INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS ITUMBIARA
BACHARELADO EM ENGENHARIA ELÉTRICA

JENNYFER HART SANTANA MARTINS

DESENVOLVIMENTO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO MICROCONTROLADOR ARDUÍNO MEGA PRO E ESP8266

ITUMBIARA-GO
2024

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[X] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor: Jennyfer Hart Santana Martins
Matrícula:20152040070032
Título do Trabalho: Desenvolvimento de um sistema de automação residencial utilizando microcontrolador Arduíno Mega Pro e ESP8266

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Itumbiara-GO, 21/08/2024.
Local Data

Jennyfer Hart Santana Martins
_____
Assinatura do Autor e/ou Detentor dos Direitos Autorais

JENNYFER HART SANTANA MARTINS

DESENVOLVIMENTO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO MICROCONTROLADOR ARDUÍNO MEGA PRO E ESP8266

Trabalho de Conclusão de Curso apresentado à Banca examinadora do Curso de Bacharelado em Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Itumbiara, como parte dos requisitos necessários para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador(a): Dr. Josemar Alves dos Santos Junior.

ITUMBIARA-GO

2024

Dados Internacionais de Catalogação na Publicação (CIP)

M386d Martins,Jennyfer Hart Santana

Desenvolvimento de um sistema de automação residencial utilizando microcontrolador arduíno Mega Pro e ESP8266. / Jennyfer Hart Santana Martins. -- Itumbiara, 2024.
75p. : il.

Trabalho de conclusão de curso (Engenharia Elétrica) – Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Itumbiara, 2024.
Orientador: Prof. Dr. Josemar Alves dos Santos Junior.

1. Automação residencial. 2. Demótica. 3. Microcontroladores. I. Título. II. Lourenço, Giovani Aud. IV. Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

CDD 004.16

Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Biblioteca Maria Gabriela Pacheco Pardey / Câmpus Itumbiara
Bibliotecário: Rosiane Gonçalves de Lima CRB1/1684

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS ITUMBIARA

Termo de Aprovação

JENNYFER HART SANTANA MARTINS

DESENVOLVIMENTO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO MICROCONTROLADOR ARDUÍNO MEGA PRO E ESP8266

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Goiás – Câmpus Itumbiara, como parte dos requisitos para a obtenção do título de Bacharel em Engenharia Elétrica.

Trabalho de Conclusão de Curso defendido e aprovado, em 06 de agosto de 2024, pela banca examinadora constituída pelos seguintes membros:

BANCA EXAMINADORA

Prof. Dr. Josemar Alves dos Santos Junior - Orientador

Instituto Federal de Goiás, Câmpus Itumbiara.

(Assinado Eletronicamente).

Prof. Dr. Giovani Aud Lourenço - membro interno

Instituto Federal de Goiás, Câmpus Itumbiara.

(Assinado Eletronicamente).

Prof. Me. Wellington do Prado - membro interno

Instituto Federal de Goiás, Câmpus Itumbiara.

(Assinado Eletronicamente).

Itumbiara – GO

2024

Documento assinado eletronicamente por:

- **Wellington do Prado**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 09/08/2024 15:53:56.
- **Giovani Aud Lourenco**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 06/08/2024 18:00:45.
- **Josemar Alves dos Santos Junior**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 06/08/2024 16:39:40.

Este documento foi emitido pelo SUAP em 06/08/2024. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifg.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 548148
Código de Autenticação: cca2d39f02

Dedico este trabalho a minha querida avó Idalina Silva (in memorian), que não pôde estar aqui nesse momento, mas sei que estaria imensamente feliz com a realização desse sonho.

AGRADECIMENTOS

A Deus, agradeço por guiar meus passos e por me conceder a coragem e força necessárias para chegar até esse momento de conclusão do curso, foram várias noites mal dormidas e inúmeras dificuldades, mas Deus se fez presente em todos os momentos.

A minha família, em especial minha mãe Daniela Silva e meu pai Cleuber Vicente, por me apoiarem e se esforçarem para que pudesse ter todas as oportunidades possíveis. Ao meu noivo Vitor Hugo, agradeço por segurar minha mão a cada vitória ou derrota, por me manter firme e acreditar no meu potencial.

Ao meu orientador doutor Josemar Alves, agradeço por toda paciência e auxílio durante o desenvolvimento desse projeto. Aos meus professores, deixo uma palavra sincera de gratidão, todos os seus ensinamentos foram fundamentais para o meu desenvolvimento acadêmico.

“As pessoas não são lembradas pelo número de vezes que fracassam, mas sim pelo número de vezes que têm sucesso”

(Thomas Edison)

RESUMO

A busca incessante em ganhar tempo na execução de atividade rotineiras como irrigar o jardim, desligar ou ligar uma luz ou até mesmo abrir ou fechar um portão, fez com que a automação residencial ganhasse espaço no mercado, com o intuito de proporcionar uma maior qualidade de vida, conforto e segurança a seus utilizadores, além de uma economia de recursos energéticos. Entretanto, esse setor ainda enfrenta alguns problemas como a complexidade de desenvolver esse sistema e os altos custos para a sua implantação, tornando a sua utilização restrita a pessoas com maior poder aquisitivo. Sendo assim, este trabalho buscou desenvolver um sistema de automação eficiente, utilizando materiais com menor custo e uma linguagem de programação de fácil entendimento. Esse projeto será composto por um *hardware* responsável pelo monitoramento e acionamento dos dispositivos, um *software* responsável pelo gerenciamento e uma *interface web* responsável por validar as requisições e interagir com o *hardware*, e dessa forma o usuário poderá supervisionar e comandar o sistema de forma simples e ágil. Por fim, é apresentado a montagem do protótipo com a demonstração do sistema em funcionamento.

Palavras-chave: Automação residencial. Domótica. Arduino Pro Mini. ESP8266.

ABSTRACT

The incessant search to save time in carrying out routine activities such as irrigating the garden, turning off or turning on a light or even opening or closing a gate, has made home automation gain space in the market, with the aim of providing a higher quality of life, comfort and safety for its users, in addition to saving energy resources. However, this sector still faces some problems such as the complexity of developing this system and the high costs of implementing it, making its use restricted to people with greater purchasing power. Therefore, this work sought to develop an efficient automation system, using lower-cost materials and an easy-to-understand programming language. This project will consist of hardware responsible for monitoring and activating the devices, software responsible for management and a web interface responsible for validating requests and interacting with the hardware, and in this way the user will be able to supervise and command the system in a simple and agile. Finally, the prototype assembly is presented and the system is demonstrated in operation.

Keywords: Home automation. Domotics. Arduino Pro Mini. ESP8266.

LISTA DE FIGURAS

Figura 01 – Os benefícios da domótica 18
Figura 02 – Integração entre os equipamentos na domótica 19
Figura 03 - Taxa de penetração de casa inteligentes no Brasil 20
Figura 04 - Funcionamento lógico de um sistema embarcado 22
Figura 05 - Placa Mega 2560 Pro Mini 25
Figura 06 – ESP8266 modelo ESP-01 27
Figura 07 – *Pinout* do ESP-01 28
Figura 08 – Módulo relé 8 canais 30
Figura 09 – Optoacoplador PC817 32
Figura 10 - Módulo RF (TX e RX) 433MHz 33
Figura 11 – Módulo regulador de tensão LM2596 Conversor DC-DC 34
Figura 12 – Protocolo TCP/IP 37
Figura 13 – Arquitetura cliente-servidor 38
Figura 14 – Tela inicial do ISIS 41
Figura 15 – Tela inicial do ARES 42
Figura 16 – IDE Arduino 43
Figura 17 – Esquemático da placa optoacopladora no ISIS 47
Figura 18 – Projeto base da PCB da placa optoacopladora 48
Figura 19 – Placa optoacopladora 48
Figura 20 – Circuito esquemático da placa adaptadora para módulo relé 49
Figura 21 – Projeto base da placa adaptadora para módulo relé 50
Figura 22 – Placa adaptadora para módulo relé 50
Figura 23 – Fluxograma de funcionamento 51
Figura 24 – Circuito esquemático da placa principal 52
Figura 25 – Projeto base da placa principal 53
Figura 26 – Placa principal finalizada 54
Figura 27 – Bibliotecas utilizadas 56
Figura 28 – Estruturação do void setup 57
Figura 29 – Definições das URLs 58
Figura 30 – Lógica de programação do Arduino 59
Figura 31 – Protocolo de comunicação ESP+ARDUINO 59
Figura 32 – Exemplo de função para criação de página *web* 60
Figura 33 – *Hardware* em funcionamento 62
Figura 34 – *Hardware* em funcionamento 63
Figura 35 – Logo 63
Figura 36 – Nome da rede Wi-Fi instanciada pelo ESP 64
Figura 37 – IP padrão do ESP para acesso a página *web* 64
Figura 38 – Página inicial do Auto_Res 65
Figura 39 – Página de configuração do modo Wi-Fi *Station* do Auto_Res 66
Figura 40 – IP gerado após conexão com Wi-Fi local para acesso a página *web* 67
Figura 41 – Página de configuração de nomes do Auto_Res 67
Figura 42 – Página de acionamento de cargas elétricas 68
Figura 43 – Página de monitoramento de cargas elétricas 69
Figura 44 – Simulação de acionamento de oito cargas elétricas 70

Figura 45 – Simulação de monitoramento de oito carga elétrica 70

LISTA DE TABELAS

Tabela 01 – Especificações técnicas da placa Arduino Mega Pro Mini 25
Tabela 02 – Especificação técnica do ESP-01 29
Tabela 03 – Especificação técnica do módulo relé 8 canais 31
Tabela 04 – Especificação técnica dos módulos RX e TX 33
Tabela 05 – Especificação técnica módulo regulador de tensão LM2596 Conversor DC-DC – *Step Down* 35
Tabela 06 - Levantamento de custos 46
Tabela 07 – Mapeamento dos pinos do Arduino 54

LISTA DE ABREVIATURAS

A – Ampere

GND – *Ground* (Terra)

GPIO – *General Purpose Input Output* (Saída e entrada de uso geral)

IDE – *Integrated Development Environment* (Ambiente de desenvolvimento integrado)

IoT – *Internet of Things* (Internet das coisas)

MHz – Mega Hertz

RX – *Receive Serial* (Receber serial)

TX – *Transmissor Serial* (Transmissor serial)

UART – *Universal asynchronous receiver-transmitter* (Receptor-transmissor assíncrono universal)

V - Volts

Wi-Fi – *Wireless Fidelity* (Fidelidade sem fio)

SSL – *Secure Socket Layer* (Camada se soquete segura)

TLS – *Transport Layer Security* (Segurança da camada de transporte)

NVS - *Non-Volatile Storage* (Armazenamento não volátil)

SUMÁRIO

1 INTRODUÇÃO 14

1.1 OBJETIVO GERAL 14
1.2 OBJETIVO ESPECIFICOS 15
1.3 JUSTIFICATIVA 15
2 REFERENCIAL TEÓRICO 17

2.1 AUTOMAÇÃO 17
2.2 DOMÓTICA 17
2.3 POTENCIAL DA AUTOMAÇÃO RESIDENCIAL NO BRASIL 20
3 DISPOSITIVOS E TECNOLOGIAS 22

3.1 SISTEMAS EMBARCADOS 22
3.2 MICROCONTROLADORES 23
3.2.1 ARDUINO 24
3.2.2 ESP8266 26
3.3 MÓDULO RELÉ 29
3.4 OPTOACOPLADORES 31
3.5 MÓDULO RF 433MHZ 32
3.6 CONVERSOR DE TENSÃO DC-DC 34
3.7 TECNOLOGIA DE INFORMAÇÃO E COMUNICAÇÃO 35
3.7.1 WI-FI 36
3.8 PROTOCOLOS DE COMUNICAÇÃO 36
3.8.1 HTTP 37
3.9 AMBIENTE DE DESENVOLVIMENTO DO *HARDWARE* 40
3.9.1 PROTEUS 40
3.10 AMBIENTE DE DESENVOLVIMENTO DO *SOFTWARE* 43
3.10.1 IDE ARDUINO 43
4 METODOLOGIA 45

4.1 DESENVOLVIMENTO DO *HARDWARE* 45
4.1.1 PLACA OPTACOPLADORA 46
4.1.2 PLACA ADAPTADOR PARA MODULO RELÉ 49
4.1.3 PLACA PRINCIPAL 51
4.2 DESENVOLVIMENTO DO *FIRMWARE* 56
4.2.1 ESP-01 56
4.2.2 ARDUINO 58
4.2.3 DESENVOLVIMENTO DA PÁGINA *WEB* 60
5 INTEGRAÇÃO DO SISTEMA E FUNCIONAMENTO 62

6 CONCLUSÕES 71
6.1 TRABALHOS FUTUROS 72
REFERÊNCIAS 73
APÊNCICE A – INSTALAÇÃO E CONFIGURAÇÕES PARA A UTILIZAÇÃO DAS PLACAS ESP NA IDE DO ARDUINO 76
APÊNCICE B – PROGRAMAÇÃO ESP8266 79
APÊNCICE C – PROGRAMAÇÃO ARDUINO 110

1 INTRODUÇÃO

Em um cenário onde a tecnologia e a internet desempenham um papel central na vida cotidiana, visando facilitar as atividades diárias, a pandemia de COVID-19 acelerou significativamente tendências que já estavam em curso. A necessidade de adaptação rápida às novas condições impostas pela pandemia impulsionou o desenvolvimento e a adoção de tecnologias que estavam emergindo, transformando ainda mais a forma como vivemos e interagimos com o mundo digital. O isolamento social e a necessidade de trabalhar remotamente despertaram nos brasileiros um maior interesse para a automação residencial, de forma a tornam o ambiente doméstico mais eficiente e confortável.

Segundo Steven e Farinelli (2019), domótica que é o termo que caracteriza a automação residencial, resulta da junção da palavra romana domus, que se refere a casas, com a palavra robótica que; por sua vez, refere-se à realização de controle automatizado de algo por robôs. A junção das duas palavras resulta na definição do processo de automatização do ambiente residencial ou doméstico. Sendo assim, a domótica tem como objetivo principal o gerenciamento e execução das atividades diárias de seus usuários, a fim de proporcionar segurança, comodidade, conforto e até mesmo economia de energia.

A busca incessante em ganhar tempo na execução de atividades rotineiras como irrigar o jardim, desligar ou ligar uma luz ou até mesmo abrir ou fechar um portão, fez com que a automação residencial ganhasse espaço no mercado. Entretanto, esse setor ainda enfrenta alguns problemas como a complexidade de desenvolver esse sistema e os altos custos para a sua implantação, tornando a sua utilização restrita a pessoas com maior poder aquisitivo.

A vista disso, este trabalho tem como objetivo o desenvolvimento de uma automação residencial, através de uma placa controladora universal, com a utilização dos microcontroladores Arduino Mega Pro e ESP8266, de forma que esse sistema seja capaz de realizar o gerenciamento de cargas elétricas de uma residência.

1.1 OBJETIVO GERAL

O presente trabalho tem como principal objetivo desenvolver um sistema de automatização dos acionamentos elétricos existentes em uma residência, utilizando o Arduino ATMEGA 2560 Pro Mini e o microcontrolador ESP8266 modelo ESP-01 para o desenvolvimento do *hardware*. E

mediante a utilização dos conceitos da domótica, implementar um sistema capaz de executar ações de maneira automática através do controle de cargas elétricas que são responsáveis pelo acionamento de dispositivos como por exemplo a iluminação. Além disso, este trabalho busca a elaboração de uma *interface* simplificada que possibilite ao usuário o acionamento remoto por controle *web*.

1.2 OBJETIVO ESPECIFICOS

- Desenvolver os circuitos elétricos no *software* ISIS Proteus;
- Desenvolver a placa PCB no software ARES Proteus;
- Construir os protótipos das placas de circuito impresso responsáveis pelas atividades de acionamento e monitoramento;
- Implementar os *softwares* na IDE do Arduino;
- Elaborar *interface web* para controle e acionamento remoto;
- Realizar a integração do *software* e o *hardware* construído;
- Analisar e testar o funcionamento do sistema;

1.3 JUSTIFICATIVA

Como caracterizado por Almeida (2009), a automação residencial ainda possui um alto custo para o grupo ao qual se destina, entretanto com o crescimento dessa área, pode-se aumentar o número de pessoas interessadas nessa tecnologia. Segundo a AURESIDE, Associação Brasileira de Automação Residencial, de 2021 a 2025, o mercado de casas inteligentes deve crescer cerca de 178% e através de uma pesquisa realizada pela GSI Brasil (Associação Brasileira de Automação), revelou-se que 83% dos entrevistados possuem interesse em utilizar algum recurso de automação residencial e dentre os 17% que não apresentaram interesse, 27% revelam como empecilho os elevados custos para a implantação de um sistema de automação residencial.

Analisando os dados é possível observar que a automação está em constante crescimento, porém será necessário a implementação de sistemas com valores mais acessíveis aos usuários para que esse crescimento seja realmente alcançado.

Diante do exposto, este trabalho se justifica na possibilidade de realizar uma automação residencial que vise o bem-estar do usuário e que seja acessível financeiramente a pessoas com poder aquisitivo menor, com o objetivo de desmistificar a ideia de que a automação residencial é uma tecnologia de alto custo e pouco acessível.

2 REFERENCIAL TEÓRICO

2.1 AUTOMAÇÃO

Desde os primórdios, o homem busca maneiras de melhorar a sua qualidade de vida, preocupando-se em facilitar seu trabalho, inventou a eletricidade, o computador, a internet. E a busca constante pela facilidade fez surgir a automação.

A palavra automação, que do latim Automatus significa mover-se por si só, é compreendido como um sistema pelo qual os mecanismos necessitam de mínima ou nenhuma interferência humana para seu funcionamento. Em termos mais atuais, a automação pode ser conceituada como o conjunto de máquinas e equipamentos autônomos que podem ser aplicados sobre um processo, com o objetivo de torná-lo mais eficiente, de forma a melhorar o processo com menor consumo de energia e mínimo de intervenção humana. (PESSÔA; SPINOLA, 2014)

O conceito de automação varia de acordo com o ambiente, e os principais ramos da automação são: automação industrial que consiste em escolher as melhores tecnologias que se adaptam ao processo industrial e desenvolver uma maneira de as interligar para garantir a melhor relação custo/benefício. A automação comercial, utiliza as técnicas de automação para otimizar os processos comerciais. E a automação residencial, mais conhecida como domótica, que será abordado no item 2.2, por ser o ramo da automação escolhido para o desenvolvimento deste projeto (PINHEIRO, 2004).

2.2 DOMÓTICA

Segundo Bolzani (2010), a domótica é uma ciência multidisciplinar que estuda a relação entre homem e casa. A imersão de pessoas em ambientes computacionalmente ativos revelou a necessidade do uso de técnicas mais sutis que gerenciassem a complexidade e o dinamismo das interações dos moderadores com o ambiente residencial saturado de diminutos dispositivos eletrônicos interligados a rede.

Domótica é um processo ou sistema que prioriza a melhoria do estilo de vida (das pessoas), do conforto, da segurança e da economia da residência, através de um controle centralizado das funções desta, como água, luz, telefone e sistemas de segurança, entre outros (NUNES, 2002).

Os benefícios de se utilizar um sistema de automação residencial, conforme apresentado na Figura 1, podem ser notados imediatamente pelos moradores de uma residência automatizada. Economia de energia, conforto, comodidade, acessibilidade e segurança são alguns dos ganhos ligados a automação.

Figura 01 – Os benefícios da domótica

Fonte: Página da internet.[1].

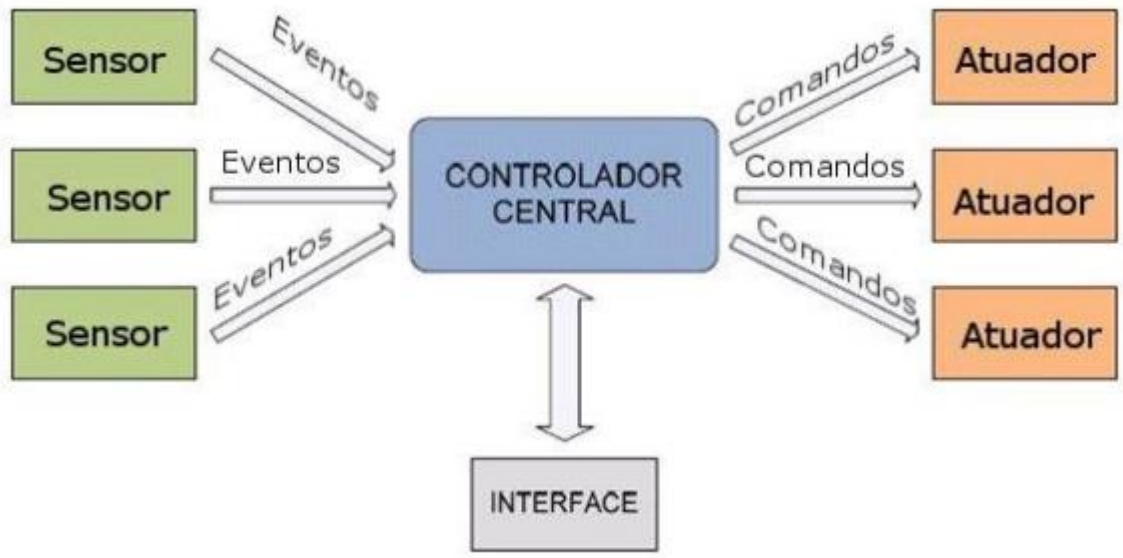
Para corresponder as exigências, a domótica faz uso de vários equipamentos distribuídos pela residência de acordo com as necessidades dos moradores. Estes equipamentos podem ser divididos em três principais grupos (TAKIUCHI, 2004):

- Atuadores: são responsáveis pelos acionamentos dos equipamentos elétricos e eletrônicos como, por exemplo, um relé acionando uma lâmpada,
- Sensores: são responsáveis por receber as informações do ambiente para que o controlador possa executar as rotinas e tarefas.
- Controladores: são responsáveis por coordenar os acionamentos depois de interpretar as informações captadas pelos sensores, ou seja, é o administrador de todo o sistema de automação.

1 Disponível em: < https://www.fiteck.com.br/automacao-residencial/>. Acesso em: 20 de junho de 2023.

A Figura 02 apresenta o esquemático de como funciona a integração entre os principais grupos de equipamentos utilizados na domótica.

Figura 02 – Integração entre os equipamentos na domótica



Fonte: FERREIRA (2008).

A grande guinada da domótica foi após o surgimento e aprimoramento de dispositivos como os microprocessadores, relés e sensores, pois todas as áreas em que a automação estava presente sofreram significativas mudanças quanto à qualidade dos equipamentos, principalmente a área da automação residencial, uma vez que os novos equipamentos não exigiam grandes espaços reservados, passaram a ser capazes de interagir com outros equipamentos e, talvez o mais importante, não precisavam de manutenção constante de técnicos (BOLZANI, 2004).

De uma forma geral a domótica tem como objetivo conseguir manipular o maior número possível de elementos e tarefas domésticas, fazendo com que as tarefas mais complexas sejam facilitadas e as que são simples, porém repetitivas gastem menos do tempo e da energia dos indivíduos (OLIVEIRA, 2017).

