Ministério da Educação
Universidade Federal de Ouro Preto - UFOP
Escola de Minas
Colegiado do Curso de Engenharia de Controle e Automação - CECAU

MONITORAMENTO DO CONSUMO DE ENERGIA ELÉTRICA E CONTROLE DE EQUIPAMENTOS VIA APLICATIVO

MONOGRAFIA DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

LUIZ HENRIQUE JUNIOR PEREIRA

Ouro Preto - MG
Fevereiro de 2018

LUIZ HENRIQUE JÚNIOR PEREIRA

MONITORAMENTO DO CONSUMO DE ENERGIA ELÉTRICA E CONTROLE DE EQUIPAMENTOS VIA APLICATIVO

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção de Grau de Engenheiro de Controle e Automação.

Orientadora: Prof(a). Msc Adrielle de Carvalho Santana.

Ouro Preto - MG
Fevereiro / 2018

P4361 Pereira, Luiz Henrique Júnior.
Monitoramento do Consumo de Energia Elétrica e Controle de Equipamentos Via Aplicativo [manuscrito] / Luiz Henrique Júnior Pereira. - 2018.

60f.: il.: color.

Orientadora: Profª. MScª. Adrielle de Carvalho Santana.

Monografia (Graduação). Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais.

1. Internet das Coisas (IoT). 2. Energia elétrica - Consumo. 3. Automação Residencial. 4. Acesso Remoto. I. Santana, Adrielle de Carvalho. II. Universidade Federal de Ouro Preto. III. Titulo.

CDU: 681.5

Catalogação: ficha@sisbin.ufop.br

Monografia defendida e aprovada, em 21 de fevereiro de 2018, pela comissão avaliadora constituída pelos professores:

Profa. M.Sc. Adrielle de Carvalho Santana - Orientadora

Prof. Dr. Luiz Fernando Rispoli Alves – Professor Convidado

Prof. M. Sc. João Carlos Vilela de Castro – Professor Convidado

Agradecimento

À minha mãe Ângela por todos os momentos dedicados, pelas palavras e conselhos, por seu amor, honestidade e simplicidade. Sem você este sonho não seria possível.

Aos meus irmãos André e Bruna, obrigado por partilharem as pequenas coisas da vida.

À minha avó Guiomar, minha segunda mãe, muito obrigado.

À Luísa por todo incentivo e carinho!

À Universidade federal de Ouro Preto e a todos os mestres, obrigado pelo ensinamento compartilhado. Aos professores do DECAT, em especial à Adrielle pela orientação, o meu muito obrigado.

E por fim, obrigado à grandiosa República kamikaze, meu lar, por me acolher durante toda essa jornada.

"A ciência é, portanto, uma perversão de si mesma, a menos que tenha como fim último, melhorar a humanidade"

Nikola Tesla

Resumo

O presente projeto descreve o desenvolvimento de uma aplicação para *Internet of Things* - (*IoT*), que envolve o desenvolvimento de um módulo protótipo, que tem por finalidade a medição da corrente elétrica que flui para um determinado equipamento o qual é ligado na rede elétrica por intermédio de tal módulo. A partir do valor dessa corrente pode-se então calcular,aproximadamente, a potência consumida. Esse valor é então enviado para um banco de dados formado por planilhas do *Google Docs*. Com o módulo ainda é possível realizar o acionamento remoto de um ou mais equipamentos cuja conexão com a rede elétrica ele intermedia. Esse acionamento pode ser feito por acesso a computadores ou por uma aplicação desenvolvida por meio da plataforma para *smatphones MQTT Dash*. Tal módulo é baseado nas funcionalidades do microcontrolador *ESP8266* o qual está presente no hardware de circuitos integrados *NodeMCU* que foi utilizado. A medição da corrente é feita com o uso do sensor *ACS712* e o acionamento por intermédio de um módulo relé. Após o implemento do módulo, a sua utilização obteve os resultados esperados com a medição da energia consumida e o controle remoto do equipamento ligado a ele, tais resultados possiblitam o seu emprego em diversos trabalhos futuros aos quais se tenha interesse no acionamento e monitoramento de equipamentos via rede *Wifi*.

PALAVRAS-CHAVE: Internet das Coisas (*IoT*), monitoramento de energia elétrica, *NodeMCU*, automação residencial,acesso remoto.

Abstract

This project describes the development of an application for *Internet Of Things – IoT*, which involves the development of a prototype module which has as purpose the measurement of the electric current that flows to a certain equipment which is connected in the electric network by means of such module. From the value of this current one can then calculate, approximately,the power consumed. This value is then sent to a database consisting of Docs Spreadsheets. With the module it is still possible to carry out the remote of one or more equipment whose connection to the electrical network it inter- mediates. This activation can be done by access to computers or by an application developed through the platform for *smartphones* MQTT Dash. This module is based on the functionalities of the microcontroller ESP8266 which is present in integrated circuit hardware *NodeMCU* that was used. The current measurement will be made using the sensor ACS712 and the drive via a relay module. After the implementation of the module its use achieved the expected results with the measurement of the electric energy consumed and the remote control of the equipment connected to it. Such results make itThese results make possible their use in several future works that are interested in the activation and monitoring of equipments with *Wifi* network.

KEY-WORDS: *Internet of Things*, Electric power monitoring, *NodeMCU*, Home automation, Remote access.

Lista de Abreviaturas e Siglas

ABESCO Associação Brasileira das Empresas de Serviços de Conservação de Energia

ANATEL Agência Nacional de Telecomunicações

CECAU Colegiado do Curso de Engenharia de Controle e Automação

IBGE Instituto Brasileiro de Geografia e Estatística

IBM International Business Machines

IDE Interface Development

IOT Internet of Things

MQTT Message Queue Telemetry Transport

SCADA Supervisory Control and Data Acquisition

UFOP Universidade Federal de Ouro Preto

UFRGS Universidade Federal do Rio Grande do Sul

Sumário

Lista de Figuras

1 - INTRODUÇÃO

Este capítulo apresenta uma contextualização acerca da importância da energia para o desenvolvimento da sociedade humana. Realizando um breve comentário das fontes energéticas durante algumas eras da humanidade, até chegar-se à energia elétrica, uma das principais fontes atuais e responsável por inúmeras mudanças impactantes no modo de vida da sociedade.

Falar-se-á também de como é a utilização da energia elétrica no Brasil. Enfatizando desperdícios que têm provocado um prejuízo de bilhões de reais.

Por fim, apresenta-se algumas tecnologias, que também podem ser tratadas como fruto desse desenvolvimento proveniente da energia elétrica. Tais tecnologias são utilizadas na construção de um sistema, que tem por finalidade, permitir o acionamento e a visualização do consumo individual de algum equipamento elétrico.

1.1 Energia e sociedade

1.1.1 Contextualização

Sob um olhar termodinâmico rigoroso deve-se ver a energia, com seu uso geral, qualidade, intensidade e conversão energética como o fator crucial na história da humanidade. Fluxos e conversões da energia sustentam e delimitam a vida de todos os organismos e superorganismos, como civilizações e sociedades. Dentro dessa abordagem energética, divide-se a evolução da espécie humana em distintas eras de energia (SMIL, 2004).

A primeira era teve início há mais de 300 mil anos atrás, quando o *Homo sapiens* se diferenciou do *Homo erectus*, prevalecendo até o início das sociedades estabelecidas há 10 mil anos. Neste período todos os esforços para controlar maiores fluxos de energia limitaram-se pelo poder do metabolismo, uso ineficiente do fogo e o início da domesticação animal. Na segunda transi-

ção, sendo essa não tão universal, as sociedades tradicionais substituíram grandes partes dos esforços musculares por rodas de água e moinhos de vento. Há poucos séculos atrás em alguns países europeus aconteceu a terceira mudança, caracterizada pela substituição dos equipamentos motrizes movidos por biomassa, pelos motores cuja fonte de energia provinha da utilização de combustíveis fósseis. A última transição teve início em 1882, estendendo-se até os dias atuais sendo marcada pela afirmação da energia elétrica para consumo em massa e construção das primeiras estações produtoras(SMIL, 2004).

Um dos principais fatores para o desenvolvimento das nações trata-se da disponibilidade de energia elétrica, sendo assim, ela torna-se uma variável estratégica do crescimento pretendido. Obviamente, como resultado de tal progresso econômico, gerarão implicações no sistema de produção dessa energia. Sendo ratificada pelo fato desse crescimento abranger todas as fontes de consumo, desde final individual e coletivo, além do importante papel como fator de produção (BORENSTEIN et al, 1999).

A energia elétrica é de extrema importância para a sociedade, por este motivo, o seu consumo racional é fundamental, contribuindo dessa forma para a preservação ambiental. Desperdiçar essa energia provoca impreterivelmente no aumento da potência instalada de geração. Tal aumento simboliza num elevado custo ambiental como também em investimentos voltados para equipamentos (MARTINS et al, 1999).

1.1.2 Panorama de consumo energético do brasileiro

O desperdício da energia elétrica no Brasil atinge anualmente a cota de 50 mil gigawatts/hora que são equivalentes a 12,6 bilhões de reais, valor estimado à partir da tarifa cobrada no ano de 2014. Essa ineficiência acontece pelo uso de equipamentos que consomem uma maior potência ou numa situação de desperdício, segundo o presidente da Associação Brasileira das Empresas de Serviços de Conservação de Energia, ABESCO. Num cenário próximo ao ideal, sem desperdícios, possivelmente poderia reduzir o número de termelétricas em funcionamento, o que reduziria os custos de produção levando a não necessidade de cobrança da bandeira vermelha, que é uma tarifa acrescida na conta de luz à cada 100 quilowatts/hora consumidos(ABESCO, 2015).

No Brasil, há um potencial para economizar 10% de todo o consumo de energia. Por tipo de cliente, o maior potencial de redução (eficiência energética) está na residência dos brasileiros.

A ABESCO estima que poderia ocorrer uma redução de 15% nas casas. Esse percentual é maior do que o dos consumidores industrial, comercial e outros que poderiam economizar, em média, 6,20%, 11% e 10%(ABESCO, 2015).

Dentro desse cenário é conveniente ter um olhar mais atendo sobre como é o consumo de energia elétrica residencial. Porém, as concessionárias fornecem o valor total de potência consumida, o que para as mesmas é suficiente ao cobrar o que foi utilizado. O usuário final não conhece o gasto individual de cada equipamento da residência. Existem algumas estimativas do consumo dos eletrodomésticos mais usuais, tais como lavadeiras, chuveiros, geladeira, entre outros. Como mostrado na Figura 1.1. Porém, tal suposição não garante a verdade sobre o seu consumo individual. Pois cada consumidor pode apresentar um tipo diferente do mesmo eletrodoméstico além de utilizá-lo por um tempo diferente aos usado para criar esta estimativa.

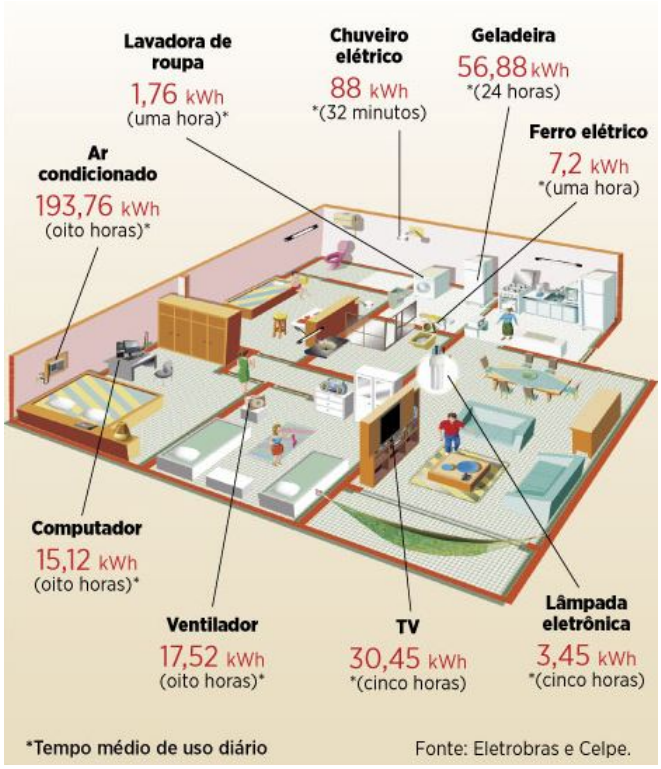


Figura 1.1: Gasto Mensal de energia de alguns equipamentos residenciais

Fonte: ABESCO, 2015

1.2 Era da informação

1.2.1 Domótica

Domótica pode ser definida como a aplicação dos conceitos de automatização e controle aplicados a uma residência. É realizado através da utilização de equipamentos comunicando entre si e/ou com os diferentes ambientes da casa. Desempenhando funções previamente definidas, sendo passíveis de alteração conforme o desejo do usuário. Tais aplicações propiciam uma melhor qualidade de vida, um aumento do bem-estar, reduz o trabalho doméstico MURATORI e DAL BO (2013).

