CENTRO UNIVERSITÁRIO DE BRASÍLIA- UniCEUB

FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADAS – FATECS

CURSO DE ENGENHARIA DA COMPUTAÇÃO

AMBER LEITE DE AZEVEDO JUNIOR

SISTEMA DE MONITORAMENTO E CLIMATIZAÇÃO DE ESTUFA DE PEQUENO PORTE EM UM CONTEXTO DOMÉSTICO

BRASÍLIA – DF

JULHO, 2016

AMBER LEITE DE AZEVEDO JUNIOR

SISTEMA DE MONITORAMENTO E CLIMATIZAÇÃO DE ESTUFA DE PEQUENO PORTE EM UM CONTEXTO DOMÉSTICO

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.
Orientador: Prof. MsC. Francisco Javier De Obaldía Díaz

Brasília

Julho, 2016

Curso de Engenharia de Computação

AMBER LEITE DE AZEVEDO JUNIOR

SISTEMA DE MONITORAMENTO E CLIMATIZAÇÃO DE ESTUFA DE PEQUENO PORTE EM UM CONTEXTO DOMÉSTICO

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.

Este Trabalho foi julgado adequado para a obtenção do Título de Engenheiro de Computação, e aprovado em sua forma final pela Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS.

_____
Prof. Dr. Abiezer Amarilia Fernandes
Coordenador do Curso

Banca Examinadora:

_____
Prof. Francisco Javier de Obaldía Díaz, MsC.
Orientador

_____
Prof. Nilo Sérgio Soares Ribeiro, MsC.
UniCEUB

_____
Prof. Roberto Avila Paldês, MeS.
UniCEUB

AGRADECIMENTOS

Agradeço aos meus pais, Amber e Lira, e à minha irmã, Débora, por terem me proporcionado não apenas o suporte necessário para realização deste trabalho, mas em todos os aspectos da vida.

Agradeço a minha namorada, Camila, por estar sempre ali quando precisei, pela paciência nos momentos difíceis e pelo amor.

Aos amigos e colegas de curso, obrigado pela ajuda e suporte providos por vocês ao longo dos anos, em especial aqueles que estão comigo desde o início.

Aos meus professores, sem vocês nada disso seria possível. Obrigado por dedicarem suas vidas a ensinar as futuras gerações.

A equipe da Embrapa Hortaliças de Brasília, em especial ao Dr. Juscimar e o Dr. Marcos, obrigado pela ajuda e ensinamentos a qual me proporcionaram.

Ao professor Francisco Javier, obrigado por todo o apoio durante a realização deste trabalho, sua conclusão não seria possível sem a sua ajuda.

"A imaginação é mais importante que o conhecimento. O conhecimento é limitado, enquanto a imaginação abarca o mundo inteiro, estimulando progresso, dando luz a evolução. Ela é, estritamente falando, um fator real na pesquisa científica."

Albert Einstein

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AC/DC – Alternating Current/Direct Current

Com – Common Connection

I/O – Input/Output

ICSP – In Circuit Serial Programming

IDE – Integrated Development Environment

LCD – Liquid-Crystal Display

NC – Normally Closed

NFT – Nutrient Film Technique

NO – Normally Open

PC – Personal Computer

RESUMO

O consumo de hortaliças folhosas é um importante hábito alimentar do brasileiro, em especial entre os residentes nas áreas urbanas, o que traz benefícios à saúde. Entretanto, o cultivo contínuo e eficaz destas plantas no nosso território enfrenta importantes dificuldades, tais como a variabilidade do clima e dos solos, a crescente urbanização que diminui as áreas eficientes para o plantio e a preocupação da população com o consumo de alimentos seguros que oferecem teor nutritivo adequado e qualidade física dos produtos. A criação e o desenvolvimento deste tipo de plantas em ambientes protegidos são capazes de minimizar consideravelmente os impactos negativos na produção deste gênero alimentar. O ambiente da estufa permite um controle e monitoramento elevado sob os fatores que influenciam o crescimento das plantas, protegendo o ciclo de efeitos climáticos adversos. Considerando este cenário, este projeto busca criar uma estufa de tamanho reduzido para a produção de hortaliças folhosas em um contexto doméstico, desenvolvendo o sistema através da plataforma Arduino, sensores e *hardwares* de apoio para atuarem no controle e monitoramento de uma estufa, tornando acessível e eficiente o crescimento e a produção destas plantas em ambientes protegidos de escala reduzida.

Palavras-chave: Arduino, Estufa, Irrigação Hidropônica, Produção de Hortaliças.

