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

LUCAS LEITE COSTA FRANCO

AUTOMAÇÃO RESIDENCIAL: SISTEMA DE ILUMINAÇÃO INTELIGENTE CONTROLADO POR DISPOSITIVO MÓVEL

JATAÍ - GO
2023

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: Lucas Leite Costa Franco
Matrícula: 20191020040031
Título do Trabalho: **AUTOMAÇÃO RESIDENCIAL:** SISTEMA DE ILUMINAÇÃO INTELIGENTE CONTROLADO POR DISPOSITIVO MÓVEL

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.

Jataí, 03 / 02 /2024.
Local Data

Lucas Leite Costa Franco

Assinatura do Autor e/ou Detentor dos Direitos Autorais

LUCAS LEITE COSTA FRANCO

AUTOMAÇÃO RESIDENCIAL: SISTEMA DE ILUMINAÇÃO INTELIGENTE CONTROLADO POR DISPOSITIVO MÓVEL

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Campus Jataí, para obtenção do grau de bacharel em Engenharia Elétrica.
Orientadora: Profa. Ma. Patrícia Gomes de Souza Freitas

JATAÍ – GO

2023

Dados Internacionais de Catalogação na Publicação na (CIP)

Franco, Lucas Leite Costa.

Automação residencial: sistema de iluminação inteligente controlado por dispositivo móvel [manuscrito] / Lucas Leite Costa Franco. - Jataí: IFG, Coordenação do Curso de Bacharelado em Engenharia Elétrica, 2023.

82 f.; il.

Orientadora: Profa. Ma. Patrícia Gomes de Souza Freitas.

Bibliografias.

Apêndices.

1. Aplicativo. 2. Arduino. 3. Automação residencial. 4. Controle de iluminação. I. Freitas, Patrícia Gomes de Souza. II. IFG, Câmpus Jataí. III. Título.

Ficha catalográfica elaborada pela Seção Téc.: Aquisição e Tratamento da Informação.
Bibliotecária – Rosy Cristina O. Barbosa – CRB 1/2380 – C. Jataí. Cód. F009/2024-1.

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

Folha de Aprovação

LUCAS LEITE COSTA FRANCO

AUTOMAÇÃO RESIDENCIAL: sistema de iluminação inteligente controlado por dispositivo móvel

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

Monografia defendida e aprovada, em 29 de novembro de 2023, pela banca examinadora constituída pelos seguintes membros:

BANCA EXAMINADORA

Profa. Ma. Patrícia Gomes de Souza Freitas
Presidente da banca / Orientadora
Instituto Federal de Educação, Ciência e Tecnologia de Goiás - Câmpus Jataí

Profa. Ma. Kennya Resende Mendonça
Membro
Instituto Federal de Educação, Ciência e Tecnologia de Goiás - Câmpus Jataí

Profa. Ma. Camila Dias de Jesus
Membro
Instituto Federal de Educação, Ciência e Tecnologia de Goiás - Câmpus Jataí

Jataí – GO

2023

Documento assinado eletronicamente por:

- **Kennya Resende Mendonca**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 01/02/2024 09:31:27.
- **Camila Dias de Jesus**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/01/2024 16:22:02.
- **Patricia Gomes de Souza Freitas**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/01/2024 15:28:04.

Este documento foi emitido pelo SUAP em 12/11/2023. 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: 474938
Código de Autenticação: f61d623007

Dedico este trabalho a todos aqueles que acreditaram no meu potencial, família, amigos e professores, e me incentivaram a nunca desistir dos meus sonhos.

AGRADECIMENTOS

Engenharia é mais do que apenas um curso ou uma profissão, é transformar a vida das pessoas e construir um mundo melhor. Por isso agradeço, primeiramente, a Deus, por ter me dado forças para superar os obstáculos que surgiram ao longo da minha caminhada.

Agradeço à minha família, em especial aos meus pais, Giulena Rosa Leite e Milton Pereira da Costa Filho, que compreenderam minhas ausências e não mediram esforços para que eu pudesse alcançar meus sonhos, celebrando cada vitória ao meu lado.

Agradeço à minha namorada, Manuela Carvalho Garcia de Assis, por sua constante presença e incansável apoio. Seu amor e compreensão foram luzes que iluminaram cada passo que dei, me incentivando a persistir naquilo que acreditava ser correto. Agradeço também à sua família por toda receptividade em suas vidas e todo suporte, em especial para conclusão deste trabalho.

Aos amigos, tanto aqueles que já faziam parte da minha vida quanto aos que conheci nesta jornada, dedico minha gratidão. Suas risadas e momentos de descontração tornaram os desafios mais leves, e a certeza de poder contar com o auxílio de vocês sempre que precisei foi inestimável.

Por último, expresso minha sincera gratidão aos meus professores e à minha orientadora, Patrícia Gomes de Souza Freitas. Suas orientações e o conhecimento que compartilharam moldaram meu crescimento pessoal e profissional, preparando-me para me tornar um Engenheiro Eletricista capaz e dedicado. Cada oportunidade que me proporcionaram foi um presente, pelo qual serei eternamente grato.

“Talvez não tenhamos conseguido fazer o melhor, mas lutamos para que o melhor fosse feito. Não somos o que deveríamos ser, não somos o que iremos ser... mas Graças a Deus, não somos o que éramos.”

(Martin Luther King)

RESUMO

A automação residencial abrange a integração de diversas ferramentas e tecnologias em um espaço residencial, proporcionando a execução automática de tarefas e aprimorando o conforto no dia a dia. Dentro desse contexto, a iluminação inteligente tem ganhado destaque, impulsionada pela crescente demanda global e pelo seu potencial em reduzir o consumo de energia, considerando que a iluminação representa uma parcela significativa do consumo elétrico. O sistema de iluminação inteligente é composto por lâmpadas equipadas com sensores e controladores interconectados, permitindo o gerenciamento e a transmissão de dados para otimizar a iluminação e evitar desperdícios energéticos. Desta forma, o presente trabalho teve como objetivo criar um ambiente de teste e desenvolver um aplicativo para dispositivos móveis integrando o conceito e elementos da automação residencial. O sistema foi composto por lâmpadas, sensores de presença, fotorresistor, módulo Arduino e *smartphones*, permitindo a comunicação entre eles e o controle remoto da iluminação do ambiente. Os resultados obtidos revelaram limitações no sistema como: a comunicação eficaz em até 10 metros devido ao uso do módulo *Bluetooth*; desafios estéticos no aplicativo, pois o *MIT App Inventor*, embora robusto, não é otimizado para desenvolvimento profissional; e além disso, a restrição no número de componentes e ambientes integrados ao sistema, decorrente das portas limitadas disponíveis no Arduino Uno. Apesar das barreiras mencionadas, o sistema opera de maneira eficaz, demonstrando a fácil integração de dispositivos simples e o *smartphone* para o desenvolvimento de um sistema de iluminação inteligente que traz consigo conforto e praticidade.

Palavras-Chave: Aplicativo; Arduino; Automação Residencial; Controle de Iluminação.

ABSTRACT

Home automation encompasses the integration of various tools and technologies in a residential space, providing the automatic execution of tasks and enhancing day-to-day comfort. Within this context, smart lighting has gained prominence, driven by the growing global demand and its potential to reduce energy consumption, considering that lighting represents a significant portion of electrical usage. The smart lighting system consists of bulbs equipped with sensors and interconnected controllers, allowing for the management and transmission of data to optimize lighting and prevent energy waste. Thus, the present work aimed to create a test environment and develop a mobile application integrating the concept and elements of home automation. The system comprised bulbs, motion sensors, a photoresistor, Arduino module, and smartphones, enabling communication between them and remote control of the ambient lighting. The results revealed limitations in the system, such as effective communication within a range of up to 10 meters due to the use of Bluetooth module; aesthetic challenges in the application, as the MIT App Inventor, although robust, is not optimized for professional development; and moreover, restrictions on the number of components and environments integrated into the system, resulting from the limited ports available on the Arduino Uno. Despite these mentioned barriers, the system operates effectively, demonstrating the easy integration of simple devices and smartphones for the development of a smart lighting system that brings comfort and convenience.

Keywords: Application; Arduino; Home Automation; Lighting Control.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CLP – Controlador Lógico Programável

DC – *Direct Current*

GND – *Ground*

LDR – *Light Dependent Resistor*

LED – *Light Emitting Diode*

PIR – *Passive Infrared Sensor*

PLC – *Power Line Carrier*

PWM – *Pulse With Modulation*

TRIAC – Tríodo de Corrente Alternada

Vin – Tensão de Entrada

SUMÁRIO

1 INTRODUÇÃO

A descoberta da eletricidade revolucionou o entendimento da humanidade a respeito do mundo e das diversas possibilidades de desenvolvimento que poderiam surgir através dela. A mudança dos costumes e hábitos da sociedade que conhecemos foi imprescindível na evolução da tecnologia que temos e na revolução de diversos sistemas, como o de iluminação, que ocorreu desde a invenção da primeira lâmpada ,por Thomas Edson, que utilizava energia elétrica para emissão de luz, propiciando assim que as pessoas pudessem se aventurar em atividades noturnas (Soares, 2014).

Ao longo dos anos, a humanidade tem se tornado cada vez mais dependente da luz e da energia, integrando-as em sua rotina e sendo influenciada por elas. Estudos realizados revelam que uma iluminação adequada tem um impacto direto no bem-estar e na disposição das pessoas, além de desempenhar um papel fundamental na prevenção de acidentes (Fernandes, 2011).

Neste contexto, o consumo de energia elétrica tem aumentado anualmente devido à ação humana e às crescentes demandas. A economia de energia elétrica tornou-se uma questão vital em muitos países, destacando-se a importância da iluminação, que é responsável pela maior parcela do consumo nacional de eletricidade (Baharum *et al.*, 2021).

O antigo sistema de iluminação estava limitado a apenas duas opções: ligar e desligar, o que acarretava desvantagens próprias. Essa forma de operação resultava em perda de energia elétrica devido ao funcionamento constante na tensão nominal, mesmo que o requisito real pudesse ser menor, dependendo das condições de iluminação natural (Rath, 2016). Os projetos convencionais já não satisfazem mais às expectativas dos moradores, especialmente no que se refere às inovações tecnológicas em suas novas residências (Ruppel *et al.*, 2013).

Em meados da década de 70, nos Estados Unidos, um novo conceito surgiu revolucionando o controle de iluminação daquela época, e com a utilização de módulos inteligentes que integravam o conceito de *Power Line Carrier* (PLC), permitiu o acionamento remoto de equipamentos e luzes através do envio de sinais pela própria rede elétrica (Muratori e Dal Bó, 2013). O avanço tecnológico em questão se tornou pioneiro na automação residencial, proporcionando um ajuste e controle muito mais eficientes da iluminação e resultando em um significativo aproveitamento energético aprimorado.

Desde então, a automação residencial conquistou uma crescente notoriedade a nível global, sendo este progresso fortemente impulsionado pelo rápido avanço da informática e pelo desenvolvimento constante de *softwares* de supervisão e gestão (Teza, 2002).

1.1 Objetivo Geral

Desenvolver um sistema de iluminação residencial inteligente que integre os conceitos e componentes da automação residencial, como sensores e o módulo Arduino. O objetivo central é viabilizar o controle eficiente e personalizado da iluminação por meio de um aplicativo móvel intuitivo e de fácil utilização, proporcionando aos usuários maior conforto e eficiência energética.

1.2 Objetivos Específicos

- Realizar um estudo aprofundado acerca da automação residencial, sistemas de iluminação inteligente e controle de elementos residenciais via aplicativo móvel, visando gerar uma base teórica sólida sobre o tema;
- Identificar e analisar as soluções existentes em outras bibliografias voltadas para o controle de iluminação residencial, com o objetivo de compreender as tecnologias, funcionalidades disponíveis e limitações inerentes a esses sistemas;
- Projetar e desenvolver o sistema de iluminação, considerando a integração dos sensores, o módulo Arduino e o aplicativo móvel, levando em conta os requisitos de eficiência energética, conforto e facilidade de uso;
- Projetar um ambiente de teste (banca didática/maquete) que simule de forma realista a implementação do sistema, proporcionando uma visualização mais precisa do seu funcionamento;
- Avaliar a viabilidade e efetividade do sistema de iluminação inteligente residencial, considerando aspectos técnicos e de aceitação por partes dos usuários.

1.3 Justificativa

Com a evolução da tecnologia, os recursos disponíveis para a implementação da automação de iluminação em residências expandiram significativamente, se tornando cada vez mais acessíveis e oferecendo uma ampla gama de possibilidades de integração e associação, o que reflete consideravelmente no aumento da demanda no mercado mundial de iluminação inteligente, que de acordo com o relatório de 2021 da *Research and Market*, tende a crescer 20,4% ao ano entre 2021 e 2028, atingindo a marca de US$46,90 bilhões até o final do período (EXPOLUX, 2022).

Diante desse cenário, o presente trabalho foi conduzido com o intuito de explorar as possibilidades que a automação de iluminação residencial oferece. A escolha desse tema como

objeto de estudo se justifica não apenas pelo seu impacto no mercado e no cenário atual, mas também pelo seu potencial de melhorar a qualidade de vida dos moradores, tornando suas residências mais eficientes e confortáveis. Além disso, a contribuição para a redução do consumo de energia elétrica, que é uma das principais preocupações em termos de sustentabilidade, torna o estudo desse tema ainda mais relevante.

2 FUNDAMENTAÇÃO TEÓRICA

A automação residencial, também conhecida como domótica, pode ser entendida como a integração de sistemas tecnológicos que são capazes de interagir entre si e seguir programações definidas pelos moradores. Em essência, a automação residencial é a aplicação da automatização e controle nas residências, visando melhorar a qualidade de vida, proporcionar conforto, aumentar a segurança e otimizar o consumo de energia (Muratori e Dal Bó, 2013).

Para Dias e Pizzolato (2004), a domótica é constituída por uma rede de comunicação que possibilita a interligação de diversas interfaces, dispositivos, componentes e sistemas, com a finalidade de coletar informações sobre a residência e executar certas ações para administrar este ambiente de forma eficaz.

Para Almeida e Rall (2015) e Souza Neto (2021), a automação residencial busca simplificar o dia a dia das pessoas, auxiliando nas tarefas mais simples até as mais complexas, trazendo conforto, praticidade e segurança, trabalhando em um conjunto de *Software* e *Hardware*.

A automação residencial abrange uma variedade de sistemas, incluindo telefonia, informática, rede elétrica, segurança, climatização e, em especial, iluminação. No contexto da automação, a iluminação inteligente é a técnica que traz otimização para a iluminação levando em conta o bem-estar luminoso e a eficiência energética (Leite, 2023).

Devido à variedade de dispositivos e métodos de programação disponíveis, diversos estudos foram realizados com o propósito de empregar essas ferramentas na criação de um sistema capaz de controlar e otimizar a iluminação de forma eficaz.

Segundo Rath (2016), o controle de iluminação implementado utiliza um fotorresistor para ajustar a intensidade luminosa do *Light Emitting Diode* (LED). Com o auxílio do módulo Arduino UNO, foi desenvolvido um código que aumenta o brilho do LED à medida que a noite se aproxima e o ambiente escurece, diminuindo o consumo de energia elétrica ao evitar o uso da tensão nominal disponível durante os períodos em que o ambiente já apresenta alta luminosidade. No estudo de Baharum *et al.* (2021), é demonstrado que o controle da iluminação por meio de um fotorresistor requer que a porta de saída do Arduino conectada ao LED ou lâmpada seja do tipo *Pulse Width Modulation* (PWM). Esse tipo de porta permite a variação da saída em uma escala de 0 a 255, gerando uma onda quadrada estável com o ciclo de trabalho especificado para controlar o brilho de forma precisa.

No estudo de Ruppel *et al*. (2013), procurou-se aplicar o princípio de funcionamento do *dimmer* para controlar um “interruptor” microcontrolado. Com a utilização do microcontrolador Atmega8, o sistema permitiu a regulação da intensidade luminosa dos LEDs por meio do atraso de disparo do tríodo de corrente alternada (TRIAC). Quanto menor o atraso, maior será a potência dos LEDs. A variação do controle de iluminação, por meio do TRIAC, é realizada através da comunicação entre o microcontrolador e um *personal* computer, utilizando um módulo *wi-fi* e por meio de uma interface visual desenvolvida em Visual Basic para facilitar o ajuste de luminosidade.

No trabalho de Almeida e Rall (2015), a função *dimmer* para o controle de iluminação foi implementada utilizando o dispositivo *Dimmer Shield*, que, em conjunto com o código desenvolvido para o Arduino UNO R3, oferece meios para regular a potência de funcionamento das lâmpadas. Com o objetivo de automatizar ainda mais o sistema, foi criado um aplicativo para celular utilizando o *software MIT Inventor 2*, estabelecendo comunicação com o módulo Arduino por meio do dispositivo *Ethernet Shield*, que permite o acesso a uma rede local ou a internet.

Segundo estudos de Baharum *et al.* (2021), o controle de iluminação implementado não utiliza apenas o fotorresistor, conforme mencionado anteriormente, mas também incorpora o Módulo Sensor PIR como um sensor de presença. Esse sensor é responsável por detectar movimentação no ambiente em que está instalado. Quando o sensor percebe um movimento, um sinal "verdadeiro" é enviado para o Arduino utilizado, acionando o LED. Após um período de 3 minutos sem movimento, o LED é desligado. No trabalho de Socolowiski (2022), é realizado uma alteração no código do Arduino para evitar que o Sensor PIR funcione durante períodos do dia em que a iluminação ambiente já está suficientemente alta. Dessa forma, os LEDs e lâmpadas conectados ao sensor apenas recebem informações dele dentro de uma faixa horária específica, reduzindo o desperdício de energia durante o dia.

Assim, os estudos realizados possibilitaram identificar a variedade de dispositivos em um sistema de iluminação inteligente e a lacuna em pesquisas voltadas para a integração dos interruptores residenciais. E com base nessa compreensão, surge a ideia de elaborar um esquema para integrar esses elementos de automação residencial, visando uma maior proximidade com a realidade residencial.

3 ELEMENTOS DA AUTOMAÇÃO RESIDENCIAL E DO SISTEMA DE ILUMINAÇÃO INTELIGENTE

A automação residencial é uma tecnologia avançada que abrange uma ampla gama de elementos essenciais para seu funcionamento eficaz. Dentro desse ecossistema, destacam-se componentes fundamentais, tais como sensores, controladores, atuadores e interface.

Os sensores desempenham um papel fundamental na automação, pois são responsáveis por detectar as condições do ambiente por meio da monitorização de grandezas físicas, como temperatura, pressão, nível e luminosidade. Esses sensores convertem essas informações em valores que podem ser processados por sistemas computacionais. Ao detectar um evento ou uma mudança nas grandezas físicas, os sensores enviam essas informações aos controladores (Camargo, 2010; Silva *et al.*, 2019).

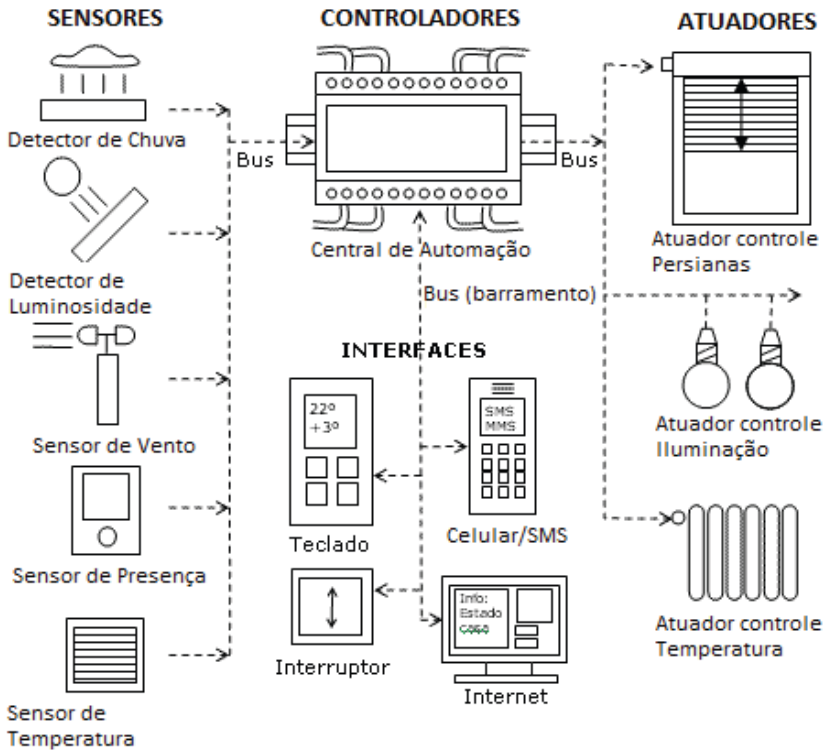
Os controladores são dispositivos que supervisionam as informações dos sensores e interagem com os atuadores. Eles são responsáveis por tomar decisões com base nas informações recebidas dos sensores, no estado do processo e em regras específicas, a fim de manter o sistema funcionando corretamente (Camargo, 2010). Essas decisões podem resultar no envio de comandos para ativar ou desativar equipamentos por meio dos atuadores. Portanto, os controladores desempenham um papel central na coordenação e na execução das operações de automação, garantindo o funcionamento eficaz do sistema como um todo (Accardi e Dodonov, 2012).

Os atuadores são dispositivos eletromecânicos que recebem comandos do sistema e atuam sobre os equipamentos automatizados (Accardi e Dodonov, 2012). Quando o controlador, por meio dos sensores, identifica a necessidade de ajustes no processo, ele envia comandos para esses atuadores, que têm a função de modificar variáveis do sistema (Camargo, 2010). Portanto, os atuadores desempenham um papel crucial na execução das ações determinadas pelo sistema de automação, contribuindo para o controle e a otimização de processos automatizados.