1.2.2 Internet das coisas (IoT)

Em sua essência a Internet das Coisas, do inglês Internet of Things (IoT), nada mais é do que um ambiente que reúne informações de vários dispositivos através de uma conexão com a internet (GRUMAN, 2014). O conceito de IoT juntamente com o advento das aplicações em nuvem proporcionam a fácil conexão e comunicação com qualquer dispositivo integrado à sua automação residencial. Segundo Diniz (2006) a ideia de IoT surge com o paradigma da conexão fornecida pela internet, essa nova definição indica a conexão e troca de dados de qualquer coisa em qualquer instante de tempo.

1.2.3 Computação em nuvem

Outro quesito importante trata-se do termo computação em nuvem. Para Velte et al. (2010) tal nome é uma metáfora da internet. Uma vez que a internet em si é representada com ligações em rede tal como uma nuvem. A figura da nuvem vem para retratar “tudo isso e um pouco mais” que possibilita a rede de funcionar. Se tratando deste conceito, as aplicações são hospedadas em servidores compartilhados e interligados por meio da internet. O acesso a esses dados pode ser realizado de qualquer lugar do mundo remotamente. Esse modelo é mais viável do que a utilização de unidades físicas.

1.3 Justificativa

No Brasil, o consumo energético apresenta alto crescimento no decorrer dos últimos anos. O aumento populacional e consequente aumento no número de domicílios proporcionam um acréscimo no consumo de energia elétrica residencial. De acordo com o censo demográfico do Instituto Brasileiro de Geografia e Estatística, IBGE (2010) estima-se que os 56.5 milhões de domicílios passarão a 75 milhões de unidades em 2020. Um crescimento percentual de apenas 33% em uma década.

Dentro de todo o cenário apresentado justifica-se o surgimento de novas tecnologias, que visam fornecer ao consumidor, uma forma de obter informações acerca do consumo energético individualizado por equipamento residencial. Com um banco de dados de gastos dos equipamentos, espera-se um uso mais racional da energia elétrica residencial, uma vez que o consumidor poderá tomar medidas agindo sobre as fontes que mais consomem dentro do seu lar.

1.4 Objetivo

O presente trabalho tem por objetivo o desenvolvimento de um sistema acessível, conectado individualmente a algum equipamento alimentado por eletricidade. Esse sistema é capaz de fornecer e informar em tempo real, por meio de uma aplicação online, qual o gasto de potência elétrica consumido, apresentando também qual o seu custo monetário. Tal sistema, ainda permitirá o acionamento (LIGA/DESLIGA) do aparelho utilizando uma conexão *wireless*. O acionamento poderá ser feito a partir de qualquer lugar por meio da utilização de dispositivos com acesso à internet, tais como *notebooks*, *smartphones*, *tablets*, entre outros.

1.5 Organização do Trabalho

O presente trabalho será apresentado em seis capítulos, sendo eles em ordem crescente nomeados como, introdução, revisão bibliográfica, metodologia, resultados, considerações finais.

No primeiro capítulo, introdução, apresenta resumidamente o tema que será desenvolvido e de qual forma o trabalho abordará tal tema nos capítulos subsequentes.

No segundo capítulo, revisão bibliográfica, tratar-se-á da abordagem científica dos diversos assuntos abordados nesse trabalho, fornecendo ao leitor uma linha de raciocínio que guia a

leitura.

No capítulo terceiro, metodologia, trata-se do estudo dos caminhos, dos métodos para se chegar à um determinado fim.

No capítulo quarto, resultados, aborda quais os resultados foram obtidos, apresentando uma análise se tais resultados foram satisfatórios para atender ao objetivo do perante projeto.

O quinto capítulo, considerações finais, apresenta-se uma síntese dos elementos constantes no texto do trabalho realizando uma análise das soluções apresentadas.

2 - REVISÃO BIBLIOGRÁFICA

Este capítulo tem por finalidade apresentar os conceitos teóricos que foram utilizados em todo o trabalho. Assim, apresenta-se os componentes físicos e as ferramentas online que possibilitaram a constituição desta monografia.

Portanto, analisa-se:

- Tecnologia *Wirelles - Wifi*
- Planilhas *Google Docs*
- Módulo Wifi ESP8266 -*NodeMCU*
- Protocolo de troca de mensagens - *MQTT BROKER*
- Plataforma de desenvolvimento da programação
- Módulo relé
- Sensores de Corrente

2.1 Tecnologia *Wireless-Wifi*

Wireless, ou rede sem fio, é uma tecnologia capaz de transmitir dados e informações sem a necessidade da utilização de cabos. Isso é possível, pois a comunicação usualmente é realizada por aparelhos de radiofrequência ou de infravermelho. Essa tecnologia inclui desde (*walkie-talkies*) a satélites no espaço. Sua popularização está nas redes de computadores, servindo como forma de acesso à internet através de locais remotos, ou na comunicação entre dispositivos. Denomina-se *Wifi* este tipo de rede sem fio (SILVA, 2008).

Wifi é uma abreviação do termo em inglês, Wireless Fidelity, que significa fidelidade sem fio. É um tipo de rede sem fio que comunica por ondas de rádio e não necessita de licença para

instalação e/ou operação. Sendo necessário apenas que o aparelho receptor seja homologado pela ANATEL e esteja ao alcance das ondas de rádios transmitidas pelo aparelho emissor do sinal (ROVERE, 2016).

A corporação *Wi-Fi Alliance*®, Figura 2.2, trata-se da rede mundial de empresas que oferece os serviços *Wifi*, onde trabalha-se a cada dia para se fazer do *Wifi* uma das tecnologias mais valorizadas e utilizadas em todo o mundo. Define tecnologias e programas wifi inovadores e baseados em padrões de qualidade, desempenho, segurança. Oferece liderança ao setor e defende globalmente regras de uso justo (Alliance®, 2018).

Figura 2.2: Logo da Wifi Alliance
Fonte:WIFI ALLIANCE, INC, 2017

2.2 Planilhas *Google Docs*

O *Google Docs*, Figura 2.3, é um conjunto de aplicativos da gigante *Google LLC*. Seus aplicativos são compatíveis com os do *Microsoft Office*, compondo-se atualmente de um editor de texto, um de apresentações, um de planilhas e um editor de formulários. Sua grande contribuição são que seus aplicativos funcionam totalmente online, possibilitando a edição através de um computador conectado à internet e que possua um navegador compatível (EULER, 2008).

Com o *Google Docs* abre a possibilidade de contar com colaboradores em seus arquivos, isto é, o usuário poderá selecionar pessoas que terão acesso ao seu documento delimitando as que só visualizarão como as que também terão possibilidade de realizar alterações. Tais modificações ficam salvas e, caso o criador do arquivo queira desfazê-las basta acessar o controle de versões do documento, onde obtém informações de quem acessou, quando alterou e o que foi modificado. Tudo simples e direto (EULER, 2008).

Para a realização do projeto aproveitar-se-á do *Google Docs* sua ferramenta de criação e edição de planilhas. Sua utilização possibilita a geração de um banco de dados, além de propiciar

ao usuário a visualização das informações referentes ao consumo elétrico do componente que esteja ligado juntamente com o módulo proposto nessa monografia.A escolha ocorreu pelo fato do *Google Docs* ser uma plataforma online, gratuita e simples, além não precisar de um servidor dedicado à hospedar o banco de dados deste projeto.

Figura 2.3: Logo *Google Docs*

Fonte:GANBATTE, 2017

2.3 Módulo Wifi ESP8266 -*NodeMCU*

Escolheu-se para a concretização do projeto a plataforma *NodeMCU* ilustrada na Figura 2.4. Ela é composta pelo módulo *Wireless ESP8266*, um microprocessador de 32 bits, conectividade *Wifi*, portas de alimentação, onze pinos de entrada e saída, circuitos de regulação de tensão, conectividade USB para programação, sendo compatível com a *Arduino IDE*, GPIO com funções de PWM, tensão de operação: 4.5 à 9V. Suporta 5 conexões TCP/IP e apresenta um baixo custo de mercado girando em torno de R$ 35,00.

O *NodeMCU* é uma placa da família ESP8266, sendo completa e de simples uso. Substitui perfeitamente o uso de outras tecnologias disponíveis tais como microcontroladores, *Arduino*, *Raspberry* na construção de projetos para IoT. Nesta aplicação de IoT, todo o processamento de dados é realiado no *NodeMCU* e enviado por meio do *Wifi WebSocket* para acesso na nuvem. Com o *NodeMCU* a troca de mensagens com a nuvem pode ser feita por meio do protocolo de troca de mensagens *MQTT*.

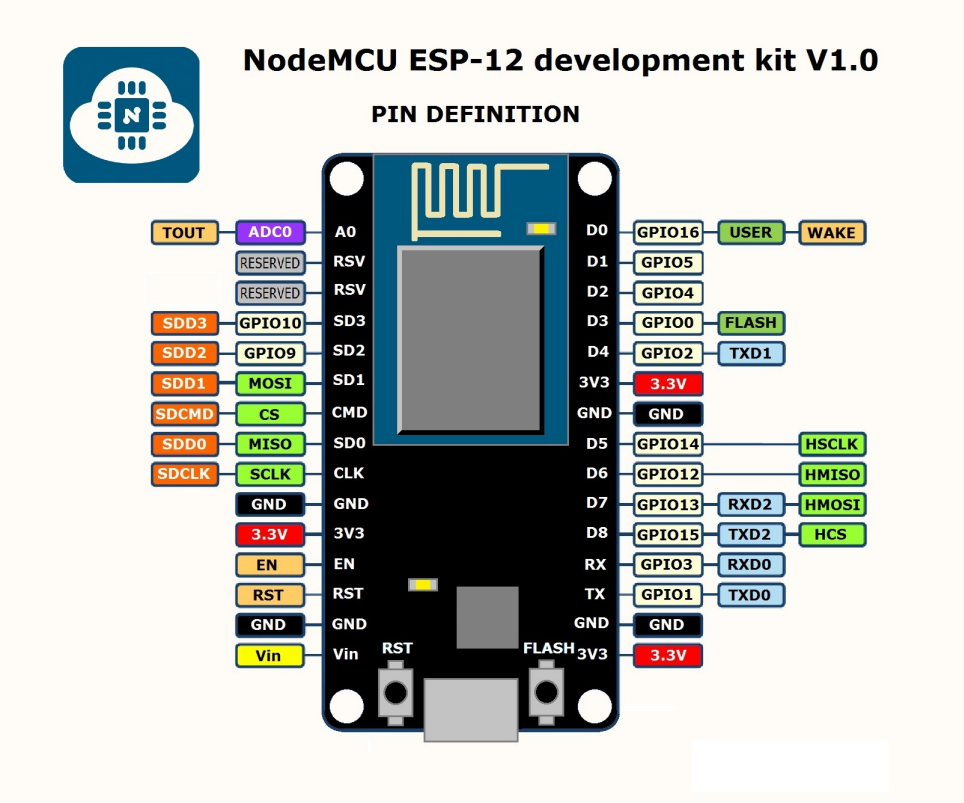


Figura 2.4: NodeMCU família ESP, com definição dos pinos.

Fonte:FILIPEFLOP, 2017

2.4 Protocolo Para Troca de Mensagens - *MQTT*

MQTT é uma abreviação em inglês para *Message Queue Telemetry Transport*, que significa uma fila de mensagens em transporte por telemetria, numa tradução livre. É basicamente um protocolo de troca de mensagens para IoT em uso recente. Foi desenvolvido pela *IBM*, *International Business Machines*, no final da década de 90. Foi criado originalmente para supervisão e coleta de sistemas do tipo *SCADA*, *Supervisory Control and Data Acquisition*, contudo o *MQTT* encontrou lugar no extenso mercado de IoT (BARROS, 2015).

Basicamente o padrão de troca de mensagens é o de publisher/subscriber (publicador/subscritor) conforme ilustrado na Figura 2.5. Desta forma, quando algum dispositivo da rede necessita de uma informação, ele a subscreve, gerando uma requisição para outro elemento da rede responsável por gerenciar publicações e subscrições. Dentro de uma rede *MQTT* este elemento gerenciador é conhecido como *broker*. A publicação de informações também é feita por intermédio do *broker* (BARROS, 2015).

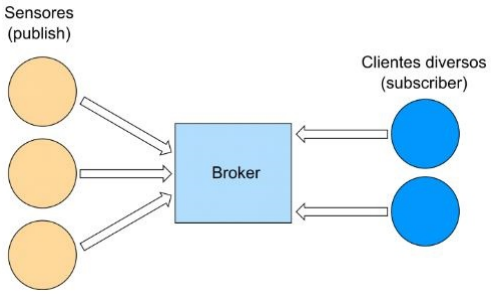


Figura 2.5: *MQTT Overview*

Fonte:EMBARCADOS, 2017

Embora o *broker* represente um ponto de fragilidade na rede pelo fato de centralizar comunicações, ele acaba permitindo um desacoplamento entre os elementos comunicantes, fato não viável em modelos de comunicação do tipo cliente/servidor. O *MQTT* identifica as mensagens por meio de tópicos. O tópico pode conter a informação escolhida por quem o define e cada nível do tópico é separado por barras (“ / ”). Elementos da rede podem enviar vários tópicos para o broker e os subscritores podem escolher os tópicos que desejam subscrever (BARROS, 2015).

