UNIVERSIDADE FEDERAL DE PERNAMBUCO

Universidade Federal de Pernambuco
Centro de Tecnologia e Geociências
Departamento de Eletrônica e Sistemas

Graduação em Engenharia Eletrônica

Thállen Kettyllen Carvalho Galvão

Sistema de monitoramento de iluminância em ambiente interno

Recife
Março de 2023

Thállen Kettyllen Carvalho Galvão

Sistema de monitoramento de iluminância em ambiente interno

Trabalho de Conclusão apresentado ao Curso de Graduação em Engenharia Eletrônica do Departamento de Eletrônica e Sistemas da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia Eletrônica.

Orientador: Prof. Guilherme Nunes Melo, D.Sc.

Recife

Março de 2023

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Galvão, Thállen Kettyllen Carvalho.
Sistema de monitoramento de iluminância em ambiente interno / Thállen Kettyllen Carvalho Galvão. - Recife, 2023.
48 : il., tab.

Orientador(a): Guilherme Nunes Melo
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, Engenharia Eletrônica - Bacharelado, 2023.

1. Sistemas embarcados. 2. Automatização . 3. Internet das Coisas (IoT). 4. Economia de energia. 5. Iluminância normativa. I. Melo, Guilherme Nunes. (Orientação). II. Título.

620 CDD (22.ed.)

Thállen Kettyllen Carvalho Galvão

Sistema de monitoramento de iluminância em ambiente interno

Trabalho de Conclusão apresentado ao Curso de Graduação em Engenharia Eletrônica do Departamento de Eletrônica e Sistemas da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia Eletrônica.

Banca Examinadora

Documento assinado digitalmente
gov.br
GUILHERME NUNES MELO
Data: 06/03/2023 16:40:13-0300
Verifique em https://verificador.iti.br

Prof. Guilherme Nunes Melo, D.Sc.
Universidade Federal de Pernambuco

Documento assinado digitalmente
gov.br
JOAO MARCELO XAVIER NATARIO TEIXEIRA
Data: 06/03/2023 16:32:20-0300
Verifique em https://verificador.iti.br

Prof. João Marcelo Xavier Natário Teixeira, D.Sc.
Universidade Federal de Pernambuco

Recife

Março de 2023

Agradecimentos

A Deus, pois sem ele nada disso seria possível.

À minha avó e minha mãe, que me criaram e me deram toda a oportunidade necessária que me fez alcançar tantos objetivos e com muita força, luta e resiliência, me tornaram a pessoa que sou hoje.

À minha família em geral, por ter me apoiado em todas as minhas decisões profissionais e ser a minha base e alicerce para tudo que construí e construirei.

Ao meu esposo, que esteve comigo em todos os momentos da graduação e me estimulou até o final me apoiando e me incentivando sempre.

Ao professor Guilherme, por ter sido meu professor orientador e ter tido a paciência de me auxiliar durante o processo de execução deste trabalho.

Aos meus colegas de curso, com quem pude aprender bastante durante os anos da faculdade e que foram importantes para o meu desenvolvimento e aprendizado.

Resumo do Trabalho de Conclusão de Curso apresentado ao DES/UFPE como parte dos requisitos necessários para a obtenção do grau de Bacharel em Engenharia Eletrônica (Eng.)

Sistema de monitoramento de iluminância em ambiente interno

Thállen Kettyllen Carvalho Galvão

A tecnologia vem auxiliando o homem em vários aspectos da vida. Com ela, foi possível a criação de veículos, computadores, sensores, como também desenvolvimento de sistemas para monitoramento de casas, edifícios, indústrias, e muito mais. Pode-se afirmar que o grande avanço da tecnologia se deu com a criação dos semicondutores, possibilitando, em seguida, o surgimento dos microcontroladores, pequenos dispositivos que, com a programação, transformam-se no que conhecemos como sistemas embarcados. Esses sistemas são tão complexos quanto um computador e são utilizados para os mais diversos fins possíveis. Uma das utilizações dos microcontroladores, e a mais atual, seria os seus usos para a Internet das Coisas(IoT). É possível utilizar esses sistemas juntamente com algum protocolo de comunicação para, por exemplo, monitorar toda uma casa, desde suas lâmpadas até portas. Seguindo essa ideia, este trabalho tem como intuito agregar o uso da tecnologia para facilitar o monitoramento da iluminância em ambientes internos, especificamente, ambientes de trabalho tendo em vista que isso é algo obrigatório por norma. A norma brasileira ABNT NBR ISO/CIE 8995-1 especifica os requisitos necessários para uma adequada iluminação em ambientes internos de trabalho a fim de assegurar ao trabalhador a eficiência no desempenho de atividades visuais com conforto e segurança. Sendo assim, o empregador deve assegurar tais requisitos para que o empregado esteja capacitado para realizar suas tarefas visuais em relação ao ambiente de trabalho no qual está inserido. A forma mais inteligente de realizar tal monitoramento é através da tecnologia.

Palavras-chave: *IoT*, Sistemas embarcados, iluminância, microcontrolador, sensor, norma.

Abstract of Course Conclusion Work presented to DES/UFPE as a partial fulfillment of the requirements for the degree of Bachelor of Electronic Engineering (Eng.)

Indoor lighting monitoring system

Thállen Kettyllen Carvalho Galvão

Technology has been helping man in various aspects of life. With it, it was possible to create vehicles, computers, sensors, as well as the development of systems for monitoring homes, buildings, industries, etc. It can be said that the great advance in technology took place with the creation of semiconductors, enabling the emergence of microcontrollers, small devices that, with programming, become what we know as embedded systems. These systems are as complex as a computer and are used for the most diverse purposes possible.One of the uses of microcontrollers, and the most current, would be their uses for the Internet of Things (IoT). It is possible to use these systems together with some communication protocol to, for example, monitor an entire house, from its lamps to its doors. Following this idea, this work aims to add the use of technology to facilitate the monitoring of luminescence in indoor environments, specifically, work environments, considering that this is mandatory by default. The Brazilian standard ABNT NBR ISO/CIE 8995-1 specifies the necessary requirements for adequate lighting in indoor work environments in order to ensure the efficiency of the worker in the performance of visual activities with comfort and safety. Therefore, the employer must ensure such requirements so that the employee is able to perform his visual tasks in relation to the work environment in which he is inserted. The smartest way to carry out such monitoring is through technology.

Keywords: IoT, Embedded systems, illuminance, microcontroller, sensor, standard.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

Capítulo 1

Introdução

1.1 Contextualização e Motivação

A revolução industrial, iniciada no século XVIII, foi essencial para a evolução tecnológica. Até a sua primeira fase, conforme ilustrado na Figura 1.1 [1], a necessidade do homem em trabalhos manuais e a dependência do vento ou da água para as indústrias funcionarem eram enormes. O invento da máquina a vapor foi o ponto chave para o início da revolução industrial e o crescimento da indústria, em distintos ramos, ao redor do mundo, nas grandes cidades. A segunda revolução industrial se deu com a descoberta do petróleo e da eletricidade, mas foi na terceira que a tecnologia teve um salto gigantesco [2].

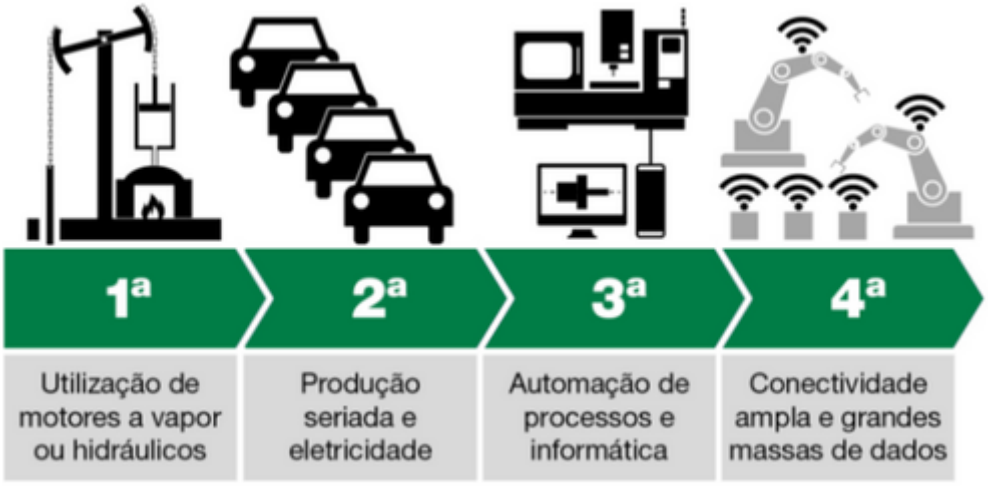

Figura 1.1: As 4 fases da revolução industrial. Fonte: Industria 4.0

Conhecida como a era da Automação industrial, a terceira revolução é marcada com o surgimento de semicondutores, microcontroladores, computadores, onde o primeiro foi criado por Alan Turin [3], conforme a Figura 1.2, e a internet [4], que possibilitaram uma automação de processos, fazendo com que a dependência humana fosse ainda menor, nas indústrias principalmente. Máquinas automatizadas passaram a realizar processos repetitivos em todos os setores industriais [2].

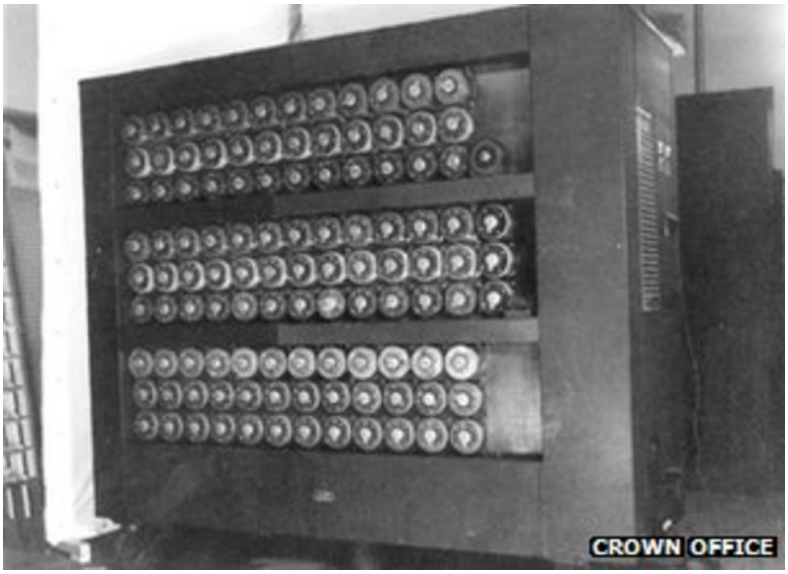

Figura 1.2: Primeiro computador criado por Alan Turin. Fonte: Télios

Hoje estamos presenciando o que chamamos de quarta revolução industrial ou indústria 4.0 onde não só temos a automação industrial, como também temos a adesão de sistemas ciber-físicos, inteligência artificial, realidade aumentada, *big data*, Internet das Coisas (IoT) [5], dentre outras tecnologias que podem ser observadas na Figura 1.3. A internet das coisas é um dos principais usos das tecnologias da indústria 4.0. Está ligada à conectividade de objetos do dia a dia à internet, a partir de sistemas que possibilitam o monitoramento, análise e atuação, comunicação, entre outros sistemas e geração de dados em tempo real para os mais diversos fins [6].

O fato é que a indústria 4.0 e suas tecnologias não estão presentes apenas nas indústrias, mas nos mais diversos ramos da tecnologia trazendo novas formas de solucionar problemas do dia a dia do ser humano, seja em ambientes de trabalho ou em casa, como, por exemplo, um sistema de segurança que se conecte em tempo

real com o usuário e muito mais [3].

No ambiente de trabalho, muito pode ser feito para otimizar e precisar processos nos mais diversos aspectos. Além disso, também é possível utilizar a tecnologia ao nosso favor, para garantir um ambiente de trabalho adequado para o colaborador, seja no monitoramento dos ruídos sonoros, na segurança ou na ergonomia, tendo em vista que esses são fatores importantes previstos por diversas normas brasileiras, como por exemplo a norma ABNT NBR ISO/CIE 8995-1 [7].