Por fim, as interfaces são elementos que possibilitam aos usuários interagir com o sistema de automação e acessar informações de forma intuitiva (Accardi e Dodonov, 2012). Esses dispositivos ou mecanismos, que podem variar desde navegadores de internet e dispositivos móveis até painéis, controles remotos e interruptores, desempenham o papel crucial de proporcionar uma experiência eficaz ao usuário, permitindo a visualização de informações e o controle das operações do sistema de automação.

Ao analisarmos a Figura 1, podemos compreender a comunicação entre esses quatro elementos de maneira clara. O sensor capta informações do ambiente e as transmite ao controlador, que processa os dados e emite comandos para os atuadores. Isso permite que o usuário interaja com o sistema por meio da interface, seja para analisar seu *status* ou efetuar alterações conforme necessário.

Figura 1 – Comunicação entre os elementos básicos da Automação Residencial



Fonte: Adaptado de Accardi e Dodonov (2012)

A integração eficaz desses elementos pode ser adaptada para uma variedade de sistemas, expandindo as possibilidades da automação em diversos setores, como segurança, áudio e vídeo, comunicação e, especialmente, iluminação. Todos esses campos compartilham um objetivo comum: proporcionar conforto e praticidade aos usuários do ambiente automatizado.

3.1 Sistema de Iluminação Inteligente

O sistema de iluminação inteligente, à semelhança de qualquer sistema de automação, incorpora em sua arquitetura os componentes fundamentais da automação: sensores, controladores, atuadores e interface.

Com base na comunicação apresentada na Figura 1, podemos conceber que em um sistema de iluminação inteligente, um fotorresistor desempenharia a função de sensor,

monitorando a luminosidade ambiente e transmitindo esses dados a um Controlador Lógico Programável (CLP). O CLP, por sua vez, avaliaria a necessidade de ativar ou desativar as lâmpadas, que funcionam como atuadores no sistema. Além disso, por meio de um aplicativo móvel e um interruptor, os usuários podem verificar o status da iluminação e interagir com o sistema, oferecendo controle adicional sobre a iluminação do ambiente.

A seguir, iremos explorar cada um dos elementos de automação residencial que foram aplicados neste trabalho, com foco especificamente no contexto da iluminação inteligente.

3.1.1 Sensores

a) Sensor de Presença PIR HC-SR501:

O Sensor PIR (*Passive Infrared Sensor*, em Português, Sensor Passivo de Infravermelho), apresentado na Figura 2, é um sensor de movimento compatível com Arduino que ao detectar movimentação em seu *range* de leitura, que possui ângulo de abertura de aproximadamente 120°, o pino de saída retorna nível lógico alto. Ele é projetado para detectar movimento de objetos ou pessoas com base em mudanças na radiação infravermelha emitida por corpos em movimento.

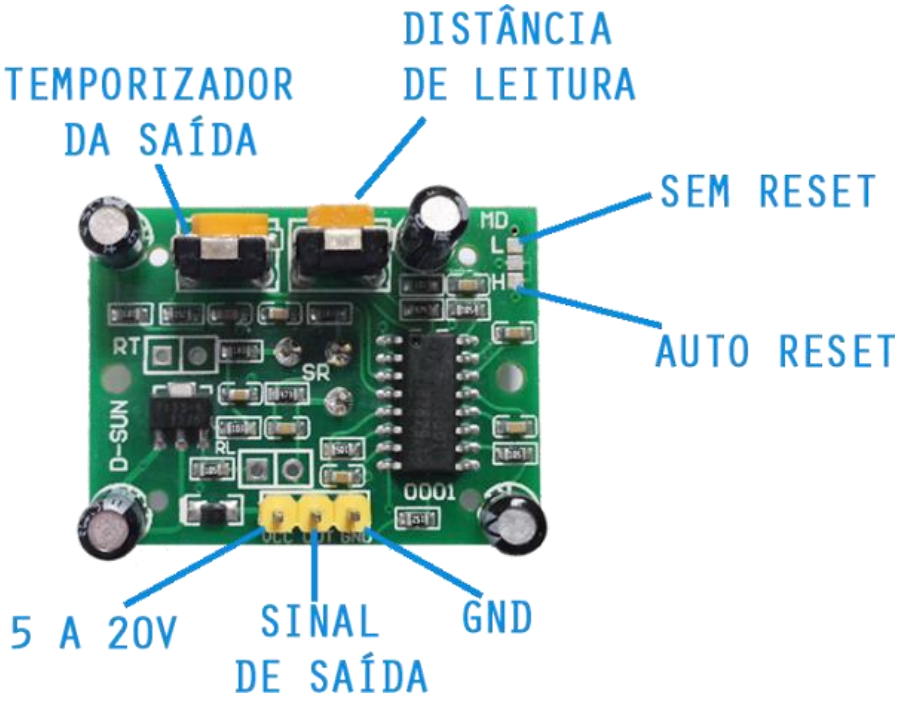
Figura 2 – Sensor PIR HC-SR501

Fonte: Robocore (2008)

Sua composição, conforme ilustrado na Figura 3, inclui três pinos essenciais: um para a saída de sinal (tensão de saída de 3,3V), outro para fornecimento de energia ao próprio dispositivo (tensão de alimentação de 5V a 20V) e um terceiro para o GND (*ground*, conhecido como "terra" ou "aterramento" em Português). Além disso, apresenta dois potenciômetros localizados na parte traseira do dispositivo. Um deles permite ajustar o tempo durante o qual o sinal de saída permanece em nível alto, com uma faixa de variação que vai de 5 segundos a 2,5

minutos. O segundo potenciômetro possibilita a configuração da distância máxima de detecção, com uma variação que vai de 3 a 7 metros.

Figura 3 – Estrutura do Sensor PIR HC-SR501



Fonte: Robocore (2008)

b) Sensor Fotorresistor LDR 5528

O Sensor Fotorresistor LDR (*Light Dependent Resistor*) 5528, Figura 4, é um dispositivo sensível à luz cuja sua resistência elétrica varia em resposta à intensidade da luz incidente sobre ele. Quanto mais intensa for a luz, menor será sua resistência, e vice-versa.

Figura 4 – Sensor Fotorresistor LDR 5528

Fonte: Adaptado de Alves (2020)

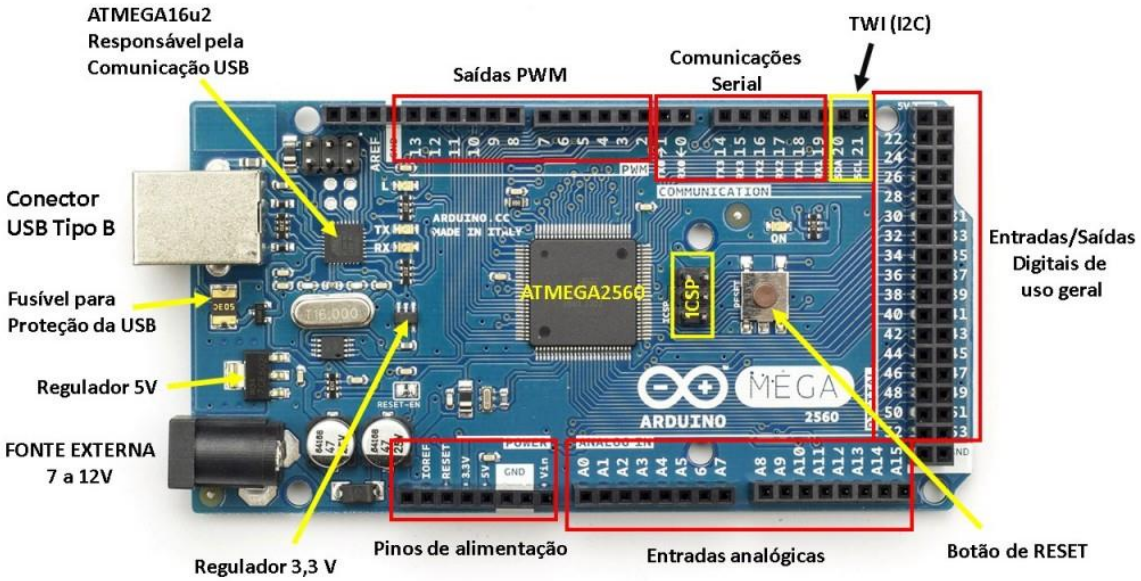
Quando a luz incide sobre o material semicondutor do LDR, os fótons de luz são absorvidos pelo material, fazendo com que alguns elétrons que estão em sua banda de valência ganhem energia suficiente para saltar para a banda de condução, e dessa forma a condutividade do LDR aumenta e sua resistência diminui (Alves, 2020).

3.1.2 Controladores

a) Arduino MEGA 2560

O Arduino MEGA 2560, ilustrado na Figura 5, é uma placa Arduino que utiliza o microcontrolador ATmega2560 como base. Essa placa oferece a flexibilidade de programação, permitindo ao usuário adaptá-la de acordo com suas necessidades específicas, o que abre um vasto leque de possibilidades para a criação de uma ampla variedade de sistemas e aplicações. Ela oferece uma variedade de recursos, incluindo 54 pinos digitais que funcionam tanto como entradas quanto como saídas, suportando operações digitais em nível *HIGH* e *LOW*. Dentre esses 54 pinos, 15 deles são capazes de produzir saídas PWM, o que possibilita o controle da intensidade média de um sinal elétrico não por meio da amplitude, mas sim variando a largura dos pulsos desse sinal. Além disso, a placa apresenta 16 pinos de entrada analógica que possibilitam a leitura de valores analógicos. Também estão disponíveis pinos de alimentação, incluindo 5V, 3.3V, Vin e GND, bem como pinos de comunicação serial que viabilizam a transferência de dados em série entre dispositivos eletrônicos (Souza, 2014).

Figura 5 – Arduino MEGA 2560

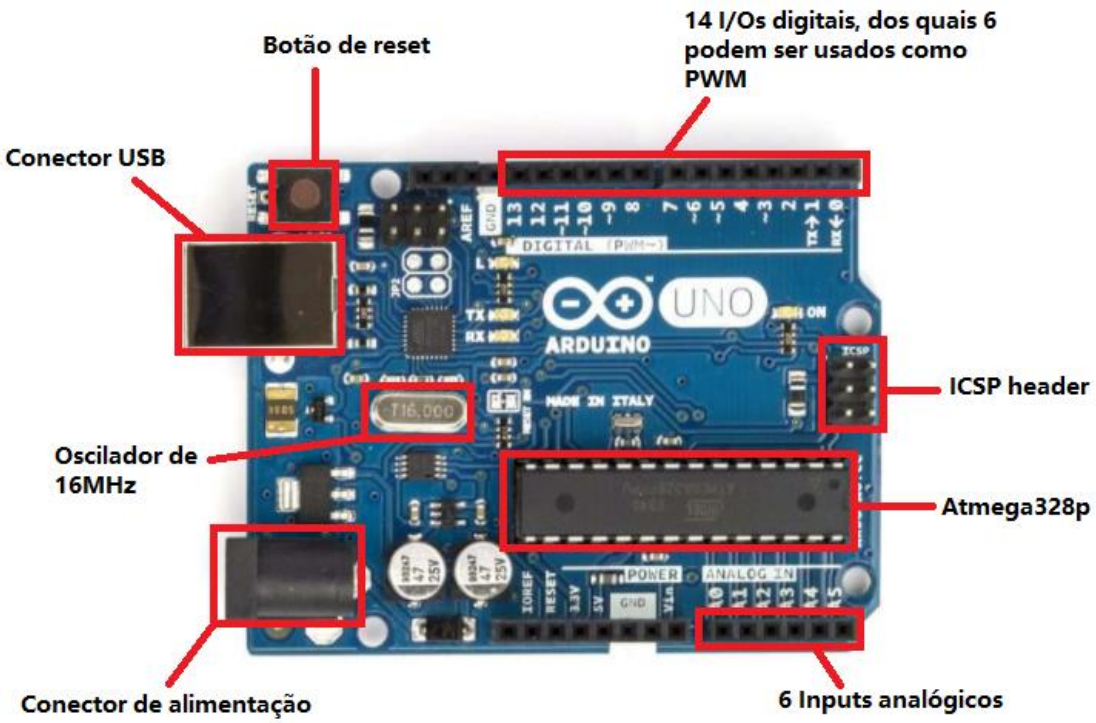


Fonte: ELETRU’S (2023)

b) Arduino UNO

O Arduino UNO, como ilustrado na Figura 6, é uma placa Arduino que utiliza o microcontrolador ATmega328P como base. Assim como, o Arduino MEGA 2560 oferece uma vasta flexibilidade e aplicabilidade que podem ser adaptadas de acordo com as necessidades do usuário ou programador ao escrever o código. Esta placa dispõe de diversos recursos, incluindo 14 pinos digitais que podem funcionar tanto como entradas quanto como saídas, permitindo operações digitais em níveis *HIGH* e *LOW*. Entre esses 14 pinos, 6 são capazes de gerar saídas PWM, e há ainda 6 pinos de entrada analógica que possibilitam a leitura de valores analógicos. Além disso, a placa apresenta pinos de alimentação, incluindo 5V, 3.3V, Vin e GND, bem como pinos de comunicação serial (Souza, 2013).

Figura 6 – Arduino UNO



Fonte: ELETROGATE (2020)

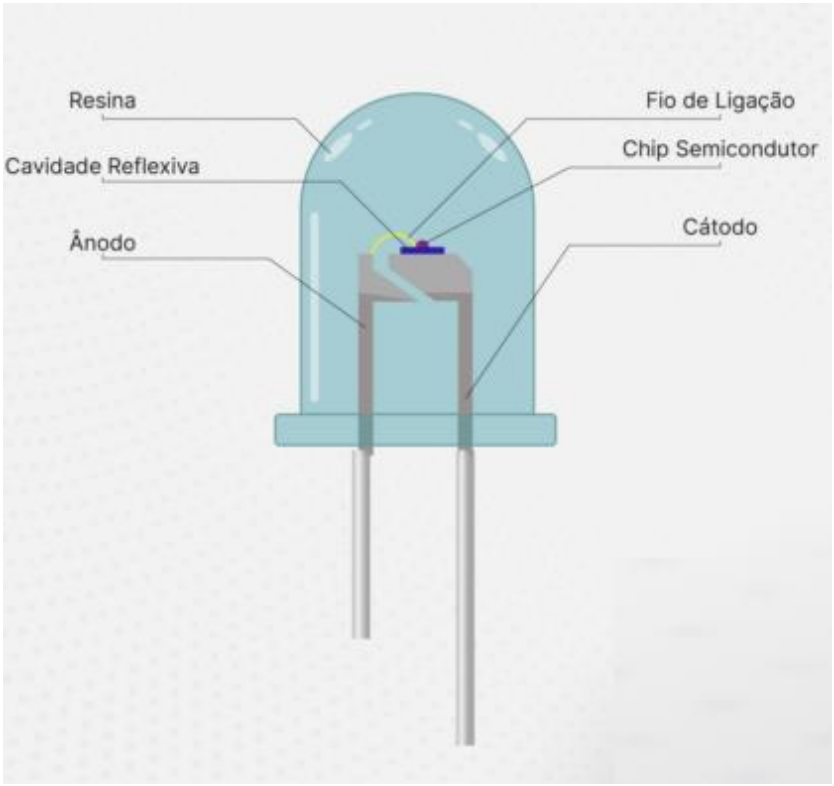
Um dos diferenciais do Arduino UNO em relação a outras placas Arduino, especialmente o Arduino MEGA 2560, é sua simplicidade de uso, portabilidade e tamanho compacto, que tendem a consumir menos energia. Essas características tornam o Arduino UNO uma escolha conveniente para projetos onde a eficiência energética, o espaço limitado ou a simplicidade são essenciais.

3.1.3 Atuadores

a) LED

O LED, representado na Figura 7, é um dispositivo eletrônico que transforma eletricidade em luz por meio do fenômeno da eletroluminescência. Isso ocorre quando uma corrente elétrica é aplicada aos terminais do LED, resultando na recombinação de elétrons em um terminal com lacunas no outro terminal, o que gera a emissão de fótons luminosos (Alecrim e Marques, 2023). Essa conversão de energia elétrica em luz é altamente eficiente, tornando os LEDs uma escolha popular para iluminação e dispositivos de exibição, devido à sua economia de energia e longa vida útil.

Figura 7 – Estrutura do LED



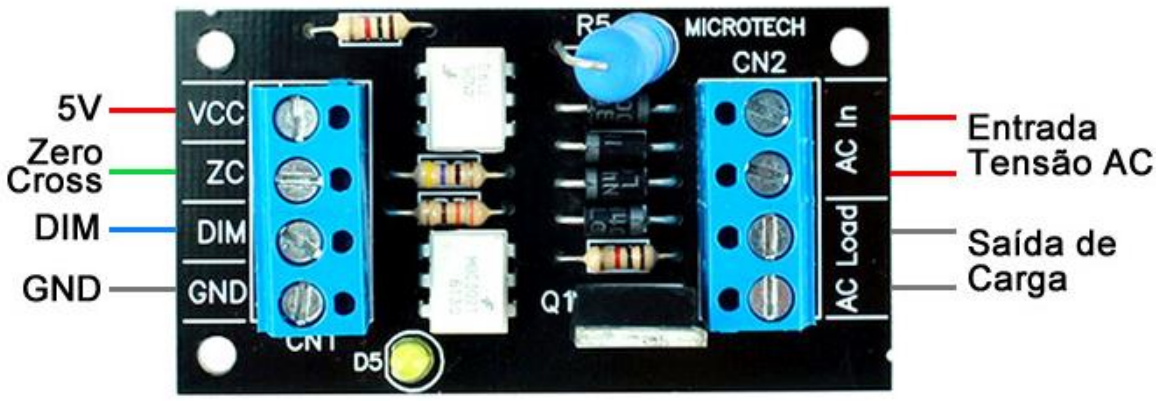
Fonte: Adaptado de Alecrim e Marques (2023)

b) Módulo *Dimmer* Arduino MC-8A

O Módulo *Dimmer* Arduino MC-8A, apresentado na Figura 8, é um dispositivo projetado para regular com precisão a quantidade de energia enviada para uma lâmpada que seja dimerizável. Ele opera por meio do TRIAC BT137, que controla a corrente alternada de entrada. Funcionando como um relé, o módulo dispõe de pinos de alimentação (5V e GND), juntamente com dois pinos DIM e *Zero Cross*, que se comunicam para ajustar a saída de energia para a carga, de acordo com o sinal recebido pelo Arduino e no momento em que a tensão de

entrada em corrente alternada cruza o zero, respectivamente. Essa configuração proporciona um controle eficiente sobre a intensidade luminosa da lâmpada.

Figura 8 – Módulo *Dimmer* Arduino MC-8A

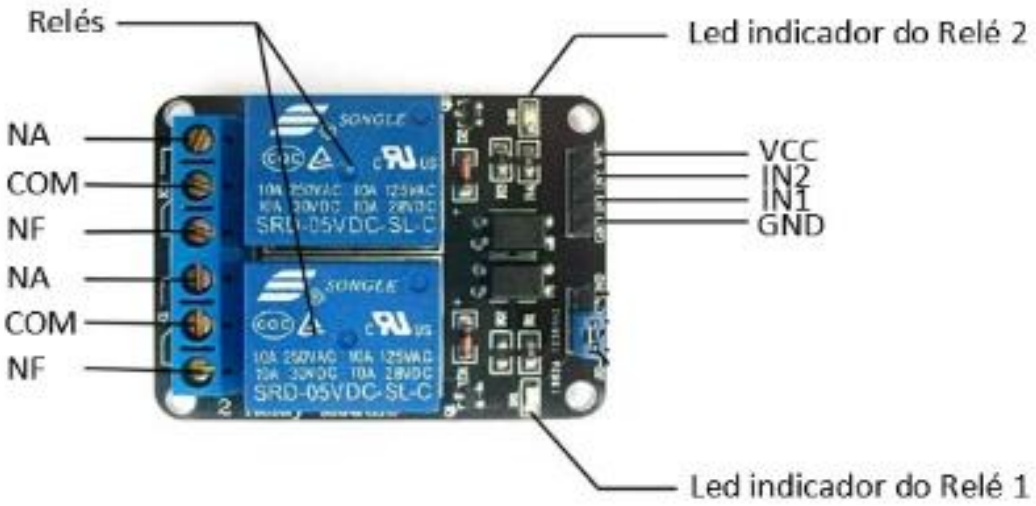


Fonte: Adaptado de USINAINFO (2023)

c) Módulo Relé

O Módulo Relé é um componente eletrônico fundamental para o controle de dispositivos de alta potência ou corrente por meio de sinais de baixa potência, como os emitidos por microcontroladores e placas Arduino. Este módulo, conforme ilustrado na Figura 9, inclui diversos pinos essenciais, como o pino VCC para alimentação do dispositivo, o pino GND para conexão à terra, os pinos INX, que são utilizados para controlar cada um dos relés presentes no módulo, bem como saídas que incluem os contatos comum – COM, normalmente aberto – NA e normalmente fechado – NF (STA, 2023).

Figura 9 – Componentes do Módulo Relé



Fonte: STA (2023)

O funcionamento básico de um relé é baseado no uso de uma bobina eletromagnética. Quando uma corrente elétrica é aplicada à bobina, ela cria um campo magnético que atrai um contato móvel, resultando na alteração de sua posição. Esse movimento, por sua vez, modifica a conexão elétrica entre os contatos do relé, alternando o estado do circuito controlado.

3.1.4 Interfaces

a) Interruptor *Dimmer*

O Interruptor *Dimmer* é um dispositivo projetado para ajustar a intensidade luminosa emitida por uma lâmpada, permitindo criar ambientes com diferentes níveis de iluminação. Esse controle de luminosidade é alcançado através da variação da resistência elétrica no circuito, utilizando componentes como potenciômetros ou dispositivos como o TRIAC, que ajustam a quantidade de corrente elétrica que flui para a lâmpada (Mattede, 2020).

