UNIVERSIDADE DO ESTADO DO AMAZONAS ESCOLA SUPERIOR DE TECNOLOGIA

FELIPE LUNIERE DOS SANTOS

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UM PROTÓTIPO DE ADAPTADOR DE TOMADA INTELIGENTE PARA INTERNET DAS COISAS BASEADO NO MICROCONTROLADOR ESP8266.

Manaus

2022

FELIPE LUNIERE DOS SANTOS

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UM PROTÓTIPO DE ADAPTADOR DE TOMADA INTELIGENTE PARA INTERNET DAS COISAS BASEADO NO MICROCONTROLADOR ESP8266.

Pesquisa desenvolvida durante a disciplina de Trabalho de conclusão de curso II e apresentada à banca avaliadora do Curso de Engenharia Eletrônica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas como pré-requisito para a obtenção do título de Engenheiro em Eletrônica.

Orientador: Jozias Parente de Oliveira

Manaus

2022

Universidade do Estado do Amazonas – UEA
Escola Superior de Tecnologia – EST

Reitor:
Cleinaldo de Almeida Costa, Dr.

Vice-Reitor:
Cleto Cavalcante de Souza Leal, Me.

Diretora da Escola Superior de Tecnologia:
Ingrid Sammyne Gadelha Figueiredo, Me.

Coordenador do Curso de Engenharia:
Bruno da Gama Monteiro Me.

Banca Avaliadora composta por: *Data da defesa: 21/01/2022*
Prof. Jozias Parente de Oliveira, Dr. (Orientador)
Prof. Fábio de Sousa Cardoso, Dr. (Avaliador 1)
Prof. Bruno da Gama Monteiro Me. (Avaliador 2)

CIP – Catalogação na Publicação

Santos, Felipe Luniere

Desenvolvimento e implementação de um protótipo de adaptador de tomada inteligente para internet das coisas baseado no microcontrolador esp8266 / Felipe Luniere dos Santos; [orientado por] Prof. Jozias Parente de Oliveira – Manaus: 2021.
55 f. p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Eletrônica). Universidade do Estado do Amazonas, 2021.

1. Baixo Custo, 2. Internet das Coisas
3. ESP8266, 4. Consumo em tempo real, 5. *Blynk*.
I. Oliveira, Jozias Parente.

FELIPE LUNIERE DOS SANTOS

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE UM PROTÓTIPO DE ADAPTADOR DE TOMADA INTELIGENTE PARA INTERNET DAS COISAS BASEADO NO MICROCONTROLADOR ESP8266.

Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia Eletrônica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro em Eletrônica.

Nota obtida: _9,6_ (nove pontos e seis décimos)

Aprovada em _21_/_01_/_2022.

Área de concentração: Sistemas embarcados, Internet das Coisas.

BANCA EXAMINADORA

Orientador: Jozias Parente de Oliveira, Dr.

Avaliador: Fábio de Sousa Cardoso, Dr.

Avaliador: Bruno da Gama Monteiro Me.

Manaus

2022

DEDICATÓRIA

Dedico este trabalho aos meus pais, ao meu querido coordenador no trabalho, a minha esposa e a todos os meus familiares e amigos que direta ou indiretamente me apoiaram nesta caminhada. Meu muito obrigado.

AGRADECIMENTOS

A Deus por me manter saudável e firme em meus objetivos. À minha família que sempre me apoiou e confiou em minhas escolhas. À minha esposa pelo incentivo em momentos que tive vontade de desistir. Ao meu orientador pela paciência, bons direcionamentos e incentivo. Ao professor da disciplina de TCC pelas instruções de formatação do trabalho escrito**.**

RESUMO

Este documento apresenta o projeto de desenvolvimento de um protótipo de baixo custo de um sistema de um protótipo de adaptador de tomada inteligente para internet das coisas baseado no microcontrolador ESP8266. O intuito desse protótipo é dar novos meios de economia de energia elétrica ao consumidor. A construção do protótipo é baseada na tecnologia do microcontrolador ESP8266 onde um sensor de corrente não invasivo verifica o consumo em tempo real de um equipamento conectado a ele e, através do processamento e envio de dados para o aplicativo *Blynk*, o consumidor conseguirá acompanhar todo o gasto com o consumo de energia elétrica em tempo real. Os resultados obtidos comprovam que é possível o monitoramento de forma fácil e em tempo real, com um dispositivo de baixo custo, fazendo com quem o protótipo seja uma ferramenta de conscientização pela economia de energia, mudando o cotidiano das pessoas, transformando o dispositivo em uma solução de eficiência energética na área de Internet das coisas.

Palavras-chaves: Baixo custo; Internet das coisas; ESP8266; consumo em tempo real; *Blynk*.

ABSTRACT

This document presents the project to develop a low-cost prototype of a prototype smart plug adapter system for the internet of things based on the ESP8266 microcontroller. The purpose of this prototype is to provide new means of saving electricity to the consumer. The construction of the prototype is based on the ESP8266 microcontroller technology where a non-invasive current sensor verifies the consumption in real time of an equipment connected to it and, through the processing and sending of data to the *Blynk* application, the consumer will be able to follow all the spent on electricity consumption in real time. The results obtained prove that it is possible to monitor easily and in real time, with a low-cost device, making the prototype an awareness tool for energy savings, changing people's daily lives, transforming the device into a energy efficiency solution in the Internet of Things area.

Key-words: *Internet of things; ESP8266; consumption in real time;Blynk.*

LISTA DE FIGURAS

SUMÁRIO

INTRODUÇÃO

A Internet das Coisas (IoT) é um termo criado por Kevin Ashton, um pioneiro tecnológico britânico que concebeu um sistema de sensores onipresentes conectando o mundo físico à *internet*, enquanto trabalhava em identificação por rádio frequência (RFID). Embora as coisas, a *internet* e a conectividade sejam os três componentes principais da *internet*, o valor está no fechamento das lacunas entre os mundos físico e digital em sistemas com recursos de reforço e aprimoramento automáticos (AWS, 2019).

A *IoT* cria esses sistemas ao conectar coisas, animadas ou inanimadas, à *internet* com identificadores exclusivos que oferecem contexto, o que proporciona visibilidade à rede, aos dispositivos e ao ambiente. Com todos esses dados possibilitados pela sua implementação e usando análise avançada, a *IoT* pode nos oferecer informações importantíssimas sobre o nosso mundo, por exemplo: qual eletrodoméstico consome mais energia em sua casa, quanto dispositivos em modo *stand by* consomem por mês de energia, qual a intensidade da luz necessária para iluminar um ambiente (AWS, 2019).

Visando minimizar o desperdício de energia elétrica efetivo é necessário realizar o monitoramento do consumo de energia elétrica em tempo real para que o consumidor saiba a quantidade de quilowatts consumidos por cada um de seus equipamentos eletrônicos, identificando os que consomem mais energia tendo a possibilidade de mudar seus hábitos para evitar desperdícios.

Uma das opções que auxiliam o consumidor a monitorar o consumo de sua energia elétrica são os adaptadores de tomada inteligentes. A maioria deles realizam apenas o controle para desligar e ligar a carga remotamente. As tomadas inteligentes que monitoram o consumo em tempo real que foram pesquisadas ainda tem um custo inacessível para a maior parte da população.

Esta pesquisa tem por hipótese a ideia de que é possível monitorar remotamente, via aplicativo em dispositivo móvel, os dados de consumo de energia elétrica de eletrodomésticos

em uma residência em tempo real e controlar tais dispositivos por meio de um adaptador de tomada inteligente de baixo custo conectado em rede baseada na *Internet* das Coisas utilizando o microcontrolador ESP8266.

Esse adaptador fornece ao usuário o controle de "ligar e desligar" dispositivos conectados, aquisição de dados de consumo destes dispositivos e transmissão desses dados via *internet* para análise através de gráficos de consumo. O enfoque deste trabalho se dá na construção do protótipo e da rede necessária para fazer com que os dispositivos domésticos se conectem à *internet* aproveitando toda estrutura já existente nas residências.

A pesquisa justifica-se por oferecer uma alternativa para o monitoramento de cargas residenciais em tempo real através de um adaptador de tomada inteligente de baixo custo que mostrar dados de consumo e controlar eletrodomésticos via aplicativo de dispositivo móvel, oferecendo a vantagem de ser "*plug and play*", isto é, não altera a instalação elétrica existente.

Para construir esse dispositivo foi necessário aplicar os conhecimentos adquiridos nas disciplinas de Linguagem de Programação, Sistemas Embarcados, Propagação de Antenas, Sensores, Eletrônica Analógica e Digital durante o curso de Engenharia Eletrônica.

Este trabalho está divido em quatro capítulos: Referencial Teórico; Métodos e Materiais; Implementação e Análise e Interpretação dos Resultados.

Capítulo 1 – Referencial Teórico: Tem a finalidade de descrever os conceitos fundamentais das tecnologias envolvidas neste projeto, sensores e o microcontrolador do circuito de controle.

Capítulo 2 – Métodos e Materiais: Tem a finalidade de descrever quais foram os passos necessários para ser dado o início deste projeto e embasar o leitor dos métodos que foram utilizados para se obter os resultados.

Capítulo 3 – Realização do projeto: Tem a finalidade de descrever detalhadamente o desenvolvimento passo-a-passo do circuito eletrônico e do sistema completo.

Capítulo 4 – Resultados Obtidos: Tem a finalidade de descrever os testes realizados e os resultados obtidos decorrentes dos testes do protótipo com cargas, gerando subsídios para a conclusão que é apresentada no final deste trabalho.

1 REFERENCIAL TEÓRICO

Este capítulo exibe toda a fonte de conhecimento teórico necessária para compreender o trabalho em questão. Ele está estruturado em 4 etapas:

a) *Hardware*;
b) Potência elétrica;
c) *Software*;
d) Aplicação *Web/Android/iOS*.

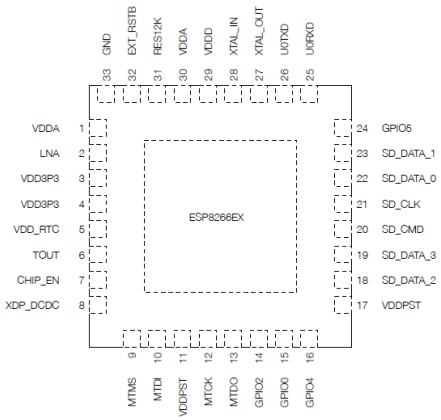
1.1 *HARDWARE*

Nesta seção será feita uma pesquisa exploratória para a aquisição dos componentes com melhor custo-benefício para montagem do protótipo.

1.1.1 ESP8266

O ESP8266 é um sistema em um chip (SoC), manufaturado pela empresa chinesa Espressif. Ele consiste de uma unidade microcontroladora, um transceptor *Wi-Fi*, portas *Input/Output* (I/O) e entradas analógicas (PIETER, [s.d]). O microchip no encapsulamento *Quad Flat No leads (*QFN*)* pode ser visto na Figura 1:

Figura 1 - ESP8266 no encapsulamento QFN.



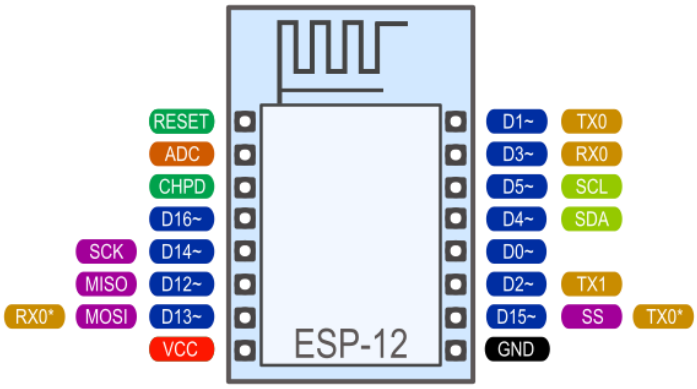
Fonte: (ESPRESSIF SYSTEMS, 2018).

Esse dispositivo possui arquitetura do tipo *Reduced Instruction Set Computer* (RISC) 32 bits com *clock* de 80MHz à 160MHz. Ele ainda possui saídas *Serial Peripheral Interface*

(SPI), *Inter-Integrated Circuit* (I2C), *Universal Asynchronous Receiver/Transmitter* (UART) e *Pulse width modulation* (PWM), além de um *Analog Digital Converter* (ADC). Possui 32 KBytes de *Random Acess Memory* (RAM) para instruções e 96 KBytes de RAM para dados (ESPRESSIF SYSTEMS, 2018).

