

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO**

**SISTEMA DE
CONSCIENTIZAÇÃO PARA
ATAQUES DE *PHISHING* DE
BAIXO CUSTO E
ENERGETICAMENTE
AUTOSUSTENTÁVEL**

GABRIEL RADZKI FRANÇA

Trabalho de Conclusão apresentado
como requisito à obtenção do grau de
Bacharel em Engenharia de Computação
na Pontifícia Universidade Católica do
Rio Grande do Sul.

Orientador: Prof. Julio César Marques de Lima

**Porto Alegre
2025**

DEDICATÓRIA

Dedico este trabalho à minha família, especialmente ao meu avô Sergio, que me incentivou, mesmo que de maneira inconsciente, a seguir o caminho da computação. Aos meus amigos, namorada e a todos que me motivaram durante esta jornada.

*“Do not pity the dead, Harry. Pity the living.
And above all, those who live without love.”*
(Dumbledore, em *Harry Potter and the Deathly Hallows*, por J.K. Rowling)

SISTEMA DE CONSCIENTIZAÇÃO PARA ATAQUES DE *PHISHING* DE BAIXO CUSTO E ENERGETICAMENTE AUTOSSUSTENTÁVEL

RESUMO

A democratização da Internet possibilitou que grande parte da população esteja conectada à rede. Atualmente, o número de ataques cibernéticos vem crescendo a cada ano. Entretanto, a educação da população em geral a respeito do assunto pouco evolui. Neste contexto, a rede torna-se alvo para a ação de cibercriminosos. Este trabalho busca construir, integrando *hardware* e *software*, um sistema de conscientização para ataques de *phishing*. Para atingir este objetivo, o projeto utiliza tecnologias como *NodeMCU*, redes *Wi-Fi*, explora falhas em protocolos conhecidos como o *deauthentication attack* e utiliza técnicas de engenharia reversa para solucionar problemas. A evidenciação da viabilidade é realizada por meio de um protótipo energeticamente autossustentável, portátil e de baixo-custo. A aplicação do protótipo em campo possibilitou a demonstração e o sucesso da arquitetura proposta.

Palavras-Chave: *Phishing*, *IoT*, Engenharia Reversa, *Hardware*, *Software*.

LOW COST AND ENERGY SUFFICIENT *PHISHING* AWARENESS SYSTEM

ABSTRACT

The democratization of the Internet made possible that a big part of the population stays connected to the network. Currently, the cibercriminal attacks numbers are rising each year. In the opposite direction, population's knowledge about the subject has little to no evolution. In this context, the internet becomes a target for cibercriminal actions. This present work aims to build, integrating hardware and software, an awareness system for phishing attacks. In order to fulfill this objective, the project uses technologies such as NodeMCU, Wi-Fi networks, exploits flaws in known protocols, such as the deauthentication attack, and utilizes reverse engineering techniques for problem solving. The evidentication of the feasibility is achieved by constructing a prototype that is low-cost, portable and energy sufficient. After using the prototype on the field, it demonstrated the success of the proposed architecture.

Keywords: Phishing, IoT, Reverse Engineering, Hardware, Software.

LISTA DE FIGURAS

Figura 2.1 – Diagrama de blocos do projeto	14
Figura 2.2 – ESP-12E WiFi Module Datasheet [AI-11]	15
Figura 3.1 – Prova de Conceito - Portal Cativo	19
Figura 3.2 – Divisor de tensão nativo na entrada A0.....	20
Figura 3.3 – Abordagem simples para obter o consumo médio de corrente	21
Figura 3.4 – Curva de carga	21
Figura 3.5 – Curva I-V característica	22
Figura 4.1 – Página inicial do portal desenvolvido	23
Figura 4.2 – Exemplos de Portais Cativos.....	24
Figura 4.3 – Comparativo: página clonada à esquerda, página original à direita ..	25
Figura 4.4 – Página final	26
Figura 4.5 – PDF contendo mais dicas	26
Figura 4.6 – Página de configurações	27
Figura 4.7 – Layout da memória Flash [Pla]	27
Figura 4.8 – Assembly da função	29
Figura 4.9 – Assembly da função	30
Figura 5.1 – Divisor de tensão [Cir]	32
Figura 5.2 – Divisor de tensão projetado.....	33
Figura 5.3 – Engenharia reversa no circuito de carga	35
Figura 5.4 – CH340G: Pinout	35
Figura 5.5 – Jumper adicionado ao CI USB/UART (canto superior esquerdo)	36
Figura 5.6 – Curva com carga linear	36
Figura 5.7 – Irradiação solar no plano horizontal - Porto Alegre [CRE18].....	37
Figura 5.8 – Função de consumo energético diário (mA)	37
Figura 5.9 – Função de superávit/déficit energético diário (mA)	37
Figura A.1 – Esquemático completo do circuito	43
Figura B.1 – Shield desenvolvido neste trabalho	44
Figura B.2 – Shield desenvolvido neste trabalho	45
Figura C.1 – Circuito de carga onde foi aplicada engenharia reversa	46
Figura D.1 – Ligação completa (sem os painéis)	47
Figura E.1 – Caixa antes de selar	48
Figura E.2 – Caixa selada com os painéis	49

LISTA DE TABELAS

Tabela 3.1 – Aferições de tensão e corrente em período de zênite solar. Ângulo de incidência aprox. 90º	22
Tabela 5.1 – Parâmetros dos divisores de tensão	32
Tabela 5.2 – Dimensionamento. Os sinais de +/- representam o sentido da corrente.	36

LISTA DE SIGLAS

A – Amperes

AD – Analógico/Digital

API – Interface de Programação de Aplicações

CI – Circuito Integrado

CSS – *Cascading Style Sheets*

CTA – *Call-to-Action*

DNS – *Domain Name Server*

ELF – *Executable and Linkable Format*

GPIO – General Purpose Input/Output

IMP – Corrente de máxima potência

IOT – Internet das Coisas

HTML – *HyperText Markup Language*

IDE – *Integrated Development Environment*

IP – *Internet Protocol*

MAC – *Media Access Control*

MB – *MegaByte*

MOSFET – *Metal Oxide Semiconductor Field Effect Transistor*

MUX – Multiplexador

NSA – *National Security Agency*

OTA – *Over-The-Air*

POC – Prova de Conceito

RTC – *Real-Time Clock*

SOC – System On Chip

SWA – Servidor Web Assíncrono

TX – Transmissão de dados, neste caso, por meio de ondas eletromagnéticas (Wi-Fi)

VMP – Tensão de máxima potência

SUMÁRIO

1	INTRODUÇÃO	11
1.1	MOTIVAÇÃO	12
1.2	OBJETIVOS	12
2	ARQUITETURA DO PROJETO	14
2.1	NODEMCU	15
2.1.1	ESP-12E	15
2.1.2	<i>LITTLEFS</i>	16
2.2	PLATFORMIO	16
2.3	IEEE 802.11 B/G/N	16
2.3.1	<i>DEAUTHENTICATION ATTACK</i>	16
2.4	PAINEL FOTOVOLTAICO	17
2.5	PORTAL CATIVO	17
3	VALIDAÇÃO INICIAL	18
3.1	POC DE <i>SOFTWARE</i>	18
3.1.1	PREPARAÇÃO DO AMBIENTE DE DESENVOLVIMENTO	18
3.1.2	SERVIDOR WEB ASSÍNCRONO (SWA)	18
3.1.3	SERVIDOR DNS	19
3.1.4	INTEGRAÇÃO DO <i>LITTLEFS</i> COM O SERVIDOR WEB	19
3.2	POC DE <i>HARDWARE</i>	20
3.2.1	CONSUMO ENERGÉTICO DO MICROCONTROLADOR	20
3.2.2	ESPECIFICAÇÕES DO PAINEL FOTOVOLTAICO NA PRÁTICA	21
4	PROJETO DE <i>SOFTWARE</i>	23
4.1	DESENVOLVIMENTO <i>WEB</i>	23
4.2	ABERTURA AUTOMÁTICA DO PORTAL CATIVO EM DIFERENTES DISPOSITIVOS	25
4.3	<i>UPLOAD REMOTO - OTA</i>	27
4.4	FUNÇÕES DE EXECUÇÃO PERIÓDICA - <i>TICKERS</i>	28
4.5	CONTROLE DE CONEXÃO POR ENDEREÇO MAC	28
4.5.1	ENGENHARIA REVERSA DE BINÁRIO COM O <i>GHIDRA</i>	29
4.6	CONTROLE DE ENERGIA - <i>DEEP SLEEP</i>	30

5	PROJETO DE <i>HARDWARE</i>	32
5.1	DIVISORES DE TENSÃO	32
5.1.1	MULTIPLEXAÇÃO E CONSUMO DOS DIVISORES	33
5.1.2	COMPENSAÇÃO POR TEMPERATURA	33
5.2	CIRCUITO DE CARGA DA BATERIA	34
5.3	REDUZINDO O CONSUMO ENERGÉTICO DO NODEMCU	34
5.4	DIMENSIONAMENTO DOS PAINÉIS	35
6	VALIDAÇÃO	38
7	CONCLUSÃO	40
	REFERÊNCIAS	41
	APÊNDICE A – Esquemático completo	43
	APÊNDICE B – <i>Shield</i>	44
	APÊNDICE C – Circuito de Carga	46
	APÊNDICE D – Ligação completa	47
	APÊNDICE E – Protótipo final	48

1. INTRODUÇÃO

A democratização dos dispositivos móveis e, principalmente, da internet possibilitou que grande parte da população mundial esteja conectada à rede. No Brasil, estima-se que, em 2017, 96% dos internautas tenham acessado o ciberespaço por meio do telefone celular[PLO20]. Em contrapartida à facilidade de acesso, as condições socioeconômicas da população impedem que uma parte dos usuários de internet tenham acesso às redes móveis de banda larga, ficando à mercê dos pontos de acesso estáticos e às redes Wi-Fi públicas.

Neste contexto, a rede torna-se propícia para que cibercriminosos tomem vantagem da falta de informação sobre segurança digital para a população em geral, e executem ataques de engenharia social, *phishing* ou até mesmo *Man-In-The-Middle (MITM)*. Em 2017, o Brasil ocupava a terceira posição no ranking dos países que mais originam ataques cibernéticos [Khi20]. O roubo de credenciais, pode ser o ponto de entrada para uma investida *blackhat* mais sofisticada como o ataque de *ransomware*, onde o adversário criptografa os arquivos da organização e, em seguida, solicita um resgate para descriptografá-los.

Esse último tipo de ataque tem crescido nos últimos anos, tanto em números quanto geograficamente [OM12]. Muitas empresas de pequeno porte até multinacionais podem ser extremamente prejudicadas financeiramente por esse tipo de ameaça. Estudos realizados a partir de campanhas de *phishing* empresariais expõem os perigos da falta de conhecimento e negligenciamento do assunto por parte dos gerentes de tecnologia [MJVB19].

No cenário de segurança ofensiva, existem inúmeros projetos *open-source* voltados à exploração de redes sem-fio. Alguns utilizam microcontroladores da família ESP8266, devido a seu chip *Wi-Fi* e por ser capaz de suportar a pilha TCP/IP. São chips de baixo custo, extremamente populares na comunidade *maker*.

O projeto irá explorar o kit de desenvolvimento NodeMCU, que utiliza o chip ESP-12E, junto à sua capacidade de comunicação em redes sem fio, para construir um *framework* voltado à campanhas de *phishing*, de forma a possibilitar e popularizar a conscientização do assunto. Visando o uso *standalone*, a solução deve ser energeticamente autossustentável, i.e., não deverá estar ligada à rede de energia elétrica. Para isso, um pequeno painel solar carregará uma bateria, de maneira a sustentar o microcontrolador em períodos sem luz solar. Em casos extremos, onde a bateria pode atingir níveis de tensão abaixo do recomendado, o microcontrolador deverá entrar em estado de *deep sleep*.

O requisito de baixo custo da solução permite a inserção do conceito de reutilização (também conhecido como *upcycling*), que é (ou ao menos deveria ser) muito importante nos dias de hoje. Serviços de cunho social como o "Projeto Recondicionar", do Polo Marista de Formação Tecnológica, dão fins dignos aos resíduos de equipamentos eletrônicos, apli-

cando a concepção de reutilização de maneira admirável, aproveitando o que, por muitos, é considerado lixo, para transformar em robôs, meta arte ou até mesmo novos computadores [Mar14]. Partindo deste princípio, é vital que boa parte do projeto tenha componentes advindos de lixo eletrônico.

Por fim, um portal cativo será o ponto de entrada, onde o usuário, desavisado, informará suas credenciais da rede social que escolher, aproveitando-se do mecanismo de login *Single Sign-On (SSO)*. Ao final da requisição, o internauta será apresentado com uma tela evidenciando o que aconteceu e exibindo dicas e boas práticas para que o cidadão, futuramente, tenha o discernimento para não cair em ataques fraudulentos.

1.1 Motivação

A principal motivação para o desenvolvimento deste projeto é conscientizar as pessoas a respeito de segurança digital, ensinando-as de maneira prática sobre como é possível prevenir-se do *phishing* - um dos ataques mais sofisticados de engenharia social. Sendo portátil e autônomo, o projeto deve ter baixo consumo de energia.

A solução final também pode ser usada no contexto empresarial, permitindo que qualquer companhia eduque seus funcionários a respeito de ofensivas cibercriminosas, visto que esse tipo de prática reduz consideravelmente os incidentes cibernéticos indesejados [MJVB19].

1.2 Objetivos

O objetivo deste trabalho é construir um sistema *open-source* de conscientização para ataques de *phishing* que possa ser usado ao ar livre, seja à prova de intempéries, energeticamente autossustentável e com *front-end* customizável. O trabalho não tem como objetivo construir ou desenvolver um compartimento a prova d'água para manter o circuito, e, portanto, este será adquirido comercialmente e posteriormente usado no projeto.