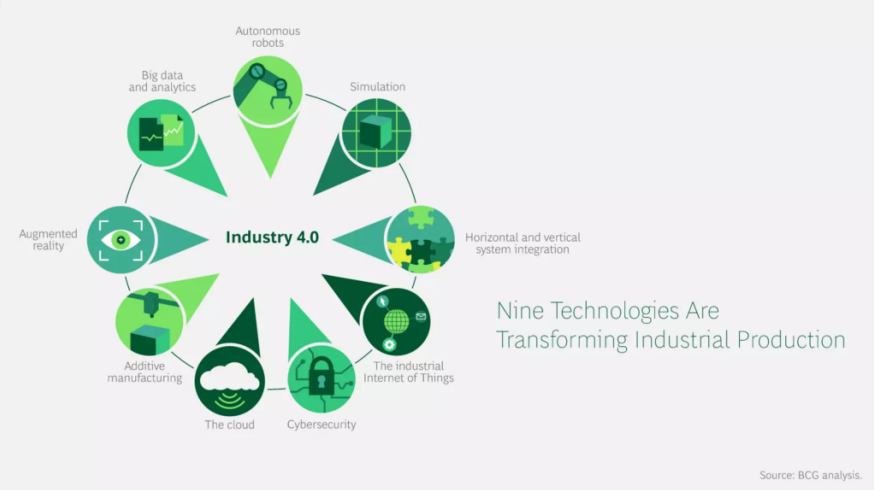

Figura 1.3: 9 tecnologias da indústria 4.0. Fonte: Télios

Esta norma define os requisitos de iluminação necessários para um ambiente interno de trabalho adequado para o colaborador, proporcionando um conforto, desempenho e segurança visuais, evitando futuros danos à visão e assegurando as melhores condições para realização das atividades. Para isto, é importante dar a devida atenção aos parâmetros estabelecidos em norma, como: distribuição da luminância, iluminância, ofuscamento, luz natural, manutenção, dentre outros que serão mais abordados no próximo tópico [7].

Para que o ambiente de trabalho cumpra com todos os requisitos normativos, ele deve ser projetado de maneira que a iluminância esteja dentro dos limites estabelecidos por norma em todos os momentos do dia, seja durante o dia ou a noite [7]. Muito embora os projetos arquitetônicos prevejam essas questões, é importante que a empresa se certifique, frequentemente, que está cumprido os requisitos cem por cento do tempo.

1.2 Objetivos

Visando o uso da tecnologia para se certificar que o ambiente de trabalho esteja respeitando os requisitos normativos em relação aos padrões de iluminância de forma mais otimizada, este projeto teve por objetivo implementar um sistema que monitore a intensidade da iluminância dos ambientes internos e tenha autonomia para desligar/ligar lâmpadas, caso necessário, para que os limites de iluminância sejam ajustados adequadamente no decorrer do dia, considerando que a luz natural do dia afeta diretamente nessa questão e é um tema abordado na norma citada, onde é orientado que o ambiente seja controlável quanto a iluminância, garantido a integração apropriada entre luz artificial e natural. Utilizando esse sistema, será possível não só monitorar e adequar a intensidade da luz no ambiente, como também estabelecer uma rotina apagando e ligando as luzes nos momentos corretos, cooperando até mesmo na economia de energia da empresa.

A norma ABNT NBR ISO/CIE 8995-1 estabelece parâmetros de iluminância para os mais diversos tipos de ambientes empresariais. Sendo assim, a nível de experimento, foi selecionado o ambiente de número 22, escritórios. Dentro dessa categoria é estabelecido o limite inferior de iluminância mantida (E_m, lux), o índice limite máximo de ofuscamento unificado (UGRL) para cada área especificada e o índice de reprodução de cor (Ra) que é uma característica das lâmpadas utilizadas no projeto. Na Tabela 1.1 podemos verificar os índices de cada área em escritórios [7].

Tabela 1.1: Índices limitantes para Escritórios

Ambientes específicos em Escritórios	$E_m(lux)$	UGRL	Ra
Arquivamento, cópia, circulação etc.	300	19	80
Escrever, teclar, ler, processar dados	500	19	80
Desenho técnico	750	16	80
Estações de projeto assistido por computador	500	19	80
Salas de reunião e conferência	500	19	80
Recepção	300	22	80
Arquivos	200	25	80

Como os dados de UGRL e Ra são dados inerentes à lâmpada sendo pré esta-

belecidos no projeto arquitetônico e fornecidos pelos fabricantes das luminárias, o dado passível de variância e estudado neste trabalho será a iluminância mantida, segunda coluna da Tabela 1.1.

1.3 Organização do Texto

O trabalho está segmentado em 6 capítulos com referências bibliográficas no final.

Capítulo 2 - Fundamentação Teórica. Este capítulo apresenta a fundamentação teórica necessária para a realização do projeto.

Capítulo 3 - Metodologia. Neste capítulo está a descrição detalhada da metodologia utilizada no projeto como equipamentos, linguagem e ferramentas.

Capítulo 4 - Desenvolvimento. É onde está o corpo do projeto, toda a implementação realizada.

Capítulo 5 - Resultados. Este capítulo mostra os resultados obtidos a partir do sistema desenvolvido e sua funcionalidade.

Capítulo 6 - Conclusão. Considerações finais e propostas de melhorias para o tema tratado.

Capítulo 2

Fundamentação Teórica

Neste capítulo serão abordadas as definições de alguns conceitos importantes para este trabalho, desde os mais abrangentes e básicos aos mais específicos e complexos, a fim de que leitores de diversas áreas do conhecimento absorvam os principais conceitos necessários para o entendimento deste trabalho. Será adotada uma abordagem onde os conceitos mais abrangentes e inerentes a diversos temas serão explanados inicialmente e em seguida serão definidos conceitos específicos para este trabalho.

Iluminância e Sistemas Embarcados serão os temas iniciais abordados, que são temas abrangentes, mas importantes para entender a proposta e motivação do trabalho. Seguindo, será definido o conceito de microcontroladores e especificado o escolhido para o projeto e os motivos para tal. Além disso, serão definidos os sensores e atuadores utilizados no projeto.

2.1 Iluminância

A iluminância é definida como a relação do fluxo luminoso que incide de forma perpendicular em uma superfície plana de acordo com a equação 2.1. [8]

$$E = \frac{\phi}{A} \tag{2.1}$$

Onde ϕ é o fluxo luminoso medido em lúmen (lm) e A é a área da superfície. Toda construção deve ser projetada respeitando os limites mínimos necessários de iluminância, de acordo com a norma ABNT NBR ISO/CIE 8995-1 [7].

Em outras palavras, iluminância é a quantidade de luz que incide em uma superfície, como em uma mesa de trabalho [8], conforme ilustrado na Figura 2.1. Essa quantidade de luz é medida em lux (lx), unidade de medida do SI (Sistema Internacional). Luminância, por outro lado, é a luz refletida pela superfície/objeto, que permite a sua visualização.

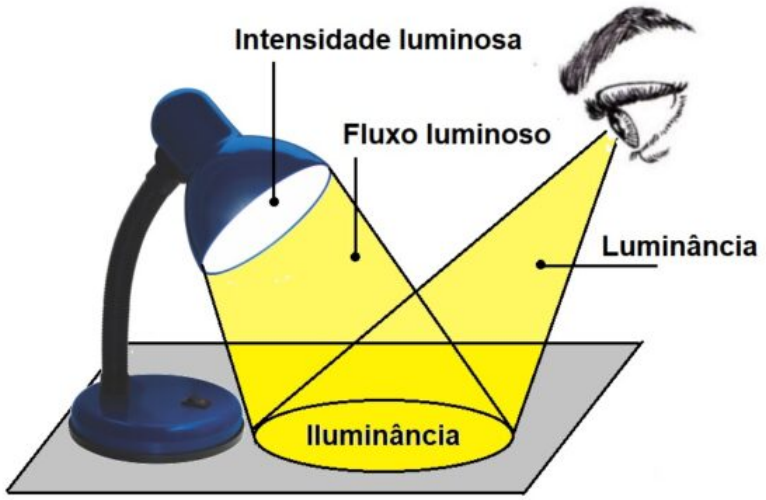

Figura 2.1: Ilustração sobre iluminância. Fonte: Guia da engenharia

É possível determinar a iluminância de um ambiente utilizando um equipamento chamado de luxímetro [9], ilustrado na Figura 2.2 [10]. Existem também alguns sensores e resistores, como os chamados *Light Dependent Resistor* (LDR), utilizados na eletrônica e em projetos eletrônicos que fornecem, através das suas implementações em Sistemas Embarcados, do inglês *Embedded System* (ES), direta ou indiretamente, a depender do sensor escolhido, a medida da iluminância em um ambiente [11].

Figura 2.2: Luxímetro. Fonte: Instru sul blog

2.2 Sistemas Embarcados

Um ES é um sistema micro-controlado combinando *hardware*, como um micro-controlador e seus periféricos, e *software*, que seria a lógica utilizada para que o sistema funcione, sendo projetado para uma função específica. Diferentemente de computadores pessoais que são de uso geral, por exemplo, o ES tem suas tarefas predefinidas pelo *software*. O sistema pode ser programável ou ter específicas funcionalidades, não podendo ser alteradas durante o uso [12].

Tais sistemas são utilizados em diversas áreas hoje, principalmente após a utilização da IoT e da quarta revolução industrial, tópico abordado no capítulo 1. O ES está presente nas indústrias, automóveis, câmeras, equipamentos médicos, aviões, celulares e muito mais.

2.3 Microcontroladores

Microcontroladores são pequenos dispositivos eletrônicos compostos por um processador dedicado a executar uma aplicação específica [13]. Os microcontroladores são de inteligência programável possuindo uma unidade lógica aritmética onde as operações matemáticas e lógicas são executadas. [14] Sua principal função é executar a aplicação para a qual ele foi programado interagindo com o meio externo.

Hoje os microcontroladores são muito requisitados por serem parte essencial dos sistemas eletrônicos embarcados, pois eles integram placas de desenvolvimentos

utilizadas em tais sistemas e funcionam com o dispositivo onde será implementada a aplicação [13].

2.3.1 Arduino

O arduino, ilustrado na Figura 2.3 [15], é uma plataforma de prototipagem *open-source*, composta por um microcontrolador de programação específico e pinos de entrada e saída digitais e analógicas, além de pinos próprios para alimentação e comunicação diferenciada com protocolo *Inter-Integrated Circuit* (I2C) [16]. O arduino possibilita o desenvolvimento dos mais diversos projetos eletrônicos com um custo relativamente baixo.

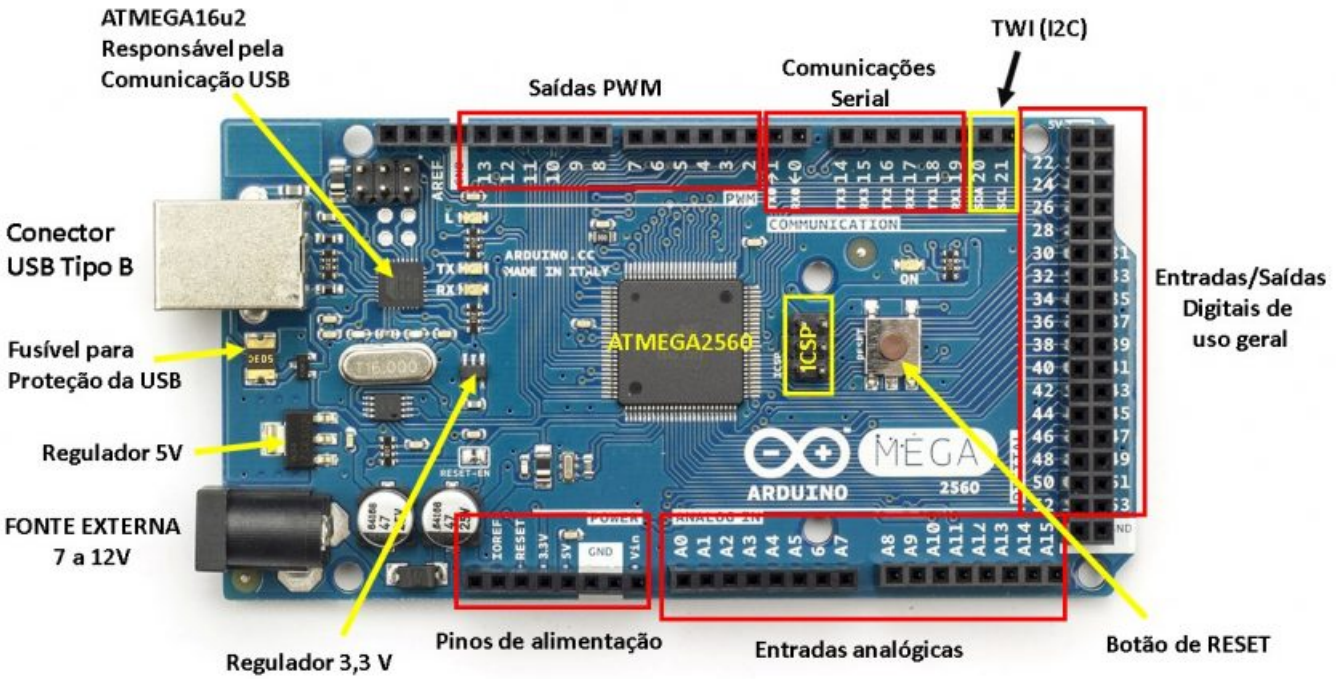

Figura 2.3: Arduino Mega. Fonte: Embarcados

O funcionamento do arduino se dá por códigos feitos, por exemplo, através da Arduino *Integrated Development Environment* (IDE) que é um programa criado para implementação de códigos em linguagem C para arduino, que pode ser baixado gratuitamente. A conexão do arduino com o computador é feita através de cabo USB, permitindo que os comandos definidos no programa sejam devidamente transferidos

até a placa [16].