A configuração dos pinos pode ser vista na Figura 2 (ESPRESSIF SYSTEMS, 2018).

Figura 2 - ESP-12 pinos.



Fonte: (OOSTENVELD [s.d], p.1).

Para comunicação serial o ESP8266 tem dois UARTS de *hardware* (portas seriais): UART0 nos pinos 1 e 3 (TX0 e RX0 respectivamente). O UART0 também possui controle de fluxo de *hardware* nos pinos 15 e 13 (RTS0 e CTS0 respectivamente). Esses dois pinos também podem ser usados como pinos TX0 e RX0 alternativos (PIETER, 2017).

Em relação as entradas e saídas digitais para definir a função de um pino digital ao programar deve-se usar a função *pinMode*(pino, modo); onde pino é o número do *General Purpose* Input/Output (*GPIO)* e o modo pode ser *INPUT*, que é o padrão, *OUTPUT* ou *INPUT_PULLUP* para habilitar os resistores pull-up integrados para *GPIO*-15. Para habilitar o resistor *pull-down* para GPIO16, você deve usar *INPUT_PULLDOWN*_16 (PIETER, 2017).

Dependendo da placa de desenvolvimento usada, pode-se ter um mapeamento de pinos diferente. Para endereçar um pino na placa de desenvolvimento NodeMCU, por exemplo, usa-se *pinMode* (D5, *OUTPUT*) (PIETER, 2017).

Para definir um pino de saída com nível lógico alto (3,3V) ou baixo (0V), usa-se digitalWrite (pino, valor); onde pino é o pino digital e valor 1 ou 0 (ou *HIGH* e *LOW*) (PIETER, 2017).

Para ler os dados enviados por um sensor deve-se usar a função *analogRead* (A0) para obter a tensão analógica na entrada analógica. (0 = 0V, 1023 = 1,0V) (PIETER, 2017).

O *ESP* também pode usar o *ADC* para medir a tensão de alimentação (*VCC*), para fazer isso, usa-se a função *ADC_MODE* (*ADC_VCC*) no topo do seu esboço, e use *ESP.getVcc* () para obter a voltagem. Ao usá-la para ler a tensão de alimentação, nada poderá ser conectado ao pino analógico (PIETER, 2017).

Embora os dispositivos agora estejam fisicamente conectados (seja por fios reais (*Ethernet*) ou por ondas de rádio (*WiFi*), eles ainda não podem se comunicar de fato, porque não têm como saber para quem enviar a mensagem. É aí que entra o *Internet Protocol* (*IP*). Cada dispositivo na rede tem um endereço IP pessoal. O *Dynamic Host Configuration Protocol* (DHCP) garante que esses endereços sejam exclusivos. Isso significa que pode-se enviar uma mensagem para um endereço específico (PIETER, 2017).

Existem duas versões do protocolo da *Internet*: IPv4 e IPv6. O IPv6 é uma versão aprimorada do IPv4 e tem muito mais endereços do que o IPv4. O IPv4 ainda é o mais utilizado e ele será utilizado neste trabalho, pois ainda é o mais ultilizado na maioria das *Local Area Network* (LAN) (PIETER, 2017).

O endereço IP consiste em 4 números, por exemplo “192.168.1.5” é um endereço IPv4 válido. Na verdade, consiste em duas partes: a primeira parte é 192.168.1, este é o endereço da rede local. O último dígito, 5 neste caso, é específico do dispositivo (PIETER, 2017).

Usando endereços IP, podemos encontrar o ESP8266 na rede e enviar mensagens para ele. O *ESP* também pode localizar o nosso computador ou o nosso telefone, caso conheça os respectivos endereços IP (PIETER, 2017).

O ESP8266 pode operar em três modos diferentes: estação WiFi, ponto de acesso WiFi e ambos ao mesmo tempo. O código para se conectar a um ponto de acesso sem fio é relativamente simples: insira o *Service Set Identifier* (SSID) e a senha da rede à qual deseja se conectar e chame a função WiFi.*begin*. Em seguida, aguarde a conexão ser concluída, e o ESP8266 agora estará conectado à rede local, o código de exemplo pode ser visto na Figura 3 (PIETER, 2017).

Figura 3 - Código de exemplo para conectar ESP à uma LAN.

```
#include <ESP8266WiFi.h>        // Include the Wi-Fi library

const char* ssid     = "SSID";         // The SSID (name) of the Wi-Fi network you want to connect to
const char* password = "PASSWORD";     // The password of the Wi-Fi network

void setup() {
  Serial.begin(115200);         // Start the Serial communication to send messages to the computer
  delay(10);
  Serial.println('\n');

  WiFi.begin(ssid, password);             // Connect to the network
  Serial.print("Connecting to ");
  Serial.print(ssid); Serial.println(" ...");

  int i = 0;
  while (WiFi.status() != WL_CONNECTED) { // Wait for the Wi-Fi to connect
    delay(1000);
    Serial.print(++i); Serial.print(' ');
  }

  Serial.println('\n');
  Serial.println("Connection established!");
  Serial.print("IP address:\t");
  Serial.println(WiFi.localIP());         // Send the IP address of the ESP8266 to the computer
}

void loop() { }
```

Fonte: (PIETER, 2017).

Para configurar o ESP8266 como um ponto de acesso, para permitir que outros dispositivos como smartphones ou laptops se conectem a ele, você pode usar a função softAP exemplificada na Figura 4 (PIETER, 2017).

Figura 4 - Usando ESP como ponto de acesso.

```
#include <ESP8266WiFi.h>        // Include the Wi-Fi library

const char *ssid = "ESP8266 Access Point"; // The name of the Wi-Fi network that will be created
const char *password = "thereisnospoon";   // The password required to connect to it, leave blank for an open network

void setup() {
  Serial.begin(115200);
  delay(10);
  Serial.println('\n');

  WiFi.softAP(ssid, password);            // Start the access point
  Serial.print("Access Point \"");
  Serial.print(ssid);
  Serial.println("\" started");

  Serial.print("IP address:\t");
  Serial.println(WiFi.softAPIP());        // Send the IP address of the ESP8266 to the computer
}

void loop() { }
```

Fonte: (PIETER, 2017).

O ESP8266 pode trabalhar em 5 modos de energia: Ativo, *Modem-sleep*, *Deep-sleep*, *light-sleep* e desligado. No modo ativo ele pode receber e transmitir dados, pois o chip de radiofrequência está ativo, no modo *Modem-sleep* a *Central Processing Unit* (CPU) está

operando e o Wi-Fi e o chip de radiofrequência estão desabilitados, no modo *light-sleep* a CPU e todos os periféricos estão hibernando esperando que algum evento requisite seus serviços e no modo *Deep-sleep* somente o *Real Time Clock (*RTC) (ESPRESSIF SYSTEMS, 2018).

O consumo de energia pode ser verificado na Tabela 1:

Tabela 1 - Modos de consumo de energia do ESP8266

Modo de energia	**Descrição**	**Consumo de energia**
Ativo	Wi-Fi TX Wi-Fi RX	Entre 56mA e 170mA
Modem-sleep	CPU operando	15mA
Deep-sleep	Somente RTC operando	20uA
Light-sleep	-	0,9mA
Desligado	-	0,5uA

Fonte: Autoria própria.

Existem muitos módulos diferentes disponíveis, módulos autônomos como a série ESP-XX da AI Thinker, ou placas de desenvolvimento completas como o NodeMCU DevKit ou o WeMos D1. Placas diferentes podem ter pinos diferentes não utilizados, antenas Wi-Fi diferentes ou uma quantidade diferente de memória flash na placa (PIETER, 2017).

1.1.2 Módulo ESP8266 D1 Mini - Wemos D1 Mini Wifi

A placa de desenvolvimento Wemos D1 Mini WiFi da Figura 5 é baseada no microcontrolador ESP8266 12F para utilização em projetos embarcados e projetos voltados à *internet* das coisas (IoT). Com tamanho reduzido, possui botão de reset e 11 pinos de entrada/saída. Pode ser programado utilizando a *Integrated Development Environment* (IDE) do Arduino.

Figura 5 - Placa Wemos D1 Mini Pro

Fonte: (SMART PROJECTS, [s.d]).

Placa Wemos D1 Mini Wifi ESP8266 12F - Especificações:

a) Microcontrolador: ESP8266 12F;
b) Wireless padrão 802.11 b/g/n;
c) Antena embutida;
d) Conector micro-usb;
e) Modos de operação: STA/AP/STA+AP;
f) Suporta 5 conexões TCP/IP;
g) Portas GPIO: 11 GPIOs com funções de PWM, I2C, SPI etc;
h) Tensão de Operação: 4.5 - 9V;
i) Taxa de transferência: 110-460800bps;
j) Suporta *upgrade* remoto de *firmware*;
k) Conversor analógico digital (ADC);
l) Distância entre pinos: 2.54mm;
m) Dimensões: 39 x 25 x 3mm.

Na

Tabela **2** é descrita a função de cada pino da placa Wemos D1.

Tabela 2 - Função dos Pinos do ESP8266

Pinos	Função	Pinos do ESP8266
TX	TXD	TXD
RX	RXD	RXD
A0	Entrada analógico, máximo de 3.2V	A0
D0	IO	GPIO16
D1	IO, SCL	GPIO5
D2	IO, DAS	GPIO4
D3	IO, 10k Pull-up	GPIO0
D4	IO, 10k Pull-up, BUILTIN_LED	GPIO2
D5	IO, SCK	GPIO14
D6	IO, MISO	GPIO12
D7	IO, MOSI	GPIO13
D8	IO, 10k Pull-down, SS	GPIO15
G	Terra	GND
5V	5V	-
3V3	3.3V	3.3V
RST	Reset	RST

Fonte: (WEMOS, [s.d]).

Esta placa possui uma única entrada analógica, com uma faixa de entrada de 0 a 1,0 V. Se for fornecido 3,3 V, por exemplo, o chip será danificado. O ADC possui resolução de 10 bits (PIETER, 2017).

1.1.3 Sensor de Corrente

Um sensor é um transdutor capaz de converter parâmetros físicos, biológicos ou químicos em sinal elétrico, por exemplo, temperatura, pressão e etc. Sensores podem gerar respostas mensuráveis de uma grandeza física e retornar esses dados em forma digital.

Neste trabalho será utilizado o transformador de corrente toroidal, Figura 6, que funciona baseado nas leis de Faraday (eletromagnetismo), Lenz (indução eletromagnética) e Lei de Ampére (BRITO, 2016).

Figura 6 - Sensor de corrente SCT013-010.

Fonte: YHDC ([s.d], p.1).

A Lei de Faraday mostra que a variação de fluxo magnético entre os instantes de tempo t0 e t1 com a DDP induzida em uma espira, pode ser representada pela equação (1):

$$\in = -\frac{\Delta \emptyset}{\Delta t} \qquad \text{(Equação 1)}$$

Onde ε = força eletromotriz; ΔØ= variação do fluxo magnético e; Δt = variação do tempo (FIGUEIREDO; MARTINS, 2017).

A Lei de Lenz diz que ao aproximar um campo magnético de uma bobina, surge nesta, uma corrente elétrica induzida, que tem um campo oposto a variação do fluxo (FIGUEIREDO; MARTINS, 2017).

A Lei de Ampére comprova matematicamente a existência de um campo magnético em volta de um condutor conduzindo uma corrente elétrica, A Lei de Ampére na sua forma integral, é descrita na equação (2) (FIGUEIREDO; MARTINS, 2017).

$$\oint B \; x \; dl \; = \; \mu_0 X \; I \qquad \text{(Equação 2)}$$

Onde μ_0 é a permeabilidade magnética no vácuo, B é o campo magnético, I é a corrente (FIGUEIREDO; MARTINS, 2017).