Os Interruptores *Dimmer* são amplamente utilizados em ambientes residenciais e comerciais para personalizar a iluminação de espaços, criando atmosferas aconchegantes, ajustando a intensidade da luz para tarefas específicas e economizando energia elétrica quando uma iluminação total não é necessária. Eles estão disponíveis em designs e estilos variados, que incluem modelos com controles deslizantes, botões giratórios, como representado na Figura 10, e muito mais.

Figura 10 – Interruptor *Dimmer* com botão giratório

Fonte: Leroy Merlin (2023)

O *dimmer* pode ser encontrado em duas categorias: analógico ou digital. O primeiro é tradicionalmente operado por um potenciômetro, que ajusta a intensidade luminosa alterando de forma linear e contínua a quantidade de energia elétrica fornecida à lâmpada. Em contraste,

o segundo é frequentemente associado a circuitos microcontroladores, oferecendo um controle por valores discretos, que permite ajustes eficientes e menos suscetíveis a ruídos.

b) Interruptor Pulsador

O Interruptor Pulsador, representado na Figura 11, é um componente amplamente utilizado na automação de sistemas de iluminação. Ao contrário dos interruptores convencionais que permanecem em estado ligado ou desligado, o interruptor pulsador tem a característica de retornar automaticamente à sua posição inicial quando a pressão sobre o botão é aliviada. Quando pressionado, o mecanismo do interruptor pulsador fecha temporariamente o circuito elétrico, permitindo a passagem de corrente elétrica e enviando um sinal de nível alto para o controlador ao qual está conectado.

Figura 11 – Interruptor pulsador

Fonte: BAT (2023)

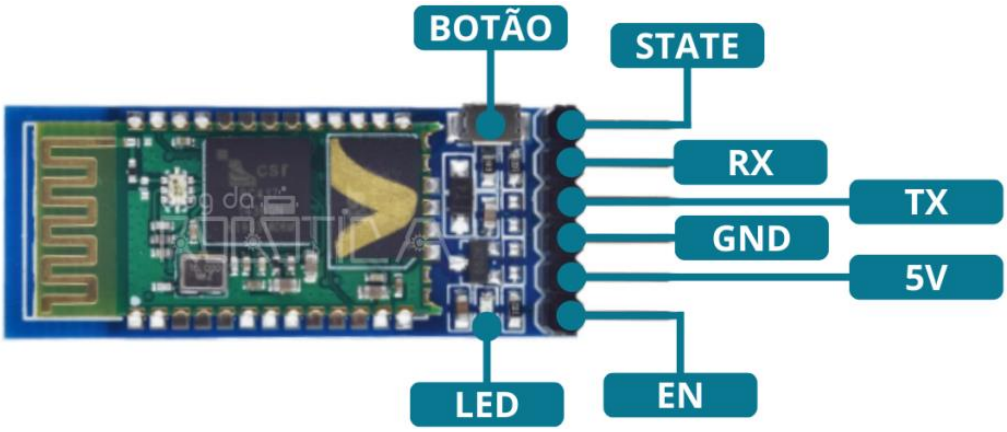
c) Módulo *Bluetooth* HC-05

O Módulo *Bluetooth* HC-05 é um componente que oferece conectividade sem fio via tecnologia *Bluetooth*, permitindo a troca de dados e comunicação entre dispositivos em um raio de até 10 metros. Este módulo pode operar em dois modos distintos: o modo mestre (*master*), que possibilita a conexão a vários dispositivos escravos, e o modo escravo (*slave*), no qual ele aguarda conexões de dispositivos mestres.

Conforme mostrado na Figura 12, o módulo HC-05 possui 6 pinos essenciais: alimentação de 5V, aterramento (GND), pino Rx (recebimento de dados seriais via *Bluetooth*), pino Tx (transmissão de dados seriais via *Bluetooth*), pino State (indicador de status do módulo,

retornando nível alto quando emparelhado com um dispositivo) e pino EN (usado para alternar entre o modo de dados e o modo de comando) (Viana, 2023).

Figura 12 – Pinos do Módulo *Bluetooth* HC-05

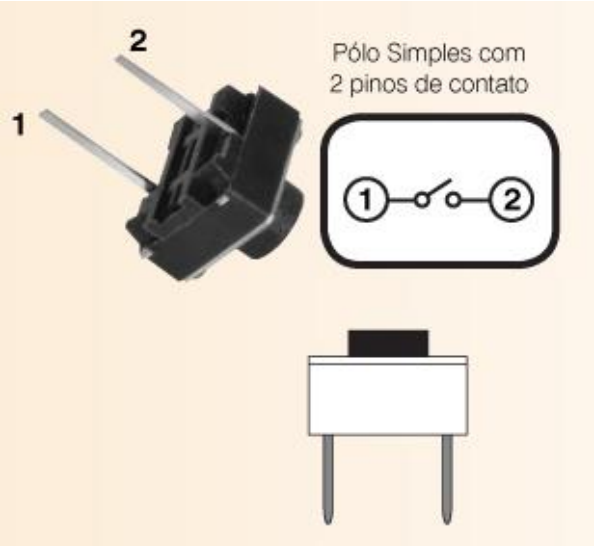


Fonte: Viana (2023)

d) *Push Button* (Chave Táctil) de 2 Pinos

O *Push Button* ou Chave Táctil de 2 pinos, é um componente elétrico amplamente empregado para estabelecer uma conexão elétrica momentânea ao ser pressionado, assemelhando-se ao funcionamento do interruptor pulsador. Este componente simples apresenta dois terminais, como evidenciado na Figura 13, sendo um deles conectado ao pino de entrada do dispositivo, enquanto o outro está ligado ao GND. Assim, ao pressionar o botão, ele temporariamente fecha o circuito, possibilitando o fluxo de corrente elétrica.

Figura 13 – *Push Button* de 2 Pinos

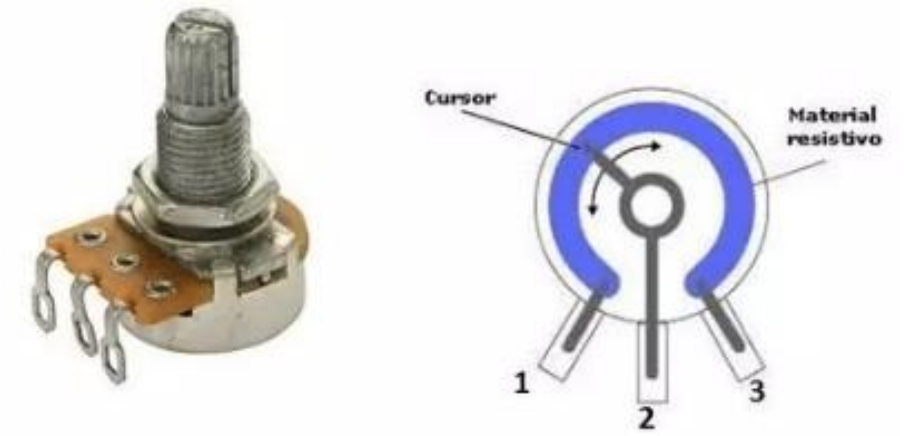


Fonte: Adaptado de Instituto Federal de Santa Catarina (2018)

e) Potenciômetro

O potenciômetro, ilustrado na Figura 14, é um componente elétrico usado para regular a resistência elétrica em um circuito. Ele é composto por um resistor de material condutivo, frequentemente em formato de trilha circular, com três terminais: o terminal central, conhecido como cursor, e dois terminais extremos. Quando uma tensão elétrica é aplicada aos terminais extremos do potenciômetro, a corrente percorre a trilha resistiva. Ao girar o eixo do potenciômetro, a posição do cursor na trilha se modifica, variando a resistência entre o cursor e um dos terminais extremos (Mattede, 2019). Isso resulta em uma saída de tensão ajustável no terminal central, permitindo o controle de diversos dispositivos elétricos.

Figura 14 – Potenciômetro e seu funcionamento interno



Fonte: Mattede (2019)

Após explorar o funcionamento dos diversos dispositivos essenciais para o sistema de iluminação inteligente, foi possível compreender a complexidade e a interconexão desses elementos na busca por um sistema de iluminação inteligente eficiente. No próximo capítulo, será apresentado a metodologia empregada para desenvolver e integrar esses componentes, delineando as etapas e abordagens que orientaram a implementação prática deste sistema.

4 METODOLOGIA

Este trabalho foi realizado em três fases: pesquisa bibliográfica, estudo de caso e desenvolvimento de um sistema prático; com o intuito de criar e implementar um sistema de iluminação inteligente para residências, utilizando conceitos e componentes da automação residencial, como sensores e módulo Arduino, controlado por um aplicativo móvel intuitivo e de fácil utilização.

A primeira fase consistiu em uma revisão bibliográfica abrangente sobre automação residencial, sistema de iluminação inteligente, sensores, módulo Arduino e aplicativos móveis. Foram consultadas fontes como artigos científicos, livros, dissertações e trabalhos de conclusão de curso para um embasamento teórico sólido para o desenvolvimento do trabalho.

Em seguida, foram realizados estudos de caso, em cima das bibliografias trabalhadas, a respeito das ferramentas utilizadas no desenvolvimento de cada um dos sistemas de iluminação inteligente. Isso envolveu a aquisição dos componentes necessários e a realização de testes para garantir seu funcionamento adequado. Nesta fase, foi decidido que o sistema proposto seria composto por cinco ambientes controlados por um Arduino. As lâmpadas seriam gerenciadas por uma variedade de dispositivos: interruptores pulsadores, frequentemente empregados na automação de iluminação; *dimmer* para ajuste da intensidade luminosa; sensor de presença; e fotorresistor para analisar a luminosidade ambiente. Além dos componentes físicos, também seria desenvolvido um aplicativo com conectividade *Bluetooth* para o controle remoto da iluminação.

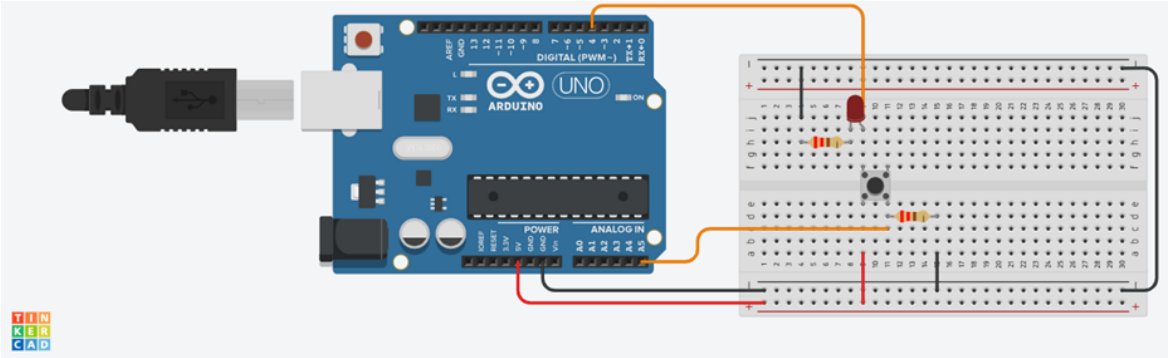
Para garantir uma progressão organizada, o desenvolvimento do sistema de iluminação inteligente foi dividido em etapas distintas. Inicialmente, foram conduzidos testes individuais em cada subsistema e ambiente, com o auxílio de *softwares* como o *TinkerCAD* e Arduino IDE para facilitar a programação. Em seguida, o foco se voltou para a criação do aplicativo móvel, desenvolvido na plataforma *MIT App Inventor*, seguido pela integração deste com os subsistemas previamente testados. Finalmente, uma bancada didática foi implementada para simular as condições reais encontradas nas redes elétricas residenciais, permitindo uma avaliação mais precisa e próxima do ambiente final.

No processo de desenvolvimento dos ambientes, foi decidido criar espaços simulando diferentes áreas de uma casa, incluindo sala, cozinha, quarto, garagem e área externa. Para a sala e cozinha, foram implementados sistemas de gerenciamento com interruptores pulsadores. O quarto foi equipado com um *dimmer* para controle da intensidade luminosa. Na garagem, foram instalados tanto um interruptor pulsador quanto um sensor de presença, este último

operando apenas durante a noite. Por fim, na área externa, foram empregados um interruptor pulsador e um fotorresistor para o controle da iluminação.

Para os ambientes que utilizam interruptores pulsadores, foi desenvolvido um sistema empregando um botão de pressão (*push button*) como substituto, conforme ilustrado na Figura 15. Este botão é alimentado pelo Arduino com uma tensão de 5V em um dos pinos e, através de um divisor de tensão com um resistor de 220 Ω em seu outro pino de saída, o sinal é devolvido ao Arduino para indicar se o botão está pressionado ou não.

Figura 15 – Sistema *Push Button* + LED



Fonte: Acervo do autor (2023)

O código deste sistema, apresentado na Figura 16, funciona da seguinte forma: quando o botão é pressionado, ele adiciona um ao valor de uma variável auxiliar (aux_Botao). Se o valor da variável for par, o sistema retorna 'verdadeiro' e mantém o LED apagado; caso contrário, retorna 'falso' e acende o LED. Para garantir um tempo de processamento adequado, foi incluído um atraso de 300 milissegundos para que apenas um clique seja contabilizado após a pressão do botão. Assim, foi possível simular o funcionamento do interruptor pulsador no sistema e observar sua interação com a lâmpada.

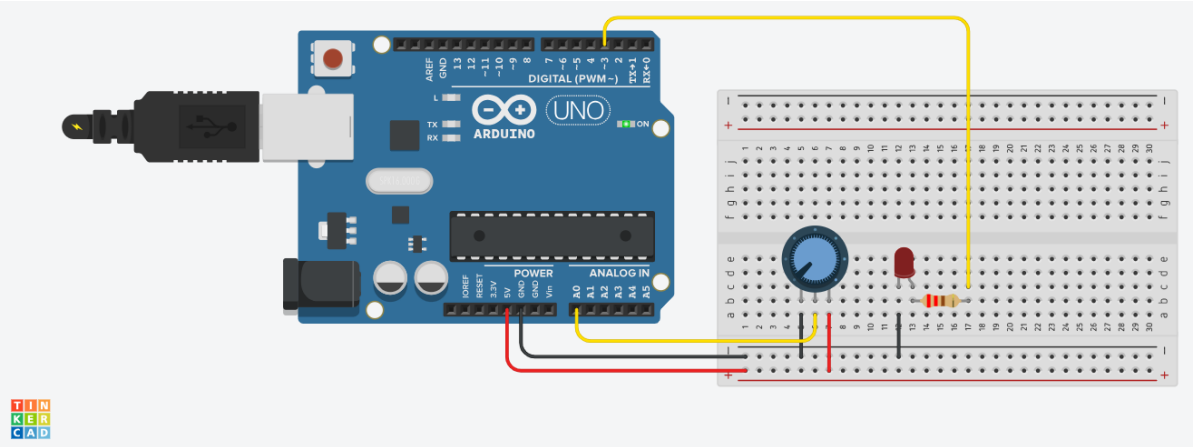
Figura 16 – Código *Push Button*

```
// função para o controle do botão como interruptor pulsador
bool Interruptor(){
  if (analogRead(BT)> 400){
    aux_Botao ++;   //adiciona 1 à variável
    delay(300);
  }
  if(aux_Botao % 2 == 0){   //verficação da variável como
    return true;            //ímpar ou par
  } else {
    return false;
  }
}
```

Fonte: Acervo do autor (2023)

Para controlar a intensidade luminosa no quarto, onde o *dimmer* foi implementado, optou-se por utilizar um potenciômetro como substituto, conforme ilustrado na Figura 17. Nesse arranjo, o potenciômetro é alimentado com uma tensão de 5V pelo Arduino. O terminal central do potenciômetro retorna o sinal ao Arduino, permitindo que, ao ajustar o potenciômetro, uma tensão proporcional seja enviada.

Figura 17 – Sistema Potenciômetro (*Dimmer*) + LED



Fonte: Acervo do autor (2023)

O código implementado, ilustrado na Figura 18, opera da seguinte maneira: uma variável (leitura_Pot) é responsável pela leitura da tensão ajustada pelo potenciômetro. Em seguida, essa variável mapeia o valor lido da porta analógica de entrada, que varia de 0 a 1023, para a faixa de saída da porta digital PWM, que varia de 0 a 255. Esse valor é, então, transmitido para a porta PWM, que controla o LED, regulando sua intensidade luminosa de acordo com o valor mapeado.

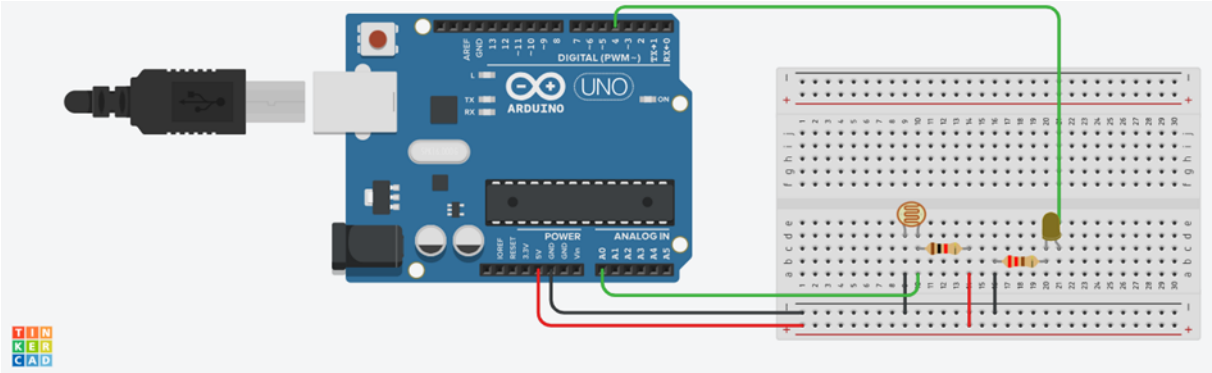
Figura 18 – Código Potenciômetro (*Dimmer*)

```
//função para controle do potenciômetro como dimmer
void Dimmer()
{  // Lê o valor analógico do potenciômetro.
  int leitura_Pot = analogRead(Potenciometro);
  // Mapeia o valor lido de 0-1023 para 0-255
  leitura_Pot = map(leitura_Pot, 0, 1023, 0, 255);
  // Define o brilho do LED com base na leitura do potenciômetro.
  analogWrite(LED, leitura_Pot);
}
```

Fonte: Acervo do autor (2023)

Na área externa, as luzes são controladas tanto por um interruptor pulsador quanto por um fotorresistor. Considerando que a função e a explicação do interruptor foram fornecidas anteriormente, é abordado na Figura 19 o esquema estrutural do fotorresistor. Este componente é alimentado com 5V provenientes do Arduino e está em série com um resistor de 1kΩ para obter uma variação de tensão correspondente à luminosidade ambiente. Através deste divisor de tensão, é possível monitorar a tensão de saída, que varia conforme a resistência do fotorresistor muda em resposta à luminosidade.

Figura 19 – Sistema Fotorresistor + LED



Fonte: Acervo do autor (2023)

Quanto ao código do fotorresistor, sua operação é simples, conforme ilustrado na Figura 20. O Arduino efetua a leitura da voltagem sobre o fotorresistor; caso essa leitura seja inferior ao valor estipulado na condicional, a função retorna ‘falso’ e mantém o LED apagado; do contrário, retorna ‘verdadeiro’ e acende o LED. Isso possibilita determinar o estado da luminosidade ambiente externa e gerenciar o funcionamento da lâmpada.

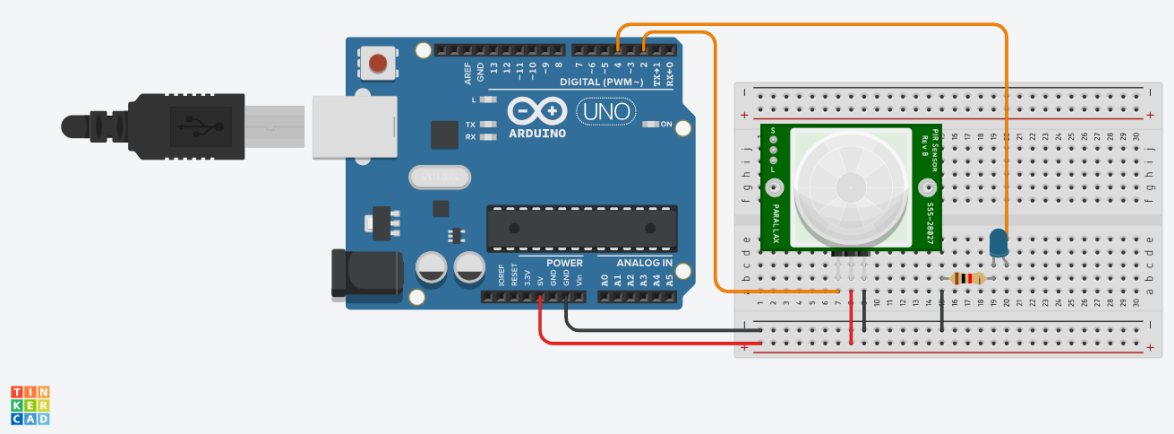
Figura 20 – Código Fotorresistor

```
//função luminosidade do ambiente
bool Escuro() {
  //leitura da tensão em cima do fotorresistor
  if (analogRead(Fotorresistor)<800) {
    return false;
  } else {
    return true;
  }
}
```

Fonte: Acervo do autor (2023)

Finalmente, o último sistema implementado é o sensor de presença posicionado na garagem, que entra em funcionamento somente quando o fotorresistor indica que o ambiente externo está escuro, indicando a noite. Dado que a função e explicação do fotorresistor foram apresentadas anteriormente, a Figura 21 abaixo representa o sistema do sensor PIR. Este sensor é alimentado pelo Arduino com uma tensão de 5V. Quando detecta movimento dentro de sua área de alcance, ele emite um sinal de nível alto no pino de saída para o Arduino por meio de uma porta digital.