2.4.1 *CloudMQQT*

Para efetuar a troca de mensagens entre a aplicação física e o meio online utilizar-se-á a plataforma de desenvolvimento online *CloudMQQT*, hospedada no endereço *https://www.cloudmqtt.com/*. Trata-se basicamente de uma aplicação *web*, voltada para a hospedagem e troca de mensagens em projetos de *IoT*. Torna-se capaz o envio de filas de mensagens perfeitamente configuradas e otimizadas utilizando poucos segundos. A escolha dessa plataforma se deveu pelo fato da mesma apresentar um layout amigável, um serviço gratuito e abranger de maneira simples e efetiva a necessidade do projeto em questão.

2.4.2 *MQTT DASH - Iot, Smart Home*

Com o advento dos *Smartphones* e a quantidade de aplicativos que se tem em mãos nos dias atuais, o presente trabalho utilizará do aplicativo para *IoT*,*MQTT DASH* cujo logo se encontra ilustrado na Figura 2.6, desenvolvido pela *Routix SOftware*. Este componente possibilitará a criação de um painel de indicadores ou painel de bordo. Basicamente será uma forma de apresentar visualmente os elementos que serão acionados remotamente, assim como o *status*

(ligado/desligado) dos mesmos. O aplicativo android trabalhará em conjunto com a aplicação *web* criada e configurada no *CloudMQQT*.

Figura 2.6: Logo: *MQTT Dash*

Fonte:*GOOGLE PLAY*, 2017

A escolha deste aplicativo ocorreu pelo fato do mesmo apresentar uma interface amigável e simples. Seu *download* pode ser feito gratuitamente pela *Play Store*. Como é um aplicativo executado em *android*, é compatível com a maioria dos *Smartphones* que se têm no mercado. E o mais importante, apresenta suporte para comunicação com o *NodeMCU – ESP8266*, que já se utilizará no projeto. Dessa forma conseguir-se-á interligar a aplicação física com o meio da internet possibilitando um acesso para visualização e/ou controle remoto, por meio de um *Smartphone*.

2.5 Plataforma de desenvolvimento da programação

2.5.1 *Arduino IDE*

A programação do *NodeMCU* será feita utilizando o software de programação *Arduino IDE*, de versão 1.8.5, ilustrado na Figura 2.7, utilizando a linguagem de programação em "C". Essa Interface Development, IDE originalmente desenvolvida para ser utilizadas com os microcontroladores da família *Arduino*, também pode ser adaptada para que consiga reconhecer a família de placas *ESP8266*. Ela inclui um editor de códigos com recursos para realce da sintaxe, parênteses correspondentes e identação automática sendo capaz de compilar e carregar os programas diretamente para a placa *ESP8266*. Tal fato foi crucial na escolha desta *IDE*, uma vez que não existirá a necessidade de rodar programas em ambientes de linha de comando.

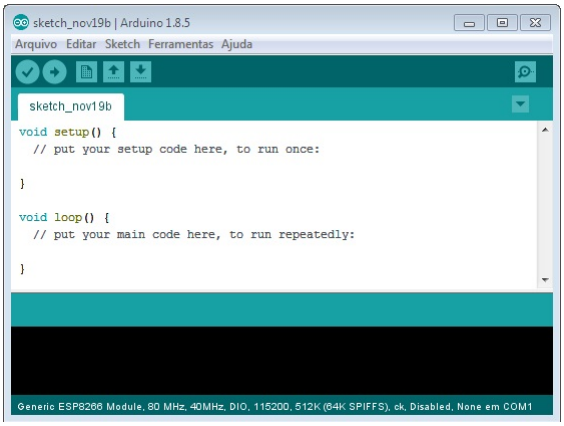


Figura 2.7: *Interface Development - IDE*

Fonte:*SOFTWARE ARDUINO*, 2017

2.6 Módulo relé - Songle

O relé é um dispositivo elétrico com a finalidade de fornecer alterações súbitas e predeterminadas em um ou mais circuitos de saídas. Isso ocorre no momento em que determinadas condições no circuito de entrada, que controla o dispositivo, são alcançadas. Dessa maneira, pode-se enxergar que a principal função do relé não é de interromper o funcionamento do circuito principal, mas sim de o fazer atuar conforme o seu sistema de manobra (CUNHA, 2009).

Dentre algumas características do dispositivo, pode-se destacar que um relé possibilita o acionamento de mais de um circuito simultaneamente com apenas um sinal. Além do mais, os sinais de saída e entrada são independentes; sendo assim a tensão da bobina pode ser diferente da tensão nos contatos; além de poder controlar sinais de corrente alternada por meio de tensão contínua e vice-versa (CUNHA, 2009).

É constituído por uma bobina, uma armadura de ferro móvel, conjunto de contatos, uma mola de rearme e terminais. Para seu funcionamento uma corrente percorre a bobina, tal fato cria-se um campo magnético que atua sobre a armadura atraindo-a. Nesta atração ocorre um movimento que ativa os contatos, podendo ser aberto, fechados, ou comutados.

Para a concepção do projeto foi utilizado o módulo relé Songle, ilustrado na Figura 2.8. O módulo está disposto com uma placa composta com diodo, transistor, leds, conectores e o relé. Por suas características, esse módulo apresenta uma maior facilidade na interligação do projeto com o *NodeMCU*.

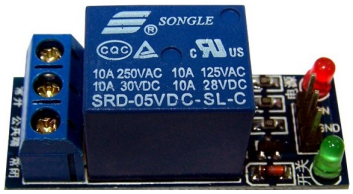


Figura 2.8: Módulo relé

Fonte:FILIPEFLOP, 2017

Com base em seu *Datasheet* algumas das especificações são:

- Tensão de alimentação: 5V DC
- Modelo do relé: SRD-05VDC-SL-C
- Suporta cargas de até 250V AC
- Corrente de operação: 15 a 20mA
- Tempo de resposta: 5 a 10ms

2.7 Medição da Corrente

Atualmente encontra-se no mercado alguns tipos de sensores para medida de corrente elétrica. Estes podem ser baseados em técnicas resistivas, transformadores de corrente e efeito *Hall*.

2.7.1 Efeito Hall

A Figura 2.9 auxilia na descrição do efeito Hall e foi feita pelo (Instituto de física da UFRGS, 2017). Tem-se uma fita condutora com seção reta A=Ld que é percorrida por um feixe de elétrons com velocidade 'v'. Empregando um campo magnético 'B' na direção horizontal, resultará numa força magnética 'FB' na direção perpendicular ao movimento eletrônico, no sentido de cima para baixo. Tal força implicará que o movimento dos elétrons seja desviado para a parte de baixo. Dessa maneira, com o passar do tempo, a parte inferior apresentará um acúmulo de cargas negativas e cargas positivas se concentrarão na face superior. Devido à esse acúmulo de cargas, surgirá uma diferença de potencial denominada tensão *Hall*.
Em síntese, o Efeito Hall é o resultado do desvio da trajetória de uma corrente por intermédio de um campo magnético. Tal desvio ocasiona o surgimento da tensão *Hall*, que pode ser utilizada

por meio de fontes externas, e é como os sensores *Hall* atuam. Seu uso no sensoriamento se deve pelo fato dessa tenão ser proporcional ao campo que geram (Wikipedia, 2018).

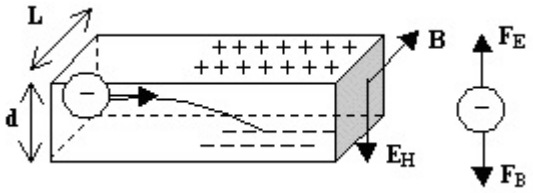


Figura 2.9: Esquema para estudo do efeito Hall

Fonte:INSTITUTO DE FÍSICA DA UFRGS, 2017

2.7.2 Sensor ACS-712

Nesse projeto será utilizado o sensor de corrente linear ACS712, fabricado pela *Allegro MicroSystems* ilustrado na Figura 2.10,fundamentado no efeito *Hall* . Sua escolha se deve por este sensor apresentar uma solução precisa e econômica para a medição de corrente no sistema desenvolvido. Possibilitando medidas de corrente contínua ou alternada. Pode ser encontrado com uma capacidade máxima de medida: 5A, 20A ou 30A. Para o desenvolvimento deste projeto o sensor utilizado será o de 20A, o que possibila a utilização em equipamentos de média potência (ALLEGRO MICROSYTEMS, 2010).

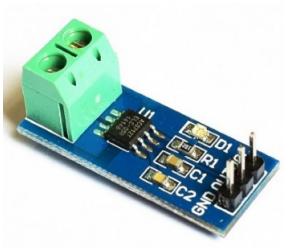


Figura 2.10: Sensor de Corrente ACS721

Fonte:ELETRODEX,2017

METODOLOGIA

3.1 Adaptação da IDE do Arduino

Como a *IDE* de programação utilizada não foi desenvolvida originalmente para o *NodeMCU*, uma pequena adaptação é necessária para que se consiga estabelecer o uso da mesma. Deve-se então instalar um pacote adional que forneça suporte às placas da família *ESP8266* THOMSEN (2016) fornece um conjunto de passos para a instalação desse pacote. O processo é simples e rápido.

1 - Na *IDE* do arduino vá em Preferências, de acordo com a Figura 3.11.

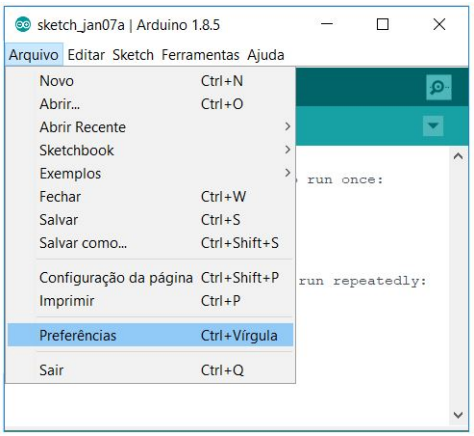


Figura 3.11: Instalação do pacote de placas da família *ESP8266* - Passo 1

Fonte:*ARDUINO IDE*, 2018

2 - Na tela que se abrir digite o link, `http://arduino.esp8266.com/stable/package_esp8266com_index.json`, na parte de URLs adicionais para Gerenciadores de Placas e em seguida clique no OK". Como demonstrado na Figura 3.12.

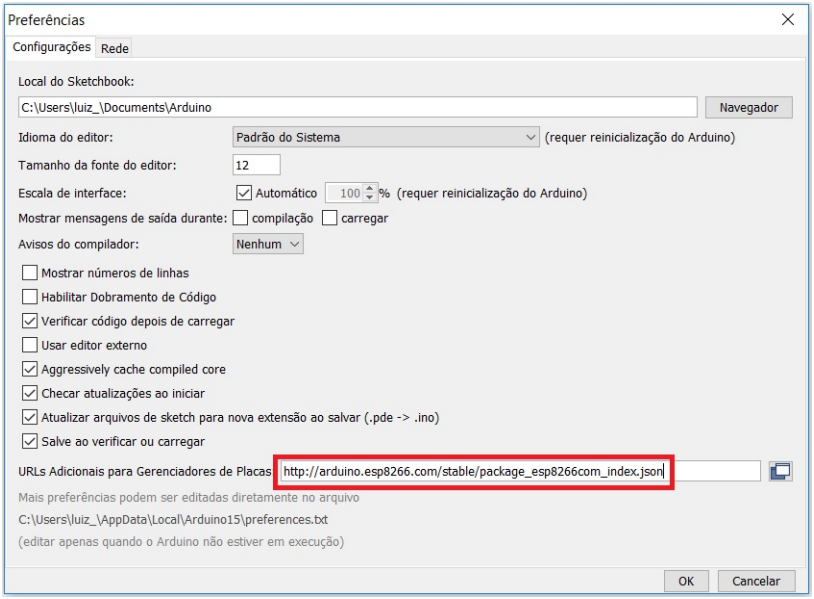


Figura 3.12: Instalação do pacote de placas da família *ESP8266* - Passo 2

Fonte:*ARDUINO IDE*, 2018

3 - Vá até a aba Gerenciador de Placas, ilustrada na Figura 3.13.

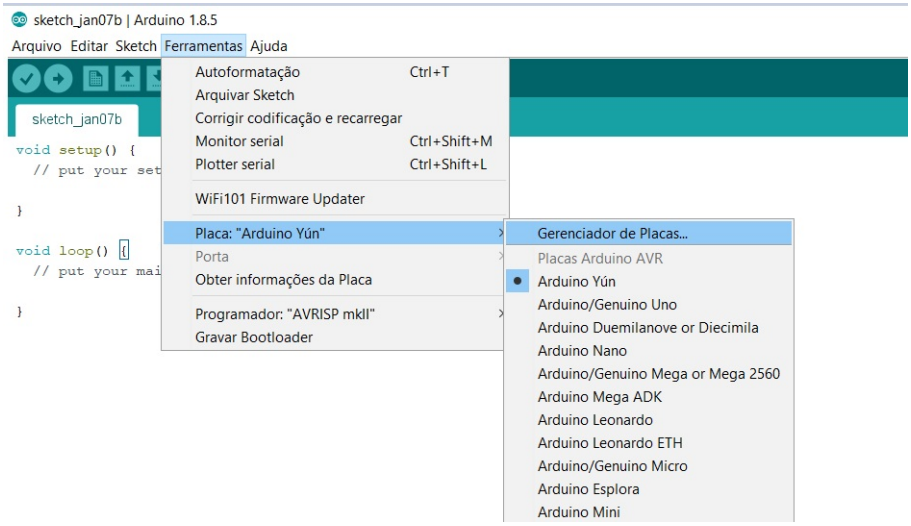


Figura 3.13: Instalação do pacote de placas da família *ESP8266* - Passo 3

Fonte:*ARDUINO IDE*, 2018

4 - Procure pelo pacote *esp8266 by ESP8266 Community* e clique em instalar, de acordo com a Figura 3.14.