Os sensores e atuadores podem ser conectados na plataforma do arduino através de *jumpers* ou em placas de ensaio, as chamadas *protoboards*.

2.3.2 ESP32

A esp32, ilustrada na Figura 3.1 [17], é uma placa de desenvolvimento assim como o arduino, possuindo todos os recursos necessários para o desenvolvimento de projetos. Ela é um pouco mais abrangente que o arduino, por possuir algumas características a mais, principalmente, em relação ao módulo *Wi-Fi* embutido, o que auxilia bastante em desenvolvimento de projetos IoT [18].

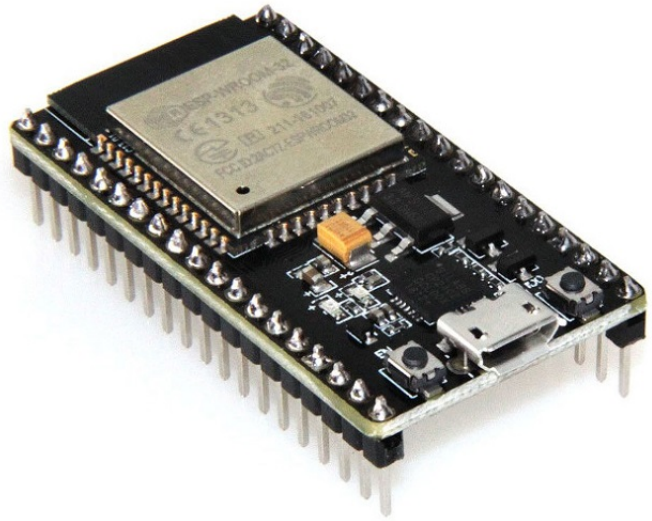

Figura 2.4: Esp32. Fonte: Blog Master Walker

Além disso, ele também possui Bluetooth híbrido, que seria o clássico e o *Bluetooth Low Energy* (BLE), múltiplos sensores embutidos como por exemplo o sensor de temperatura, dentre outros. Na Figura 2.5 podem ser observadas todas as funcionalidades, inputs, outputs e módulos da esp32 [19].

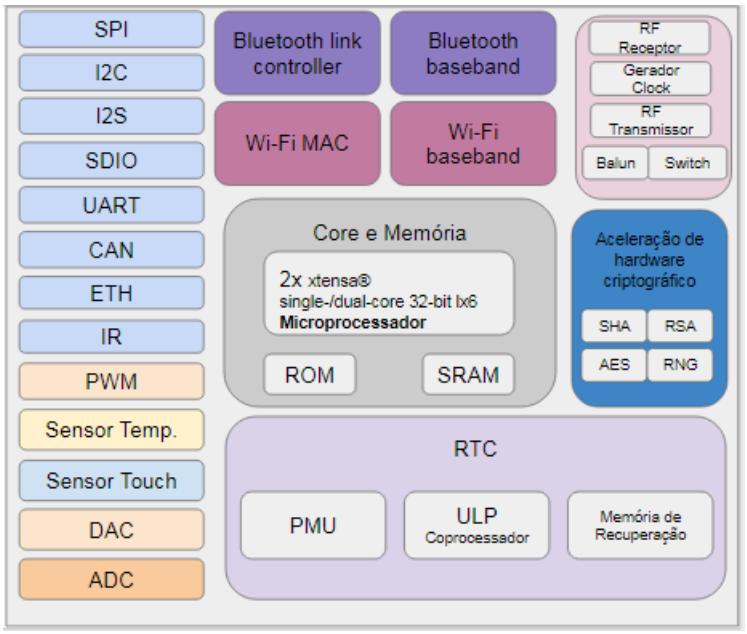

Figura 2.5: Diagrama da Esp32. Fonte: Blog Master Walker

2.4 Periféricos

Para a implementação do projeto foram utilizados alguns componentes periféricos, para que o propósito deste trabalho fosse alcançado. Os periféricos utilizados foram: Sensor TEMT6000, módulo relé e módulo RTC.

2.4.1 Sensor TEMT6000

O sensor TEMT6000, ilustrado na Figura 2.6, é um sensor de luminosidade que detecta a intensidade de luz incidida no ambiente, através da medida da iluminância, mensurada em lux. Ela basicamente, analisa o quanto de luz está sendo incidida sobre o metro quadrado daquela superfície. Para isso o sensor deve estar inserido sobre a superfície onde é desejado saber a iluminância [20].

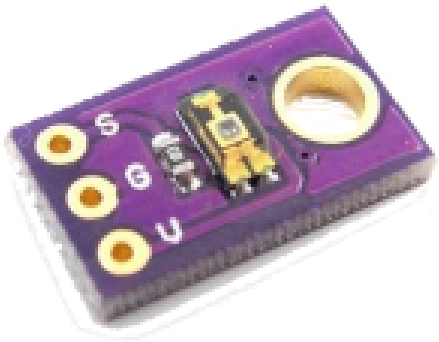

Figura 2.6: Sensor TEMT6000. Fonte: Auto Core Robotica

Resumindo sua funcionalidade, acontece da seguinte forma: quanto mais iluminância detectada, maior a corrente e vice-versa seguindo uma relação de proporcionalidade entre lux e corrente. Essa relação pode ser facilmente observada no *datasheet* do sensor, onde é ilustrado no gráfico da Figura 2.7 [21].

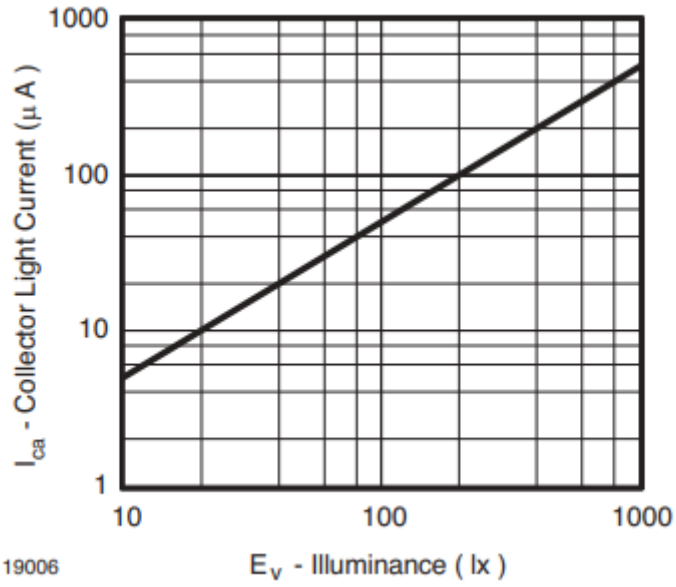

Figura 2.7: Relação proporcional entre iluminância em lux e corrente em Ampere. Fonte: Datasheet

2.4.2 Módulo RTC

O módulo RTC ou Relógio de Tempo Real, ilustrado na Figura 2.8 [22], é um circuito capaz de guardar a data e hora mantendo-as atualizadas em tempo real. Ou seja, quando acionado, o módulo retorna o valor da data e hora atual [23]. Para que isso aconteça, o RTC precisa estar sempre sendo alimentado por uma fonte de energia, garantindo assim o armazenamento das informações de data e hora, independente da alimentação do microcontrolador, por isso, é utilizada uma bateria externa no módulo. Por exemplo, nos computadores, a hora é normalmente medida na placa mãe, onde há uma bateria para garantir o funcionamento no caso da energia acabar.

Além da bateria, o RTC possui um cristal utilizado na geração do clock interno para contagem dos segundos. É muito comum do RTC utilizar um cristal de 32768 Hz. Internamente ele acaba dividindo o *clock* até obter uma frequência próxima de 1 Hz para contar os segundos corretamente [23].

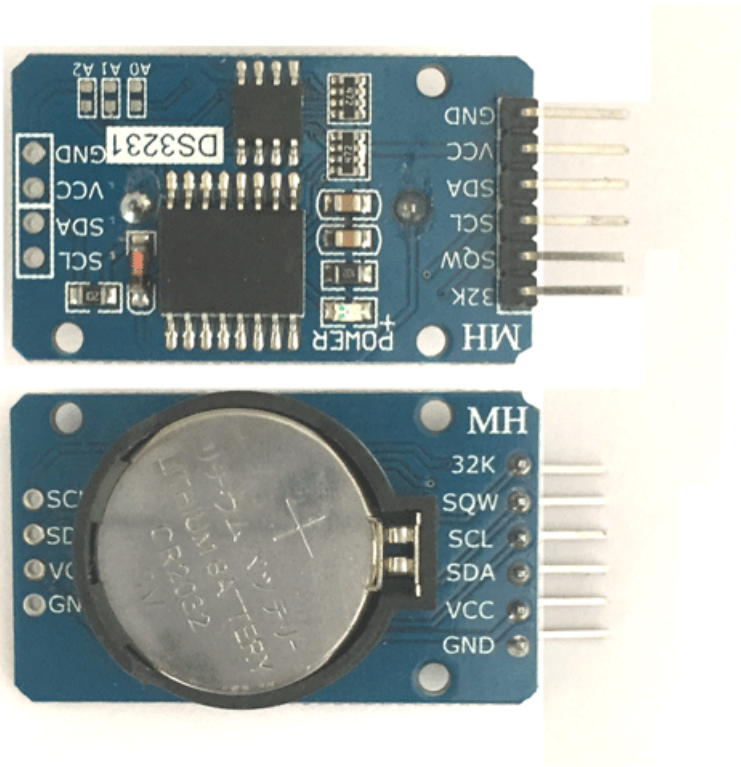

Figura 2.8: Módulo RTC DS3231. Fonte: Components101

Existem vários tipos e complexidades de módulos RTC, alguns com funções a

mais do que apenas armazenar a data e hora, como acionamento de alarme, medição de temperatura, dentre outras [23].

Capítulo 3

Metodologia

Este capítulo define as etapas realizadas para alcançar o objetivo deste trabalho, detalhando as ferramentas e requisitos para o desenvolvimento do projeto. 5 etapas foram realizadas, são elas:

- Análise de requisitos do sistema;
- Definição do escopo do projeto;
- Escolha das ferramentas utilizadas;
- Implementação do projeto;
- Análise de resultados.

3.1 Análise de requisitos do sistema

Ao definir o tema do projeto, foi preciso analisar tudo que era necessário para a sua implementação. Portanto, nesta etapa foram realizados as pesquisas e estudos, tanto da norma em si quanto do resultado esperado, que deram o pontapé inicial para que a solução do problema fosse resolvida utilizando a tecnologia e a eletrônica como recursos de acordo com as necessidades do projeto.

Tendo em vista que podem haver várias soluções para o mesmo problema, de maior ou menor complexidade, foi preciso analisar o escopo do projeto para que este

fosse simples porém completo e estivesse de acordo com a motivação inicial.

É importante ainda mencionar que todo projeto pode ter melhorias no futuro podendo ser otimizado e trazer novas funcionalidades.

3.2 Definição do escopo do projeto

Com o tema e análise de requisitos bem definidos, foi necessário definir o escopo do projeto, ou seja, suas limitações. Nesta etapa foi escolhido quais seriam as principais funcionalidades do sistema, de modo que ele cumprisse aquilo para o qual ele foi planejado, sem se tornar tão complexo quanto o necessário.

A partir disso, seguiu-se para a etapa de escolha ferramentas.

3.3 Escolha de ferramentas

Com as funcionalidades bem definidas, surgiu a necessidade de escolher quais ferramentas seriam necessárias para atingir tais objetivos do projeto. A linguagem de programação foi uma das escolhas realizadas, assim como as bibliotecas indispensáveis para o funcionamento do sistema.

Nas Seções 3.3.1 a 3.3.4 são explanadas algumas dessas ferramentas escolhidas para o projeto.

3.3.1 Linguagem C++

C++ é uma linguagem de programação baseada na linguagem C. Considerada a quarta linguagem mais utilizada no mundo, a linguagem C++ foi criada em 1983, por Bjarne Stroustrup. Ela é considerada uma extensão da linguagem C por possuir características semelhantes utilizando comandos e estruturas dessa linguagem, porém a diferença fundamental entre as duas linguagens é que o C++ permite a orientação a objeto, trazendo conceitos como herança múltipla, classes abstratas, métodos estáticos, métodos constantes e membros protegido [24].

Com o decorrer dos anos, foram surgindos novas versões e melhorias para C++. Considerando que é uma linguagem *open source*, ou seja, aberta à comunidade, ela vem ganhando melhorias até os dias atuais.

C++ é utilizada tanto como linguagem de máquina, quanto para *softwares* empresariais, acadêmicos, jogos e muito mais. Por isso, ela é considerada uma linguagem tanto de baixo, quanto de alto nível. Empresas como Google, Adobe, NASA, Microsoft, Mozilla, dentre outras, utilizam C++ atualmente [25].