Figura 21 – Sistema Sensor PIR + LED



Fonte: Acervo do autor (2023)

No que diz respeito ao código do Sensor PIR, seu funcionamento é simples, como apresentado na Figura 22. Quando o código detecta um sinal de saída de nível alto do sensor, ele retorna 'verdadeiro' para o Arduino, o que resulta no acionamento do LED. Caso contrário, se o sinal for de nível baixo, ele retorna 'falso', mantendo o LED desligado. Dessa maneira, é possível gerenciar a interação entre o sensor de presença e o controle da iluminação.

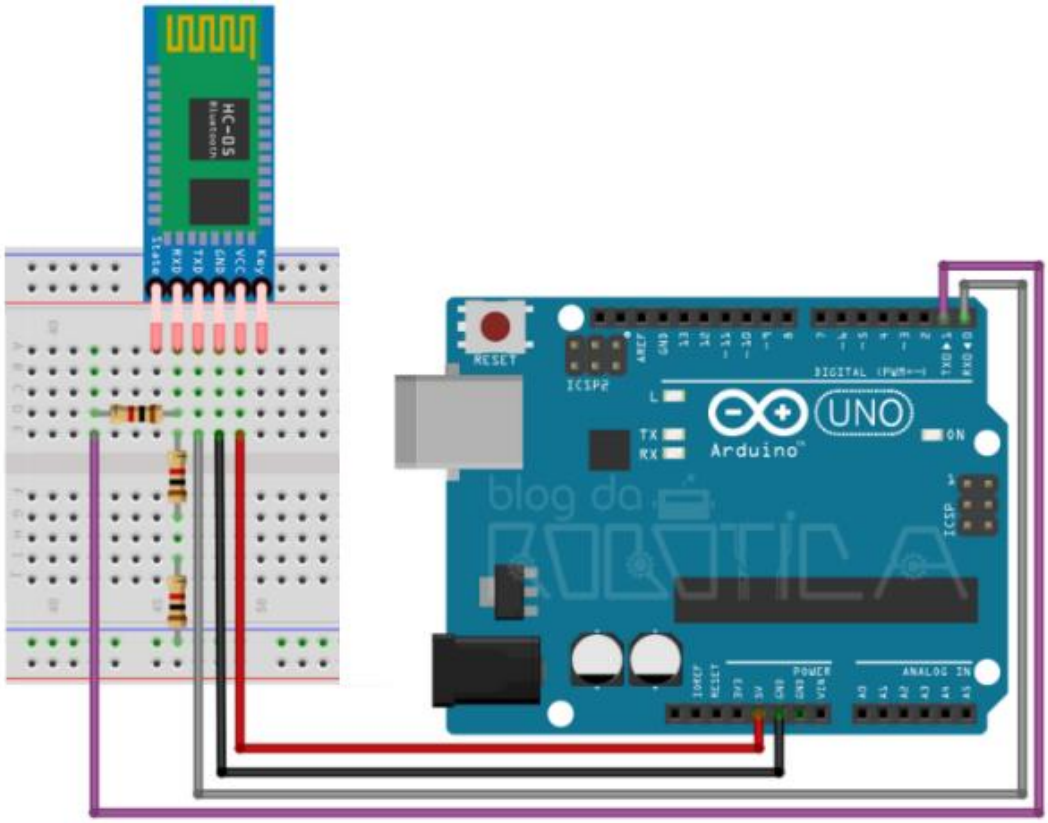
Figura 22 – Código Sensor PIR

```
//função status do Sensor PIR
bool PIR(){
  // leitura do sinal de saída do Sensor PIR
  if (digitalRead(Sensor_PIR) == HIGH){
    return true;
  }else{
    return false;
  }
}
```

Fonte: Acervo do autor (2023)

Após o desenvolvimento dos sistemas individuais, a atenção voltou-se para a implementação da comunicação *Bluetooth*. Para esse propósito, o módulo HC-05 foi integrado ao sistema, conforme ilustrado na Figura 23. O módulo é alimentado com 5V pelo Arduino, e seu pino de transmissão de dados (TX) foi conectado ao pino de recebimento de dados do Arduino (RX), e vice-versa. O pino RX do HC-05 suporta uma tensão de apenas 3.3V, o que exigiu o uso de resistores de 1k para criar um divisor de tensão.

Figura 23 – Conexão HC-05 com o Arduino

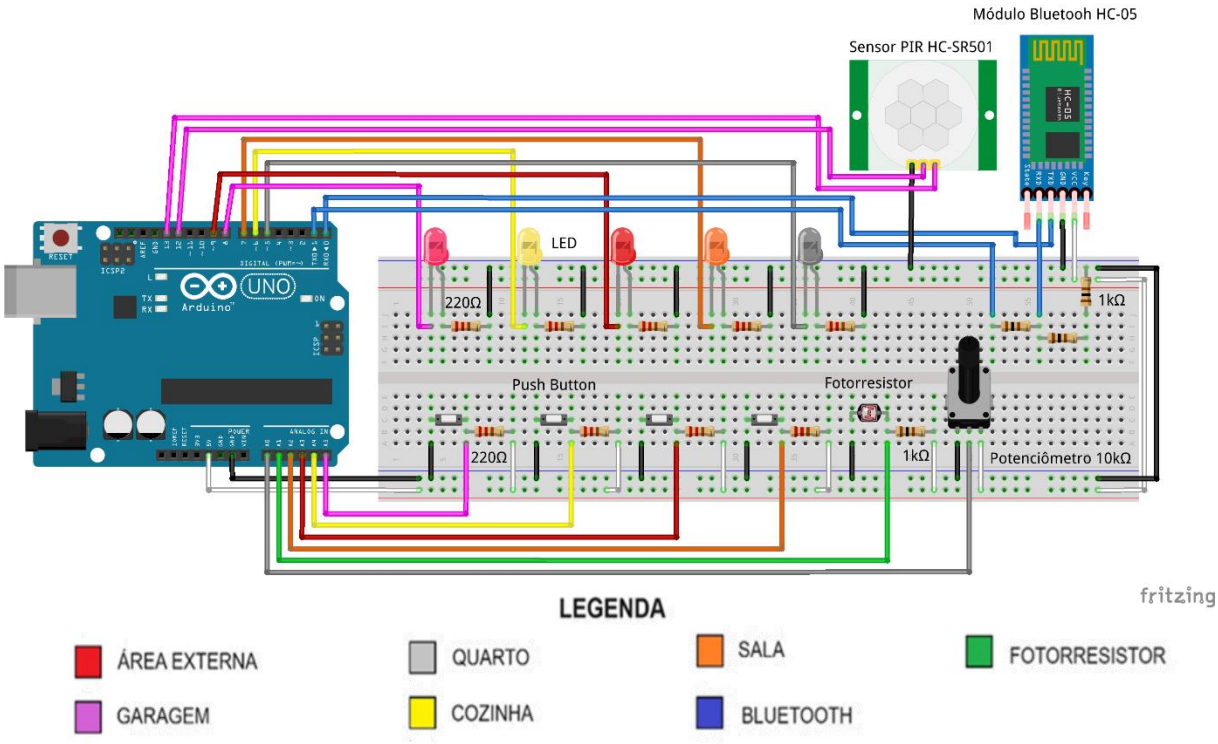


Fonte: Adaptado de Viana (2023)

Inicialmente, foram realizados testes utilizando o Arduino MEGA 2560 como controlador devido ao maior número de portas disponíveis. Contudo, a comunicação não ocorria de forma bidirecional; enquanto era possível enviar dados, a recepção não ocorria de forma adequada. Na procura por uma solução, deparou-se com relatos no fórum do Arduino.cc (2020) de outros usuários enfrentando a mesma questão. No entanto, mesmo com o passar dos anos, nenhuma resolução foi desenvolvida. Desta forma, optou-se pelo Arduino UNO, que resolveu o problema de comunicação, embora tenha reduzido as possibilidades e portas de comunicação do sistema com o Arduino.

Após a conclusão dos testes no *software TinkerCAD* e a finalização dos códigos, o circuito completo do sistema foi montado na plataforma *Fritzing*, utilizando os componentes adquiridos, conforme mostrado na Figura 24. Cada sistema foi identificado por uma cor específica de cabo para distinguir os diferentes ambientes. O conjunto da área externa foi marcado com cabos vermelhos, a garagem com o sensor PIR em lilás, o fotorresistor em verde (não utilizou-se o vermelho devido à sua aplicação em dois ambientes distintos, na garagem e na área externa), o quarto em cinza, a cozinha em amarelo, a sala em laranja e o sistema *Bluetooth* em azul. A codificação por cores facilitou a identificação de falhas durante o processo de construção física do sistema, contribuindo para o sucesso do resultado final.

Figura 24 – Sistema geral com todos os ambientes e *Bluetooth* inclusos



Fonte: Acervo do autor (2023)

Após a conclusão do sistema geral, a próxima etapa do projeto envolveu o desenvolvimento de um aplicativo móvel. Para isso, foi utilizado o *software MIT App Inventor*, uma plataforma de desenvolvimento criada pelo MIT que permite a criação de aplicativos usando programação em blocos. Com o auxílio das ferramentas disponíveis, foi criado o aplicativo denominado *Smart Light* (Luz Inteligente, em Português), que possui duas interfaces distintas: uma destinada a fornecer informações gerais sobre o projeto e outra que funciona como um painel de controle para gerenciar a iluminação.

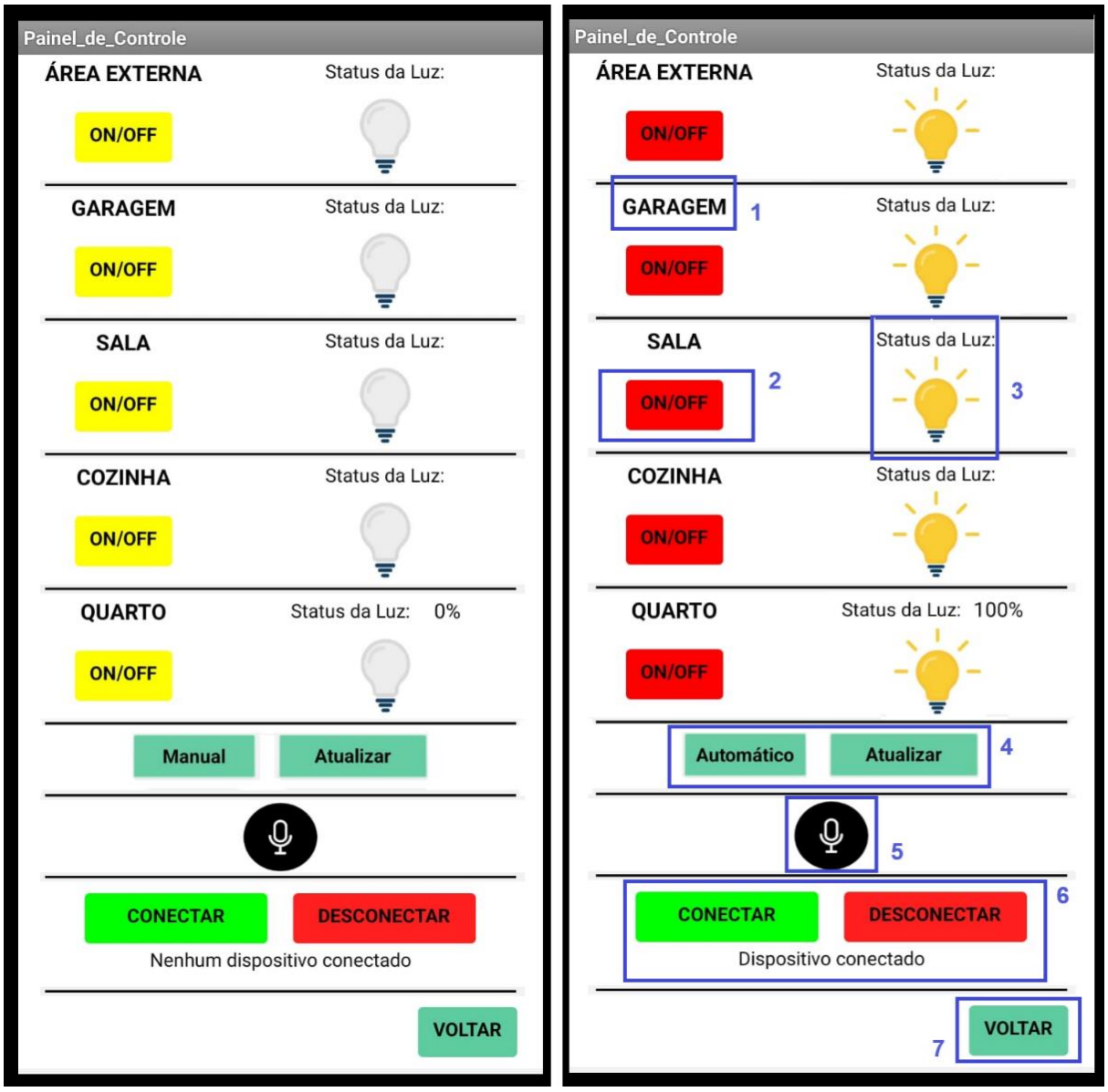
A tela inicial do aplicativo, como mencionado anteriormente, foi desenvolvida para exibir informações sobre o aplicativo e o projeto, como título, nome do discente e orientador(a), além de conter um botão de acesso direto ao painel de controle, conforme ilustrado na Figura 25. A programação desta tela envolveu apenas a funcionalidade do botão 'ACESSAR', que estabelece a conexão com a outra interface.

Figura 25 – Interface inicial do aplicativo *Smart Light*

Fonte: Acervo do autor (2023)

A segunda interface, denominada Painel de Controle, foi criada para englobar todas as funcionalidades de gerenciamento relacionadas à comunicação da iluminação e à conexão com o Arduino UNO, conforme ilustrado na Figura 26. Proporcionando ao usuário, ao acessar essa tela, uma experiência clara e intuitiva no controle da iluminação nos diversos ambientes existentes.

Figura 26 – Interface do Painel de Controle



Fonte: Acervo do autor (2023)

Na Tabela 1, é possível identificar cada elemento da interface de controle da iluminação, compreendendo qual a sua utilidade no aplicativo e a sua funcionalidade resultante da interação com o usuário. Cada um desses componentes, após a fase de *design*, passou pelo processo de implementação da programação para garantir uma execução de suas ações conforme planejado. Dessa forma, com as lógicas de programação estabelecidas, a integração do aplicativo *Smart Light* com o sistema geral criado começou a ser testada, o que possibilitou que as falhas de comunicação entre os dois pudessem ser encontradas e corrigidas ao longo do processo de desenvolvimento.

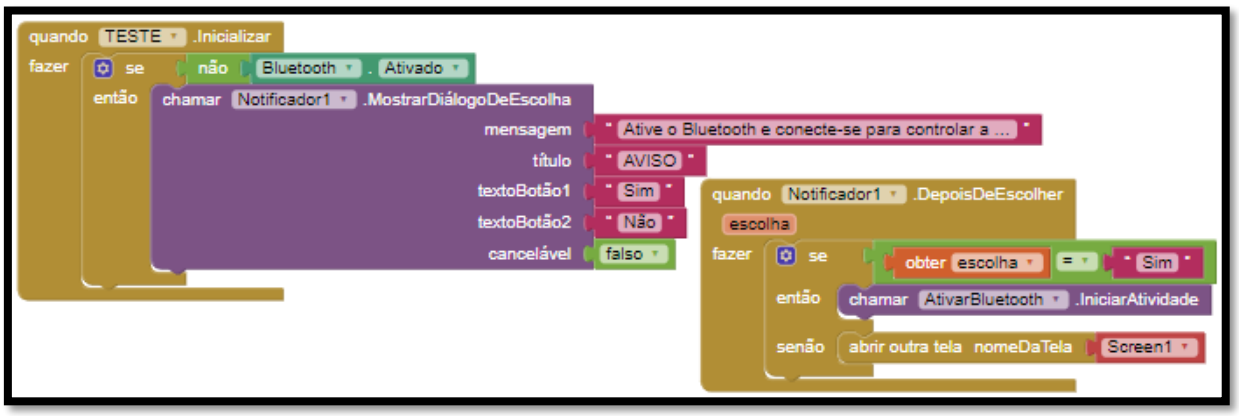
Tabela 1 – Elementos da interface de controle

Item	Descrição
1	Identificação do cômodo.
2	Botão *'ON/OFF'*, que gerencia o estado das luzes. Ao ser pressionado, liga a lâmpada quando o fundo está amarelo e desliga quando o fundo está vermelho.
3	Apresenta o *status* da luz do ambiente de forma intuitiva, através de imagens. No quarto, por ser um cômodo que possibilita a dimerização da iluminação, é incluído a porcentagem da intensidade luminosa da lâmpada.
4	À direita, um botão que permite atualizar o *status* dos ambientes, e à esquerda, que alterna entre os modos de iluminação manual ou automático dos cômodos externos.
5	Botão que possibilita o controle da iluminação através de comandos de voz.
6	Botões que permitem conectar e desconectar o *smartphone* com o módulo HC-05, e abaixo há um texto apresentando o *status* atual do emparelhamento.
7	Botão de voltar, que retorna à interface de inicialização do aplicativo.

Fonte: Acervo do autor (2023).

Inicialmente, foi necessário programar a interface do Painel de Controle ao ser acessada. Para isso, foi criada uma função para verificar o *status* do *Bluetooth* no dispositivo móvel, como ilustrado na Figura 27. A função opera de modo que, se o *Bluetooth* estiver desativado, uma notificação aparece no aplicativo, indicando que é necessário ativá-lo para prosseguir. Se o usuário recusar, será redirecionado à interface inicial. Caso aceite, um iniciador de atividades (ferramenta disponível na plataforma) é acionado para ativar o *Bluetooth* do dispositivo.

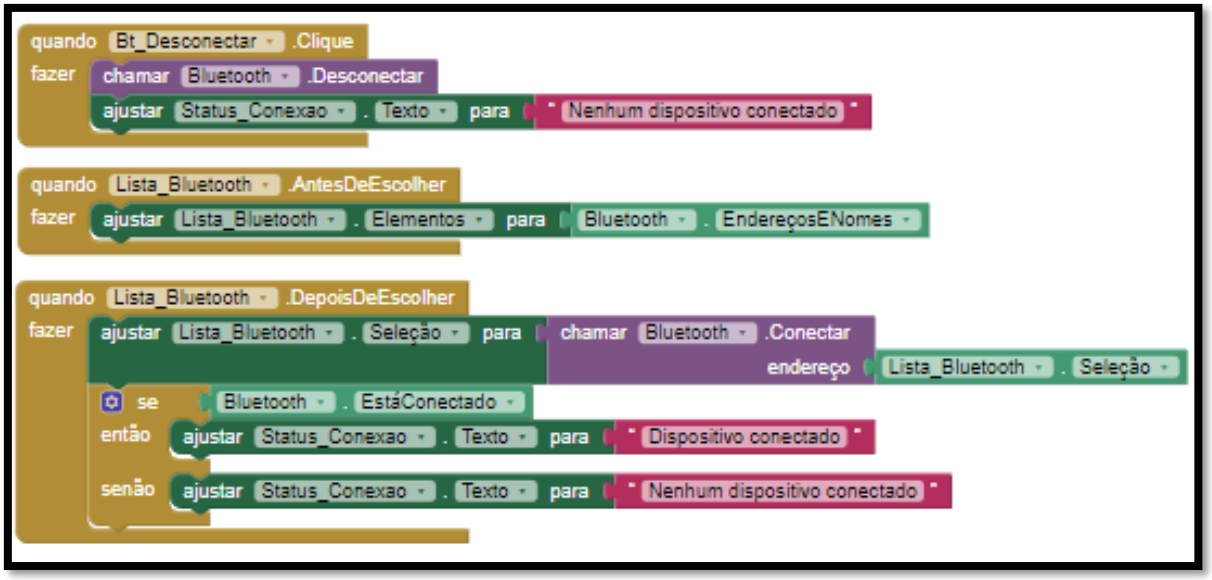
Figura 27 – Programação da inicialização da interface do Painel de Controle



Fonte: Acervo do autor (2023)

A seguir, foram programadas as funções de conexão e desconexão do *Bluetooth* com outros dispositivos, conforme ilustrado na Figura 28. O botão de desconexão ativa a função para desconectar o *Bluetooth*. Por sua vez, a função de conexão apresenta uma lista dos dispositivos *Bluetooth* previamente emparelhados. Antes de ser aberta, essa lista associa automaticamente os dispositivos *Bluetooth* já emparelhados aos itens da lista, permitindo a seleção e conexão. Dependendo da função escolhida, o texto indicando o *status* da conexão é alterado dinamicamente.

Figura 28 – Conectar e Desconectar *Bluetooth*



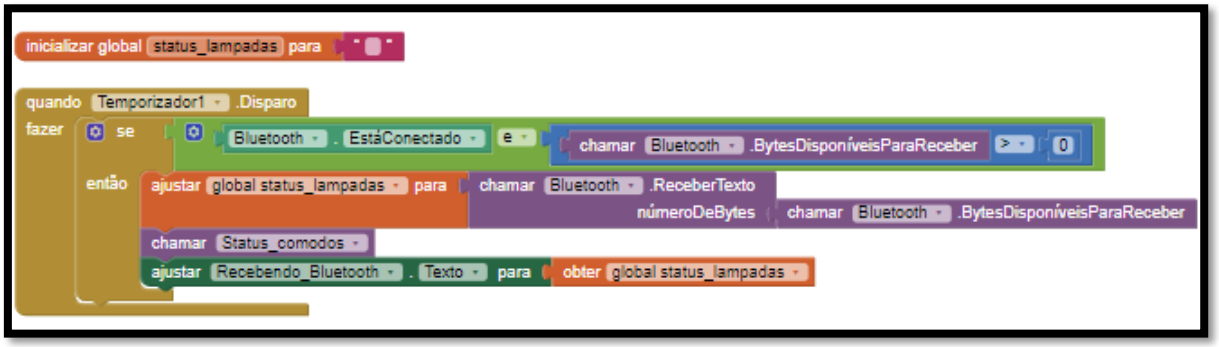
Fonte: Acervo do autor (2023)

Após a implementação das conexões *Bluetooth* no aplicativo, surgiu a preocupação de como as informações sobre a iluminação no sistema geral seriam recebidas. Para resolver isso, a biblioteca *'SoftwareSerial.h*' foi integrada ao código do Arduino, permitindo a comunicação *Bluetooth*. Foi criada uma variável *'bluetooth'* com o auxílio dessa biblioteca para acessar o Módulo HC-05. Dessa forma, foi possível atualizar o *status* dos LEDs. Sempre que o estado era alterado, o comando *'bluetooth.print()*' enviava uma mensagem correspondente à mudança, usando 'a' para área externa, 'g' para garagem, 's' para sala, 'c' para cozinha e 'q' para o quarto seguido pelo número que indica a porcentagem da intensidade luminosa. Quando esses caracteres estavam em letras minúsculas, indicavam que o LED estava desligado; já em maiúsculas, que o LED estava ligado. Para isso, foram necessárias pequenas alterações nas

funções originais ('Escuro', 'Interruptor', *'Dimmer'* e 'PIR'), conforme evidenciado no código final do Arduino presente no Apêndice A.

