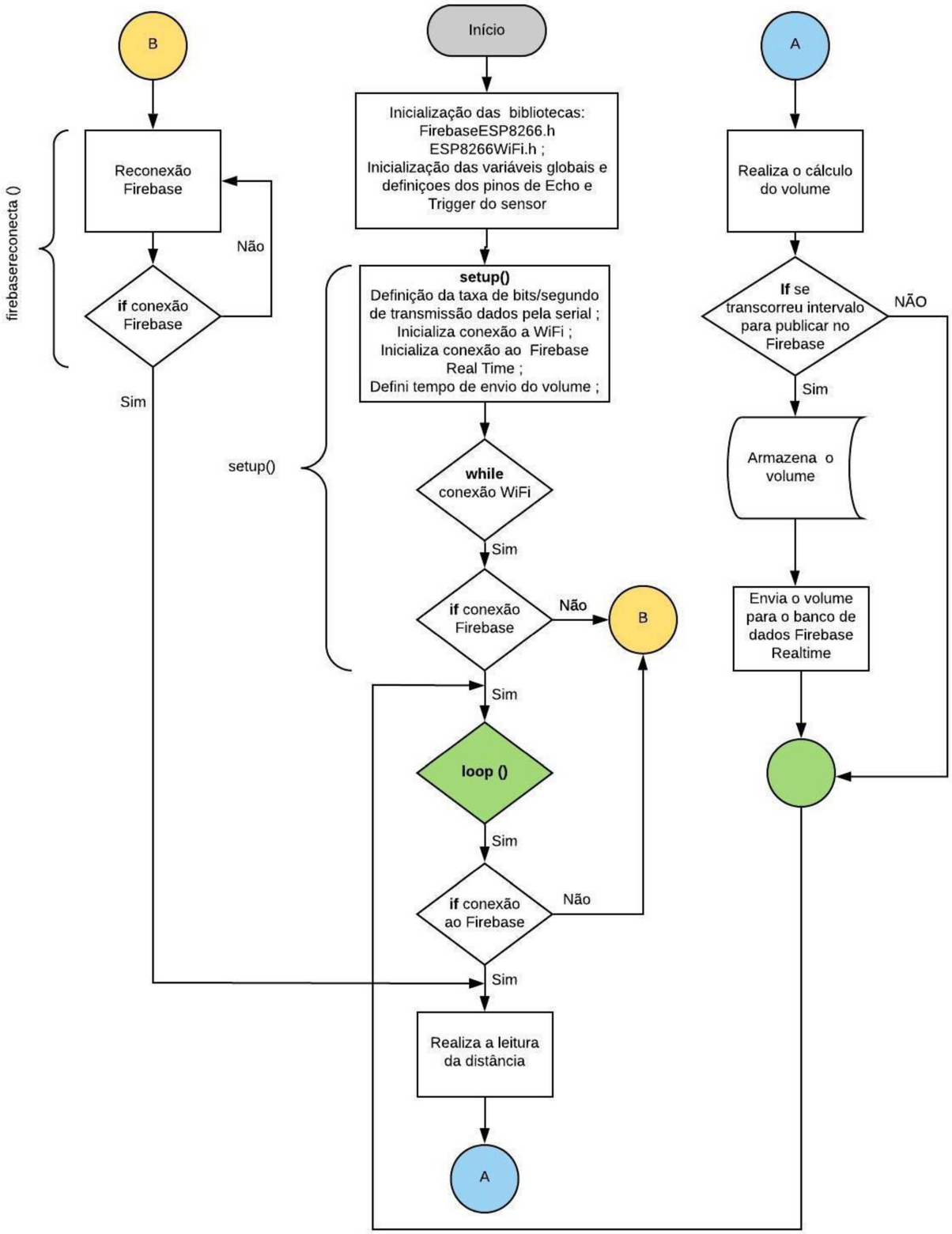
Figura 29 – Código da função *loop ()*

```
void loop(){
  if(Firebase.failed()){
     firebasereconecta();}

  DisparoPulsoUltrassonico();

  tempo = pulseIn(echoPino,HIGH);
  tempo = tempo/1000000;
  distancia = (tempo*340)/2;

  VolumeCalculo();

  Serial.print("Volume em litros: ");
  Serial.println(volume);

  if(publishNewState){
   Serial.println ("Novo enviado de dado...");
  if(!isnan(volume)){
   Firebase.pushFloat(firebaseData, path + "/Volume",volume);
   publishNewState = false;
  }
  else{
   Serial.println("Erro de envio de dado...");
  }
}
 delay(200);
}
```

Fonte: Autoria própria ()

Essa sequência de trechos apresentados do programa, podem ser resumidos através do fluxograma demonstrado pela Figura 30. O programa completo pode ser visto no apêndice A.