O *front-end* customizável também não implica no desenvolvimento de inúmeras páginas de autenticação. A customização têm como premissa a inserção de qualquer página nova no sistema, por qualquer colaborador, assim que o projeto for liberado como *open-source*.

O fluxo fim-a-fim perfeito do projeto deverá ser o seguinte:

- Usuário conecta-se à rede *Wi-Fi* maliciosa.
- Usuário é automaticamente direcionado ao portal cativo.

- Usuário seleciona alguma rede social para autenticação.
- Usuário é apresentado com a tela de *login* desta rede social e insere suas informações.
- Ao pressionar o botão de *login*, o usuário é redirecionado para uma página detalhando o que acaba de ocorrer e como ele pode se proteger de ataques maliciosos no futuro.
- O microcontrolador desconecta o usuário.

O sistema deverá manter a contagem de quantos usuários completaram o ciclo de autenticação, mas não deve salvar suas credenciais.

2. ARQUITETURA DO PROJETO

O projeto usará o kit de desenvolvimento NodeMCU, que utiliza chip ESP-12E da família ESP8266, painel fotovoltaico, divisor de tensão e multiplexador. O último se dá necessário devido à existência de apenas um conversor AD (Analógico/Digital) no SoC (*System-On-Chip*). Os divisores de tensão são transistorizados, permitindo assim, que sejam desativados enquanto não estão sendo usados.

Os divisores têm como objetivo medir os níveis de tensão da bateria e do painel solar, visando decidir pela suspensão da transmissão do sinal de rede sem fio, permitindo um gerenciamento eficiente de energia. Eles podem ser ativados ou desativados, a partir de transistores, com o intuito, também, de economizar energia.

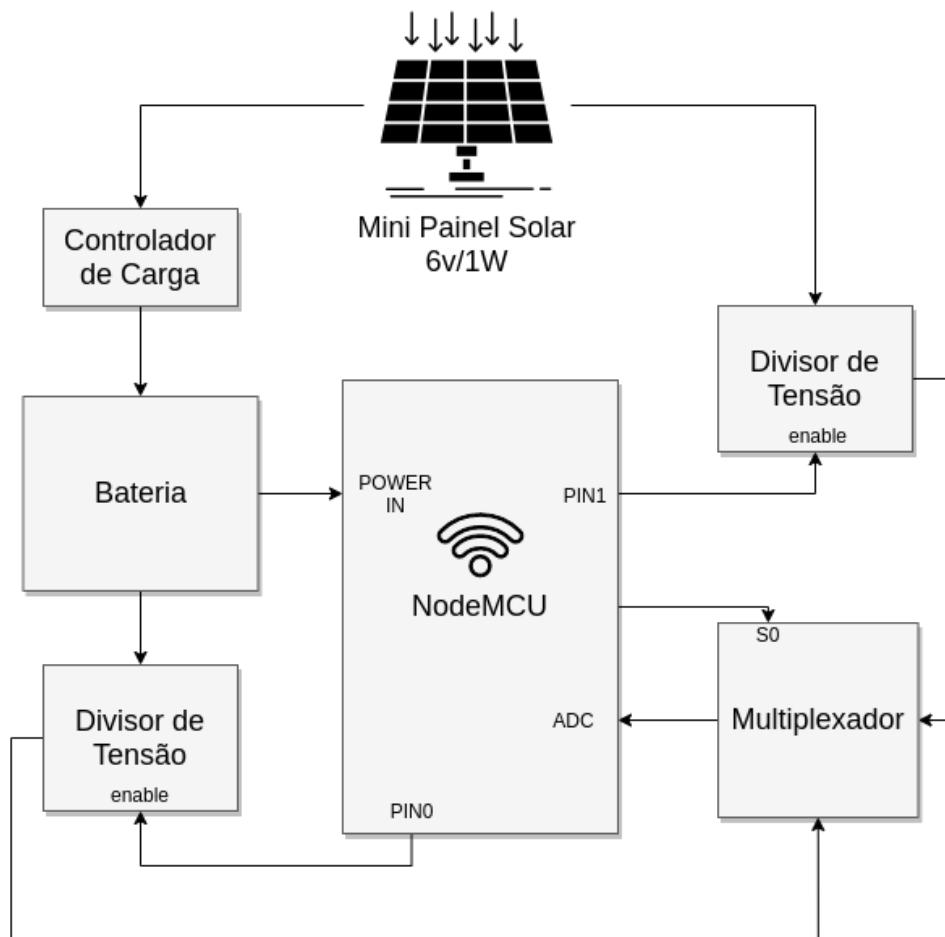


Figura 2.1 – Diagrama de blocos do projeto

2.1 NodeMCU

NodeMCU é um kit de desenvolvimento para prototipação de projetos IoT (*Internet of Things*), baseado na linguagem LUA. É amplamente utilizada na comunidade *maker* e também na comunidade de segurança ofensiva, em projetos focados, principalmente, em redes sem fio.

2.1.1 ESP-12E

O chip ESP-12E possui pinos de *GPIO* (*General Purpose Input/Output*), *PWM* (*Pulse Width Modulation*), *I²C* (*Inter-Integrated Circuit*) e conversor AD. Suporta completamente a pilha TCP/IP. O SoC tem uma limitação conhecida: devido à sua baixa memória RAM (*Random Access Memory*) quando está operando em modo AP (*Access Point*), a rede é limitada a quatro usuários simultâneos, por padrão, sendo possível aumentar para oito [Gro17]. Para que não haja congestionamento, ao final do ciclo de autenticação o usuário deverá ser desconectado.

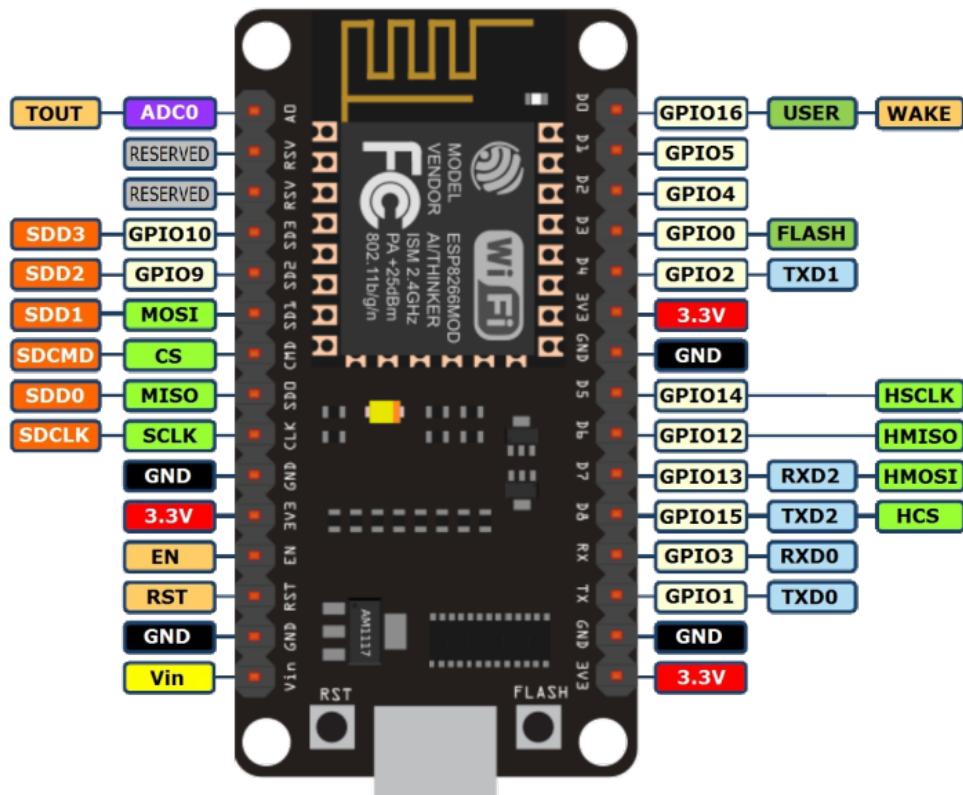


Figura 2.2 – ESP-12E WiFi Module Datasheet [AI-11]

2.1.2 LittleFS

O *LittleFS* é um sistema de arquivos para memórias *flash* do tipo NOR usadas em dispositivos embarcados. O sistema será usado para armazenar as páginas *web* clonadas a partir das páginas de autenticação originais das redes sociais contempladas neste trabalho, além de armazenar o contador de vítimas da rede maliciosa.

2.2 PlatformIO

PlatformIO é um ecossistema de código aberto para o desenvolvimento de aplicações IoT, possuindo gerenciador de bibliotecas e suporte completo ao desenvolvimento usando ESP8266. Podendo ser instalado como uma extensão do IDE (*Integrated Development Environment*) *VSCode*, permite a fácil comunicação entre o computador e o microcontrolador. Oferece muitas funções que facilitam a escrita de *firmware*, construção do sistema de arquivos supramencionado e comunicação serial.

2.3 IEEE 802.11 b/g/n

O padrão IEEE 802.11 faz parte do conjunto de normas técnicas IEEE 802 para redes locais, implementando controle de acesso ao meio e protocolos na camada física que permitem comunicação por meio de redes sem fio. A família 802.11 oferece uma série de modulações *half-duplex* OTA (*over-the-air*). Os padrões b/g/n utilizam a faixa de 2,4GHz. O protocolo possui algumas funcionalidades que são exploradas por atores mal intencionados para efetuar alguns tipos de ataque.

2.3.1 Deauthentication attack

O *deauthentication attack* consiste na desconexão de um cliente específico da rede, informado no *deauthentication frame*. Está especificado no protocolo 802.11, entretanto seu mau uso torna-se uma ferramenta para cibercriminosos. Ao clonar uma rede sem fio, transmitindo a rede maliciosa com o mesmo *SSID*, um atacante consegue desconectar um cliente da rede legítima e, dependendo da intensidade do sinal, faz com que a vítima reconecte-se no *access point (AP)* malicioso [SYG10]. Esse tipo de ataque é conhecido como *Evil Twin Access Points*. A técnica utilizada neste trabalho é inspirada neste ataque, porém, não há desconexão de usuários conectados a outras redes.

2.4 Painel Fotovoltaico

São usados painéis solares com tensão de operação de 6 volts e corrente de curto-círcito próximas a 170mA, aproximando-se de 1 watt de potência. Em ambientes externos os painéis são uma boa solução para o problema de energia em dispositivos IoT [PAG09]. Em tese, um painel é suficiente para alimentar o NodeMCU, visto que seu consumo médio é de até 140mA, de acordo com o *datasheet*.

2.5 Portal Cativo

Um portal cativo é uma página da internet acessada por um navegador, exibida para usuários recém conectados a uma rede - seja ela *Wi-Fi* ou com fios - limitando o acesso a outros recursos. Normalmente são utilizados visando a autenticação a partir de pagamento, realização de pesquisas de marketing ou exibição de propagandas. No Brasil, uma das redes mais populares deste tipo é a "#NET-CLARO-WIFI", oferecida pela operadora de telefonia Claro. Os dispositivos móveis e computadores abrem automaticamente o portal cativo quando são conectados a este tipo de rede. Por um lado é algo extremamente útil, por outro é extremamente perigoso, pois um atacante pode manipular um portal cativo falso [WLEM19].

Existem algumas maneiras distintas de disponibilizar portais cativos a partir de pontos de acesso de redes sem fio: redirecionamento HTTP (*HyperText Transfer Protocol*), redirecionamento ICMP (*Internet Control Message Protocol*) e redirecionamento DNS (*Domain Name Server*). Neste projeto usaremos a primeira e a última opção. Esta escolha dá-se ao fato de que grande parte dos dispositivos fazem checagens do tipo *heartbeat* para verificar a conexão com a rede. A maioria destas requisições são feitas por meio do protocolo HTTP ou por requisições de DNS.

3. VALIDAÇÃO INICIAL

Este capítulo descreve o desenvolvimento do projeto, a partir da arquitetura, englobando o desenvolvimento das soluções de *hardware* e *software*. A validação inicial do plano foi feita a partir de duas provas de conceito: *software* e *hardware*. Após a validação inicial, o desenvolvimento também foi fragmentado, como mencionado previamente. Algumas tarefas de *hardware* e *software* foram executadas paralelamente, pois são correlacionadas. Por fim, a validação da solução final também será relatada neste capítulo.

3.1 POC de *Software*

Visando garantir a viabilidade do projeto a partir da perspectiva de *software*, a prova de conceito foi dividida em quatro partes.

3.1.1 Preparação do ambiente de desenvolvimento

O IDE escolhido para o desdobramento do projeto foi o *Visual Studio Code*, da Microsoft. Para auxiliar no desenvolvimento junto ao *VSCode*, a ferramenta *PlatformIO* também foi instalada e configurada.

3.1.2 Servidor Web Assíncrono (SWA)

Uma das premissas importantes do projeto é que a rede possa ser acessada simultaneamente por mais de um indivíduo. Para isso, é necessário que o servidor *web*, utilizado para servir as páginas ao usuário, seja capaz de lidar com mais de uma requisição ao mesmo tempo. A biblioteca *ESPAsyncWebServer* atende esses requisitos [No 21]. O código faz uso do módulo *ESPAsyncTCP* [No 19], que é justamente o que garante requisições não bloqueantes, permitindo a entrega de páginas HTML de maneira rápida e eficiente, a partir do sistema de arquivos do microcontrolador.

No sentido de legitimar o servidor, foi desenvolvida uma pequena aplicação com o único intuito de servir uma página com o texto "Alô Mundo", enquanto a função de *loop* do microcontrolador executava tarefas bloqueantes (*delays* de alguns segundos). O servidor respondeu com sucesso às solicitações praticadas, demonstrando a capacidade de atender requisições e fazer processamentos de forma simultânea.