Para que o aplicativo pudesse receber informações constantes via *Bluetooth*, foi necessário implementar um temporizador, conforme mostrado na Figura 29. Esse temporizador é acionado em intervalos de 50 milissegundos para verificar se a conexão *Bluetooth* está estabelecida e se há *bytes* disponíveis para serem recebidos. Se a verificação for verdadeira, os dados recebidos são atribuídos à variável 'status_lampadas', permitindo análises posteriores. Esse curto intervalo de tempo para o temporizador foi escolhido e configurado de modo a garantir que nenhuma informação seja perdida e possibilitasse uma comunicação mais fluida entre os dispositivos emparelhados.

Figura 29 – Temporizador de atualização dos cômodos via recebimento de dados



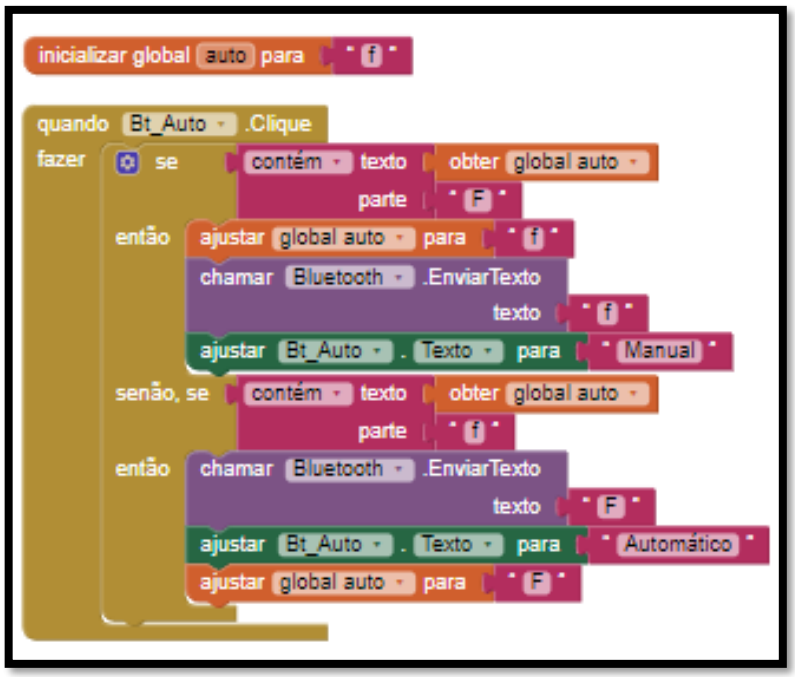
Fonte: Acervo do autor (2023)

Para que o aplicativo pudesse distinguir qual ação tomar ao receber os dados, foi implementado um procedimento, denominado ‘Status_comodos’ que atualiza as configurações de imagem dos *status* das lâmpadas e da cor de fundo do botão para cada tipo de dado recebido, como ilustrado no Apêndice B. Isso permitiu controlar cada ambiente no aplicativo e atualizar seus status de acordo com as interações realizadas no sistema geral.

Após concluir a função de recebimento de dados pelo aplicativo, foram realizados testes nas configurações de envio de dados. Para isso, foi criada uma função no código do Arduino chamada 'Status_Comodos_App', presente no Apêndice C, que utiliza a ferramenta '*bluetooth.available()*' para verificar a presença de dados a serem recebidos pelo Módulo HC-05. Em seguida, por meio de '*bluetooth.read()*', os dados são lidos e transmitidos para uma variável de apoio, permitindo assim a análise e execução das ações necessárias. Esta função foi incorporada ao '*void loop()*' do Arduino, garantindo que a verificação dos dados ocorra de forma contínua.

Com a configuração de envio de dados concluída, abriu-se um leque de oportunidades para aprimorar o sistema geral, tornando o controle da iluminação mais eficiente e flexível. Desta forma, a primeira necessidade observada estava ligada ao gerenciamento dos cômodos externos (área externa e garagem), tendo em vista que na programação inicial, o controle da iluminação externa ocorria apenas durante a noite. Assim, foi implementado um botão no aplicativo capaz de alternar entre os modos 'manual' e 'automático'. No primeiro, o aplicativo envia o caractere 'f' para o Arduino, enquanto no segundo, é enviado o caractere 'F', conforme ilustrado na Figura 30.

Figura 30 – Programação do botão 'AUTOMÁTICO/MANUAL'



Fonte: Acervo do autor (2023)

Ao pressionar o botão e habilitar o modo 'manual', os interruptores pulsadores dos cômodos externos podiam ser utilizados independentemente da luminosidade externa. Por outro lado, ao habilitar o modo 'automático', o funcionamento dos interruptores era desativado, e as luzes eram controladas apenas em condições de baixa luminosidade externa, ou seja, acendiam quando estava escuro na área externa, e o sensor de presença passava a controlar a lâmpada da garagem. Por fim, para preservar as configurações de 'manual' e 'automático' no aplicativo, foi criada uma variável global denominada 'auto', na qual o caractere do botão é armazenado.

Na programação do Arduino, foi necessário a criação de estruturas de controle associadas aos caracteres enviados através do botão 'automático/manual' e às funções

'Escuro()', 'PIR()' e 'Interruptor()', como exemplificado na Figura 31. Estas condicionais foram fundamentais para a execução das funções conforme detalhado anteriormente. E com o intuito de otimizar o consumo de energia, o sensor PIR tem sua alimentação regulada pelos modos automático e manual. No modo primeiro, o Arduino envia um sinal *HIGH* para a porta de saída conectada ao pino de alimentação do sensor, enquanto no modo manual, é enviado um sinal *LOW* para a mesma porta. Desta maneira, o sistema identifica quando deve acionar cada uma dessas funcionalidades com base nos caracteres recebidos via *Bluetooth*, e realizar o controle da iluminação ambiente.

Figura 31 – Associação dos caracteres enviados pelo botão e às funções locais de cada um dos cômodos

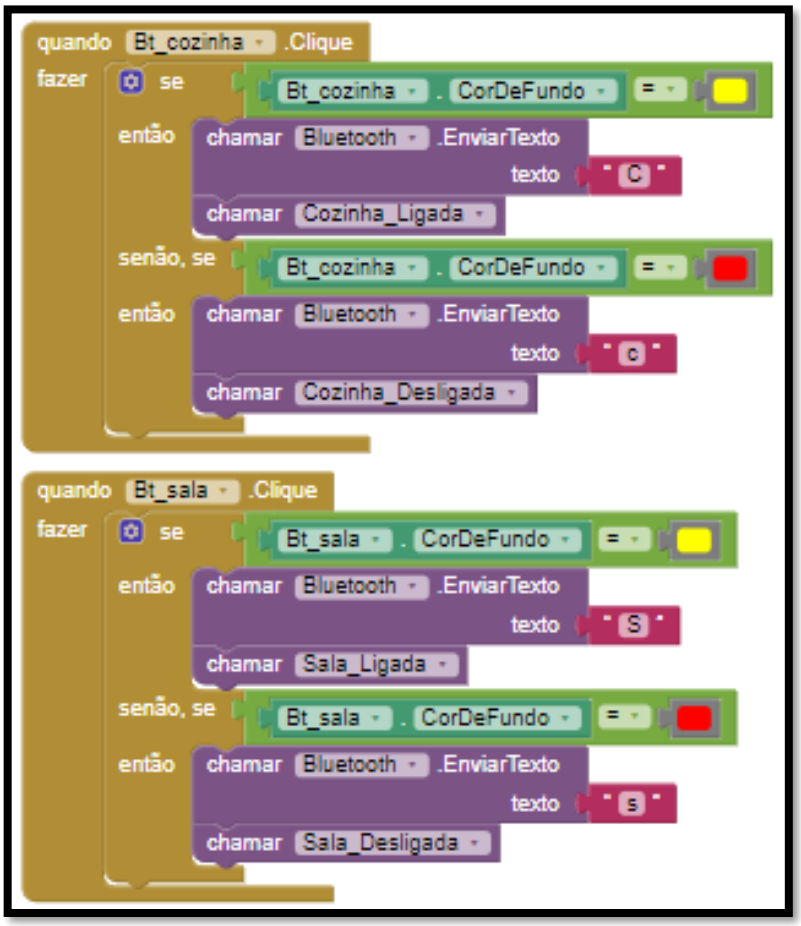
```
//Análise do estado do botão 'automático/manual'
if(App_Comodos == 'F'){
        //Iluminação Automática
        //Alimenta o sensor PIR
        digitalWrite(A_PIR, HIGH);

        //Análise da iluminação ambiente externa
        if(Escuro()){

                //Análise do Sensor de Presença
                if(PIR()){
                        //Liga a lâmpada da Garagem
                        digitalWrite(L_Garagem, HIGH);
                } else {
                        //Desliga a lâmpada da Garagem
                        digitalWrite(L_Garagem, LOW);
                }
                //Liga a lâmpada da Área Externa
                digitalWrite(L_Area, HIGH);
        } else {
                //Desliga a lâmpada da Garagem e da Área
                digitalWrite(L_Garagem, LOW);
                digitalWrite(L_Area, LOW);
        }

} else if (App_Comodos == 'f'){
        //Iluminação Manual
        //Permite interação com os interruptores
        Interruptor_Garagem();
        Interruptor_Area();

        //Corta a alimentação do sensor PIR
        digitalWrite(A_PIR, LOW);
}
```

Fonte: Acervo do autor (2023)

Após testar e confirmar a funcionalidade das comunicações *Bluetooth*, a programação dos botões 'ON/OFF', que controlam diretamente os cômodos, foi iniciada. Começou-se pelos

botões da sala e da cozinha, por esses cômodos estarem interligados apenas aos interruptores pulsadores. Quando pressionados, esses botões verificam a cor de fundo enviando um caractere maiúsculo quando o fundo é amarelo e minúsculo quando é vermelho, conforme mostrado na Figura 32. Posteriormente, um procedimento específico é chamado para cada situação, resultando na mudança da cor de fundo do botão e na atualização da imagem ligada aos *status* do ambiente.

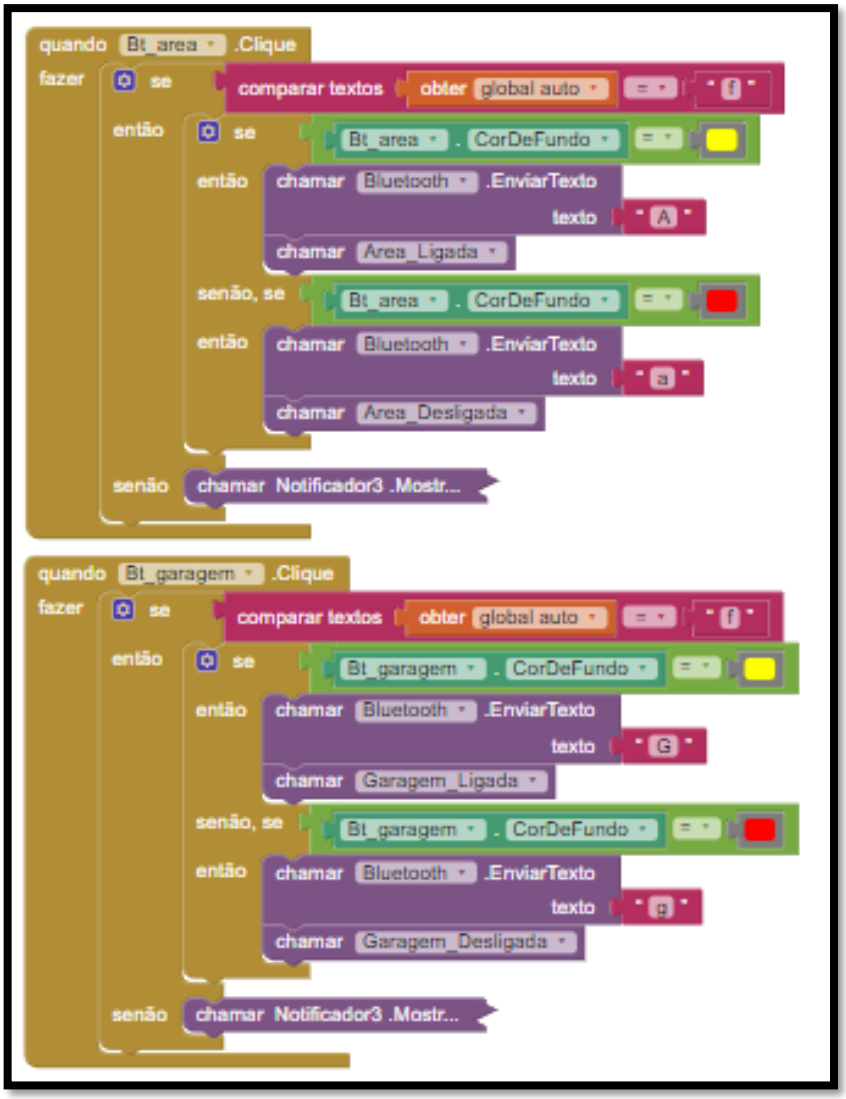
Figura 32 – Programação dos botões da Sala e da Cozinha



Fonte: Acervo do autor (2023)

Os botões da área externa e da garagem foram programados seguindo a mesma lógica anterior. A diferença está no fato de que esses cômodos também interagem com o botão 'automático/manual'; portanto, o usuário só pode interagir com eles quando o modo 'manual' estiver habilitado. Foi utilizado a variável de apoio 'auto', mencionada anteriormente, para determinar qual modo está ativado, conforme mostrado na Figura 33. Se o modo ‘automático’ estiver habilitado e o usuário tentar interagir com os botões, um notificador será exibido comunicando a impossibilidade de interação.

Figura 33 – Programação dos botões da Área Externa e da Garagem

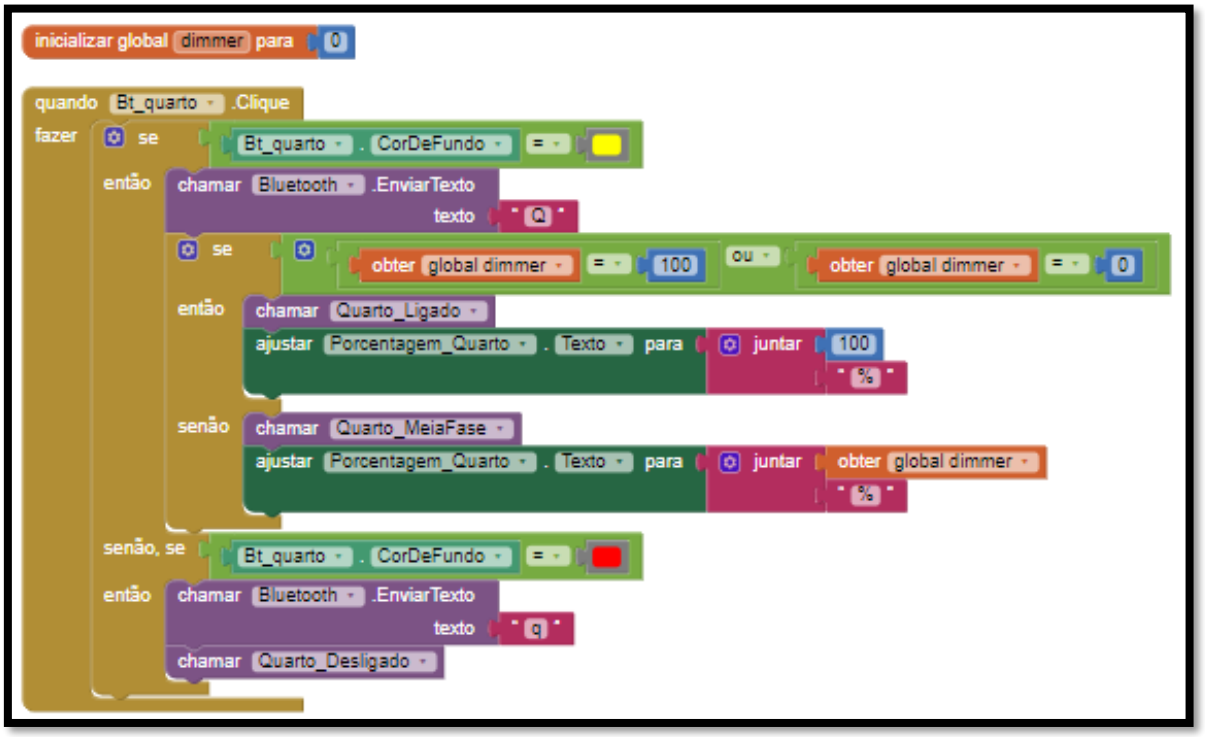


Fonte: Acervo do autor (2023)

O botão do quarto se diferencia dos demais por ser controlado por um potenciômetro que atua como *dimmer*. Para gerenciar essa funcionalidade, conforme ilustrado na Figura 34, foi necessário utilizar uma variável de apoio para armazenar a porcentagem de intensidade luminosa da lâmpada, conforme a interação do usuário com o potenciômetro. Quando o usuário pressiona o botão para ligar a lâmpada, o sistema verifica o valor da variável de apoio. Se for 0, indicando que o cursor do potenciômetro está em sua posição original, a lâmpada é ligada com brilho máximo. Caso contrário, a lâmpada é ligada com o brilho correspondente à posição do cursor do potenciômetro. O processo de desligamento da lâmpada foi programado seguindo a mesma lógica dos demais botões. Após cada interação, o aplicativo atualiza o *status* da lâmpada, indicando a porcentagem de brilho atual e ativando o procedimento correspondente,

sendo: ‘Quarto_Ligado’ para 100%, ‘Quarto_Desligado’ para 0% e ‘Quarto_MeiaFase’ para os demais valores.

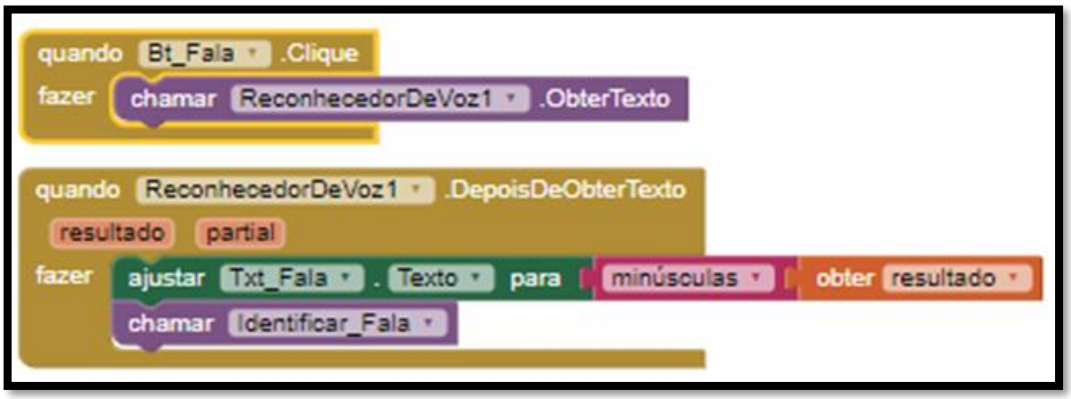
Figura 34 – Programação do botão do Quarto



Fonte: Acervo do autor (2023)

A fim de aprimorar a funcionalidade do aplicativo, optou-se por implementar comandos de voz. Dessa forma, durante a programação, foi utilizado um mecanismo de reconhecimento de voz que era ativado ao pressionar o botão de microfone no aplicativo, conforme ilustrado na Figura 35.

Figura 35 – Programação do botão de comando por voz

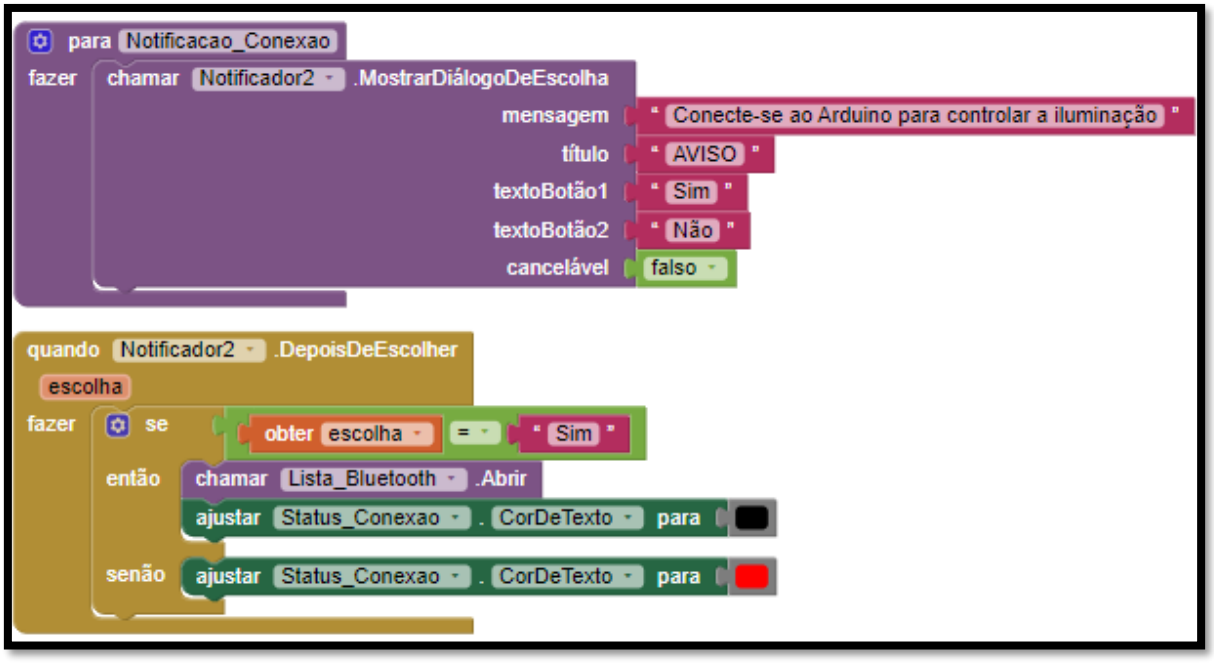


Fonte: Acervo do autor (2023)

Após o comando ser falado, uma legenda de suporte, oculta ao usuário, era atualizada com o texto e convertida em letras minúsculas de forma a facilitar a análise posterior. Com essa legenda atualizada, o texto era interpretado para determinar a ação do aplicativo: ligar/desligar um dos cômodos ou retornar ‘inválido’ caso o comando não fosse reconhecido, como apresentado no Apêndice D.