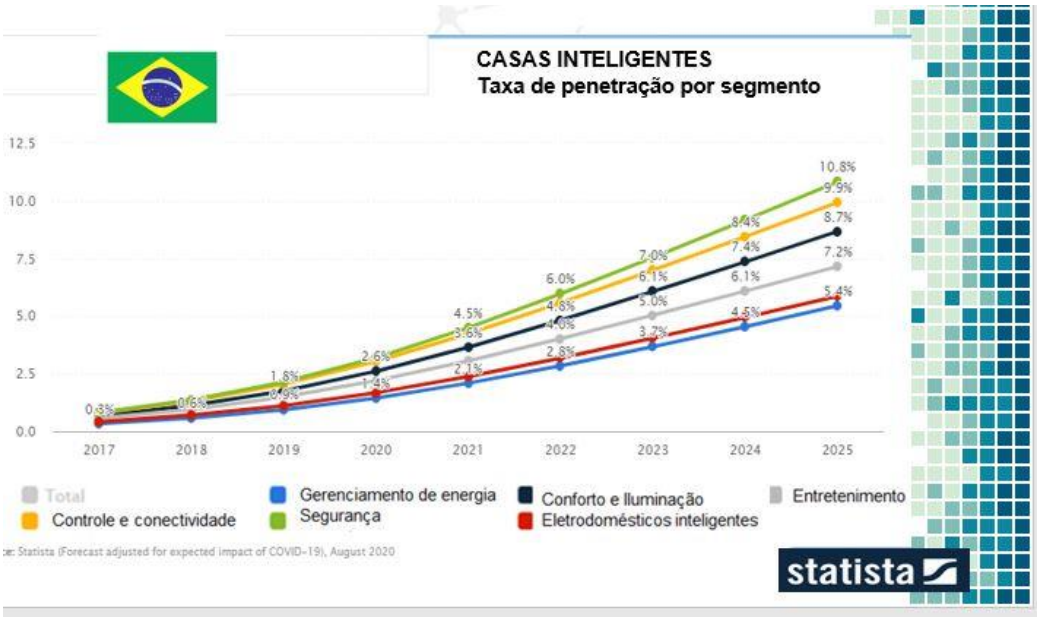
2.3 POTENCIAL DA AUTOMAÇÃO RESIDENCIAL NO BRASIL

Existe no mercado Brasileiro um potencial atual para o fornecimento de equipamentos para 1,8 milhões de residências e estima-se que cerca de 300 mil residências no Brasil já possuam equipamentos de automação residencial. Portanto, existe ainda um mercado inexplorado de pelo menos 1,5 milhões de residências. No Brasil, existe um interesse pela aquisição de uma automação residencial por parte de 78% dos consumidores, no mundo a média é de 66%. (AURESIDE).

Segundo um estudo realizado pelo *Business Insider*, o crescimento contínuo da indústria de IoT será uma força transformadora em todas as organizações. Ao integrar todos os nossos dispositivos modernos com conectividade com a Internet, o mercado IoT está a caminho de crescer para mais de U$ 2,4 trilhões anualmente até 2027.

Na Figura 03 é apresentada a taxa de penetração de casa inteligentes no Brasil, segundo pesquisa da Statista (2020), na qual é subdividida por segmentos, evidenciando um crescente aumento no mercado. Sendo os segmentos de segurança, controle e conectividade e conforto e iluminação os segmentos com maior taxa de penetração.

Figura 03 - Taxa de penetração de casa inteligentes no Brasil



Fonte: STATISTA (2020).

Segundo Statista (2020), estima-se que este setor continue a se expandir em ritmo acelerado até 2030, evidenciando assim o grande potencial de crescimento do ramo de automação residencial. O desenvolvimento contínuo de dispositivos conectados, como assistentes virtuais, e a integração de inteligência artificial nas residências também são fatores-chave que impulsionam essa tendência.

3 DISPOSITIVOS E TECNOLOGIAS

3.1 SISTEMAS EMBARCADOS

Um sistema é classificado como embarcado quando este é dedicado a uma única tarefa e interage continuamente com o ambiente a sua volta por meio de sensores e atuadores (BALL, 2005). Por exigir uma interação contínua com o ambiente, este tipo de sistema requer do projetista um conhecimento em programação e construção de *hardwares*.

A denominação “sistemas embarcados” (do inglês *Embedded Systems*) são sistemas computacionais de uso específico, com seus recursos computacionais como memória e poder de processamento projetos restritamente para este propósito especial. Sendo assim, um sistema embarcado realiza um conjunto de tarefas predefinidas e não podem ter sua funcionalidade alterada durante o uso e caso haja necessidade de alteração, é necessário reprogramar todo o sistema. A Figura 04 apresenta a lógica de um sistema embarcado (CHASE, 2007).

Figura 04 - Funcionamento lógico de um sistema embarcado

Fonte: CHASE, (2007).

Todo o sistema embarcado é composto por uma unidade de processamento, que é um circuito integrado, fixado a uma placa de circuito impresso. Possuem uma capacidade de processamento de informações vinda de um *software* que está sendo processado internamente nessa unidade. Todo *software* embarcado é classificado de *firmware* (BALL, 2005).

O mercado de sistemas embarcados é atrativo devido ao fato de que quase todos os equipamentos conectados a eletricidade possuem algum sistema computacional embutido. Entretanto, alguns aspectos como custo, desempenho e o tempo de desenvolvimento torna esta área desafiadora, principalmente quando envolve inteligência artificial, tomada de decisões e processamento de sinais (BARROS; CAVALCANTE, 2010).

3.2 MICROCONTROLADORES

Microcontroladores são circuitos integrados que possuem todos os componentes necessários para o seu funcionamento dependendo apenas da fonte de alimentação externa. Pode-se dizer que os microcontroladores são computadores de um único chip o qual contém um núcleo de processador, memória e periféricos programáveis de entrada e saída (KERSCHBAUMER, 2018).

Segundo Kerschbaumer (2018), a versatilidade dos microcontroladores que são compactos e possuem uma ótima relação custo/benefício, faz com que esse dispositivo possa ser utilizado em uma infinidade de aplicações. O seu desempenho depende principalmente do *firmware* que nele é gravado, logo basta apenas fazer uma alteração neste, para que as tarefas executadas pelo microcontrolador se alterem.

Na automação residencial, os microcontroladores são um dos principais dispositivos utilizados para o desenvolvimento desses sistemas. Eles são responsáveis por capturar os dados e informações dos sensores e realizar o acionamento dos atuadores, de forma a controlar e gerenciar todo o sistema.

Os diversos tipos de microcontroladores que existem se diferem em suas propriedades físicas, tais como: quantidade de memória, tensão de alimentação, velocidade de processamento, quantidade de portas analógicas/ digitais. Para o desenvolvimento desse projeto, será utilizada a placa Arduino Mega 2560 Pro Mini, que possui o chip ATmega 2560, como expansor de portas para possibilitar uma maior gama de periféricos controlados e o ESP-01 para colocar o Arduino na internet através da conexão *Wi-Fi* e possibilitar a criação de uma *interface web*, itens que serão explicados nos tópicos 3.2.1 e 3.2.2.

3.2.1 ARDUINO

Segundo McRoberts (2011), a definição de Arduino refere-se a um microcontrolador de placa única e um conjunto de *software* para programá-lo, sendo capaz de se comunicar com componentes externos e controlar entradas e saída analógicas/ digitais. O Arduino é o que chamamos de plataforma embarcada, pois é capaz de interagir com o ambiente através de *hardware* e *software*.

> O Arduino pode ser utilizado para desenvolver objetos interativos independentes, ou pode ser conectado a um computador, a uma rede, ou até mesmo à internet para recuperar e enviar dados do Arduino e atuar sobre eles. (MCROBERTS, 2022, P.23)

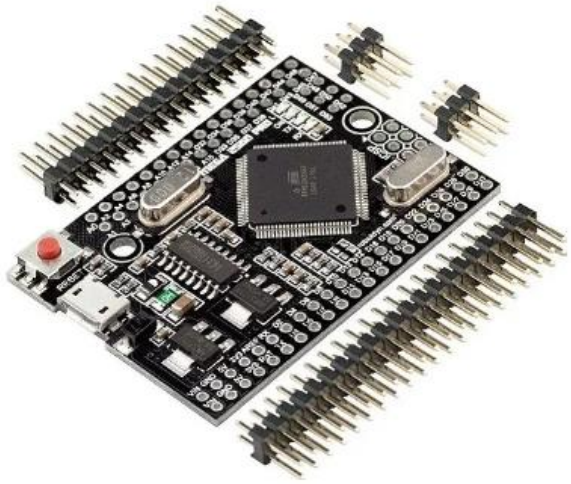
Dentre as diversas aplicações, o Arduino pode ser conectado a botões, interruptores, motores, sensores, módulos *Ethernet* ou qualquer outro dispositivo que emita dados ou possa ser controlado. E devido a sua versatilidade e por possuir uma plataforma *open-source* é um dispositivo muito utilizado por instituições de ensino a fim de adquirir conhecimento na área de programação, eletrônica e robótica.

Há diversos modelos de Arduino no mercado, sendo cada um específico para cada necessidade do usuário. Tem modelo que é do tamanho de um polegar, modelos circulares, tudo para que as placas sejam adequadas a diversas situações diferentes. Para esse projeto será utilizada uma placa que é uma variação do Arduino Mega, a placa Mega 2560 Pro Mini, que possui todas as características do Mega, porém em uma versão mais reduzida perfeito para projetos compactos.

3.2.1.1 ARDUINO MEGA 2560 PRO MINI

A placa é baseada no microcontrolador ATmega2560 e tem um *chip* de *interface* USB-UART CH340G. É uma versão reduzida da placa Arduino Mega 2560 R3, com a mesma capacidade e desempenho. Apesar do tamanho compacto adaptado para os sistemas embarcados, ela é compatível com a placa original no que diz respeito a programação, memória, processamento, número de pinos, entre outros. Na Figura 05 é apresentado o microcontrolador Arduino Mega 2560 Pro Mini (MAKERHERO).

Figura 05 - Placa Mega 2560 Pro Mini



Fonte: Página da internet.[2]

Possui 70 portas digitais I/O, das quais 15 podem ser usadas como PWM e 16 como entradas analógicas, 4 portas UARTs (portas seriais de *hardware*), um cristal oscilador de 16 MHz, uma conexão micro USB, uma entrada de alimentação, uma conexão ICSP e um botão reset, conforme especificações técnicas do Maker Hero (2023). Na Tabela 01 é apresentada as especificações técnicas da placa Arduino Mega Pro Mini (MAKERHERO).

Tabela 01 – Especificações técnicas da placa Arduino Mega Pro Mini

Microcontrolador	ATmega2560
Velocidade de clock	16 MHz
Portas I/O Digitais	70 (15 poder ser usados como PWM)
Portas analógicas	16
Tensão de Operação	5V
Tensão de Entrada	6 - 9V (pico: 12V)
Corrente máxima nos pinos I/O	40 mA
Memória Flash	256 KB (8 KB usado no bootloader)
SRAM	8 KB
EEPROM	4 KB

2 Disponível em: < https://www.eletrogate.com/placa-mega-2560-pro-mini >. Acesso em: 23 de março de 2023.

USB	Micro USB
Dimensões	54 mm x 38 mm x 6 mm

Fonte: Página da internet.[3]

A alimentação da placa poder ser por conexão USB ou por meio de uma fonte externa, sendo a entrada de alimentação selecionada automaticamente. Para alimentação com conexão micro USB, o dispositivo receberá 5V e a faixa de alimentação nominal é entre 6 – 9V, o que pode ocasionar instabilidade na placa e se aplicado mais de 12V, que é a tensão de pico, pode ocasionar superaquecimento no regulador e danificar o dispositivo. Já a alimentação com fonte externa pode ser realizada por uma fonte ou baterias através de conexão de cabos nos pinos terra (GND) e tensão de entrada (Vin), devendo obedecer ao range de 6V a 9V para o perfeito funcionamento do dispositivo (MAKERHERO).

As portas digitas operam com valores de tensão de 0V a 5V, os componentes acoplados à estas portas poderão receber ou enviar dados e informações por meio de pulsos nestes dois valores, caracterizados como valores binários 0 e 1 (TERRA, 2015).

Em contrapartida, as portas analógicas, que se referem a portas de leitura, operam com valores dentro do espectro de 0V a 5V. As 16 portas analógicas que a placa Arduino Mega Pro Mini possui são numeradas de A0 a A15 com precisão de 10 bits. Sendo assim, o range de leitura das portas analógicas é de 0 a 1023 (MAKERHERO).

A comunicação Serial (UART) é o que possibilita a comunicação entre a placa e o computador, ou outro dispositivo. A placa Mega Pro Mini possui quatro canais de comunicação: Serial: 0 (RX), 1 (TX); Serial 1: 19 (RX), 18 (TX); Serial 2: 17(RX), 16(TX); Serial 3: 15(RX), 14(TX). As portas RX são utilizadas para receber informações e TX para transmitir dados seriais TTL (TERRA, 2015).

3.2.2 ESP8266

O ESP8266 é um microcontrolador desenvolvido pela Espressif Systems, empresa com sede em Xangai, que está no mercado desde 2014 com o lançamento do ESP-01. É um dispositivo

3 Disponível em: < https://www.eletrogate.com/placa-mega-2560-pro-mini >. Acesso em: 23 de março de 2023.

de baixo custo com capacidade de realizar a comunicação via Wi-Fi e criar conexões TCP/IP, tornando esse dispositivo perfeito para projetos IoT (ESPRESSIF).

Esse módulo suporta as redes 802.11 b/g/n, muito usadas atualmente, podendo trabalhar como um Ponto de Acesso (*Access Point*) ou como uma Estação (*Station*), enviando e recebendo dados. No modo *Access Point* (AP), o dispositivo permite a criação de sua própria rede e tenha outros dispositivos conectados a ele como, por exemplo, *smartphone* e *notebook*. Já no modo *Station* (STA) o ESP8266 se conecta a uma rede Wi-Fi, rede criada pelo roteador *wireless* local. Dessa forma, um aspecto importante do ESP8266 é que ele pode operar como um "cliente" ou como um ponto de acesso, ou até mesmo em ambas as configurações (JUNIOR, 2019).

Segundo Oliveira (2017), existem diferentes modelos do ESP8266, como o ESP-01, ESP-12 e ESP Olimex, por exemplo. Todos esses modelos possuem o mesmo processador, variando apenas a quantidade de pinos de entrada e saída (GPIO), memória e disposição dos pinos. Para o desenvolvimento do projeto, foi escolhido o ESP-01, por atender a necessidade de conectar o Arduino a rede Wi-Fi, possuir um ótimo custo/ benefício e ser um módulo compacto em comparação aos outros, facilitando a montagem do *hardware*.

O ESP-01, evidenciado na Figura 06, é o módulo mais simples e compacto da linha ESP8266, ele não é compatível com protoboard e Plataforma Arduino, sendo assim para a sua utilização é necessário realizar alguns ajustes na plataforma e utilização de adaptadores para a gravação do *firmware*.

Figura 06 – ESP8266 modelo ESP-01

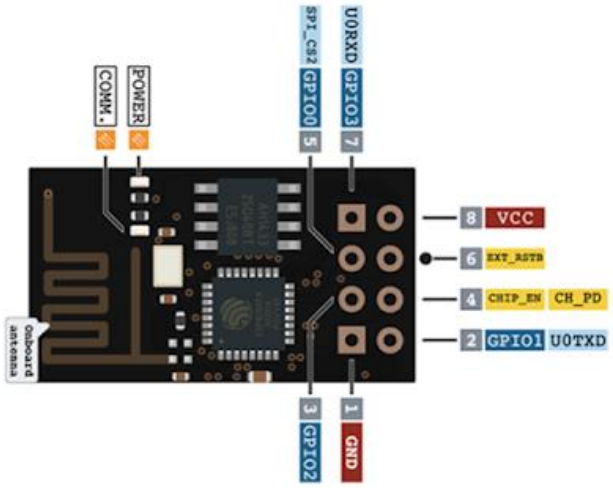
Fonte: Página da internet[4].

4 Disponível em: < https://www.makerhero.com/produto/modulo-wifi-esp8266-esp-01/>. Acesso em: 27 de março de 2023.

O módulo, conforme apresentado pela Figura 07, possui oito pinos, que para melhor compreensão, serão detalhados abaixo:

- O pino GPIO 0, pino de entrada e saída, podendo ser utilizado como PWM. Para realizar o carregamento ou a atualização do *firmware* é necessário que esse pino esteja em nível baixo;
- O pino GPIO 2, pino de entrada e saída, podendo ser utilizado com PWM;
- O pino VCC, que é o pino de alimentação do módulo, opera em 3,3V;
- O pino GND, que é o pino terra;
- Os pinos de recepção serial (RX) e transmissão serial (TX), destinados a comunicação UART
- O pino CH_PD, que habilita ou desabilita o módulo, responsável pelo funcionamento do módulo;
- E o pino RST, que é o pino de *reset*, que para resetar o sistema é necessário estar em nível lógico alto.

Figura 07 – *Pinout* do ESP-01



Fonte: Página da internet.[5]

O módulo Wi-Fi ESP8266 modelo ESP-01 foi desenvolvido para permitir a conexão Wi-Fi do seu microcontrolador de forma fácil, eficaz e a um baixo preço. Além disso, devido a possuir

5 Disponível em: < https://www.fernandok.com/2018/06/automacao-de-baixo-custo-com-esp01.html>. Acesso em: 27 de março de 2023.

um tamanho bem reduzido, facilita a sua utilização para projetos compactos. Em seguida, na Tabela 02 é apresentada as especificações técnicas do módulo Wi-Fi ESP8266 modelo ESP-01.

Tabela 02 – Especificação técnica do ESP-01

Chip	ESP8266
Modelo	ESP-01
Tensão de operação	3,3V
Corrente de operação	80mA
Suporte a redes	802.11 b/g/n
SPI Flash	8Mbits
RAM	Aproximadamente 50KB
Comunicação	UART (TX/ RX)
Suporta comunicação	TCP e UDP
Modos de operação	Station/SoftAP/SoftAP+Station
Modo de segurança	OPEN/ WEP/WPA_PSK/WPA2_PSK/WPA_WPA2_PSK
Dimensões	25 mm x 14 mm x 1 mm

Fonte: Página da internet.[6]

Neste projeto, o módulo ESP8266, modelo ESP-01, será responsável por conectar o Arduino Mega Pro Mini a internet e a partir de uma página HTML, *interface web* executada pelo ESP-01, transmitir os dados para o Arduino via comunicação serial para realizar o monitoramento e os acionamentos desejados.

3.3 MÓDULO RELÉ

O relé é um componente elétrico que através da aplicação de uma corrente gera uma indução magnética realizando a movimentação de um contato, servindo na maior parte das vezes como um interruptor a distância, podendo realizar a integração de circuitos de alta potência com circuitos eletrônicos de corrente e tensão inúmeras vezes menores (MARIMOTO, 2016).

6 Disponível em: < https://www.makerhero.com/produto/modulo-wifi-esp8266-esp-01/>. Acesso em: 27 de março de 2023.

Para a realização dos acionamentos das cargas elétrica, optou-se pela utilização do dispositivo eletromecânico, módulo relé de 8 canais demonstrado na Figura 08. Ele é capaz de acionar as cargas em corrente alternada e proporcionar o isolamento entre o sistema de corrente contínua.

Figura 08 – Módulo relé 8 canais

Fonte: Página da internet.[7]

O seu funcionamento se dá através do princípio eletromagnético, onde a bobina interna é alimentada por uma corrente criando um campo magnético que atrai os contatos, que podem ser normalmente abertos (NA) ou normalmente fechados (NF). Ou seja, quando a bobina é energizada os contatos elétricos são comutados e o contato que era NF passa a ser aberto e o que era NA passa a ser fechado.

O módulo relé escolhido permite o acionamento de cargas em 220V AC, como lâmpadas e outros dispositivos a partir do controle e comando do Arduino, sem a necessidade de montar circuito com transistores, conectores, LEDs e diodos, e assim, simplificando o sistema. Segue na Tabela 03 as especificações técnicas do módulo utilizado.

7 Disponível em: < https://www.makerhero.com/produto/modulo-rele-5v-8-canais/>. Acesso em: 27 de março de 2023.

Tabela 03 – Especificação técnica do módulo relé 8 canais

Modelo	JQC-3FF-S-Z
Tensão de operação	5V
Corrente de operação	15 a 20mA
Permite controlar cargas	220V
Tempo de resposta	5 a 10ms
Tipo de acionamento	Ativo baixo (Aciona com GND)
Dimensões	135 mm x 52 mm x 20 mm

Fonte: Página da internet.[8]

3.4 OPTOACOPLADORES

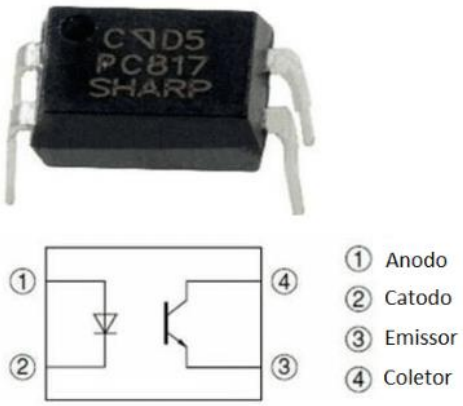
O optoacoplador ou acoplador óptico é um componente eletrônico muito utilizado nos projetos de circuitos digitais. Ele funciona como um botão que pode ser acionado eletricamente, assim como um relé. Porém, diferentemente dos relés, os circuitos são completamente isolados. (COELHO, 2021)

Esse componente é um circuito integrado que apresenta em sua estrutura um diodo emissor de luz e um fototransistor. Quando o diodo é acionado permite que o fototransistor conduza corrente, devido a sua base que é sensibilizada por luz. Esse dispositivo pode isolar tensões de entrada e saída na ordem de milhares de Volts, e devido a isso, são muito utilizados quando envolve circuitos digitais que interagem com circuito de corrente contínua (COELHO, 2021).

A Figura 09, refere-se ao optoacoplador PC817 utilizado nesse projeto e o seu esquema elétrico. Sendo que, para o funcionamento basta conectar a saída do microcontrolador ao anodo através de um resistor limitador de corrente e o catodo ao GND e o circuito que se deseja acionar no emissor e coletor do optacoplador.

8 Disponível em: < https://www.makerhero.com/produto/modulo-rele-5v-8-canais/>. Acesso em: 27 de março de 2023.

Figura 09 – Optoacoplador PC817



Fonte: Página da internet.[9]

3.5 MÓDULO RF 433MHZ

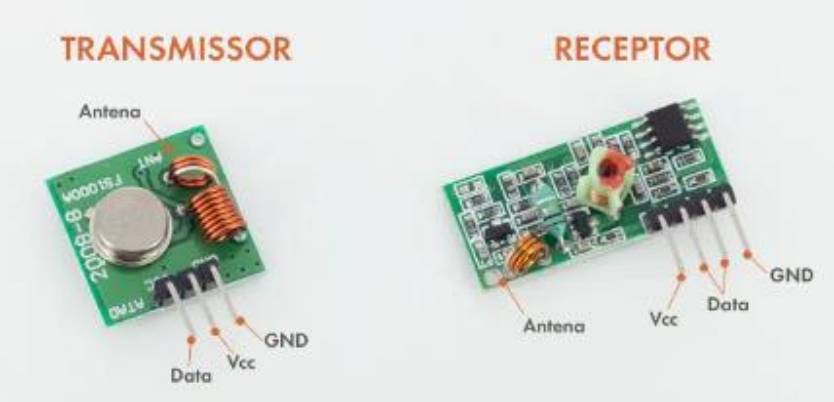
O módulo RF consiste em um transmissor e um receptor de ondas eletromagnéticas de frequência de 433Mhz. Isto é, são dois circuitos capazes de comunicar entre si sem a necessidade de fios, pois se comunicam utilizando ondas eletromagnéticas (GUIMARÃES, 2019).