Academicamente falando, ela é muito utilizada por ter compatibilidade com plataformas de prototipagens como o arduino sendo a linguagem padrão para o mesmo. Como o arduino e a Esp32 foram utilizados neste projeto, a linguagem C++ foi a escolhida para a implementação do projeto.

3.3.2 Linguagem Wiring

O wiring é um sistema de código aberto criado por Hernando Barragán. Essa plataforma serviu de base para criação da plataforma Arduino e ficou conhecida como a Linguagem Wiring. Nada mais é do que uma abstração de *hardware* para a linguagem C/C++ para programação de microcontroladores [26].

Portanto, é com essa linguagem que é feita a comunicação entre o *hardware*, microcontrolador utilizado, e o código implementado nele, feito utilizando a linguagem C++, escolhida para o projeto.

3.3.3 Visual Studio Code e PlatformIO

Como foi visto na seção 2.3.1, a ferramenta padrão para desenvolvimento em arduino é a Aduino IDE [16], porém foi utilizado uma ferramenta alternativa a ela: o Visual Studio Code, que é um editor de código *open source* bastante utilizado para desenvolvimendo de *software*, em conjunto com a extensão PlatformIO [27].

O Visual Studio Code é altamente customizável, permitindo que os usuários façam diversas alterações como mudar teclas de atalhos, adicionar extensões que facilitem no desenvolvimento, dentre outros [27]. Ele é mais vantajoso em relação à

IDE do arduino em vários aspectos, dentre os quais:

- Incorporação com Git, um sistema de versionamento que integra com GitHub e outras plataformas;
- Facilidade em debugar códigos;
- Diversas extensões.

Já o PlatformIO, uma das extensões do Visual Studio Code, é um ecossistema de código aberto para desenvolvimento em IoT, possibilitando o desenvolvimento de sistemas embarcados no Visual Studio Code, tendo suporte para placas como o aduino e Raspberry PI, permitindo a utilização de uma única ferramenta para diversos tipos de microcontroladores, diferentemente da IDE do arduino [27].

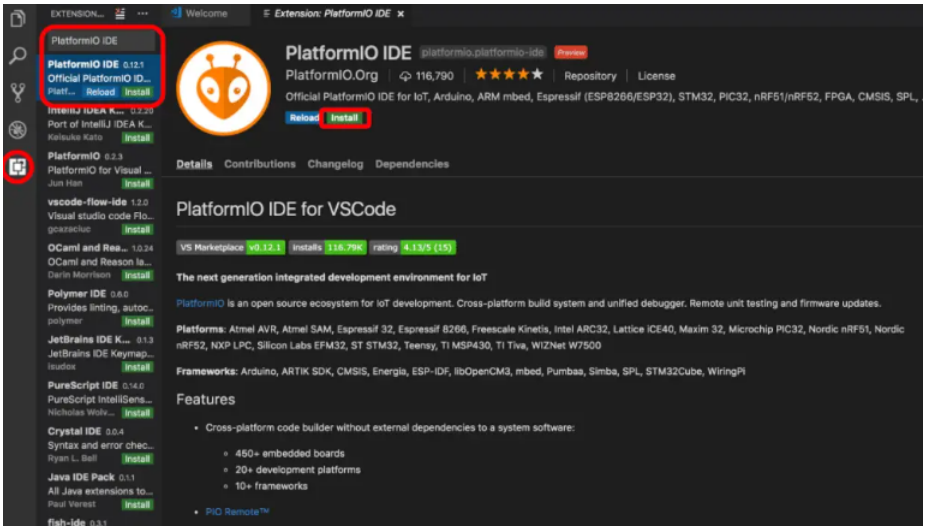

Figura 3.1: VSCode e plataformIO. Fonte: Embarcados

3.3.4 RTClib

A biblioteca RTClib é utilizada para facilitar a integração entre o arduino e o módulo RTC mostrado na seção 2.4.2. Ela utiliza um protocolo de comunicação chamado I2C e, por isso, precisa da biblioteca Wire para sua implementação [28].

O protocolo de comunicação I2C foi desenvolvido para conectar periféricos de baixa velocidade a microcontroladores para comunicação de curta distância. Ele

envolve o uso de 2 linhas para enviar e receber dados: um pino de relógio serial, o *Serial Clock pin* (SCL), que a placa controladora do Arduino pulsa em intervalos regulares e um pino de dados serial, o *Serial Data pin* (SDA), por onde os dados são enviados entre o microcontrolador e o RTC [29].

3.3.5 ESPAsyncWebServer

A biblioteca ESPAsyncWebServer é utilizada para criar um servidor web assíncrono em um microcontrolador esp. Sua documentação pode ser lida no seu Github [30]. Para usá-la em um esp32, é necessário instalar uma outra biblioteca chamada *AsyncTCP.h*, que é responsável por garantir um ambiente de rede de multiconexões. Além disso ela também precisa da biblioteca *WiFi.h* para que seja possível estabelecer uma conexão via internet.

Um servidor assíncrono se diferencia do síncrono pelo fato dele permitir múltiplas requisições simultâneas de diferentes clientes/usuários do sistema manipulando-as de forma escalável, fazendo com que as solicitações não bloqueiem uma às outras, quando feitas simultaneamente [31].

3.4 Implementação do projeto

Nesta etapa, detalhada no Capítulo 4, será realizado todo o desenvolvimento do projeto. Utilizando todas as ferramentas e tecnologias necessárias para desenvolver todas as funcionalidades estabelecidas no escopo do projeto. Para isso, será preciso fazer algumas considerações iniciais, utilizando a norma ABNT NBR ISO/CIE 8995-1 de apoio e motivação para tudo que será desenvolvido nesta etapa.

3.5 Análise de resultados

Nesta etapa, detalhada no Capítulo 5, será feita a verificação das funcionalidades do sistema desenvolvidas no Capítulo 4. Assim, serão realizados testes tanto para o

sistema no modo automático como no modo manual, afim de comprovar o resultado esperado para o projeto.

Capítulo 4

Desenvolvimento

Neste capítulo será descrita toda a implementação do projeto partindo da sua motivação inicial até as funcionalidades propostas. Para isso, foi necessário definir o escopo do projeto, levando em consideração a norma utilizada como base para o projeto, a fim de estabelecer as funcionalidades que serão desenvolvidas no final do trabalho. A partir disso, foi iniciada a montagem do *hardware* necessário para o projeto e o desenvolvimento do código para alcançar tais funcionalidades.

4.1 Definição do escopo

Partindo da motivação e objetivos do projeto, explanados nas seções 1.1 e 1.2, foi iniciado o escopo do sistema proposto. A norma ABNT NBR ISO/CIE 8995-1 define o limite mínimo de iluminância, em lux, para diversos tipos de ambientes internos de trabalho [7]. Um desses ambientes, escolhido como exemplo para este trabalho, é um ambiente de escritório, onde existem diferentes cômodos correlacionados e suas respectivas iluminâncias. Na Tabela 1.1, apresentada na seção 1.2, é possível observar cada ambiente específico e sua iluminância padrão em lux.

A medida da iluminância está ligada à área perpendicular da superfície estudada, de acordo com a equação 2.1, onde lumen, mensurada em lm, "é a unidade do Sistema Internacional para medir o fluxo luminoso" [32], e Area é a área da seção iluminada. Assim, a fim de simplificar o escopo, a nível de prova de conceito, respeitando a

iluminância mantida, definida por norma para o ambiente escolhido, e utilizando LED como fonte de luminância, foi escolhido um ambiente do tamanho de uma caixa de $100cm^2$ de área e a quantidade de 4 LEDs para simular um ambiente de trabalho. O ambiente escolhido, de acordo com a tabela 1.1 presente na norma, foi o referente a "Salas de reunião e conferência" onde é requerida uma iluminância mantida de **500 lux**, ou seja, o limite inferior para a iluminância neste ambiente. Considerando que o ambiente em questão tenha todos os padrões arquitetônicos necessários e previstos por norma, a proposta do sistema é monitorar a iluminância e adequá-la, de acordo com o padrão estabelecido, sempre que necessário permitindo uma possível economia de energia em um ambiente real.

4.2 Sistema Proposto

Tendo como base o escopo do projeto, a proposta é desenvolver um *software* embarcado, utilizando arduino e esp32, com comunicação serial entre eles, para ler um sensor de iluminância e controlar os LEDs utilizados, simulando as lâmpadas de um ambiente interno de trabalho. Para isso, será desenvolvido uma página *web* para controle do modo automático e manual e também para definição de horário inicial e final de rotina do sistema. Ou seja, a página *web* servirá como um meio de comunicação entre o usuário e o sistema.

O arduino irá ser utilizado para a leitura do sensor de iluminância TEMT6000 e controle dos LEDs e o módulo RTC. Através de comunicação serial, será passada as informações coletadas pelo servidor *web* implementado na esp32 para o arduino, que, por sua vez, será responsável pela manipulação e tratamento dos dados, a fim de que as funcionalidades propostas sejam atendidas. O sistema terá dois modos de funcionamento, o modo automático, explicado na subseção 4.2.1 e o modo manual, definido em 4.2.2.

4.2.1 Modo Automático

No modo automático, setado pelo usuário na página *web*, o dado coletado do sensor, que será a quantidade de iluminância mantida no ambiente, será utilizado como comparação com a iluminância mantida padrão, estabelecida de 500 lux para o ambiente definido, e, com base no valor mensurado pelo sensor, será realizada uma atuação no sistema, ligando ou desligando um ou mais LEDs, caso necessário. Desse modo, será possível garantir a iluminância mantida normativa e possibilitar uma economia na utilização dos LEDs no projeto, o que refletiria em uma economia de energia, no ambiente real.

Na página *web* também será definido o horário de início e fim da rotina do sistema, simulando o horário de início/fim do expediente trabalho, em um ambiente de trabalho interno. Essa informação coletada na página *web* será utilizada como base para ambos os modos. Ou seja, tanto no modo manual, quanto no modo automático, ao iniciar o sistema, no horário definido, os LEDs serão todos ligados e no horário final, os LEDs serão todos desligados.

4.2.2 Modo Manual

Caso o usuário deseje por pausar o sistema automático por alguma razão, ele pode setar o modo manual, onde as luzes ficarão sempre ligadas independentemente, do valor coletado pelo sensor e o usuário poderá desligar ou ligar, manualmente, via página *web*, cada LED caso assim desejar. Importante mencionar que o sistema foi projetado para suprir a necessidade de iluminância de acordo com a norma. Ou seja, todos os LEDs ligados, sem iluminação externa, estabelece o valor requerido por norma de iluminância, medida em lux, no total de 500 lux.

4.3 Projeto de *Hardware*

Nesta seção será descrito a implementação do projeto de *hardware* necessário para o trabalho proposto.

Para isso, foram utilizados alguns elementos descritos na seção 2, bem como, para deixar o sistema mais estável, foi utilizado uma placa de prototipação de fenolite já perfurada, para acoplar os LEDs, utilizados para simular as lâmpadas no ambiente.

O motivo principal da utilização da placa foi para reduzir instabilidades decorrentes de *jumpers* comuns. Isso reduziu em pouco mais da metade as utilizações de *jumpers* no projeto.

4.3.1 Diagrama do *Hardware*

Na Figura 4.1, pode-se observar um diagrama desenhado com o auxílio do site *wokwi* [33], do circuito desenvolvido para o projeto, onde foi utilizado um arduino Mega, uma esp32, um módulo RTC, 4LEDs de alto brilho e o sensor TEMT6000, que foi representado por um potenciômetro no diagrama, por possuir os mesmos pinos que o sensor utilizado.

Como foi desenvolvida uma página *web*, com um servidor *web* para comunicação entre usuário e sistema, foi escolhido a esp32 para esse fim, tendo em vista que ela possui módulo *Wi-Fi* embutido, facilitando o desenvolvimento. Já o arduino foi escolhido pois, após alguns teste com a esp32 e o arduino, percebeu-se que o sensor funcionou melhor utilizando a tensão de 5V como tensão de referência, diferentemente dos 3.3V da esp32. Além disso, optou-se por utilizar dois microcontroladores, para que, assim, fosse possível que um deles, neste caso o arduino, ficasse responsável pelo controle e manipulação dos dados e o outro, a esp32, pela interface do usuário e parametrização do sistema. Assim, as tarefas foram devidamente distribuídas entre eles.