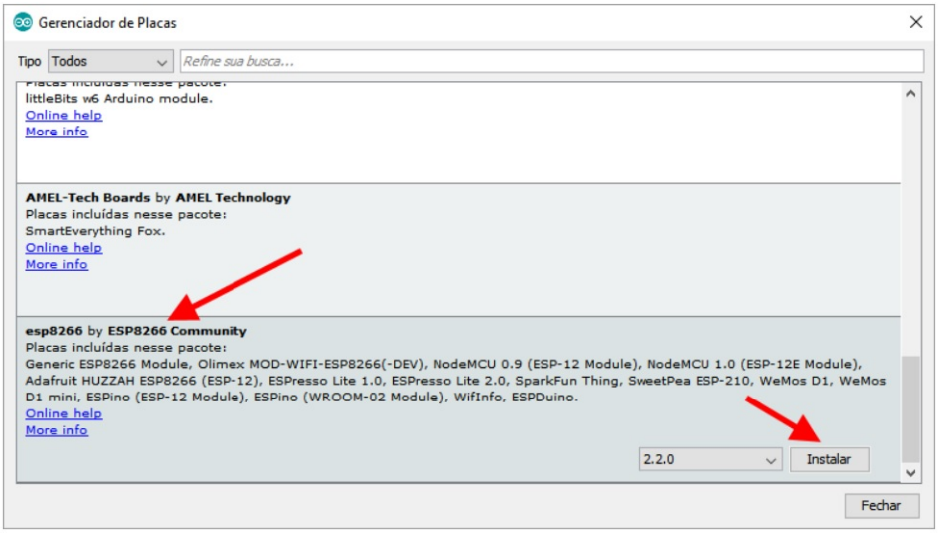


Figura 3.14: Instalação do pacote de placas da família *ESP8266* - Passo 4

Fonte:*ARDUINO IDE*, 2018

3.2 Conectando o *NodeMCU* ao *Wifi*

Uma vez que a comunicação *Wifi* é um dos pilares desse projeto, criou-se um código base para a mesma, apresentado na Figura 3.15. Todo o restante de código desenvolvido para as demais funções, é incrementado nessa primeira estrutura, para assim formarem o conjunto de elementos capazes de atender ao que se propõe nesta presente monografia.

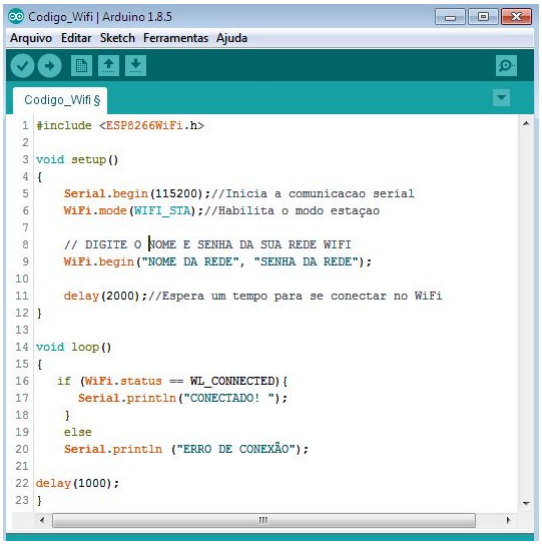


Figura 3.15: Código de comunicação *NodeMCU* com rede *Wifi*

3.3 Criação do Banco de Dados

No desenvolvimento do banco de dados, podem-se considerar duas partes distintas. A primeira delas tratando da criação do sistema onde efetivamente serão armazenados os dados enviados pelo *NodeMCU*. A outra parte fica a cargo do desenvolvimento da programação, o que possibilitará o envio das informações para o banco de dados criado.

3.3.1 Planilha Google

O banco de dados em questão trata-se da ferramenta de planilhas do *Google Docs*. A criação da mesma é feita de forma simples por meio da plataforma online do *Google Drive*. Basta acessar uma conta na plataforma, ir na opção novo e inserir nova planilha, como mostrado na Figura 4.30.

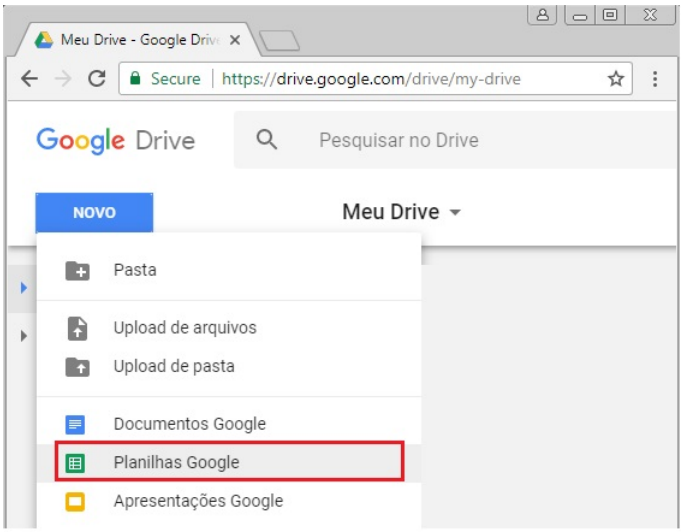


Figura 3.16: Criação da planilha no *Google Drive*

Não há necessidade de preencher a planilha com informações. O preenchimento da mesma se dará por intermédio de um formulário, também criado no *Google Drive* de maneira parecida com a anterior. É importante ressaltar que os títulos das perguntas criadas, se tornarão as colunas da planilha de dados. E as respostas preencherão as lacunas das linhas. Vinculando o formúlario com a planilha, todo dado enviado por intermédio do formulário é salvo automaticamente na planilha.Para isto,basta ir na aba *Respostas*, posteriormente em *Selecionar destino da resposta*, Figura 3.17, e escolher a planilha para vínculo (MORAIS, 2017).

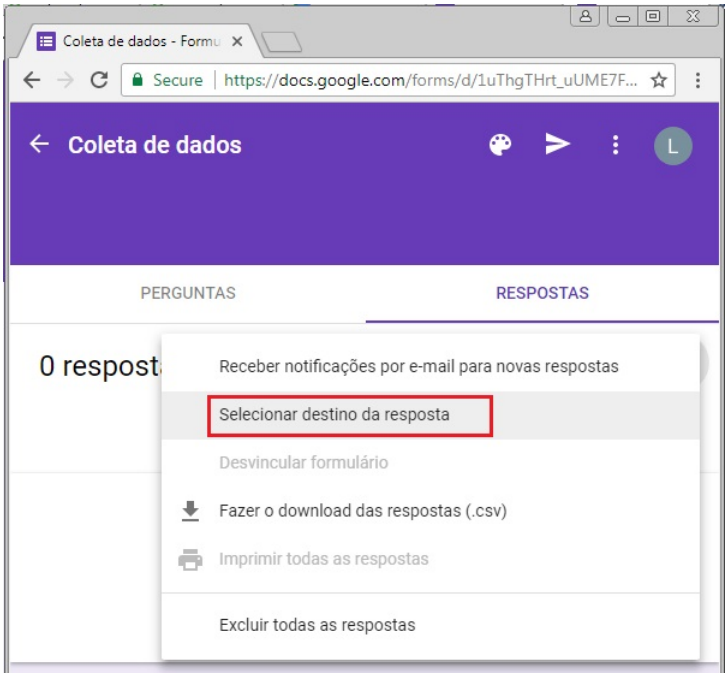


Figura 3.17: Vinculando o formulário com a planilha

Após o término dessas etapas já é possível enviar dados pelo formulário e eles serem armazenados na planilha. Todavia o sistema que se propõe nesta monografia fará tais ações automaticamente, sendo assim é necessário programar o *NodeMCU* de maneira que se consiga realizar tal função.

3.3.2 Programando o *NodeMCU* e sincronizando com o banco de dados

Nessa etapa do processo, precisa-se informar ao *NodeMCU* a identidade do formulário ao qual ele será sincronizado. Cada formulário criado no *Google Docs* apresenta sua própria identidade ou chave, *ID/KEY* (MORAIS, 2017).

A partir da visualização do formulário já se consegue tirar o primeiro identificador, bastando copiar todo o link entre "docs.google.com"e "/viewform"como ilustrado na Figura 3.18.

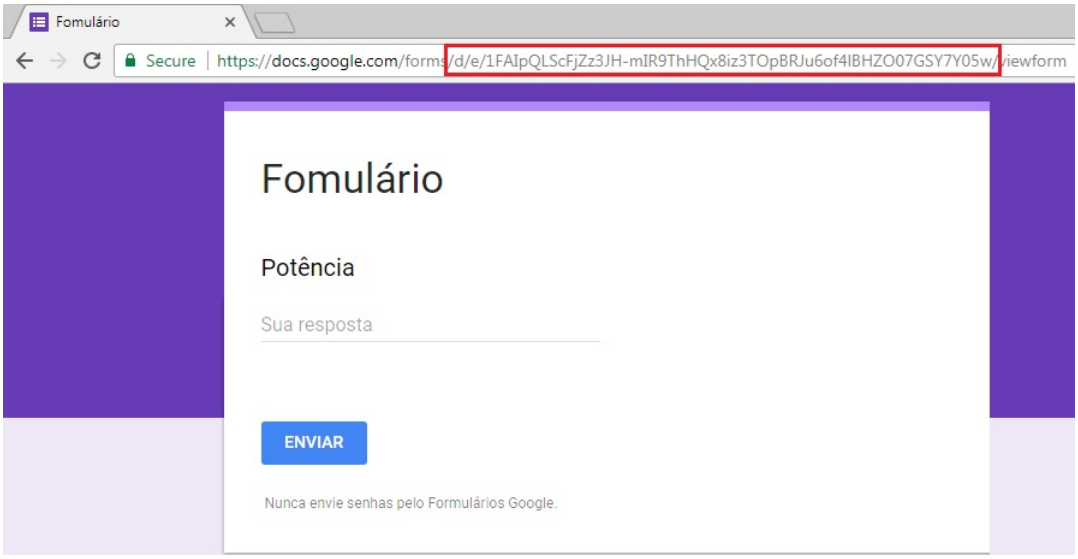


Figura 3.18: Primeiro identificador do formulário

Ainda na aba de visualização clique com o botão direito do mouse no campo destinado à escrita da resposta e vá em inspecionar elemento. Uma outra aba deverá se abrir, conforme ilustra a Figura 3.19. Deve-se copiar o item referente ao nome.

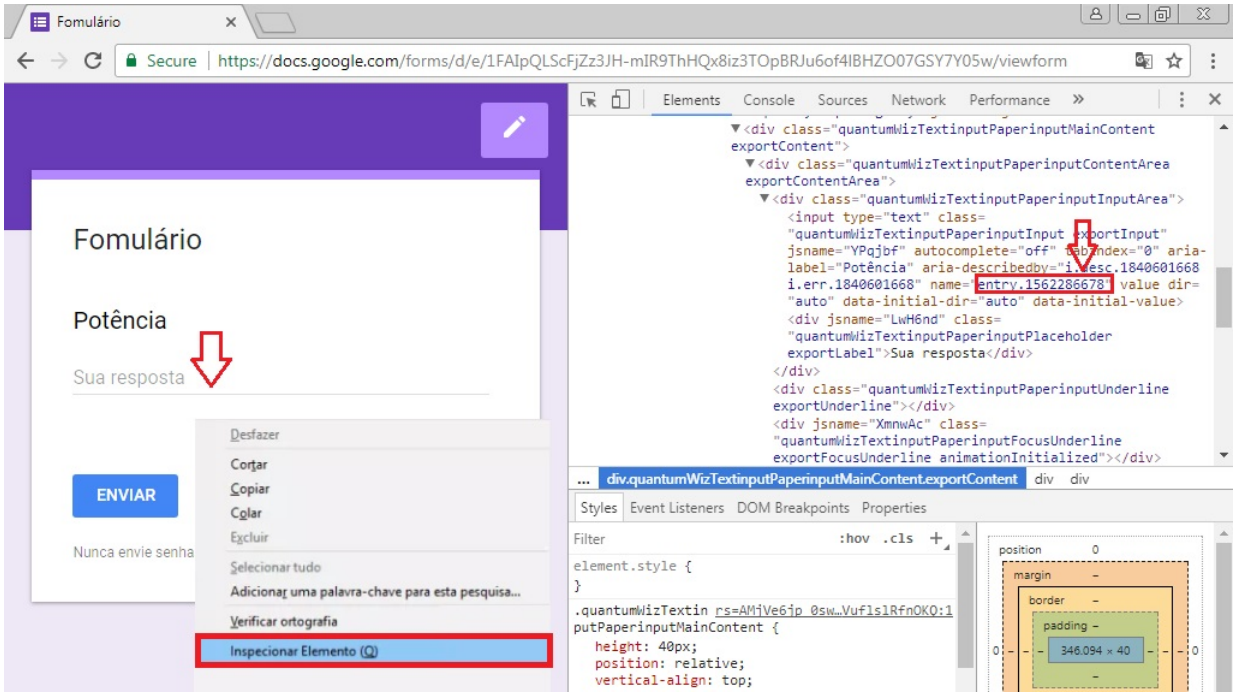


Figura 3.19: Segundo identificador do formulário

Com esses identificadores em mãos torna-se possível usá-los no código de programação para referenciar especificamente o formulário que está vinculado com a planilha. Dessa maneira se

consegue enviar dados por intermédio do NodeMCU, de uma forma com que sejam armazenados na planilha à qual trata-se do banco de dados desse projeto.