Figura 30 – Fluxograma do algoritmo do projeto

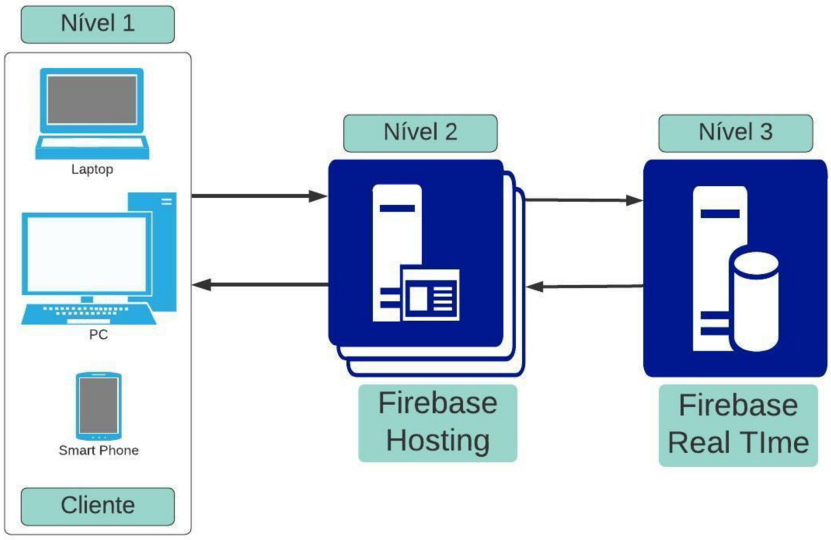


Fonte: Autoria própria (2019)

3.3 Software Desenvolvido

A arquitetura de software utilizada nessa aplicação é a cliente-servidor em três níveis, como mostra Figura 31 , sendo o primeiro nível o cliente ou solicitante de informação via *browser* (navegador) através do endereço URL da página *Web* criada. O segundo nível é o servidor *Web*, onde o a página *Web* está hospedado, no caso o Firebase Hosting. E por fim, o banco de dados Firebase Real Time, é o terceiro nível e disponibiliza os dados requisitados pelo cliente ao servidor *hosting* da aplicação.

Figura 31 – Arquitetura em três níveis cliente-servidor



Fonte: Autoria própria (2019)

Essa arquitetura tem por vantagem a descentralização das funções em computadores independentes em uma rede, o que facilita a execução de manutenção. Outra vantagem, relaciona-se à segurança oferecida pelos servidores da aplicação e de dados, pois controlam melhor o acesso a recursos.

3.3.1 Linguagem de programação

A linguagem de programação usada na criação da página *Web* para o projeto desenvolvido baseou-se em HTML, CSS e JavaScript. Essas são linguagens *open source* e têm comunidades grandes para suporte.

3.3.2 Google Charts

Uma maneira fácil e intuitiva de se apresentar a informação do volume medido de combustível diesel é através de gráficos, e para facilitar isso, foi usado a biblioteca de gráficos Google Charts, que é baseada em JavaScript. Essa biblioteca foi criada para melhorar os aplicativos da *Web*, adicionando recursos de gráficos interativos, fornecendo uma imensa variedade de gráficos, como gráficos de linhas, spline charts, gráficos de área, gráficos de barras, gráficos de pizza e assim por diante.

4 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados obtidos, após construção e integração dos elementos que dão corpo a este projeto, que foram: a montagem do circuito na *protoboard*, implementação do código em linguagem Arduino, configuração do banco de dados Firebase Real Time, confecção da página *Web* e sua hospedagem no banco de dado Firebase Hosting.

4.1 Protótipo Tanque Abastecimento

No intuito de validar o projeto desenvolvido, foi realizado uma simulação de medição de volume do tanque com um jarro plástico em formato cilíndrico para um volume de 2 litros, como visto na Figura 32. As dimensões do jarro são: altura 25,4 cm e raio de 7,4cm, contudo ele não é um cilindro perfeito, então para minimizar erros de medição foi considerado como base do recipiente a porção já preenchida de água na altura onde não há escorrimento de água pela torneira, ou seja, esse ponto foi considerado o "nível 0", e a partir dele foi definido o "nível máximo" de 2 litros na altura de 11,62 cm. O sensor ultrassônico foi instalado no topo da jarra, por meio da adaptação de dois furos na tampa da jarra. Dessa maneira, foi implementado no código do programa, o cálculo de volume para cilindro com as dimensões estipuladas anteriormente. Os testes consistiram em descarregar a água a cada 200ml.