Para o cálculo de um campo magnético toroidal, o campo magnético é tangente a circunferência $\oint B \; x \; dl = B(2\pi r)$ e a corrente interior total é Ie $=$ N X I, onde N é o número total de espiras do núcleo toroidal . (FIGUEIREDO; MARTINS, 2017).

$$B(2\pi r) \; = \; \text{N X I} \qquad \text{(Equação 3)}$$

Os transformadores de corrente são construídos com enrolamentos de fios de cobre, revestidos por um esmalte, são constituídos de uma parte primária e uma parte secundário. De forma resumida, um transformador é um dispositivo de corrente alternada que opera rebaixando ou amentando um certo valor de tensão ou corrente em que uma corrente alternada que passa por um condutor induzirá uma corrente no condutor primário e essa corrente induzirá uma corrente no enrolamento secundário que será uma amostra da corrente do enrolamento primário (BRITO, 2016).

Os cálculos de transformação de tensão e corrente em um transformador são dados pelas seguinte equações:

Quando o número de espiras é proporcional à tensão.

$$\frac{E1}{E2} = \frac{N1}{N2} \qquad \text{(Equação 4)}$$

Quando a tensão é inversamente proporcional a corrente.

$$\frac{E1}{E2} = \frac{I2}{I1} \qquad \text{(Equação 5)}$$

O sensor que será utilizado neste trabalho será o SCT013-010, um sensor não invasivo utilizado para aplicações envolvendo automação residencial, ele pode medir corrente AC até 100 A, o sensor fornecerá na saída a corrente, abaixo as especificações do sensor (YHDC, [s.d]).

1. Corrente de entrada: 0-100A;
2. Sinal de saída: Corrente/33mA;
3. Material do Core: Ferrite;
4. Dielétrico: 6000V AC/1min;
5. Plug de saída: 3,5mm;
6. Dimensão abertura: 13 x 13mm;
7. Temperatura de trabalho: -25 a +70°C.

1.2 POTÊNCIA ELÉTRICA

Por definição é a quantidade de trabalho que pode ser executado durante um período de tempo, logo é a rapidez com que um trabalho é executado (SILVA, 2018).

1.2.1 Potência elétrica em Corrente Contínua

Em um circuito de corrente contínua, não há variação da tensão e corrente em função do tempo. Logo a tensão e a corrente entre dois pontos serão constantes. Para os circuitos de corrente contínua, a potência será dada por (SILVA, 2018):

$$P = VI \qquad \text{(Equação 6)}$$

P - Potência em (W)

V - Tensão em (V)

I - Corrente em (A)

1.2.2 Potência elétrica em Corrente Alternada

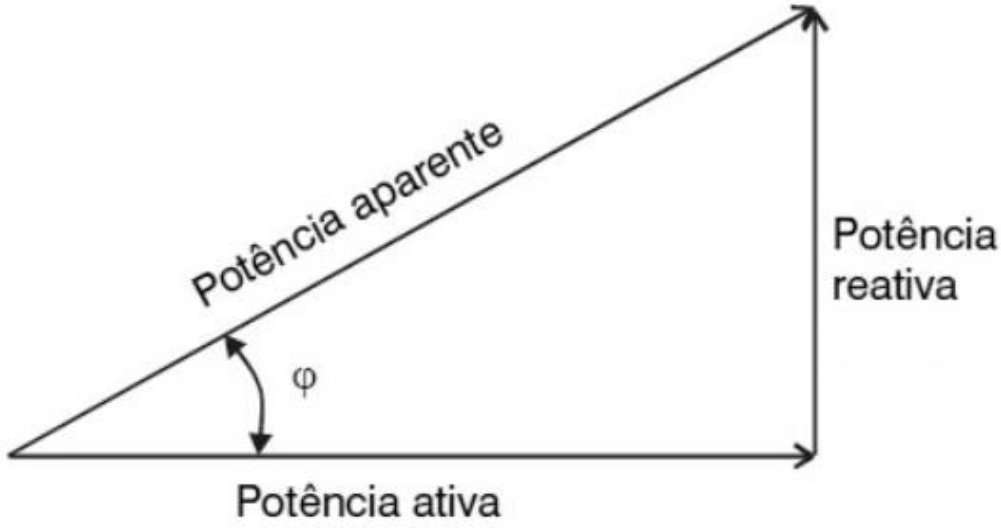
No tópico anterior foi mostrado que a potência é o produto da tensão pela corrente, porém algumas variações ocorrem nos circuitos do tipo corrente alternada. portanto nesses circuitos, os são sinais da corrente e tensão são de forma senoidal, e ficam da seguinte forma (SILVA, 2018):

$$v = V_m seno(\omega t + \phi) \qquad \text{(Equação 7)}$$

$$i = I_m seno(\omega t) \qquad \text{(Equação 8)}$$

Outros termos utilizados para quando circuitos de corrente alternada são: potência ativa, reativa e aparente e fator de potência, esses termos podem ser melhor compreendidos através do triângulo de potência da Figura 7 (SILVA, 2018).

Figura 7 - Triângulo de Potências



Fonte: (SILVA, 2018).

1.2.3 Fator de Potência

Fator de potência indica a eficiência com a qual a energia está sendo usada, para se ter uma compreensão completa precisa-se entender as componentes que o compõem, são elas (SILVA, 2018):

a) potência ativa – energia que realiza o trabalho útil, como luz, calor ou movimento.
b) potência reativa - Energia que não produz trabalho útil, sendo armazenada por componentes indutivos e capacitivos.
c) potência aparente – Energia resultante composta pela potência ativa e reativa. As potências ativa e reativa compõem a rede elétrica diminuindo a real capacidade de transmissão de trabalho útil da rede, por causa da potência reativa presente.

1.2.4 Potência Ativa

É toda potência capaz de produzir trabalho útil, por exemplo: gerar movimento em motores, ligar lâmpadas, transformadores e entre outros equipamentos. Após substituirmos a tensão e corrente na forma senoidal na Equação 1, aplicar operações trigonométricas tem-se a potência ativa que é medida em Watts(W) (SILVA, 2018):

$$P = VIcos(\varphi) \qquad \text{(Equação 9)}$$

1.2.5 Potência Reativa

Quando um circuito tem capacitores ou indutores, haverá a potência reativa e ela será determinada a partir da diferença entre as potências fornecida pela reatância de cada componente. É a energia elétrica que não é capaz de realizar trabalho, no entanto, cargas indutivas utilizam energia elétrica como fonte de força, gerando campos magnéticos para seu funcionamento. A partir do triângulo de potência, pode-se expressar a potência reativa por (SILVA, 2018):

$$Q = S\,sen(\varphi) \qquad \text{(Equação 10)}$$

1.2.6 Potência Aparente

É a soma dos vetores da potência ativa e reativa. Sua unidade de medida é o volt-ampere (VA). Sua forma complexa é dada por (SILVA, 2018):

$$S = P + JQ = VI_{conjugado\ da\ corrente\ complexa} \qquad \text{(Equação 11)}$$

Fazendo uma associação entre o triângulo de potência e o Teorema de Pitágoras podemos determinar a potência aparente como o módulo da soma quadrática da potência ativa e reativa (SILVA, 2018):

$$|S| = \sqrt{P^2} + Q^2 \qquad \text{(Equação 12)}$$

1.3 *SOFTWARE*

Nesta seção serão estudadas as linguagens de programação e IDEs possíveis para o desenvolvimento do *software* que será embarcado no microcontrolador ESP8266.

1.3.1 Linguagem LUA

Lua é uma linguagem de programação que nasceu em 1993 no Instituto Tecgraf de Desenvolvimento de *Software* Técnico-Científico (Tecgraf), na Universidade Católica do Rio

de Janeiro (PUC-Rio), foi criada para ser utilizada em um projeto da Petróleo Brasileiro S.A. (Petrobrás). A linguagem se destaca por ser portável, capaz de descrever dados facilmente, é amigável com a linguagem C e tem sintaxe fácil. Lua pode ser compilada em qualquer plataforma que possua um compilador C (MARTINELLI; SIMONASSI; TAVARES, [s.d]).

O *NodeMCU* é uma plataforma *open source* que disponibiliza uma *Applications Protocol Interface* (API) completa com suporte a *WiFi*, *Transmission Control Protocol/Internet Protocol* (TCP/IP), *User Datagram Protocol* (UDP), *Message Queue Telemetry Transport* (MQTT), I2C, SPI (OLIVEIRA, [s.d]). O *software nodeMCU* é feito em eLua baseado no *SDK* ESP8266 NONOS da Espressif, a maioria do código é escrita em C, a linguagem Lua é utilizada para reunir e organizar as bibliotecas deste *software* (JUNIOR, 2017).

1.3.2 Linguagem C++

C++ é uma linguagem de programação multiplataforma, multi-paradigma e de médio nível, isto é, combina características de linguagens de alto e baixo níveis (Portal GSTI, [s.d]).

A linguagem foi desenvolvida no decorrer da década de 80, com objetivo de expandir os recursos da linguagem C (daí a analogia “++” que remete a incremento, expansão), e a princípio chamava-se “C with Classes”. Já foi conhecido também por apelidos como "Novo C" (Portal GSTI, [s.d]).

Algumas IDE´s para desenvolver em C++:

a) Code::Blocks;
b) CodeLite;
c) C++ Builder;
d) Dev C++;
e) Eclipse;
f) GNAT Programming Studio;
g) Netbeans;
h) Qt Creator.

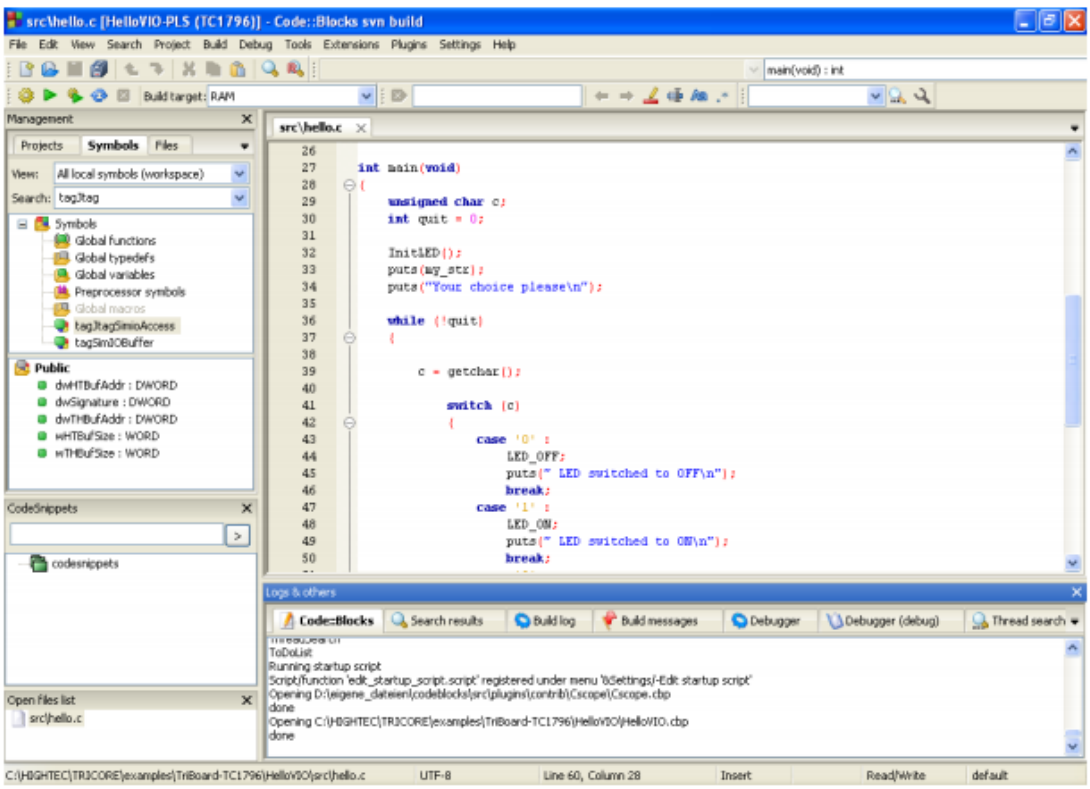
Essencialmente, um programa C++ consiste de uma ou mais partes chamadas funções. Além disso, um programa em C++ deve definir pelo menos uma função chamada *main*. Esta função marca o ponto de início de execução do programa. Programas C++ tem a seguinte estrutura geral (DELGADO, 2018):

```
#include <iostream> using namespace std;
definição de constantes
funções
int main() {
        declaração de variáveis
        ....
        sentenças
        ....
}
```

1.3.3 CODE::BLOCKS IDE

Existem diversos ambientes de desenvolvimento integrado ou *Integrated Development Environment* (IDE) que podem ser utilizados para a programação em linguagem C/C++. Um deles é o Code::Blocks, Figura 8, uma IDE de código aberto e multiplataforma que suporta múltiplos compiladores (BACKES, [s.d]).