Considerando que as funções do aplicativo estavam operacionais, embora a conexão via *Bluetooth* ainda não estivesse estabelecida, foi implementado um procedimento que ativava um notificador sempre que se iniciava uma interação com o aplicativo sem que houvesse emparelhamento entre os dispositivos, indicando ao usuário a necessidade de se conectar ao Arduino para controlar a iluminação, conforme apresentado na Figura 36. Se o usuário optasse por 'sim', uma lista com os dispositivos disponíveis para conexão era exibida; caso contrário, a cor do texto indicando o *status* de conexão era alterado para vermelho.

Figura 36 – Procedimento de notificação de conexão não estabelecida

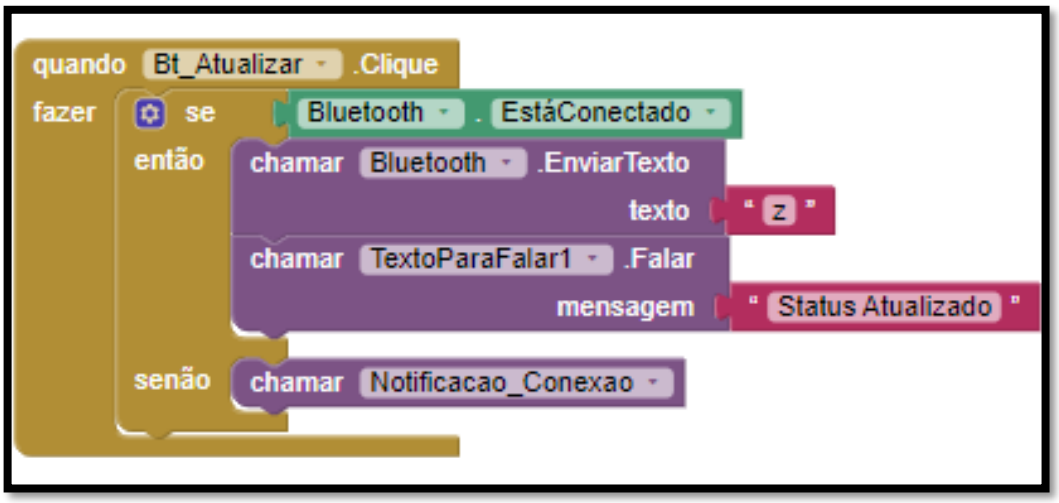


Fonte: Acervo do autor (2023)

Por fim, ao conduzir uma série de testes na comunicação entre o aplicativo e o sistema, notou-se que, em situações de múltiplas ações simultâneas no sistema, o Módulo HC-05 e o aplicativo não conseguiam acompanhar as alterações e as demandas de envio e recebimento de dados. Portanto, foi necessário implementar um botão de atualização para verificar o estado das lâmpadas e atualizar os *status* no aplicativo. Quando o botão é pressionado, é enviado um

caractere 'z' para o Arduino, como demonstrado na Figura 37. O Arduino, ao receber esse caractere, executa uma função chamada 'Att_Comodos', descrita no Apêndice E, que analisa o estado das lâmpadas e envia os dados de volta para o aplicativo.

Figura 37 – Programação do botão de atualização



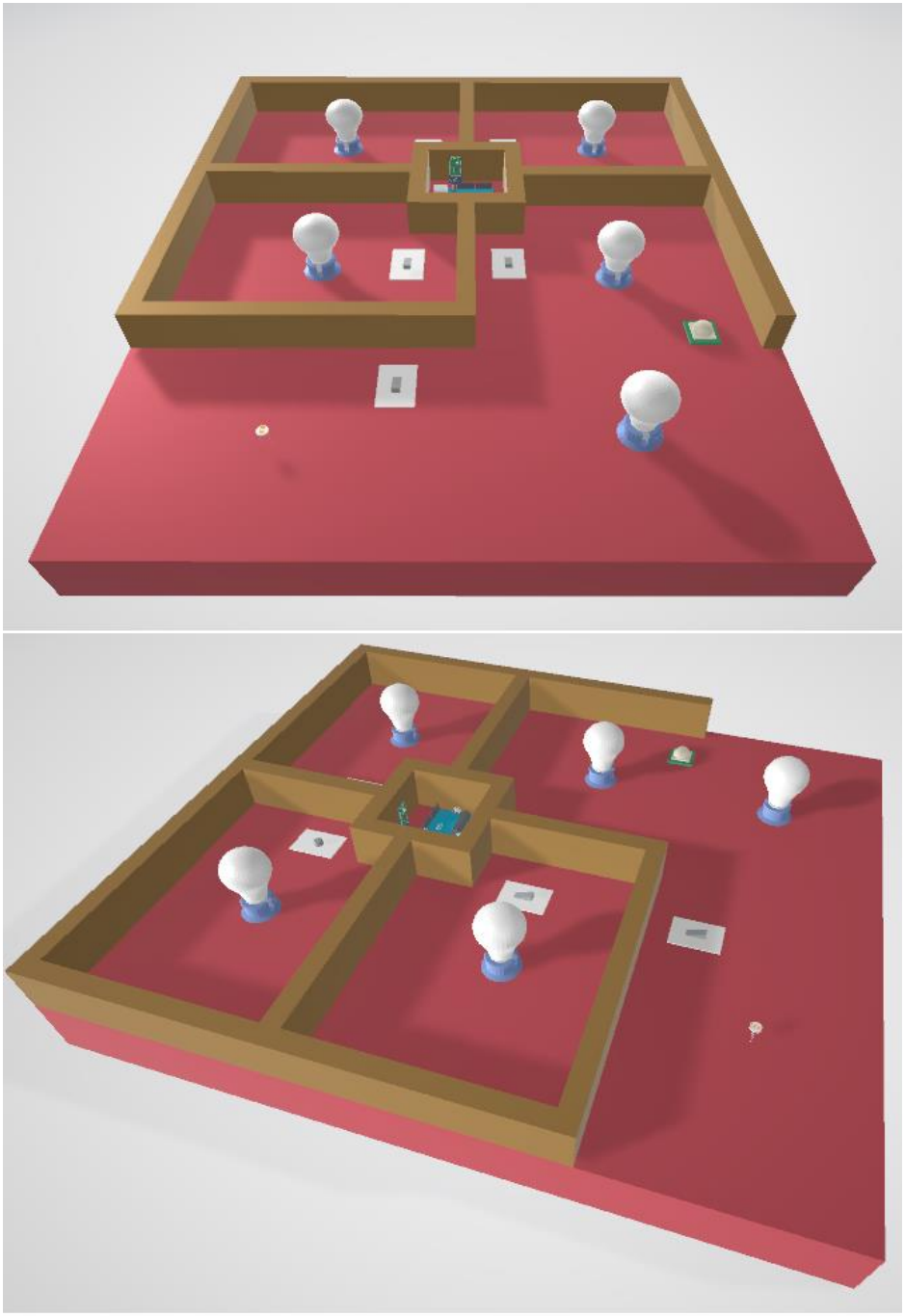
Fonte: Acervo do autor (2023)

Aproveitando a função criada e percebendo que ao conectar com o Módulo HC-05 via *Bluetooth* o aplicativo não atualizava automaticamente o estado dos cômodos, inseriu-se então o envio do caractere 'z' imediatamente após a conexão ser estabelecida com o Arduino, para que assim pudesse ser realizada a atualização do *status* das lâmpadas.

Com o sistema funcionando e interagindo perfeitamente com o aplicativo desenvolvido, avançou-se para a última etapa: a criação da maquete/bancada didática. O objetivo era permitir a interação do sistema com uma rede elétrica residencial de 220V e corrente alternada. Para alcançar esse propósito, foi necessário inserir e alterar alguns componentes para garantir uma melhor compatibilidade entre os principais elementos do sistema.

Inicialmente, foi elaborado um esboço tridimensional da maquete utilizando o *software TinkerCAD*, como mostrado na Figura 38. Nesse esboço, as lâmpadas foram posicionadas para substituir os LEDs, os interruptores pulsadores foram incorporados em vez dos *push buttons* e o *dimmer* rotativo foi introduzido no lugar do potenciômetro. A maquete foi estruturada com divisórias para representar os diferentes cômodos da residência, enquanto os controladores foram colocados no centro, proporcionando uma representação visual do funcionamento do sistema dentro de um ambiente residencial.

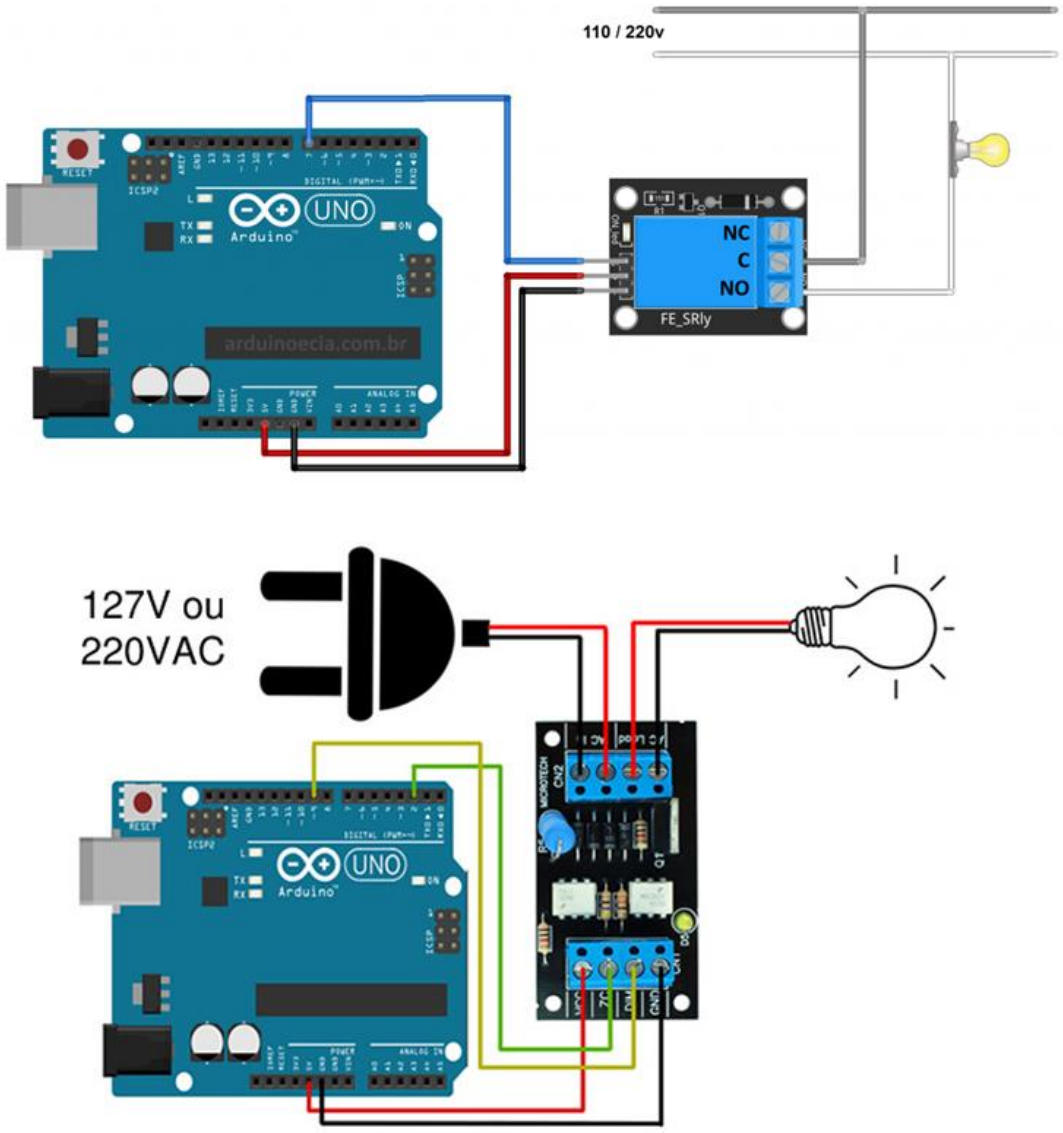
Figura 38 – Esboço tridimensional de um ambiente residencial



Fonte: Acervo do autor (2023)

Em relação às alterações necessárias, o processo começou com as lâmpadas, que foram conectadas nos módulos relé e à rede residencial, como ilustrado na Figura 39. Anteriormente, o sinal de saída de cada cômodo estava ligado ao LED; agora, eles estão conectados ao relé. Esse ajuste permitiu que o relé, ao receber o sinal correspondente, controlasse a abertura ou fechamento da conexão elétrica, acendendo ou desligando a lâmpada conforme necessário, ou, no caso do controle de iluminação do quarto, fosse permitido a dimerização da iluminação.

Figura 39 – Conexão das lâmpadas no novo sistema



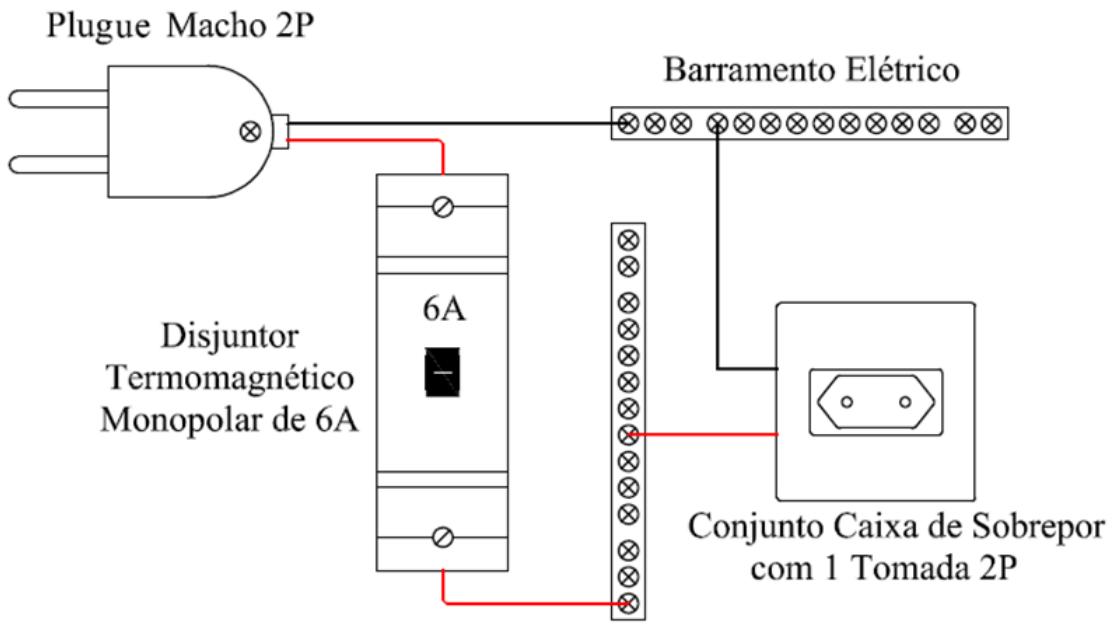
Fonte: Adaptado de Arduino e Cia (2013) e USINAINFO (2023)

É importante ressaltar que, para a funcionalidade de dimerização do quarto fosse possível, a lâmpada instalada no ambiente precisava ser do tipo dimerizável, podendo ser incandescente, halógena ou mesmo LED, desde que permitisse a variação da intensidade luminosa. Além disso, foi necessário inserir no código o comando '*attachInterrupt()*', que emite um sinal para o pino *Zero Cross* do MC-8A, interrompendo brevemente a alimentação da lâmpada. Mas como a frequência da rede elétrica é de 60Hz, essas interrupções são extremamente breves, na ordem dos microssegundos, tornando-as imperceptíveis aos nossos olhos, sendo que a única alteração visível é o ajuste no brilho da lâmpada.

Quanto à integração dos interruptores pulsadores e do *dimmer* rotativo no novo sistema, a substituição foi realizada apenas trocando o *push button* e o potenciômetro, sem a necessidade de ajustes significativos. Todos esses componentes foram conectados a barras elétricas alimentadas pelo Arduino e, para garantir uma alimentação estável e minimizar as flutuações de tensão, foi adicionado um capacitor de 1000µF entre os pinos 5V e GND do Arduino.

Para fornecer energia ao sistema, como ilustrado na Figura 40, um plugue macho de 2 pinos foi usado para simular a entrada da rede elétrica em uma residência. Um dos pinos do *plug* foi conectado a um barramento elétrico com o auxílio de cabos, enquanto o outro foi ligado em série a um disjuntor termomagnético monopolar de 6A para garantir a proteção do sistema e, em seguida, conectado a outro barramento elétrico. Com a alimentação da maquete pronta, a conexão das lâmpadas foi realizada pelos barramentos elétricos utilizados. Já a alimentação externa para o Arduino UNO foi assegurada inserindo um conjunto de caixa de sobrepor com 1 tomada 2P, ligado aos barramentos, para possibilitar a conexão de uma fonte de 12V DC com pino P4 de saída e alimentar o Arduino. Assim, a representação da rede elétrica residencial com as lâmpadas e a alimentação do Arduino foi efetuada de maneira precisa.

Figura 40 – Esquema da alimentação externa



Fonte: Acervo do autor (2023)

Dessa maneira, a metodologia adotada permitiu o desenvolvimento de um sistema de iluminação inteligente para residências, integrando conceitos de automação residencial, sensores e o módulo Arduino. O aplicativo móvel foi desenvolvido de forma intuitiva e de fácil

utilização, possibilitando o controle eficiente e personalizado da iluminação. Além da estrutura garantir a realização adequada dos procedimentos necessários para alcançar os objetivos do trabalho.

5 RESULTADOS E DISCUSSÕES

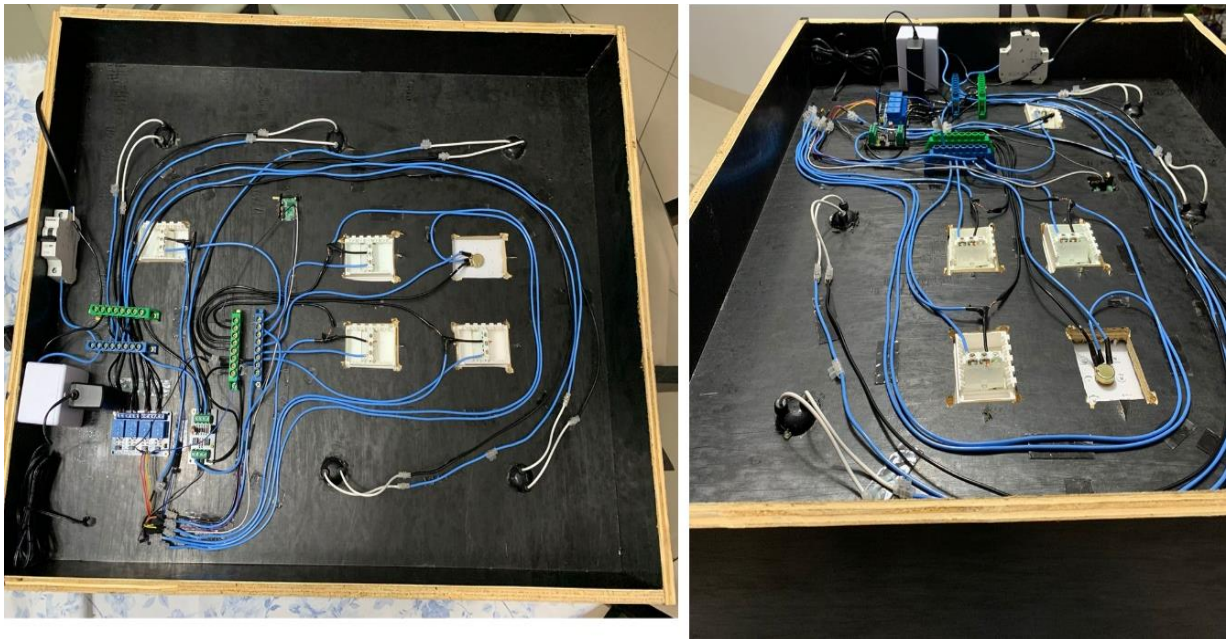
O projeto final compreende de um sistema e uma maquete residencial com dimensões de 60x70x25cm, conforme ilustrado na Figura 41. A maquete é composta por cinco ambientes distintos: área externa, garagem, sala, cozinha e quarto. Enquanto este último oferece a funcionalidade de dimerização da iluminação, os demais possuem opções convencionais de ligar e desligar as luzes. Cada ambiente é dotado de interfaces físicas e digitais que permitem o gerenciamento da iluminação, seja por meio de interruptores, *dimmer* rotativo ou aplicativo móvel. Além disso, os espaços externos, área externa e garagem, proporcionam a opção de controle automático, respondendo às variações de luminosidade externa.