Um trecho de código foi criado para executar essas etapas. A variável do tipo *String* e de nome auxiliar, recebe um dos indicadores. Ainda na variável auxiliar, consegue-se acrescentar as informações das variáveis aux1_consumo seguida de sua chave e a variável equipamento também com sua identificação. Como ilustrado na Figura 3.20.

```
String auxiliar = "GET /forms/d/e/1FAIpQLSeISf3DB qV5LtL7P66 smBmCzRCEMwMlH8k3KN6owW6D
                                                  -gsw/formResponse?ifq&entry.1461018459=";

if (client.connect("docs.google.com", 443) == 1)
{
    String envia = auxiliar; //atribuindo para a string envia
    envia += aux1_consumo;
    envia += "&entry.441927641=";
    envia += equipamento;
    envia += "&submit=Submit HTTP/1.1";//enviando para o formulario

    client.println(toSend);
    client.println("Host: docs.google.com");//-
    client.println();//-
    client.stop();//Encerra a conexao com o servidor
}
else
{
    Serial.println("Erro ao se conectar");
}
```

Figura 3.20: Excerto de código para envio das informações planilha

3.4 Cálculo da Potência

3.4.1 Medição da Corrente

Para a medição da corrente utiliza-se do sensor *ACS-712*. Tal sensor faz a interligação da rede elétrica com o equipamento monitorado, assim toda a corrente consumida passará pelo mesmo, que a aferirá. Partindo do valor medido e com base no valor de tensão da rede elétrica tem-se os dados necessários para o cálculo da potência consumida pelo aparelho monitorado.

O *ACS-712* apresenta uma tensão de saída (Vcc) que varia conforme a sua tensão de alimentação. Recebendo uma tensão de alimentação de 5V, apresentará 0V para -20A (corrente mínima) e 5V para 20A (corrente máxima). Logo, quando não se apresenta corrente fluindo no sistema, a

leitura do sensor será 2,5V, o que representa (Vcc/2). Assim a corrente (i) medida é uma relação linear com a tensão de saída do sensor. E pode ser descrita pelo seguinte conjunto de equações.

$$Vcc = m * i + Vcc/2 \tag{3.1}$$

Resolvendo a equação 3.1 para "i", tem-se:

$$i = Vcc/(2 * m) \tag{3.2}$$

Em que: i = corrente medida.

Vcc = tensão de saída do *ACS-712*.

m = sensibilidade do sensor.

Todo esse cálculo descrito será realizado pelo *NodeMCU*, o sinal de saída (Vcc) do sensor de corrente, será lido através da porta analógica (A0). Neste caso, deve-se fazer uma adaptação para a utilização dessa porta. Uma vez que a mesma suporta apenas sinais de tensão com valor máximo de 3.3V, e o *ACS-712* poderá apresentar um valor de saída superior ao suportado. Assim o uso de um circuito divisor de tensão com o uso de dois resistores de 330 Ohms, como descrito na Figura 3.21 apresentou-se necessário.

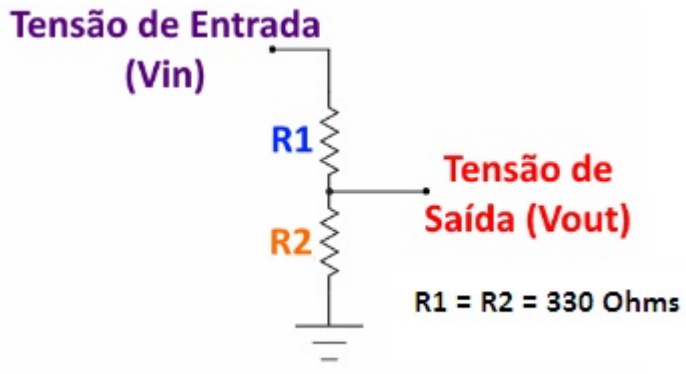


Figura 3.21: Circuito divisor de tensão

Fonte:*ARDUINO E CIA*, 2018

Como os resistores R1 e R2, utilizados na concepção do divisor apresentam a mesma resistência, a tensão no ponto de leitura será a metade da tensão de entrada (Vin) no divisor. Sendo a alimentação do circuito divisor recebida pela tensão de saída do *ACS-712*. Dessa forma, para tensão máxima de saída 5V, tem-se uma leitura de 2,5V. Logo o valor de 2,5V será utilizado

como o tensão de saída (Vcc) de referência para os cálculos da corrente.

O *NodeMcu* é equipado com um conversor analógico (ADC) de 10 bits e 3.3V máximo, representados de 0 a 1023. Assim, uma outra adaptação no desenvolvimento da programação será necessária. Pois pela proteção do divisor,tem-se no máximo 2,5V na entrada do conversor. Dessa maneira deve estabelecer que tal valor se refere ao *range* máximo de 1023. Um excerto do código onde se realiza esta adaptação é apresentado na Figura 3.22 .Para o *ACS-712*, com faixa de -20A a +20A, de acordo com informações do *Datasheet* o valor da sensibilidade equivale a 100mV/A.

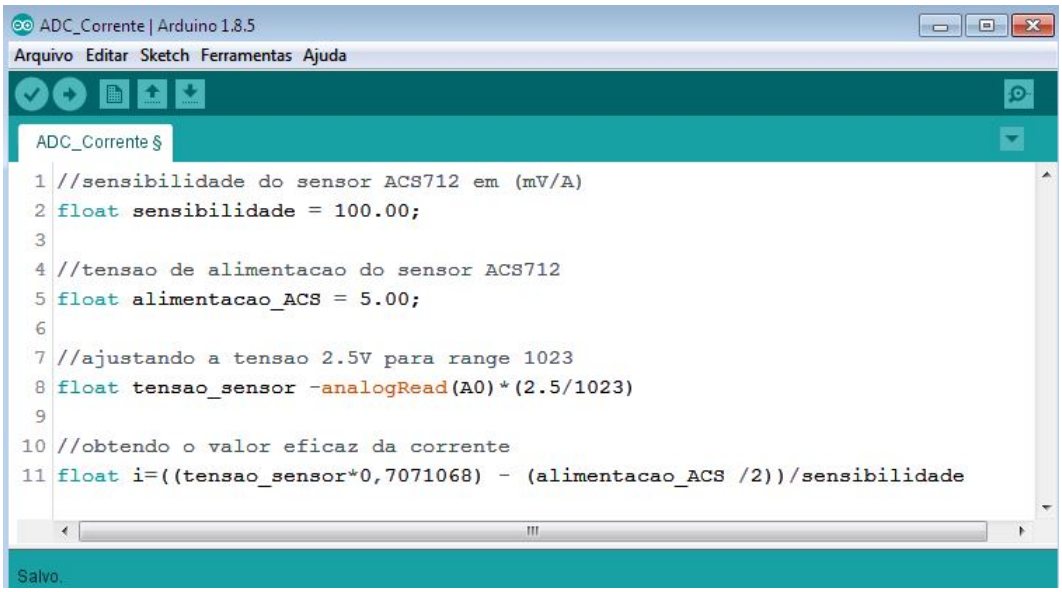


Figura 3.22: Conversão do valor do ADC em corrente

Com o valor da corrente em mãos, basta aplicá-lo à equação 3.3, equivalente da potência.

$$P = V * i \tag{3.3}$$

Em que:

P = Potência.

V = Tensão da rede elétrica.

i = Corrente medida.

3.5 Acionamento

O *NodeMcu* apresenta um conjunto de 8 saídas digitais, enumeradas de D0 à D8. Esse grupo de saídas, se caracteriza por apresentar valores booleanos 0 ou 1 e pode-se, por meio do código de programação, estabelecer um desses valores à saída escolhida.

Para o acionamento de algum dispositivo conectado ao sistema desenvolvido, utiliza-se do módulo relé. O equipamento que se deseja acionar, apresenta uma extremidade conectada diretamente na rede elétrica e a outra passando pelo módulo relé, o qual estará conectado na porta digital (D0) do *NodeMCU*. Uma vez que (D0) estiver configurada com o valor booleano "0", não ocorrerá o acionamento da bobina do módulo relé, sendo assim o circuito apresentará uma descontinuidade, a corrente não fluirá e o equipamento não será ligado. Caso contrário, (D0) assumindo valor booleando "1", a bobina aciona, a corrente flui e o equipamento será conectado.

Um trecho de código que representa o acionamento da saída digital (D0) e consequente acionamento do relé, pode ser observado na Figura 3.23.

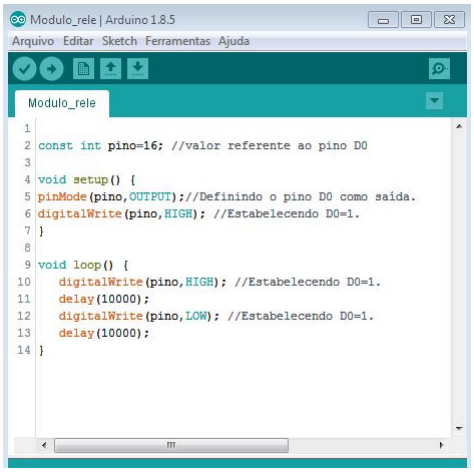


Figura 3.23: Trecho de código acionando o relé

3.6 Aplicação online usando o *MQTT Broker*

3.6.1 Configuração do *CloudMQTT*

No desenvolver da aplicação houve a necessidade de utilizar a plataforma online *CloudMQTT*. O primeiro passo é criar um *login* no site `https://www.cloudmqtt.com/`. Partindo dessa conta criada, já se tem acesso à todas as funcionalidades fornecidas pela mesma. Uma das

formas de se realizar a configuração do *CloudMQTT* e posterior comunicação com o *NodeMCU*, pode ser assim descrita (RABELO, 2017).

1- Crie uma instância e a renomeie, como ilustra a Figura 3.24.

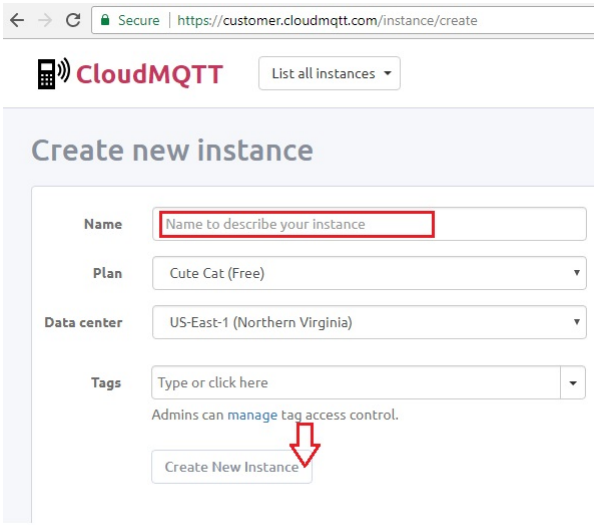


Figura 3.24: Criando uma instância no *CLoudMQTT*

2- Vá em detalhes da instância, conforme Figura 3.25.

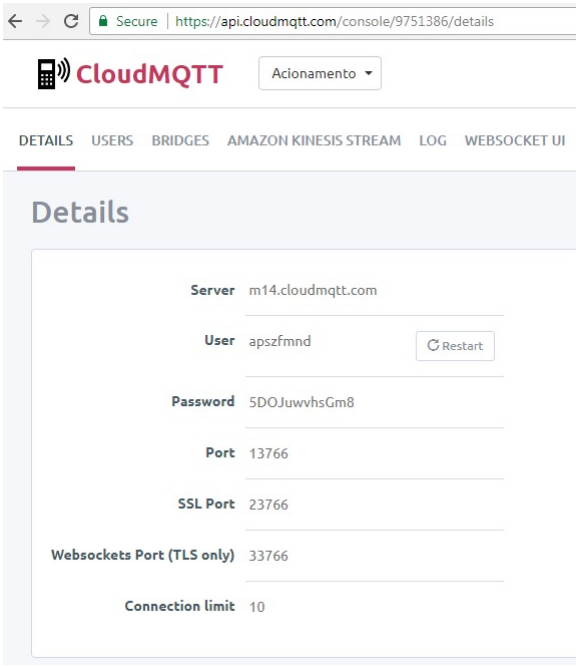


Figura 3.25: Informações avançadas da instância

Essa segunda parte é de extrema importância uma vez que as identificações disponíveis na Figura 3.25 serão utilizadas no código de programação do *NodeMCU* a fim de direcionar para

o *CloudMQTT* criado.

3.6.2 Configurando o *MQTT Dash* no *smatphone android*

A primeira coisa a se fazer é a instalação do aplicativo *MQTT Dash* no *smartphone*. Como trata-se de um aplicativo gratuito, basta acessar a *Play Store* por meio do smatphone, procurar pelo *MQTT Dash* e em seguida instalar.