Tabela 3 – Medições volume real e medido

Vol. Real (ml)	Vol. Medido (ml)	Erro (%)
2000	1975,32	1,23
1800	1773,50	1,47
1600	1653,60	3,35
1400	1393,22	0,48
1200	1194,31	0,47
1000	1015,88	1,60
800	890,10	11,26
600	708,74	18,12
400	439,63	9,90
200	255,35	27,67
0,00	53,51	–

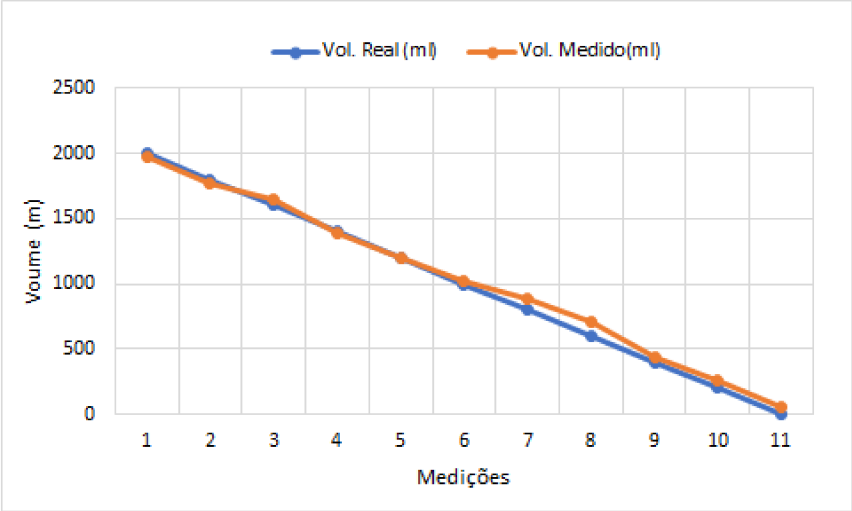
Fonte: Autoria própria (2019)

Figura 32 – Simulação tanque abastecimento

Fonte: Autoria própria (2019)

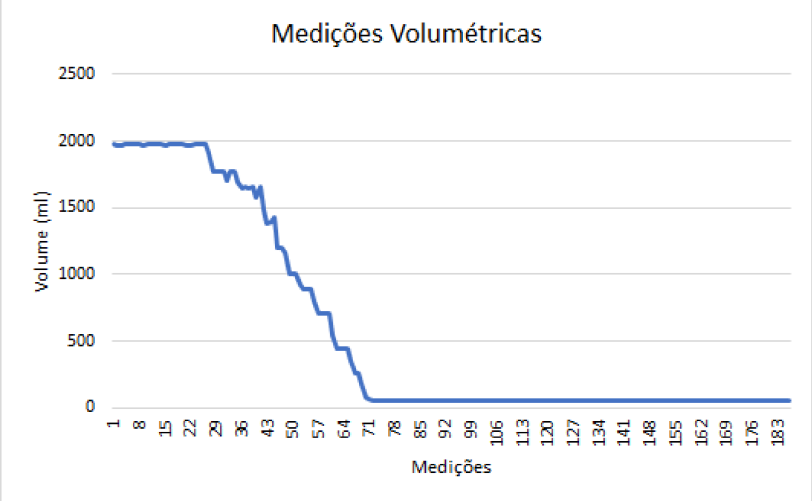
Nos resultados apresentados na Tabela 3, percebe-se que os resultados são bastante satisfatórios, com erros menores a 2%, e corresponde à metade do volume aferido. A partir da sétima medição o percentual de erro aumenta e oscila consideravelmente próximo aos 10%, causas prováveis para tais erros podem estar relacionadas as dimensões reduzidas do tanque utilizado e ao ângulo de detecção do sensor. No entanto, ao propósito final do projeto a precisão apresentada é considerada satisfatória. A Figura 33 apresenta os dados da Tabela 4, na forma de gráfico de linha para facilitar a análise de precisão do sensor ultrassônico empregado e Figura 34 gera uma visualização mais intuitiva do decaimento do fluido, como será usado na aplicação *Web*.

Figura 33 - Gráfico comparativo entre as medidas real e aferidas



Fonte: Autoria própria (2019)

Figura 34– Gráfico volume por amostra



Fonte: Autoria própria (2019)

É valido salientar que o sensor ultrassônico não deve ser instalado no limite de nível máximo do tanque, pois isso vai implicar em leituras incorretas, haja visto que segundo o *datasheet* do fabricante as distâncias aferidas são válidas a partir de 2 cm a 4m. Assim, deve-se desconsiderar da distância aferida a distância do ponto de fixação do sensor até o ponto considerado o limite de nível máximo do combustível, pode-se chamar essa diferença de distância de *offset* (PONGRACZ, 2015).