Figura 8 - Tela Principal da IDE Code Blocks

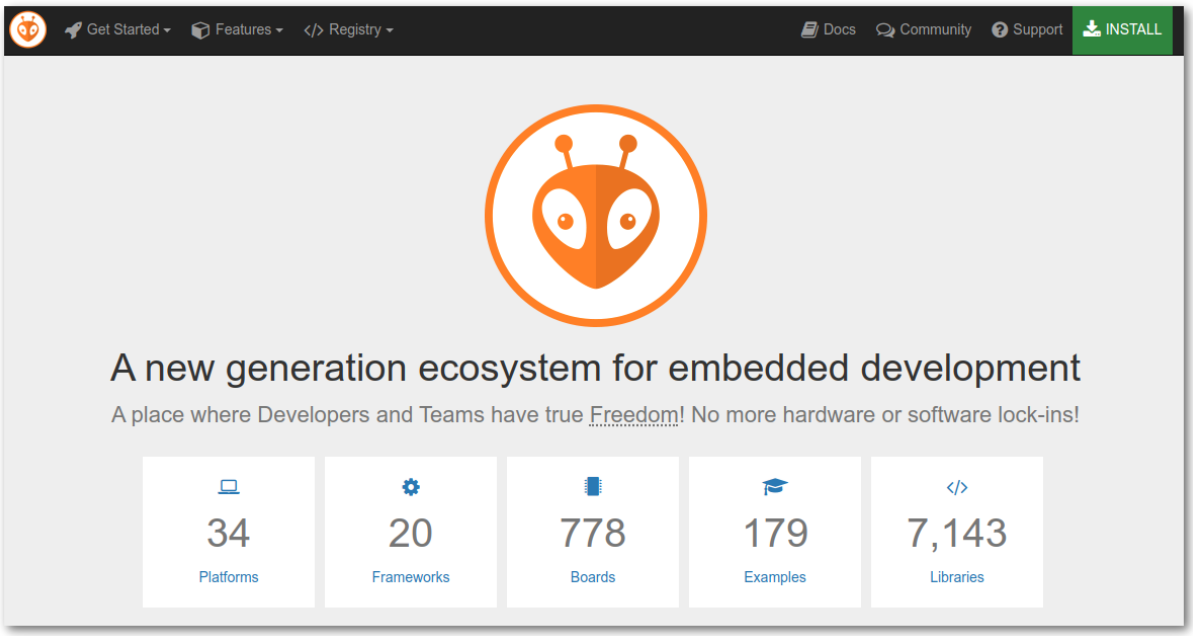


Fonte: (PRASAD; AIBARA; KARMALI, 2014).

1.3.4 PlatformIO

PlatformIO, Figura 9, é um conjunto de ferramentas para desenvolvimento de sistemas embarcados em C/C++. Começou como uma plataforma onde era necessário pagar para se ter todas as funcionalidades, mas em junho de 2019 foi anunciado como *open source* e com todas as funcionalidades gratuitas (PLATFORMIO, [s.d]).

Figura 9 - Tela Principal do PlatformIO



Fonte: (PLATFORMIO, [s.d]).

1.4 APLICAÇÃO *WEB/ANDROID/IOS*

1.4.1 Blynk

Blynk é uma plataforma com aplicativo iOS e Android para controlar arduino, *Raspberry Pi* e outros kits de desenvolvimento pela *internet*. É um painel onde você pode construir uma interface gráfica para o seu projeto por meio de *widgets*, conforme a Figura 10. *Blynk* pode ser utilizado com qualquer *hardware* ou *shield*, este aplicativo foi projetado para *Internet* das coisas para controlar seu *hardware* remotamente além de armazenar dados de sensores e poder mostrá-los ao usuário (DOSHI; SHAH; SHAIKH, 2017).

Figura 10 - *Blynk widgets*

Fonte: (DOSHI; SHAH; SHAIKH, 2017, p.626).

Existem 3 componentes principais na plataforma:

a) Aplicativo: Permite criar interfaces para os projetos por meio de *widgets*;
b) Servidor: Responsável pela comunicação entre o celular e o *hardware*. É possível usar o *Blynk* cloud ou um servidor *Blynk* localmente;
c) Bibliotecas: Permitem a conexão entre todas as plataformas de *hardware* com o servidor e processa todos os comandos de entrada e saída (DOSHI; SHAH; SHAIKH, 2017).

1.5 CÁLCULO DE ERROS DE MEDIÇÃO

De acordo com Toginho Filho e Andrello (2010), fazer uma medida física significa comparar uma quantidade e uma dada grandeza, com outra quantidade da mesma grandeza, definida como unidade ou padrão da mesma. As medições, por mais bem feita que seja, são aproximações em relação a um valor verdadeiro, o qual nunca é conhecido com total certeza. Sabendo isso, para apresentar o resultado de medições de grandezas físicas com confiança, o valor medido deve ser expresso associado ao erro, desvio, ou incerteza da medida. O erro absoluto corresponde à diferença algébrica entre o valor obtido e o valor verdadeiro, conforme a equação 13.

$$Erro\ absoluto = Valor\ medido - Valor\ verdadeiro \qquad \text{(Equação 13)}$$

Em muitos casos, é interessante apresentar valores relativos, quando se exprimem erros de medições. A forma mais usual é apresentar o erro relativo em porcentagens (%). O cálculo utilizado para o erro relativo é feito através da equação 14.

$$Erro\ relativo = Erro\ absoluto/(Valor\ verdadeiro) * 100\% \qquad \text{(Equação 14)}$$

2 MÉTODOS E MATERIAIS

Neste capítulo é mostrado o desenvolvimento do projeto desde a pesquisa teórica em conjunto com as primeiras medições em laboratório, passando pela montagem do protótipo e testes de comunicação entre o sensor e o microcontrolador, até a criação do sistema e sua integração, assim estabelecendo a interface do sistema de monitoramento. O sistema foi desenvolvido nas seguintes etapas.

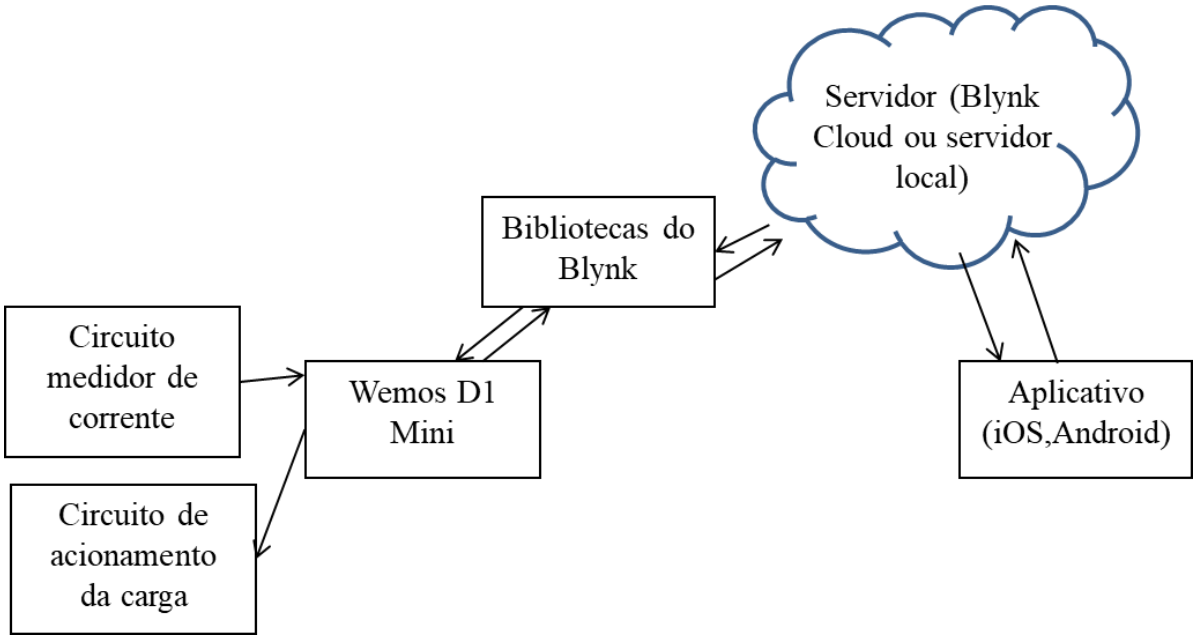
Na primeira etapa, montou-se o protótipo, utilizando uma placa de ensaio (em inglês, *protoboard*), o módulo ESP8266 Wemos D1 mini, o sensor de corrente SCT-013, mini fonte 3V3, nesta mesma etapa foram feitos os testes dos componentes e de integração entre o microcontrolador e o sensor, a fim de confirmar o correto funcionamento dos mesmos.

Na segunda etapa, foi realizado o desenvolvimento e gravação do *firmware* através do *software Code::Blocks*, utilizando a biblioteca “emonlib”, para leitura do valor de corrente medido pelo sensor e transmissão dos dados para que seja realizado o monitoramento em tempo real.

Na terceira etapa foi desenvolvida a parte do código para acionamento *ON/OFF* da carga e integraçao do circuito de acionamento e circuito de medição de corrente.

Na quarta etapa, o protótipo montado em uma placa de fenolite perfurada e o aplicativo foi criado no *Blynk* para receber a leitura dos valores do sensor e controlar o acionamento da carga. Nesta etapa, forma realizados os testes de sistema: verificação do envio das leituras do sensor; de interface: o correto funcionamento dos ícones de interface do usuário e integração: a performance do sistema de integração e de interface funcionando em conjunto e por fim, a análise e interpretação dos resultados obtidos de acordo com as ocorrências para retificação da eficácia do projeto. A Figura 11 ilustra o diagrama em blocos do sistema.

Figura 11 - Diagrama em blocos do sistema.



Fonte: Autoria própria.

Durante a montagem do protótipo de sistema de monitoramento foram utilizados os materiais listados abaixo:

1 capacitor de 10µF.

1 capacitor de 1000µF.

2 resistores de 1KΩ.

1 resistor de 220Ω.

1 mini fonte 3.3V.

1 mini fonte 5V.

1 módulo relé 5V.

1 Microcontrolador ESP8266 Wemos D1 mini.

1 Sensor de corrente SCT-013.

3 REALIZAÇÃO DO PROJETO

O trabalho apresentado é uma pesquisa aplicada, e tem como objetivo a realização de pesquisa experimental que é embasado em um material bibliográfico, de laboratório e de campo. São aplicados os procedimentos técnicos de pesquisa bibliográfica e experimental. É utilizado o método de abordagem hipotético-dedutivo e o método de procedimento monográfico em sua elaboração. Para coleta de dados é utilizada a observação direta intensiva e documentação indireta, e a análise e interpretação de seus dados qualitativos, ocorre globalmente.

Para que seja obtida uma maior compreensão das etapas deste projeto, é necessário que se acompanhe os subcapítulos seguintes:

a) como medir a corrente consumida;
b) desenvolvimento do protótipo;
c) desenvolvimento do *firmware* para aquisição de dados do sensor de corrente;
d) desenvolvimento do *firmware* para ligar e desligar o módulo relé;
e) desenvolvimento do aplicativo na plataforma *Blynk*.

3.1 COMO MEDIR A CORRENTE CONSUMIDA

Para medir a corrente consumida será utilizado um sensor que não precise de contato elétrico com o circuito, para efetuar a medição ele utilizará as propriedades magnéticas da corrente elétrica. Para este trabalho será utilizado o Transformador de corrente SCT-013 *SCT* é a sigla para *Split-core Current Transformer*, ou seja, Transformador de corrente de núcleo dividido. Para fazer a medição da corrente elétrica o SCT-013 possui uma bobina interna em sua estrutura, como podemos ver na Figura 12.

Figura 12 - Bobina interna do sensor de corrente SCT013.

Fonte:(VIDA DE SILÍCIO, 2017).