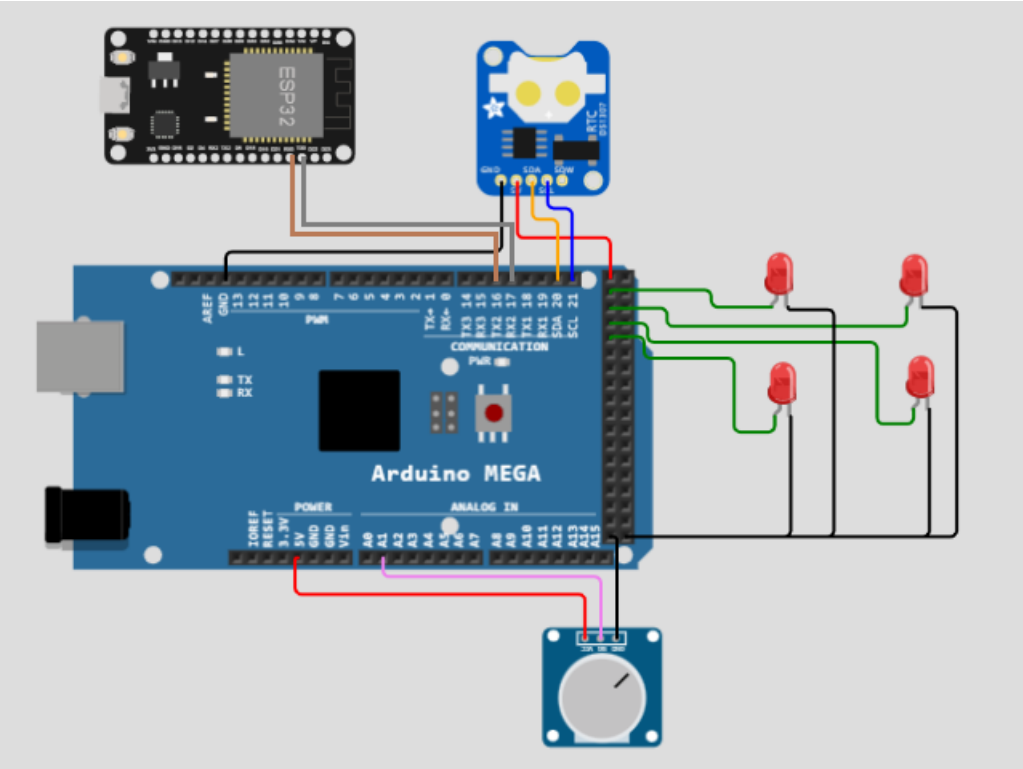

Figura 4.1: Diagrama de *hardware* do projeto. Fonte: A autora

4.3.2 Comunicação serial entre arduino e esp32

Como foram utilizados dois microcontroladores distintos, um para comportar a página *web* e o outro para os periféricos do projeto, assim como o tratamento e manipulação dos dados, foi necessário realizar uma comunicação entre ambos, a escolhida foi a comunicação serial *Universal Asynchronous Reception and Transmission* (UART).

Comunicação serial nada mais é do que uma transmissão de dados de forma sequencial, bit a bit, através de um canal de comunicação ou barramento [34].

O protocolo de comunicação UART permite que haja transmissão e recepção de dados entre dispositivos. A comunicação serial UART é assíncrona, ou seja, não apresentando *clock*, sinal de relógio. Assim, é preciso que os microcontroladores mantenham uma velocidade de execução de tarefas semelhante. Para isto, é utilizado o que chamamos de *baud rate*, que nada mais é do que a taxa de transmissão configurada para que haja uma comunicação serial assíncrona entre os microcontro-

ladores [35].

Os *bits* são transmitidos através das linhas RX e TX, recepção e transmissão, respectivamente. Na Figura 4.2, pode-se observar a configuração das conexões entre a esp32 e o arduino mega, onde a linha RX de um é conectada na TX do outro e vice-versa.

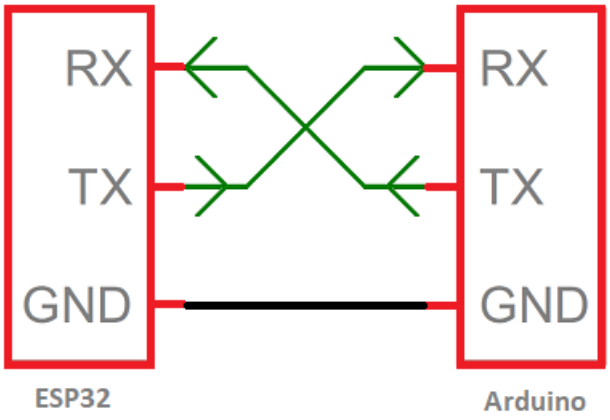

Figura 4.2: Comunicação serial entre arduino e esp32. Fonte: Fonte: ProgrammingBoss

Para definir início e fim da transmissão, é transmitido um *Start bit* antes dos *bits* de dados e ao final deles, é transmitido um *Stop bit*. Antes de iniciar a transmissão de dados, a tensão no fio que está no TX do arduino e RX da esp32 estará em 5V e assim que é iniciada a transmissão, a tensão vai para 0V e o *Start bit* é gerado [36]. É possível observar este comportamento na Figura 4.3.

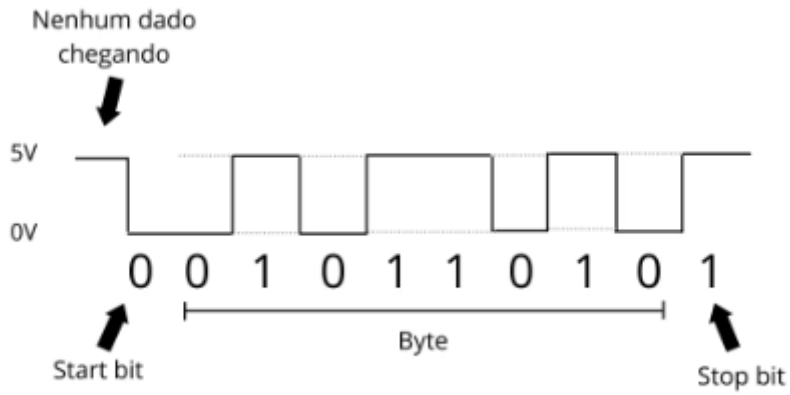

Figura 4.3: Tensão durante transmissão de dados. Fonte: Maker Hero

No diagrama da Figura 4.1 apresentada, as linhas marrom e cinza que conectam a esp32 ao arduino representam as linhas RX e TX da esp32, respectivamente.

4.3.3 Periféricos

O sensor TEMT6000, representado pelo potenciômetro no diagrama da Figura 4.1, possui três pinos distintos, S, G e V. O pino S é onde será enviado o sinal analógico referente à medição da iluminância mensurada pelo sensor. G é *Ground* (GND) e V o *Voltage Common Collector* (VCC), alimentado por 5V pelo arduino. O pino S foi conectado no pino A1 do arduino, que é um pino de entrada analógica. O sensor envia o dado analógico para o arduino através da porta A1 e, internamente, como o arduino trabalha com dados digitais, ele precisa realizar uma conversão A/D, analógico para digital, de acordo com a resolução do conversor do microcontrolador, que é dada pela equação resolução $= V_{ref}/2^n$, onde V_{ref} é a tensão de referência do microcontrolador, no caso do arduino, 5V, e n o número de *bits* do conversor, que seria 10 *bits* para o arduino MEGA [37].

O RTC utilizado foi o DS3231. Foram utilizados os pinos VCC, GND, SCL e SDA. O pino VCC foi alimentado pelo arduino com 5V e o GND que também foi conectado ao GND do arduino. Além disso, como o módulo RTC utiliza o protocolo de comunicação I2C, explicado na subseção 3.3.4, foi utilizado SDA, conectado ao pino 20 do arduino mega, referente ao pino SDA do arduino e o SCL, conectado ao pino 21 do arduino, como mostrado no diagrama da Figura 4.1.

O pino SDA é responsável pela transmissão de dados coletados pelo RTC para o arduino e o pino SCL é o pino de *clock*, necessário para que haja a comunicação via protocolo I2C. A motivação do uso do RTC é explicada na seção 4.4.

4.3.4 Montagem do circuito e protótipo final

O circuito desenhado no diagrama da Figura 4.1 foi implementado separadamente. Como dito anteriormente, foi utilizado uma placa perfurada de fenolite para o circuito referente aos LEDs, afim de reduzir instabilidade no sistema. Essa placa

foi acoplada acima de uma caixa de $100cm^2$, utilizada para simular o ambiente interno de trabalho. O sensor de iluminância, por sua vez, foi acoplado na parte inferior da caixa, de forma centralizada, para ficar exatamente sobre a superfície iluminada. Essa parte do protótipo pode ser vista na Figura 4.4

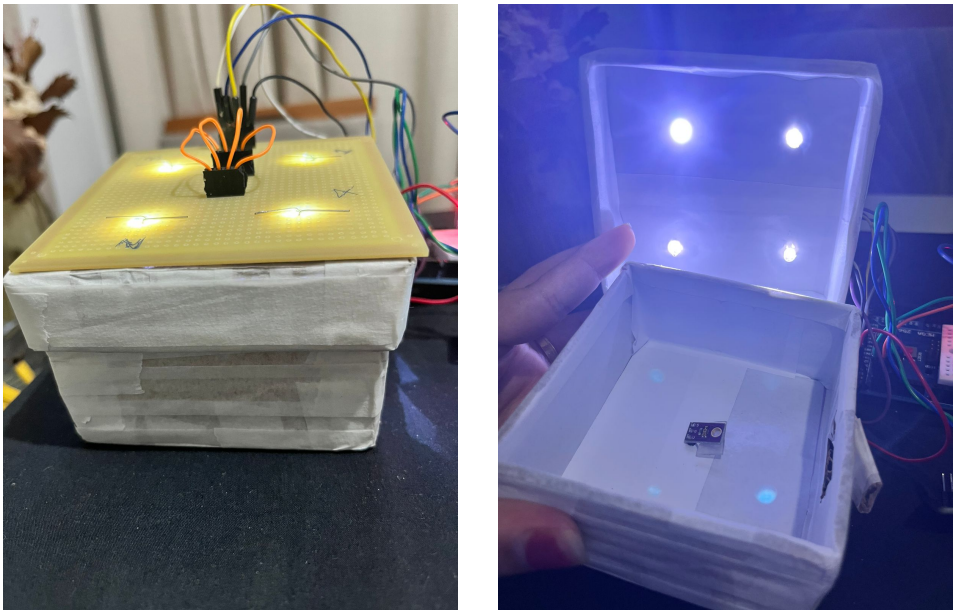

Figura 4.4: Circuito dos LEDs soldado na placa de fenolite e acoplamento do sensor no ambiente. Fonte: A autora

Para a outra parte do circuito, foi utilizada uma *protoboard* para acoplamento do RTC e todos os periféricos foram conectados ao arduino, fechando o circuito. Assim como a esp32, através dos pinos RX e TX de ambos. A Figura 4.5 mostra o protótipo final do projeto.

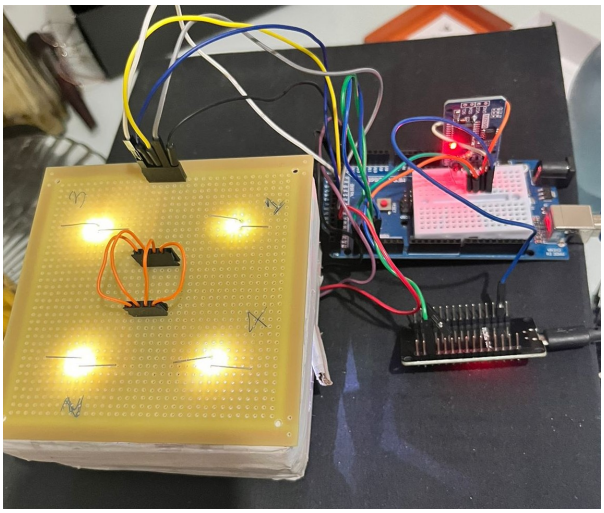

Figura 4.5: Protótipo final. Fonte: A autora

4.4 Projeto de *Software*

Esta seção contém as etapas de implementação do projeto de *software* necessárias para as funcionalidades do projeto.

Para isso, foi preciso dividir as etapas entre o código utilizado para criação do servidor *web*, implementado diretamente na esp32, e o código referente à manipulação e tratamento de dados vindos tanto da página *web*, quanto dos periféricos (sensor de iluminância e RTC), implementado no arduino.

4.4.1 Servidor *Web*

Para que haja uma interface entre o usuário e sistema, foi implementado um servidor *web* responsável por processar os dados inseridos pelo usuário através da interface desenvolvida em *HyperText Markup Language* (HTML). Foi utilizada a esp32 como microcontrolador para o servidor *web* tendo em vista que a esp32 possui módulo *Wi-Fi* imbutido, facilitando o desenvolvimento.

Inicialmente, foram incluídas as bibliotecas necessárias, descritas na subseção 3.3.5, *ESPAsyncWebServer* que, por sua vez, necessita da *AsyncTCP* e da *WiFi*. Após isso, foi instanciado um objeto da classe AsyncWebServer, pertencente a biblioteca ESPAsyncWebServer, onde foi definida a porta do servidor. Em seguida, foram realizadas as configuração da interface de rede estática com a biblioteca *WiFi*,

com isso, foi definido um *Internet Protocol* (IP) padrão, 192.168.0.117 com o método *Wifi.config()*, para que não fosse atribuído um IP aleatório pelo roteador. Assim, foi conectado ao ponto de acesso, com as devidas credenciais de rede, nome da rede e senha, utilizado o método *Wifi.begin()*.