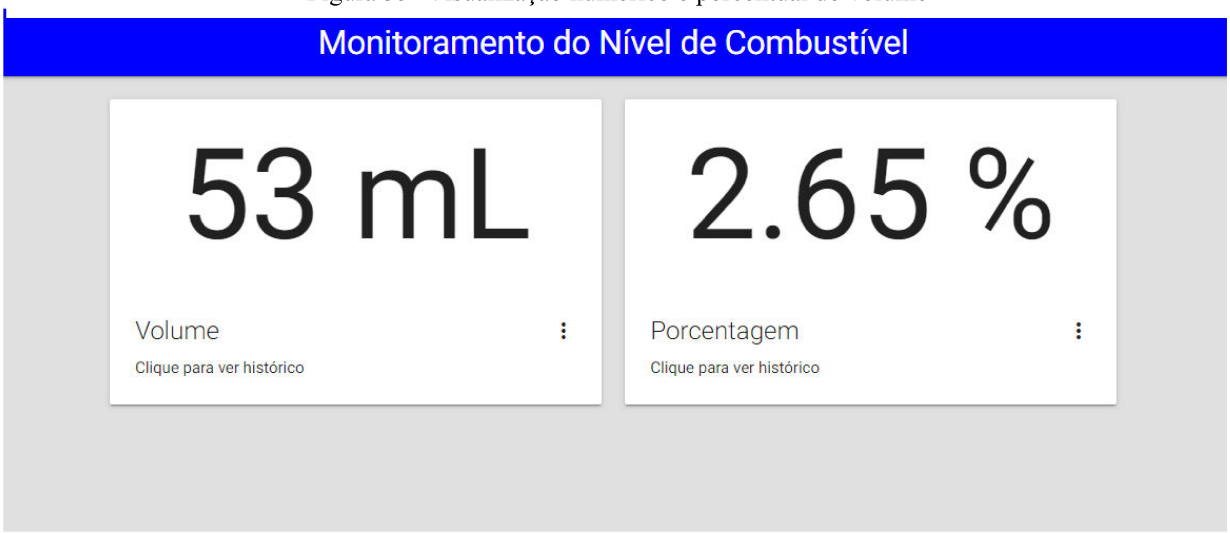
4.2 Visualização Nível Combustível

O acompanhamento do nível de combustível em tempo real através da página *Web* desenvolvida é o elemento central do projeto proposto. É nela onde é possível verificar de qualquer local com acesso à internet aferir o nível de combustível em momentos de falta de energia elétrica e/ou nos dias programados para acionamento do grupo gerador. O que gera comodidade ao evitar o deslocamento de equipe nos finais de semana ou durante as madrugadas para realizar averiguações da capacidade de suprimento do diesel.

A aplicação *Web* que está hospeda no Firebase Hosting (servidor *Web*, como explicado anteriormente) é responsável por buscar as informações armazenadas no banco de dados Firebase Real Time em formato de chave-valor (*JSON*), enviados pelo módulo ESP8266 a cada 3 minutos, e plotar na forma linear de gráfico interativo a cada amostra obtida pelo Google Charts, em que posicionando-se o *mouse* sobre gráfico é possível visualizar o volume daquela medição. Realizou-se no código fonte do aplicativo *Web* uma regra de três para apresentação da porcentagem em relação a última leitura.

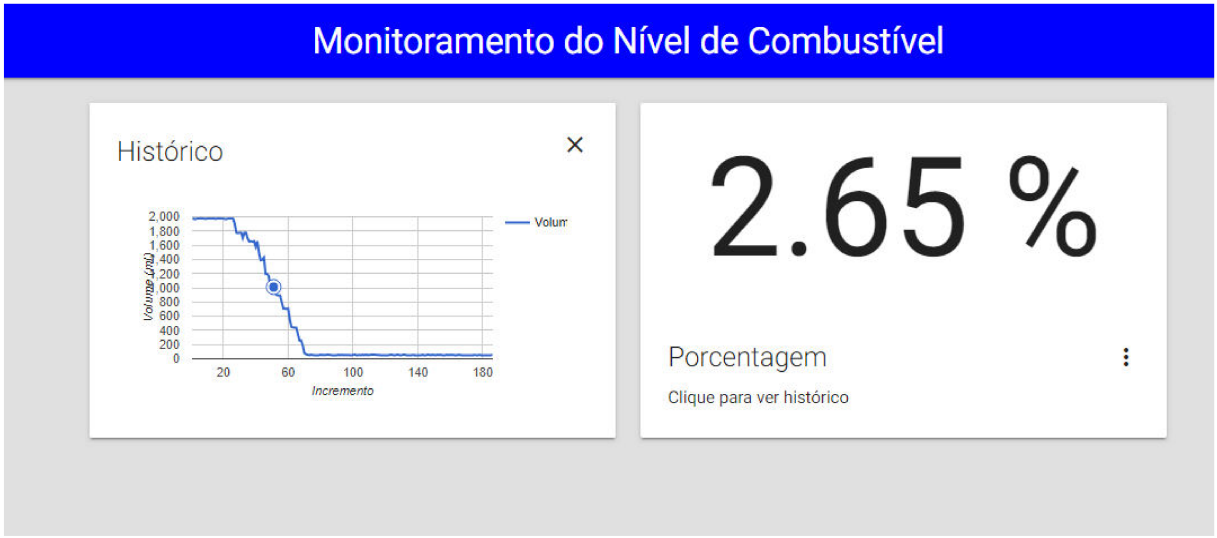
O volume aferido é disponibilizado assim que se tem acesso a página *Web* de monitoramento, essa informação é apresentada numericamente, como mostra Figura 35 . O histórico pode ser mostrado clicando-se no ícone “histórico” e porcentagem do volume monitorado é mostrada o tempo todo na janela a direita, Figura 36.