O sensor de corrente pode medir valores de 0 a 100A, no entanto o microcontrolador só mede tensão. Para o microcontrolador conseguir ler os valores medidos pelo sensor precisa-se usar um resistor de shunt, para que se possa medir a tensão proporcional àquela corrente.

Para calcular o resistor de *shunt* deve-se estabelecer o limite máximo de corrente a ser medido, neste projeto será estabelecido o máximo de 10A, para que se tenha mais precisão nas medições, com isso determina-se a máxima corrente que pode ser medida no primário pelo sensor convertendo essa corrente máxima eficaz para corrente de pico no primário, como na equação 15:

$$I_{p1} = Irms \cdot \sqrt{2} = 14{,}14A \qquad \text{(Equação 15)}$$

Com o valor relação de transformação K sendo 2000, calcula-se na equação 16 a corrente de pico no secundário:

$$Ip2 = \frac{14{,}14}{2000} = 7{,}07mA \qquad \text{(Equação 16)}$$

Para se ter uma melhora na resolução do ESP8266, fazendo com que a corrente fique dentro da faixa de operação de tensão, utiliza-se um resistor de *offset* que divide a tensão de operação do ESP pela metade, portanto para calcular o resistor de *shunt* deve-se utilizar a metade da tensão máxima do ESP que é 1,65V, como visto na equação 17:

$$R_{shunt} = \frac{1,65}{7,07} = 233,38\Omega \qquad \text{(Equação 17)}$$

3.2 DESENVOLVIMENTO DO PROTÓTIPO

O *hardware* pode ser dividido em 3 partes para melhor entendimento:

a) circuito de alimentação;

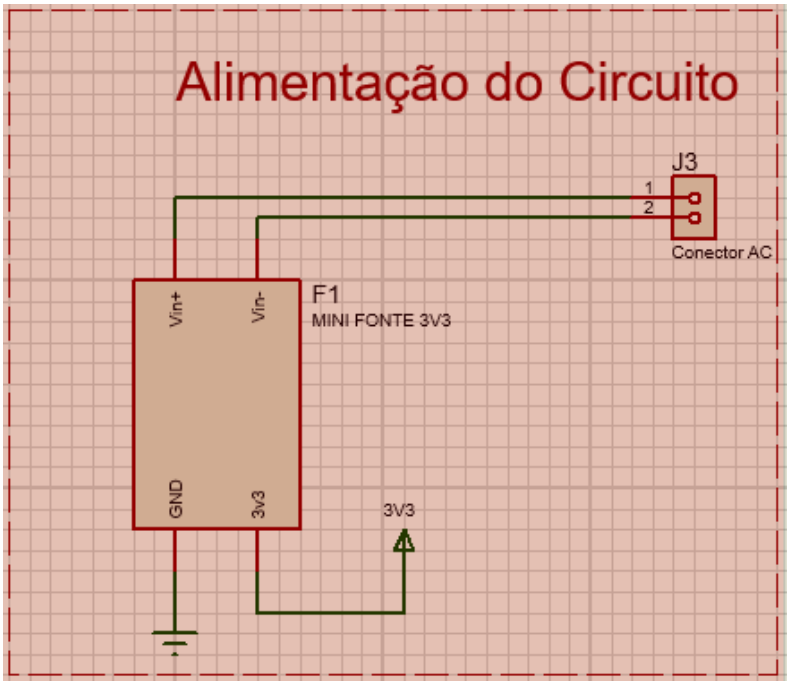
b) circuito de medição do consumo de corrente;

c) circuito de acionamento.

3.2.1 Circuito de alimentação

A alimentação do microcontrolador virá da tomada, 127V ou 220V, em que o adaptador será conectado. Essa tensão passará por uma mini fonte que converte 100~240VAC para 3.3V, para a alimentação do microcontrolador Wemos D1 Mini. Na Figura 13 tem-se o esquemático elétrico feito no *software* Proteus deste circuito:

Figura 13 - Circuito de alimentação do microcontrolador.

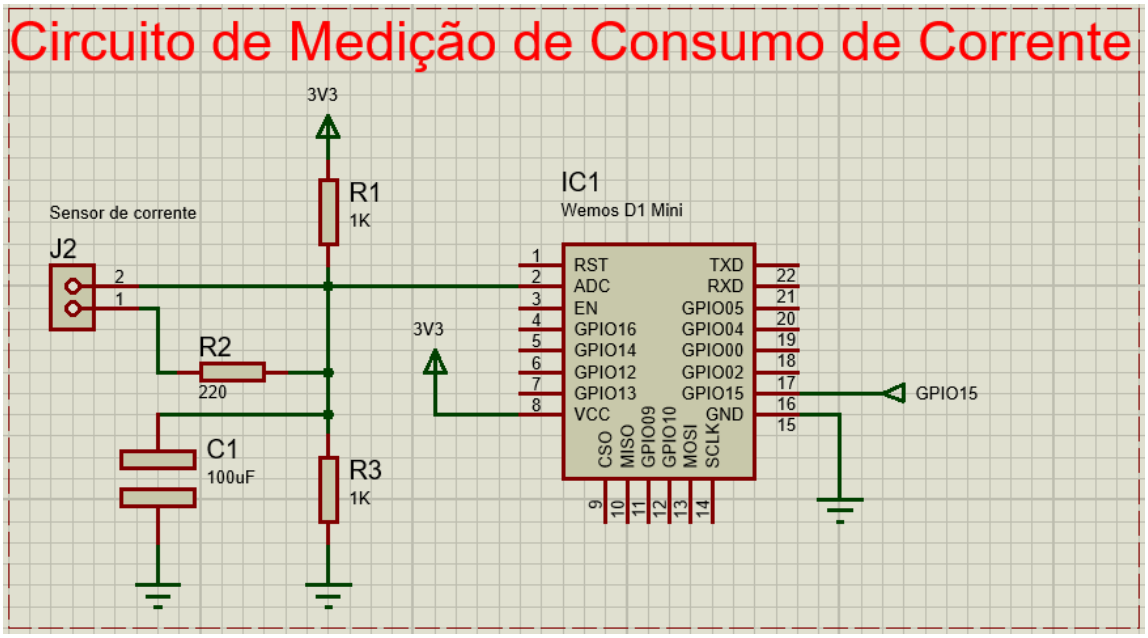


Fonte: Autoria própria.

3.2.2 Circuito de medição do consumo de corrente

Segundo informações do *datasheet*, o sensor SCT-013-000 (100A), tem na saída uma variação de corrente. Para isso será usado um componente adicional: o resistor de *shunt*, que foi calculado na equação 15, para gerar a variação de tensão que precisamos para efetuar a leitura no Wemos D1 Mini. O circuito pode ser visto na Figura 14:

Figura 14 - Circuito de medição de corrente.



Fonte: Autoria própria.

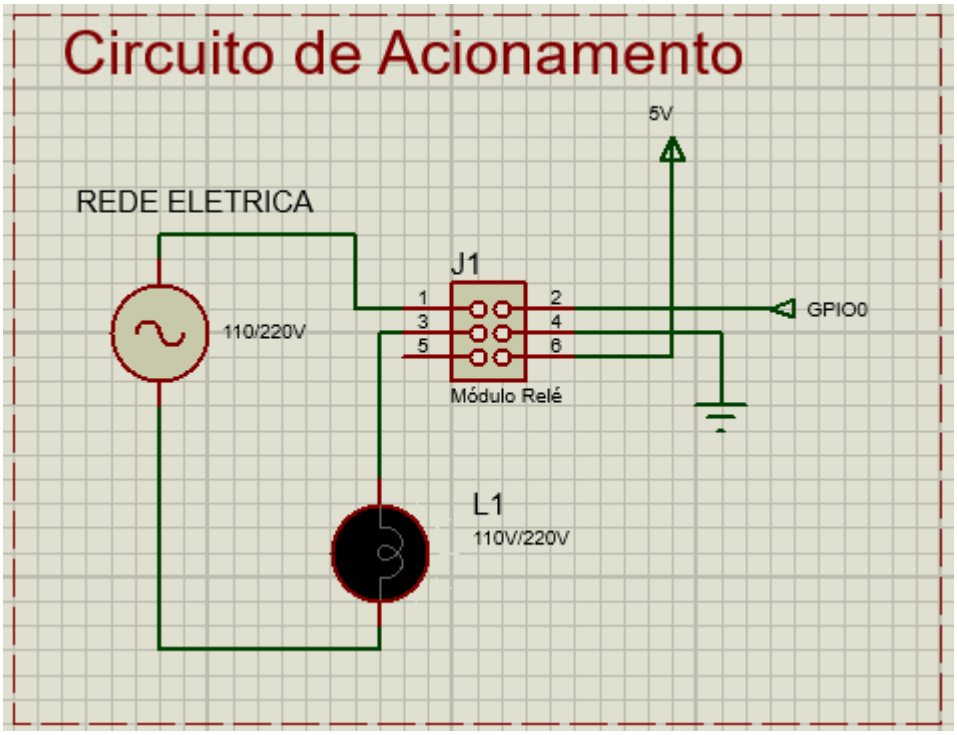
3.2.3 Circuito de acionamento

A alimentação do relé virá da tomada, 110V ou 220V, em que o adaptador será conectado. Essa tensão passará por uma mini fonte que converte 100~240VAC para 5V.

O sinal digital enviado pelo Wemos D1 Mini é aplicado ao contato NF, normalmente fechado, nível lógico 1 irá energizar a bobina do relé e desligar o equipamento e nível lógico 0 irá desenergizar o equipamento e ligar o equipamento.

O módulo relé possui os contatos NF e NA, escolheu-se usar os contatos NF para economizar energia do microcontrolador, pois assume-se que os eletrodomésticos irão passar mais tempo ligados do que desligados. Segue o esquema elétrico do circuito na Figura 15:

Figura 15 - Circuito de acionamento das cargas.

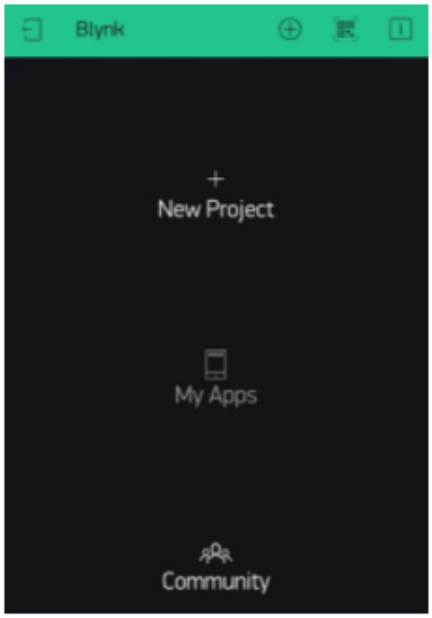


Fonte: Autoria própria.

3.3 DESENVOLVIMENTO DO APLICATIVO NA PLATAFORMA *BLYNK*.

Para desenvolver o aplicativo foi feito o *download* do aplicativo do *blynk* para celular, primeiro foi criado uma conta, após criar a conta será direcionada para a tela da Figura 16.

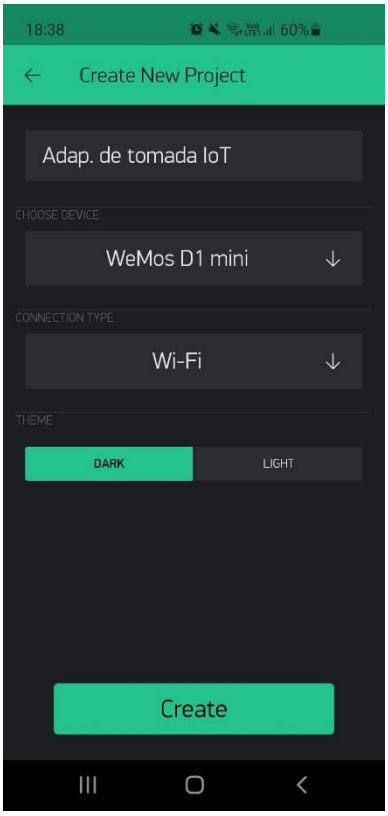
Figura 16 - Tela de novo projeto.



Fonte: (SILVINO, 2021).