Após isso, é feito o tratamento das requisições do tipo *GET*, atualizando as variáveis do sistema e retornado a resposta para o cliente. Esse processo é feito utilizando métodos da biblioteca *ESPAsyncWebServer*.

Para que as informações sejam coletadas pelo arduino através da comunicação serial UART, foi criado uma variável para ser usada como um *buffer*, para ser enviado pela UART com os dados que são obtidos através da interface usuário/sistema, necessários para a sua rotina, são eles: hora inicial e final da rotina, o status do modo automático e o status de todos os 4 LEDs. Na Figura 4.6, é possível observar o servidor *web* com a interface na configuração de modo manual.

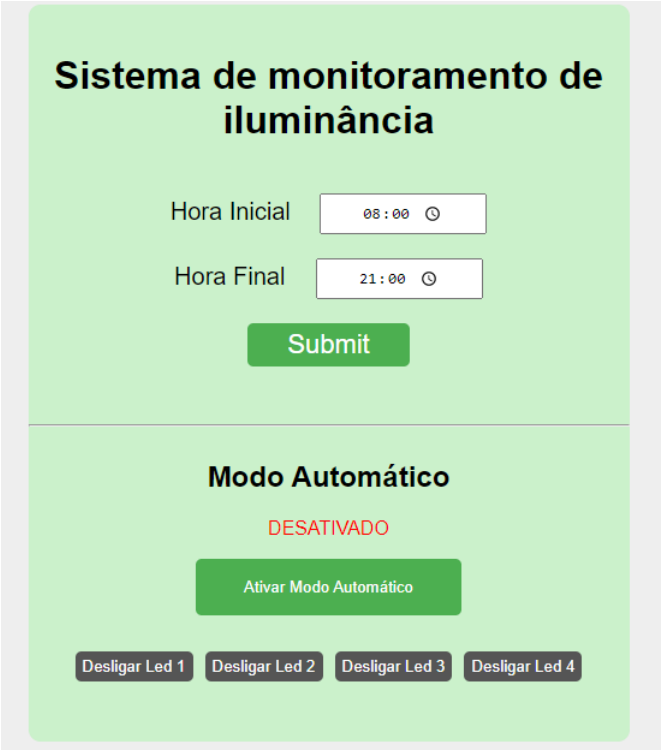

Figura 4.6: Servidor *web*. Fonte: A autora

4.4.2 Leitura do sensor TEMT6000

Para saber a medida lida pelo sensor em lux, foi preciso realizar algumas manipulações no valor obtido através da função *anologRead()*, tendo em vista que o *datasheet* do sensor descreve a relação da iluminância com a corrente em micro ampére.

Portanto, foi preciso realizar uma manipulação no dado lido, tendo em vista que o valor recebido é um dado digital que vai de 0 a 1023, pelo fato do arduino mega possui um conversor A/D de 10 *bits*, dando um total de 1024 valores. Para isso, foi feito uma transformação de valor digital para sua proporção em volts, afim de obter no final, um valor entre 0 a 5V e, após isso, de acordo com o gráfico da Figura 2.7, foi possível obter o valor mensurado em lux.

A equação total utilizada para a conversão de *bits* para lux é dada por:

$$V_{volts} = ((Valor_{digital} * 5Volts)/1024) \tag{4.1}$$

A partir disso, utilizando a fórmula $V = R*i$ onde V é o valor em volts, R, o valor do resistor interno do microcontrolador que é 10.000Ω e i a corrente. Foi possível obter o valor da corrente e transformá-la em micro ampere da seguinte forma:

$$i = (Valor_{volts}/10.000\Omega) * 10^6 \tag{4.2}$$

E, por fim, a partir da análise do gráfico da Figura 2.7, é possível obter uma relação entre lux e corrente da seguinte forma:

$$Valor_{lux} = 2 * i \tag{4.3}$$

Assim foi feita a conversão de *bits* para lux.

4.4.3 Leitura e manipulação dos dados do RTC

Para a leitura do RTC foi utilizada a biblioteca RTClib, descrita na subseção 3.3.4. A partir dela, foram utilizados alguns métodos, são eles: *rtc.begin() e rtc.now().*

Como o RTC utiliza a comunicação serial I2C, inicialmente foi feita a sua inicialização dentro do *void setup()*, com o auxílio do método *rtc.begin()*, para que, a partir disso, seja configurada e estabelecida a comunicação entre o arduino e o RTC. Após isso, foi utilizado o método rtc.now(), que nos retorna algumas informações, dentre elas, a hora e minuto atual. Essa informação foi armazenada em uma variável chamada "agora" utilizada para conferir se os horários inicial e final, configurados pelo usuário na página *web*, tenham sido atingidos e, a partir disso, iniciar ou finalizar a rotina de controle dos LEDs.

4.4.4 Controle dos LEDs

A etapa de controle dos LEDs foi implementada no código presente no arduino. Esse controle possui dois modos, o automático e o manual. Para ambos os modos, é configurado um horário *default* inicial e final da rotina do sistema, sendo substituídos caso o usuário insira e envie novos horários através da interface do servidor *web* transmitidos através da comunicação serial UART entre a esp32 e o arduino.

Para isso, após a definição das variáveis do sistema e inicialização do RTC, é desenvolvida a rotina de funcionamento do sistema, representada pelo fluxograma que pode ser visto na Figura 4.7.

A etapa de "Leitura da Serial" é responsável por ler as informações transmitidas via comunicação serial UART pela esp32 ao arduino, através do método *readSerial()* implementado, que retorna um *buffer* com a mensagem lida na Serial. Como a informação é transmitida *bit* a *bit*, o *buffer* é iterado até que o método *Serial.available()* seja igual a 0, significando que a transmissão foi finalizada e não há mais *bytes* para serem lidos [38]. Sabendo que o tamanho da informação é sempre igual, quando enviada, é realizado o seu tratamento, na etapa seguinte, atribuindo cada dado às suas variáveis correspondentes.

As etapas de “Leitura do sensor” e “Leitura do RTC e verificação da hora atual” são descritas nas subseções 4.4.2 e 4.4.3, respectivamente.

Após isso, é realizado o controle dos LEDs, tanto para o modo manual quanto para o modo automático, de acordo com os valores da hora atual e do modo de funcionamento. Para o modo automático, a variável utilizada como o status do modo automático, inicializada como *true* e atualizada pelo usuário na interface *web* do sistema, deve ser *true*. Caso isso ocorra, é feita a comparação entre o valor de lux medido pelo sensor de iluminância no estado “leitura do sensor”, um valor de referência de limite máximo, escolhido como 620lux, pois, de acordo com testes realizados, um LED agrega em torno de 120lux de iluminância ao ambiente interno utilizado, e o valor de iluminância mantida do ambiente, limite mínimo, definida como 500lux pela norma ABNT NBR ISO/CIE 8995-1 para o ambiente escolhido.

Essa comparação ocorre da seguinte forma, caso o valor em lux seja maior que 620lux, é decrementada de uma unidade a variável *leds_acesos*. Caso o valor de lux seja menor que 500lux, é incrementada de uma unidade a variável *leds_acesos*. Dessa forma, o padrão de iluminância será sempre mantido e respeitará o limite mínimo normativo de 500lux e o limite máximo escolhido para o sistema, fazendo com que, caso a iluminância natural do ambiente some mais de 120lux ao valor total mensurado, será possível economizar a energia emitida por pelo menos um LED, cumprido com a motivação inicial do projeto, que é respeitar a norma em todos os horários de expediente de trabalho e otimizar a economia de energia do local.

Já no modo manual, caso, por alguma razão, o usuário escolha mudar o modo automático para o manual, os LEDs serão todos ligados independente do status anterior e o controle deles pode ser feito através da interface *web*, onde o usuário poderá ligar/desligar o LED que desejar. É importante, novamente, mencionar que o ambiente foi projetado para respeitar a iluminância mantida prevista em norma em qualquer horário do dia, ou seja, todos LEDs ligados, mensuram um total de aproximadamente 500lux, portanto o controle manual está a cargo do usuário.

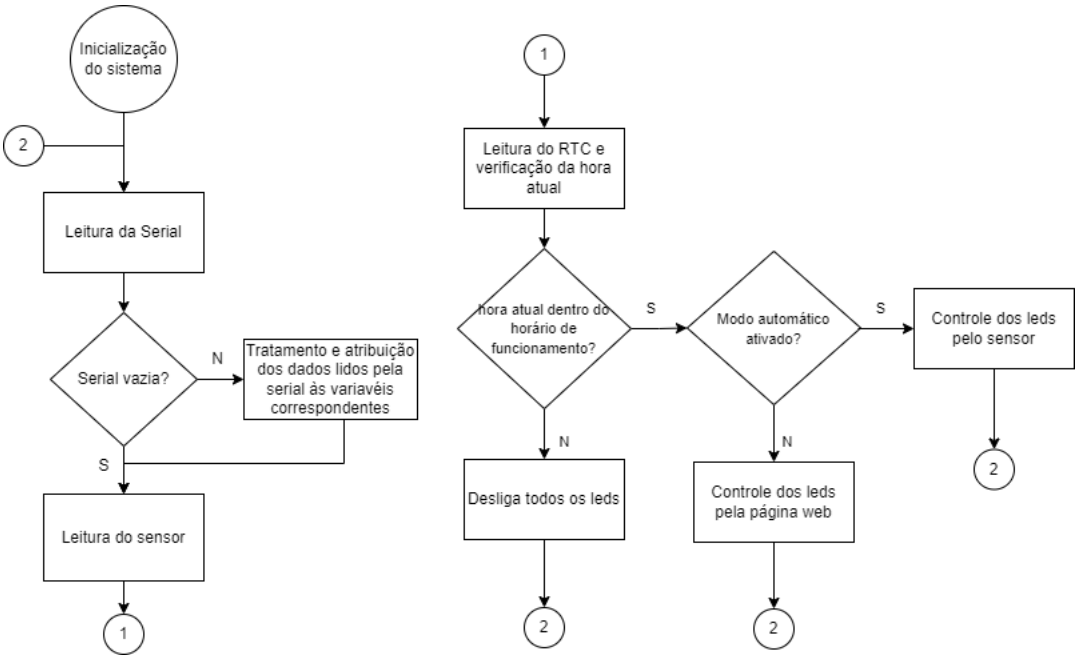

Figura 4.7: Fluxograma da rotina do sistema implementada. Fonte: A autora

Capítulo 5

Resultados

Neste capítulo será apresentado o resultado final do sistema com uma demonstração das diferentes funcionalidades desenvolvidas no projeto. Serão mostrados os dois modos implementados para o sistema: automático e manual, tanto para o projeto de *hardware* quanto para o projeto de *software*.

5.1 Inicialização da rotina do sistema

Como dito no Capítulo 4, a configuração das horas inicial e final da rotina do sistema acontece tanto para o modo manual quanto para o automático. Tendo em vista isso, o sistema é inicializado com uma configuração de horas *default*, inciando às 8:00 e finalizando às 21:00, até que o usuário envie alguma requisição através da interface *web* para alterar esses horários. Para isso, o usuário precisa configurar os novos horários nos campos "Hora Inicial" e "Hora Final" e clicar no botão de *"Submit"* enviando, assim, a requisição *Hypertext Transfer Protocol* (HTTP) do tipo *GET*, através do servidor *web* implementado. Fora do horário de funcionamento definido, os LEDs se encontram desligados.

O sistema inicia a rotina, após a hora inicial configurada ser atingida, com todos os LEDs acesos e no modo automático, como demonstrado nas Figuras 5.1 e 5.2 onde pode ser visto o *hardware* no momento de inicialização do sistema, assim como a interface *web* e o monitor serial com a mensagem "Sistema de monitoramento em

funcionamento" enviada assim que o horário de funcionamento se inicia. Além disso, é possível observar no monitor serial o valor mensurado pelo sensor de iluminância, no início do sistema, de aproximadamente 520lux.