Figura 35 -Visualização numérico e porcentual do volume



Fonte: Próprio autor (2019)

Figura 36- Visualização histórico e percentual do volume



Fonte: Autoria própria (2019)

5 CONCLUSÃO

O projeto desenvolvido implementa mais uma facilidade ao monitoramento do ambiente do *data center* da UFT, ao se aferir em tempo real o nível de combustível de abastecimento do grupo gerador. Tal funcionalidade, confere maior segurança e confiabilidade aos serviços de tecnologia da informação prestados à comunidade da Universidade, principalmente em momentos de falhas de fornecimento de energia elétrica pela concessionária pública de energia. Os objetivos propostos foram alcançados de forma satisfatórias, em que o acompanhamento do nível de combustível pode apresentado em forma numérica e gráfica através da página *Web* criada.

Na construção deste protótipo, deu-se preferência a componentes de baixo custo e *open source* que se integrassem sem maiores dificuldades, o que aconteceu com o microcontrolador NodeMCU ESP8266/ ESP 12-E e o Google Firebase. Os conhecimentos exigidos foram nas áreas da programação em linguagem Arduino, HTML, CSS e Java Script, circuitos digitais, banco de dados, bem como estudos sobre os tipos de medição de nível de combustível para escolha da tecnologia de medição adequada a este caso específico. Todo o trabalho se deu em três etapas principais: estudo dos objetos que compõem a solução e escolha das ferramentas de construção, e implementação do hardware e software.

O estudo dos objetos do ambiente de implementação e escolha dos componentes de construção, serviu para despertar as especificidades deste trabalho. Na etapa de implementação do hardware, ocorreu a montagem do circuito e elaborado a programação do microcontrolador e sua integração ao banco de dados Google Firebase. Já no software, realizou-se a programação do *front end* (parte de visualização) da aplicação, em que será acompanhado o nível de combustível.

Com respeito a propostas de trabalhos futuros, uma melhoria possível seria a criação de um sistema de alerta definido para o caso do nível de combustível do tanque chegue a um nível crítico, então ocorreriam o envio de e-mail e mensagem SMS via celular, indicando a necessidade de reabastecimento do tanque.

REFERÊNCIAS

4LINUX OPEN SOFTWARE SPECIALIST. **O que é Middleware**. [S.I]. Disponível em: <https://www.4linux.com.br/o-que-e-middleware> Acesso em: 18.jun. 2019.

ABRANTES, Sílvio.; PINTO, Vasco. **Como evoluíram as normas Wi-Fi IEE 802.11?** Out. 2013. Disponível em: <https://paginas.fe.up.pt/~projfeup/submit_13_14/uploads/relat_1MIEEC01_3.pdf>. Acesso em: 15 mai. 2019.

ALECRIM, Emerson. **Bancos de dados são mais importantes nas nossas vidas do que a gente imagina.** [S.I]. TECNOBLOG, 2018. Disponível em: < https://tecnoblog.net/245120/banco-de-dados-importancia/>. Acesso em: 17 jun. 2019.

ALFACOMP. **Transmissores e sensores utilizados na telemetria do saneamento**. [S.I] [2019]. <https://alfacompbrasil.com/2019/05/31/transmissores-e-sensores/>. Acesso em: 15 out. 2019.

_____. **Transmissor de nível hidrostático – TNH20**. [S.I] [2019]. <https://www.alfacomp.ind.br/sensores/transmissor-ultrassonico-de-nivel-tun20>. Acesso em: 16 nov. 2019.

ALVES, Gustavo Furtado de Oliveira**. O mínimo que você precisa saber sobre JSON para ser um bom programador!**.[S.I] 2018. Disponível em:< https://dicasdeprogramacao.com.br/o-que-e-json/>. Acesso em: 20 out. 2019.

AMAZON AWS. **O que é a computação em nuvem?** [S.I] Amazon AWS, 2019a. Disponível em: <https://aws.amazon.com/pt/what-is-cloud-computing/?nc1=f_cc> Acesso em: 6 jun. 2019.

AMAZON. **O que é computação em nuvem**. [S. I]. AWS, 2019. Disponível em: <https://aws.amazon.com/pt/what-is-cloud-computing/>. Acesso em: 18.jun. 2019.

_____. **O que é banco de dados relacional?** [S.I] Amazon AWS, 2019b. Disponível em: <https://aws.amazon.com/pt/relational-database/>. Acesso em: 6 jun. 2019.