ABSTRACT

Leafy vegetables consumption is an important Brazilian eating habit, especially among those living in urban areas, which brings health benefits. However, the continuous and effective cultivation of these kind of plants in our country is facing major difficulties, such as the variability of the climate and soil, increasing urbanization which reduces the effective areas for planting and concern of the population with the consumption of safe food offering appropriate nutritional value and physical quality of these products. The creation and development of such plants in greenhouses can greatly minimize the negative impacts on the production of this food genre. The greenhouse environment allows a higher control and monitoring on the factors influencing plants growth, protecting the cycle from adverse climatic effects. Considering this scenario, this project aims to create a reduced-size greenhouse for the production of leafy vegetables in a domestic context, developing the system through the Arduino platform, sensors and *hardware* support to act towards controlling and monitoring a greenhouse, making it affordable and efficient the growth and the production of these plants in protected environments of reduced scale.

Key words: Arduino, Greenhouse, Hydroponic irrigation, Leafy vegetables production.

1 INTRODUÇÃO

1.1 Apresentação do Problema

Este trabalho possui como objetivo principal o desenvolvimento de um sistema de controle e monitoramento, instalado em uma estufa de proporções reduzidas para a produção de hortaliças folhosas em um contexto doméstico. Pretende-se examinar os principais fatores que afetam o crescimento deste tipo de hortaliça, tais como a umidade, a luminosidade, a umidade do solo e a temperatura do ambiente (BEZERRA, 2003, p.11). Para a elaboração do sistema proposto, consideramos a plataforma Arduino e seus *hardwares* de apoio, devido aos aspectos destacados por Yarnold (2015) que referencia suas vantagens: custo reduzido, funcionamento em múltiplos sistemas operacionais (Windows, Mac e Linux), simplicidade e flexibilidade, *software* e *hardware* livres.

O cultivo de hortaliças é um importante setor da agricultura brasileira. Segundo Vilela (2013), a área cultivada foi de 800 mil hectares com produção de 18.769 mil toneladas em 2012. Entre os produtos preferenciais estão a batata, o tomate, a cebola, além das leguminosas. Nos hábitos alimentares do brasileiro, as hortaliças aparecem como elemento importante no cardápio dos grandes centros urbanos (BEZERRA, 2003). O modo de vida destas populações encontra afinidades em relação a uma dieta diversificada composta por hortaliças, o que permite ter acesso ao seu alto valor nutritivo, composto por sais minerais e vitaminas, proporcionando sensação de saciedade por um período maior de tempo e facilitando o processo digestivo (Ibid., p.9). Por isso, suas áreas de plantio são geralmente próximas das cidades, o que por um lado diminui os custos de transporte e desperdício dos alimentos, mas por outro acarreta custos elevados na produção, uma vez que as áreas, as variações climáticas e a mão-de-obra do entorno demandam maior investimento do produtor.

Guedes (2015) aponta que a variabilidade do clima e dos solos brasileiros, a crescente urbanização, a mudança de hábitos e valores para o consumo da população voltada para a preocupação com alimentos seguros, a qualidade física dos produtos são outros aspectos que influenciam na relação produtividade-consumo de hortaliças no Brasil. Desta forma, uma solução que minimizaria os impactos negativos na produção brasileira seria a criação e desenvolvimento destas plantas em ambientes protegidos:

> Atualmente o Brasil testemunha a expansão da produção de hortaliças em ambiente protegido, o que tem se tornado possível graças ao conhecimento, além do comportamento das plantas e do solo, da física do ambiente interno nesse tipo de estrutura (Ibid., p.2).

Os ambientes protegidos permitem o aumento da eficiência na produção, no que se refere aos principais fatores que influenciam no desenvolvimento destas plantas: o uso controlado da água, dos insumos, fertilizantes e luminosidade. Bezerra (2003) indica os benefícios do cultivo de mudas de hortaliças nesses espaços:

> Com o advento do sistema de cultivo protegido, a produção de mudas, em geral, vem apresentando um nível tecnológico mais elevado, resultando em material de qualidade com riscos bastante reduzidos. Dessa forma, o produtor pode elaborar um cronograma de produção de mudas por um período maior e, consequentemente, obter melhor remuneração, como também maior estabilidade dos preços das mudas durante o ano, uma vez que fatores ambientais como temperatura, umidade, luminosidade, dentre outros, podem ser controlados, proporcionando um microclima favorável, principalmente nos estádios iniciais de desenvolvimento das mudas (Ibid., p.11).