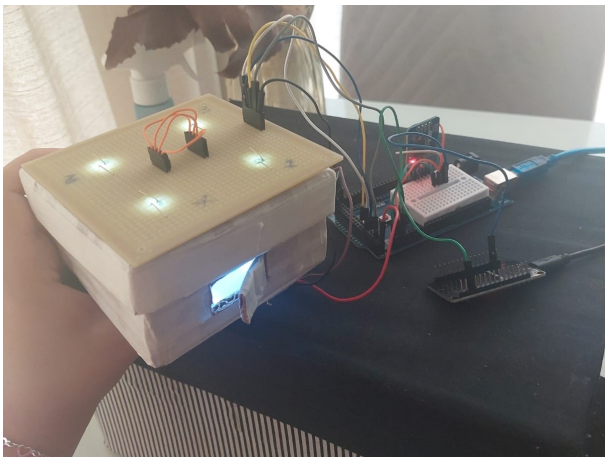

Figura 5.1: *Hardware* ao inicializar o sistema . Fonte: A autora

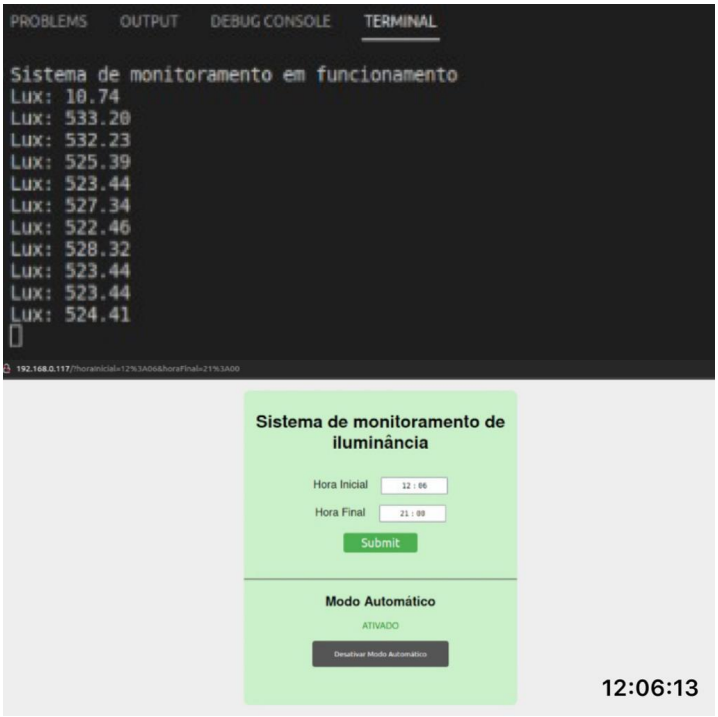

Figura 5.2: Monitor serial e interface *Web* ao inicializar o sistema. Fonte: A autora

5.2 Modo automático

No modo automático, a atuação do usuário será apenas na configuração das horas, caso assim desejar, pois o controle dos LEDs é feito através do arduino. Para a prova de conceito, foi utilizada a lanterna de um celular como luz externa, conforme a Figura 5.3, onde foi realizada uma mudança na intensidade da lanterna para demonstrar que os LEDs respondem à variação da iluminância no ambiente.

No monitor serial da Figura 5.4, é possível observar o momento em que é adicionada a luz externa ao sistema. Assim, é iniciado o processo de readequação da iluminância, desligando a quantidade de LEDs necessários para que o sistema volte ao seu estado padrão de iluminância mantida, no caso em que a iluminação externa sozinha tenha o mesmo valor da iluminância mantida normativa, de 520lux.

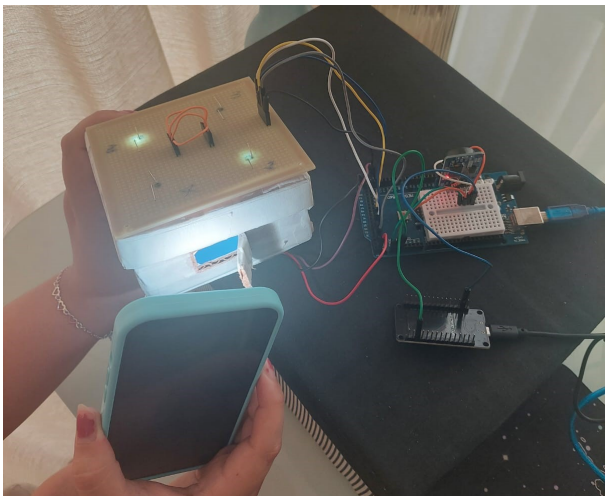

Figura 5.3: *Hardware* ao ser exposto a uma iluminância externa ao ambiente.
Fonte: A autora

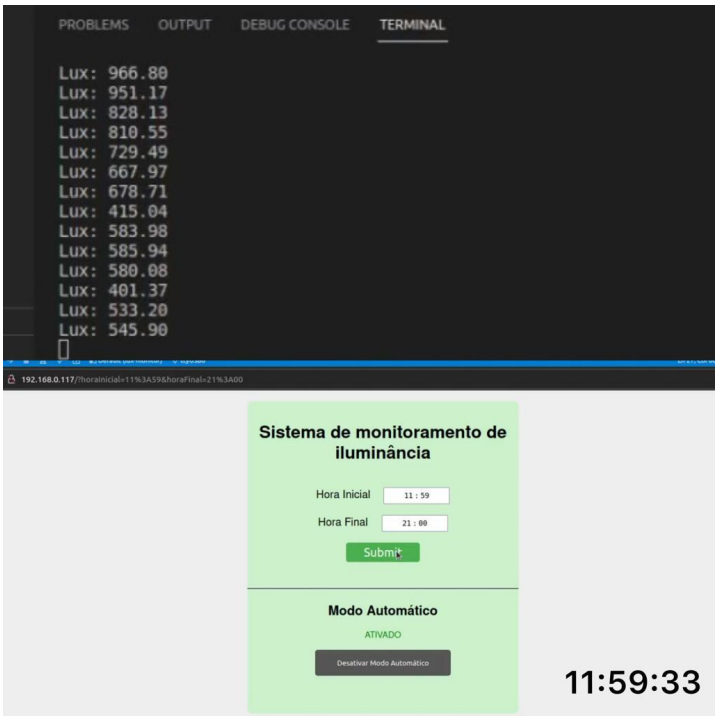

Figura 5.4: Monitor serial e interface *Web* ao ser exposto a uma iluminância externa ao ambiente. Fonte: A autora

Da mesma forma, ao diminuir a intensidade de iluminância externa ao ambiente, o processo de readequação da iluminância é realizado, verificando se o valor mensurado em lux, pelo sensor, esteja abaixo de 500lux. Em caso positivo, são ligados os LEDs necessários para que a iluminância mantida seja respeitada. Na Figura 5.5 é possível observar os valores em lux da iluminância mensurada no momento em que é diminuída a intensidade da lanterna celular até que seja nula. É possível observar o valor de lux decrescendo e logo após sendo incrementado através da ação de ligar os LEDs de acordo com o necessário, retornando para o padrão de iluminância mantida esperado.

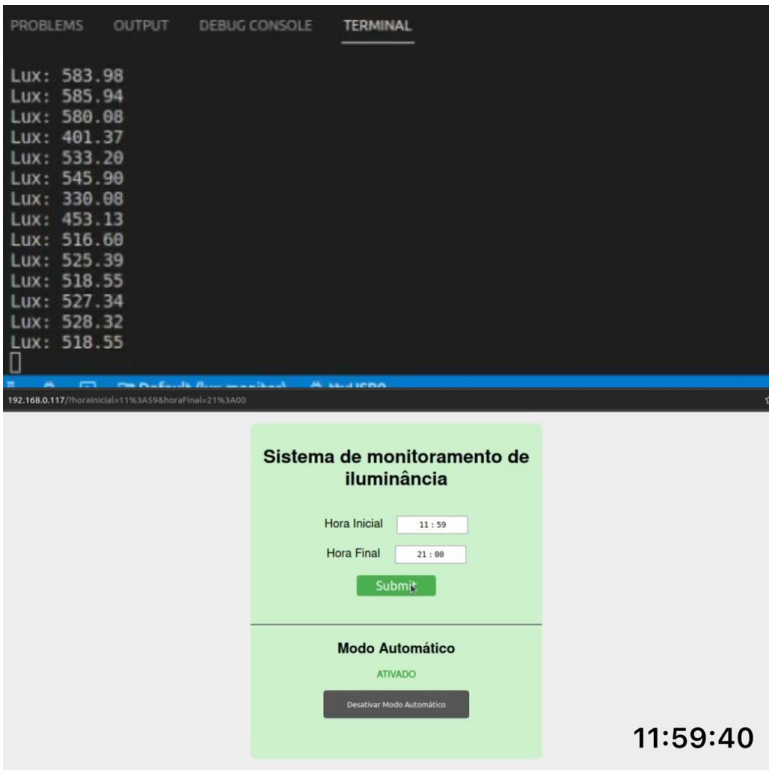

Figura 5.5: Monitor serial e interface *Web* no momento em que é diminuído a exposição de iluminância externa ao ambiente. Fonte: A autora

5.3 Modo manual

No modo manual, os estados dos LEDs dependem diretamente da ação do usuário através da interface *web*. Este modo foi desenvolvido em caso de, por alguma razão, o usuário desejar desativar o modo automático do sistema. Assim, ao ser configurado o modo manual pelo usuário, através do botão "Desativar modo automático" da interface e o horário atual esteja dentro do horário de funcionamento do sistema, todos os LEDs são ligados e 4 botões são mostrados na interface *web* referentes aos 4 LEDs, onde o usuário poderá, caso necessário, desligar ou ligar os LEDs um a um. De fato, o modo manual não é o mais indicado para o sistema, pois desligando os LEDs de forma aleatória não necessariamente irá respeitar a iluminância mantida do sistema prevista por norma. Por isso, este modo é apenas em caso de alguma necessidade urgente do usuário. Nas Figuras 5.6 e 5.7 é demonstrado o caso manual em que o usuário deixa apenas dois LEDs ligados.

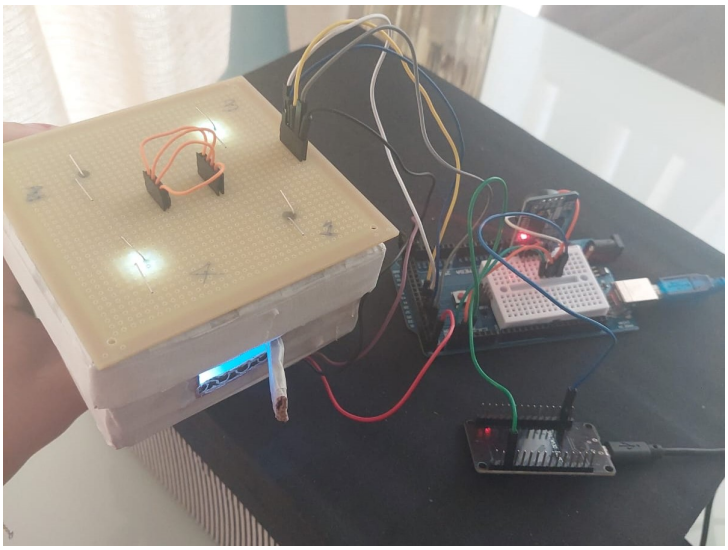

Figura 5.6: *Hardware* no modo manual com 2 LEDs ligados. Fonte: A autora

Na Figura 5.7, no monitor serial, pode ser observado o momento que antecede a alteração do usuário, onde a mensagem enviada via comunicação UART, pela esp32 é lida pelo arduino contendo os dados de hora inicial, hora final, status do modo automático e status dos LEDs. Ao desligar os dois LEDs, é possível ver a alteração na mensagem relacionada ao status, em *boolean*, dos LEDs, onde 0 significa desligado e 1 ligado. Logo, o primeiro e segundo LEDs estão com status 0 na última linha do monitor serial e os demais estão com status 1, como esperado.

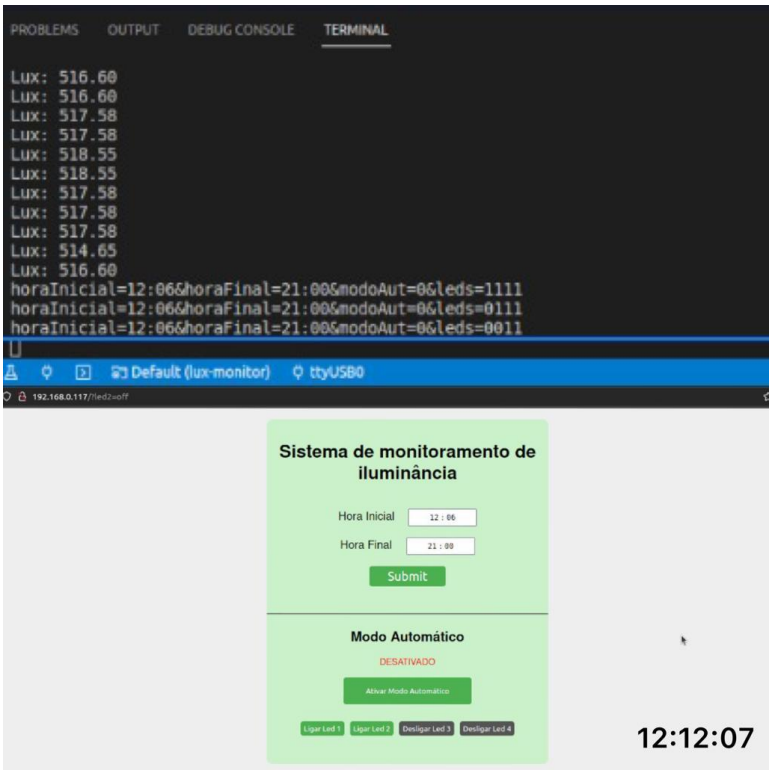

Figura 5.7: Monitor serial e interface *Web* no modo manual com 2 LEDs ligados. Fonte: A autora

5.4 Finalização da rotina do sistema

A finalização da rotina do sistema se dá no momento em que a hora atual ultrapassa a hora final configurada, seja a *default* ou a configurada pelo usuário através da interface *web*. Ao finalizar a rotina, a mensagem "Sistema de monitoramento fora do horário de funcionamento" disparada no monitor serial, como visto na Figura 5.9 e os LEDs são todos desligados, como visto na Figura 5.8.