_____. **O que é o NoSQL?** [S.I] Amazon AWS, 2019c. Disponível em: <https://aws.amazon.com/pt/nosql/>. Acesso em: 6 jun. 2018.

ARDUINO. **Arduino – Introduction**. Arduino, [S.I] [2018?]. Disponível em: <https://www.arduino.cc/en/Guide/Introduction>. Acesso em: 2 out. 2019.

BAPTISTA, João P. **Microcontroladores**. PORTO: ISEP, 2001. Disponível em: <http://www.lsa.isep.ipp.pt/~aprender/pagina_lsa/aprender/datasheets/Micros.pdf>. Acesso em: 17 jun. 2019.

BASTOS, Alex Vidigal. **Microcontroladores – Eng. Computação**. [S.I][2013]. Disponível em: <http://www.decom.ufop.br/alex/arquivos/sof_bas_EC/Microcontroladores.pdf>. Acesso em: 16 out. 2019.

BATSCHINSKI, G. **O que é um Backend as a Service?** Back4App Blog, 2016. Disponível em: <https://blog.back4app.com/2016/07/11/o-que-e-backend-as-a-service/>. Acesso: em 6 maio 2019.

CASSIOLATO, César. **Medição de pressão: tudo o que você precisa saber**. [S.I] [2018]< http://www.profibus.org.br/images/arquivo/pdf-1-543ebf8b1e0a2.pdf>. Acesso em: 15 out. 2019.

CHAPMAN, Stephen J. **Fundamentos de Máquinas Elétricas**. 5. ed. Porto Alegre: Amgh, 2013.

COELHO, João Paulo. **Sensores e Actuadores**. 2ª ed. BRAGANÇA: IPB - ESTIG, 2004. Disponível em: < http://www.ipb.pt/~jpcoelho/downloads/SeA_.pdf> Acesso em: 8 de out. 2019.

DUNN, William C. **Fundamental of Industrial Instrumentation and Process Control.** [S.I]. McGraw-Hill, 2005.

ECMA INTERNATIONAL. **The JSON Data Interchange Syntas**.Geneva, 2017. Disponível em: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf. Acesso em 20 out. 2019.

EIS, Diego; FERREIRA, Elcio. **HTML5 e CSS3 com farinha e pimenta**. São Paulo, 2012. Disponível em: < http://www.greyhouse.com.br/wp-content/uploads/2014/07/HTML5-e-CSS3.pdf>. Acesso: 15 dez. 2019.

ELECFREAKS. **Utrassonic Ranging Module HC –SR04**. [S. I] [2011]. Disponível em: < https://img.filipeflop.com/files/download/Datasheet_HCSR04.pdf>. Acesso: 16 out. 2019.

ESPRESSIF SYSTEMS. **ESP8266EX Datasheet.** Espressif Systems, 2018. Disponível em:<https://www.espressif.com/sites/default/files/documentation/0aesp8266ex_datasheet_en.pdf>. Acesso em: 28 abr. 2018.

EVANS, Dave.**A Internet das Coisas Como a próxima evolução da Internet está mudando tudo**. [S.I] 2011.CISCO. Disponível em : <https://pdfs.semanticscholar.org/a1a5/918a972fa6172f9e06c429a8bd74b3f9b8d3.pdf>. Acesso em: 20 jun. 2019.

FABLAB. **Projeto Smart Parking**.[S. I] 2018. Disponível em: <http://fabmanager.facens.br/#!/projects/smart-parking.>. Acesso em 22 set. 2019.

FIREBASE**. Firebase helps mobile and web app teams succeed**. Disponível em: <https://firebase.google.com/?hl=pt-BR>. Acesso em: 19 jun. 2019.

FIRJAN. **Industria 4.0: Internet das Coisas**. 2016. Disponível em: <https://www.firjan.com.br/lumis/portal/file/fileDownload.jsp?fileId=2C908A8A555B47FF01557E033FAC372E&inline=1.>. Acessado em 27 jun. 2019.

GAMA, Alexandre. **O que é JSON**. São Paulo: Devmedia, 2011. Disponível em: <https://www.devmedia.com.br/o-que-e-json/23166>. Acesso em: 13 out. 2019.

GONÇALVES, Marcelo Giglio. **Monitoramento e controle de processos:** Série Qualificação Básica de Operadores. RIO DE JANEIRO/BRSÍLIA: Petrobras/Senai-DN, 2003.

GOOGLE DEVELOPERS ,2019. Disponível em :< https://developers.google.com/iot/>. Acesso em: 25 mai. 2019.

INSTRUMENÇÃO E CONTROLE. **Guia de Medição de Nível em Silos e Tanques Industriais**. [S.I] [201?]. Disponível em:<https://instrumentacaoecontrole.com.br/>Acesso em: 10 out.2019.