Os ambientes protegidos possuem outras vantagens para o desenvolvimento de hortaliças de acordo com Bezerra (2003):

- Maior precocidade.
- Menor possibilidade de contaminação fitopatogênica.
- Maior relação percentual entre sementes plantadas e mudas obtidas.
- Melhor aproveitamento da área destinada à produção de mudas.
- Maior facilidade na execução de tratos culturais como desbaste, irrigação, adubação, tratamento fitossanitário.
- Menor estresse por ocasião do transplantio.
- Redução do ciclo da cultura do campo (Ibid., p. 9-10).

Estes aspectos justificam a importância da produção de hortaliças em ambientes protegidos, estruturas que diminuem os efeitos climáticos que incidem sobre as plantações agrícolas, preocupando-se com uma maior eficiência da produção. As telas plásticas, por exemplo, são ambientes que permitem a proteção das colheitas do frio e da chuva, de ventos fortes, além de criar condições favoráveis para seu desenvolvimento inicial. Porém, esta forma de proteção não permite o controle de fatores climáticos, tais como a variação abrupta de temperatura ou a luminosidade necessária ao longo do processo de desenvolvimento das plantas (JENSEN; MALTER, 1995).

O ambiente da estufa, por outro lado, permite um controle e monitoramento elevado sobre os fatores que influenciam o crescimento das plantas, sendo esta uma estrutura composta de paredes e telhado de plástico ou vidro, protegendo todo o ciclo dos efeitos climáticos adversos (Ibid., p. 7).

Entretanto, Guedes (2015) informa sobre as dificuldades de implementação desses espaços em toda a extensão do território brasileiro, considerando o alto custo dos materiais e

dos recursos elétricos para o controle de temperatura, do desenvolvimento tecnológico restrito e do desequilíbrio nos preços oferecidos ao consumidor. Estes problemas interferem no consumo doméstico médio de hortaliças que tem decrescido no país, ampliando a baixa eficiência da nossa produtividade.

Neste cenário, pretende-se apontar para a viabilidade técnica da criação de uma estufa em tamanho reduzido, para uso doméstico, que permita uma produção urbana alternativa que atenda parte das necessidades diárias de consumo de hortaliças folhosas nos ambientes particular e familiar brasileiro, principalmente para o pequeno agricultor com recursos limitados. A programação a ser utilizada na estufa deste trabalho permite a configuração de variáveis como limite de temperatura e umidade, que serão utilizadas para a realização de tarefas de controle do ambiente. Neste trabalho é utilizado um microcontrolador e sensores para o monitoramento e controle de alguns dos principais fatores do desenvolvimento de hortaliças em uma estufa, sendo eles: a quantidade de água, a temperatura interna, a luminosidade e a umidade do ar.

1.2 Objetivos

1.2.1 Objetivo Geral

Desenvolver e implementar um sistema de monitoramento e controle de ambiente em uma estufa reduzida, permitindo a automação de parte do processo de cultivo de hortaliças folhosas em ambientes domésticos.

1.2.2 Objetivos Específicos

Os objetivos específicos identificam aspectos relevantes para o desenvolvimento do projeto, pautando-se em:

- Realizar uma revisão bibliográfica de todas as tecnologias a serem utilizadas no projeto;
- Desenvolver um *software* embasado na plataforma Arduino, capaz de realizar a leitura e o registro das informações da estufa provenientes dos sensores instalados, sendo elas a luminosidade, a umidade do ar e a temperatura, além de executar comandos à fim de controlar estas grandezas;
- Construir uma estufa em material transparente a ser controlada pelo *software* criado;

- Instalar o sistema proposto na estufa criada e testar seu funcionamento;
- Realizar testes de *software* e de *hardware*.

1.3 Motivação

O acesso a uma dieta diversificada através do consumo de hortaliças pela população brasileira esbarra em diferentes fatores negativos que influenciam o trajeto necessário para a eficiência das etapas de produção (qualidade de sementes, cultivo de mudas, desenvolvimento das plantas e consumo), como destacam Bezerra (2003) e Guedes (2015). Ao se projetar a criação de uma estufa automatizada e alternativa para ambientes domésticos, pretende-se atentar para a democratização e acessibilidade da produtividade:

> Historicamente o consumo de hortaliças no Brasil tem permanecido baixo apesar de o país ser internacionalmente reconhecido como uma superpotência agrícola. As razões para esse aparente paradoxo são diversas e de resolução relativamente difícil. Há razões culturais, mas em geral concorda-se que o alto custo das hortaliças é uma das principais causas do baixo consumo (GUEDES, 2015, p. 1).