3.1.3 Servidor DNS

Outro aspecto a ser considerado é a implementação de um servidor DNS. De nada adianta servir as páginas HTML sem que haja resolução de nomes de domínio. Desta maneira, sua existência é de suma importância, tendo em vista que as mais distintas checagens de conectividade, efetuadas pelos mais diversos dispositivos conectados na rede, não seriam possíveis, pois não haveria tradução de nome para o endereço IP solicitado.

3.1.4 Integração do *LittleFS* com o Servidor Web

Ao final das validações supracitadas, verifica-se a promessa de integração entre o SWA (Servidor Web Assíncrono) o sistema de arquivos do microcontrolador. A maior apreensão está em saber se o SWA é eficiente na resposta de múltiplas requisições simultâneas, com arquivos distintos, como imagens, *javascript* e CSS. A POC desenvolvida foi uma página com inúmeras imagens, *scripts* e arquivos de estilo. Constatou-se que o NodeMCU, em conjunto com o SWA, é bastante eficiente em lidar com o consumo de arquivos paralelamente a partir de um cliente.

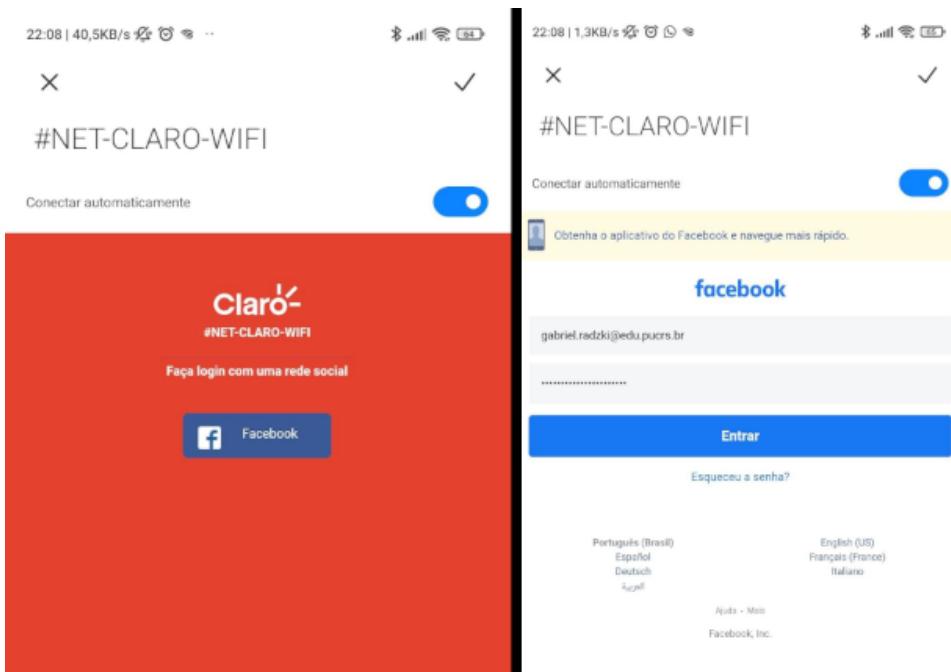


Figura 3.1 – Prova de Conceito - Portal Cativo

3.2 POC de *Hardware*

O aspecto mais imprescindível para a exequibilidade do trabalho é que o painel fotovoltaico seja o suficiente para os requisitos de energia do NodeMCU, além de suprir a carga de uma bateria, que servirá para manter o funcionamento do sistema em dias chuvosos, nublados ou até mesmo durante a noite. Por conseguinte, o dimensionamento do sistema de carga torna-se um dos maiores desafios do trabalho.

3.2.1 Consumo energético do microcontrolador

Objetivando o correto dimensionamento do sistema, é necessária a obtenção das medidas de consumo de corrente média do NodeMCU. Portanto, foi desenvolvido um *data logger* com outro microcontrolador, onde a cada minuto, em um intervalo de uma hora, o consumo de corrente instantânea do NodeMCU é armazenado. É importante frisar que a placa de desenvolvimento NodeMCU já possui um divisor de tensão na entrada do conversor AD [Nodb], observado na figura 3.2.

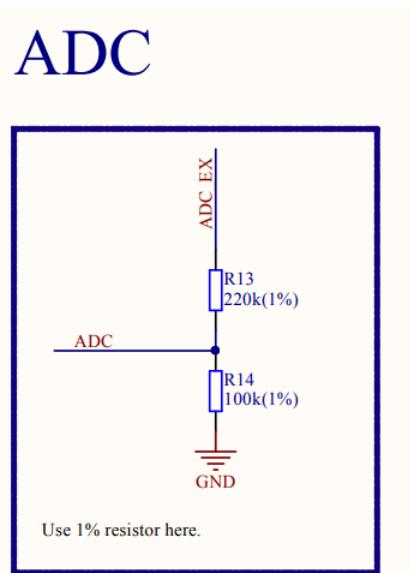


Figura 3.2 – Divisor de tensão nativo na entrada A0

Para a obtenção das medições, foi construído um pequeno circuito com um resistor shunt (figura 3.3). A tensão fornecida para a placa foi de cinco volts, a partir de uma fonte de bancada. O consumo de corrente médio ficou perto dos 85mA, como pode ser constatado pela linha azul da figura 3.4. Os picos de corrente de aproximadamente 140mA ocorrem quando o microcontrolador efetua alguma operação de TX.

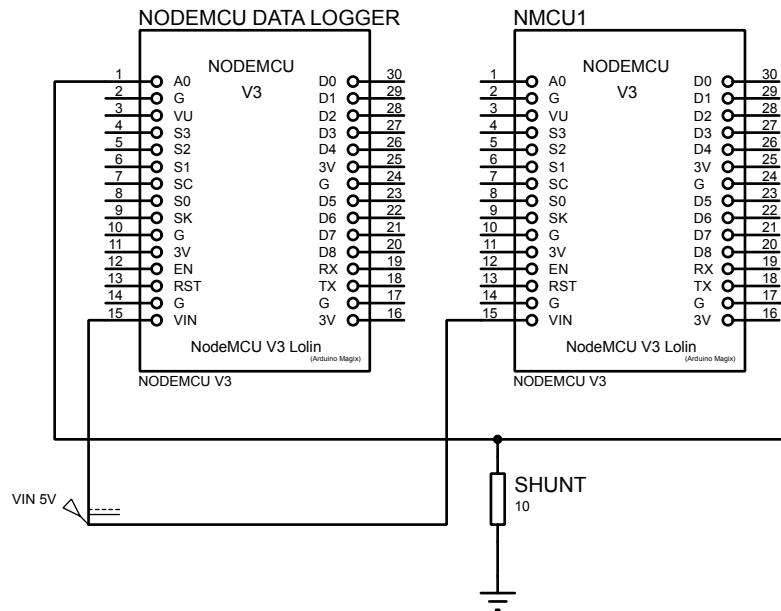


Figura 3.3 – Abordagem simples para obter o consumo médio de corrente

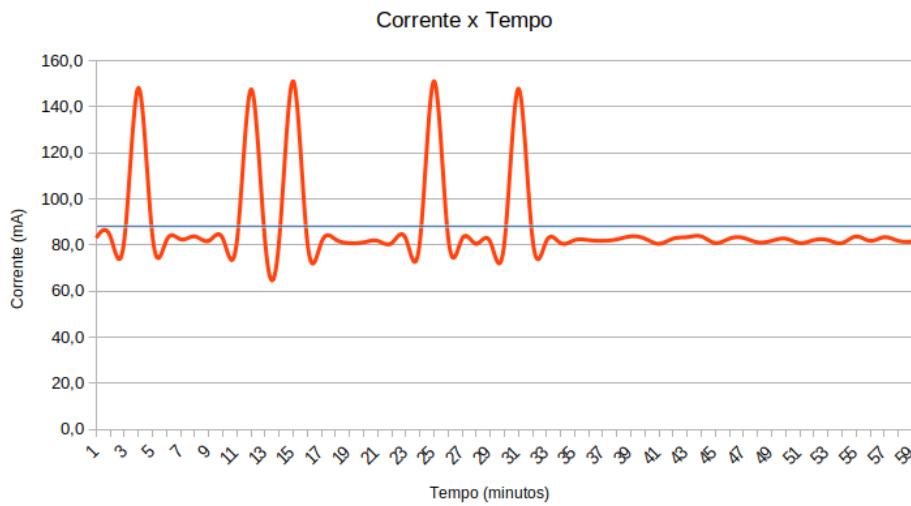


Figura 3.4 – Curva de carga

3.2.2 Especificações do painel fotovoltaico na prática

O fabricante do módulo fotovoltaico afirma que a tensão é de seis volts, enquanto também declara que a placa tem a potência de 1 watt. A partir das Lei de Joule e Lei de Ohm podemos inferir que a corrente máxima do painel é de aproximadamente 160mA. É elementar compreender que estas determinações são obtidas em laboratório com temperatura controlada (25°C), considerando a irradiação média de 1000 watts/m^2 (média da incidência solar na superfície terrestre), ainda levando em conta a massa do ar como 1.5

[BR88]. Na prática, os valores podem diferir consideravelmente e, desta forma, é fundamental que o equipamento seja averiguado.

Conhecendo-se a curva característica I-V da maioria dos painéis fotovoltaicos (Figura 3.5), munidos, também, com as medidas I_{sc} (corrente de curto-círculo) e V_{oc} (tensão de circuito aberto), podemos inferir o ponto máximo de potência (MPP).

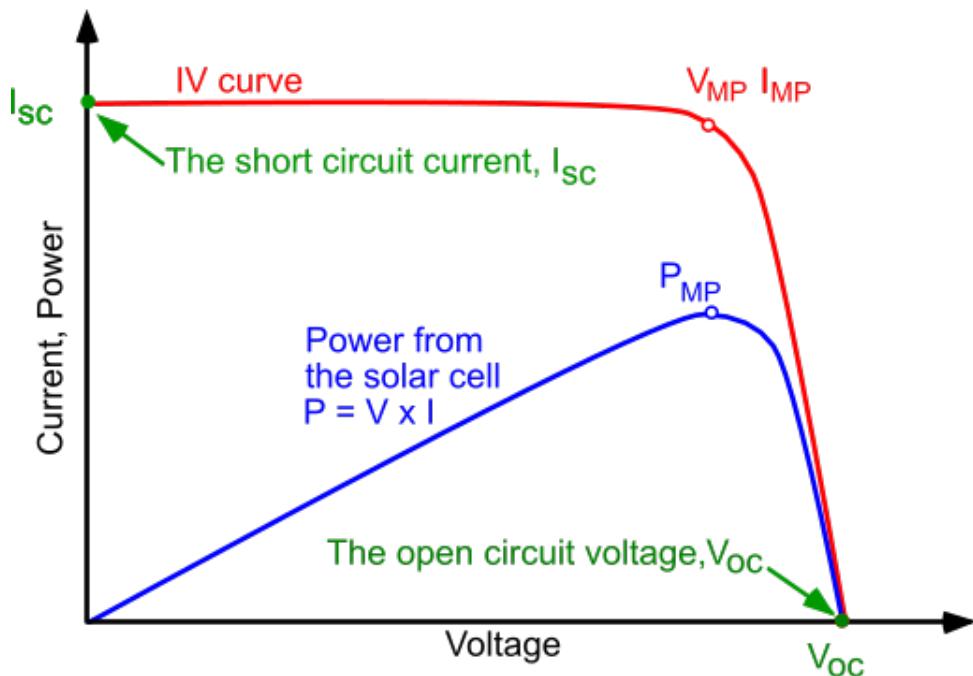


Figura 3.5 – Curva I-V característica

Um fator que influencia a eficiência energética é o ângulo de incidência solar. Sendo assim, as medidas de corrente de curto-círcito (I_{sc}) e tensão de circuito aberto (V_{oc}) instantâneas foram feitas no período de zênite solar. Estas medições podem ser observadas na tabela 3.1.

Tabela 3.1 – Aferições de tensão e corrente em período de zênite solar. Ângulo de incidência aprox. 90º

Corrente de curto-círcuito	130 mA
Tensão de circuito aberto	6,4 Volts

Após as averiguações supracitadas, pode-se inferir, de maneira pessimista, que a V_{mp} (tensão de máxima potência) e a I_{mp} (corrente de máxima potência) estão aproximadamente em 6 Volts e 100mA, respectivamente. Esse resultado leva a uma potência de 0,6 watts.

4. PROJETO DE SOFTWARE

O desenvolvimento do *software* para o trabalho foi dividido em seis etapas. As linguagens utilizadas foram *C++*, *Javascript*, *CSS* e *HTML*.

4.1 Desenvolvimento Web

Visto que um dos objetivos do trabalho é servir páginas HTML dentro da rede fornecida pelo NodeMCU, o desdobramento do projeto de *software* inicia a partir desta etapa. Ela envolve linguagens como HTML, *Javascript* e CSS. Além de exigir habilidades de engenharia reversa em *scripts* "minificados" (técnica utilizada para dificultar a leitura de código e/ou diminuir o tamanho do mesmo) e ofuscadores de estilo. Elas são necessárias para poder ajustar o comportamento das páginas, permitindo afinidade junto às páginas originais. Junto às páginas, também foi desenvolvida uma pequena API de configuração.