O circuito do transmissor é composto por uma chave ligada a um gerador de onda de 433MHz, ao acionar a chave o sinal sai na antena do módulo e percorrer o caminho até chegar o módulo receptor. Já o circuito do módulo receptor possui uma antena para captar as ondas eletromagnéticas de 433MHz e um circuito para filtrar os ruídos e amplificar o sinal recebido. Além disso, ele possui uma malha de captura de fase para sincronizar o sinal de saída (GUIMARÃES, 2019).

A Figura 10 é apresentado os módulos RX e TX e seus respectivos pinos, sendo que para o módulo transmissor o pino Data é o que envia os dados para o receptor, o pino VCC é o pino de alimentação do módulo que suporta uma tensão entre 3 e 12V, sendo que, quanto maior a tensão maior a distância de transmissão e o pino GND que é o pino terra. Já o módulo receptor é constituído do pino VCC que deve ser alimentado em 5V, os pinos Data que possuem a função de receber os dados transmitidos e o pino GND que é o pino terra. A Tabela 04 apresenta as especificações técnicas desses componentes (GUIMARÃES, 2019).

9 Disponível em: <https://www.makerhero.com/blog/o-que-e-um-optoacoplador/>. Acesso em: 27 de março de 2023.

Figura 10 - Módulo RF (TX e RX) 433MHz



Fonte: Página da internet.[10]

A Tabela 04 apresenta as especificações técnicas desses componentes.

Tabela 04 – Especificação técnica dos módulos RX e TX

Módulo RF 433MHz	Transmissor	Receptor
Tensão de operação	3V ~ 12V	5V
Distância de alcance	20-200M	-
Taxa de transferência	4KB/s	-
Corrente de operação	max 40mA; min 9mA	4mA
Frequência de trabalho	433.92MHz	315MHz – 433.92MHz
Erro de frequência	150kHz	Ativo baixo (Aciona com GND)
Velocidade	≤ 10Kbps	≤ 9.6Kbps
Dimensões	19 mm x 19 mm	30 mm x 14 mm x 7 mm

Fonte: Página da internet.[11]

10 Disponível em: < https://www.makerhero.com/blog/modulo-rf-transmissor-receptor-433mhz-arduino/>. Acesso em: 27 de março de 2023.
11 Disponível em: < https://curtocircuito.com.br/modulo-rf-433mhz-trans-recep.html/>. Acesso em: 27 de março de 2023.

3.6 CONVERSOR DE TENSÃO DC-DC

O conversor DC-DC é um circuito eletrônico capaz de converte uma tensão ou corrente contínua que possui uma determinada amplitude, em outra tensão ou corrente com amplitude diferente. Sendo assim, eles são capazes de aumentar, manter ou diminuir o valor de tensão CC de sua saída. É um componente utilizado quando a tensão utilizada para a alimentação é diferente da necessária para o funcionamento de placas, circuito ou componentes (GUSE, 2022).

> Pode-se classificar os conversores em três categorias, elevador de tensão, abaixador de tensão e elevador-abaixador de tensão, além de poder classificá-los como isolados e não isolados. (PAZ, 2018, P.10)

Os conversores com topologia *Step-Down* ou *Buck*, realizam a redução dos níveis de tensão. Já os *Step-Up* ou *Boost* são destinados a aumentar a tensão de saída em relação a entrada e os *Buck-Boost* pode realizar as duas funções, tanto elevar como reduzir a tensão de saída.

Neste projeto foi utilizado o módulo apresentado na Figura 11 para realizar a alimentação dos componentes, exceto a alimentação do Arduino e ESP-01 (GUSE, 2022).

Figura 11 – Módulo regulador de tensão LM2596 Conversor DC-DC *Down*

Fonte: Página da internet.[12]

12 Disponível em: <https://www.makerhero.com/produto/regulador-de-tensao-lm2596-conversor-dc-dc-step-down/>. Acesso em: 27 de março de 2023.

Tabela 05 – Especificação técnica módulo regulador de tensão LM2596 Conversor DC-DC – *Step Down*

Módulo RF 433MHz	Receptor
Tensão de entrada	3,2V – 40V
Tensão de Saída	1,5 - 35V
Corrente de saída	2A - 3A
Eficiência de conversão	92%
Temperatura de operação	-40°C a 85°C
Dimensões	46 mm x 22 mm

Fonte: Página da internet.[13]

3.7 TECNOLOGIA DE INFORMAÇÃO E COMUNICAÇÃO

Tecnologia da informação e comunicação (TIC) pode ser definida como um conjunto de recursos tecnológicos, utilizados de forma integrada, com um objetivo comum. A TIC é utilizada das mais diversas formas, na indústria (no processo de automação), no comércio (no gerenciamento, nas diversas formas de publicidade), no setor de investimentos (informação simultânea, comunicação imediata) e na educação (PACIEVITCH, 2014).

De maneira simplificada, a TCI é qualquer forma de transmissão de informação e caracterizam a todas as tecnologias que interferem nos processos informacionais e comunicativos.

Através da internet, novos sistemas de comunicação e informação foram criados, formando uma verdadeira rede. Criações como e-mail, o chat, os fóruns, a agenda de grupo online, comunidades virtuais, web cam, entre outros revolucionaram os relacionamentos humanos (PACIEVITCH, 2014).

Além dessas ferramentas citadas acima, podemos considerar o Bluetooth e o Wi-Fi como exemplos de TCI.

13 Disponível em: < https://www.makerhero.com/produto/regulador-de-tensao-lm2596-conversor-dc-dc-step-down/>. Acesso em: 27 de março de 2023.

3.7.1 WI-FI

A abreviação Wi-Fi (*Wireless Fidelity*), que em português significa fidelidade sem fio, é uma das tecnologias de comunicação mais utilizadas atualmente e não utiliza cabos, sendo transmitida através de frequência de rádio ou infravermelho (ALLIANCE, 2018).

O termo Wi-Fi é usado frequentemente como sinônimo para padrão IEEE 802.11. Este padrão especifica as camadas físicas e de enlace para a implementação de uma rede local sem fio (WLAN) que opera na faixa de frequências de 2,4GHz e 5GHz. Além disso, este padrão de comunicação sem fio é flexível, suporta diversos dispositivos conectados, possui segurança, controle de acesso e está presente na maioria das residências. Sendo assim, o torna viável para a finalidade de domótica, gerando economia dos recursos estruturais e físicos. (SANTOS, 2019)

3.8 PROTOCOLOS DE COMUNICAÇÃO

Os protocolos de comunicação são um conjunto de regras e procedimentos objetivos que controla a troca de dados entre máquina e sistema de automação, garantindo que as trocas de dados sejam eficientes e sem perdas. Sendo assim, os protocolos são um tipo de ordem ou regras que caracteriza a sintaxe, semântica e sincronização da comunicação. Para que seja possível realizar a comunicação entre dois dispositivos em uma rede, eles devem “falar a mesma língua”, logo eles devem utilizar o mesmo protocolo de comunicação (ALMEIDA, 2013).

Com o intuito de criar uma padronização para essa comunicação, a ISO (*Internatonal Organization for Standardization)* criou o modelo OSI (*Open System Interconnection*) que tinha como principal objetivo tornar-se um modelo padrão para os diferentes tipos de sistemas. O Modelo OSI permite a comunicação entre máquinas heterogêneas e define diretivas genéricas para a construção de redes de computadores independente da tecnologia utilizada (ALMEIDA, 2013).

Com o objetivo semelhante ao do Modelo OSI, no que diz respeito à divisão da arquitetura em camadas, o TCP/IP é a linguagem de comunicação e conexão mais utilizada entre computadores e seu nome se dá pela junção de dois protocolos, TCP (*Transmission Control Protocol*) e o IP (*Internet Protocol*). Tem por objetivo padronizar todas as comunicações de rede, principalmente na *web* (ALMEIDA, 2013).

Esse protocolo possui apenas 4 camadas que englobam as 7 camadas do modelo OSI. As camadas superiores recebem informações e as distribui para as camadas inferiores, atribuindo a cada uma delas a função que exercerá durante a comunicação. A Figura 12 detalha como essas camadas se relacionam e as suas funções.

Figura 12 – Protocolo TCP/IP

Fonte: Página da internet[14].

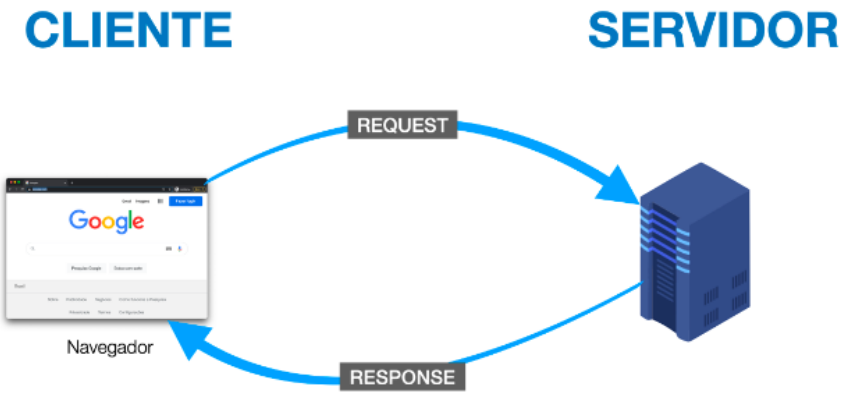
3.8.1 HTTP

O protocolo HTTP (*Hyper Text Transfer Protocol*) é o protocolo usado na *World Wide Web* para a destruição e recuperação de informações. A troca de informações entre um *browser* e um servidor *web* é toda feita através desse protocolo, que foi criado especificamente para a *World Wide Web*. Fazendo parte da camada de aplicação, o protocolo HTTP costuma atuar em conjunto com o protocolo TCP/IP, que faz parte da camada de transporte.

14 Disponível em: < https://www.devmedia.com.br/via-de-mao-dupla-com-websockets/28281>. Acesso em: 28 de março de 2023.

Este protocolo se baseia no padrão cliente-servidor em que se comunicam utilizando o método *request-response*. Toda a conversação se dá no formato ASCII (texto puro) através de um conjunto de comandos simples baseados em palavras da língua inglesa. Na Figura 13 é apresentado a arquitetura cliente-servidor (ALMEIDA, 2013).

Figura 13 – Arquitetura cliente-servidor



Fonte: Página da internet.[15]

Os clientes de uma conexão HTTP são os *browsers* como, por exemplo, Google Chrome e Internet Explorer. Já os servidores são os servidores da *web* e temos como exemplo o Apache HTTP Server, o *Internet Information Server* e o *Enterprise Server*.

Um pedido HTTP é composto de quatro partes básicas, o método, a URL (Localizador Uniforme de Recursos) a versão do protocolo HTTP e as informações adicionais. O método é a ação a ser realizada é pode ser classificado como:

- GET: é utilizado para recuperar dados de um servidor recorrendo a um URI em particular. Ou seja, retorna à informação requisitada.
- POST: é utilizado para enviar dados para o servidor *web*.
- PUT: é utilizado para modificar diretamente um recurso *web* existente ou cria uma URI, se necessário.
- DELETE: é utilizado para apagar as informações contidas no servidor *web*.

15 Disponível em: <https://materialpublic.imd.ufrn.br/curso/disciplina/3/78/3/2>. Acesso em: 30 de março de 2023.

A URL é o tipo de URI (Identificador Uniforme de recursos) utilizado pelo protocolo HTTP, sendo composta por três parte, a identificação do protocolo, o endereço do computador servidor e o documento requisitado.

3.8.1.1 *WEB* SERVER

O servidor *web*, do inglês *web server*, é um *software* responsável por aceitar requisições em HTTP de clientes e servi-los com respostas em HTTP. O termo *web server* refere-se a parte física, conhecida como *hardware* e o sistema operacional ou *software*, que são partes integrantes de um conjunto de armazenamento, transmissão e gerenciamento de dados (TEIXEIRA, 2004).

O *hardware* na web *server* é um computador que armazena arquivos que compõem os sites, como por exemplo, documentos em HTML, imagens folhas de estilo e arquivos *JavaScript*, com o intuito de os entregar para o cliente final. Já o *software*, inclui diversos componentes que controlam como os clientes acessam os arquivos hospedados. Um servidor HTTP é um *software* que compreende URLs e o protocolo que seu navegador utiliza para visualizar as páginas *web* (TEIXEIRA, 2004).

O pedido HTTP que se referem as páginas em HTML que são realizados através dos navegadores. Esse processo se inicia com a conexão entre o computador onde está instalado o servidor *web* e o computador do cliente. Ao realizar uma pesquisa em um *browser* e definir quais são as necessidades e essa requisição alcançar o servidor *web* correto, o *software* realizará a busca através do protocolo HTTP e ao encontrar a informação requisitada realizara o envio do documento necessário, também via HTTP.

Para publicar um site, a origem do conteúdo enviado pelo *web* server numa resposta a um pedido em HTTP pode ser Estática ou dinâmica.

- Servidor *web* estático: é chamado assim porque o servidor envia seus arquivos tal como foram criados e armazenador ao navegador.
- Servidor *web* dinâmico: o servidor atualiza os arquivos hospedados antes de enviá-los ao navegador através do servidor HTTP.

3.9 AMBIENTE DE DESENVOLVIMENTO DO *HARDWARE*

3.9.1 PROTEUS

O *Proteus Design Suite* surgiu em 1988, desenvolvido pela empresa *Labcenter Eletronics*. Criado para facilitar o mundo da eletrônica o *software* Proteus é uma ferramenta ideal para os estudantes e profissionais que desejam desenvolver aplicações analógicas e digitais. Ele permite desenvolvimento completo de projetos através de esquemáticos, simulações e *layout's Printed Circuit Board* (PCB) (ANACOM, 2010).

O grande diferencial desse *software* com relação a outros é a capacidade de simular circuitos elétricos e circuito microcontrolados, pois além de fornecer componentes animados, também possui as ferramentas necessárias para depurar o sistema ou *software* desenvolvido para o microcontrolador, acompanhando seu comportamento na simulação do *hardware*.

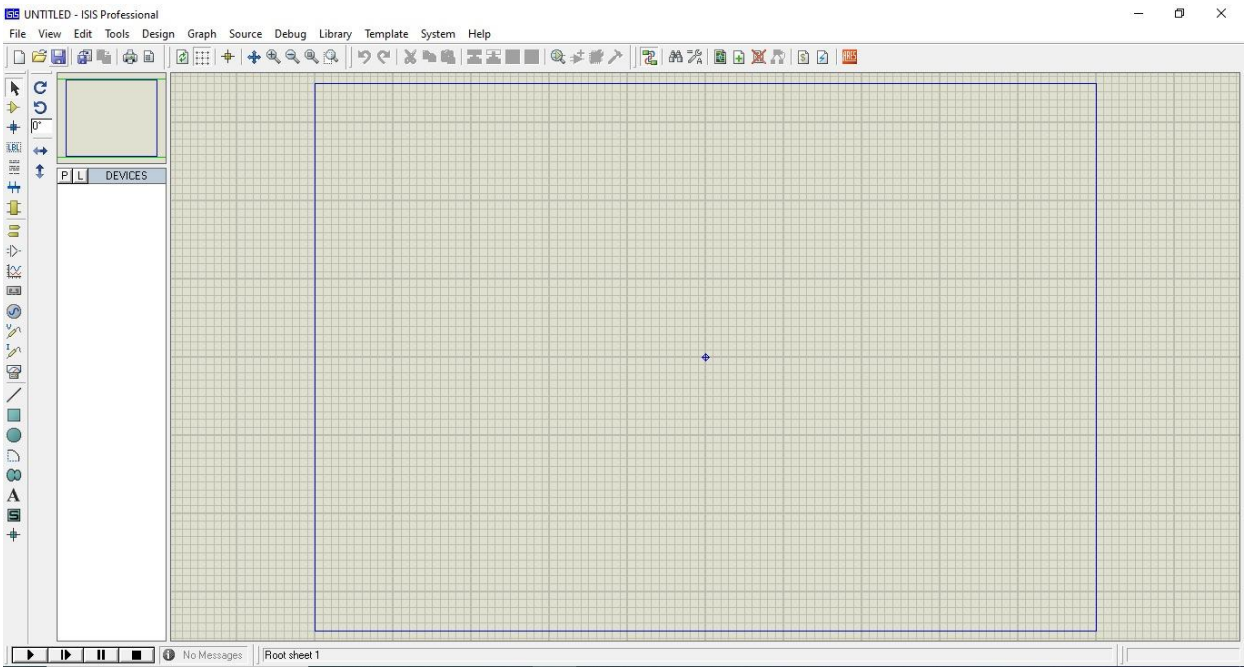
No Proteus, dois módulos trabalham em conjunto fornecendo todas as ferramentas necessárias para o processo de desenvolvimento, que são eles:

- ISIS – *Inteligent Schematic Input System* (Sistema de entrada de esquemático inteligente)
- ARES – *Advanced Routing and Editing Software* (Roteamento Avançado e Edição de *software*)

3.9.1.1 ISIS

O *software* ISIS, que é uma *interface* do Proteus, tem como finalidade funcionar como um simulador de circuitos digitais, analógicos e microprocessados. Esse programa possui uma interface gráfica simples e intuitiva, a Figura 14 demostra a tela inicial do programa.

Figura 14 – Tela inicial do ISIS



Fonte: Elaborado pelo próprio autor.

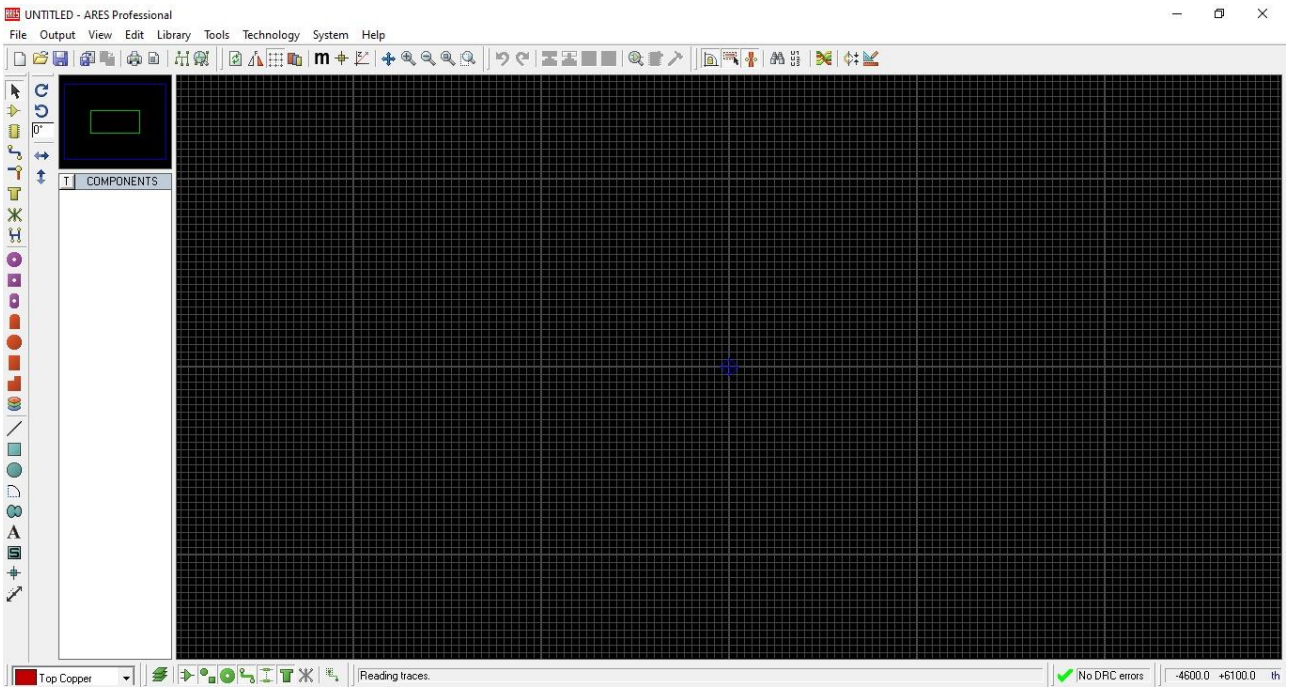
A captura esquemática do Proteus é utilizada tanto para a simulação de projetos como para a fase de projeto de um layout de PCB, sendo então, um componente central desse *software*.

Um componente utilizado na área de desenvolvimento do ISIS é uma instância de um modelo eletrônico de dispositivo ou componente eletrônico cujo símbolo/ desenho não representa necessariamente o modelo físico. Nessa área de desenvolvimento o intuito é criar o modelo matemático do circuito e realizar simulações de forma a testar a conectividade e funcionamento do circuito (ANACOM, 2010).

3.9.1.2 ARES

O programa ARES é uma *interface* do Proteus especialmente elaborada para criação de *layout's* de circuito impresso sem esquemático. Sendo que, possui a opção de construir um *layout* ou importar o arquivo *netlist*, criado previamente no ISIS. Sendo assim, esse programa é responsável por criar o projeto físico da placa, criar as trilhas e definir as regras de roteamento. A Figura 15 apresenta a tela inicial do ARES.

Figura 15 – Tela inicial do ARES



Fonte: Elaborado pelo próprio autor.

Ao construir o esquemático do circuito no ISIS usando os componentes é gerado o *netlist* do circuito, que é responsável por descrever a conectividade de um circuito eletrônico. Ao iniciar o ARES esse *netlist* é importado para a área de design do *layout* da placa, transformando os componentes em encapsulamentos. Diferentemente do componente, o encapsulamento é a representação física de um componente eletrônico de um fabricante e/ou modelo específico, com as mesmas dimensões e características. E dessa forma, torna-se possível a construção de uma PCB do circuito desejado (ANACOM, 2010).

O visualizador 3D do Proteus, permite extrair de um *layout* criado no ARES a visualização da placa em 3D, como que na realidade. Permitindo assim, uma melhor visualização da disposição dos encapsulamentos e dimensões do projeto.

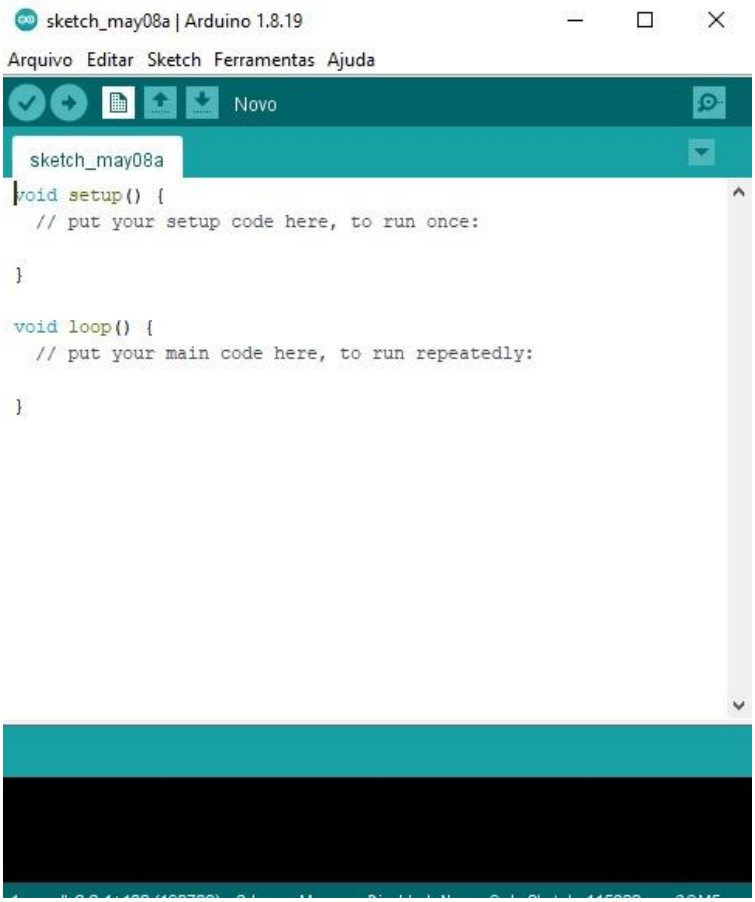
3.10 AMBIENTE DE DESENVOLVIMENTO DO *SOFTWARE*

3.10.1 IDE ARDUINO

A Arduino *Integrated Development Environment* (IDE Arduino) é um *software open source* escrito em funções de C e C++ utilizado no desenvolvimento da programação de qualquer placa Arduino e de alguns outros microcontroladores, tais como os da família ESP (OLIVEIRA, 2019).