A concepção de uma estufa de tamanho reduzido advém da necessidade de otimizar o espaço e a produção em ambientes pequenos, buscando, ao mesmo tempo, uma maior eficiência neste ciclo. A estufa é um sistema complexo, pois qualquer alteração brusca no clima pode gerar um efeito adverso no desenvolvimento das plantas. Sendo assim, a confecção do sistema de monitoramento e controle automático oportunizado pela aplicação dos conceitos adquiridos ao longo do curso de Engenharia de Computação, permite a redução dos esforços diários e manuais necessários para a eficiência no trabalho de criação de hortaliças, no interior do ambiente protegido proposto.

A contribuição desta pesquisa, portanto, reside em desenvolver uma forma alternativa de cultivo de hortaliças para consumo próprio, atentando-se para as possibilidades de baixo custo de automação e democratização do acesso aos produtos alimentícios.

Através do desenvolvimento deste projeto, obter-se-á uma estufa de pequeno porte a ser utilizada em ambientes urbanos para o cultivo de hortaliças.

1.4 Resultados esperados

Espera-se deste projeto a criação de um sistema baseado na plataforma Arduino, capaz de monitorar a luminosidade, temperatura e umidade do ar de uma estufa, bem como efetuar o

controle de seu sistema de irrigação e subsistemas que ajudem a manter os níveis desejados destes fatores. O sistema mostrará informações dos dados da estufa em um visor de cristal líquido (*LCD - Liquid-Crystal Display*), sendo eles: umidade do ar, temperatura e luminosidade. Estes dados serão transmitidos via Internet para uma página da *web* da plataforma *ThingSpeak*, que poderá ser monitorada *online* pelo usuário.

Ainda, espera-se que a estufa construída neste projeto opere em condições apropriadas por definição do usuário de temperatura, irrigação, iluminação e umidade do ar para o desenvolvimento automatizado das hortaliças cultivadas em ambientes protegidos.

1.5 Estrutura do Trabalho

A estrutura deste trabalho contempla os seguintes capítulos:

Capitulo 1 – apresenta a introdução do trabalho, seus objetivos, metodologia, justificativa e organização da monografia.

Capítulo 2 – descreve o funcionamento de uma estufa comum, assim como as tecnologias que serão utilizadas para seu funcionamento, destacando os fundamentos teóricos utilizados para o sistema proposto.

Capítulo 3 – consiste na apresentação da solução proposta, detalhando seu processo de criação e desenvolvimento.

Capítulo 4 – destina-se a análise dos resultados que foram obtidos no sistema proposto, tais como os objetivos esperados e os problemas encontrados.

Capítulo 5 – aponta a conclusão do trabalho e sugestões para futuras pesquisas.

2 REFERENCIAL TEÓRICO

Este capítulo destaca os fundamentos teóricos que orientam o processo de cultivo de hortaliças folhosas em estufas, descrevendo o funcionamento desta estrutura em formato reduzido, os fatores climáticos que interferem na sua produção e as tecnologias que serão utilizadas para o desenvolvimento do sistema proposto.

2.1 Cultivo de Hortaliças em Estufas

Hortaliça é um termo agrícola utilizado para nomear uma planta herbácea em que uma ou mais partes são utilizadas para alimentação em sua forma natural (ANVISA, 1978). Luengo (2001) afirma que o consumo de hortaliças traz uma série de benefícios para a população, devido a suas propriedades: ricas em fibras, vitaminas e sais minerais, além de serem alimentos leves, tanto do ponto de vista nutricional, quanto pelo seu peso.

O processo de cultivo de hortaliças pode ser caracterizado como um conjunto de cuidados especiais destinados a contribuir com o desenvolvimento destas plantas. Fatores como o substrato, a irrigação da água, a temperatura, a luminosidade, a umidade do ar e do solo interferem diretamente no processo de seu crescimento podendo dificultar, ou até mesmo, impedir seu progresso. Ao ressaltar a produção de mudas de hortaliças que tenham um melhor rendimento para o ciclo da planta, Bezerra (2003) aponta que os ambientes protegidos proporcionam "*um microclima favorável*" para o seu desenvolvimento.

A formação de um "*microclima*" que permita o controle das variáveis que incidem sobre a produção de uma planta influencia significativamente o seu desenvolvimento, não só no caso das hortaliças. As orquídeas, por exemplo, necessitam de condições apropriadas para seu crescimento e alguns fatores, como a quantidade de água, a luminosidade, a temperatura, umidade do ar e ventilação, determinam o sucesso de sua produção (PLOUGHMAN, 2007).