INSTRUMENÇÃO E CONTROLE. **Sensor de Nível – qual é o melhor para o processo?** [S.I] [2019]. Disponível em:<https://instrumentacaoecontrole.com.br/sensor-de-nivel-qual-e-o-melhor-para-o-seu-processo/>Acesso em: 10 out.2019.

INSTRUMENTAÇÃO E CONTROLE**. Medição de Nível Contínuo, e agora?** [S.I] [2017]. Disponível em:<https://instrumentacaoecontrole.com.br/instrumentacao-industrial-medicao-de-nivel-continuo-e-agora/>Acesso em: 10 out.2019.

INTEL. **Noções básicas sobre IEEE * 802,11 autenticação e Associação**. [S.I] 2019. Disponível em: <https://www.intel.com.br/content/www/br/pt/support/articles/000006508/network-and-io/wireless-networking.html>. Acesso em: 26 jun. 2019.

LEMOS, Manoel. **Conheça os shields e incremente seu Arduino com eles**. [S.I] 2013. Disponível em < https://blog.fazedores.com/conheca-os-shields-e-incremente-seu-arduino-com-eles/ >. Acesso em: 17 out. 2019.

MAGRANI, Eduardo. Tecnologia, inovação e internet das coisas (IoT).In: _____ **A internet das coisas** 1º ed. Rio de Janeiro : FGV Editora, 2018, p. 20-59. Disponível em: < https://bibliotecadigital.fgv.br/dspace/bitstream/handle/10438/23898/A%20internet%20das%20coisas.pdf?sequence=1&isAllowed=y>. Acesso em: 13 jun. 2019.

MANCINI, Mônica. **Internet das Coisas: História, Conceitos, Aplicações e Desafios**. Universidade Presbiteriana Mackenzie. 29 jun. 2018. Disponível em: <https://www.researchgate.net/publication/326065859_Internet_das_Coisas_Historia_Conceitos_Aplicacoes_e_Desafios>. Acesso em: 20 out. 2019.

MARGOLIS, Michael. **Arduino Cookbook**: Recipes to Begin, Expand, and Enhance Your Projects. O'Reilly Media, Inc.", 2011.

MOTA, Allan. **HC-SR04 – Sensor Ultrassônico de distância com Arduino**. [S. I] 2018? Disponível em:< https://portal.vidadesilicio.com.br/hc-sr04-sensor-ultrassonico/>. Acesso: 16 out. 2019.

NETSOT. **Protocolos de Segurança de Rede Sem Fio: WEP, WPA, WPA2 e WPA3**. [S.I] 2019. Disponível em: <https://www.netspotapp.com/pt/wifi-encryption-and-security.html>. Acesso em: 26 jun. 2019.

OLANDI, Claudio. **Firebase: serviços, vantagens, quando utilizar e integrações**. Rocketseat Blog.Disponível em:<https://blog.rocketseat.com.br/firebase/>. Acesso em: 19 jun. 2019.

OLIVEIRA, Gutemberg P. **Automação residencial com o módulo esp8266 e aplicativo android.** TCC - Curso de Engenharia Elétrica - UFT, Palmas, 2018.

PANSANATO, L. T. E. **Redes locais de computadores.** Curitiba. Editora, UFTPR, 2016.

PATTERSON, David A.; HENNESSY, John L.; **Organização e projeto de computadores:** a interface hardware/software. 3. ed. Rio de Janeiro: Elsevier, 2005.

PIRES, Paulo F. et al. **Capítulo 3 – Plataformas para a Internet da Coisas**. *In*: XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos.Vitória, 2015. Disponível em: <http://sbrc2015.ufes.br/wp-content/uploads/Ch3.pdf>. Acesso em: 16 jun. 2019.

SANTOS, Bruno P. e t al. **Capítulo 1 - Internet das Coisas: da Teoria à Prática.** Departamento de Ciência da Computação-UFMG. 2016, Belo Horizonte. Disponível em: <https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf>. Acessado em: 16 jun. 2019.

STANGARLIN, Douglas P. **Análise de Desempenho de Redes Sem Fio com Diferentes Protocolos de Criptografia: Um Estudo de Caso**. CTISM- TECNOLOGIA EM REDES DE COMPUTADORES: Santa Maria, 2014. Disponível em:

<http://www.redes.ufsm.br/docs/tccs/Douglas_Pegoraro_Stangarlin.pdf>. Acesso em: 10 jun. 2019.

THOMAZINI, Daniel; ALBUQUERQUE, Pedro Urbano Braga. **Sensores Industriais: fundamentos e aplicações**. 8ª ed. SÃO PAULO: Érica, 2011.

THOMSEN, Adilson. **Como comunicar com o Arduino Ethernet Shield W5100**.[S.I] 2004. Disponível em: < https://www.filipeflop.com/blog/tutorial-ethernet-shield-w5100/>. Acesso em: 15 out. 2019.