Este ambiente de desenvolvimento contém um editor de texto para escrita do código, uma caixa de mensagem na parte inferior onde são apresentados o status do programa, serial monitor e uma barra de ferramentas com uma série de menus e funções, como apresentado na Figura 16.

Figura 16 – IDE Arduino



Fonte: Elaborado pelo próprio autor.

Esse ambiente de programação possui um *layout* completo e de fácil navegação, tornando esse *software* perfeito para desenvolvedores iniciantes. Essa plataforma se destaca devido a quantidade de material existente para consulta, bibliotecas desenvolvidas que facilitam a programação e exemplos de aplicações, tornando simples a sua utilização (OLIVEIRA, 2019).

Segundo Oliveira (2017), os programas que são denominados "*sketchs*" possuem duas funções obrigatórias, a função *setup* e a função *loop*. A função *setup* é chamada durante a inicialização do dispositivo, nela deve conter todos os parâmetros de instruções de configuração do código que deverão ser executadas pelo Arduino apenas uma vez.

Já a função *loop* funciona como um laço de repetição, sempre ao seu final, a função é chamada novamente até que um comando seja dado para a função parar ou o dispositivo desligar. Nessa função são realizadas todas as chamadas que são pertinentes ao funcionamento normal da placa, o que o programa deve realizar ao longo do seu funcionamento. (OLIVEIRA, 2017)

Para a utilização das placas ESP na IDE Arduino é necessário realizar alguns ajustes para que a programação seja possível. O Anexo A explica detalhadamente o passo a passo da instalação e configuração da IDE Arduino para as placas ESP.

4 METODOLOGIA

4.1 DESENVOLVIMENTO DO *HARDWARE*

De acordo com Siqueira Filho e Silva Filho (2006), o *hardware* é constituído da parte mecânica e física do computador sendo composto pelos componentes eletrônicos, peças e equipamentos que fazem o computador funcionar.

Tendo como premissa o desenvolvimento de um protótipo acessível e de baixa complexidade para a realização da automação de alguns acionamentos elétricos presentes em uma residência, com o intuito de oferecer acessibilidade, comodidade e segurança aos usuários.

O *hardware* proposto neste trabalho tem como papel principal o recebimento de informações do ambiente através das portas de entrada analógicas e digitas do microcontrolador e após o processamento das informações, ativar ou desativar as portas de saída do microcontrolador, sendo ele composto por três parte. São elas:

- Placa optoacopladora: responsável pela proteção da placa principal, através da utilização de componentes fotoacopladores;
- Placa de adaptação ao módulo relé: responsável pela conexão dos módulos relés a placa principal;
- Placa principal: responsável pelo processamento, coordenação e execução de todos os comandos contidos no *software*.

Com a finalidade de atuar no controle e monitoramento remoto dos acionamentos de cargas elétricas, o protótipo utilizará além do software para o perfeito funcionamento do hardware, uma *web server* para que assim possa ser possível o acesso, controle e monitoramento pelos principais dispositivos com acesso a rede Wi-Fi.

Com o propósito de tornar a reprodução e montagem o mais simples possível, o protótipo desenvolvido inclui materiais de baixo custo e acessíveis. Na Tabela 06 estão dispostos todos os componentes utilizados para a elaboração do protótipo, bem como seus custos e quantidades:

Tabela 06 - Levantamento de custos

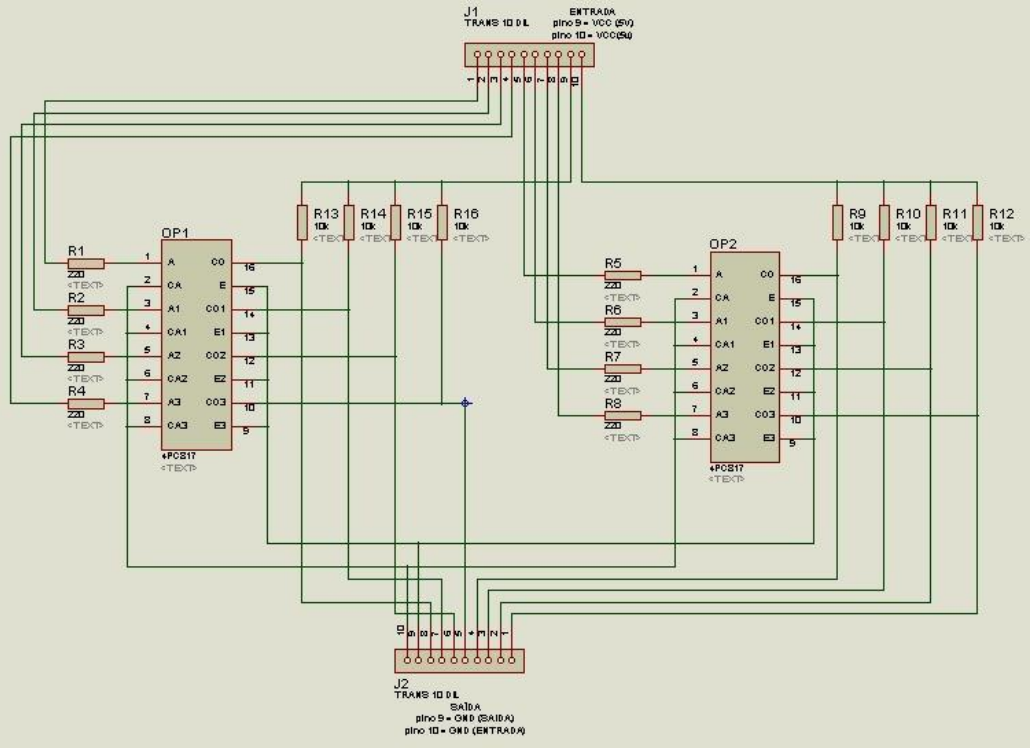
DESCRIÇÃO	QUANT.	VALOR UNITÁRIO	PREÇO TOTAL
Arduino Atmega 2560 Pro Mini	1	148,50	148,50
ESP-32	1	44,25	44,25
Módulo relé	1	52,79	52,79
Regulador de tensão LM2596	1	12,90	12,90
Módulo RF 433MHz	1	12,90	12,90
PC817C	64	0,59	37,76
Resistores 220Ω	64	0,07	4,48
Resistores de 10k Ω	64	0,07	4,48
Borne KRE duas vias	16	0,99	15,84
Conector Jack RJ45	16	3,90	62,40
Soquete de CI 16 pinos	16	0,75	12,00
Barra de pinos Macho 1x40	10	2,90	29,00
Barra de pinos Fêmea	10	2,37	23,70
Impressão das placas	1	140,00	140,00
CUSTO TOTAL			**601,00**

Fonte: Elaborado pelo próprio autor.

4.1.1 PLACA OPTACOPLADORA

A Figura 17, apresenta o circuito esquemático desenvolvido na interface Proteus ISIS, que é compostos além dos optoacopladores, por resistores limitadores de corrente interligados aos anodos dos optoacopladores e resistores *pull-up* para garantir o nível lógico desejado na saída.

Figura 17 – Esquemático da placa optoacopladora no ISIS



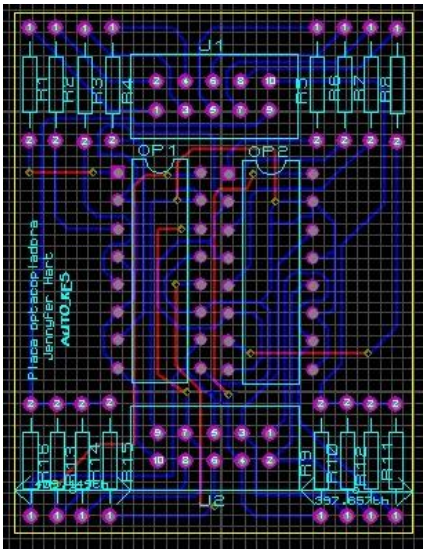
Fonte: Elaborado pelo próprio autor.

Tendo em vista que o componente optoacoplador será o responsável pela proteção de todo o sistema, sendo então suscetível a maiores chances de defeito, os componentes representados como OP1 e OP2 no ISIS, foram criados para representar um soquete para CI de 16 pinos que tem capacidade para receber 4 optoacopladores, respectivamente.

O soquete para CI é muito utilizado no desenvolvimento de projetos eletrônicos, principalmente em placas de circuito impresso, pois permitem com maior facilidade a troca de componentes danificados sem que seja necessário a retirada da soldagem dos componentes.

O circuito esquemático foi exclusivamente desenvolvido para que fosse possível a importação desse circuito para o ARES, não sendo possível a simulação do mesmo. Após importar o circuito esquemático, foi possível organizar a disposição dos encapsulamentos e fazer o roteamento das trilhas, criando o projeto base da PCB, conforme mostra a Figura 18.

Figura 18 – Projeto base da PCB da placa optoacopladora

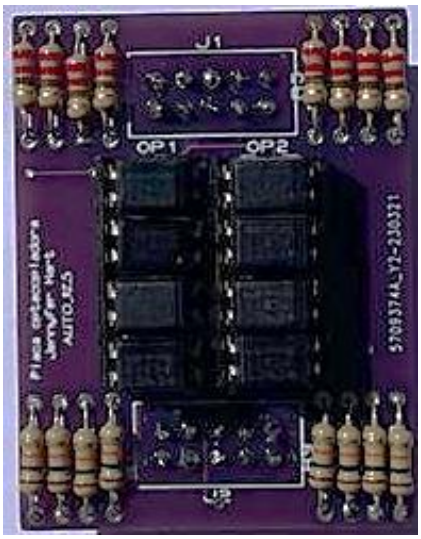


Fonte: Elaborado pelo próprio autor.

Com o intuito de facilitar o processo de fabricação da PCB, torná-la mais robusta, com uma melhor qualidade e visivelmente atraente no aspecto comercial, optou-se pela fabricação da PCB com empresa especializada.

E para a finalização da placa optoacopladora foi realizada a soldagem e instalação dos componentes eletrônicos, a Figura 19 demonstra a placa finalizada.

Figura 19 – Placa optoacopladora



Fonte: Elaborado pelo próprio autor

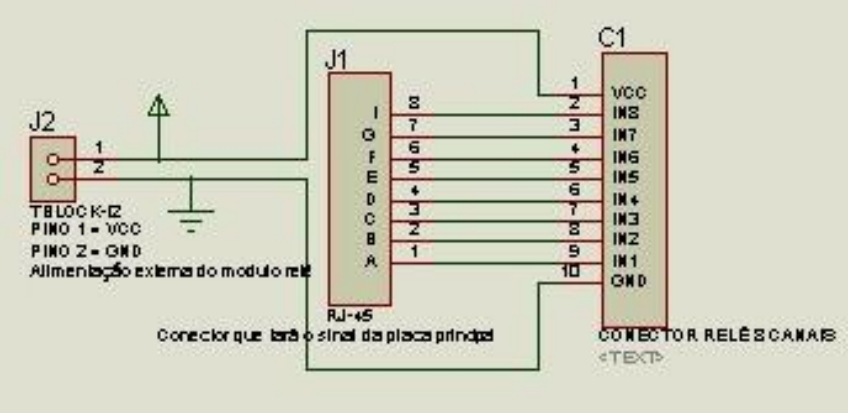
4.1.2 PLACA ADAPTADOR PARA MODULO RELÉ

O módulo relé é utilizado para facilitar o acionamento de cargas através de um microcontrolador, necessitando apenas de conectores jumpers no relé e em uma placa Arduino, por exemplo, dispensando a necessidade de montar placas ou circuitos para a sua ativação.

Pensando na possibilidade da instalação do protótipo em uma residência, no qual a placa principal estará a uma distância considerável das cargas que deverão ser acionadas e tendo como premissa a facilidade e organização, a placa adaptadora para módulo relé foi desenvolvida com o intuito de diminuir a quantidade de fios que devem ser ligados ao módulo relé.

Utilizando apenas uma barra de pinos fêmea, um borne KRE duas vias e um conector Jack RJ45, o circuito esquemático da placa adaptadora para módulo relé, apresentado na Figura 20, é simples e prático. Diminuindo a quantidade de fios para o acionamento de 10 para apenas 3 fios. Sendo necessário apenas um cabo RJ45 para enviar os sinais de acionamentos, um para alimentação VCC e um para GND.

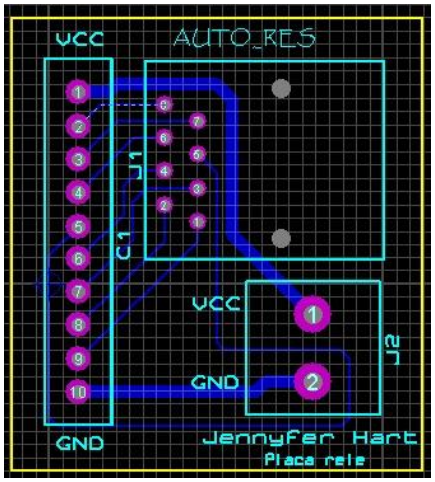
Figura 20 – Circuito esquemático da placa adaptadora para módulo relé



Fonte: Elaborado pelo próprio autor.

Utilizando o mesmo processo de desenvolvimento da placa optoacopladora, após a finalização do esquemático, o circuito foi importado no ARES, para a criação do projeto base da PCB. O projeto base é demonstrado na Figura 21.

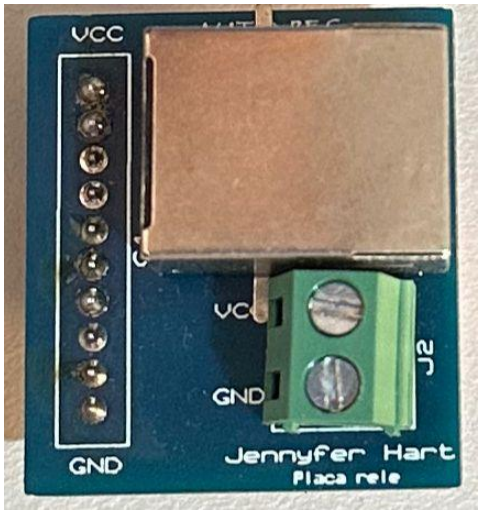
Figura 21 – Projeto base da placa adaptadora para módulo relé



Fonte: Elaborado pelo próprio autor.

Na Figura 22 é apresentada a placa adaptadora para módulo relé finalizada com a instalação de todos os componentes eletrônicos.

Figura 22 – Placa adaptadora para módulo relé



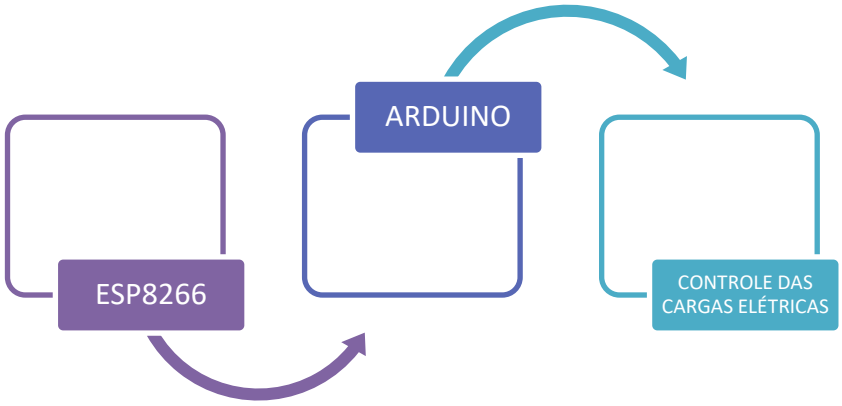
Fonte: Elaborado pelo próprio autor.

4.1.3 PLACA PRINCIPAL

Visando atingir os objetivos deste trabalho, o protótipo da placa principal foi desenvolvido para ser responsável por realizar toda a coordenação do sistema de automação residencial. Utilizando os microcontroladores ESP-01 e o Arduino ATMEGA 2560 Pro Mini, sendo o ESP-01 o microcontrolador principal, programado como servidor *web*. O servidor *web* disponibiliza uma página HTML, permitindo que o usuário, através da rede Wi-Fi realize o acionamento das cargas elétricas desejadas.

Utilizando-se da comunicação mestre-escravo, onde apenas um único dispositivo pode iniciar a comunicação e os demais devem apenas responder enviando os dados solicitados pelo mestre ou executando alguma ação solicitada, o ESP-01 será o mestre dessa comunicação com uma *interface web* e se comunicará com o escravo que será o Arduino através da porta Serial, de forma a comandar e ler as portas digitais e analógicas. À vista disso, o Arduino será utilizado para ampliar a quantidade de portas digitais e analógicas do ESP-01. A Figura 23 demonstra como ocorrerá a comunicação entre os dispositivos.

Figura 23 – Fluxograma de funcionamento



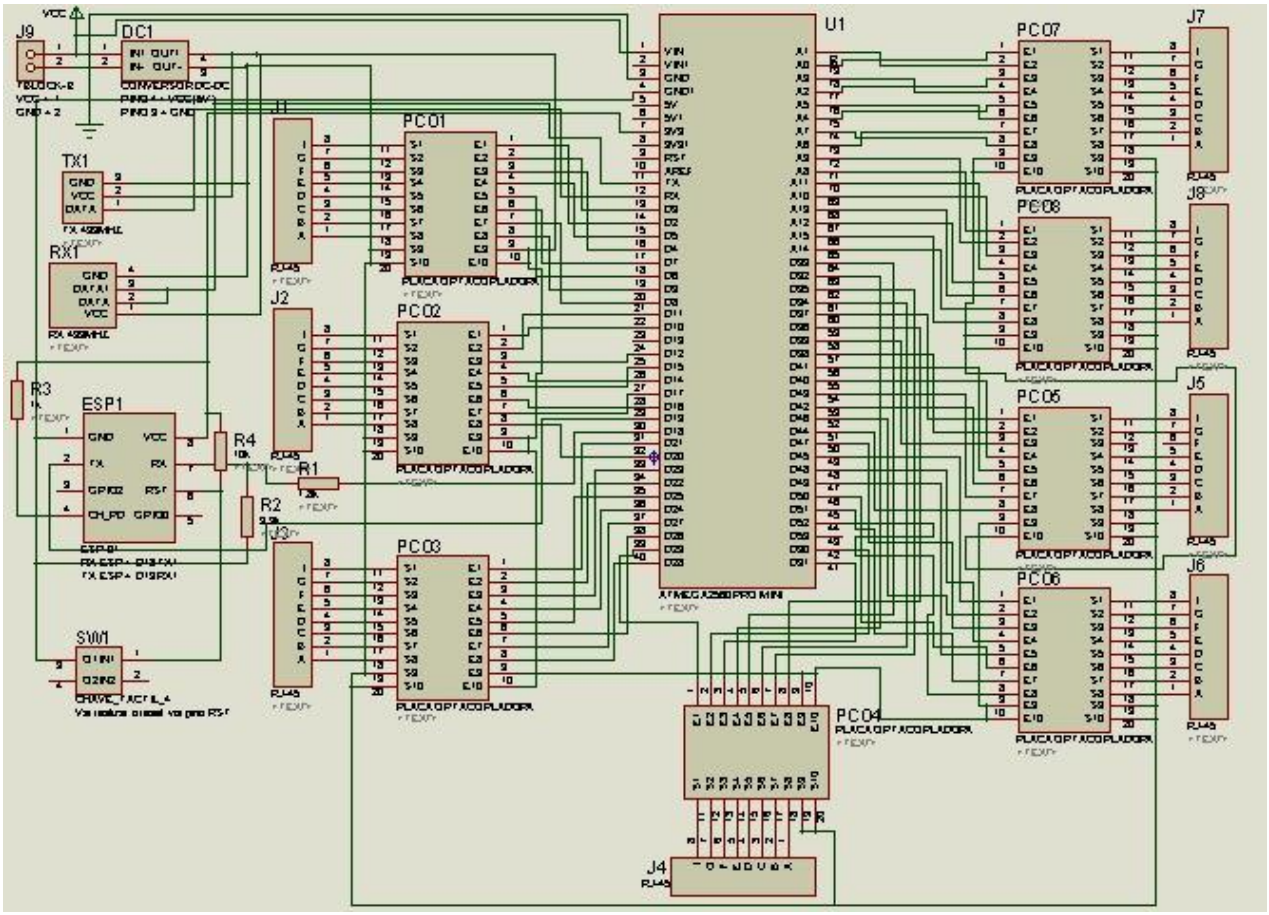
Fonte: Elaborado pelo próprio autor.

Com a utilização do Arduino para expandir a quantidade de porta disponíveis para automação, o protótipo possui 64 portas que podem ser configuradas como entrada ou saída, de acordo com a necessidade do usuário.

Além dos microcontroladores, a placa principal é composta por um módulo RF 433MHz, possibilitando o controle de portões e cortinas automáticas, oito placas optoacopladoras que

realizaram a proteção das entradas e saídas do Arduino, um regulador de tensão DC-DC *step-down* LM2596 que regulará a tensão de alimentação dos componentes a 5V, exceto o Arduino que receberá a alimentação diretamente da fonte externa. Ademais, a placa principal é composta por conectores *jack* RJ45 que facilitaram a conexão com os atuadores e um botão para realizar o *reset* do ESP-01. A Figura 24 apresenta o circuito esquemático da placa principal feito na *interface* ISIS.