Com o aplicativo instalado tem-se que configurar o *MQTT Dash* para trabalhar em conjunto com o *CloudMQTT*. Dessa maneira pode-se acionar o dispositivo conectado ao sistema tanto via computador quanto *smatphone*, aumentando dessa maneira a aplicabilidade do sistema desenvolvido nessa monografia. Segundo RABELO (2017), pode-se realizar tal configuração com os seguintes passos.

1 - "Abra o *MQTT Dash* no *smartphone* e adicione um novo *broker*, ver Figura 3.26."

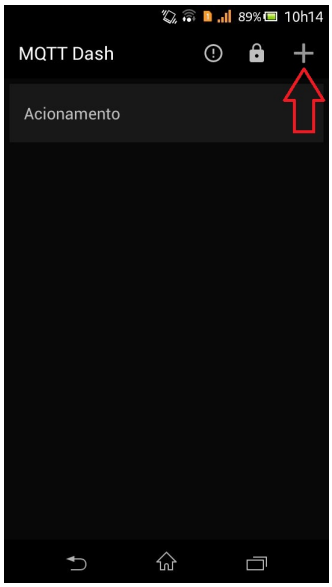


Figura 3.26: Criando um Broker no MQTT Dash

2- "Na janela que se abre, preencha os campos destacados, ver Figura 3.27."

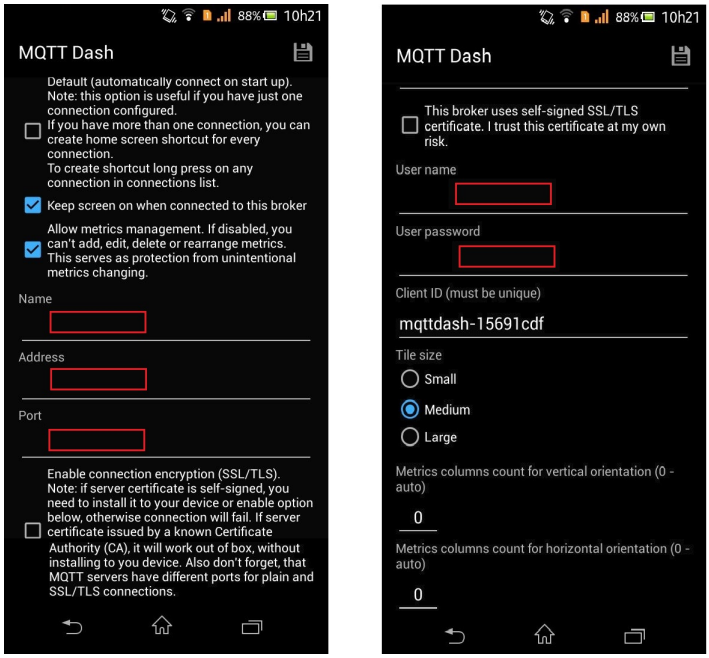


Figura 3.27: Configurando o *MQTT Dash*

O campo *name* fica à critério de escolha e os demais da Figura 3.27 devem ser preenchidos com os dados da Figura 3.25 com seus respectivos semelhantes. Assim, tem-se a união da *aplicação web* com o acesso por *smartphone*.

RESULTADOS

4.1 Montagens elétricas

Para que o módulo desenvolvido seja conectado à rede elétrica fez-se a seguinte configuração. Um terminal elétrico do tipo macho foi conectado à dois cabos elétricos. Um desses cabos foi plugado em na entrada de conexão do sensor de corrente *ACS-712*, na segundo acoplamento do sensor de corrente sai um fio elétrico ao qual conecta-se à uma das extremidades do relé,na segunda junção do relé tem-se outro cabo elétrico, o qual se ligará à um terminal elétrico do tipo fêmea. O segundo cabo do terminal macho se interligará diretamente com o terminal fêmea. Dessa maneira, ao conectar-se a extremidade macho à rede elétrica, a corrente fluirá através do sensor *ACS-712* , possibilitando a sua medição, e energizará o equipamento que recebe a alimentação elétrica do terminal fêmea. O módulo relé conectado entre as ligações fornece a possibilidade de interrupção do fluxo elétrico, sendo assim, pode-se desligar o equipamento conectado. A montagem descrita se encontra ilustrada na Figura 4.28.

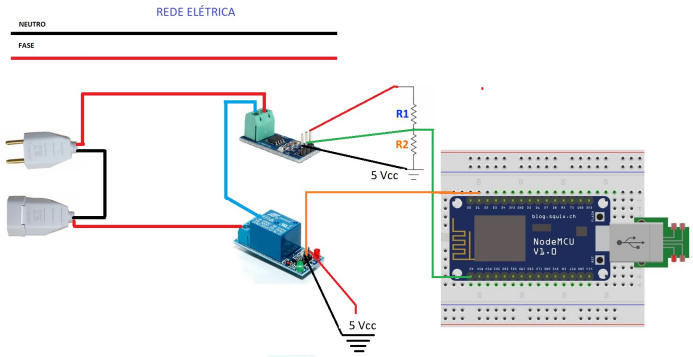


Figura 4.28: Esquemático da montagem do protótipo

O protótipo foi montado em um *protoboard*. Para alimentação dos componentes do circuito eletrônico teve-se a seguinte configuração, o *NodeMCU* é alimentado por intermédio de um cabo fonte análogo aos usados em celulares e *smartphones* de padrão microusb. O próprio *No-*

deMCU fornece a alimentação do sensor de corrente *ACS-712*. No desenvolvimento do módulo percebeu-se que o *NodeMCu* não era capaz de alimentar o módulo relé usado. Dessa maneira, usou-se um cabo fonte de celular ao qual foi cortada a extremidade *micro usb* permitindo o acesso aos fios referentes à saída de 5 Vcc e o terra GND, aos quais foram utilizados para fornecer a alimentação do relé.

4.2 Enviando os dados para a planilha *Google Docs*

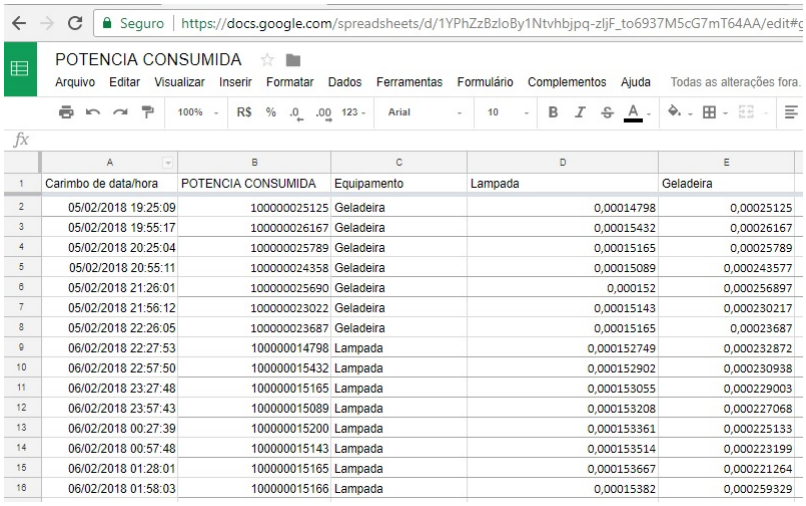
Após realizada a medição da corrente realiza-se o cálculo da potência, em *Watts*, tomando como base a tensão da rede. Para obter o consumo esse valor é convertido em kilowatts hora, KWh, valor como base para ser cobrada a conta pela distribuídora de energia elétrica. A planilha online será formada por três colunas, coluna "A"referente a data e hora, coluna "B"referente aos dados de consumo enviados e coluna "C"referente ao equipamento, como ilustrado na Figura 4.30. Como os valores de consumo serão números relativamente pequenos, o envio para a planilha se dá à cada trinta minutos.

É importante ressaltar que ao realizar o envio do consumo para a planilha *Google Docs*, a variável consumo teve que ser transformada do tipo *float*, usado para tratar números, para o tipo *string*, que é a forma utilizada à referir aos elementos textuais. Para o seu envio, o valor do consumo foi multiplicado por 10^5, dessa forma o número fracionário com quatro casas inteiras “i” e oito decimais “d”, iiii,dddddddd, será convertido num número inteiro da seguinte grandeza iiiiddddd, onde i e d $\in$ {0,1,2,3,4,5,6,7,8,9}. Após esse ajuste o valor de consumo foi tratado conforme a adaptação de código referente à Figura 4.29, e guardado numa variável *string* chamada aux1_consumo, a qual sempre tem o termo textual “1” no início dos seus caracteres, dessa forma, não se perderá casas decimais caso apresente zeros à esquerda. Dessa forma a string auxiliar ficará composta por elementos textuais da seguinte forma, aux1_consumo=”1iiiiddddd”, sendo enviado o conteúdo de aux1_consumo para a coluna “B” da planilha *Google Docs*.

```
//realiza o envio a cada 5 minutos 300000 milisegundos
        if (tempoEnvio>300000){
           String aux1_consumo;
             int aux_consumo=consumo*100000; //vamos considerar 5 casas decimais
             int i=0;
             int a=1;
             aux1_consumo="1"; //variavel auxiliar que armazena o valor do consumo como String
             while (i<7){
                int num =aux_consumo/(100000000/a);
                if (num == 9){
                   aux1_consumo+="9";
                  }
                  else if (num==8){
                    aux1_consumo+="8";
                    }
                    else if (num==7){
                    aux1_consumo+="";
                    }
                    else if (num==6){
                    aux1_consumo+="6";
                    }
                    else if (num==5){
                    aux1_consumo+="5";
                    }
                    else if (num==4){
                    aux1_consumo+="4";
                    }
                    else if (num==3){
                    aux1 consumo+="3";
                    }
                    else if (num==2){
                    aux1_consumo+="2";
                    }
                    else if (num==1){
                    aux1_consumo+="1";
                    }
                    else aux1_consumo+="0";
                aux_consumo=aux_consumo%(100000000/a);
                a=a*10;
                i=i+1;
             }
             a=1;
             i=0;
```

Figura 4.29: Trecho de código utilizado para converter *float* em *String*

Apesar de o consumo ser enviado como uma variável textual a planilha Google Docs o compreende como valores numéricos e os salva na coluna “B” da Figura 4.30. Porém, tais valores não representam efetivamente o dado medido. Sendo assim, criou-se as colunas “D”, “E” onde cada célula realiza uma operação de seleção baseada no equipamento relativo à coluna “C”, da Figura 4.30, e executam as operações inversas que foram executadas no código ajustando assim para o valor medido efetivamente.



	A	B	C	D	E
1	Carimbo de data/hora	POTENCIA CONSUMIDA	Equipamento	Lampada	Geladeira
2	05/02/2018 19:25:09	100000025125	Geladeira	0,00014798	0,00025125
3	05/02/2018 19:55:17	100000026167	Geladeira	0,00015432	0,00026167
4	05/02/2018 20:25:04	100000025789	Geladeira	0,00015165	0,00025789
5	05/02/2018 20:55:11	100000024358	Geladeira	0,00015089	0,000243577
6	05/02/2018 21:26:01	100000025690	Geladeira	0,000152	0,000256897
7	05/02/2018 21:56:12	100000023022	Geladeira	0,00015143	0,000230217
8	05/02/2018 22:26:05	100000023687	Geladeira	0,00015165	0,00023687
9	06/02/2018 22:27:53	100000014798	Lampada	0,000152749	0,000232872
10	06/02/2018 22:57:50	100000015432	Lampada	0,000152902	0,000230938
11	06/02/2018 23:27:48	100000015165	Lampada	0,000153055	0,000229003
12	06/02/2018 23:57:43	100000015089	Lampada	0,000153208	0,000227068
13	06/02/2018 00:27:39	100000015200	Lampada	0,000153361	0,000225133
14	06/02/2018 00:57:48	100000015143	Lampada	0,000153514	0,000223199
15	06/02/2018 01:28:01	100000015165	Lampada	0,000153667	0,000221264
16	06/02/2018 01:58:03	100000015166	Lampada	0,00015382	0,000259329

Figura 4.30: Planilha *Google Docs* com os dados enviados

Em uma outra aba da planilha criou-se uma pequena representação, ilustrada na Figura 4.31, que realiza a soma do consumo filtrado por equipamento. Gerou-se um gráfico com a soma dos valores, o qual demostra um custo individual por equipamento e uma barra com o custo total.Existe um campo para a inserção do custo da energia elétrica, o qual usou-se R$0.69342 que era o valor corrente no estado do Rio de Janeiro no mês de março de 2017 de 52 até 300 KWh residencial (LIGHT, 2017).