Nesta tela, foi selecionada a opção “New Project”, onde será configurado as informações do projeto como: nome do projeto, tipo de conexão e microcontrolador utilizado, como na Figura 17. Ao criar o projeto, um token será enviado para o *e-mail* cadastrado, esse token é utilizado para o microcontrolador se conectar ao servidor do *blynk.*

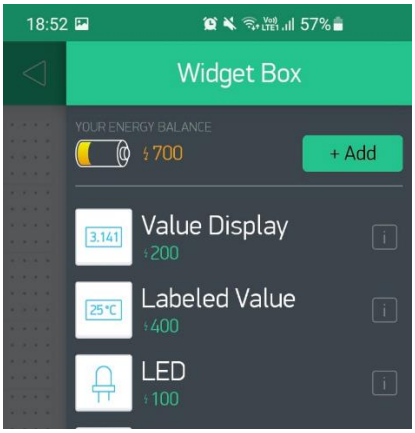
Figura 17 - Tela de criação do projeto.



Fonte: (SILVINO, 2021).

Com o projeto criado, é possível montar o próprio aplicativo apenas adicionando *widgets* como exemplificado na Figura 18.

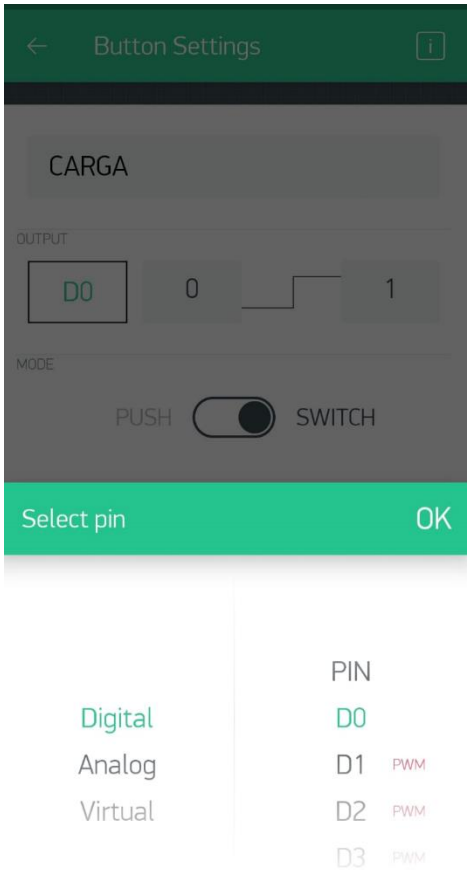
Figura 18 - Tela de inserção de *widgets*.



Fonte: Autoria Própria.

Neste trabalho serão utilizados os *widgets* “*button*”, onde já é possível indicar qual pino físico, como mostrado na Figura 19,irá controlar o acionamento da carga monitorada e 2 “*value display*” para mostrar a corrente consumida e valor em reais gasto até o momento.

Figura 19 - Tela de inserção de botão

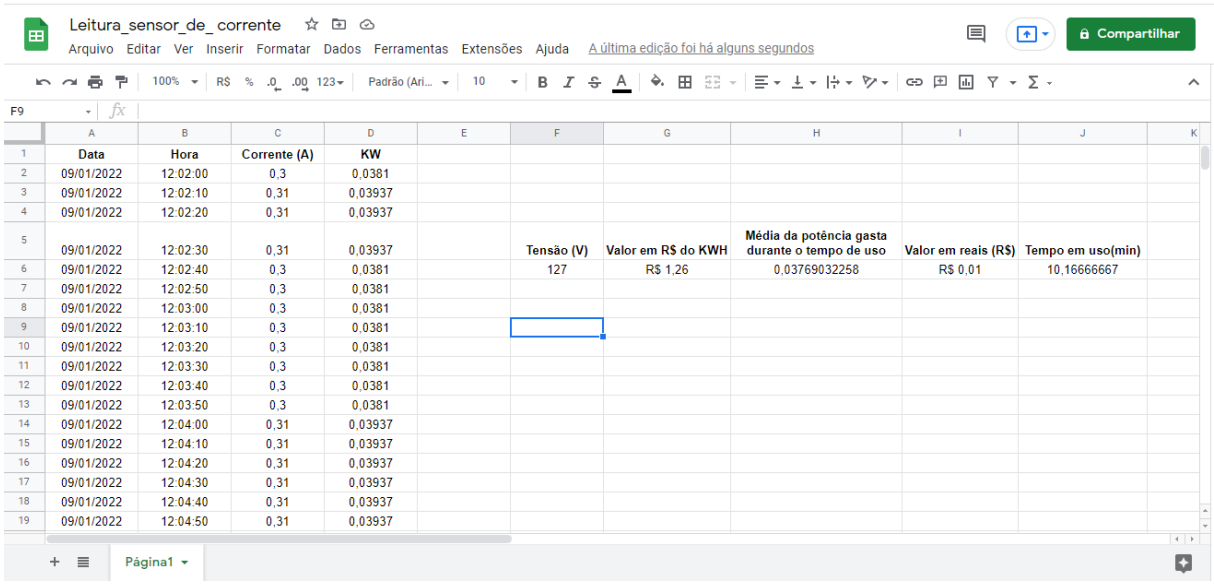


Fonte: Autoria Própria.

3.4 DESENVOLVIMENTO DO *GOOGLE APP SCRIPT*

Para receber os valores de corrente eficaz foi criada a planilha do *google sheets* com a estrutura conforme a Figura 20:

Figura 20 - Planilha com dados do sensor de corrente.



Leitura_sensor_de_ corrente

Arquivo Editar Ver Inserir Formatar Dados Ferramentas Extensões Ajuda A última edição foi há alguns segundos

Compartilhar

	A	B	C	D	E	F	G	H	I	J
1	Data	Hora	Corrente (A)	KW						
2	09/01/2022	12:02:00	0,3	0,0381						
3	09/01/2022	12:02:10	0,31	0,03937						
4	09/01/2022	12:02:20	0,31	0,03937						
5	09/01/2022	12:02:30	0,31	0,03937		Tensão (V)	Valor em R$ do KWH	Média da potência gasta durante o tempo de uso	Valor em reais (R$)	Tempo em uso(min)
6	09/01/2022	12:02:40	0,3	0,0381		127	R$ 1,26	0,03769032258	R$ 0,01	10,16666667
7	09/01/2022	12:02:50	0,3	0,0381						
8	09/01/2022	12:03:00	0,3	0,0381						
9	09/01/2022	12:03:10	0,3	0,0381						
10	09/01/2022	12:03:20	0,3	0,0381						
11	09/01/2022	12:03:30	0,3	0,0381						
12	09/01/2022	12:03:40	0,3	0,0381						
13	09/01/2022	12:03:50	0,3	0,0381						
14	09/01/2022	12:04:00	0,31	0,03937						
15	09/01/2022	12:04:10	0,31	0,03937						
16	09/01/2022	12:04:20	0,31	0,03937						
17	09/01/2022	12:04:30	0,31	0,03937						
18	09/01/2022	12:04:40	0,31	0,03937						
19	09/01/2022	12:04:50	0,31	0,03937						

Página1

Fonte:Autoria própria.

A planilha é atualizada a cada 10 segundos através de um *Google app script* que a cada 10 segundos recebe requisições com o valor da corrente eficaz medida e envia o valor atual gasto em reais pelo equipamento, *Google app script*, que é uma plataforma para o desenvolvimento de aplicativos para os serviços do G *Suite*, baseado no *JavaScript*. Um recurso muito parecido com a programação em *Visual Basic for Applications* (VBA) para os produtos *Office* (META DEV STUDIO,2019).

3.5 DESENVOLVIMENTO DO *FIRMWARE* DO MICROCONTROLADOR

Para desenvolvimento do *firmware* foram usadas bibliotecas para simplificar o desenvolvimento do código.

3.3.1 Bibliotecas

O programa principal possui 6 bibliotecas necessárias para seu funcionamento. Para gerenciar as bibliotecas necessárias para o desenvolvimento do *firmware* utilizou-se a ferramenta *PlatformIO*, é um ecossistema de código aberto para desenvolvimento de IoT com

um sistema de construção de plataforma cruzada, um gerenciador de biblioteca e suporte completo para desenvolvimento de projetos com o ESP8266 (PLATFORMIO, [s.d]).

O ciclo de trabalho utilizando o *PlatformIO* é o seguinte: o usuário escolhe a placa que será utilizada no projeto no arquivo de configuração "platformio.ini", com base na lista de placas listadas nesse arquivo, a ferramenta baixa as bibliotecas necessárias e os instala automaticamente (PLATFORMIO, [s.d]).

A primeira biblioteca "Arduino.h", serve para permitir a utilização de estruturas e funções bem conhecidas do arduino como setup(), loop(), digitalWrite(), Pinmode() e etc com outras plataformas (CERQUEIRA, 2018).

A biblioteca *Open Source* "EmonLib.h" foi feita pela *Open Energy Monitor*, sua função é extrair do sistema valores de fator de potência, potência real, potência aparente, tensão eficaz e corrente eficaz (BRITO, 2016).

A biblioteca "ESP8266WiFi.h" é a biblioteca que contém os comandos relativos à conexão WiFi para programar o módulo (ATHOS ELETRONICS, [s.d]).

A biblioteca "BlynkSimpleEsp8266.h", cuja principal característica é permitir que a comunicação com o microcontrolador possa ser realizada, essa biblioteca também permite que outras plataformas sejam controladas remotamente, de forma que cargas sejam acionadas e dados de sensores e módulos possam ser obtidos e exibidos no aplicativo que fica instalado no dispositivo móvel (SILVINO, 2021).

E por fim, a biblioteca "WifiClientSecure" que foi utilizada para escrever os dados lidos do sensor de corrente em uma planilha do *Google* e a biblioteca "HTTPClient" que foi utilizada para enviar a cada 10 segundos para o *blynk* quanto o equipamento está gastando em reais enquanto está ligado.

3.3.2 *Firmware*

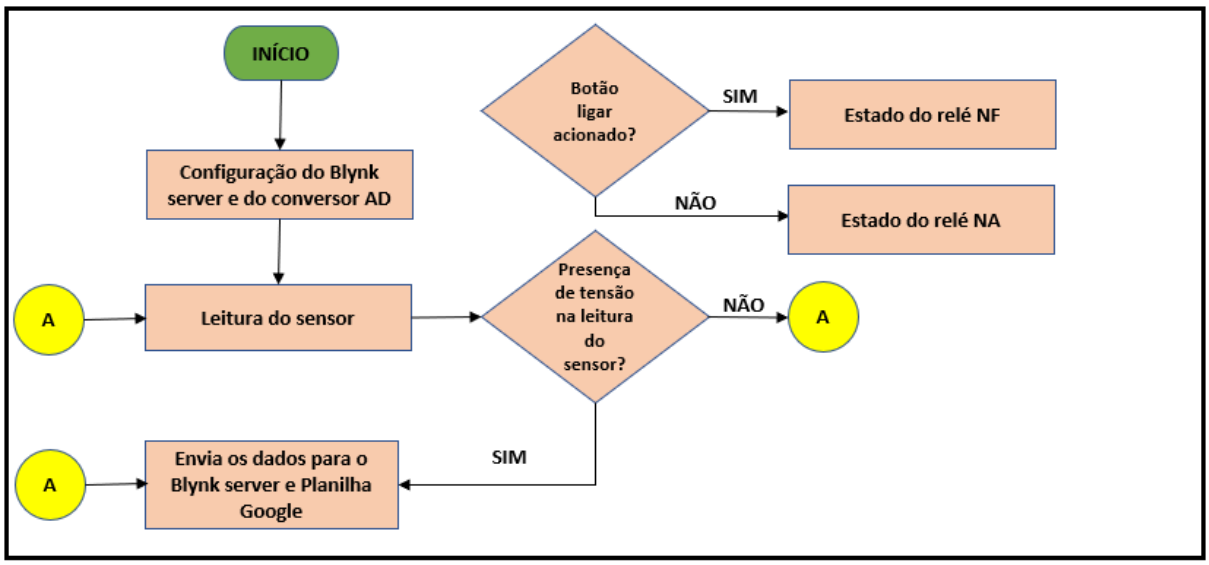
Após a verificação dos *sofwares* necessários e instalação das bibliotecas necessárias para o desenvolvimento do *firmware*, o código foi desenvolvido para funcionar conforme o fluxograma da Figura 21. Primeiramente é realizada a configuração do microcontrolador para conexão com o *Blynk* server, para isso o microcontrolador se conecta à uma rede WiFi e inicializa a comunicação com o *blynk* server por meio do *blynk token*, também é necessário usar um fator de correção para fazer a leitura do sensor de corrente, pois não há um resistor comercial

com o valor do resistor *shunt* calculado, na equação 16 tem-se a equação para calcular o fator de correção:

$$Fator_{correção} = \frac{CT\ Turns}{R_{shunt}} \qquad \text{(Equação 16)}$$

Após as configurações citadas acima, o microcontrolador realiza a leitura dos dados do sensor de corrente e calcula o valor eficaz e depois envia os dados para o servidor *blynk* e para uma planilha do *google* por meio de um *Google App Script* desenvolvido para esta finalidade, onde são feitos cálculos de quanto o dispositivo ligado está consumindo em tempo real, este valor em real é retornado para o microcontrolador que atualiza o aplicativo *blynk* com essa informação, também é possível ligar e desligar o adaptador mudando o estado do relé conectado ao microcontrolador por meio do botão "Ligar" via aplicativo, que será mostrado com mais detalhes na próxima seção.