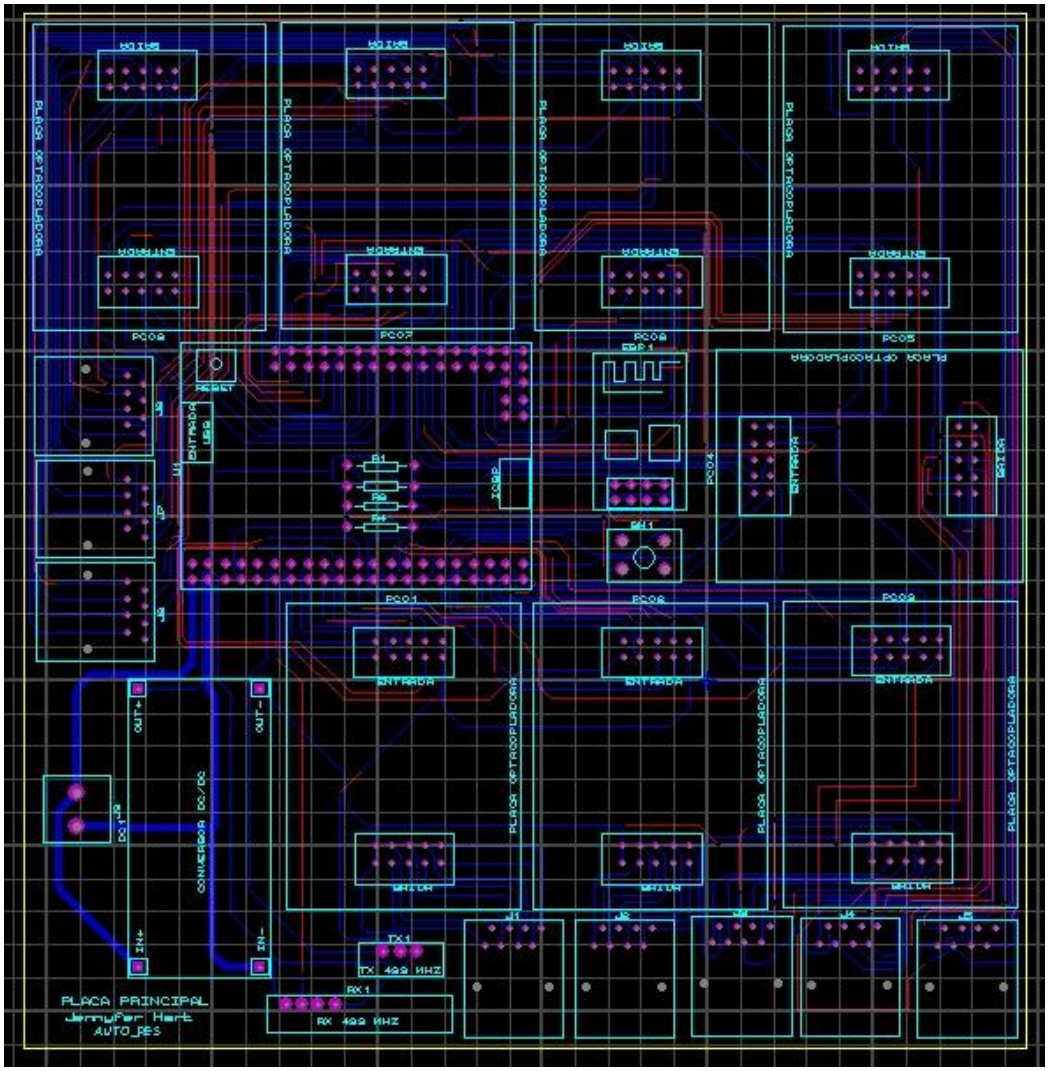
Figura 24 – Circuito esquemático da placa principal



Fonte: Elaborado pelo próprio autor.

A Figura 25 apresenta o projeto base feito no ARES da placa principal. Com o intuito de facilitar a manutenção, todos os componentes serão instalados na placa principal através de conectores feitos com barras de pinos fêmea e macho.

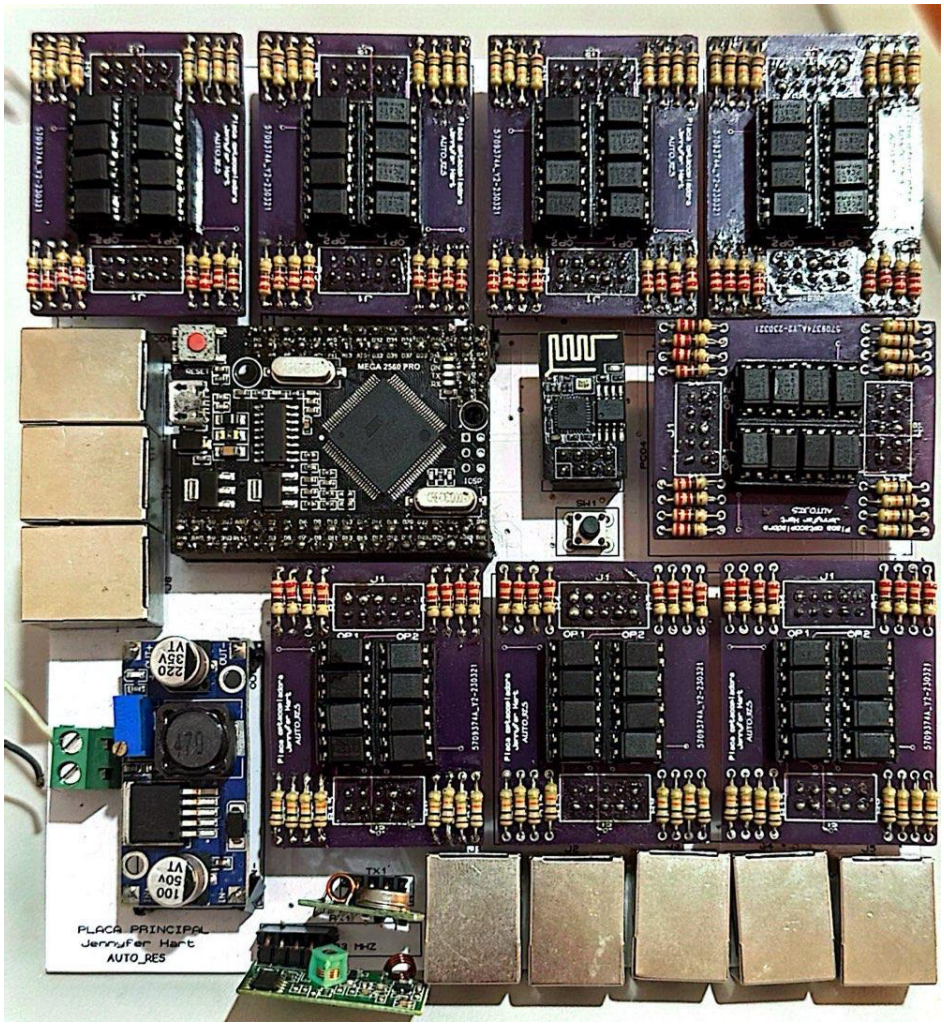
Figura 25 – Projeto base da placa principal



Fonte: Elaborado pelo próprio autor.

Já a Figura 26, apresenta a placa principal finalizada após a soldagem de todos os componentes eletrônicos e inserção das placas complementares.

Figura 26 – Placa principal finalizada



Fonte: Elaborado pelo próprio autor.

Com a finalidade de facilitar durante a utilização, a Tabela 07 apresenta alguns dados técnicos da placa principal, como por exemplo as portas de saída ou entrada do Arduino e suas respectivas portas destinadas ao controle de entradas e saídas.

Tabela 07 – Mapeamento dos pinos do Arduino

MAPEAMENTE DOS PINOS DE CONTROLE			
SAÍDA	**PINO**	**ENTRADA**	**PINO**
PC1		**PC5**	
S1	D2	E1	D37
S2	D3	E2	D38

S3	D4	E3	D39
S4	D5	E4	D40
S5	D6	E5	D41
S6	D7	E6	D42
S7	D8	E7	D43
S8	D9	E8	D44
PC2		**PC6**	
S1	D10	E1	D45
S2	D11	E2	D46
S3	D12	E3	D47
S4	D14	E4	D48
S5	D15	E5	D49
S6	D16	E6	D50
S7	D17	E7	D51
S8	D20	E8	D52
PC3		**PC7**	
S1	D21	E1	A0
S2	D22	E2	A1
S3	D23	E3	A2
S4	D24	E4	A3
S5	D25	E5	A4
S6	D26	E6	A5
S7	D27	E7	A6
S8	D28	E8	A7
PC4		**PC8**	
S1	D29	E1	A8
S2	D30	E2	A9
S3	D31	E3	A10
S4	D32	E4	A11
S5	D33	E5	A12
S6	D34	E6	A13
S7	D35	E7	A14
S8	D36	E8	A15

Fonte: Elaborado pelo próprio autor.

4.2 DESENVOLVIMENTO DO *FIRMWARE*

Após a realização de muitas pesquisas, a utilização da IDE do Arduino mostrou-se extremamente viável para o desenvolvimento do firmware, devido a ser um *software open-source* e possuir uma variedade de projetos já implementados na área de automação.

Para atingir os objetivos esperados para esse projeto, a programação foi dividida em duas partes. Primeiramente é realizada a programação do microcontrolador ESP-01, o qual realizará a conexão Wi-Fi e realizará a criação e disponibilização das páginas *web*, para serem acessadas através de um navegador da internet. Posteriormente é realizada a programação do microcontrolador Arduino, o qual realizará a comunicação via porta Serial com o ESP-01 e de acordo com os comandos recebido, realizará o acionamento e monitoramento de suas portas digitais e analógicas.

Nos itens 4.2.1 e 4.2.2 serão abordadas as metodologias utilizadas para a realização dos *firmwares* para o ESP-01 e para o Arduino.

4.2.1 ESP-01

Para a implementação da programação do ESP, foi necessário a inclusão de algumas bibliotecas, para que seja possível o funcionamento da programação. Na Figura 27 é demostrado as bibliotecas utilizadas, sendo que a primeira permite a conexão Wi-Fi da placa com a rede, a segunda possibilita as requisições HTTP da placa, com o IP gerado pelo próprio ESP.

Figura 27 – Bibliotecas utilizadas

```
#include <ESP8266WiFi.h>
#include <WebServer.h>
#include <Preferences.h>
```

Fonte: Elaborado pelo próprio autor.

Já a terceira permite salvar dados na memória NVS (Armazenamento não Volátil), sendo essa um invólucro localizado em torno do armazenamento não volátil no processador do ESP,

sendo assim, uma vez que sejam gravados os dados, esses serão conservados mesmo que haja quedas abruptas de energia.

Diferentemente da biblioteca “SPIFFS.h” que permite salvar arquivos na memória flash do ESP, a biblioteca “Preferences.h” permite salvar dados na memória e foi utilizada para facilitar a inclusão de nome amigável para os dispositivos que serão controlados via *web*.

Na função de inicialização do microcontrolador, o void setup(), que deve possuir todos os parâmetros iniciais para execução do código fonte, foi realizada a inicialização da porta Serial e da memória NVS, além da conexão com o Wi-Fi. Na Figura 28 é apresentado parte da programação.

Figura 28 – Estruturação do void setup

```
void setup() {
  Serial.begin(115200);
  nvs.begin("auto_res", false);

  if(nvs.isKey("ssid") == false){
    WiFi.mode(WIFI_AP_STA);
    WiFi.softAP(serial_number, "12345678");
  }

  else{
    String ssid = nvs.getString("ssid");
    String password = nvs.getString("password");
    WiFi.mode(WIFI_AP_STA);
    WiFi.softAP(serial_number, "12345678");
    WiFi.begin(ssid, password);

    int i = 0;
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
        i++;
        if(i >=20){
          Serial.println("");
          break;
        }
    }
}
```

Fonte: Elaborado pelo próprio autor.

Para finalizar as configurações realizou-se o mapeamento do caminho das requisições que serão executadas pelo servidor *web*, no qual é realizada a configuração que define qual URL vai executar cada função e inicializa o servidor, conforme mostrado na Figura 29.

Figura 29 – Definições das URLs

```
void setup() {

  server.on("/", handle_on_connect);
  server.on("/?", handle_on_connect);
  server.on("/outputs", handle_outputs);
  server.on("/set_output", handle_set_output);
  server.on("/set_all_outputs", handle_set_all_outputs);
  server.on("/inputs", handle_inputs);
  server.on("/names", handle_names);
  server.on("/names?", handle_names);
  server.on("/set_names", handle_set_names);
  server.on("/wifi", handle_wifi);
  server.on("/wifi?", handle_wifi);
  server.on("/set_wifi", handle_set_wifi);
  server.on("/set_wifi?", handle_set_wifi);
  server.on("/restart", handle_restart);
  server.on("/restart", handle_restart);
  server.onNotFound(handle_not_found);

  server.begin();
}
```

Fonte: Elaborado pelo próprio autor.

Já na função void loop(), é inserido apenas a função handleClient(), que tem a função de gerenciar as requisições, respondendo conforme o mapeamento e configurações criadas do *setup*.

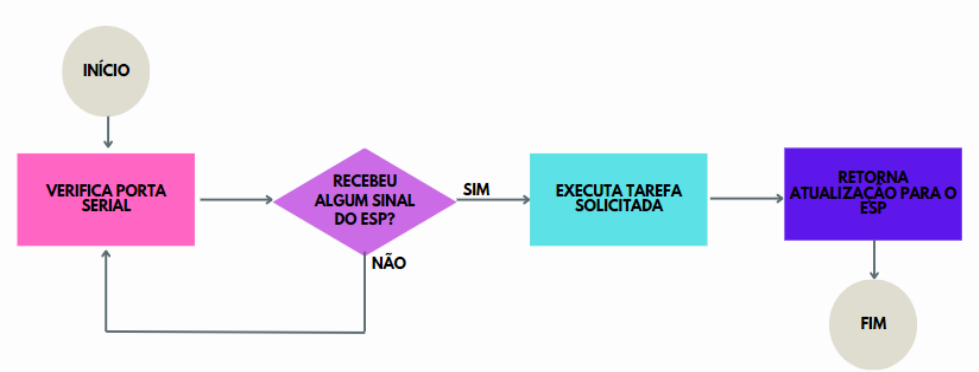
Para cada URL criada é necessário a implementação de função que serão executadas de acordo com cada página *web* e as requisições do cliente, permitindo assim, a criação de páginas de configurações, informações e controle.

4.2.2 ARDUINO

Para dar início a programação do Arduino, todos os pinos são configurados como entrada ou saída, a depender do dispositivo que está conectado a esse pino, ou ação a ser executada. Com o intuito de exemplificar, para a configuração dos pinos foi seguido a Tabela 7 apresentada anteriormente.

Conforme a lógica de programação, o Arduino verifica constantemente a porta serial, com o intuito de realizar a comunicação serial com o ESP. Quando algum sinal é recebido, esse sinal é interpretado através do *firmware* carregado no Arduino e assim, é realizada a execução da tarefa, seja ela o acionamento de algum periférico ou leitura de algum dado. Na Figura 30 é demostrado um fluxograma com a lógica de programação seguida pelo o Arduino.

Figura 30 – Lógica de programação do Arduino



Fonte: Elaborado pelo próprio autor.

Para a comunicação entre o ESP e Arduino foi criado um protocolo de comunicação a ser seguido pelos dois microcontroladores, como demonstrado na Figura 31. Os caracteres especiais foram utilizados no protocolo pois são mais fáceis de rastrear dentro de uma *string*.

Figura 31 – Protocolo de comunicação ESP+ARDUINO

```
// Exemplos de comando
// Estrutura de requisição de status Outputs / Inputs
// Requisição + Tipo de entrada / Saida + Porta + ?
// Ex: STATUS+OUTPUT=1?

// Estrutura da Resposta
// RESP + Requisição + Tipo de entrada / Saida = Porta,Valor  + #
// RESP+STATUS+OUTPUT=25,1#

// Estrutura de requisição de status Outputs / Inputs
// Requisição + Tipo de entrada / Saida + Porta + ?
// Ex: STATUS+INPUT=25?

// Estrutura da Resposta
// RESP + Requisição + Tipo de entrada / Saida = Porta,Valor  + #
// RESP+STATUS+INPUT=25,1#

// Exemplo para definir um valor em um Output
// Requisição + Tipo de entrada / Saida + Porta,Valor + ?
// Ex: SET+OUTPUT=25,1?

// Estrutura da Resposta
// RESP + Requisição + Tipo de entrada / Saida = Porta,Valor  + #
// RESP+SET+OUTPUT=1#
```

Fonte: Elaborado pelo próprio autor.

As principais funções utilizadas na programação do Arduino foram a digitalWrite(), sendo responsável por atribuir um valor *HIGH* (alto) ou *LOW* (baixo) em um pino que tenha sido

configurado como saída (*OUTPUT*) e a função digitalRead(), que é responsável por realizar a leitura de um valor de um pino cuja a configuração atribuída a ele seja a de entrada (*INPUT*).

4.2.3 DESENVOLVIMENTO DA PÁGINA *WEB*

O desenvolvimento da página *web* foi realizado dentro do *firmware* do ESP-01, com o intuito de facilitar a programação e diminuir a ocupação de memória do dispositivo, evitando travamentos indesejados.

Ao se conectar no *WebServer* é gerado a página criada baseada nas funções de HTML, essas funções transformam pedaços de código HTML em *String* na qual é usado para facilitar a criação de página estática no ambiente de desenvolvimento.

Assim, toda página contém cabeçalho, botões e tabelas, dessa forma essas funções concatenam essas informações em uma única *String* e informa ao navegador a função pronta. Na Figura 32 é demostrado um exemplo, no qual é criado a "*string contente"*, essa recebe os dados da função "html_header" que consequentemente retorna todo o cabeçalho HTML, após isso é usado as funções de criação de botões para finalizar a tela.

Figura 32 – Exemplo de função para criação de página *web*

```
166 void handle_on_connect() {
167
168   seconds_to_time(esp_timer_get_time() / 1000000, active_time);
169
170   String content = html_header();
171   content += "<table align=center>";
172   content += insert_table_line("Nome do dispositivo", serial_number);
173   content += insert_table_line("Versão do Firmware ", app_version);
174   content += insert_table_line("Tempo ativo", active_time);
175
176   content += "</table>";
177   content += "<br>";
178   content += "<br>";
179   content += "<center>";
180
181   content += insert_form_line_get("outputs", "CONFIG. SAIDAS");
182   content += insert_form_line_get("inputs", "CONFIG. ENTRADAS");
183   content += insert_form_line_get("names", "CONFIG. NOMES");
184   content += insert_form_line_get("wifi", "CONFIG. WIFI");
185   content += insert_form_line_get("restart", "REINICIAR SISTEMA");
186
187   content += "</center></body></html>";
188
189   server.send(200, "text/html", content);
190 }
```

Fonte: Elaborado pelo próprio autor.

Para cada página *web* foi criada diversas funções utilizando a mesma lógica de programação, apresentado na Figura 32.

5 INTEGRAÇÃO DO SISTEMA E FUNCIONAMENTO

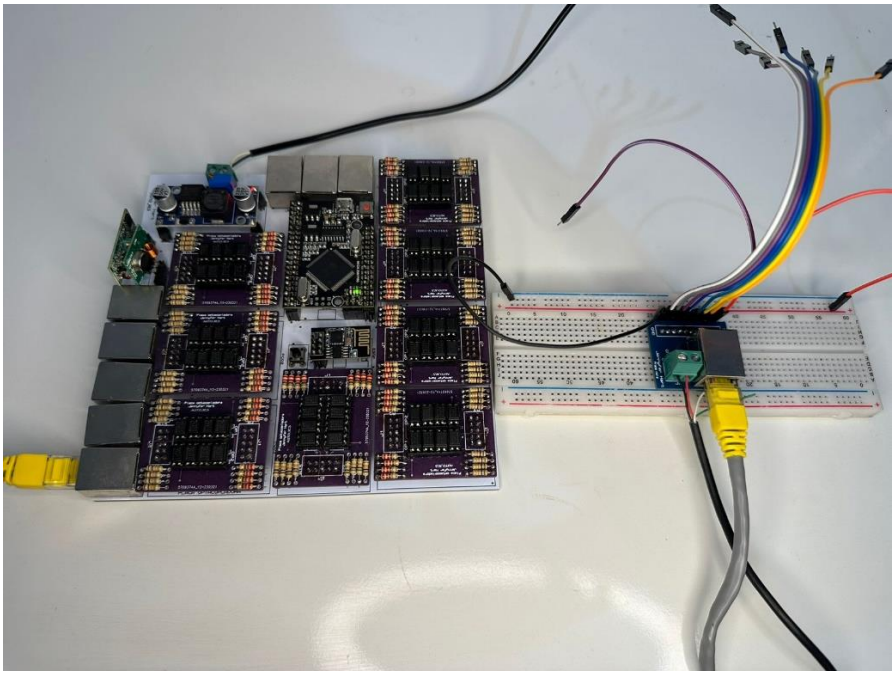
Com intuito de demonstrar o funcionamento do *hardware*, foi interligada a placa principal a placa adaptadora de relés afim de simular o acionamento de cargas elétricas. Na Figura 33 é demostrado o *hardware* em funcionamento.

Figura 33 – *Hardware* em funcionamento

Fonte: Elaborado pelo próprio autor.

Entretanto, a mesma placa adaptadora de relés também pode ser utilizada para simular as entradas e dessa forma, sendo possível a realização de testes e verificação do funcionamento. Na Figura 34 é demostrado a adaptação da placa adaptadora de relés para a realização da leitura das entradas. Foi utilizado *jumper* para simular interruptores, sendo possível a leitura de *ON* ou *OFF*.

Figura 34 – *Hardware* em funcionamento



Fonte: Elaborado pelo próprio autor.

Além disso, para que seja possível a realização do monitoramento das portas de entrada do Arduino é necessário inverter as placas optacopladoras, permitindo que os circuitos com fototransistores libere a passagem de corrente.

O *software* de controle foi batizado com o nome Auto_Res, o qual é responsável pela comunicação entre o ESP e Arduino, enviando e recebendo dados para a execução dos comandos. Na Figura 35 é apresentado a logo escolhida para o *software*.

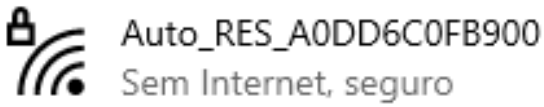
Figura 35 – Logo

Fonte: Elaborado pelo próprio autor.

Para que seja possível o gerenciamento das portas do Arduino via página *web* é necessário que o ESP esteja conectado a uma rede Wi-Fi. Para essa conexão serão utilizados os dois modos de operação do ESP-01, modo *Access Point* e *Station*, já explicados anteriormente.

A conexão com o modo *Access Point* (AP) NO ESP8266 é realizada através de uma rede disponibilizada pelo próprio dispositivo e com uma senha padrão. Essa rede possui um nome específico, que foi configurado conforme o projeto e logo escolhida, e um número serial que é único para cada dispositivo ESP e a senha de acesso padrão é “12345678”. Na Figura 36 e demostrado a rede a se conectar para acessar via *Access Point*.

Figura 36 – Nome da rede Wi-Fi instanciada pelo ESP



Fonte: Elaborado pelo próprio autor.

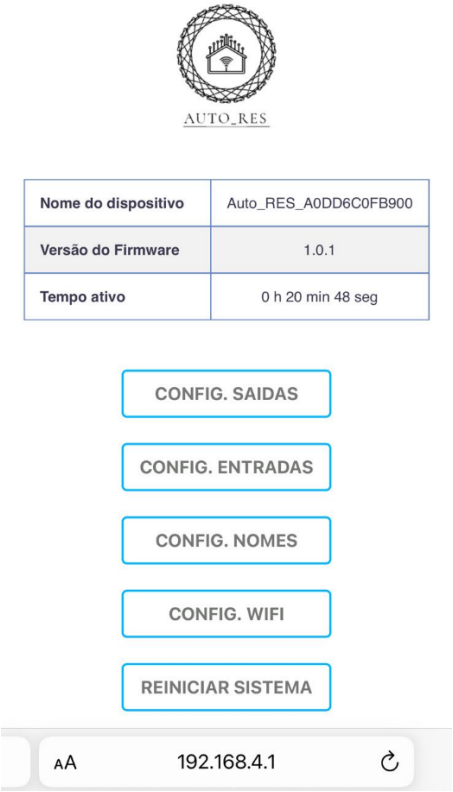
Nesse modo o computador ou smartphone se conecta à rede criada pelo ESP e para acessar a página *web* é utilizado o IP padrão do ESP, conforme apresentado na Figura 37.