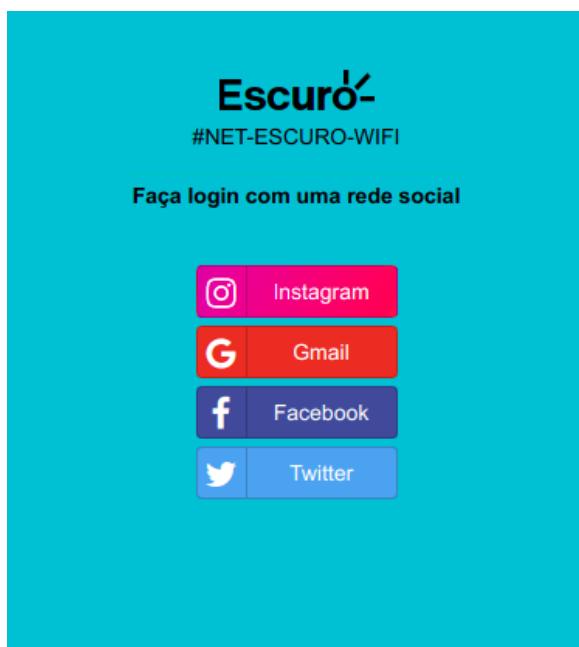


Figura 4.1 – Página inicial do portal desenvolvido

Para o desenvolvimento da prova de conceito já mencionada, usou-se como exemplo a rede pública "#NET-CLARO-WIFI". A primeira página a ser desenvolvida é a página inicial do portal cativo, ela abre automaticamente logo após a conexão do usuário com a rede, e, por isso e pelas características do ataque, i.e., *phishing* por meio de engenharia social, a página deve passar confiança para o desavisado. Também é imprescindível que a página seja responsiva, permitindo a adequação aos mais diversos tipos de tela, presentes em computadores e *smartphones*. Alguns exemplos de portais cativos foram observados

pelo autor durante o dia-a-dia. Suas visualizações estão disponíveis na figura 4.2. Ambas as redes observadas são vulneráveis ao tipo de ataque que este trabalho aborda.

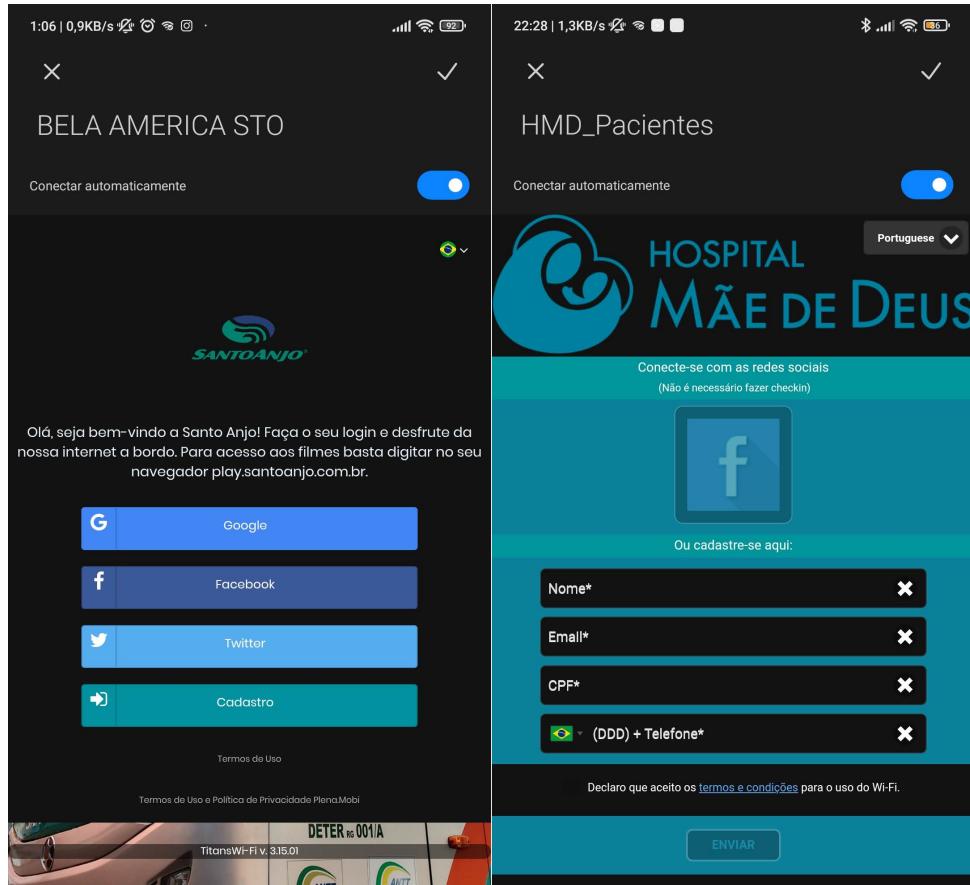


Figura 4.2 – Exemplos de Portais Cativos

No âmbito de campanhas de conscientização de cibersegurança é importante que o portal cativo siga a identidade visual da empresa, incluindo cores, símbolos e figuras. Desta forma, visando a execução da investida na rede fictícia "#NET-ESCURO-WIFI", estabeleceu-se um esquema de cores da companhia ilustrativa, juntamente com o logo. Os botões de CTA para a autenticação por meio de redes sociais também foram desenvolvidos seguindo o padrão de cores das plataformas escolhidas, tal qual observado na figura 4.1.

A primeira página de autenticação elaborada foi a do *Facebook*. As definições de CSS fundamentais da página foram clonadas manualmente, e tudo o que não é essencial para o funcionamento da proposta foi removido. Desta forma, economiza-se espaço na memória do microcontrolador. A comparação entre o portal original e o portal clonado pode ser vista na figura 4.3. Em todas as páginas a funcionalidade de criação de conta e recuperação de senha foram removidas, pois não fazem parte do escopo deste trabalho.

As demais páginas de autenticação, referentes às outras redes sociais foram elaboradas seguindo o mesmo processo. A tarefa mais árdua para o desenvolvimento desta etapa foi a de reduzir o tamanho das páginas, juntamente com fontes, imagens, estilos, entre outros, para não ultrapassar o limite de 1MB do sistema de arquivos. O microcontrolador

permite aumentar esse tamanho, entretanto a função de *upload* de código OTA exige um espaço mínimo na memória de 2M.

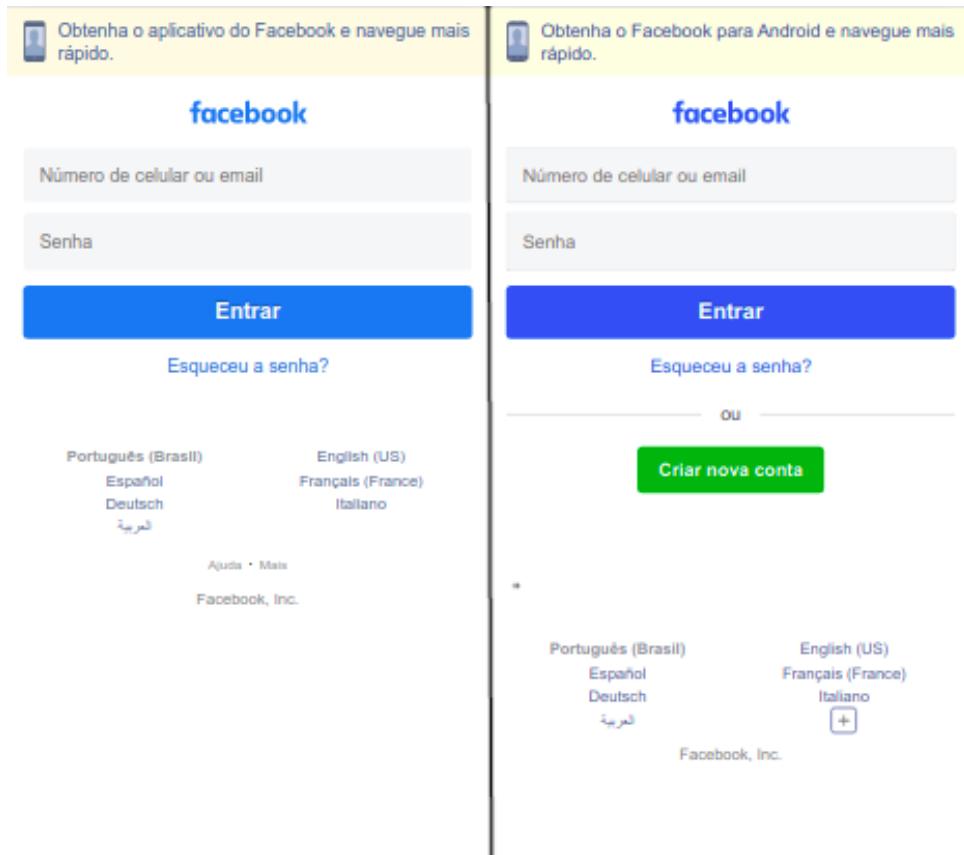


Figura 4.3 – Comparativo: página clonada à esquerda, página original à direita

A página exibida ao usuário no momento em que a autenticação é efetuada foi desenvolvida de modo a exibir dicas de comportamento em redes públicas e como não cair em ataques de *phishing*. A página pode ser vista na figura 4.4. Caso o usuário clique no link disponibilizado (ver figura), ele baixará um arquivo *.pdf* contendo mais dicas. Este documento pode ser visto na figura 4.5.

Por fim, uma página simples de configuração foi elaborada e pode ser observada na figura 4.6.

4.2 Abertura automática do portal cativo em diferentes dispositivos

Essencialmente, o portal cativo deve ser aberto automaticamente no momento em que o usuário conecta-se à rede. Para isso, os dispositivos fazem checagens distintas. Foram adicionadas rotas com os endereços mais comuns de checagem no servidor *web*. Desta forma, uma gama enorme de computadores e celulares terá exatamente o comportamento esperado.

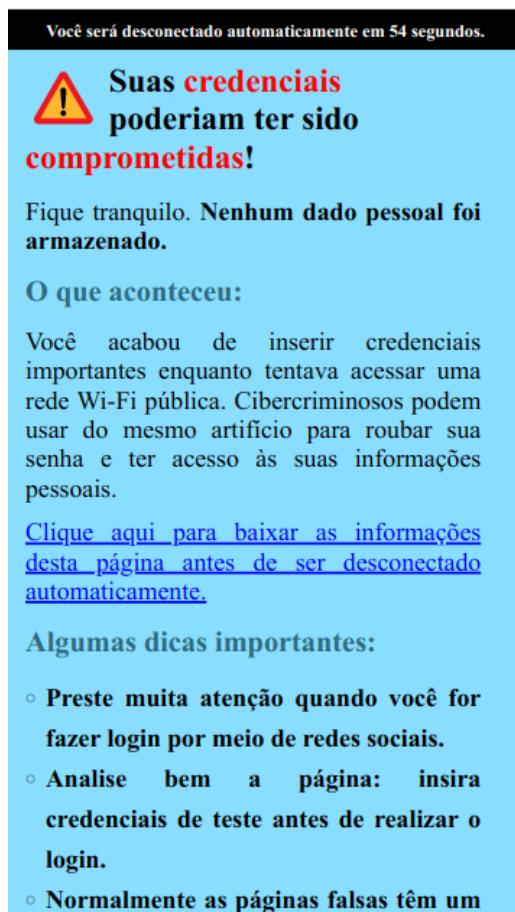


Figura 4.4 – Página final

Ops! Parece que você ainda não aprendeu...

- Nunca baixe arquivos de fontes desconhecidas. Cibercriminosos, em alguns casos, podem obter **controle total** do seu dispositivo a partir de um arquivo malicioso.
- Nos dispositivos Android, mantenha a opção "Permitir instalação de fontes desconhecidas" desativada.
- Em computadores, procure sempre manter seu sistema operacional atualizado. As atualizações corrigem falhas de segurança e mantêm o seu dispositivo protegido.

Abaixo, seguem alguns links com artigos interessantes sobre segurança online.

- <https://www.bbc.com/portuguese/geral-38313175>
- <https://nordvpn.com/pt-br/blog/identificar-sites-falsos/>

Para lugares que pedem cadastro de e-mail, você pode usar um e-mail temporário:

- <https://temp-mail.org/pt/>

A tecnologia é uma dádiva, mas, assim como em (quase) todos os âmbitos de nossa vida, sempre tem alguém querendo lhe passar a perna ;)

Seja cuidadoso (a).

Figura 4.5 – PDF contendo mais dicas

Alguns telefones *Android* possuem uma verificação diferente das outras. E a solução das rotas não é suficiente. A partir disso, define-se o IP de *Gateway Padrão* da rede como "8.8.8.8". Isto se dá porque alguns modelos (especialmente os da *Samsung*) utilizam o servidor de DNS do Google (8.8.8.8) de maneira fixa no código do sistema operacional.

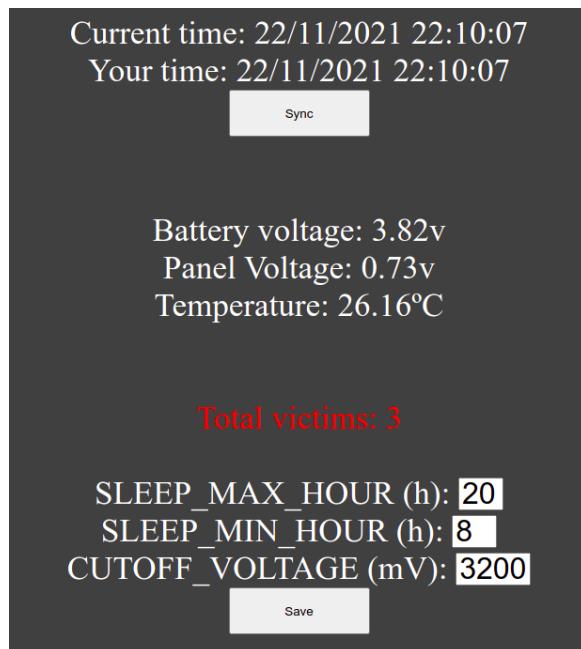


Figura 4.6 – Página de configurações

Após essa modificação, todos os telefones que foram testados abriram automaticamente o portal.