Figura 21 - Fluxograma do *firmware* embarcado.



Fonte: Autoria própria.

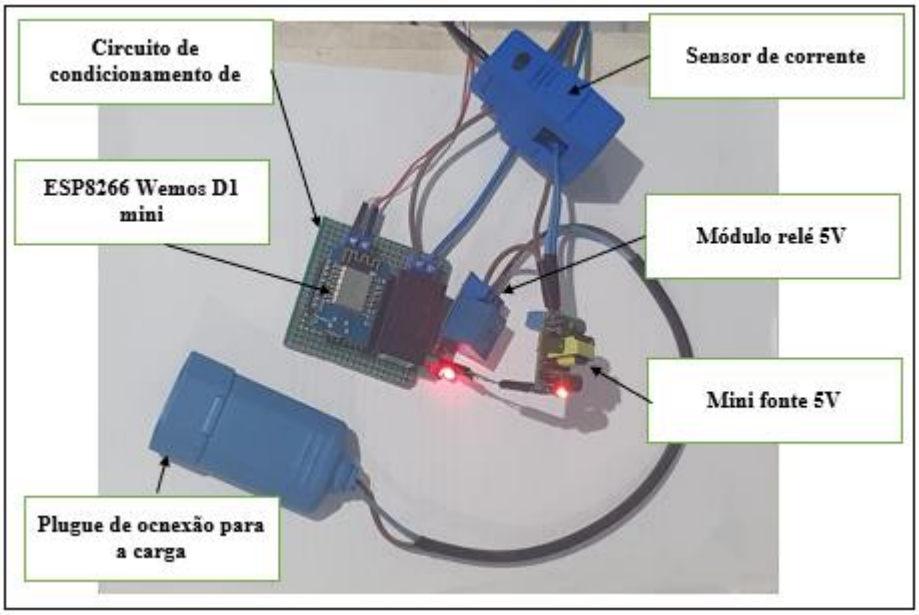
4 RESULTADOS OBTIDOS

Este capítulo mostra os resultados obtidos após a implementação do projeto. Importante frisar, que este é apenas um primeiro passo para a implementação de um sistema mais robusto, barato e capaz de funcionar em um modelo de escala comercial.

Depois de todo o estudo levantado, possibilidades discutidas chegou-se ao resultado planejado, com o sistema em pleno funcionamento monitorando o consumo em tempo real.

O protótipo é composto de 3 placas de circuito, quais sejam; a placa de condicionamento de sinais do sensor de corrente que foi feita utilizando uma placa de fenolite ilhada de 5,30 cm x 5,30 cm; a placa de mini fonte de alimentação de 5V e o módulo relé 5V. A montagem completa pode ser vista na Figura 22.

Figura 22 - Protótipo de tomada inteligente.



Fonte: Autoria própria.

O próximo passo foi o teste de leitura do sensor, para isso foi utilizada uma carga simulada no laboratório de eletrônica da Escola Superior de Tecnologia (EST). Nesse primeiro teste foi utilizado somente a conexão serial. Na Figura 23 observa-se os valores medidos pelo sensor convergindo para os valores medidos pelo alicate amperímetro.

Figura 23 - Teste comparando valor medido e valor real de corrente.

Fonte: Autoria própria.

Após a calibração do sensor foram feitos testes com cargas variadas para mensurar o erro de medição do protótipo, a Tabela 3 apresenta esses dados.

Tabela 3 - Comparação entre medições de corrente realizadas pelo multímetro e pelo protótipo.

Carga	Medição do multímetro (A)	Medição do protótipo (A)	Erro absoluto	Erro relativo
Liquidificador Mondial, Turbo Inox L-1000W	7,87	7,85	-0,02	0%
Sanduicheira	5,9	5,93	0,03	1%
Micro-ondas	5,22	5,24	0,02	0%
Batedeira Planetária Britânia	3,93	3,94	0,01	0%
TV QLED 55''	1,37	1,36	-0,01	-1%
Furadeira	1	1,02	0,02	2%
Geladeira Panasonic Inverter 387 L	0,86	0,87	0,01	1%
Notebook	0,31	0,31	0	0%
Ventilador	0,31	0,31	0	0%
Lâmpada fluorescente 20 W	0,15	0,15	0,01	0%

Fonte: Autoria própria.

O próximo teste incluiu todos os equipamentos juntos, isto é, o microcontrolador conectado à *internet* com o código de pareamento com o *Blynk* e Google planilhas. A carga testada foi um ventilador residencial de 127V de tensão de entrada e 110W de potência máxima.

Para mostrar os valores de corrente medido, o consumo em reais o acionamento da carga, montou-se a interface com 3 *widgets*, um botão para controlar a carga ligando e desligando, e outros 2 displays onde é mostrado a corrente e o valor em reais a ser pago devido ao tempo em que o equipamento permanecer ligado como visto na Figura 24.

Figura 24 - Aplicativo *Blynk* para o adaptador de tomada inteligente.

Fonte: Autoria própria.

4.1 RESULTADO DO SISTEMA DE MONITORAMENTO

O sistema construído a partir do protótipo se mostrou relativamente simples, tanto em relação a sua construção, quanto em face da sua montagem. A calibração do sensor e redução da faixa de corrente a ser medida de 100A para 10A tornou o experimento preciso e confiável, no entanto para que houvesse uma melhor facilidade na instalação por parte do usuário final o protótipo precisa de ajustes na montagem das placas para que todos os compontes possam ser acomodados em um a única placa e esta possa ser acomodada em uma caixa plástica ou de acrílico com um plugue de tomada macho na entrada e fêmea na saída.

A comunicação do *firmware* com o *Google App Script* atendeu às expectativas e fornece a funcionalidade de o usuário ajustar os valores de *Kilowatt* hora (*KWh)* cobrado pela concessionário e tensão ao qual o adaptador de tomada está conectado para que o cálculo de gasto em reais do equipamento possa ser realizado.

4.2 CUSTOS PARA A CONSTRUÇÃO DO PROJETO

Todos os custos podem ser observados na Tabela 4.

Tabela 4 - Custos totais investidos no projeto.

Quantidade	**Dispositivo**	**Preço total(R$)**
1	Capacitor de 10µF.	0.40
1	Capacitor de 1000µF.	0.40
2	Resistor de 1KΩ.	0,2
1	Resistor de 220Ω.	0.20
1	1 mini fonte 3.3V.	6,92
1	1 mini fonte 5V.	3,64
1	1 módulo relé 5V.	1,61
1	1 Microcontrolador ESP8266 Wemos D1 mini.	10,98
1	Sensor de corrente SCT-013.	18,49
3	Conector borne 2 polos	3,39
1	Placa de fenolite ilhada	2,37
	Total	**47,6**

Fonte: Autoria própria.

Fazendo uma comparação entre os custos do projeto com os adaptadores de tomada inteligente de 10A vendidos, comprova-se que é possível construir um adaptador de tomada inteligente por até metade do custo do que é oferecido pelo mercado, como pode ser observado na Tabela 5.

Tabela 5 - Pesquisa de preço dos adaptadores de tomada inteligente.

Adapatador de tomada inteligente	Marca	Preço (R$)	Site pesquisado em 15/01/2022
Tomada WiFi universal inteligente	Tomshin	96,59	Amazon
Tomada inteligente WiFi	AGL	148,98	Amazon
Adaptador de Tomada Inteligente	Geonav	93,39	Amazon
Tomada Inteligente S26	Sonoff	98,79	Mercado Livre
Tomada Inteligente Wifi	Nova Digital	78,00	Mercado Livre

Fonte: Autoria própria.

CONCLUSÃO

Neste trabalho foram apresentados conceitos de IoT e acerca do consumo de energia elétrica, com o intuito de mostrar a necessidade de discutir o tema devido a atual situação energética do país. Todos os dias podem ser observadas propostas sustentáveis e melhoria urbana. Essas propostas sempre tem uma imposição de quem as constrói, iniciativa pública ou privada, por isso elas tem pouca efetividade.

Quanto a iniciativas no setor energético, uma importante é a abordada neste trabalho, poucos dispositivos IoT estão à disposição do consumidor final de baixa renda, constatou-se que a leitura do consumo de energia elétrica em tempo real de forma remota gera maior conhecimento para o usuário e em seguida conscientização.

A parte difícil é convencer a população a utilizar o sistema pelo fato de ainda não ser oferecido por um preço barato no mercado, para efeito de comparação, apresentou-se na tabela 3 os gastos envolvidos no projeto e fazendo uma pesquisa no site da Amazon ou Mercado Livre, um adaptador de tomada inteligente com as mesmas especificações deste trabalho custa o dobro. Iniciativas como este projeto podem ser soluções para o consumidor ajudando esta tecnologia a se popularizar no mercado.

Para trabalhos futuros é sugerido utilizar um sensor para medir a tensão das cargas, para que se tenha maior precisão no monitoramento, outra sugestão é a integração de todos os componentes em uma única placa para que se possa reduzir o tamanho do adaptador. Pode-se ainda utilizar outro aplicativo para acionar e monitorar as cargas, pois os *widgets* da plataforma do *blynk* funcionam com um sistema de créditos, cada projeto recebe 1000 créditos para comprar os *widgets*, ao acabar os créditos é necessário comprar mais se quiser adicionar mais *widgets*.

O desenvolvimento de nossa sociedade se dará com a educação, parte da educação está extremamente relacionada com este projeto por causa da conscientização. O consumo de energia elétrica, é um dos temas que se agrava a cada dia pelo crescimento populacional, quanto mais pessoas mais energia deve ser gerada. No entanto os recursos de nosso planeta são finitos, mas nossas necessidades são infinitas, nossa única solução é saber utilizar a energia de forma consciente, evitando desperdícios.

REFERÊNCIAS

ATHOS ELETRONIC. **ESP8266 – O que é e para que serve?** [s.d]. Disponível em: https://athoselectronics.com/esp8266-o-que-e/>. Acesso em 02 jan. 2022.

AWS**. O QUE É A INTERNET DAS COISAS?.** 2019**.** Disponível em: < https://aws.amazon.com/pt/iot/what-is-the-internet-of-things/**>.** Acesso em3 abr. 2019.

BACKES, André. **Utilizando o Code Blocks**. [s.d]. Disponível em:< http://www.facom.ufu.br/~backes/gsi002/Aula02-UtilizandoCodeBlocks.pdf>. Acesso em 3 abr. 2019.

BRITO, João Luis Grizinsky, **Sistema para monitoramento de consumo de energia elétrica particular , em tempo real e não invasivo utilizando a tecnologia Arduino**. 2016. Universidade Estadual de Londrina. Disponível em: <http://www.uel.br/ctu/deel/TCC/TCC2016_JoaoLuisGrizinskyBrito.pdf>. Acesso em 3 abr. 2019.

CERQUEIRA, Giovanni. **Como programar o Arduino com o Visual Studio Code e PlatformIO IDE.** 2018. Disponível em: < https://www.embarcados.com.br/arduino-vscode-platformio/>. Acesso em 02 jan. 2022.