Figura 37 – IP padrão do ESP para acesso a página *web*

Fonte: Elaborado pelo próprio autor.

Ao acessar o servidor *web*, através de qualquer um dos modos de conexão, a primeira página *web* que pode ser visualizada é a página Inicial, apresentada na Figura 38. Nesta página é apresentada a logo, o nome do dispositivo, versão do *firmware*, tempo ativo e os botões de redirecionamento para as outras páginas *web*.

Figura 38 – Página inicial do Auto_Res



Fonte: Elaborado pelo próprio autor.

Ao clicar no botão Wi-Fi o usuário é redirecionado para a página de configuração da rede a qual o ESP irá se conectar no modo *Station*, essa página é apresentada na Figura 39.

Figura 39 – Página de configuração do modo Wi-Fi *Station* do Auto_Res

Fonte: Elaborado pelo próprio autor.

Após acessar a página *web* pelo IP padrão é possível realizar a conexão Wi-Fi pelo modo *Station*, inserindo os dados de credenciamento, como nome da rede e senha. Após a conexão com a rede Wi-Fi local o dispositivo recebe um endereço de IP, sendo que este será utilizado para acessar as páginas *web* quando o usuário estiver conectado ao Wi-Fi local. A Figura 40 apresenta o IP obtido através das informações capturadas pelo monitor serial do ESP após a conexão com o Wi-Fi local.

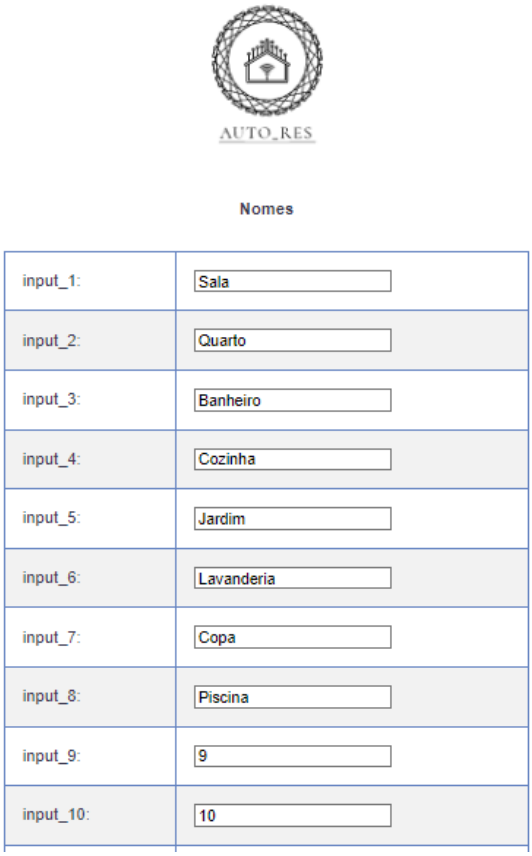
Figura 40 – IP gerado após conexão com Wi-Fi local para acesso a página *web*

```
Connecting to Cleuber
.....WiFi STA connected
IP address:
192.168.100.73
WiFi AP
IP address: 192.168.4.1
HTTP server iniciado
```

Fonte: Elaborado pelo próprio autor.

Ademais, temos a página de configuração de nomes, sendo através dessa possível adotar nomes amigáveis para os dispositivos a serem controlados ou monitorados nas páginas de controle e monitoramento. Na Figura 41 é apresentada a página de configuração de nomes com exemplos preenchidos.

Figura 41 – Página de configuração de nomes do Auto_Res



Fonte: Elaborado pelo próprio autor.

E por fim serão apresentadas as páginas de acionamento e monitoramento. A primeira denominada como “Config. Saidas”, tem a função de realizar o acionamento das saídas do Arduino e, consequentemente, atuar no controle de cargas elétricas a elas interligadas, determinando se estarão ligadas (*ON*) ou desligadas (*OFF*). Na Figura 42 é apresentada a página de acionamento.

Figura 42 – Página de acionamento de cargas elétricas

Fonte: Elaborado pelo próprio autor.

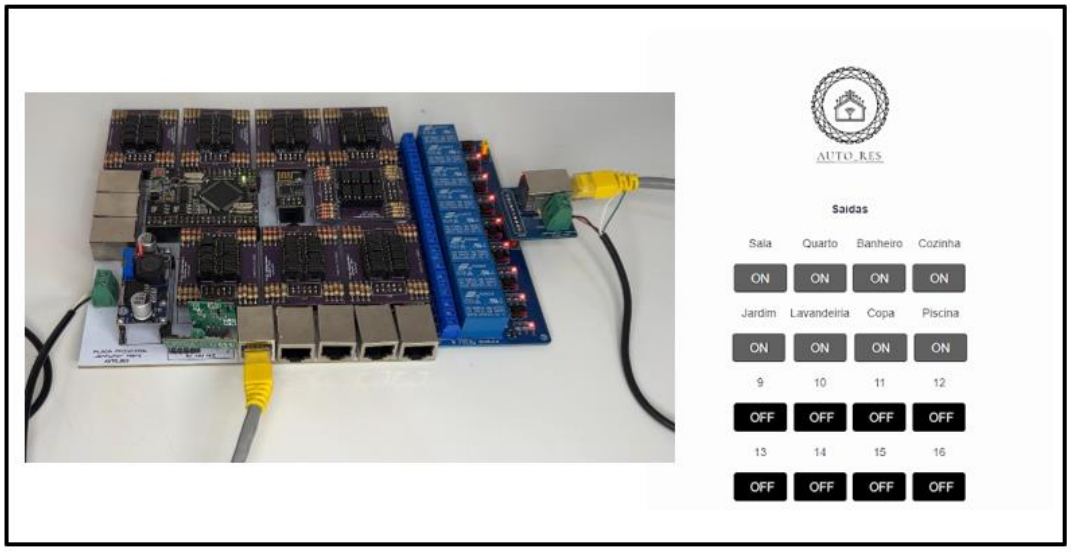
Já a página de monitoramento, apresentada na Figura 43, tem a função de realizar a leitura das entradas do Arduino, determinando o status das portas e consequentemente as informações que estão sendo recebidas pelo microcontrolador.

Figura 43 – Página de monitoramento de cargas elétricas

Fonte: Elaborado pelo próprio autor.

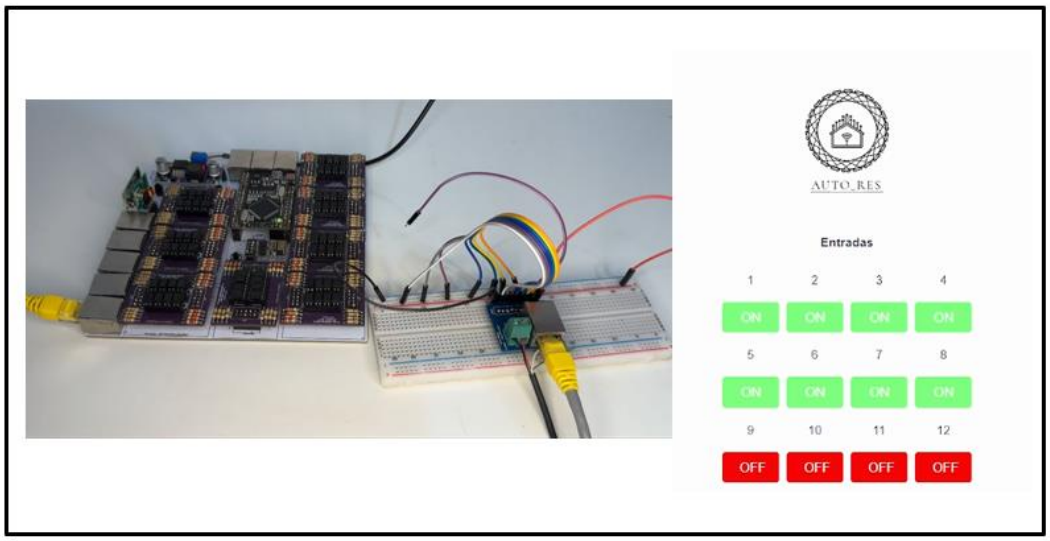
Nas Figuras 44 e 45 são apresentados a integração do sistema, demostrando o funcionamento do *hardware* em conjunto com o *software* e página *web*.

Figura 44 – Simulação de acionamento de oito cargas elétricas



Fonte: Elaborado pelo próprio autor.

Figura 45 – Simulação de monitoramento de oito carga elétrica



Fonte: Elaborado pelo próprio autor.

6 CONCLUSÕES

A busca incessante em ganhar tempo na execução de atividades rotineiras como irrigar o jardim, desligar ou ligar uma luz ou até mesmo abrir ou fechar um portão, fez com que a automação residencial ganhasse espaço no mercado. Entretanto, esse setor ainda enfrenta alguns problemas como a complexidade de desenvolver esse sistema e os altos custos para a sua implantação, tornando a sua utilização restrita a pessoas com maior poder aquisitivo.

O trabalho apresentado possuía como motivação para o desenvolvimento deste projeto acadêmico, construir um protótipo de um sistema para gerenciamento de cargas elétricas dentro de um ambiente residencial de baixo custo, de forma a automatizar via *web*. O protótipo denominado como Auto_Res, possui a capacidade de realizar o monitoramento e acionamento de cargas elétricas. Deste modo, garantindo a segurança, comodidade, conforto para seus usuários.

O protótipo se baseia em um *hardware* relativamente simples, com custo estimado de, aproximadamente, R$600,00. Demonstrando assim, a viabilidade de soluções de domótica de baixo custo.

O desenvolvimento e teste do protótipo de automação residencial demostraram a viabilidade e eficiência do sistema em simplificar e otimizar as rotinas domésticas. O protótipo se mostrou capaz de integrar diversos dispositivos, permitindo controle centralizado e operação remota através de uma interface amigável. Além disso, durante os testes, o sistema também se mostrou capaz de responder rapidamente aos comandos solicitados pelos usuários.

Entre os principais benefícios observados, destacam-se a economia de energia, através da programação e automação de dispositivos e a possibilidade de monitoramento. No entanto, algumas limitações foram identificadas, como a dependência de uma conexão estável para o funcionamento pleno do sistema e a necessidade de ajustes na interface para melhorar a usabilidade.

Em conclusão, o protótipo de automação residencial provou ser uma solução promissora para tornar as residências mais inteligentes e eficientes. Com alguns aprimoramentos, o sistema tem o potencial de se tornar uma ferramenta essencial para a gestão moderna e prática das rotinas domésticas.

Com o objetivo de dar continuidade ao desenvolvimento desse projeto, o próximo tópico apresenta possíveis temas que podem ser desenvolvidos.

6.1 TRABALHOS FUTUROS

Conforme apresentado anteriormente, a seguir encontram-se listados por meio de tópicos, os temas que podem ser abordados para dar continuidade ao trabalho desenvolvido até o momento:

- Implementação de um software mais robusto, tornando o protótipo mais atrativo e comercializável.
- Implementação de um sistema de backup para garantir a continuidade do serviço em caso de falhas na internet;
- Implementação do protótipo em situações reais do cotidiano de uma residência.
- Implementação de controle via MQTT.

REFERÊNCIAS

ALLIANCE, Wi-Fi. 2023. Disponível em: < https://www.wi-fi.org/discover-wi-fi>. Acesso em: 19 de agosto de 2024.

ALMEIDA, A. V. Implementação de um Sistema de automação residencial sem fio: Módulo periférico. Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica com ênfase em Sistemas de Energia e Automação) – Universidade de São Paulo, São Carlos, 2009.

ALMEIDA, F. M. Internet das Coisas Aplicada à Domótica. TCC (Graduação) – Curso Engenharia de Computação, Universidade Federal de Sergipe, São Cristóvão, 2013.

ANDRADE, L. F. Adaptador Inteligente para Automação Residencial Baseado em Internet das Coisas. Monografia (Graduação em Engenharia Eletrônica) – Universidade Federal de Santa Catarina, Florianópolis, 2018.

AURESIDE, Associação Brasileira de Automação Residencial e Predial. Relatório de previsões para o Mercado global de Automação residencial. Disponível em: < http://www.aureside.org.br/noticias/previsoes-para-o-mercado-global-de-aut.-residencial>. Acesso em: 02 de março de 2022.

BALL, Stuart. “Embedded Microprocessor Systems: Real World Design”, 3^{rd} edition, Editora: MCPros. EUA, 2005.

BARROS, E; CAVALVANTE, S. Introdução aos sistemas embarcados. Artigo apresentado na Universidade Federal de Pernambuco-UFPE, 2010.

BOLZANI, C.A.M. Residências Inteligentes – domótica, redes domésticas, automação residencial. São Paulo: Livraria da Física, 2004.

BOLZANI, C.A.M. Análise de Desenvolvimento de uma Plataforma para Residências Inteligentes. Tese (Doutorado em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

CHASE, Otavio. Sistemas embarcados. 2007. Disponível em: < http://www.lyfreitas.com.br/ant/pdf/Embarcados.pdf>. Acesso em: 28 de março de 2023.

COELHO, Ítalo. O que é um aptoacoplador?. 2021. Disponível em: < https://www.makerhero.com/blog/o-que-e-um-optoacoplador/>. Acesso em: 19 de agosto de 2024.

ESPRESSIF. ESP8266. Disponível em: < https://www.espressif.com/en/products/socs/esp8266>. Acesso em: 19 de agosto de 2024.

FERREIRA, João Alexandre Oliveira. Interface homem-máquina para domótica baseado em tecnologias WEB. Faculdade de Engenharia da Universidade do Porto, Porto – Portugal, 2008.

GUERRA, Hithallo Nicson Silva. Automação residencial usando software Openhab. TCC (Graduação) – Curso de Engenharia de Controle e Automação, Instituto Federal de Goiás, Itumbiara, Goiás, 2021.

GUIMARÃES, Fábio. Módulo RF 433Mhz com Arduino. 2019. Disponível em: < https://mundoprojetado.com.br/modulo-rf-433mhz-com-arduino/>. Acesso em: 19 de agosto de 2024.

GUSE, Rosana. Como funciona um conversor de tensão DC-DC?. 2022. Disponível em: < https://www.makerhero.com/blog/como-funciona-um-conversor-de-tensao-dc-dc/>. Acesso em: 19 de agosto de 2024.

JAFFEY, T. MQTT and CoAP, IoT protocols. Disponível em: < http://www.eclipse.org/community/eclipse_newsletter/2014/february/article2.php>. Acesso em: 06 de maio de 2023.

JUNIOR, Josihel de Andrade Silva. Sistema de Monitoramento e Acionamento de Cargas Elétricas Via Web. TCC (Graduação) – Curso de Engenharia Elétrica, Faculdade de Tecnologia e Ciências Sociais Aplicadas – Centro Universitário de Brasília, Brasília, Distrito Federal, 2019.

KERSCHBAUMER, Ricardo. Apostila de Microcontroladores. Instituto Federal de Educação, Ciência e Tecnologias Catarinense, Luzerna, Santa Catarina, 2018.

ANACOM. Treinamento Proteus. 2010. Disponível em: < https://www.eletrojota.com.br/Download/Aquivos%20para%20DOWLOANDS/MANUAIS%20Proteus_Anacon.pdf>. Acesso em: 19 de agosto de 2024.

MARIMOTO, Carlos E. Dicionário, Termos, Técnicos de Informática. 3. Ed. São Paulo: Ensino Profissional, 2016. Disponível em: <https://www.livrosgratis.com.br/ler-livro-online-25213/dicionario-de-termos-tecnicos-de-informatica---3a-edicao>. Acesso: 27 de março de 2023.

MAKERHERO. Placa Mega 2560 Pro Mini. Disponível em: <https://www.makerhero.com/produto/placa-mega-2560-pro-mini/>. Acesso em: 19 de agosto de 2024.

MCROBERTS, Michael. Arduino Básico. São Paulo: Novatec Editora, 2011. 465 p.

MQTT.ORG. Frequently Asked Questions. MQTT.ORG, 2014. Disponivel em: < https://mqtt.org/faq/>. Acesso em: 06 de maio de 2023.

NUNES, Renato Jorge Caleira. Arquitetura de um Controlador Domótico Flexível de Baixo Custo. INESC-ID. Lisboa – Portugal, 2002.

OLIVEIRA, Ricardo Rodrigues. Uso do Microcontrolador ESP8266 para Automação Residencial. 2017. TCC (Graduação) – Curso de Engenharia de Controle e Automação, Escola Politécnica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2017.

OLIVEIRA, Isabella Ferreira. Desenvolvimento de um sistema de sutomação residencial baseado em IOT para controle e monitoramento de dispositivos elétricos. TCC (Graduação) – Curso de Engenharia de Controle e Automação, Universidade Federal de Ouro Preto, Ouro Preto, Minas Gerais, 2019.

PACIEVITCH, Thais. Tecnologia da informação e comunicação. 2014. Disponível: < https://www.infoescola.com/informatica/tecnologia-da-informacao-e-comunicacao/ >. Acesso em: 28 de março de 2023.

PAZ, Humberto Pereira. Implementação de um Conversor DC-DC para Máxima Estração de Potência de um Gerador Termoelétrico. 2018. TCC(Graduação) – Curso de Engenharia Eletrônica e de Computação, Escola Politécnica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2018.

PESSÔA, Marcelo S. de P.; Spinola, Mauro de M. Introdução à automação para cursos de Engenharia e Gestão. Rio de Janeiro: Elsevier, 2014.

PINHEIRO, José Maurício Santos. Falando de Automação Predial. 2004. Disponível em: < https://www.projetoderedes.com.br/artigos/artigo_falando_de_automacao_predial.php>. Acesso em 06 de maio de 2023.

QUINDERÉ, P. R. F. Casa Inteligente – Um Protótipo de Sistema de Automação Residencial de Baixo Custo. Monografia – Faculdade Farias Brito, Fortaleza, CE. 2009.

STATISTA. Smart Home Brazil. 2020. Disponível em: <https://www.statista.com/out-look/dmo/smart-home/brazil#smart-homes>. Acesso em: 19 de agosto de 2024.

STEVEN, S. L; FARINELLI, F. A. Domótica: automação residencial e casas inteligentes com Arduino e ESP8266. São Paulo: Érica, 2019.

SANTOS, J. W. Sistema de Automação Residencial de Baixo Custo Controlado pelo Microcontrolador ESP32 e Monitorado via Smartphone. Monografia – Universidade Tecnológica Federal do Parana, Ponta Grossa, PR. 2019.

TAKIUCHI, Marcelo; MELO, Érica; TONIDANDEL, Flávio. Domótica Inteligente: Automação Baseada em Comportamento. Centro Universitario da FEI – São Bernado do Campo – SP, 2004.

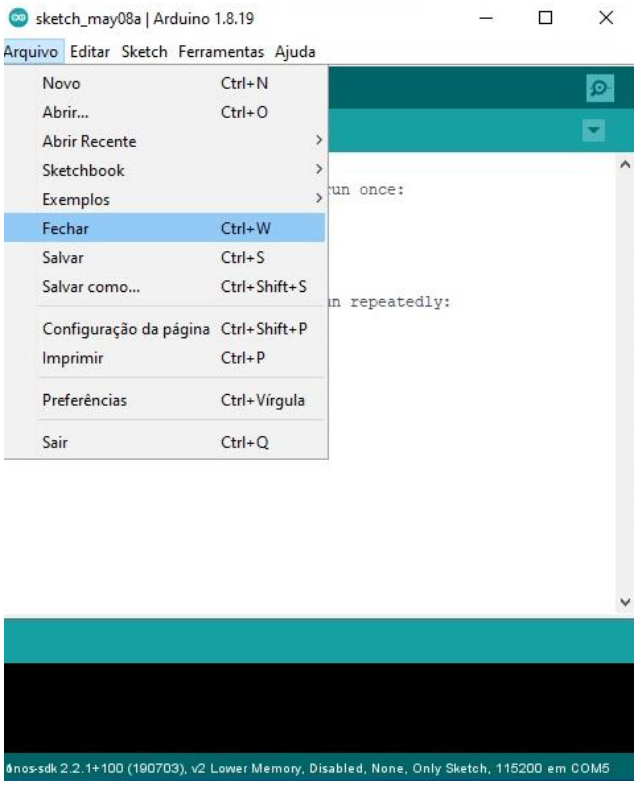
TEIXEIRA, Mário Antonio Meireles. Suporte a serviços diferenciados em servidores web: modelos e algoritmos. 2004. Tese (Doutorado) – Instituto de Ciências Matemáticas e de Computação, São Carlos, São Paulo, 2004.

TERRA, Marcos Damont. Desenvolvimento de calibrador de processor de baixo custo para padrão de trabalho em laboratórios de chão de fábrica. 2015. 144 f. TCC (Graduação) – Curso de Engenharia Elétrica, Doctum, Caratinga – MG, 2015.

APÊNCICE A – INSTALAÇÃO E CONFIGURAÇÕES PARA A UTILIZAÇÃO DAS PLACAS ESP NA IDE DO ARDUINO

Abre-se a janela especificada como “Arquivos” e selecione a opção preferências, como demostrado na Figura A.01.

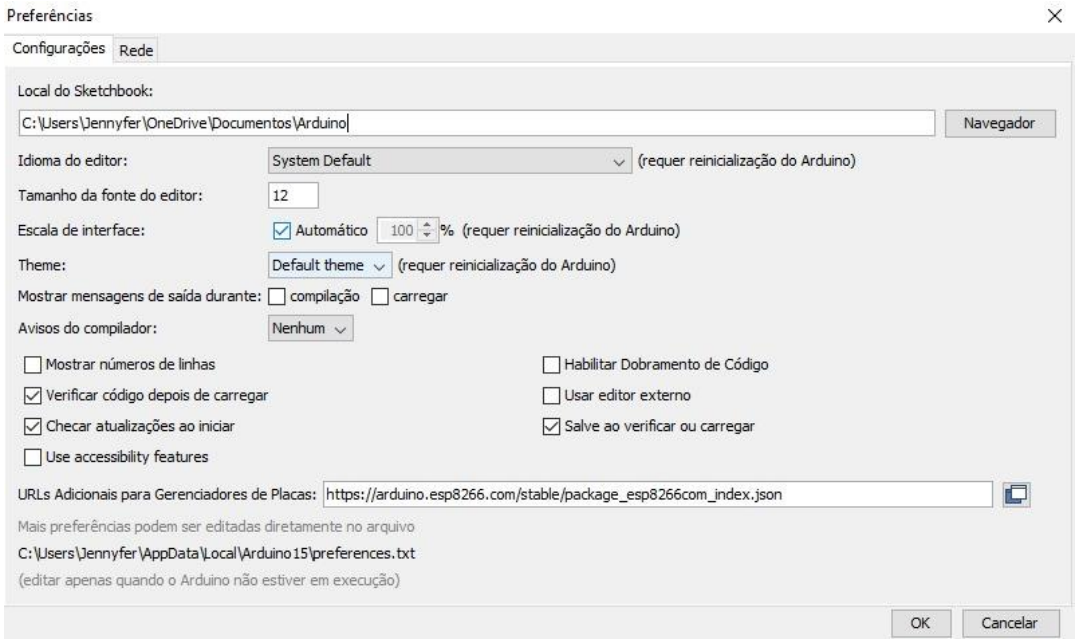
Figura A.01 – Janela Arquivos da IDE Arduino.