TOCCI, R. J.; WIDMER, N. S.; MOSS, G. L. Computadores Digitais. In: _____ **Sistemas Digitais:** Princípios e Aplicações. 10. ed. São Paulo: Pearson Prentice Hall, 2007. p. 18 – 19.

WIZNET. **About Wiznet**.[S.I] 2016.Disponível em:< https://www.wiznet.io/about/about-wiznet/ >. Acesso em: 16 out. 2019.

WIZNET. **W5100S Datashee**.[S.I] 2018.Disponível em: <https://wizwiki.net/wiki/lib/exe/fetch.php?media=products:w5100s:w5100s_ds_v125e.pdf>. Acesso em: 16 out. 2019.

ZUIM, Edgar. **Arduino -Sensor Ultrassônico**. [S. I] 2016.Disponível em:<https://www.ezuim.com/arduino/arduin03.pdf>. Acesso: 16 out. 2019.

APÊNDICE A

```
#include <FirebaseArduino.h>
#include <ESP8266WiFi.h>
#include <Ticker.h>
const char* wifi_ssid = "LABORATORIO";       //entrar com nome da rede de conexão WiFi
const char* wifi_password = "20microondas16";    //entrar com a senha da rede WiFi
#define FIREBASE_HOST "monitoramentocombustivel.firebaseio.com"
#define FIREBASE_AUTH "Yeg2FauUYoox0pu2K8sYuvfm2Filr7yBGXPzw8kx"
#define PUBLISH_INTERVAL 1000*60*3      // Tempo de publicação no banco de dados
Ticker ticker;
bool publishNewState = true;
void publish()
{
  publishNewState=true;
}
int echoPino = 5;                    // D1, Pino 12 recebe o pulso do echo
int trigPino = 16;                   // D6, Pino 13 envia o pulso para gerar o echo
int x=0;
float tempo = 0;
float distancia = 0;
float altTanq = 11.62;                 //altura do tanque em cm
float Abase = 172.034;                 //base do tanque em cm2
float altNivel = 0;
float volume = 0;
void setup()
{
 Serial.begin(115200);                           // Inicia comunicação serial
 WiFi.begin(wifi_ssid, wifi_password);
 pinMode(echoPino, INPUT);                       // Define o pino 12 como entrada (recebe)
 pinMode(trigPino, OUTPUT);                      // Define o pino 13 como saida (envia)
 while (WiFi.status() != WL_CONNECTED) {          // Inicia loop de conexão com a WiFi
 delay(500);                                   // Aguarda meio segundo para uma nova tentativa
 Serial.println("Connecting to WiFi...");     // Informa conexão pelo serial monitor
```

```
 }
 Serial.println("Connected to the WiFi network:");
 Serial.println(WiFi.localIP());
 Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);          // Incializa conexão Firebase
 if (Firebase.failed())                          // Passa para a função loop de reconexão ao Firebase
 {
  FirebaseReconnect();
 }
ticker.attach_ms(PUBLISH_INTERVAL, publish);                                  // Registra o ticker
delay(100);
}
void FirebaseReconnect()                        // Loop para reconexão ao banco de Dados Firebase
{
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  if (Firebase.failed())
  {
   FirebaseReconnect();
  }
}
void DisparoPulsoUltrassonico()                       // Função para leitura de distância pelo sensor
ultrassônico
{
  digitalWrite(trigPino, LOW);      // Pino trigger inicia em nível LOW por 10 microssegundos
  delayMicroseconds(10);                 // Delay de 10 microssegundos
  digitalWrite(trigPino, HIGH);          // Pino trigger com pulso HIGH por 10 microssegundos
  delayMicroseconds(10);                 // Delay de 10 microssegundos
  digitalWrite(trigPino, LOW);           // Pino trigger colocado em nível LOW, novamente
}
void loop(){
if(Firebase.failed())          // Testa se vai para o Loop de reconexão ao banco de Dados
Firebase
 {
  FirebaseReconnect();
 }
```

```
if(publishNewState){
  DisparoPulsoUltrassonico();
  tempo = pulseIn(echoPino,HIGH);
  tempo = tempo/1000000;                  // Converte o tempo de microssegundos para segundos
  distancia = (tempo*340.3)/2;            // Calcula a distância com base na velocidade do som
  distancia = (distancia*100) - 8.36;     // Calculo da altura real do volume
  altNivel = altTanq - distancia ;        // Calcula o altura do nível
  volume = altNivel * Abase;              // Calcula o volume do tanque
  Serial.print("Volume em ml: ");
  Serial.println(volume);
  {
   Serial.println ("Publish New State");
  }
  if(!isnan(volume))                       // publica depois de ocorrido o tempo definido acima
  {
   Firebase.pushFloat("Volume",volume);
   publishNewState = false;
  }
  else {
  Serial.println("Error Publishing");
  }
 }
  delay (200);
 }                                         // retorna início loop ()
```