DELGADO, Armando Luiz. **Programação Básica em C++.** 2018. Universidade Federal do Paraná. Disponível em: <https://www.inf.ufpr.br/ci208/NotasAula.pdf>. Acesso em 3 abr. 2019.

DOSHI, Hiral S.; SHAH, Minesh S.; SHAIKH, Umair S A. **INTERNET of THINGS (IoT): INTEGRATION of BLYNK for DOMESTIC USABILITY.** 2017**.** Department Of Computer Engineering VIIT PUNE. Disponível em: < http://www.vjer.in/vol1issue4/vjer010429.PDF>. Acesso em: 14 abr. 2019.

ESPRESSIF SYSTEMS. **ESP8266EX Datasheet**. 2018. Disponível em: < https://espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf >. Acesso em 3 abr 2019.

FIGUEIREDO, Alexandre Carvalho; MARTINS, Lucas Milléo. **Desenvolvimento de um sensor de corrente elétrica a partir de um sensor de efeito hall**. 2017. Universidade Tecnológica Federal do Paraná. Disponível em: <https://repositorio.utfpr.edu.br/jspui/bitstream/1/16890/1/PG_COAUT_2017_2_06.pdf>. Acesso em 3 abr. 2019.

JUNIOR, Olair Ricardo. **Sistema de Monitoramento Residencial baseado em Internet das Coisas**. 2017. Monografia - Universidade Estadual de Londrina, Londrina, 2017.

MARTINELLI, Ruan R.; SIMONASSI, Kaio C. F.; TAVARES, Felipe P. C. **A linguagem Lua**. [s.d]. Disponível em: <https://www.inf.ufes.br/~vitorsouza/wp-content/uploads/teaching-lp-20132-seminario-lua.pdf>. Acesso em: 14 abr. 2019.

MERCADO LIVRE. **Adaptador Usb Serial Ttl Conversor Pl2303 / RS232 P/ Arduino**. [s.d]. Disponível em:< https://produto.mercadolivre.com.br/MLB-833015982-adaptador-usb-serial-ttl-conversor-pl2303-rs232-p-arduino-_JM?quantity=1>. Acesso em: 14 abr. 2019.

META DEV STUDIO. **O que é e para que serve o Google Apps Scripts**. 2019. Disponível em:< https://www.metadevstudio.com.br/artigo/o-que-e-e-para-que-serve-o-google-apps-scripts>. Acesso em 9 jan. 2022.

OLIVEIRA, Greici. **Nodemcu – Uma Plataforma Com Características Singulares Para O Seu Projeto Iot**. [s.d]. Disponível em:
< http://blogmasterwalkershop.com.br/embarcados/nodemcu/nodemcu-uma-plataforma-com-caracteristicas-singulares-para-o-seu-projeto-iot/>. Acesso em: 14 abr. 2019.

OOSTENVELD, Robert. **ESP-12 bootloader modes and GPIO state at startup**. [s.d]. Disponível em:<https://robertoostenveld.nl/esp-12-bootloader-modes/>. Acesso em 14 abr. 2019.

PIETER, P. **A Beginner's Guide to the ESP8266**. 2017. Disponível em:< https://tttapa.github.io/ESP8266/Chap01%20-%20ESP8266.html >. Acesso em 3 abr. 2019.

PLATFORMIO. **What is PlatformIO?**. [s.d]. Disponível em: < https://docs.platformio.org/en/latest/what-is-platformio.html?utm_source=arduino-esp8266>. Acesso em 02 jan. 2022.

PORTAL GSTI. **O que é C++?** [s.d]. Disponível em:< https://www.portalgsti.com.br/cplusplus/sobre/>. Acesso em 3 abr. 2019.

PRASAD, Sandeep; AIBARA, Firuza; KARMALI, Nagesh. **Code :: Blocks Manual**. 2014. Department of Computer Science and Engineering Indian Institute of Technology. Disponível em: < https://www.codeblocks.org/docs/manual_codeblocks_en.pdf>. Acesso em 14 abr. 2019.

SILVA, Pedro Lucas da. **Medindo Potência Elétrica Com Sensor Não-Invasivo Controlado Por Arduino**. 2018. Universidade Federal Rural Do Semi-Árido. Disponível em: <https://repositorio.ufersa.edu.br/bitstream/prefix/2410/2/PedroLS_MONO.pdf>. Acesso em 3 abr. 2019.

SILVINO, Arthur. **Como usar o aplicativo blynk com nodemcu**. 2021. Disponível em:< https://autocorerobotica.blog.br/como-usar-o-aplicativo-blynk-com-nodemcu>. Acesso em 02 jan. 2022.

SMART PROJECTS. **Módulo ESP8266 D1 Mini - Wemos D1 Mini Wifi**. [s.d]. Disponível em:<https://www.smartprojectsbrasil.com.br/modulo-esp8266-d1-mini-wemos-d1-mini-wifi>. Acesso em 3 abr. 2019.

TOGINHO FILHO, D. O., ANDRELLO, A.C. **Conceitos de medidas e teoria de erros**. Londrina 2010. Disponível em: <http://www.uel.br/pessoal/inocente/pages/arquivos/03-Conceitos%20de%20medidas%20e%20teoria%20de%20erros.pdf>. Acesso em: 27 out. 2021.

VIDA DE SILÍCIO. **SCT-013 – Sensor de Corrente Alternada com Arduino**. 2017. Disponível em: <https://portal.vidadesilicio.com.br/sct-013-sensor-de-corrente-alternada/>. Acesso em 14 abr. 2019.

WEMOS. **LOLIN D1 mini**. [s.d]. Disponível em:< https://www.wemos.cc/en/latest/d1/d1_mini.html>. Acesso em 3 abr. 2019.

YHDC. **Split Core Current Transfomer**. [s.d]. Disponível em:
< https://www.toroid.com.br/novo/wp-content/uploads/2018/04/Transf-corrente.pdf>. Acesso em: 14 abr. 2019.

APÊNDICE A - CÓDIGO DESENVOLVIDO PARA O MICROCONTROLADOR ESP8266 WEMOS D1 MINI

```
/***
********** Calculo do resistor de shunt ********************
ADC_VOLTAGE -> Tensão maxima do conversor AD = 3.3V
I_RMS -> corrente RMS Maxima que se quer medir = 10A
CT_TURNS -> Quantidade de voltas do transformador de corrente = 2000

SHUNT = (ADC_VOLTAGE / 2) / ((I_RMS * 1.414) / CT_TURNS)
SHUNT = 233 Ohms

********** Calculo do fator de correção **********************
Fator de correção do SHUNT, usado pra corrigir o valor real do
resistor
Resitor usado = 220 Ohm
FATOR = CT_TURNS / SHUNT
FATOR = 2000 /

***/
#include <Arduino.h>
#include "EmonLib.h"
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <WifiClientSecure.h> //para atualizar google sheet
#include <HTTPClient.h>

#define SHUNT 220
#define I_RMS 10
#define CT_TURNS 2000
#define PIN_UPTIME V4
#define AMOUNT_SPENT V5
#define BLYNK_PRINT Serial
```

```
double fator = (double) CT_TURNS / (double) SHUNT;
char token[] = "yBwXiiZkclVdVVsguQnsKO81QIqdBpyB";
char ssid[] = "VIVOFIBRA-0359";
char password[] = "e6FbX3nwpV";
bool state = false;
EnergyMonitor sensor = EnergyMonitor();

BlynkTimer timer;

//----------------------------------------Host & httpsPort
const char* host = "script.google.com";
const int httpsPort = 443;
//----------------------------------------
WiFiClientSecure client; //--> Create a WiFiClientSecure object.

String GAS_ID = "AKfycbzri0r2bT60ECfOWS1KFagnZKjOt0HxhLgTIPMf"; //--> spreadsheet script ID

void sendingData()
{
  digitalWrite(LED_BUILTIN, state);
  state = !state;

  double current = sensor.calcIrms(1480);
  Blynk.virtualWrite(PIN_UPTIME, current);
}

// Subroutine for sending data to Google Sheets
void sendData(float current) {
  Serial.println("==========");
  Serial.print("connecting to ");
  Serial.println(host);
```

```
//----------------------------------------Connect to Google host
if (!client.connect(host, httpsPort)) {
  Serial.println("connection failed");
  return;
}
//----------------------------------------

//----------------------------------------Processing data and sending data
String string_current =  String(tem);
// String string_temperature =  String(tem, DEC);
String string_humidity =  String(hum, DEC);
String url = "/macros/s/" + GAS_ID + "/exec?current=" + string_current;
Serial.print("requesting URL: ");
Serial.println(url);

client.print(String("GET ") + url + " HTTP/1.1\r\n" +
       "Host: " + host + "\r\n" +
       "User-Agent: BuildFailureDetectorESP8266\r\n" +
       "Connection: close\r\n\r\n");

Serial.println("request sent");
//----------------------------------------

//----------------------------------------Checking whether the data was sent successfully or
not
while (client.connected()) {
  String line = client.readStringUntil('\n');
  if (line == "\r") {
    Serial.println("headers received");
    break;
  }
}
String line = client.readStringUntil('\n');
if (line.startsWith("{\"state\":\"success\"")) {
```

```
    Serial.println("esp8266 CI successfull!");
   } else {
    Serial.println("esp8266 CI has failed");
   }
   Serial.print("reply was : ");
   Serial.println(line);
   Serial.println("closing connection");
   Serial.println("==========");
   Serial.println();
   //----------------------------------------
  }

  //function to read data from Google sheet
  void spreadsheet_comm(void) {
    HTTPClient http;
    String url="https://script.google.com/macros/s/"+String GAS_ID+"/exec?read";
  //   Serial.print(url);
        Serial.print("Making a request");
        http.begin(url.c_str()); //Specify the URL and certificate
   http.setFollowRedirects(HTTPC_STRICT_FOLLOW_REDIRECTS);
        int httpCode = http.GET();
   String payload;
    if (httpCode > 0) { //Check for the returning code
       payload = http.getString();

       Serial.println(httpCode);
       Serial.println(payload);
       Blynk.virtualWrite(AMOUNT_SPENT, current); // write the amount spent in
Blynk display
     }
    else {
     Serial.println("Error on HTTP request");
    }
        http.end();
```

```
}

void setup(){
   Serial.begin(9600);
   sensor.current(A0, fator);
   pinMode(LED_BUILTIN, OUTPUT);
   Blynk.begin(token, ssid, password);
   timer.setInterval(1000L, sendingData);
   client.setInsecure();//para ler google sheet

}

void loop(){
   Blynk.run();
   timer.run();
   sendData(PIN_UPTIME); //--> Calls the sendData Subroutine
   spreadsheet_comm();
}
```

APÊNDICE B - CÓDIGO DESENVOLVIDO PARA CRIAR O GOOGLE APP SCRIPT

```
function doGet(e) {
  Logger.log( JSON.stringify(e) );
  var result = 'Ok';
  if (e.parameter == 'undefined') {
    result = 'No Parameters';
  }
  else {
    var sheet_id = '1UdRZcuNgBrTJ4H2wEMqnI8RaSks6_wV2pBoil2JDyWU';  // Spreadsheet ID
    var sheet = SpreadsheetApp.openById(sheet_id).getActiveSheet();
    var newRow = sheet.getLastRow() + 1;
    var rowData = [];
    var Curr_Date = new Date();
    rowData[0] = Curr_Date; // Date in column A
    var Curr_Time = Utilities.formatDate(Curr_Date, "America/Manaus", 'HH:mm:ss');
    rowData[1] = Curr_Time; // Time in column B
    for (var param in e.parameter) {
      Logger.log('In for loop, param=' + param);
      var value = stripQuotes(e.parameter[param]);
      Logger.log(param + ':' + e.parameter[param]);
      switch (param) {
        case 'current':
          rowData[2] = value; // Current in column C
          result = 'Current Written on column C';
          break;
        default:
          result = "unsupported parameter";
      }
    }
    Logger.log(JSON.stringify(rowData));
```

```
    var newRange = sheet.getRange(newRow, 1, 1, rowData.length);
    newRange.setValues([rowData]);
  }
  return ContentService.createTextOutput(result);
}

function stripQuotes( value ) {
  return value.replace(/^["']|["']$/g, "");
}

function doGet(e){
  var read = e.parameter.read;

  if (read !== undefined){
    return ContentService.createTextOutput(sheet.getRange('I6').getValue());
  }
}
```