Fonte: Elaborado pelo próprio autor.

No campo gerenciamento de placa adicionais, insira a seguinte URL: <http://arduino.esp8266.com/stable/package_esp8266com_index.json>. A Figura A.02 apresenta a inclusão dessa URL.

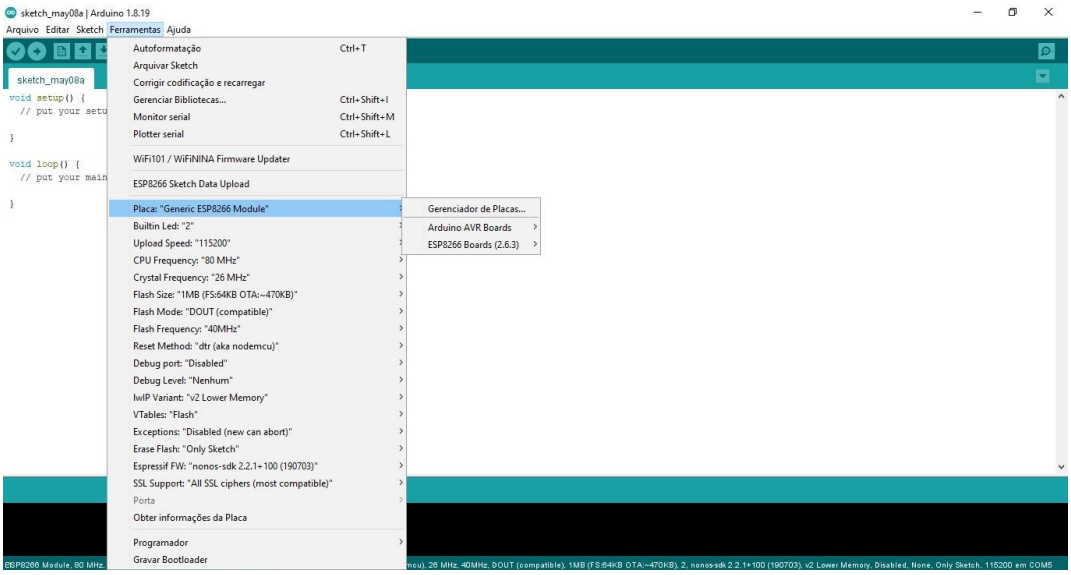
Figura A.02 – Janela preferências da IDE Arduino.



Fonte: Elaborado pelo próprio autor.

Após a adição da URL, abra a janela ferramentas e selecione a opção gerenciador de placas, como mostra a Figura A.03.

Figura A.03 – Janela Ferramentas da IDE Arduino.

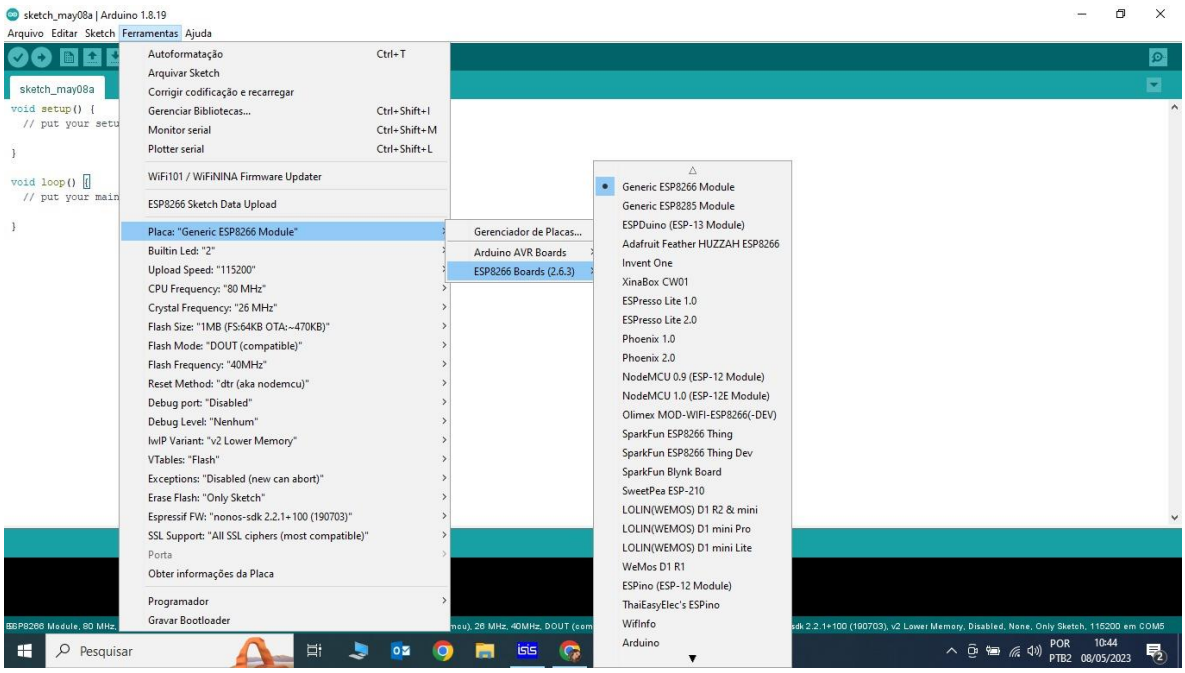


Fonte: Elaborado pelo próprio autor.

Após a abertura da janela do gerenciados de placas, procure o item ESP8266 by ESP8266 Community e selecione a opção instalar.

Ao finalizar a instalação dos modelos de placa ESP8266, elas ficaram disponíveis para utilização na aba ferramentas, como mostrado na Figura A.04.

Figura A.04 – Janela gerenciador de placas da IDE Arduino.



Fonte: Elaborado pelo próprio autor.

APÊNCICE B – PROGRAMAÇÃO ESP8266

```
/*********************************************************************************
                              SOFTWARE PARA O ESP8266
  INSTITUTO FEDERAL DE GOIÁS - CAMPUS ITUMBIARA
 PROJETO DESENVOLVIDO PARA O TRABALHO DE CONCLUSÃO DE CURSO
 ALUNA: JENNYFER HART
*********************************************************************************/

#include <ESP8266WiFi.h>
#include <WebServer.h>
#include <Preferences.h>

Preferences nvs;
WebServer server(80);

char serial_number[64];
char active_time[32];
const char* app_version =  "1.0.1";

int status_inputs[32] = {0};
int status_outputs[32] = {0};
int status_all_outputs = 0;

TaskHandle_t http_task_handle = NULL;
void http_task();

void setup() {
  Serial.begin(115200);
  nvs.begin("auto_res", false);
```

```
  String temp;
  String nvs_name;

  if(nvs.isKey("nvs_init") == false){
    for(int i = 1; i<=32; i++){
      nvs_name = "input_" + String(i);
      nvs.putString(nvs_name.c_str(), String(i));
    }
    for(int i = 1; i<=32; i++){
      nvs_name = "output_" + String(i);
      nvs.putString(nvs_name.c_str(), String(i));
    }
    nvs.putInt("nvs_init",1);
  }

  strncpy(serial_number, "Auto_RES_", sizeof(serial_number));

  uint8_t mac[6];
  esp_read_mac(mac, ESP_MAC_WIFI_STA);
  char wifi_sta_mac[18] = {0};

  sprintf(wifi_sta_mac, "%02X%02X%02X%02X%02X%02X", mac[0], mac[1], mac[2], mac[3],
mac[4], mac[5]);
  strcat(serial_number, wifi_sta_mac);

  if(nvs.isKey("ssid") == false){
    WiFi.mode(WIFI_AP_STA);
    WiFi.softAP(serial_number, "12345678");
  }
  else{
    String ssid = nvs.getString("ssid");
```

```
  String password = nvs.getString("password");

  WiFi.mode(WIFI_AP_STA);
  WiFi.softAP(serial_number, "12345678");
  WiFi.begin(ssid, password);

  int i = 0;
  while (WiFi.status() != WL_CONNECTED) {
     delay(500);
     i++;
     if(i >=20){
       break;
     }
  }
}
server.on("/", handle_on_connect);
server.on("/?", handle_on_connect);
server.on("/outputs", handle_outputs);
server.on("/set_output", handle_set_output);
server.on("/set_all_outputs", handle_set_all_outputs);
server.on("/inputs", handle_inputs);
server.on("/names", handle_names);
server.on("/names?", handle_names);
server.on("/set_names", handle_set_names);
server.on("/wifi", handle_wifi);
server.on("/wifi?", handle_wifi);
server.on("/set_wifi", handle_set_wifi);
server.on("/set_wifi?", handle_set_wifi);
server.on("/restart", handle_restart);
server.on("/restart", handle_restart);
server.onNotFound(handle_not_found);
```

```
  server.begin();
  xTaskCreatePinnedToCore(http_task, "http_task", 8192, NULL, 10, &http_task_handle,
tskNO_AFFINITY);
}

void http_task(void* parameters){
  while(1){
    server.handleClient();
  }
}

void loop() {
  server.handleClient();
}

void handle_on_connect() {
  seconds_to_time(esp_timer_get_time() / 1000000, active_time);

  String content = html_header();
  content += "<table align=center>";
  content += insert_table_line("Nome do dispositivo", serial_number);
  content += insert_table_line("Versão do Firmware ", app_version);
  content += insert_table_line("Tempo ativo", active_time);
  content += "</table>";
  content += "<br>";
  content += "<br>";
  content += "<center>";
  content += insert_form_line_get("outputs", "CONFIG. SAIDAS");
  content += insert_form_line_get("inputs", "CONFIG. ENTRADAS");
  content += insert_form_line_get("names", "CONFIG. NOMES");
```

```
  content += insert_form_line_get("wifi", "CONFIG. WIFI");
  content += insert_form_line_get("restart", "REINICIAR SISTEMA");
  content += "</center></body></html>";
  server.send(200, "text/html", content);
}

void handle_outputs() {
  Serial.println("STATUS+OUTPUT?");

  int64_t start = esp_timer_get_time() / 1000000;
  int64_t end = esp_timer_get_time() / 1000000;
  int uart_status = 0;
  while(end - start < 10 ){

    if (Serial.available())
    {
      uart_status = 1;
      break;
    }
    else{
      uart_status = 0;
    }
    end = esp_timer_get_time() / 1000000;
  }

  if(uart_status == 1){
    String payload = Serial.readString();
    if(payload.indexOf("RESP+STATUS+OUTPUT=")  == 0){
      payload.remove(0,strlen("RESP+STATUS+OUTPUT="));
      payload.remove(payload.length() - 1,1);
```

```
char values[128];
strcpy(values,payload.c_str());

char *outputs = NULL;
int index = 0;
outputs = strtok(values, ",");  // delimiter
while (outputs != NULL)
{
  status_outputs[index] = atoi(outputs);
  index++;
  outputs = strtok(NULL, ",");
}
String content = html_header();

content += "<center>";
content += "<b>Saidas</b>";
content += "<br><br>";

String name_1;
String name_2;
String name_3;
String name_4;
String nvs_name;

int i = 1;

while(i <= 32){
  nvs_name = "output_" + String(i);
  name_1 = nvs.getString(nvs_name.c_str());

  nvs_name = "output_" + String(i + 1);
```

```
      name_2 = nvs.getString(nvs_name.c_str());

      nvs_name = "output_" + String(i + 2);
      name_3 = nvs.getString(nvs_name.c_str());

      nvs_name = "output_" + String(i + 3);
      name_4 = nvs.getString(nvs_name.c_str());

      content += insert_four_buttons("set_output", name_1 , name_2, name_3, name_4, i - 1, i , i
+ 1, i + 2, status_outputs[i - 1],status_outputs[i], status_outputs[ i + 1], status_outputs[i + 2]);
      i = i + 4;
    }

    content += "<br>";
    content += "<br>";
    content += insert_button("set_all_outputs","Todas as saidas", status_all_outputs);
    content += "<br>";
    content += "<br>";
    content += insert_form_line_get("/", "VOLTAR");
    content += "<br><br>";
    content += "</center></body></html>";
    server.send(200, "text/html", content);
  }

  else{
    String content = html_header();

    content += "<center>";
    content += "<b>Erro ao carregar dados das saidas! </b>";
    content += "<br>";
    content += "<br>";
```

```
      content += insert_form_line_get("/outputs", "VOLTAR");
      content += insert_form_line_get("/", "INICIO");
      content += "</center></body></html>";
      server.send(200, "text/html", content);
    }
  }
}

void handle_set_output() {
  String status;
  String pin;
  status = server.arg("status");
  pin = server.arg("pin");

  if(status == "on"){
    String content = "SET+OUTPUT=" + pin + ",1?";
    Serial.println(content);
  }
  if(status == "off"){
    String content = "SET+OUTPUT=" + pin + ",0?";
    Serial.println(content);
  }

  int64_t start = esp_timer_get_time() / 1000000;
  int64_t end = esp_timer_get_time() / 1000000;
  int uart_status = 0;

  while(end - start < 10 ){
    if (Serial.available())
    {
      uart_status = 1;
```

```
    break;
  }
  else{
    uart_status = 0;
  }
  end = esp_timer_get_time() / 1000000;
}

if(uart_status == 1){
  String payload = Serial.readString();

  if(payload.indexOf("RESP+SET+OUTPUT=")  == 0){
    String content = html_header();

    content += "<center>";
    content += "<b>Saida alterada com sucesso! </b>";
    content += "<br>";
    content += "<br>";
    content += insert_form_line_get("/outputs", "VOLTAR");
    content += insert_form_line_get("/", "INICIO");
    content += "</center></body></html>";
    server.send(200, "text/html", content);
  }
  else{
    String content = html_header();

    content += "<center>";
    content += "<b>Erro ao alterar saida! </b>";
    content += "<br>";
    content += "<br>";
    content += insert_form_line_get("/outputs", "VOLTAR");
```

```
    content += insert_form_line_get("/", "INICIO");
    content += "</center></body></html>";
    server.send(200, "text/html", content);
   }

  }else{
    String content = html_header();

    content += "<center>";
    content += "<b>Erro ao alterar saida! </b>";
    content += "<br>";
    content += "<br>";
    content += insert_form_line_get("/outputs", "VOLTAR");
    content += insert_form_line_get("/", "INICIO");
    content += "</center></body></html>";
    server.send(200, "text/html", content);
  }
}

void handle_set_all_outputs() {
  String status;

  status = server.arg("status");
  if(status == "on"){
    String content = "SET+OUTPUT+ALLL=1?";
    Serial.println(content);
    status_all_outputs = 1;
  }

  if(status == "off"){
    String content = "SET+OUTPUT+ALL=0?";
```

```
  Serial.println(content);
  status_all_outputs = 0;
}

int64_t start = esp_timer_get_time() / 1000000;
int64_t end = esp_timer_get_time() / 1000000;
int uart_status = 0;
while(end - start < 10 ){

  if (Serial.available())
  {
    uart_status = 1;
    break;
  }
  else{
    uart_status = 0;
  }
  end = esp_timer_get_time() / 1000000;
}

if(uart_status == 1){
  String payload = Serial.readString();

  if(payload.indexOf("RESP+SET+OUTPUT+ALL=")  == 0){
    String content = html_header();

    content += "<center>";
    content += "<b>Saidas alterada com sucesso! </b>";
    content += "<br>";
    content += "<br>";
    content += insert_form_line_get("/outputs", "VOLTAR");
```

```
      content += insert_form_line_get("/", "INICIO");
      content += "</center></body></html>";
      server.send(200, "text/html", content);
    }
    else{
      String content = html_header();

      content += "<center>";
      content += "<b>Erro ao alterar saida! </b>";
      content += "<br>";
      content += "<br>";
      content += insert_form_line_get("/outputs", "VOLTAR");
      content += insert_form_line_get("/", "INICIO");
      content += "</center></body></html>";
      server.send(200, "text/html", content);
    }
  }
  else{
      String content = html_header();

      content += "<center>";
      content += "<b>Erro ao alterar saida! </b>";
      content += "<br>";
      content += "<br>";
      content += insert_form_line_get("/outputs", "VOLTAR");
      content += insert_form_line_get("/", "INICIO");
      content += "</center></body></html>";
      server.send(200, "text/html", content);
  }
}
```

```
void handle_inputs() {

  Serial.println("STATUS+INPUT?");

  int64_t start = esp_timer_get_time() / 1000000;
  int64_t end = esp_timer_get_time() / 1000000;
  int uart_status = 0;
  while(end - start < 10 ){

    if (Serial.available())
    {
      uart_status = 1;
      break;
    }
    else{
      uart_status = 0;
    }
    end = esp_timer_get_time() / 1000000;
  }

  if(uart_status == 1){
    String payload = Serial.readString();

    payload.remove(0,strlen("RESP+STATUS+INPUT="));
    payload.remove(payload.length() - 1,1);

    char values[128];
    strcpy(values,payload.c_str());

    char *inputs = NULL;
    int index = 0;
```

```
  inputs = strtok(values, ",");  // delimiter
  while (inputs != NULL)
  {
    status_inputs[index] = atoi(inputs);
    index++;
    inputs = strtok(NULL, ",");
  }
}

String content = html_header();

content += "<center>";
content += "<b>Entradas</b>";
content += "<br><br>";

String name_1;
String name_2;
String name_3;
String name_4;
String nvs_name;

int i = 1;
while(i <= 32){

  nvs_name = "input_" + String(i);
  name_1 = nvs.getString(nvs_name.c_str());

  nvs_name = "input_" + String(i + 1);
  name_2 = nvs.getString(nvs_name.c_str());

  nvs_name = "input_" + String(i + 2);
```

```
    name_3 = nvs.getString(nvs_name.c_str());

    nvs_name = "input_" + String(i + 3);
    name_4 = nvs.getString(nvs_name.c_str());

    content += insert_four_info_square(name_1, name_2, name_3, name_4, status_inputs[i - 1],
status_inputs[i], status_inputs[i + 1], status_inputs[i + 2]);
    i = i + 4;
  }

  content += "<br>";
  content += "<br>";
  content += insert_form_line_get("/", "VOLTAR");
  content += "<br>";
  content += "<br>";
  content += "<script>";
  content += "setTimeout(function() {window.location.reload(1);}, 5000);";
  content += "</script>";
  content += "</center></body></html>";
  server.send(200, "text/html", content);
}

void handle_names(){
  String content = html_header();

  content += "<center>";
  content += "<b>Nomes</b>";
  content += "<br> <form method='get' action='set_names'> <br> <br> <table align=center>";

  String temp;
  String nvs_name;
```

```
  for(int i = 1; i<=32; i++){
    nvs_name = "input_" + String(i);
    temp = nvs.getString(nvs_name.c_str());
    content += insert_form_label_line(nvs_name.c_str(), nvs_name.c_str(), temp);
  }

  for(int i = 1; i<=32; i++){
    nvs_name = "output_" + String(i);
    temp = nvs.getString(nvs_name.c_str());
    content += insert_form_label_line(nvs_name.c_str(), nvs_name.c_str(), temp);
  }

  content += "</table> <br> <li><input type='submit' class='button' value='SALVAR'></li>
</form>";
  content += insert_form_line_get("/", "INICIO");
  content += "<br>";
  content += "<br>";
  content += "</center></body></html>";
  server.send(200, "text/html", content);
}

void handle_set_names(){
  String http_get_arg;
  String nvs_name;

  for(int i = 1; i<=32; i++){
    nvs_name = "input_" + String(i);
    http_get_arg = server.arg(nvs_name);
    nvs.putString(nvs_name.c_str(),http_get_arg);
  }
```

```
  for(int i = 1; i<=32; i++){
    nvs_name = "output_" + String(i);
    http_get_arg = server.arg(nvs_name);
    nvs.putString(nvs_name.c_str(),http_get_arg);
  }

  String content = html_header();

  content += "<center>";
  content += "<b>Nomes salvos com sucesso! </b>";
  content += "<br>";
  content += "<br>";
  content += insert_form_line_get("/name", "VOLTAR");
  content += insert_form_line_get("/", "INICIO");
  content += "</center></body></html>";
  server.send(200, "text/html", content);
}

void handle_wifi(){
  WiFi.disconnect();

  char html[256] = {0};
  String content = html_header();

  content += "<center>";
  content += "<font face='verdana' size=3><b>Redes Pr&oacute;ximas</b></font><br><br>";
  content += "<table align=center>";

  int n = WiFi.scanNetworks();
```

```
  if (n == 0)
  {
     content += "<tr><td> Não há redes disponiveis </td></tr>";
  }
  else{
    content += "<tr> <th> nº </th> <th> Rede Wi-Fi </th> <th> dBm </th> <th> AuthMode </th>
</tr>";

    for (int c = 0; c < n; c++)
    {
       sprintf(html,
             "<tr><td> %i </td>"
             "<td onclick=\"handleInputSSID(event)\">%s</td>"
             "<td> %li </td>"
             "<td> %s </td></tr>",
             c, WiFi.SSID(c).c_str(), WiFi.RSSI(c), (WiFi.encryptionType(c) ==
WIFI_AUTH_OPEN)?" ":"*");

       content += html;
    }
  }
  content += "</table>";
  content += "<br> <form method='get' action='set_wifi'> <br> <br> <table align=center>";
  content += insert_form_label_line_with_id("Rede Wi-Fi","ssid","ssid",32);
  content += insert_form_label_line_with_id("SENHA","password","password",64);
  content += "<script>";
  content += "function handleInputSSID(event) {";
  content += "const ssid = event.target.innerHTML;";
  content += "const inputSSID = document.getElementById(\"ssid\");";
  content += "const inputPassword = document.getElementById(\"password\");";
  content += "inputSSID.setAttribute(\"value\",ssid);";
```

```
  content += "inputPassword.setAttribute(\"value\",\"\");";
  content += "inputPassword.focus();";
  content +=  "}";
  content += "</script>";
  content += "</table> <br> <li><input type='submit' class='button' value='CONFIRMAR'></li>
</form>";
  content += insert_form_line_get("/", "INICIO");
  content += "<br>";
  content += "<br>";
  content += "</center></body></html>";
  server.send(200, "text/html", content);
 }

 void handle_set_wifi(){
  String ssid;
  String password;

  ssid = server.arg("ssid");
  password = server.arg("password");

  nvs.putString("ssid",ssid);
  nvs.putString("password",password);

  String content = html_header();

  content += "<center>";
  content += "<b>SSID E Senha salvos com sucesso! </b>";
  content += "<br>";
  content += "<br>";
  content += insert_form_line_get("/", "INICIO");
  content += "</center></body></html>";
```

```
  server.send(200, "text/html", content);
}

void handle_restart(){
  String content = html_header();

  content += "<center>";
  content += "<b>Reiniciando!</b>";
  content += "<br>";
  content += "<br>";
  content += "</center></body></html>";
  server.send(200, "text/html", content);

  delay(2000);

  ESP.restart();
}

void handle_not_found(){
  server.send(404, "text/plain", "Not found");
}

String html_header(){

  String header =" <!DOCTYPE html>";
  header += "<html lang='pt'>";
  header += "<head> <meta charset='utf-8'><meta name='viewport' content='width=device-width,
initial-scale=1' ><title> Auto RES </title>";
  header += "<style type='text/css'>";
  // fundo da pag:
```

```
  header += "@media screen and (min-width: 499px) { html { background: rgba(60,57,80,0.6);
}";
  header += "body {max-width: 499px;min-height: 630px;margin: auto !important;-webkit-
transform: translate3d(0, 0, 0);transform: translate3d(0, 0, 0);}}";
  // tabela cor fonte texto, cor de fundo corpo central //
  header += "body {padding: 0;margin: 0;font-family: 'Roboto Condensed', sans-serif;font-size:
14px;font-weight: 500;line-height: 1;color: #3c3950;background-color: #fff;}";
  //
  header += "#app {max-width: 499px;margin: 0 auto;}";
  header += "li {display: inline-block;}";
  header += ".informations-project {padding: 30px 0;text-align: center;min-height: 115px;}";
  header += ".informations-project h2 {margin-bottom: 5px;font-size: 17px;line-height:
19px;font-weight: 600;text-transform: uppercase;}";
  // botão:
  header += ".button {display: block;width: 200px;padding: 10px 0;margin: 5px auto;text-align:
center;font-size: 17px;border: 2px solid  #00b7ff;background-color: #fff;color: #767876;font-
weight: bold;}";
  header += ".button-on {background-color: #000000;}";
  header += ".button-off {background-color: #616161;}";
  //botões output
  header +=".button-outputs {display: inline-block;width: 25px;background-color:
#000000;border: none;color: white;padding: 10px 20px;text-align: center;text-decoration:
none;font-size: 15px;margin: 4px 2px;cursor: pointer;border-radius: 4px;}\n";
  header +=".button-outputs-on {background-color: #000000;}\n";
  header +=".button-outputs-off {background-color: #616161;}\n";

  //botões inputs
  header +=".button-inputs {display: inline-block;width: 25px;background-color: #000000;border:
none;color: white;padding: 10px 20px;text-align: center;text-decoration: none;font-size:
15px;margin: 4px 2px;cursor: pointer;border-radius: 4px;}\n";
  header +=".button-inputs-on {background-color: #f50505;}\n";
```

```
  header +=".button-inputs-off {background-color: #80ff80;}\n";
  // botão feedback:
  header += ".button:hover {display: block;width: 200px;padding: 10px 0;margin: 5px auto;text-
align: center;font-size: 17px;border: 2px solid  #00b7ff;background-color: #767876;color: #fff;
font-weight: bold;}";
  //
  header += "table, td, th { border: 1px solid #6483c1; text-align: left;}";
  header += "table {border-collapse: collapse;width: 90%; aling:center;}";
  header += "th, td {padding: 15px;}";
  header += ".td_negrito { font-weight: bold;}";
  // cor linhas pares tabela
  header += "tr:nth-child(even) {background-color: #f2f2f2;}";
  header +="div > div {display: inline-block;}";
  header += "</style>";

  header += "</head>";
  header += "<center>";
  header += "<body data-src="" class='app-project'>";
  header += "<div class='informations-project'>";
  header +=" <img
src=\"
KuAAAAAXNSR0IArs4c6QAAAAxQTFRF////wMDAgICAAAAAtP/LyAAADU9JREFUeNrt
2m+MG2V+wPHvzBCPuo92fZUAXbN//KYvlvlDpEbXLqduXsAp3IurdE1CvF
f5igkk+5CqQiI6MCdopaPljHQ0YVfJGrTinE3W9hGpkUClk6ISCQT3JPESsgvZPWqOsZP1dG
PvH5NkbG+ub6ruR7Jl2frq98wzmpk3Zt26devWrftfWrftfZ7zGhiS3KqDEaNlb4BYZWd0W8Ept7JLc
is5kSmRl7BijBdbOHPMul2IOZKXey9pFx/9rh5ebeXFXzvO8cpK16n+UwKWHFtuZh78cjGVZ
q3u0kZutjD/4iDd6HLFNsSYbZXTruPeqtFXfhnGvYJzPsyaT+ZxXehSMc/PwSCUZlayFeWa2X
JEDwIgC+05vXEUkrQvurjw8lkwDUQmWinqFNzWtG5kpHTcm82D8zgHD2VvKFRO0rK1kz
SYJOWCrDHCiED3kHaNVRk6FXOjIQoQeMLIY5d/mFC2yXyKnICwKRhHbIXYUQm6gRIt
6wvYsMEHsJQUTnQUBhkfLC+hK7/1zMBzMChApYym4z+mboDXiE7cDLIU5W8DIORgO3
OV29NEa29NdICAkY8+PJzOQhnD0gWdpyWBwzjsqCUMG0ysQgV7osr/8nFaY7oHMnTlv5v
```