Figura 41 – Maquete residencial

Fonte: Acervo do autor (2023)

A fiação elétrica do sistema foi estrategicamente instalada sob a maquete, como evidenciado na Figura 42, de forma a permanecer oculta e, assim, aprimorar a estética geral do ambiente.

Figura 42 – Fiação elétrica na maquete



Fonte: Acervo do autor (2023)

Após a conclusão do sistema e da maquete, inúmeros testes foram realizados, cujos resultados atenderam às expectativas definidas pela proposta inicial, demonstrando não apenas a visibilidade técnica da integração dos elementos da automação no âmbito da gestão de iluminação, mas também a sua aplicabilidade prática em um contexto residencial.

No entanto, é importante destacar algumas limitações que surgiram devido aos dispositivos e *softwares* empregados durante o desenvolvimento do projeto e que podem impactar uma eventual implementação em um ambiente real. Primeiramente, o módulo *Bluetooth* utilizado permite uma comunicação eficaz em até 10 metros, o que pode dificultar o gerenciamento dependendo das extensões da residência e do posicionamento do dispositivo. Além disso, a plataforma *MIT App Inventor*, apesar de ser um *software* educacional robusto, apresenta desafios relacionados à estética visual em dispositivos com diferentes dimensões de tela, bem como dificuldades na interação de variáveis durante a transição entre interfaces no aplicativo. Vale também ressaltar que o uso do Arduino UNO, devido ao número limitado de portas disponíveis para comunicação, impõe restrições à quantidade de ambientes que podem

ser integrados ao sistema. Por fim, devido à natureza do comando '*attachInterrupt()*', que gera interrupções quando determinadas condições são atendidas, ocorrem interferências na comunicação do sistema quando a iluminação do quarto está parcialmente acesa, resultando em atrasos na execução das funções.

Apesar das questões mencionadas, é gratificante constatar que o ambiente opera perfeitamente dentro de suas limitações, seguindo precisamente as diretrizes estabelecidas ao longo do trabalho realizado, e permite levar conforto e praticidade aos usuários.

6 CONCLUSÃO

O estudo realizado buscou explorar as vastas potencialidades da automação no contexto da iluminação inteligente. Destacou-se, assim, a importância contínua da pesquisa e inovação no campo da automação residencial, que avança incessantemente, transformando residências em experiências dinâmicas, onde a inteligência e a tecnologia são os pilares fundamentais.

O sistema desenvolvido teve como propósito integrar dispositivos simples com *smartphones*, que já são amplamente utilizados em muitos lares, destacando que devido à crescente demanda no mercado global de automação, sua integração em residências está se tornando cada vez mais acessível e abrindo um leque variado de oportunidades, resultando em ambientes mais convenientes e eficientes, que auxiliam as pessoas em suas tarefas diárias e promovem um maior conforto e praticidade no dia a dia.

Como sugestão para pesquisas futuras, recomenda-se implementar este sistema de iluminação inteligente em uma residência real, buscando analisar os pontos positivos e negativos em um ambiente prático. Esta abordagem permitiria uma compreensão mais aprofundada do desempenho do sistema em condições do mundo real, considerando variáveis como o uso contínuo, a interação com os moradores e a resposta a situações do cotidiano. Além disso, essa implementação em um ambiente residencial proporcionaria uma avaliação mais holística da eficácia do sistema, destacando aspectos práticos, comportamentais e de usabilidade que podem não ser totalmente capturados em ambientes controlados de laboratório. Essa análise mais abrangente contribuiria para refinamentos contínuos e para o desenvolvimento de sistemas de automação residencial que atendam de maneira mais precisa e eficiente às necessidades e expectativas dos usuários.

REFERÊNCIAS BIBLIOGRÁFICAS

ACCARDI, Adonis; DODONOV, Eugeni. Automação residencial: elementos básicos, arquiteturas, setores, aplicações e protocolos. **Revista TIS**, v. 1, n. 2, 2012.

ALECRIM, Emerson; MARQUES, Ana. O que é um LED? Entenda como funciona essa tecnologia de iluminação. **Tecnoblog.** 2023. Disponível em: < https://tecnoblog.net/responde/o-que-e-led/#:~:text=Como%20funcionam%20os%20LEDs,-A%20estrutura%20b%C3%A1sica&text=O%20LED%20gera%20luz%20a,energia%20que%20comp%C3%B5e%20a%20luz).>. Acesso em: 19 set. 2023.

ALMEIDA, Reginaldo Apolinario de; RALL, Ricardo. PROTÓTIPO DE ILUMINAÇÃO RESIDENCIAL UTILIZANDO DISPOSITIVOS MÓVEIS E ARDUINO. In: **4ª Jornada Científica e Tecnológica da FATEC de Botucatu**, 2015. Disponível em: <http://www.jornacitec.fatecbt.edu.br/index.php/IVJTC/IVJTC/paper/view/353>.

ALVES, Pedro. LDR – o que é e como funciona! **Manual da Eletrônica**, 2020. Disponível em: <https://www.manualdaeletronica.com.br/ldr-o-que-e-como-funciona/>. Acesso em: 18 set. 2023.

ARDUINO.CC. Problemas com módulo HC-05 e Arduino MEGA. **Arduino.cc.** 2020. Disponível em: <https://forum.arduino.cc/t/problemas-com-modulo-hc-05-e-arduino-mega/653977>. Acesso em: 10 nov. 2023.

ARDUINO E CIA. Ligando uma lâmpada com módulo relé Arduino. **ARDUINO e cia.** 2013. Disponível em: <https://www.arduinoecia.com.br/ligando-uma-lampada-com-modulo-rele-arduino/>. Acesso em: 26 out. 2023.

BAHARUM, Zirawani et al. Mobile-Based Application: The Designation of Energy Saving Smart Light System for Monitoring and Controlling. **International Journal of Interactive Mobile Technologies**, v. 15, n. 18, 2021.

BAT. Pulsador Lâmpada gravada com placa. **B.A.T. Soluções em Eletricidade.** 2023. Disponível em: <https://www.batsolucoes.com.br/pulsador-lampada-gravada-com-placa/p/4328 >. Acesso em: 26 set. 2023.

CAMARGO, Valter Luís Arlindo de. **Elementos de Automação.** 1. ed. São Paulo: Érica, 2014. Capítulo 2: Sistemas de Controle. p. 48-72.

DIAS, C. L. de A.; PIZZOLATO, N. D. Domótica: Aplicabilidade e Sistemas de Automação Residencial. **Revista Vértices**, *[S. l.]*, v. 6, n. 3, p. 9–32, 2010. Disponível em: <https://editoraessentia.iff.edu.br/index.php/vertices/article/view/1809-2667.20040015>. Acesso em: 17 set. 2023.

ELETROGATE. O que é Arduino: Para que Serve, Vantagens e como Utilizar. **ELETROGATE.** 2020. Disponível em: < https://blog.eletrogate.com/o-que-e-arduino-para-que-serve-vantagens-e-como-utilizar/>. Acesso em: 21 set. 2023.

ELETRU'S. ARDUINO MEGA 2560 R3. **Eletru's Componentes Eletrônicos.** 2023. Disponível em: <https://www.eletruscomp.com.br/produto/arduino-mega-2560-r3/>. Acesso em 15 set. 2023.

EXPOLUX. Mercado de iluminação inteligente deve chegar a US$46,90 bi. 2022. **Feira Internacional da Indústria da Iluminação**. Disponível em: <https://blog.expolux.com.br/2022/02/02/ilumdec-mercado-de-iluminacao-inteligente/>. Acesso em: 07 jun. 2023.

FERNANDES, Fabio. **Sistema de Iluminação Inteligente através de Redes de Sensores Wireless.** 2011. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) – Universidade Estadual Paulista, Guaratinguetá, 2011.

INSTITUTO FEDERAL DE SANTA CATARINA. AULA 3 - Microcontroladores - Técnico. **Wiki - Instituto Federal de Santa Catarina**, 2018. Disponível em: <https://wiki.ifsc.edu.br/mediawiki/index.php/AULA_3_-_Microcontroladores_-_T%C3%A9cnico>. Acesso em: 19 de set. 2023.

LEITE, Ana Luiza Ferreira. **Iluminação e Automação Residencial: análises e aplicações em busca de conforto lumínico.** 2023. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Controle e Automação) – Universidade Federal de Ouro Preto, Ouro Preto, 2023.

LEROY MERLIN. Variador de Luminosidade. **LEROY MERLIN**, 2023. Disponível em: <https://www.leroymerlin.com.br/variador-de-luminoosidade>. Acesso em: 26 set. 2023.

MATTEDE, Henrique. Dimmer para LED! Como escolher. **Mundo da Elétrica**, 2020. Disponível em: < https://www.mundodaeletrica.com.br/dimmer-para-led-como-escolher/>. Acesso em: 26 set. 2023.

MATTEDE, Henrique. Potenciômetro digital. Características e aplicações! **Mundo da Elétrica**, 2019. Disponível em: <https://www.mundodaeletrica.com.br/potenciometro-digital-caracteristicas-e-aplicacoes/>. Acesso em: 17 set. 2023

MURATORI, J. R.; DAL BÓ, P. H. **Automação Residencial:** Conceitos e Aplicações. Curitiba: Educere, 2013. Capítulo 1: Automação residencial. p. 70-77.

RATH, Deepak Kumar. Arduino Based: Smart Light Control System. **International Journal of Engineering Research and General Science**, Bhubaneswar, v. 4, 2ª Edição, 2016.

ROBOCORE. Sensor de Presença PIR – HC-SR501. **ROBOCORE**, 2008. Disponível em: <https://www.robocore.net/sensor-ambiente/sensor-de-presenca-pir-hc-sr501>. Acesso em: 18 set. 2023.

RUPPEL, Alexandre; UNRUH, Fábio; UNRUH, Ricardo Henrique. **Protótipo de um sistema de iluminação residencial com controle remoto sem fio: wi-fi**. 2013. Trabalho de Conclusão de Curso (Bacharelado de Tecnologia em Gestão Comercial Elétrica e Bacharelado de Tecnologia em Eletrônica). Universidade Tecnológica Federal do Paraná, Curitiba, 2013.

SILVA, Davyson Domingos da et al. **Automação Residencial com Arduino:** Utilizando Bluetooth e Android. Cabeledo: UNIESP. 2019.

SOARES, Guilherme Marcio. **SISTEMA INTELIGENTE DE ILUMINAÇÃO DE ESTADO SÓLIDO COM CONTROLE REMOTO E ANÁLISE DE PARÂMETROS DA REDE ELÉTRICA.** 2014. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Federal de Juiz de Fora, Juiz de Fora, 2014.

SOCOLOWISKI, Otávio Paixão. **SISTEMA DE CONTROLE DE AUTOMAÇÃO RESIDENCIAL (S.C.A.R).** 2022. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) – Centro Universitário de Goiás, Goiânia, 2022.

SOUSA NETO, Paulo João de. **AUTOMAÇÃO RESIDENCIAL UTILIZANDO ARDUINO.** 2021. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Pontifícia Universidade Católica de Goiás, Goiânia, 2021.

SOUXA, Fábio. Arduino UNO. **EMBARCADOS.** 2013. Disponível em: <https://embarcados.com.br/arduino-uno/>. Acesso em: 21 set. 2023.

SOUZA, Fábio. Arduino MEGA 2560. **EMBARCADOS.** 2014. Disponível em: <https://embarcados.com.br/arduino-mega-2560/>. Acesso em: 19 set. 2023.

STA. Como utilizar o módulo relé com Arduino. **STA – Sistema e Tecnologia Aplicada**. 2023. Disponível em: <https://www.sta-eletronica.com.br/artigos/arduinos/como-utilizar-o-modulo-rele-com-arduino#:~:text=O%20m%C3%B3dulo%20rel%C3%A9%20%C3%A9%20um,desligar%20dispositivos%20de%20alta%20tens%C3%A3o.>. Acesso em: 19 set. 2023.

TEZA, Vanderlei Rabelo. **ALGUNS ASPECTOS SOBRE A AUTOMAÇÃO RESIDENCIAL - DOMÓTICA.** 2002. Dissertação (Mestrado em Ciências da Computação) – Universidade Federal de Santa Catarina, Florianópolis, 2002.

USINAINFO. Módulo Dimmer para Arduino e ESP32 MC-8A com sinal Zero Cross 220V. **USINAINFO – Eletrônica & Robótica.** 2023. Disponível em: <https://www.usinainfo.com.br/dimmer-arduino/modulo-dimmer-para-arduino-pic-mc-8a-com-sinal-zero-cross-220v-5915.html?search_query=modulo+dimmer&results=9>. Acesso em 29 set. 2023.

VIANA, Carol Correia. Como utilizar o módulo Bluetooth HC-05 com Arduino. **Blog da Robótica.** 2023. Disponível em: <https://www.blogdarobotica.com/2023/02/13/como-utilizar-o-modulo-bluetooth-hc-05-com-arduino/>. Acesso em: 19 set. 2023.

APÊNDICE A – CÓDIGO FINAL DO ARDUINO

```
#include <SoftwareSerial.h>
//biblioteca de conectividade bluetooth

SoftwareSerial bluetooth(10, 11);
//objeto bluetooth, TX na porta 10, RTX na porta 11

const int L_Sala = 7;         //lâmpada da sala
const int Bt_Sala = A2;       //botão da sala
const int L_Cozinha = 6;      //lâmpada da cozinha
const int Bt_Cozinha = A4;    //botão da cozinha
const int L_Quarto = 5;       //lâmpada do quarto
const int Pot_Quarto = A0;    //potenciômetro/dimmer do quarto
const int L_Garagem = 8;      //lâmpada da garagem
const int Bt_Garagem = A5;    //botão da garagem
const int L_Area = 9;         //lâmpada da área externa
const int Bt_Area = A3;       //botão da área externa
const int Fotorresistor = A1; //fotorresistor
const int PIR_Status = 13;    //Status se PIR vai estar ligado ou desligado
const int Sensor_PIR = 12;    //Sinal de saída do PIR

// Variável auxiliar do botão e dimmer
volatile int auxBt_Sala = 0;       //auxiliar para o interruptor da sala
volatile int auxBt_Cozinha = 0;    //auxiliar para o interruptor da cozinha
volatile int auxPot_Quarto = 0;    //auxiliar para o potenciômetro do quarto
volatile int aux_Porcentagem = 0;  //auxiliar para porcentagem do brilho no quarto
volatile int auxBt_Garagem = 0;    //auxiliar para o interruptor da garagem
volatile int auxBt_Area = 0;       //auxiliar para o interruptor da área externa

//Variável auxiliar do Status dos Cômodos
String Status = "";
String St_Comodos;
```

```
32  volatile char App_Comodos = "";
33  char Fot_Auto = 'f';
34  int aux_PIR = 0;
35  int aux2_PIR = 0;
36  int aux_Escuro = 0;
37  volatile int auxBrilho_Quarto = 0;
38  volatile int aux_Quarto = 0;
39  volatile int ZC = 0;
40
41  void setup() {
42    pinMode(L_Sala, OUTPUT);
43    pinMode(L_Cozinha, OUTPUT);
44    pinMode(L_Quarto, OUTPUT);
45    pinMode(L_Garagem, OUTPUT);
46    pinMode(L_Area, OUTPUT);
47    pinMode(13, OUTPUT);
48    pinMode(PIR_Status, OUTPUT);
49    Serial.begin(9600);
50    bluetooth.begin(9600);
51    attachInterrupt(0, ZeroCross, RISING);
52  }
53
54  void loop() {
55
56    //Gerenciamento da luz da sala
57    if (Int_Sala()) {
58      digitalWrite(L_Sala, LOW);
59    } else {
60      digitalWrite(L_Sala, HIGH);
61    }
62
63    //Gerenciamento da luz da cozinha
64    if (Int_Cozinha()) {
```

```
65        digitalWrite(L_Cozinha, LOW);
66      } else {
67        digitalWrite(L_Cozinha, HIGH);
68      }
69
70      //Gerenciamento da luz da área e garagem
71      if(Fot_Auto == 'F'){
72        digitalWrite(PIR_Status, HIGH);
73        if(Escuro()){
74          digitalWrite(L_Area, HIGH);
75          if(PIR()){
76            digitalWrite(L_Garagem, HIGH);
77          }else{
78            digitalWrite(L_Garagem, LOW);
79          }
80        }else{
81          digitalWrite(L_Garagem, LOW);
82          digitalWrite(L_Area, LOW);
83        }
84      }
85        if(Fot_Auto == 'f'){
86        digitalWrite(PIR_Status, LOW);
87        if(Int_Garagem()){
88          digitalWrite(L_Garagem, LOW);
89        }else{
90          digitalWrite(L_Garagem, HIGH);
91        }
92
93        if(Int_Area()){
94          digitalWrite(L_Area, LOW);
95        }else{
96          digitalWrite(L_Area, HIGH);
97        }
```

```
98      }
99
100     Dimmer_Quarto();    //Gerenciamento da luz do quarto
101     Status_Comodos_App();   //Recebimento de dados via Bluetooth
102   }
103
104   // função interruptor da sala
105   bool Int_Sala() {
106     if (analogRead(Bt_Sala) > 400) {
107       auxBt_Sala ++;
108       if (auxBt_Sala % 2 == 0) {
109         Status = "s";
110         bluetooth.print(Status);
111       } else {
112         Status = "S";
113         bluetooth.print(Status);
114       }
115       delay(300);
116     }
117
118     if (auxBt_Sala % 2 == 0) {
119       return true;
120     } else {
121       return false;
122     }
123   }
124
125   // função interruptor da cozinha
126   bool Int_Cozinha() {
127     if (analogRead(Bt_Cozinha) > 400) {
128       auxBt_Cozinha ++;
129       if (auxBt_Cozinha % 2 == 0) {
130         Status = "c";
```

```
131         bluetooth.print(Status);
132       } else {
133         Status = "C";
134         bluetooth.print(Status);
135       }
136       delay(300);
137     }
138 
139     if (auxBt_Cozinha % 2 == 0) {
140       return true;
141     } else {
142       return false;
143     }
144   }
145 
146   // função interruptor da garegem
147   bool Int_Garagem() {
148     if (analogRead(Bt_Garagem) > 400) {
149       auxBt_Garagem ++;
150       if (auxBt_Garagem % 2 == 0) {
151         Status = "g";
152         bluetooth.print(Status);
153       } else {
154         Status = "G";
155         bluetooth.print(Status);
156       }
157       delay(300);
158     }
159 
160     if (auxBt_Garagem % 2 == 0) {
161       return true;
162     } else {
163       return false;
```

```
164     }
165   }
166
167   //função leitura de saída do sensor PIR
168   bool PIR(){
169     aux_PIR = digitalRead(Sensor_PIR);
170     if (aux_PIR == 1){
171       if(aux2_PIR == 0){
172         Status = "G";
173         bluetooth.print(Status);
174     }
175     aux2_PIR = aux_PIR;
176     return true;
177     }else if(aux_PIR == 0){
178       if(aux2_PIR == 1){
179         Status = 'g';
180         bluetooth.print(Status);
181       }
182       aux2_PIR = aux_PIR;
183       return false;
184     }
185   }
186
187   // função interruptor da área externa
188   bool Int_Area() {
189     if (analogRead(Bt_Area) > 400) {
190       auxBt_Area ++;
191       if (auxBt_Area % 2 == 0) {
192         Status = "a";
192         bluetooth.print(Status);
193       } else {
194         Status = "A";
195         bluetooth.print(Status);
```

```
196       }
197       delay(300);
198     }
199
200     if (auxBt_Area % 2 == 0) {
201       return true;
202     } else {
203       return false;
204     }
205   }
206
207   // função luminosidade do ambiente
208   bool Escuro() {
209     if (analogRead(Fotorresistor) > 900) {
210       if(aux_Escuro == 0){
211         Status = "A";
212         bluetooth.print(Status);
213         aux_Escuro = 1;
214         delay(300);
215       }
216       return true;
217     } else{
218       if(aux_Escuro == 1){
219         Status = "g";
220         bluetooth.print(Status);
221         delay(300);
222         Status = "a";
223         bluetooth.print(Status);
224         aux_Escuro = 0;
225         delay(300);
226       }
227       return false;
228     }
```

```
229  }
230
231  // função dimmer do quarto
232  void Dimmer_Quarto() {
233    // Lê o valor atual do potenciômetro
234    auxPot_Quarto = analogRead(Pot_Quarto);
235
236    // Mapeia o valor do potenciômetro para o intervalo de brilho da lâmpada (0 a 100)
237    aux_Porcentagem = map(auxPot_Quarto, 0, 1023, 0, 100);
238    if(aux_Porcentagem > 90){
239      if(auxBrilho_Quarto != 100){
240        ZC = 2;
241        Status = "Q";
242        bluetooth.print(Status);
243        auxBrilho_Quarto = 100;
244        delay(300);
245      }
246    } else if(aux_Porcentagem < 10){
247      if(auxBrilho_Quarto != 0){
248        ZC = 0;
249        Status = "q";
250        bluetooth.print(Status);
251        auxBrilho_Quarto = 0;
252        delay(300);
253      }
254    }else{
255      if(abs(auxBrilho_Quarto - aux_Porcentagem)>9){
256        ZC = 1;
257        auxBrilho_Quarto = aux_Porcentagem;
258        aux_Quarto = aux_Porcentagem - (aux_Porcentagem%10);
259        Status = aux_Quarto;
260        bluetooth.print(Status);
261        delay(100);
```

```
262       }
263     }
264   }
265
266   // lê as interações realizadas no aplicativo (ligar e desligar cômodos)
267   void Status_Comodos_App() {
268     if (bluetooth.available() > 0) {
269       App_Comodos = bluetooth.read();  //realiza a leitura dos dados vindo do aplicativo
270       if (App_Comodos == 'A') {
271         //Serial.println("Área Ligada");
272         digitalWrite(L_Area, HIGH);
273         auxBt_Area ++;
274       } else if (App_Comodos == 'a') {
275         //Serial.println("Área Desligada");
276         digitalWrite(L_Area, LOW);
277         auxBt_Area ++;
278       } else if (App_Comodos == 'G') {
279         //Serial.println("Garagem Ligada");
280         digitalWrite(L_Garagem, HIGH);
281         auxBt_Garagem ++;
282       } else if (App_Comodos == 'g') {
283         //Serial.println("Garagem Desligada");
284         digitalWrite(L_Garagem, LOW);
285         auxBt_Garagem ++;
286       }else if (App_Comodos == 'C') {
287         //Serial.println("Cozinha Ligada");
288         digitalWrite(L_Cozinha, HIGH);
289         auxBt_Cozinha ++;
290       }else if (App_Comodos == 'c') {
291         //Serial.println("Cozinha Desligada");
292         digitalWrite(L_Cozinha, LOW);
293         auxBt_Cozinha ++;
294       }else if (App_Comodos == 'S') {
```