4.3 *Upload remoto - OTA*

A biblioteca *ArduinoOTA* implementa o recebimento de arquivos binários no micro-controlador, e, posteriormente, a gravação deles na memória flash do dispositivo. Desta maneira, não há necessidade de proximidade física da *MCU*, desde que o computador que está enviando o código esteja na mesma rede.

Os binários enviados podem ser tanto o *firmware* quanto a imagem do sistema de arquivos *LittleFS*. Este último tem um tamanho limitado a 1MB, pois o espaço reservado para os binários OTA é de 2MB. Assim, somando-se estes 2MB com o tamanho do sistema de arquivos e com o tamanho máximo do código-fonte (1MB), totalizam-se 4MB, que é justamente a extensão de memória do *NodeMCU v.1.0*. Esta divisão da memória *flash* pode ser observada na figura 4.7 [Pla]. A configuração do layout da *flash* é feita na IDE, usando o parâmetro "ld-script"[For].

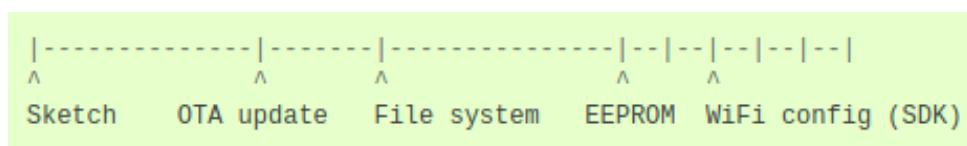


Figura 4.7 – Layout da memória Flash [Pla]

4.4 Funções de execução periódica - *Tickers*

É essencial para o projeto que algumas funções sejam executadas repetidamente, em um determinado período. Para isso, usa-se o conceito de *Tickers*. Eles são uma excelente alternativa ao uso de funções de *delay*, pois não são bloqueantes.

Ticker Soft-RTC

Para manter o controle temporal, um *ticker* com período de um segundo é utilizado. A função incrementa uma variável que armazena a data atual a partir do horário *Unix*. O valor inicial da data é definido a partir do *header* de configurações. Durante a execução do programa, a data pode ser calibrada a partir do *endpoint* de configuração desenvolvido.

Ticker de desconexão

A cada cinco segundos, a função de *deauthentication* verifica se existe algum cliente deve ter a desconexão forçada.

4.5 Controle de conexão por endereço MAC

Visto que o chip *ESP-12E* tem uma limitação de clientes conectados (devido à memória RAM), existe a necessidade de manter um controle de usuários ativos na rede. Uma vez que um internauta complete o fluxo de autenticação e seja redirecionado para a página final, ele deve ser desconectado automaticamente da rede em um tempo predeterminado. Isto se deve ao fato da limitação de conexões simultâneas, mencionado anteriormente. Portanto, com este intuito, o endereço MAC do cliente "autenticado" é armazenado no momento em que ele efetua o *login*, juntamente com a hora do evento. Após um minuto o usuário é desconectado, usando o artifício do *deauthentication frame*, mencionado no capítulo 1 deste trabalho.

O *SDK* da *Espressif* (fabricante dos *chips* *ESP*) costumava permitir o envio de *frames ethernet* manipulados, a partir da função "*wifi_send_pkt_freedom*", entretanto, após a constatação do mau uso desta função [Spa], a empresa resolveu removê-la do *SDK*. Com isso, ataques de desautenticação não são possíveis com as versões mais recentes do *software*. Existem dois contornos para esta situação: usa-se uma versão antiga do *SDK*, com *bugs* conhecidos e menos funcionalidades, ou executa-se engenharia reversa no *firmware* a nível de código de máquina *Assembly* para identificar o que foi modificado da versão an-

tiga até a versão nova. Visto que algumas funcionalidades facilitam o desenvolvimento do *software*, a segunda opção foi escolhida.

4.5.1 Engenharia reversa de binário com o *GHidra*

O *GHidra* é uma ferramenta de engenharia reversa desenvolvida pela Agência de Segurança Nacional dos Estados Unidos (NSA). É extremamente útil e poderosa para a análise de binários. Neste trabalho analisa-se o arquivo *firmware.elf*, compilado para ser escrito na memória *flash* do *NodeMCU*. A análise foi iniciada na função removida, citada anteriormente, e pode ser observada na figura 4.8.

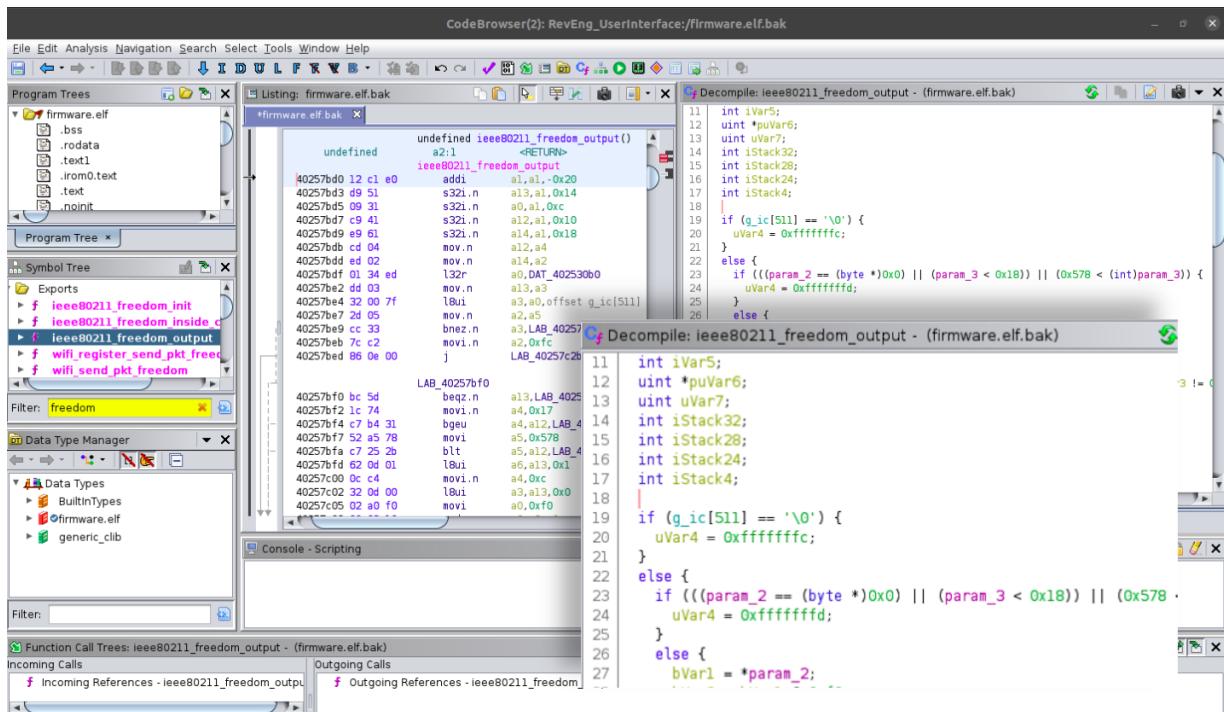


Figura 4.8 – Assembly da função

Após uma longa investigação, algumas modificações para evitar a verificação dos primeiros bytes do *deauth frame* foram feitas diretamente no binário, em hexadecimal. Posteriormente, o arquivo foi gravado na memória do microcontrolador e a função de envio voltou à funcionalidade dos *SDKs* antigos. O grande problema é que toda vez que o *firmware* for compilado, o binário deve ser modificado. Isto é maçante e contraproducente. Pesquisando e examinando mais a fundo as funções definidas no *ELF*, encontra-se a rotina "wifi_softap_deauth"(ver figura 4.9). Ela não está exposta em nenhuma documentação do *software* em *C/C++*, seja ela da *Arduino* ou da *Espressif*, porém, em uma busca no *firmware* escrito em *LUA*, existe uma função "wifi.ap.deauth()", que por sua vez, pode ser encontrada no seu respectivo repositório [Noda], onde, finalmente, encontra-se o uso da função "wifi_softap_deauth".

The screenshot shows the CodeBrowser interface with the following panes:

- Left pane:** Assembly code for the `wifi_softap_deauth` function. It includes assembly instructions like `addi a1,a1,-0x20`, `beqz a2,LAB_40274e93`, and `bnez a2,LAB_40274e93`. A filter bar at the bottom left says "Filter: deauth".
- Middle pane:** C decompiled code for the same function. It includes variables `iVar1`, `iVar2`, `iVar3`, `iVar4`, `iVar5`, and `iVar6`, and a conditional block checking if `iVar2` is 1 or 0.
- Bottom panes:** Function Call Trees, Incoming References, and Outgoing References for the `wifi_softap_deauth` function.

Figura 4.9 – Assembly da função

A fim de completar a funcionalidade foi implementado um *Ticker*, executado a cada cinco segundos e verifica se existe algum cliente na rede que precisa ser desconectado, como mencionado na seção acima. A limitação dessa abordagem é que os dispositivos mais recentes implementam a anonimização do endereço *MAC*, justamente para proteger o usuário da coleta de dados indevida. Isto faz com que, cada vez que a pessoa conecte-se à rede, um *MAC* aleatório seja gerado e usado como endereço real do aparelho. Levando este fato em consideração, é impossível bloquear o acesso de uma pessoa à rede a partir do endereço físico, caso ela queira acessá-la novamente. Não obstante, é possível, ao menos, desconectá-la periodicamente, a partir da abordagem supracitada.

4.6 Controle de energia - *Deep Sleep*

O controle de energia do *NodeMCU* é feito por meio de aferições da tensão da bateria que alimenta o sistema. Essas aferições ficam salvas em memória, e, a partir da variação de tensão, pode-se definir se a bateria está sendo carregada ou descarregada. Caso o nível esteja abaixo do limiar definido, o microcontrolador deverá entrar em estado de "sono profundo", com o objetivo de economizar energia, até que a carga possa ser retomada novamente, através dos painéis fotovoltaicos. Quando o nível de tensão nos painéis solares é baixo em períodos em que, teoricamente, deveria ser alto, assume-se que o tempo está nublado, e o microcontrolador pode entrar em *deep sleep* antes mesmo da tensão na bateria atingir o limiar inferior. O limite inferior de potencial na bateria é definido em 3,2 volts, visto que a indústria define o limite em 3 volts e as baterias são provenientes

de descarte, e, portanto, já não têm a mesma capacidade de carga. Portanto, este limite deve prover uma longevidade maior às células.

Ademais, o microcontrolador pode ser desligado em horários específicos, também configuráveis a partir do cabeçalho de configurações. Isso permite que o baixo consumo seja garantido em horários de baixa utilização, como durante a madrugada. O período padrão de sono profundo é entre as 20h e as 08h.

5. PROJETO DE *HARDWARE*

O objetivo do projeto de *hardware* deste trabalho é desenvolver um *shield* para o *NodeMCU* com o objetivo de conseguir efetuar o gerenciamento de energia do microcontrolador de maneira eficiente. Para tal propósito, pode-se aferir a diferença de potencial elétrico dos painéis fotovoltaicos, que alimentam o circuito de carga, e a tensão da bateria. Dado o fato já mencionado na seção 3.2, é necessário o uso de divisores de tensão para potenciais acima de 3,3 volts. O conceito de *upcycling*, introduzido anteriormente, será usado neste trabalho, e, portanto, todos os componentes, com exceção do microcontrolador e da placa de prototipação, serão provenientes de lixo eletrônico. O esquemático completo do circuito pode ser observado no apêndice A

5.1 Divisores de tensão

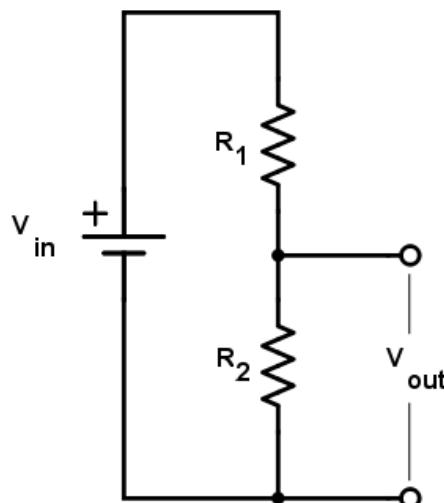


Figura 5.1 – Divisor de tensão [Cir]

Foram projetados dois divisores: um para a tensão da bateria e outro para a tensão dos painéis fotovoltaicos. As tensões instantâneas máximas, aferidas por um multímetro podem ser observadas na tabela 5.1.

Tabela 5.1 – Parâmetros dos divisores de tensão

Fonte	Parâmetros			
	V. Máx (V)	R1 (Ω)/R2 (Ω)	Vin Máx. (V)	Razão Vout/Vin
Painel Solar	6,4	15k/10k	8,3	0,4
Bateria	4,2	10k/10k	6,6	0,5

5.1.1 Multiplexação e consumo dos divisores

Imprescindivelmente, o circuito que envolve os divisores de tensão deve ter a capacidade de ser ativado ou desativado. Inicialmente, um multiplexador (*HCF4053BE*) seria usado na entrada do conversor A/D, com apenas um *MOSFET* para habilitar e desabilitar *VCC*. Infelizmente, foi constatado que havia uma fuga de energia pelos pinos do *MUX*, mesmo quando o CI estava sem tensão em *VDD* ou sem *GND*. Para contornar o problema, *MOSFETs* (*IRLB8721*, provenientes de um retroprojector encontrado na rua) foram utilizados, sendo que, na saída de cada divisor de tensão, diodos *1N4148* foram adicionados para evitar interação entre os circuitos, quando conectados em paralelo, durante as aferições de tensão pela entrada analógica.