ig7bBx5hIBhw7oMLwirRA7v9AM3zl27dIfq7w4Pv3g8aGHh3aN5youreiOfihJYanA9Hh50luU
W3xVxl4Qx2xFc/fMHgWHDsQ/TkogkL1fP6ixpHn24zTN5f5SY+YJk80mAdhYyjCAqXjz6gD
NuVGFpUjblSQ1uz3ZjZEnkvspTdnOAbAx369Or4lpAQ728Vmaim4tQxui/mwZ2e/lSWFcjSqa
yVofQPud2XHFKlHWDMMBO08T1keiDSKVzOPUMcol2Q/3bS3ShAhvbmNXJXnYoV4kWu
qFx/JTNNH+J64YL6fNy1DPPrR7AYb1txWNiRNl7xIpUYB6xqSKXSKeuMuhMdObeRkjf8Pz
IqrF+KXhSFeaxowrsgNTZbhOoGhvHS9tf6afxswP6MDe7nC9/aYi581O0FBA7KGdthHJ9cRTe
YY3ekkaCh3rp53gZW5gFhzizCkaGlhI0U0kwY0qKfp5LE9Dr1fydJmu5kZPFOnnLi1p5PCIpCv
0RZ4b/VGGftonizTy+Ra4KzPGTdgh2Y8Y2UMDhjsF0ec/4ybM/9D9WLnv0oCpszBlznMzM/s
HMCof0ID11CzC9bnKn9iegZMnaEDcf1pm+wU3FXoqA/ujEn/RB/5+u+7vZoVZYIXIRyXZY
KP+CWFWGE6wwr5avzmbCvzGVvi72CM0Py6yQjzKqnTvXuP0/Q6+jLe3ZPMc1iwxfmm7p
mbZQNg+f/DgPL7MZ3JJh6zDEqvyvd+OuSyLdJG7/4yLL2OuhGZSscT+hXuhkmBZ6PuI5Lm
P8BcrkjIuSJaYC4kLL7ksC36OmCq+jb9gmrT1DqvHP3DhYIplYoq2Tvc0/npSDAfrF5i4oFhhP
Uo72Vn8hR36//SzuvmJ+vnmc3RxXwVft23OM/BFkSUbrIWBC4fc21lifKjCfEvjKzqp2PTWPE
uiT788cOG5xOrNdFT3EmnQ/90VRcR0WLLvvWvzMyfUyhcD/XQ36N/w4F4rT83gvjPu4vHrr
1b6Ld3DtKfxdXgvXLUlVaYbVenF3lldf8hO0zaZxM9oBOvfBTX20SDVvs9MURM0NXYujZ
+pqAp2BFd6Ueu7hUuNoIiV7cdPZks+0RZhef2C9L4r1/onqbHkVcwrvfiZGHhFt21hybwgPTJ+r
c+s3B7fwTx/BD/p/ieUmKLqtmv96679idNt6uX+P0/l7XOf4mc+HsH+nKqoWuxzz6dyTrfhUG
OeV/Md2sWHUUxPYf0zVfuceWFUxEKnXuw3LvX/Le/tnZ/AhznxxYdY36dqcrEXOht6vrTYj
ygAzCuyI56O48P6ofeAEj+i6j41L/ZuK5hu9KjhvJmv9eNYejiBD/su18wHj1AVYl64UUnsOwu
Gs3QJGO9i6rh/v2kevWmCqiDz0YMu2IUDW52orPXvYO+JT/n2MU36J8Xl/tzxUBLIinNODz
UfyjaROYYP00sx8Mr8cn9FTwF0nsmu9HOq23Rn8BNM0ees9BWhAYxSSC/3U84AJz/ET0eK
oNJL/eKBSwAWt2G5z6TSjJxu1Au53EcPFqkKFGM7l/u4w8jlhv3KfDeqqDmwrUJNYlOe3Uf
wYYyniPx6qQ+dKbFE6Cw18ZDksMSHfW3/zy31WaFZYkx1Okt9kAZ9W1STfmO+fmbN4k
WwPL9hf7c2nLtrfWxbgRXm8l5M3KMY8+3F37qmik4AkKkVd7wKQOw7DgCZHzuMufiwX
5+1ZTAOwPkKgJHzCgChaQ2AqzW5y/iw9PmPEQMA5D4GiE7vyEqgJ+cAcCkfJ/sMPkytTyJ
6AfjWswBTxmB167u3SABjTvWapbfxYejUfsQRAKxrva3HvWQG6EkAYP6GblH8CD86PSY
7Pl3txfbyQ6W+uv5Z2vrSn+Fn/o238u0nV3uicjD6Z3X9v2DpdMb//ntCpruerOsz0cq2/GpvZbHz
esK3/zRLOLy5rv/D8piWdf0kPVqnffuPrm1/X11vOzLDam+fJfG6o337t09iue11fUAsfLO+P23o
e87496cWoNhR13NgXNX1Ytt2FTql8TM7Awkh6/ofml/rrZ2I02n87JqFrFB1vVmkru8wJ7Df9e/
5SnL4UL6uFyfr+z7TQXyWwtdjijNndX2/p76/Vyg6XI2vfcrIn0jXr/9H9f3JnjwRL4+vA3kzHx2o
60nW927cYdMZfAU8x1QiXNcbuq43LzkOd+fx1TaiLWn91K+3YkrzmGrQ313ZhjXj14stpBiT

+BID5RmsnF/fniDNXBJf1pGJvZi/UKs99X3fPMOm5+LLdAeC2nw6D1jPjVaVq+/Hqv235+m
PXgzjLxPumjIPOoB1yKuzUO0PpNl05Wwv/u7t6BU/U2nAUtSr9m6KA9ueieOvW8SNyXwC
MGdG603Ubs+Bq6JN4y+4bZ5QPlJ78tTTgKX03u92BPON+lMasxBS3Izg/UJHOCjx13Z+HuN
yh8PNRLgs2xLdNCDucTHOfpDmZqbMInYxQQPWI59hOCf2cBOmG81jaZcGzHdmwAk9y0
2In5QV9nsTNGCUv5I45kXFjfpOHFQEPx6mkW+8mceh4nCjeMlURNwUDfVpNIeL3MC46Fi
Kx4oODYUm0Ayf5Ab2nLQkF8dVk74kNb0hyfWCs1jSfLeNxnK5iqbL0lxvy19gSXsPjRlnozu8
F9qNEtcxil3YdGRo4sIh9a+5nzOi+DqR78V+aNylicffclKBXDJS4OtycnjDeKVSoYkeZyKFXfl
5BOrZC7yZm5Xue5LGbL1ZY/6b97SiXiS+sfKCsM6/QxNGMeRgOBvLReqYesZzEOKBozQT
ERpSnKhQp/OlygMK0WdP0EzoZy6kCe8usCpXkTaITNChGXH8MvTTYeSSq+PLe2gDW0do
yn7kK0UvtgpVFDUbKzJNN4R+7dKUeaqkCWM45iflpfxKwXQIQ277PM396tgA7ZAm0TkNE
Cg/LG3JAObcpENzEfEyO0fHvKrc6Kg3PUUPDCNi7ylaYF6Z82Ze2zm048uh3PSYNzrnedlXZ
Yonc4doiVfaJrEl+wkc35sksmFH2atMG5+cW6AVZmnKUph5hGLExShi5nlobnZO/oBW2N2hg
wo0ht49HUsKhZCYaqRCa+xD5pd5iMOWEkZ2P/SAJbMHaI3xOzXiQJiAp0C4MABC6F8pW
hPVogiC8ec1hGJJ0tB2wDzVah8omRWJ9VKBzZAxsmYegkWxl1aFCpNF7BIIFXDovCgxcirq
0ip7rhJTe1Ng6h5gRNKZNbKSVhnncu9fKfQDj2eASNEsHc9pWvf4KzlPhYGgA0xEYv/kjUla1
+mE5kodQMgBHFHOZvpYA4PgmZg3CJkJMP8h5xUiRdYknDFylV+aTh/Gbu9SSAnFmvT1S
DGTm02KUc8rsc9hjcTsNJUdOc8rv5iVAb3mnt3eovJrY954aSg2plgzMR1/WEN0xvO8EmtnHE
9bx5IbK6RjilvRqUWgUtnGPpdbZM9WKqNsktyqXXLjkfV/Ea5b9/+T8ddQ7w5JM38MsZQxZ
cwcG1Pc8VASIDYqMSbHFC9O0EwChCRhJPtMZUiqgbhfQ8JUpuqlmUdBwO2G7EMGoA9
AkIY40pa30YSxc6lnse+GYK2fgDjYkqb9dlntoQ/CIOTqfCzVvLfVaj8AtgIEyWpvapoJ7M4j5G
pvVfutEuIbYIpm/mCHXu0TYOUBMQbsOwYhRRO3k77JfEvV5pvF5n1ite9b7g0NcYAETQw
SRki+UY0jYMu68wd9zfse7OU+BG0Ags3V/ja6aUIhsBSD1V4ovlnrQxBncANhGjMVQhkaWe
1NzQ9qve0QJ2nLXhrrLNCp+ZshMLJJ2DUkAWJJo2RMjjqB6RSNbfgr7hi89o6xmFY/AEOD3
MHQkGRIsmY5T/L7uGOI/zt2jS55cejrVn54gUaGmhpcfyqtW7du3bp1v6//AciCyBwhI50tAAA
AAElFTkSuQmCC\" width=\"150\" height=\"150\" alt=\"base64 test\">";

```
  header += "<div class='descriptions-project'>";

  header += "</center>";
  return header;
}
```

```
String insert_table_line(String text, String value){
  return "<tr><td class='td_negrito' style= 'width:52%'>"+ text + "</td><td><center>" + value +
"</center></td></tr>";
}

String insert_form_line_post(String action, String value){
    String layout = "<br>";
    layout += "<form method='post' action='"+ action +"'>";
    layout += "<li><input type='submit' class='button' value='"+ value +"'></li>";
    layout += "</form>";
    return layout;
}

String insert_form_line_get(String action, String value){
    String layout = "<br>";
    layout += "<form method='get' action='"+ action +"'>";
    layout += "<li><input type='submit' class='button' value='"+ value +"'></li>";
    layout += "</form>";
    return layout;
}

String insert_form_label_line(String label, String text_input, String value){
    String layout = "<tr><td><label>" + label + ": </label></td>";
    layout += "<td><input type='text' name='" + text_input + "' value=" + value + "></td></tr>";
    return layout;
}

String insert_form_label_line_with_id(String label, String textInput, String id,  int lengthInput){
    String layout = "<tr><td><label>" + label + ": </label></td>";
    layout += "<td><input type='text' name='" + textInput + "' id='" + id + "' length=" +
String(lengthInput) + "></td></tr>";
```

```
   return layout;
}

String insert_four_buttons(String action, String value_1, String value_2, String value_3, String value_4, int name_1, int name_2, int name_3, int name_4, int status_1, int status_2, int status_3, int status_4){
  String content ="<div> <div> <div> <div>\n";

  if(status_1){
     content += "<p>";
     content += value_1;
     content += "</p><a class=\"button-outputs button-outputs-off\" href=" + action + "?status=off&pin=";
     content += name_1;
     content += ">ON</a>\n";
  }
  else{
     content += "<p>";
     content += value_1;
     content += "</p><a class=\"button-outputs button-outputs-on\" href=" + action + "?status=on&pin=";
     content += name_1;
     content += ">OFF</a>\n";
  }

  content +="</div> <div> <div> <div>\n";

  if(status_2){
     content += "<p>";
     content += value_2;
```

```
    content += "</p><a class=\"button-outputs button-outputs-off\" href=" + action +
"?status=off&pin=";
    content += name_2;
    content += ">ON</a>\n";
  }
  else{
    content += "<p>";
    content += value_2;
    content += "</p><a class=\"button-outputs button-outputs-on\" href=" + action +
"?status=on&pin=";
    content += name_2;
    content += ">OFF</a>\n";
  }

  content +="</div> </div> <div> <div>\n";

  if(status_3){
    content += "<p>";
    content += value_3;
    content += "</p><a class=\"button-outputs button-outputs-off\" href=" + action +
"?status=off&pin=";
    content += name_3;
    content += ">ON</a>\n";
  }
  else{
    content += "<p>";
    content += value_3;
    content += "</p><a class=\"button-outputs button-outputs-on\" href=" + action +
"?status=on&pin=";
    content += name_3;
    content += ">OFF</a>\n";
```

```
  }

  content +="</div> </div> </div> <div>\n";

  if(status_4){
     content += "<p>";
     content += value_4;
     content += "</p><a class=\"button-outputs button-outputs-off\" href=" + action +
"?status=off&pin=";
     content += name_4;
     content += ">ON</a>\n";
  }
  else{
     content += "<p>";
     content += value_4;
     content += "</p><a class=\"button-outputs button-outputs-on\" href=" + action +
"?status=on&pin=";
     content += name_4;
     content += ">OFF</a>\n";
  }

  content +="</div> </div> </div> </div>\n";

  return content;

}

String insert_button(String action, String value, int status){

  String content ="<div>\n";
```

```
  if(status){
     content += "<p>";
     content += value;
     content += "</p><a class=\"button-outputs button-outputs-off\" href=" + action +
"?status=off";
     content += ">ON</a>\n";
  }
  else{
     content += "<p>";
     content += value;
     content += "</p><a class=\"button-outputs button-outputs-on\" href=" + action +
"?status=on";
     content += ">OFF</a>\n";
  }

  content +="</div>\n";

  return content;
}

String insert_four_info_square(String value_1, String value_2, String value_3, String value_4, int
status_1, int status_2, int status_3, int status_4) {

  String content = "<div> <div> <div> <div>\n";

  if (status_1) {
    content += "<p>";
    content += value_1;
    content += "</p><a class=\"button-inputs button-inputs-off\"";
    content += ">ON</a>\n";
  } else {
```

```
  content += "<p>";
  content += value_1;
  content += "</p><a class=\"button-inputs button-inputs-on\"";
  content += ">OFF</a>\n";
}

content += "</div> <div> <div> <div>\n";

if (status_2) {
  content += "<p>";
  content += value_2;
  content += "</p><a class=\"button-inputs button-inputs-off\"";
  content += ">ON</a>\n";
} else {
  content += "<p>";
  content += value_2;
  content += "</p><a class=\"button-inputs button-inputs-on\"";
  content += ">OFF</a>\n";
}

content += "</div> </div> <div> <div>\n";

if (status_3) {
  content += "<p>";
  content += value_3;
  content += "</p><a class=\"button-inputs button-inputs-off\"";
  content += ">ON</a>\n";
} else {
  content += "<p>";
  content += value_3;
  content += "</p><a class=\"button-inputs button-inputs-on\"";
```

```
    content += ">OFF</a>\n";
  }
  content += "</div> </div> </div> <div>\n";

  if (status_4) {
    content += "<p>";
    content += value_4;
    content += "</p><a class=\"button-inputs button-inputs-off\"";
    content += ">ON</a>\n";
  } else {
    content += "<p>";
    content += value_4;
    content += "</p><a class=\"button-inputs button-inputs-on\"";
    content += ">OFF</a>\n";
  }
  content += "</div> </div> </div> </div>\n";
  return content;
}

void seconds_to_time(int32_t value, char *time)
{
    int32_t h, m, s, r;
    h = value / 3600;
    r = value % 3600;
    m = r / 60;
    s = r % 60;
    snprintf(time, 32, "%li h %li min %li seg", h, m, s);
}
```

APÊNCICE C – PROGRAMAÇÃO ARDUINO

```
/*********************************************************************
 *                    SOFTWARE PARA O ARDUINO MEGA PRO
 * INSTITUTO FEDERAL DE GOIÁS - CAMPUS ITUMBIARA
 * PROJETO DESENVOLVIDO PARA O TRABALHO DE CONCLUSÃO DE CURSO
 * Aluna: JENNYFER HART
*********************************************************************/

uint8_t gpio_inputs[32] = {
  37,
  38,
  39,
  40,
  41,
  42,
  43,
  44,
  45,
  46,
  47,
  48,
  49,
  50,
  51,
  52,
  54,
  96,
  56,
  94,
  58,
```

```
  92,
  60,
  90,
  62,
  88,
  64,
  86,
  66,
  84,
  68,
  82
};

uint8_t status_inputs[32] = {0};

uint8_t gpio_outputs[32] = {
  2,
  3,
  4,
  5,
  6,
  7,
  8,
  9,
  10,
  11,
  12,
  14,
  15,
  16,
  17,
```

```
  20,
  21,
  22,
  23,
  24,
  25,
  26,
  27,
  28,
  29,
  30,
  31,
  32,
  33,
  34,
  35,
  36
};

uint8_t status_outputs[32] = {0};

void setup() {

  Serial.begin(115200);
  Serial1.begin(9600);

  Serial.println("Iniciando");

  for(int i = 0; i < sizeof(gpio_inputs); i++){
    pinMode(gpio_inputs[i], INPUT);
  }
```

```
  for(int i = 0; i < sizeof(gpio_outputs); i++){
    pinMode(gpio_outputs[i], OUTPUT);
    digitalWrite(gpio_outputs[i],1);
  }
}

void loop() {

  if (Serial1.available() > 0) {
    String payload = Serial1.readString();

    if(payload.indexOf("STATUS+OUTPUT?") == 0){

      String resp = "RESP+STATUS+OUTPUT=";
      for(int i = 0; i < 32; i++){
        if(i != 31){
          resp += String(status_outputs[i]) + ",";
        }
        else{
          resp += String(status_outputs[i]) + "#";;
        }
      }
      Serial.println(resp);
      Serial1.println(resp);
    }
    else if(payload.indexOf("STATUS+INPUT?")  == 0){

      for(int i = 0; i < 32; i++){
        Serial.println(gpio_inputs[i]);
        status_inputs[i] = digitalRead(gpio_inputs[i]);
```

```
  }

  String resp = "RESP+STATUS+INPUT=";
  for(int i = 0; i < 32; i++){
    if(i != 31){
      resp += String(status_inputs[i]) + ",";
    }
    else{
      resp += String(status_inputs[i]) + "#";;
    }
  }

  Serial.println(resp);
  Serial1.println(resp);

}
else if(payload.indexOf("SET+OUTPUT")  == 0){
  if(payload.indexOf("+ALL") == 10){
    Serial.println(payload);

    int start = payload.indexOf('=');
    int end = payload.indexOf('?');

    String content = payload.substring(start + 1,end);

    Serial.println(content);

    int value = content.toInt();

    for(int i = 0; i < 32; i++){
      digitalWrite(gpio_outputs[i],!value);
```

```
    status_outputs[i] = value;
   }

   String resp = "RESP+SET+OUTPUT+ALL=" + String(value) + "#";

   Serial.println(resp);
   Serial1.println(resp);
 }
 else{
   Serial.println(payload);

   int start = payload.indexOf('=');
   int end = payload.indexOf(',');

   String content = payload.substring(start + 1,end);

   Serial.println(content);

   int port = content.toInt();

   start = payload.indexOf(',');
   end = payload.indexOf('?');

   content = payload.substring(start + 1,end);

   Serial.println(content);

   int value = content.toInt();

   String resp = "RESP+SET+OUTPUT=" + String(port) + "," + String(value) + "#";
```

```
        digitalWrite(gpio_outputs[port],!value);

        status_outputs[port] = value;

        Serial.println(resp);
        Serial1.println(resp);
      }
    }
  }
}
```