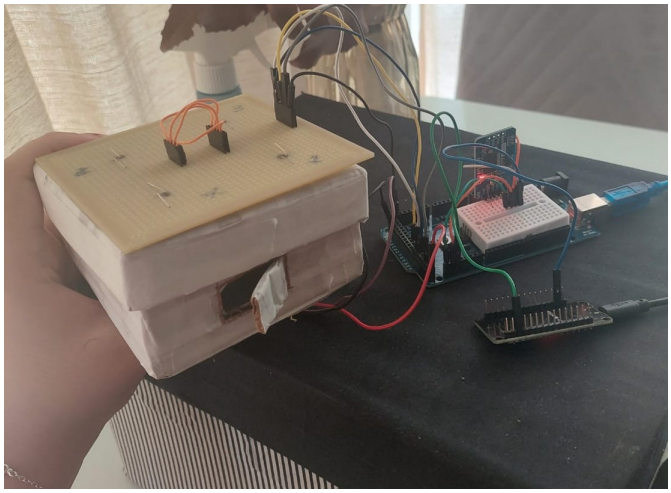

Figura 5.8: *Hardware* ao finalizar rotina do sistema. Fonte: A autora

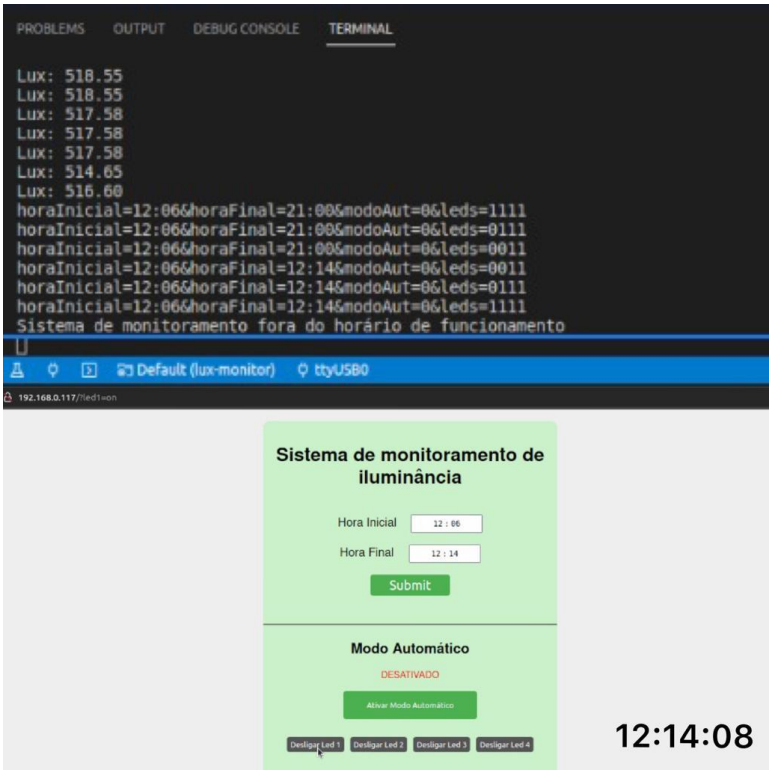

Figura 5.9: Monitor serial e interface *Web* ao finalizar rotina do sistema. Fonte: A autora

Capítulo 6

Conclusão

Este projeto teve como objetivo apresentar uma solução para uma questão real, motivada pela norma ABNT NBR ISO/CIE 8995-1, referente à iluminância em ambientes internos de trabalho, envolvendo conceitos e tecnologias vistos no decorrer do curso de Engenharia Eletrônica.

No trabalho foi possível transitar por todas as etapas necessárias para o desenvolvimento de um *software* embarcado, desde a análise de requisitos até a implementação do sistema. A definição de escopo foi bastante importante para que as demais etapas fossem realizadas, pois foi a partir do escopo do projeto que foram definidos as ferramentas e componentes utilizados.

No *hardware*, foi preciso analisar a melhor forma de simular um ambiente interno de trabalho através de um protótipo. Além disso, pelo fato de utilizar dois microcontroladores para dividir tarefas distintas, foi preciso realizar uma comunicação serial UART para obter as informações de um microcontrolador para o outro.

No *software*, além do controle dos LEDs, foi desenvolvido um servidor *web* responsável por coletar os dados inseridos pelo usuário para que, através da comunicação serial, fossem enviados da esp32 para o arduino. Algumas ferramentas foram indispensáveis para o funcionamento do sistema, como a RTCLib, auxiliando na comunicação com o módulo RTC, através do barramento I2C e a ESPAsyncWebServer, responsável por todos os métodos utilizados para a implementação do servidor *web*.

6.1 Dificuldades Encontradas

No decorrer do projeto, algumas dificuldades foram encontradas, principalmente na etapa de projeto de *hardware*.

Inicialmente, uma dificuldade encontrada foi construir um protótipo que correspondesse a um ambiente interno de trabalho, com a iluminância mantida prevista por norma, pois, como foram utilizados LEDs de alto brilho para simular lâmpadas, eles não possuem uma quantidade de lumens similar. Isso foi solucionado, utilizando um ambiente interno pequeno suficientemente para atingir a iluminância mantida definida.

Além disso, apesar do *datasheet* do sensor de iluminância escolhido indicar que o sensor comporta uma alimentação de 3.3V ou 5V, com teste realizados, percebeu-se que utilizando a tensão de 3.3V o sensor não se comportou satisfatoriamente, apresentando erros de leitura. Já com uma tensão de 5V, no arduino, o sensor respondeu de forma satisfatória, de acordo com o esperado.

6.2 Trabalhos Futuros

Ao final de um projeto, é sempre possível observar possíveis melhorias para versões futuras de um sistema. No mercado de trabalho, a atualização, visando melhorias, de diversos sistemas é feita de forma bastante recorrente. Neste trabalho, também é possível destacar algumas melhorias para versões futuras.

- Utilizar um *level shifter* para permitir a conversão de nível lógico do arduino, equivalente a 5v para o da esp32, 3.3v, possibilitando *feedbacks* na interface *web* referente a dados transmitidos do arduino para a esp32, como por exemplo, a informação de que o sistema está em funcionamento ou o valor de lux atualizado, informando se está abaixo da iluminância mantida, no caso do modo manual;

- Melhoria do protótipo de *hardware* desenvolvido, evitando uso de *jumpers* ainda mais e reduzindo possíveis maus contatos;

- Interface de *login* para usuários, afim de aumentar a segurança, permitindo que somente usuários autenticados tenham acesso ao servidor *web*;
- Adicionar uma segunda interface de usuário utilizando um *Liquid Crystal Display* (LCD) por exemplo, para não depender de uma conexão com a internet;

Com a adição dessas melhorias, o sistema ficaria mais robusto a nível de *hardware* e *software* pois estaria ainda mais estável com as melhorias no protótipo e com uma interface de usuário mais completa com a adesão de *feedbacks* ao usuário, diretamente pela página *web*. Além disso, o sistema teria uma segurança maior através da interface de *login* e autenticação de usuário.

Referências Bibliográficas

[1] "Indústria 4.0: o que é e quais os benefícios". . Disponível em: <https://www.industria40.ind.br/artigo/16485-industria-40-o-que-e-e-quais-os-beneficios/>.

[2] "A quarta revolução industrial – A história até chegar nela!" . Disponível em: <https://engprocess.com.br/a-quarta-revolucao-industrial/#>.

[3] "Indústria 4.0: Revolução ou evolução?" . Disponível em: <https://www.telios.eng.br/site/industria-4-0-revolucao-ou-evolucao/>.

[4] "As inovações geradas pela indústris 4.0". . Disponível em: <https://www.ccaexpress.com.br/blog/as-inovacoes-geradas-pela-industria-4-0/>.

[5] "Industria 4.0". . Disponível em: <https://fernandonogueiracosta.wordpress.com/2018/05/19/industria-4-0/>.

[6] "Internet das coisas". Disponível em: <https://djpautomacao.com/internet-das-coisas-aprenda-como-usa-la-em-favor-da-sua-industria/>.

[7] "ISO/CIE 8995-1: Iluminação de ambientes de trabalho Parte 1: Interior". Disponível em: <https://www.drb-m.org/av1/NBRISO_CIE8995-1.pdf>.

[8] "Luminotecnica". . Disponível em: <https://www.guiadaengenharia.com/luminotecnica/>.

[9] "O que é iluminância". . Disponível em: <https://kian.com.br/o-que-e-iluminancia/>.

[10] "Como usar Luxímetrio". Disponível em: <https://blog.instrusul.com.br/como-usar-luximetro/>.

[11] "LDR: o que é e como funciona". Disponível em: <https://www.manualdaeletronica.com.br/ldr-o-que-e-como-funciona/>.

[12] “Sistemas Embarcados”. Disponível em: <https://www.techtarget.com/iotagenda/definition/embedded-system>.

[13] “Sistemas embarcados e microcontroladores”. . Disponível em: <https://embarcados.com.br/sistemas-embarcados-e-microcontroladores/>.

[14] “Microcontrolador: História e aplicações”. . Disponível em: <https://webautomacaoindustrial.blogspot.com/2016/05/microcontrolador-historia-e-aplicacoes.html>.

[15] “Arduino Mega”. . Disponível em: <https://embarcados.com.br/arduino-mega-2560/>.

[16] “O que é o Arduino?” . Disponível em: <https://www.usinainfo.com.br/blog/o-que-e-arduino/>.

[17] “Conhecendo o node MCU”. . Disponível em: <https://blogmasterwalkershop.com.br/embarcados/esp32/conhecendo-o-nodemcu-32s-esp32>.

[18] “O que é e para que serve a esp32”. . Disponível em: <https://lobodarobotica.com/blog/o-que-e-esp32-pra-que-serve-quando-usar/>.

[19] “Conhecendo o esp32”. . Disponível em: <https://curtocircuito.com.br/blog/Categoria%20IoT/conhecendo-esp32>.

[20] “Sensor de luz TEMT6000”. Disponível em: <https://autocorerobotica.blog.br/sensor-de-luz-temt6000-com-arduino/>.

[21] “Datasheet TEMT600”. Disponível em: <https://www.sparkfun.com/datasheets/Sensors/Imaging/TEMT6000.pdf>.

[22] “Módulo RTC pinout”. . Disponível em: <https://components101.com/modules/ds3231-rtc-module-pinout-circuit-datasheet>.

[23] “Módulo RTC”. . Disponível em: <https://mundoprojetado.com.br/modulo-rtc-ds1302/>.

[24] “Linguagem C++”. . Disponível em: <https://www.infoescola.com/informatica/cpp/>.

[25] “Linguagem de programção C++”. . Disponível em: <https://blog.betrybe.com/linguagem-de-programacao/cpp/>.

[26] “Linguagens de programação para sistemas embarcados”. Disponível em: <https://embarcados.com.br/editorial-linguagens-para-sistemas-embarcados/>.

[27] “Como programar o Arduino com o Visual Studio Code e PlatformIO IDE”. Disponível em: <https://embarcados.com.br/arduino-vscode-platformio/>.

[28] “Real Time Clock - Primeiros Passos”. . Disponível em: <https://www.robocore.net/tutoriais/primeiros-passos-rtc-ds1307-com-arduino/>.

[29] “Comunicação I2C”. Disponível em: <https://blog.eletrogate.com/comunicacao-i2c-entre-arduinos/>.

[30] “ESPAsyncWebServer”. . Disponível em: <https://github.com/me-no-dev/ESPAsyncWebServer>.

[31] “Web Server Assíncrono com esp32”. . Disponível em: <https://blog.eletrogate.com/webserver-assincrono-com-esp32/>.

[32] “Diferenças entre lúmens e luxes”. Disponível em: <https://www.dicasled.pt/diferencas-entre-lumens-e-luxes/>.

[33] “Site wokwi”. Disponível em: <https://wokwi.com/>.

[34] “Comunicação serial”. . Disponível em: <https://www.programmingboss.com/2021/04/esp32-arduino-serial-communication-with-code.html>.

[35] “Comunicação serial entre arduino”. . Disponível em: <https://linuxhint.com/serial-uart-communication-between-two-arduino/>.

[36] “Comunicação serial arduino via protocolo UART”. . Disponível em: <https://www.makerhero.com/blog/comunicacao-serial-arduino-via-protocolo-uart/>.

[37] “Entradas analógicas arduino”. Disponível em: <https://embarcados.com.br/arduino-entradas-analogicas/>.

[38] “Serial Available”. Disponível em: <https://arduinogetstarted.com/pt/reference/serial-available>.