Os transistores usados no circuito têm uma tensão *V_{th}* muito alta para serem conduzidos diretamente pelo NodeMCU. Desta forma, optoacopladores (escolhidos, também, porque estavam disponíveis no descarte eletrônico) são empregados para ativar o *gate* dos *MOSFETs* com tensão suficiente, a partir dos pinos de *GPIO* do microcontrolador. Esta solução, traz, também, o benefício de não haver consumo energético (tão baixo que é quase desprezível) quando os *gates* estão em sinal lógico baixo. Já, quando os divisores são acionados, o seu consumo fica em $135\mu\text{A}$.

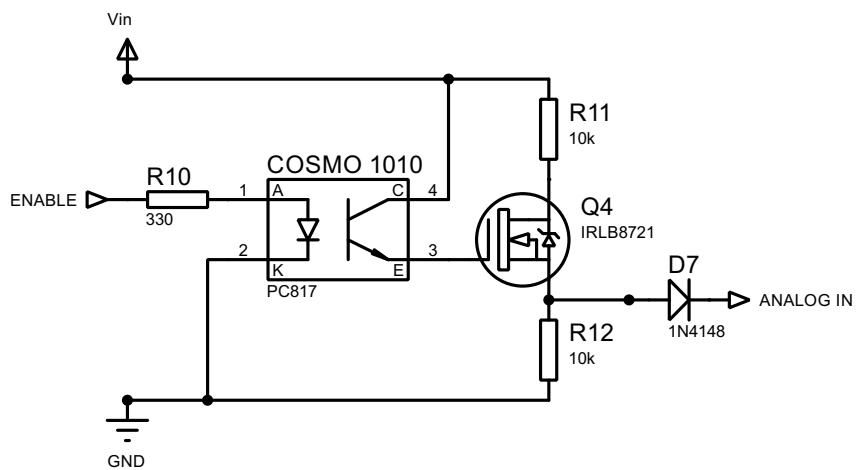


Figura 5.2 – Divisor de tensão projetado

5.1.2 Compensação por temperatura

Devido à natureza dos diodos, resistores e *MOSFETs* usados no circuito, a temperatura é um fator que influencia nas medições de tensão. Os diodos, em especial, utilizados

na entrada do conversor A/D, sofrem alterações significativas na sua tensão direta, conforme a variação de temperatura. Pensando em mitigar esta situação, o sensor *LM35* foi adicionado ao projeto. Este CI opera com uma saída linear de 10mV por grau Celsius.

Levando estas informações em consideração, o próximo problema a ser mitigado é a queda de tensão em decorrência do diodo na saída do *LM35*. Neste caso, o microcontrolador só começaria a detectar a temperatura a partir dos 70º Celsius, no mínimo. Desta forma, dois diodos foram adicionados no pino *GND* do CI. Isto cria um *ground virtual*, fazendo com que a saída do sensor tenha um *offset* de aproximadamente 1,2 volts, o que já é suficiente para o conversor A/D. Após a adição do sensor, foi implementado em software uma correção linear para a leitura de tensões, baseada na temperatura atual dos componentes. O esquemático completo pode ser encontrado nos anexos, ao fim deste trabalho.

5.2 Circuito de Carga da Bateria

O carregador de bateria foi reaproveitado de um *power bank* proveniente de desarte. Com a finalidade de entender o seu funcionamento, um processo de engenharia reversa foi executado. Com um multímetro no modo de continuidade foram seguidas todas as trilhas da placa de circuito impresso, componente por componente. Provado com os nomes dos componentes da placa, uma busca pelos *datasheets* foi efetuada. Em seguida, uma ferramenta de *CAD* foi utilizada para desenhar o circuito, que pode ser visualizado na figura 5.3. Após a pesquisa, concluiu-se que o carregador suporta todo o intervalo de tensão fornecido pelos painéis fotovoltaicos. A saída do diodo D2 fornece 5 volts, perfeito para alimentar o NodeMCU.

5.3 Reduzindo o consumo energético do NodeMCU

Conforme o *datasheet* da *Espressif*, o chip *ESP-12E* consome, em estado de sono profundo, 10 μ A. Ao medir o consumo do NodeMCU, percebe-se que em *deep sleep*, a corrente é de 15 mA, muito acima do especificado. Os esquemáticos do microcontrolador foram investigados, em busca de alguma pista sobre o alto consumo energético. O chip *CH340G*, usado como interface *USB/UART* tem 12mA como corrente típica de funcionamento, segundo suas especificações [Nan19]. O chip não é necessário quando não se está usando a USB da placa, e, portanto, pode ser desativado. Com esta finalidade, o pino 16 (*VCC*) foi removido da *PCB*. Em seu lugar foi inserido um *jumper*, para que o CI possa ser utilizado novamente, caso o uso da interface *USB* seja necessário. Após o desligamento do circuito, o consumo máximo em *deep sleep* não passou de 5mA. O maior vilão é o regulador de tensão linear *AMS1117*, usado para rebaixar a tensão de entrada até 3,3 volts. Caracteristi-

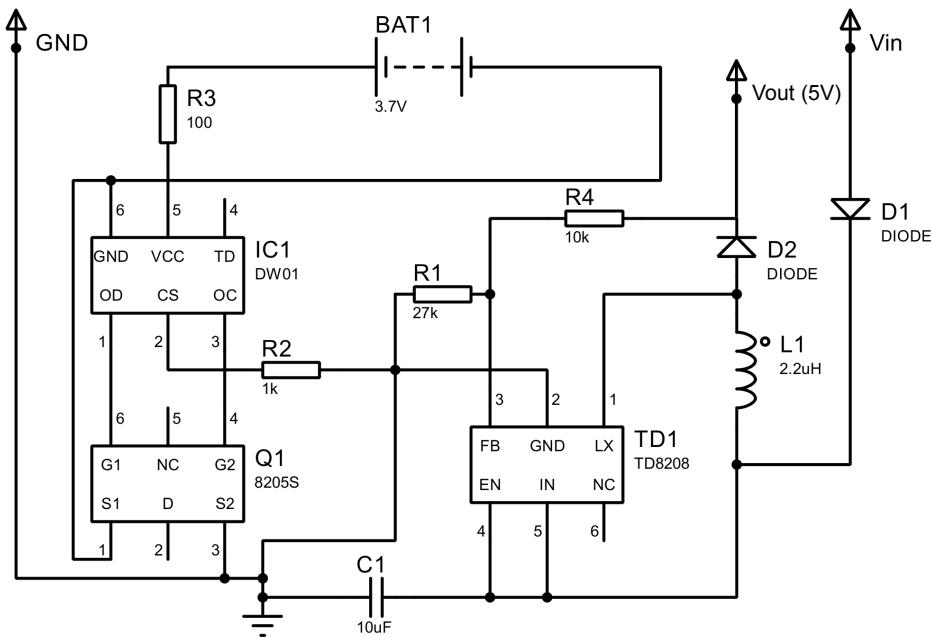


Figura 5.3 – Engenharia reversa no circuito de carga

camente, reguladores lineares são ineficientes, pois desperdiçam energia através do calor, por meio do efeito Joule. Infelizmente, não foi possível realizar a troca do regulador por um conversor DC-DC (mais eficiente), devido ao curto tempo para a execução do projeto. Ademais, as alterações feitas na placa podem ser vistas na figura 5.5.

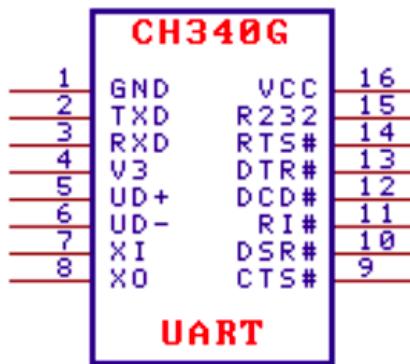


Figura 5.4 – CH340G: Pinout

5.4 Dimensionamento dos painéis

O cálculo de dimensionamento dos painéis foi feito de maneira empírica, observando o comportamento de tensão e corrente durante a conexão dos painéis ao circuito já montado, objetivando potência suficiente para a carga das células de bateria, somado ao consumo do microcontrolador. Desta forma, o NodeMCU, já com o *shield* desenvolvido, foi

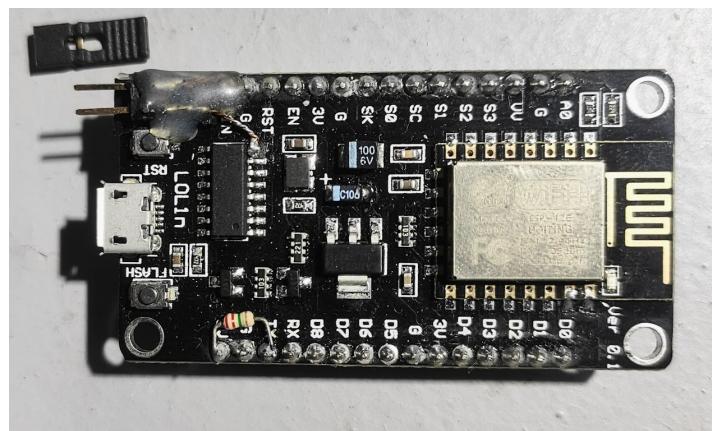


Figura 5.5 – Jumper adicionado ao CI USB/UART (canto superior esquerdo)

alimentado incrementalmente de um até cinco painéis. As medições podem ser conferidas na tabela 5.2. Além disto, uma curva da variação de corrente ao longo do tempo foi gerada, somente com uma célula solar, usando o mesmo circuito demonstrado na seção 3.2. Esta pode ser observada na figura 5.6. Após a avaliação, foi escolhido o uso de quatro células fotovoltaicas, visando reduzir o estresse referente ao calor no controlador de carga.

Tabela 5.2 – Dimensionamento. Os sinais de +/- representam o sentido da corrente.

Nº de Painéis	Medidas instantâneas	
	Tensão no painel(V)	Corrente aprox. na bateria (mA)
1	4,73	- 20mA
2	4,96	+ 80mA
3	5,10	+ 180mA
4	5,74	+ 280mA
5	5,93	+ 380mA

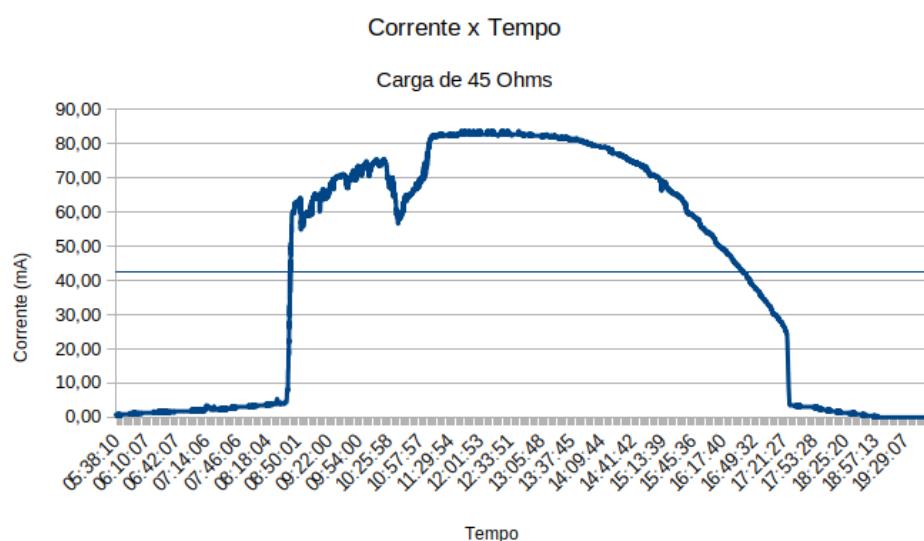


Figura 5.6 – Curva com carga linear

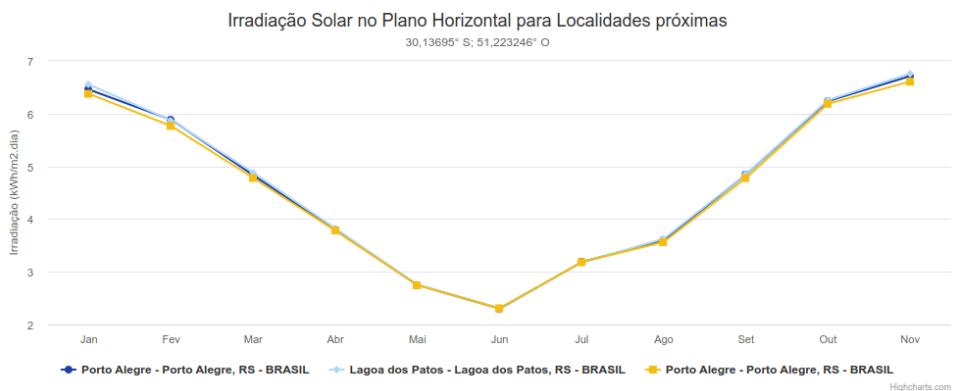


Figura 5.7 – Irradiação solar no plano horizontal - Porto Alegre [CRE18]

Cálculo de consumo energético

Observa-se na curva 5.6 que tem-se aproximadamente oito horas úteis de incidência solar. A grande variação perto das 09h e 17h, dá-se pela incidência de sombra projetada por objetos próximos ao painel. A equação da figura 5.8 ilustra o consumo diário, onde Im é o valor médio de corrente utilizado pelo microcontrolador, $Imds$ é o valor médio de corrente em sono profundo, e t é o tempo em que o microcontrolador fica com RF ligado. Já a equação da figura 5.9 ilustra o cálculo de superávit/déficit energético diário, onde ts é o tempo útil de incidência solar. Assumindo condições pessimistas para os cálculos, inferimos que há somente seis horas úteis de incidência solar, e 200mA de corrente entrando na bateria. Da mesma forma, assume-se o período de uso padrão 08h-20h, resultando em 12h de uso. Por conseguinte, ao aplicar estes valores nas equações, temos um superávit energético diário de 108mA.

$$C(t) = Im * t + Imds * (24 - t)$$

Figura 5.8 – Função de consumo energético diário (mA)

$$S(t, ts) = Ipv * ts - C(t)$$

Figura 5.9 – Função de superávit/déficit energético diário (mA)

Garantindo o superávit energético, o sistema consegue manter-se por longos períodos de tempo apenas usando a bateria. Visto que sua capacidade é de 10000mAh, o microcontrolador, em *deep sleep*, tem energia suficiente para permanecer neste estado por 1666 horas (aproximadamente 70 dias).