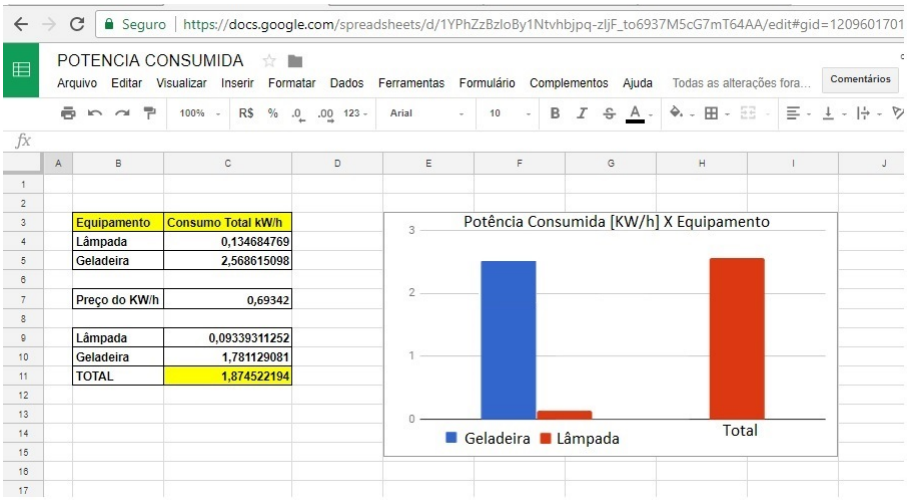


Figura 4.31: Adaptação final para melhor visualização

4.3 Aplicativo no MQTT-Dash

O aplicativo criado com a ferramenta *MQTT-Dash*, ilustrado na Figura 4.32, é de simples entendimento e prático para a utilização. Nele estão representados os equipamentos aos quais

os testes foram realizados, conforme Figura 4.32. Como apenas um protótipo de medição do consumo foi criado, todos os equipamentos realizam o *Publisher* no mesmo tópico, ou seja, acionam ou não o mesmo módulo protótipo, desta maneira o uso fica restrito ao equipamento que está conectado na rede elétrica por intermédio do protótipo desenvolvido.

Figura 4.32: Visual do aplicativo no *MQTT-Dash*

O aplicativo funcionou perfeitamente,sendo possível acionar ou desativar o elemento ao qual estava conectado no protótipo desenvolvido, tendo apenas como limitação, o fato de poder utilizar apenas um dos componentes ilustrados na Figura 4.32, por vez. Num desenvolvimento mais amplo o ideal seria a criação de vários protótipos aos quais teriam o *Publisher* independente e cada um deles trataria exclusivamente de um equipamento.

CONSIDERAÇÕES FINAIS

5.1 Conclusão

O objetivo principal deste projeto foi alcançado, criando um protótipo de baixo custo e que fornecesse um valor confiável da medida de potência consumida, além de possibilitar o acionamento remoto através de *smartphones*, *tablets* e computadores, conectados nas mais diversas redes de *Wifi*. Porém torna-se importante salientar alguns pontos.

Como foi utilizado um sensor de corrente *ACS-712* de 20A, medidas de corrente com valores pequenos, até os 2A, possuem uma precisão baixa. Porém tais aparelhos não se encontram nos vilões dos gastos energéticos numa residência, sendo assim, uma economia no uso dos mesmos não significaria um impacto relevante no consumo final total.

Para o cálculo da potência considerou-se o valor de tensão da rede elétrica igual a 220V. Sendo assim, a utilização do protótipo em redes elétricas com tensão de 127V proporcionará divergência nos resultados. Para solucionar tal incompatibilidade se torna interessante o acréscimo de um sensor de tensão ao protótipo. Dessa maneira seria possível utilizá-lo em quaisquer tensões de alimentação. Ainda nos forneceria um valor mais real da potência consumida, uma vez que o valor da tensão elétrica não é constante e sofre algumas pequenas variações.

Apesar dos testes terem sido realizados com aparelhos domésticos tal protótipo pode ser utilizado com equipamentos elétricos de baixa e média potência dos mais variados tipos. O sensor de corrente limita o uso de aparelhos que necessitam de uma corrente superior à 20A. Porém, caso haja o interesse em abranger tais equipamentos necessita-se apenas da substituição do sensor de corrente e atualização dos dados do novo sensor no código de programação.

Aplicações com o uso de *IoT* vêm ganhando destaque na automação em geral. Nesse fator o módulo *NodeMCU* surge como uma das opções de microcontroladores usuais capazes de abranger diversas aplicações. O fato de já ser integrado com um módulo *Wifi* apresenta uma grande

facilidade para interligar o processo físico do projeto desenvolvido com o meio da internet. Tal fator ficou evidente no decorrer da criação deste presente trabalho.

5.2 Trabalhos Futuros

Como o presente trabalho se fundamentou na criação de um protótipo, o qual obteve êxito no que se propunha, é interessante ressaltar pontos que possam melhorar o seu desempenho.

1- O acréscimo de um sensor de tensão, resultaria na possibilidade do uso em diversas tensões de alimentação. Forneceria também dados mais reais uma vez que acompanharia as variações da rede.

2- Os dados medidos são salvos em planilhas *Google Docs*, de modo que seria interessante o desenvolvimento de um banco de dados mais refinado, com uma boa interface, facilitando a visulisação das informações importantes,pelo usuário.

3- Aqui utilizou-se apenas de um *NodeMCU*, havendo a necessidade de trocar os equipamentos para realizar a medição. Mais conjuntos desse protótipo poderiam ser criados, dessa maneira, cada um poderia estar ao uso exclusivamente de um equipamento e a integração entre dois ou mais módulos poderia ser implementada bem como a da informação disponibilizada para o usuário.

REFERÊNCIAS BIBLIOGRÁFICAS

ABESCO, **Associação Brasileira das Empresas de Serviços de Conservação de Energia**. Desperdício de energia gera perdas de R$12,6 bilhões. Outubro 2015. Disponível em < `http://www.abesco.com.br/pt/novidade/desperdicio-de-energia-gera-perdas-de-r-126-bilhoes`>. Acesso: em 28 out 2017.

BARROS, Marcelo. MQTT – **Protocolos para IoT**. Junho de 2015. Disponível em < *https://www.embarcados.com.br/mqtt-protocolos-para-iot/*>. Acesso em: 18 nov 2017.

BORENSTEIN, Carlos Raul et al. **Regulação e gestão competitiva no setor elétrico brasileiro**. Porto Alegre: Sagra Luzzatto, 1999.

CUNHA, Lívia. Relê e contatores. **Revista: O setor elétrico**. Ed, v. 45, 2009.

DINIZ, Eduardo H. Internet das coisas. **GV-executivo**, v. 5, n. 1, p. 59, 2006.

EULER, Guilherme. **Conheça o Google Docs**, Setembro de 2008. Disponível em < `https://www.infodicas.com.br/dicas_tuto/conhea-o-google-docs`>. Acesso em: 18 nov 2017.

GRUMAN, Galen. **IoT é um grande e confuso campo à espera de explodir**. Novembro de 2014. Disponível em< `http://computerworld.com.br/tecnologia/2014/11/25/iot-e-um-grande-e-confuso-campo-a-espera-de-explodir`>. Acesso em: 02 nov 2017.

Instituto de Física da Universidade Federal do Rio grande do Sul, Física Geral II, **O Campo Magnético**. Capítulo 8. Disponível em <`https://www.if.ufrgs.br/tex/fis142/mod08/m_s03.html`>. Acesso em: 04 nov 2017.

LIGHT. **Composição da tarifa**. Março de 2017. Diponível em <`http://www.light.com.br/para-residencias/Sua-Conta/composicao-da-tarifa.aspx`>. Acesso em: 07 fev 2018.

MARTINS, Andre RS; ALVEAL, Carmem; SANTOS, Ednilson Moutinho dos; **Eficiência energética: integrando usos e reduzindo desperdícios**. In: Eficiência energética: integrando usos e reduzindo desperdícios. ANEEL/ANP, 1999.

MORAIS, José. **Banco de dados com google planilhas – ESP**. Outubro de 2017. Disponível em <https://portal.vidadesilicio.com.br/banco-de-dados-com-google-planilhas-esp/>. Acesso em: 04 nov 2017.

MURATORI, José Roberto; DAL BÓ, Paulo Henrique. **Automação residencial conceitos e aplicações**. 2013.

RABELO, Marco; **ESP8266 com broker MQTT**, Maio de 2017. Disponível em <https://www.youtube.com/watch?v=oX4ttJEULmA&list=PLYCwvtgsGtc-7ve0gCGMEkod5bCDBxHUy&index=5>. Acesso em: 03 jan 2018.

ROVERE, Rodrigo L Della. **Protótipo de um sistema inteligente de monitoramento do consumo de energia elétrica em uma residência**. Monografia em engenharia elétrica, Universidade São Paulo. São Paulo. p.63,2016.

SILVA, R.L. **Manual da tecnologia Wireless** – Módulo sobre a tecnologia Wireless.2008.

SMIL, Vaclav. World history and energy. **Encyclopedia of energy**, v. 6, p. 549-561, 2004.

THOMSEN, Adilson. **Como programar o NodeMCU com a IDE do Arduino**- Blog do FilipeFlop, 2016. Disponível em <https://www.filipeflop.com/blog/programar-nodemcu-com-ide-arduino/>. Acesso em: 07 jan 2018.

VELTE, Anthony T. et al. **Cloud computing: a practical approach**. New York: McGraw-Hill, 2010.

Wi-Fi Alliance®. **Who we are**,2018. Disponível em <https://www.wi-fi.org/ who-we-are>. Acesso em: 21 jan 2018.

Wikipedia.**Sensor de Efeito Hall**. Disponível em <https://pt.wikipedia.org/wiki/Sensor_de_efeito_Hall>. Acesso em: 21 jan 2018.