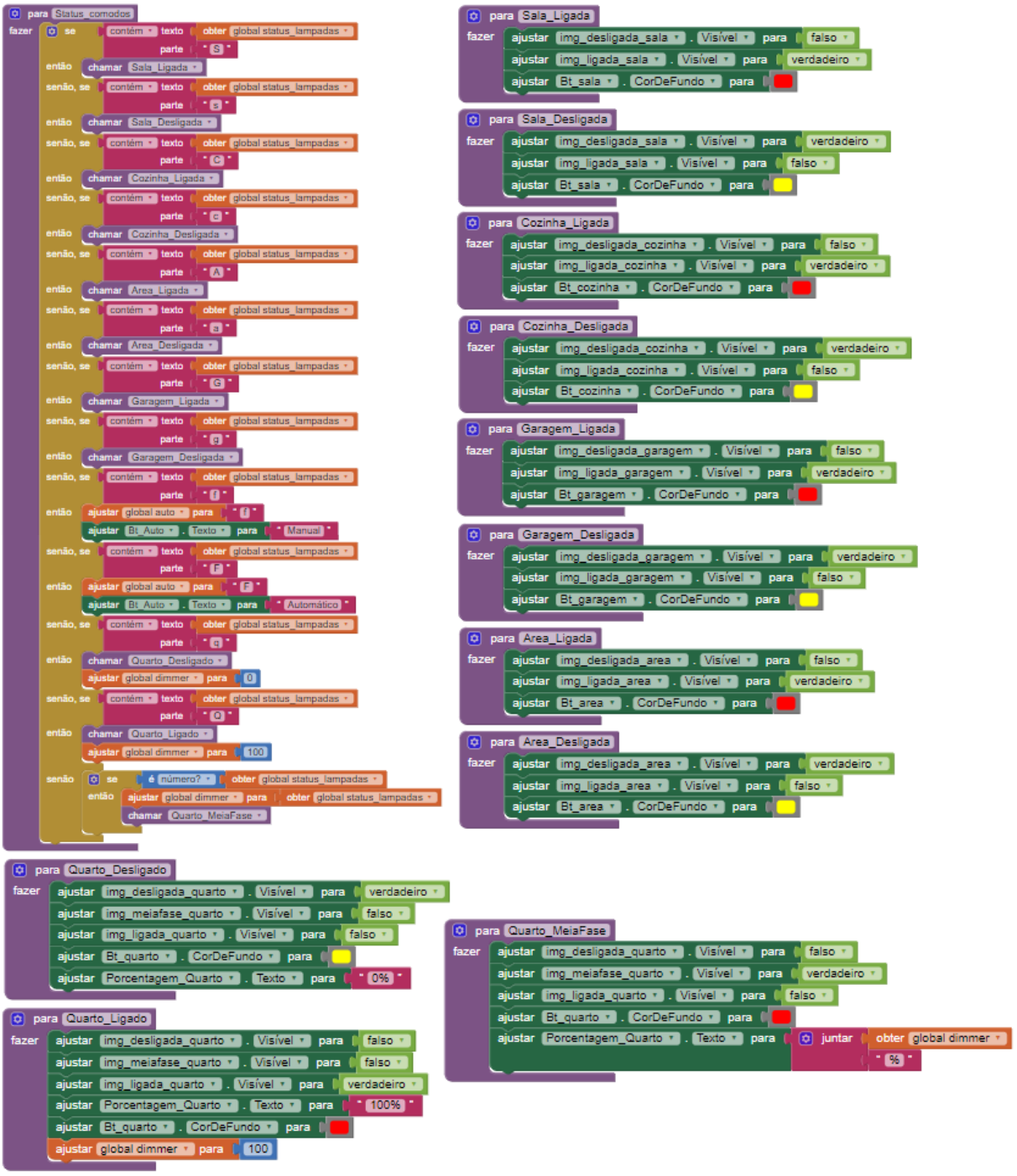
```
295       //Serial.println("Sala Ligada");
296       digitalWrite(L_Sala, HIGH);
297       auxBt_Sala ++;
298     }else if (App_Comodos == 's') {
299       //Serial.println("Sala Desligada");
300       digitalWrite(L_Sala, LOW);;
301       auxBt_Sala ++;
302     }
303     if (App_Comodos == 'Q'){
304       //Serial.println("Quarto Ligado");
305       if(auxBrilho_Quarto == 0){
306         ZC = 2;
307       }else{
308         ZC = 1;
309       }
310     }else if (App_Comodos == 'q'){
311       //Serial.println("Quarto Desligado");
312       ZC = 0;
313     }
314     if (App_Comodos == 'z'){
315       //Serial.println("Atualização dos Cômodos");
316       Att_Comodos();
317     }
318     if (App_Comodos == 'f'){
319       Fot_Auto = App_Comodos;
320       Status = "g";
321       bluetooth.print(Status);
322       delay(300);
323       Status = "a";
324       bluetooth.print(Status);
325       delay(300);
326     }else if (App_Comodos == 'F'){
327       Fot_Auto = App_Comodos;
```

```
328         if (Escuro()){
329           Status = "A";
330           bluetooth.print(Status);
331           delay(300);
332           if(PIR()){
333             Status = "G";
334             bluetooth.print(Status);
335             delay(300);
336           }
337         }
338         if (auxBt_Garagem % 2 != 0) {
339           auxBt_Garagem ++;
340           if(!Escuro()){
341             Status = "g";
342             bluetooth.print(Status);
343             delay(300);
344           }
345         }
346         if (auxBt_Area % 2 != 0) {
347           auxBt_Area ++;
348           if(!Escuro()){
349             Status = "a";
350             bluetooth.print(Status);
351             delay(300);
352           }
353         }
354       }
355       Serial.println(App_Comodos);
356     }
357   }
358
359   void Att_Comodos(){
360     if (digitalRead(L_Area) != LOW){
```

```
361       St_Comodos = "A";
362     }else{
363       St_Comodos = "a";
364     }
365     delay(400);
366     bluetooth.print(St_Comodos);
367
368     if (digitalRead(L_Garagem) != LOW){
369       St_Comodos = "G";
370     }else{
371       St_Comodos = "g";
372     }
373     delay(400);
374     Serial.println(St_Comodos);
375     bluetooth.print(St_Comodos);
376
377     if (digitalRead(L_Sala) != LOW){
378       St_Comodos = "S";
379     }else{
380       St_Comodos = "s";
381     }
382     delay(400);
383     bluetooth.print(St_Comodos);
384
385     if (digitalRead(L_Cozinha) != LOW){
386       St_Comodos = "C";
387     }else{
388       St_Comodos = "c";
389     }
390     delay(400);
391     bluetooth.print(St_Comodos);
392
393     if (auxBrilho_Quarto == 100){
```

```
394       St_Comodos = "Q";
395     }else if(auxBrilho_Quarto == 0){
396       St_Comodos = "q";
397     } else{
398       St_Comodos = aux_Quarto;
399     }
400     delay(400);
401     bluetooth.print(St_Comodos);
402 
403     delay(400);
404     bluetooth.print(Fot_Auto);
405   }
406 
407   void ZeroCross(){
408     if(ZC == 2){
409       digitalWrite(L_Quarto, HIGH);
410     } else if(ZC == 0){
411       digitalWrite(L_Quarto, LOW);
412     } else if (ZC == 1){
413       if(pare == 0){
414         long t1 = 82L * (100L - aux_Quarto);
415         delayMicroseconds(t1);
416         digitalWrite(L_Quarto, HIGH);
417         delayMicroseconds(6);
418         digitalWrite(L_Quarto, LOW);
419       }
420     }
421   }
```

APÊNDICE B – PROCEDIMENTO DE AÇÃO DO APLICATIVO CONFORME DADOS RECEBIDOS

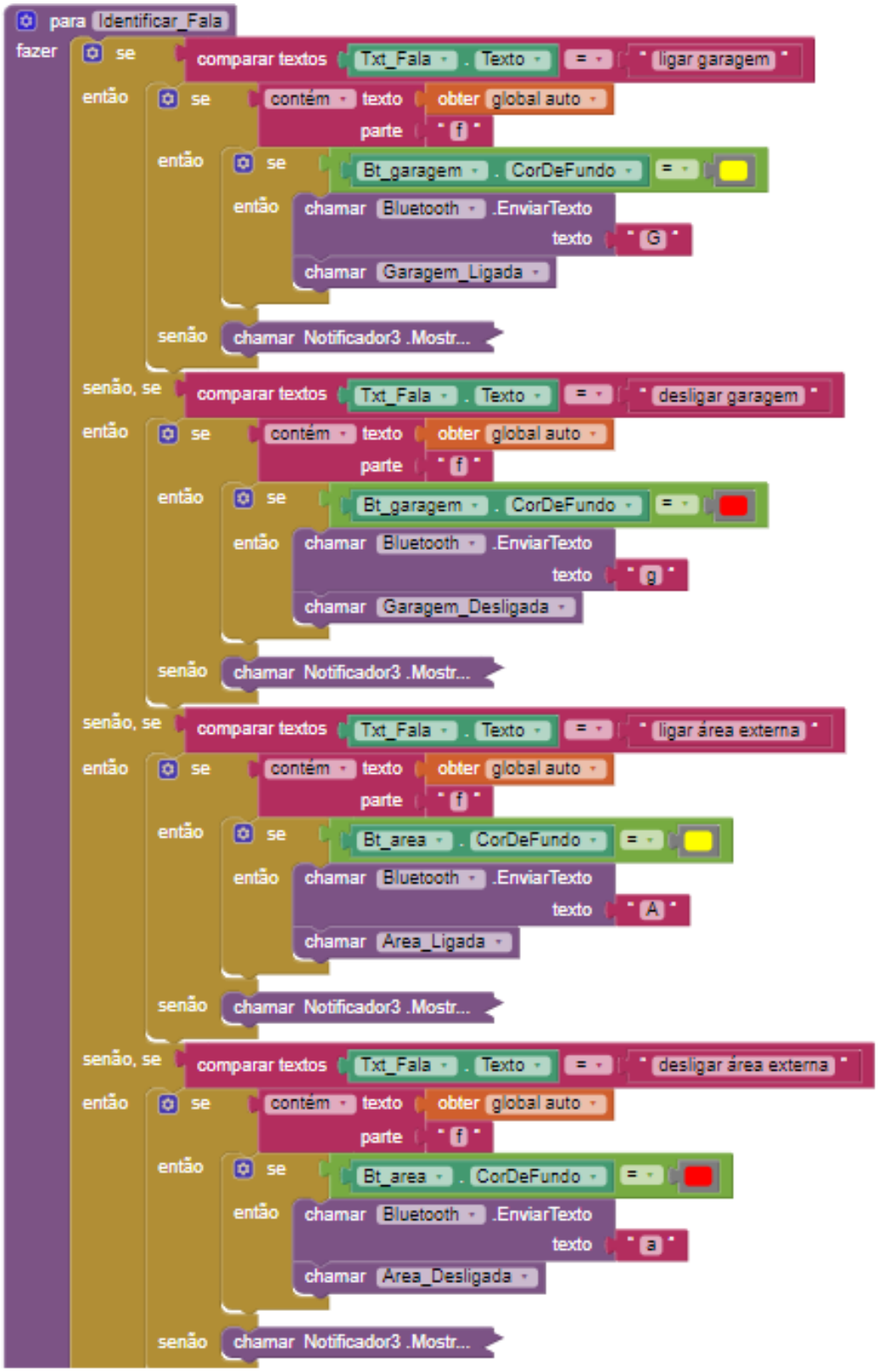


APÊNDICE C – ANÁLISE DOS DADOS RECEBIDOS PELO HC-05

```
// lê as interações realizadas no aplicativo (ligar e desligar cômodos)
void Status_Comodos_App() {
  if (bluetooth.available() > 0) {
    App_Comodos = bluetooth.read();  //realiza a leitura dos dados vindo do aplicativo
    if (App_Comodos == 'A') {
      //Serial.println("Área Ligada");
      digitalWrite(L_Area, HIGH);
      auxBt_Area ++;
    } else if (App_Comodos == 'a') {
      //Serial.println("Área Desligada");
      digitalWrite(L_Area, LOW);
      auxBt_Area ++;
    } else if (App_Comodos == 'G') {
      //Serial.println("Garagem Ligada");
      digitalWrite(L_Garagem, HIGH);
      auxBt_Garagem ++;
    } else if (App_Comodos == 'g') {
      //Serial.println("Garagem Desligada");
      digitalWrite(L_Garagem, LOW);
      auxBt_Garagem ++;
    }else if (App_Comodos == 'C') {
      //Serial.println("Cozinha Ligada");
      digitalWrite(L_Cozinha, HIGH);
      auxBt_Cozinha ++;
    }else if (App_Comodos == 'c') {
      //Serial.println("Cozinha Desligada");
      digitalWrite(L_Cozinha, LOW);
      auxBt_Cozinha ++;
    }else if (App_Comodos == 'S') {
      //Serial.println("Sala Ligada");
      digitalWrite(L_Sala, HIGH);
      auxBt_Sala ++;
    }else if (App_Comodos == 's') {
      //Serial.println("Sala Desligada");
      digitalWrite(L_Sala, LOW);;
      auxBt_Sala ++;
    }
    if (App_Comodos == 'Q'){
      //Serial.println("Quarto Ligado");
      if(auxBrilho_Quarto == 0){
        ZC = 2;
      }else{
        ZC = 1;
      }
    }else if (App_Comodos == 'q'){
      //Serial.println("Quarto Desligado");
      ZC = 0;
    }
```

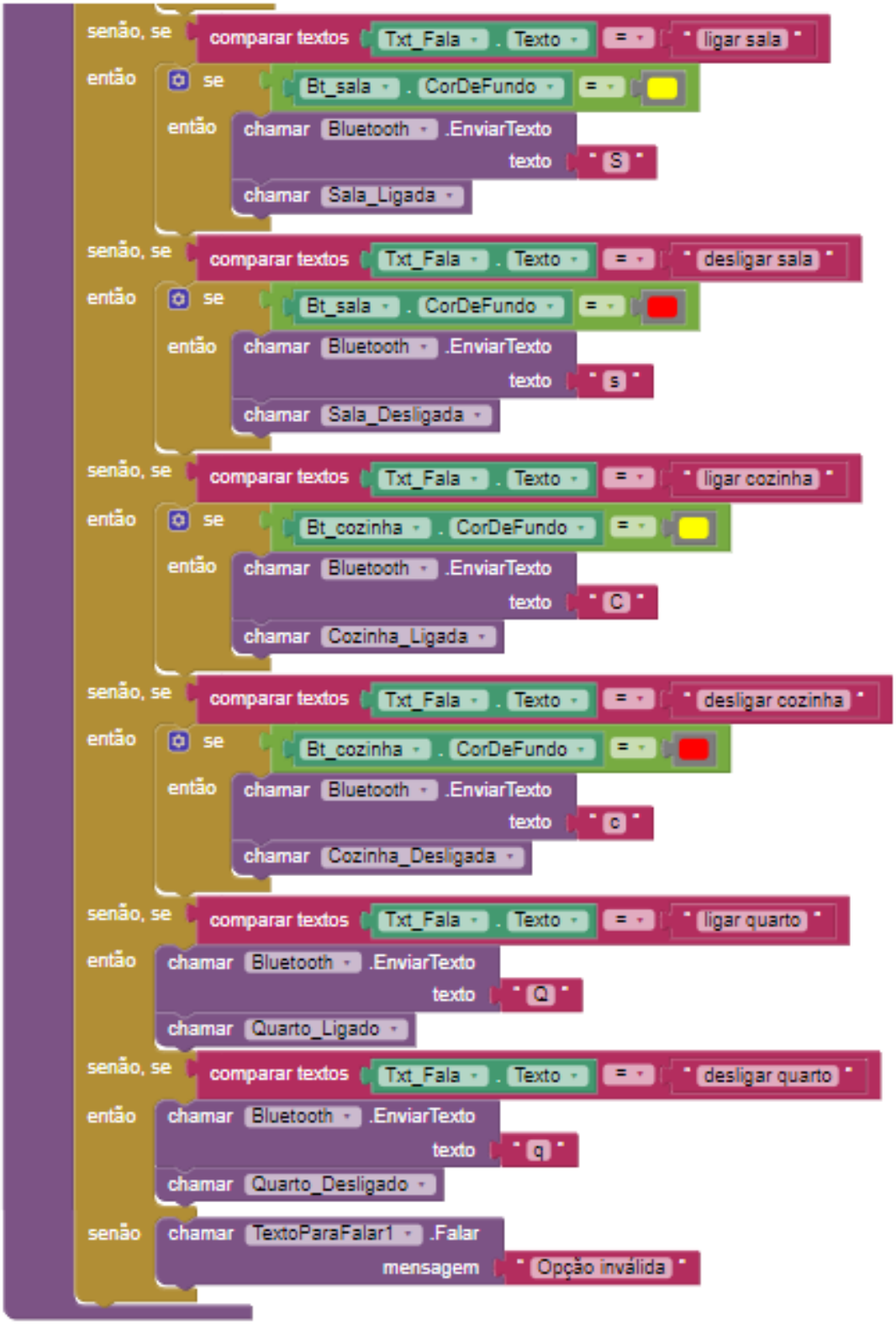
```
    if (App_Comodos == 'z'){
      //Serial.println("Atualização dos Cômodos");
      Att_Comodos();
    }
    if (App_Comodos == 'f'){
      Fot_Auto = App_Comodos;
      Status = "g";
      bluetooth.print(Status);
      delay(300);
      Status = "a";
      bluetooth.print(Status);
      delay(300);
    }else if (App_Comodos == 'F'){
      Fot_Auto = App_Comodos;
      if (Escuro()){
        Status = "A";
        bluetooth.print(Status);
        delay(300);
        if(PIR()){
          Status = "G";
          bluetooth.print(Status);
          delay(300);
        }
      }
      if (auxBt_Garagem % 2 != 0) {
        auxBt_Garagem ++;
        if(!Escuro()){
          Status = "g";
          bluetooth.print(Status);
          delay(300);
        }
      }
      if (auxBt_Area % 2 != 0) {
        auxBt_Area ++;
        if(!Escuro()){
          Status = "a";
          bluetooth.print(Status);
          delay(300);
        }
      }
    }
    Serial.println(App_Comodos);
  }
}
```

APÊNDICE D – AÇÕES DO MECANISMO DE RECONHECIMENTO POR VOZ



[...]

[...]


senão, se
comparar textos Txt_Fala . Texto = " ligar sala "
então
se Bt_sala . CorDeFundo =
então
chamar Bluetooth .EnviarTexto
texto " S "
chamar Sala_Ligada
senão, se
comparar textos Txt_Fala . Texto = " desligar sala "
então
se Bt_sala . CorDeFundo =
então
chamar Bluetooth .EnviarTexto
texto " s "
chamar Sala_Desligada
senão, se
comparar textos Txt_Fala . Texto = " ligar cozinha "
então
se Bt_cozinha . CorDeFundo =
então
chamar Bluetooth .EnviarTexto
texto " C "
chamar Cozinha_Ligada
senão, se
comparar textos Txt_Fala . Texto = " desligar cozinha "
então
se Bt_cozinha . CorDeFundo =
então
chamar Bluetooth .EnviarTexto
texto " c "
chamar Cozinha_Desligada
senão, se
comparar textos Txt_Fala . Texto = " ligar quarto "
então
chamar Bluetooth .EnviarTexto
texto " Q "
chamar Quarto_Ligado
senão, se
comparar textos Txt_Fala . Texto = " desligar quarto "
então
chamar Bluetooth .EnviarTexto
texto " q "
chamar Quarto_Desligado
senão
chamar TextoParaFalar1 .Falar
mensagem " Opção inválida "

APÊNDICE E – FUNÇÃO DE ATUALIZAÇÃO DO ESTADO DOS CÔMODOS.

```
void Att_Comodos(){
  if (digitalRead(L_Area) != LOW){
    St_Comodos = "A";
  }else{
    St_Comodos = "a";
  }
  delay(400);
  bluetooth.print(St_Comodos);

  if (digitalRead(L_Garagem) != LOW){
    St_Comodos = "G";
  }else{
    St_Comodos = "g";
  }
  delay(400);
  Serial.println(St_Comodos);
  bluetooth.print(St_Comodos);

  if (digitalRead(L_Sala) != LOW){
    St_Comodos = "S";
  }else{
    St_Comodos = "s";
  }
  delay(400);
  bluetooth.print(St_Comodos);

  if (digitalRead(L_Cozinha) != LOW){
    St_Comodos = "C";
  }else{
    St_Comodos = "c";
  }
  delay(400);
  bluetooth.print(St_Comodos);

  if (auxBrilho_Quarto == 100){
    St_Comodos = "Q";
  }else if(auxBrilho_Quarto == 0){
    St_Comodos = "q";
  } else{
    St_Comodos = auxBrilho_Quarto;
  }
  delay(400);
  bluetooth.print(St_Comodos);

  delay(400);
  bluetooth.print(Fot_Auto);
}
```