6. VALIDAÇÃO

Este capítulo descreve as atividades realizadas para validar o funcionamento do sistema.

- **Atividade 1** - Preparar ambiente de desenvolvimento.

Objetivo: Preparar e instalar ferramentas de desenvolvimento.

Resultado esperado: Ambiente de desenvolvimento configurado. Todas as bibliotecas e softwares utilizados para o desenvolvimento funcionando em versões confiáveis e estáveis.

Resultado atingido: SIM

- **Atividade 2** - Desenvolver prova de conceito com NodeMCU

Objetivo: Validar uso do microcontrolador como *SoftAP* e retorno correto do portal cativo.

Resultado esperado: Uso validado, exibição correta e automática do portal cativo.

Resultado atingido: SIM

- **Atividade 3** - Medir consumo energético do microcontrolador em seus diferentes estados

Objetivo: Obter medições nos estados de *deep sleep* e transmissão.

Resultado esperado: consumo até 140mA em *full TX* e 10uA em *deep-sleep* conforme especificado no *datasheet* do *SoC*.

Resultado atingido: PARCIALMENTE

Comentários: O consumo em *deep-sleep* revelou-se muito maior do que o esperado. Isto exigiu algumas implementações extras.

- **Atividade 4** - Validar painéis fotovoltaicos e juntá-los ao protótipo

Objetivo: Validar se a corrente média do painel é suficiente para manter o microcontrolador funcionando.

Resultado esperado: Corrente de curto-círcuito próxima a 170mA, operando em 6 volts.

Resultado atingido: PARCIALMENTE

Comentários: A corrente de curto circuito é menor do que a indicada pelo fabricante, como citado na seção 3.2. Desta forma, foi necessário o uso de mais painéis.

- **Atividade 5** - Finalizar todas as páginas de autenticação

Objetivo: Disponibilizar uma gama de métodos de autenticação no portal malicioso.

Resultado esperado: O sistema deverá suportar ao menos *Facebook*, *Instagram*, *Twitter* e *Gmail*

Resultado atingido: SIM

- **Atividade 6** - Implementar desconexão de usuário e bloqueio de reconexão por endereço MAC

Objetivo: Evitar o congestionamento da rede.

Resultado esperado: Desconectar o usuário ao final do ciclo de autenticação.

Resultado atingido: SIM

- **Atividade 7** - Teste *burn-in* local

Objetivo: Validar o comportamento do microcontrolador, juntamente com a validação do sistema de energia.

Resultado esperado: O sistema não deverá travar nem ter reinícios inesperados, assim como deve manter a bateria em um nível estável.

Resultado atingido: SIM

Comentários: O sistema ficou ligado durante uma semana, na rua, entre tempos ensolarados e chuvosos, mantendo a tensão da bateria em níveis aceitáveis e entrando em modo de energia quando necessário.

- **Atividade 8** - Implementação em campo

Objetivo: Validar a solução em local público, mantendo um contador de usuários.

Resultado esperado: Algumas pessoas, pelo menos, devem finalizar o ciclo de autenticação.

Resultado atingido: SIM

Comentários: O sistema foi posto à prova em um pequeno evento privado. No total, três pessoas percorreram todo o fluxo. Infelizmente o prazo curto e o risco de roubo do protótipo impediu o teste em locais públicos.

7. CONCLUSÃO

Os resultados obtidos no trabalho alcançaram os objetivos esperados. A exploração do poderoso microcontrolador NodeMCU foi complexa, divertida e possibilitou o descobrimento de uma gama de possibilidades para distintas soluções no âmbito da Internet das Coisas, usando o mesmo microcontrolador. O projeto de *hardware* se mostrou eficiente para a solução desejada, ainda mais com a limitação de somente utilizar componentes reutilizados de outros equipamentos. Já o projeto de *software* serviu para aprofundar os conhecimentos em desenvolvimento de sistemas embarcados, trazendo à superfície algumas de suas limitações e maneiras inteligentes de contorná-las. Alguns atrasos inesperados afetaram o projeto, tendo como exemplo a engenharia reversa feita no *firmware* do microcontrolador. Este contratempo poderia ter sido evitado caso a documentação explicitasse a existência e/ou o funcionamento das funções analisadas.

Visto que o protótipo foi testado em um evento familiar, onde três pessoas, entre dez, concluíram o fluxo (em apenas um dia de teste), é possível inferir que com um universo maior de vítimas em potencial, somado a um maior tempo de exposição da rede, a quantidade de indivíduos a completar o fluxo seria grande, tomando como métrica otimista os 30% (três, entre dez).

O simples fato de alguns usuários, em uma pequena amostra, terem chegado até o final do processo de *phishing* demonstra que realmente há um perigo no acesso às redes públicas, quando realizado negligentemente. É evidente que a ameaça é real e deve ser trazida à tona. Toda e qualquer empresa, que faz campanhas de *phishing* contra seus funcionários a fim de treiná-los, pode utilizar este sistema de maneira a educar seus empregados e coibir ataques mais graves (como *ransomware*), que podem acontecer caso haja algum vazamento de credenciais.

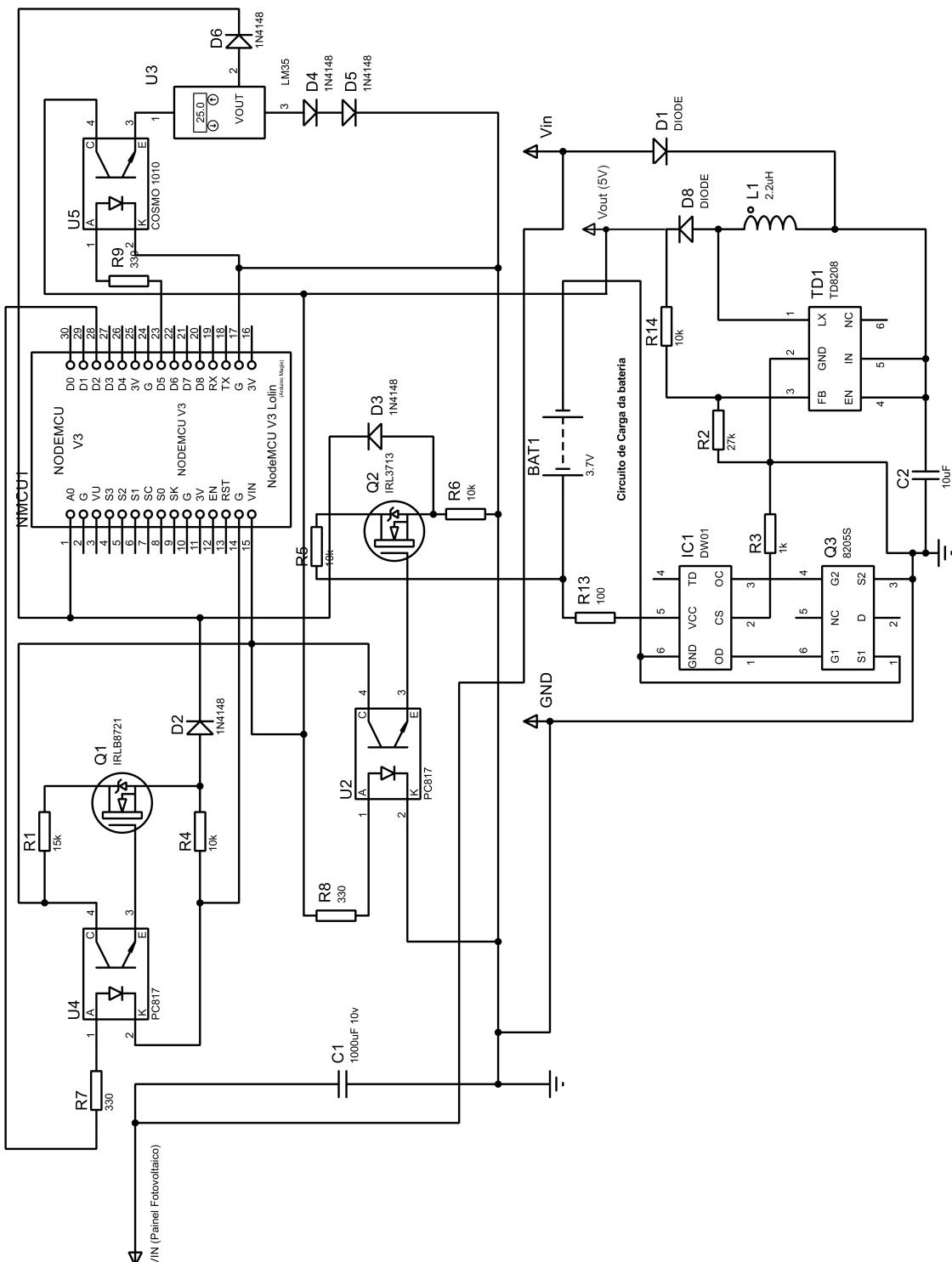
Por fim, várias melhorias podem ser aplicadas futuramente ao projeto: redução de área da PCB (*Printed Circuit Board*), otimização do regulador de tensão do microcontrolador, possibilitando diminuir o consumo em *deep sleep*, implementação de um sensor de corrente. Até mesmo alguma integração para comunicação e configuração remota entre o microcontrolador e redes como LoRaWAN, Sigfox, entre outras. A partir do *software* existente é possível implementar facilmente esse tipo de integração. Desta forma, conclui-se que o trabalho de conclusão atingiu seus objetivos.

REFERÊNCIAS BIBLIOGRÁFICAS

- [AI-11] AI-Thinker. “ESP-12E WiFi Module”, 2011, version 1.0, Capturado em: https://docs.ai-thinker.com/_media/esp8266/docs/esp12e_datasheet.pdf.
- [BR88] Blaesser, G.; Rossi, E. “Extrapolation of outdoor measurements of pv array i–v characteristics to standard test conditions”, *Solar Cells*, vol. 25–2, 1988, pp. 91–96.
- [Cir] Circuits, A. A. “Voltage divider calculator”. Capturado em: <https://www.allaboutcircuits.com/tools/voltage-divider-calculator/>, Nov 2021.
- [CRE18] CRESESB. “Potencial solar - sundata v3.0”. Capturado em: <http://www.cresesb.cepel.br/index.php#data>, Nov 2021.
- [For] Forum, E. C. “Flash split for 4m chips”. Capturado em: <https://github.com/esp8266/Arduino/blob/master/tools/sdk/lld/eagle.flash.4m1m.ld>, Nov 2021.
- [Gro17] Grokhotkov, I. “Esp8266 arduino core documentation”, *ESP8266*, vol. 1–1, 2017.
- [Khi20] Khiralla, F. A. M. “Statistics of cybercrime from 2016 to the first half of 2020”, *IJCSN-International Journal of Computer Science and Network*, vol. 9–5, 2020.
- [Mar14] Marista, R. “Projeto recondicionar”. Capturado em: <https://social.redemarista.org.br/centro/polo-cesmar/iniciativas/projeto-recondicionar1>, Nov 2021.
- [MJVB19] Mihelič, A.; Jevšček, M.; Vrhovec, S.; Bernik, I. “Testing the human backdoor: Organizational response to a phishing campaign”, *JUCS-Journal of Universal Computer Science*, vol. 25, 2019, pp. 1458.
- [Nan19] Nanjing Qinheng Microelectronics Co. “USB CH340G”, 2019, version: 2B, Capturado em: <http://j5d2v7d7.stackpathcdn.com/wp-content/uploads/2016/11/CH340.pdf>.
- [No 19] No Dev, M. “Esp async tcp”. Capturado em: <https://github.com/me-no-dev/ESPAsyncTCP>, 2019.
- [No 21] No Dev, M. “Esp async webserver”. Capturado em: <https://github.com/me-no-dev/ESPAsyncWebServer>, 2021.
- [Noda] NodeMCU. “Lua based interactive firmware for esp8266, esp8285 and esp32”. Capturado em: <https://github.com/nodemcu/nodemcu-firmware>, Nov 2021.

- [Nodb] NodeMCU. “NodeMCU-devkit-v.10”. Capturado em: https://github.com/nodemcu/nodemcu-devkit-v1.0/blob/master/NODEMCU_DEVKIT_V1.0.PDF, Nov 2021.
- [OM12] O’Gorman, G.; McDonald, G. “Ransomware: A growing menace”. Symantec Corporation Arizona, AZ, USA, 2012, 16p.
- [PAG09] Penella, M. T.; Albesa, J.; Gasulla, M. “Powering wireless sensor nodes: Primary batteries versus energy harvesting”. In: 2009 IEEE instrumentation and measurement technology conference, 2009, pp. 1625–1630.
- [Pla] PlatformIO. “Flash size”. Capturado em: <https://docs.platformio.org/en/latest/platforms/espressif8266.html#flash-size>, Nov 2021.
- [PLO20] Piazzon, L.; Lima, B.; Oyadomari, W. “Internet e participação cultural: o cenário brasileiro segundo a pesquisa tic domicílios”, *Revista Internet & Sociedade*, vol. 1–1, 2020, pp. 38.
- [Spa] SpaceKuhnTech. “Affordable wifi hacking platform for testing and learning”. Capturado em: https://github.com/SpacehuhnTech/esp8266_deauther, Nov 2021.
- [SYG10] Song, Y.; Yang, C.; Gu, G. “Who is peeping at your passwords at starbucks?—to catch an evil twin access point”. In: 2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN), 2010, pp. 323–332.
- [WLEM19] Wahyudi, E.; Luthfi, E. T.; Efendi, M. M.; Mataram, S. “Wireless penetration testing method to analyze wpa2-psk system security and captive portal”, *Jurnal Explore STMIK Mataram-Volume*, vol. 9–1, 2019.