APÊNDICE A

```
1
2  //AUTOR: LUIZ HENRIQUE JUNIOR PEREIRA
3  //DATA: 06/02/2018
4  //Tal codigo foi realizado se baseando no codigo
       MQTT_WiFiManager_Rele.ino desenvolvido por
5  //Marco Rabelo, para o curso de ESP8266.
6
7  //DECLARACAO DAS BIBLIOTECAS UTILIZADAS NO CODIGO
8  #include <ESP8266WiFi.h>
9  #include <FS.h>
10 #include <ESP8266WiFi.h>
11 #include <DNSServer.h>
12 #include <ESP8266WebServer.h>
13 #include <WiFiManager.h>
14 #include <ArduinoJson.h>
15 #include <PubSubClient.h>
16 #include <EEPROM.h>
17
18 #define DEBUG
19
20 //VARIAVEIS USADAS NA MEDIDA DE CORRENTE
21  float amostra_sensor; //valor medido pelo sensor em cada amostra,
       usaremos 1000
22  float valor_sensor; // valor medio das 500 amostras de leitura
23  float ADC_TO_VOLT =0.00244379 ;//converte os 1023 do pino ADC, em
       volts( 5V = 1023)
24  float corrente; //valor medido da corrente
```

```
25  float potencia=0; //valor onde sera armazenada a potencia
26  float consumo=0; //variavel que armazena o calculo do consumo
27  float tensao=220;
28  float sensibilidade = 100;//sensibilidade
29  float potencia_seg;
30  unsigned long tempo=0, tempoEnvio, tempoTotal;
31  String equipamento;
32
33 //VARIAVEIS USADAS NO ENVIO PARA AS PLANILHAS GOOGLE DOCS
34 WiFiClientSecure planilha;//Cria um cliente seguro (para ter acesso
      ao HTTPS)
35 String auxiliar = "GET /forms/d/e/1
      FAIpQLSeISf3DB_qV5LtL7P66_smBmCzRCEMwMlH8k3KN6owW6D-gsw/
      formResponse?ifq&entry.1461018459=";
36 //Essa String sera uma auxiliar contendo o link utilizado pelo GET,
      para nao precisar ficar re-escrevendo sempre
37
38 //ATRIBUINDO AS REFERENCIAS DO CLOUD-MQTT
39 #define MQTT_server  "m14.cloudmqtt.com"
40 #define MQTT_porta   "13766"
41 #define MQTT_usuario "apszfmnd"
42 #define MQTT_senha   "5DOJuwvhsGm8"
43 #define MQTT_topico  "acionamento"
44
45 #define pino                    5;
46 bool precisaSalvar  =           false;
47
48 //MEDICAO DA CORRENTE E CALCULO DA POTENCIA
49 float mede_corrente(){
50      for(int i=0; i<500; i++) //USANDO A MEDIA DE 1000 AMOSTRAS
51  {
52    amostra_sensor = (analogRead(A0) -511);// para 0 Ampere a leitura
        ADC do sensor equivale a 511
53    valor_sensor +=(amostra_sensor*amostra_sensor);
```

```
54   }
55   float aux = (sqrt (valor_sensor)/500)*ADC_TO_VOLT;
56   corrente = (aux/sensibilidade)*1000; // calcula o valor da corrente
        e ajusta para Ampere;
57   valor_sensor=0;
58   potencia = (corrente * tensao);    //calculo da potencia para rede
        110V
59   potencia_seg = potencia/3600;      //calculo da potencia por segundos
60   consumo = consumo + (potencia_seg/1000); //calculo do consumo
61    return consumo;
62 }//fim do mede_corrente
63
64 //Funcao para imprimir na porta serial
65 void imprimirSerial(bool linha, String mensagem){
66   #ifdef DEBUG
67     if(linha){
68       Serial.println(mensagem);
69     }else{
70       Serial.print(mensagem);
71     }
72   #endif
73 }
74
75 //Funcao de retorno para notificar sobre a necessidade de salvar as
      configuracoes
76 void precisaSalvarCallback() {
77   imprimirSerial(true, "Deve salvar as configuracoes");
78   precisaSalvar = true;
79 }
80
81 //Funcao que reconecta ao servidor MQTT
82 void reconectar() {
83   //Repete ate conectar
84   while (!client.connected()) {
```

```
85      imprimirSerial(false, "Tentando conectar ao servidor MQTT...");
86
87      bool conectado = strlen(MQTT_usuario) > 0 ?
88                       client.connect("ESP8266Client", MQTT_usuario,
                            MQTT_senha) :
89                       client.connect("ESP8266Client");
90
91      if(conectado) {
92        imprimirSerial(true, "Conectado!");
93        //Subscreve para monitorar os comandos recebidos
94        client.subscribe(MQTT_topico, 1); //QoS 1
95      } else {
96        imprimirSerial(false, "Falhou ao tentar conectar. Codigo: ");
97        imprimirSerial(false, String(client.state()).c_str());
98        imprimirSerial(true, " tentando novamente em 5 segundos");
99        //Aguarda 5 segundos para tentar novamente
100       delay(5000);
101     }
102   }
103 }
104
105 //Funcao a ser chamada quando chegar mensagem do MQTT
106 void retorno(char* topico, byte* mensagem, unsigned int tamanho) {
107   //Convertendo a mensagem recebida para string
108   mensagem[tamanho] = '\0';
109   String strMensagem = String((char*)mensagem);
110   strMensagem.toLowerCase();
111   //float f = s.toFloat();
112
113   imprimirSerial(false, "Mensagem recebida! Topico: ");
114   imprimirSerial(false, topico);
115   imprimirSerial(false, ". Tamanho: ");
116   imprimirSerial(false, String(tamanho).c_str());
117   imprimirSerial(false, ". Mensagem: ");
```

```
118   imprimirSerial(true, strMensagem);
119
120   //Executando o comando solicitado
121   imprimirSerial(false, "Status do pino antes de processar o comando:
          ");
122   imprimirSerial(true, String(digitalRead(pino)).c_str());
123
124   if(strMensagem == "ligalampada"){
125    pinMode(pino, OUTPUT);//OUTPUT, coloca o pino em estado alto 5V
126    equipamento="Lampada";
127   }else if(strMensagem == "ligageladeira"){
128    pinMode(pino, OUTPUT);
129    equipamento="Geladeira";
130   }else (strMensagem == "desliga"){
131      pinMode(pino, INPUT); //INPUT, coloca o pino em estado baixo 0V
132     }
133   }
134
135   imprimirSerial(false, "Status do pino depois de processar o comando
         : ");
136   imprimirSerial(true, String(digitalRead(pino)).c_str());
137 }
138
139 //Funcao para configuracao inicial
140 void setup()
141   #ifdef DEBUG
142     Serial.begin(115200);
143   #endif
144   imprimirSerial(true, "...");
145
146   //Iniciando o SPIFSS (SPI Flash File System)
147   imprimirSerial(true, "Iniciando o SPIFSS (SPI Flash File System)");
148   if (SPIFFS.begin()) {
149     imprimirSerial(true, "Sistema de arquivos SPIFSS montado!");
```

```
150     if (SPIFFS.exists("/config.json")) {
151       //Arquivo de configuracao existe e sera lido.
152       imprimirSerial(true, "Abrindo o arquivo de configuracao...");
153       File configFile = SPIFFS.open("/config.json", "r");
154       if (configFile) {
155         imprimirSerial(true, "Arquivo de configuracao aberto.");
156         size_t size = configFile.size();
157 
158         //Alocando um buffer para armazenar o conteudo do arquivo.
159         std::unique_ptr<char[]> buf(new char[size]);
160 
161         configFile.readBytes(buf.get(), size);
162         DynamicJsonBuffer jsonBuffer;
163         JsonObject& json = jsonBuffer.parseObject(buf.get());
164         json.printTo(Serial);
165         if (json.success()) {
166             //Copiando as variaveis salvas previamente no aquivo json
                    para a memoria do ESP.
167             imprimirSerial(true, "arquivo json analisado.");
168             strcpy(MQTT_server, json["MQTT_server"]);
169             strcpy(MQTT_porta, json["MQTT_porta"]);
170             strcpy(MQTT_usuario, json["MQTT_usuario"]);
171             strcpy(MQTT_senha, json["MQTT_senha"]);
172             strcpy(MQTT_topico, json["MQTT_topico"]);
173 
174         } else {
175           imprimirSerial(true, "Falha ao ler as configuracoes do
                arquivo json.");
176         }
177       }
178     }
179   } else {
180     imprimirSerial(true, "Falha ao montar o sistema de arquivos
          SPIFSS.");
```

```
181   }
182   //Fim da leitura do sistema de arquivos SPIFSS
183
184   //fornecendo os parametros para o WifiManager
185   WiFiManagerParameter custom_mqtt_server("server", "Servidor MQTT",
          MQTT_server, 40);
186   WiFiManagerParameter custom_mqtt_port("port", "Porta", MQTT_porta,
          6);
187   WiFiManagerParameter custom_mqtt_user("user", "Usuario",
          MQTT_usuario, 20);
188   WiFiManagerParameter custom_mqtt_pass("pass", "Senha", MQTT_senha,
          20);
189   WiFiManagerParameter custom_mqtt_topic_sub("topic_sub", "Topico
          para subscrever", MQTT_topico, 30);
190
191   //Iniciando o WifiManager
192   WiFiManager wifiManager;
193
194   //definindo a funcao que avisa para salvar as configuracoes
195   wifiManager.setSaveConfigCallback(precisaSalvarCallback);
196
197   //Adicionando os parametros para conectar ao servidor MQTT
198   wifiManager.addParameter(&custom_mqtt_server);
199   wifiManager.addParameter(&custom_mqtt_port);
200   wifiManager.addParameter(&custom_mqtt_user);
201   wifiManager.addParameter(&custom_mqtt_pass);
202   wifiManager.addParameter(&custom_mqtt_topic_sub);
203
204   //caso nnao reconhece uma rede Wifi, cria a rede interna Monografia
205   //espera receber dados de uma rede
206   if (!wifiManager.autoConnect("Monografia", "monografia123")) {
207     imprimirSerial(true, "Falha ao se conectar");
208     delay(3000);
209     ESP.reset(); //reinicia o ESP
```

```
210       delay(5000);
211     }
212     //conexao bem sucedida
213
214     imprimirSerial(true, "Conectado!! :)");
215
216     //Lendo os parametros
217     strcpy(MQTT_server, custom_mqtt_server.getValue());
218     strcpy(MQTT_porta, custom_mqtt_port.getValue());
219     strcpy(MQTT_usuario, custom_mqtt_user.getValue());
220     strcpy(MQTT_senha, custom_mqtt_pass.getValue());
221     strcpy(MQTT_topico, custom_mqtt_topic_sub.getValue());
222
223     //Salvando os parametros informados na tela web do WiFiManager
224     if (precisaSalvar) {
225       imprimirSerial(true, "Salvando as configuracoes");
226       DynamicJsonBuffer jsonBuffer;
227       JsonObject& json = jsonBuffer.createObject();
228       json["MQTT_server"] = MQTT_server;
229       json["MQTT_porta"] = MQTT_porta;
230       json["MQTT_usuario"] = MQTT_usuario;
231       json["MQTT_senha"] = MQTT_senha;
232       json["MQTT_topico"] = MQTT_topico;
233
234       File configFile = SPIFFS.open("/config.json", "w");
235       if (!configFile) {
236         imprimirSerial(true, "Houve uma falha ao abrir o arquivo de
              configuracao para incluir/alterar as configuracoes.");
237       }
238       json.printTo(Serial);
239       json.printTo(configFile);
240       configFile.close();
241     }
242
```

```
243   imprimirSerial(false, "IP: ");
244   imprimirSerial(true, WiFi.localIP().toString());
245
246   //Informando ao client do PubSub a url do servidor e a porta.
247   int portaInt = atoi(MQTT_porta);
248   client.setServer(MQTT_server, portaInt);
249   client.setCallback(retorno);
250
251   //Obtendo o status do pino antes do ESP ser desligado
252   lerStatusAnteriorPino();
253 }
254
255 //Funcao de repeticao (sera executado INFINITAMENTE ate o ESP ser
      desligado)
256 void loop() {
257   //cria um servidor para conectar ao Cloud-MQTT
258   WiFiClient espClient;                                    //Instancia
        do WiFiClient
259   PubSubClient client(espClient);                          //Trocando
        para a instancia do Publisher
260
261   mede_corrente(); //chama a funcao que medira o valor da corrente
262
263   tempoTotal=millis(); //variaveis utilizadas para o tempo de envio
264   tempoEnvio=tempoTotal-tempo;
265
266 //ENVIA O VALOR DO CONSUMO PARA O BANCO DE DADOS
267
268 //realiza o envio a cada 30 minutos
269         if (tempoEnvio>1800000){
270           espClient.stop; //para o servidor do NodeMCU com o Cloud-
                MQTT
271
272           WiFiClientSecure planilha; //Cria um servidor de Wifi para
```

```
                conectar na planilha
273             WiFi.mode(WIFI_STA);//Habilita o modo estacao
274             WiFi.begin("Monografia", "123456");//Conecta na rede
275
276             if (client.connect("docs.google.com", 443) == 1){//Tenta se
                    conectar ao servidor do Google docs na porta 443 (HTTPS
                )
277             tempo=millis();
278              String aux1_consumo;
279                int aux_consumo=consumo*100000; //vamos considerar 5
                       casas decimais
280                int i=0;
281                int a=1;
282                aux1_consumo="1"; //variavel auxiliar que armazena o
                       valor do consumo como String
283                while (i<7){
284                   int num =aux_consumo/(100000000/a);
285                   if (num == 9){
286                      aux1_consumo+="9";
287                    }
288                    else if (num==8){
289                      aux1_consumo+="8";
290                      }
291                      else if (num==7){
292                      aux1_consumo+="";
293                      }
294                      else if (num==6){
295                      aux1_consumo+="6";
296                      }
297                      else if (num==5){
298                      aux1_consumo+="5";
299                      }
300                      else if (num==4){
301                      aux1_consumo+="4";
```

```
302                     }
303                     else if (num==3){
304                     aux1_consumo+="3";
305                     }
306                     else if (num==2){
307                     aux1_consumo+="2";
308                     }
309                     else if (num==1){
310                     aux1_consumo+="1";
311                     }
312                     else aux1_consumo+="0";
313                 aux_consumo=aux_consumo%(100000000/a);
314                 a=a*10;
315                 i=i+1;
316             }
317             a=1;
318             i=0;
319         String envia = auxiliar;//Atribuimos a String auxiliar na
               nova String que sera enviada
320         envia += aux1_consumo;
321         envia +="&entry.441927641=";
322         envia += equipamento;
323         envia += "&submit=Submit HTTP/1.1";//Completamos o metodo
               GET para nosso formulario.
324         client.println(toSend);//Enviamos os valores para o
               servidor
325         client.println("Host: docs.google.com");
326         client.println();
327         client.stop();
328         Serial.println("Dados enviados.");
329         consumo=0;
330
331         planilha.stop; //termina a conex~ao com o Google Docs
332         } //fim do if da porta de conexao
```

```
333           }
334           else{
335             Serial.print("Ainda nao enviou");
336           }
337
338    if (!client.connected()) {
339      reconectar();
340    }
341    client.loop();
342  }
```

Referências Bibliográficas

ABESCO. 2015. *Desperdício de energia gera perdas de 12,6 bilhões.* Disponível em <http://www.abesco.com.br/pt/novidade/desperdicio-de-energia-gera-perdas-de-r-126-bilhoes >. Acessado em 28 de Outubro de 2017.

ALLEGRO MICROSYTEMS, lala. 2010. *Sensor acs712*.

Alliance®, Wi-Fi. 2018. *Who we are*.

BARROS, MARCELO. 2015. *Mqtt - protocolos para iot*.

BORENSTEIN et al, C.R. 1999. *Regulação e gestão competitiva no setor elétrico brasileiro*. Sagra Luzatto.

CUNHA, LÍVIA. 2009. *Relé e contatores*.

Diniz. 2006. *Internet das coisas*.

EULER, GUILHERME. 2008. *ConheÇa o google docs*.

GRUMAN, Galen. 2014. *Iot é um grande e confuso campo à espera de explodir.*

IBGE. 2010. *Censo 2010*.

Instituto de física da UFRGS, lala. 2017. *O campo magnético*.

LIGHT. 2017.

MARTINS et al, André.RS. 1999. *Eficiencia energética: integrando usos e reduzindo desperdícios.* ANEEL/ANP.

MORAIS, JOSE. 2017. *Banco de dados com google planilhas – esp*.

MURATORI e DAL BO, LALA. 2013. *Automação residencial conceitos e aplicações*.

RABELO, MARCO. 2017. *Esp8266 com broker mqtt*.

ROVERE, lala. 2016. *Protótipo de um sistema inteligente de monitoramento do consumo de energia elétrica em uma residência*.

SILVA, lala. 2008. *Manual da tecnologia wireless*.

SMIL, Vaclav. 2004. World history and energy. *Encyclopedia of energy, c. cleveland*, 1–13.

THOMSEN, ADILSON. 2016. *Como programar o nodemcu com a ide do arduino*.

Velte et al., Anthony T. 2010. *Cloud computing: a practical approach*.

Wikipedia. 2018. *Sensor de efeito hall*.