APÊNDICE A – ESQUEMÁTICO COMPLETO



APÊNDICE B – *SHIELD*

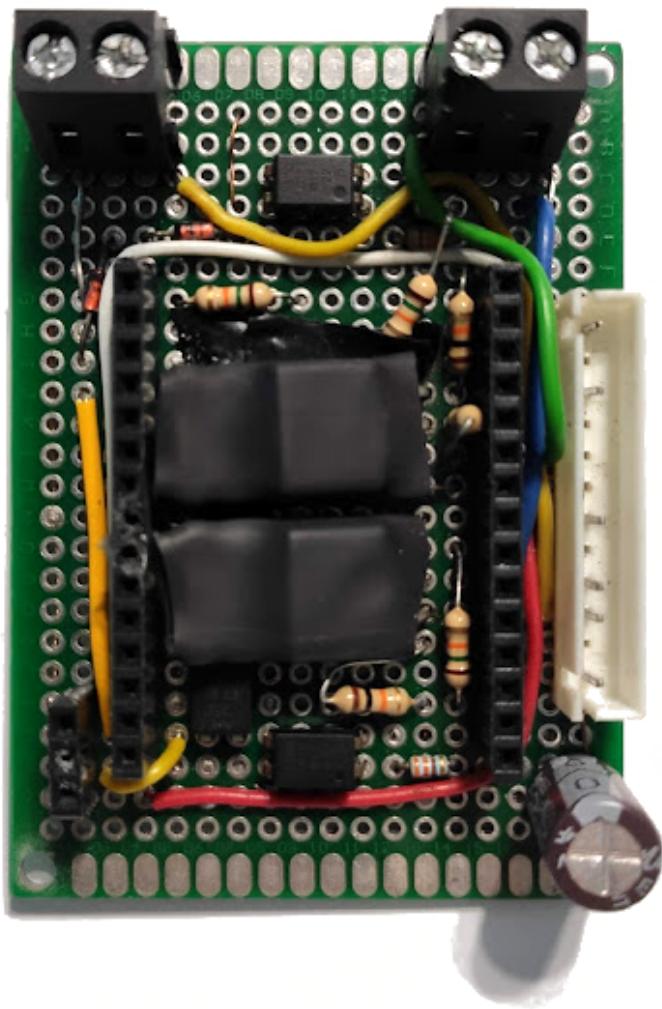


Figura B.1 – *Shield* desenvolvido neste trabalho

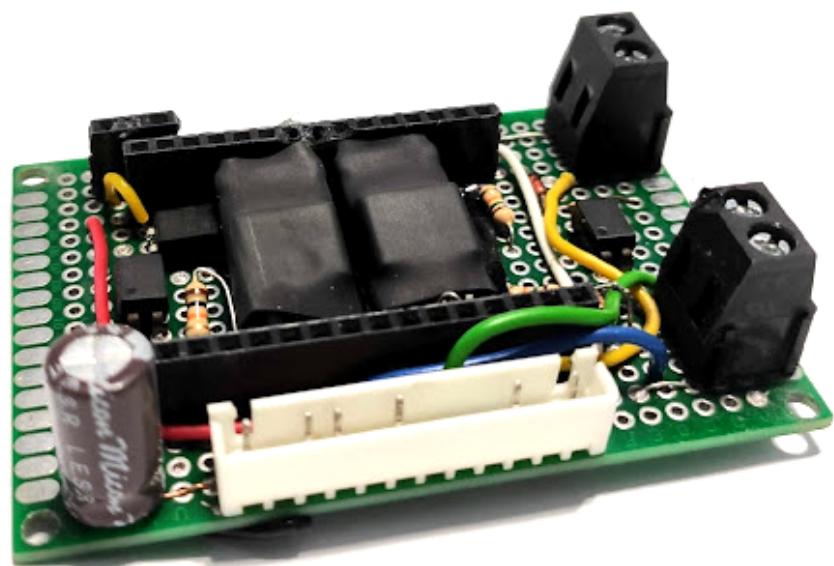


Figura B.2 – *Shield* desenvolvido neste trabalho

APÊNDICE C – CIRCUITO DE CARGA

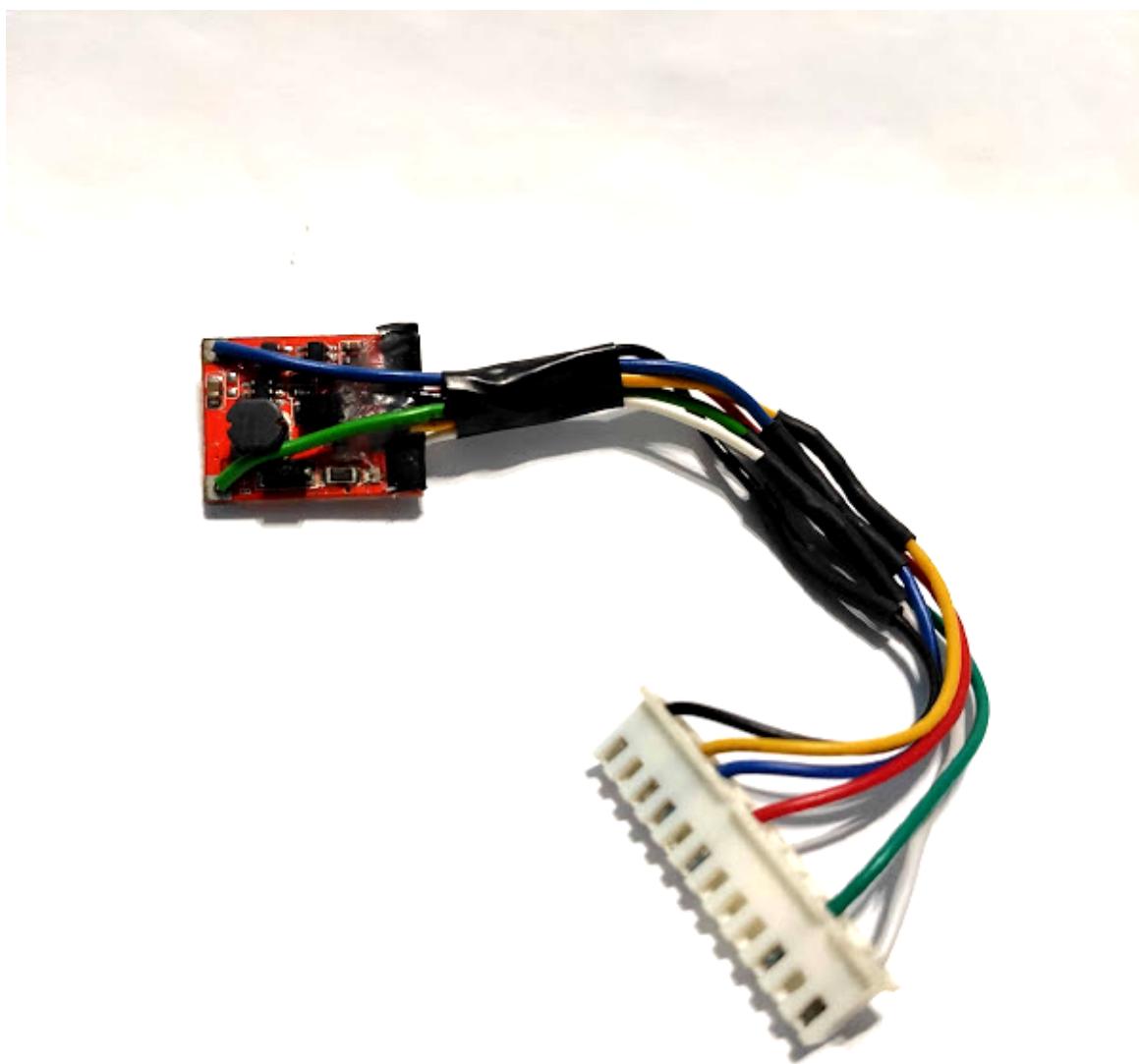


Figura C.1 – Circuito de carga onde foi aplicada engenharia reversa

APÊNDICE D – LIGAÇÃO COMPLETA

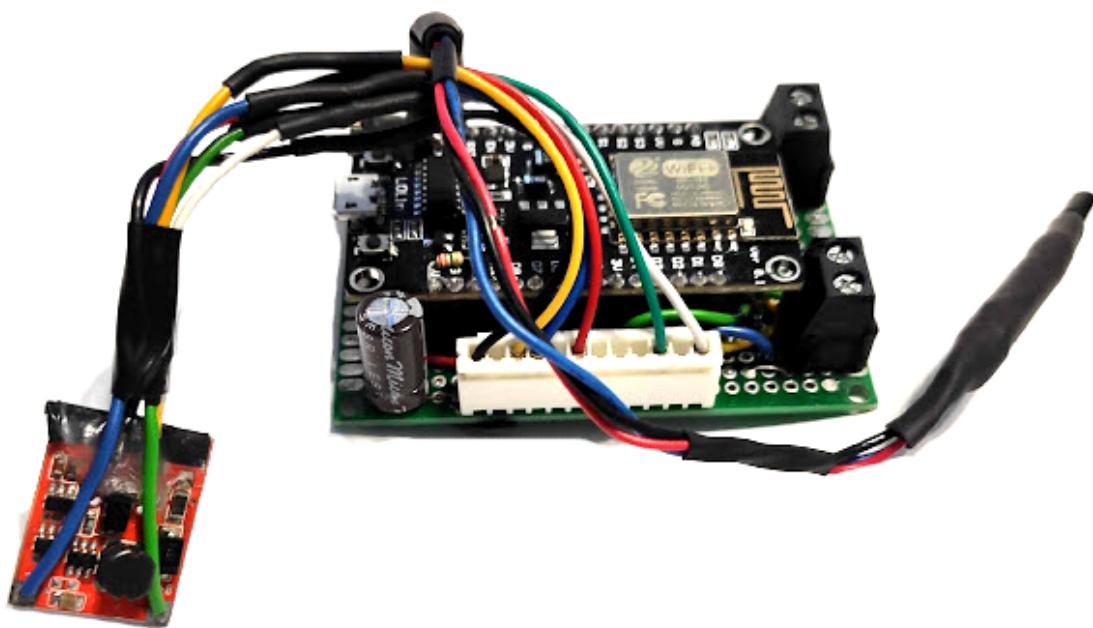


Figura D.1 – Ligação completa (sem os painéis)

APÊNDICE E – PROTÓTIPO FINAL



Figura E.1 – Caixa antes de selar

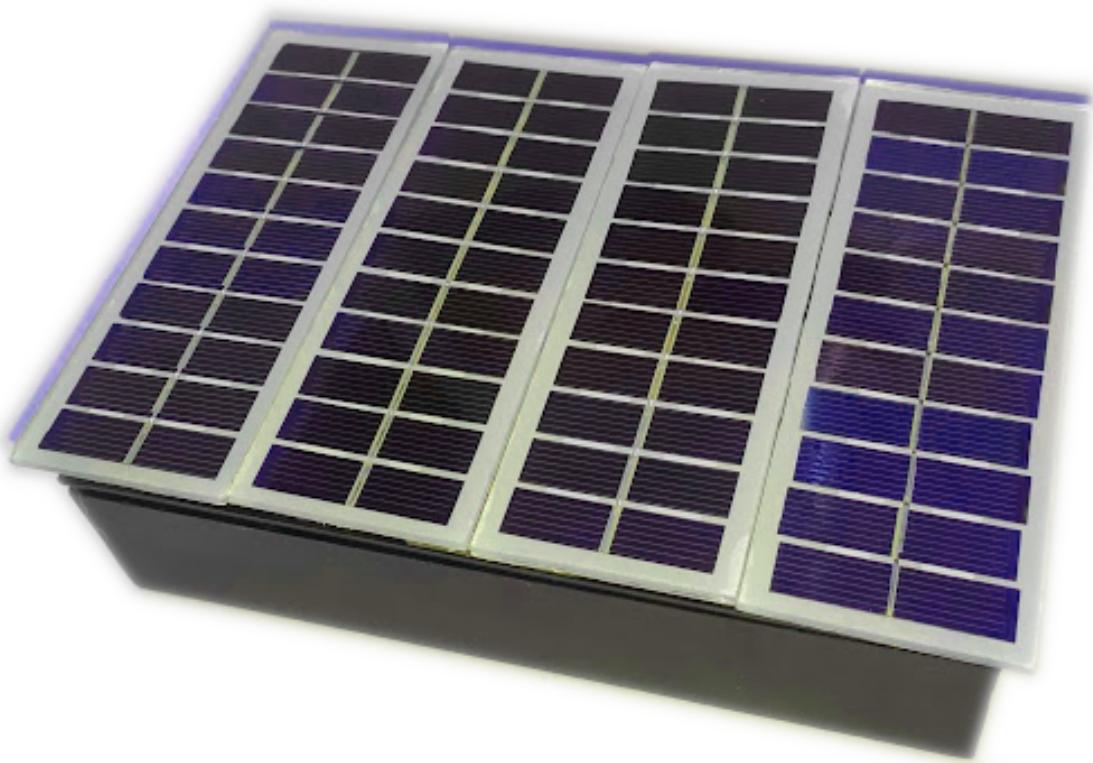


Figura E.2 – Caixa selada com os painéis