Atualmente, o uso de ambientes controlados e protegidos através das estufas é uma solução comumente adotada para o sucesso no cultivo de plantas variadas independente da região brasileira. Bezerra (2003) define estufas como "*estruturas onde se pode criar e/ou manter microclimas favoráveis ao cultivo de qualquer espécie de planta, independente das condições ambientais existentes*". (Ibid., 11-12). Além disso, a adoção das estufas permite o controle das condições ambientais e de pragas ou insetos nocivos que possam vir a danificar a produção, como formigas, lesmas e abelhas (BEZERRA, 2003; FIGUEIREDO, 2011).

Percebe-se a evolução deste tipo de estrutura física desde a Antiguidade, onde já se buscava administrar os fatores climáticos que atingem o cultivo das plantas. A construção de estufas mais aprimoradas se deu somente no século XVI na Itália e, mais tarde, a engenhosidade se espalhou para os países da Inglaterra e Holanda. No Brasil, o uso de estufas começou a ser difundido recentemente, quando iniciativas de empresas privadas e de órgãos ligados a pesquisa do ramo agrícola implementaram experiências a partir de 1970 (PINHEIRO, 2011).

As estufas podem ser construídas com uma diversidade de materiais, dependendo simplesmente da necessidade do criador e do clima predominante na região. No caso do Nordeste brasileiro, Bezerra (2003) destaca que para a produção de hortaliças o ideal é "*uma estrutura coberta com plástico transparente (150 micra), laterais com telas e de preferência com lanternim na parte mais alta para facilitar a saída do ar quente*", considerando as altas temperaturas na maior parte do ano neste território (Ibid., p. 12).

A forma, tamanho e materiais também são aspectos condicionantes que influenciam a usabilidade, a durabilidade e o preço destes ambientes protegidos:

> As estruturas das estufas podem ser construídas em madeira ou metal. As estruturas metálicas têm sido preferidas por serem mais práticas e de manutenção mais fácil e possuem maior durabilidade, porém são mais caras. A altura da estufa deve ter um pé-direito acima de quatro metros e o comprimento até 50 metros, para evitar aumento da temperatura interna (BEZERRA, 2003, p. 12).

2.2 Fatores Climáticos na Produção de Hortaliças

A variação dos fatores climáticos é considerada um dos maiores desafios para a produção de hortaliças no território brasileiro, uma vez que estes influenciam diretamente a sua formação.

> O clima é fundamental para o desenvolvimento de plantas, os fatores climáticos como temperatura e luminosidade podem interferir de forma benéfica ou maléfica no desenvolvimento da planta, sendo assim, controlar esses fatores é de suma importância e o uso do ambiente protegido vem somar a essa busca por melhores resultados. (SANTOS; JUNIOR; NUNES, 2010).

No mecanismo adaptativo a ser construído neste trabalho, considera-se o controle dos seguintes fatores a serem desenvolvidos a seguir.

2.2.1 Temperatura

A variação da temperatura influencia o desenvolvimento das espécies vegetais por interferir em processos como transpiração, respiração, fotossíntese, germinação, crescimento, floração e frutificação (CERMEÑO, 2005). Cada planta necessita de uma faixa de temperatura específica para que seu ciclo de desenvolvimento seja otimizado. Faixas de temperatura adequadas permitem um crescimento efetivo das hortaliças, enquanto variações fora de seus limites geram perda em sua qualidade visual, sabor, podendo inclusive ocasionar sua morte.

A Tabela 1 foi construída a partir de dados obtidos pela Embrapa Hortaliças, designando faixas de temperaturas apropriadas para alguns tipos de hortaliças regularmente produzidas no Brasil.

Tabela 1 – Temperatura Ideal para Algumas Hortaliças

Hortaliça	**Temperatura Ideal**
Acelga	15 °C a 25 °C
Agrião	15 °C a 25 °C
Cebolinha	Até 25 °C
Couve-Flor	15 °C a 25 °C
Pimentão	15 °C a 25 °C
Repolho	15 °C a 25 °C

Fonte: Catálogo Brasileiro de Hortaliças, 2015, adaptado.

O controle de parâmetros como ventilação e umidificação pode servir de artifício para o controle da temperatura do ambiente da estufa. No caso da estufa proposta, será utilizado um sensor DHT22 para o monitoramento da temperatura interna, sendo seus valores utilizados como parâmetro para a ativação do sistema de ventilação e umidificação da estufa.

2.2.2 Umidade do Ar

A umidade relativa do ar, expressa em percentual (%), varia com a temperatura e representa a quantidade de água em forma de vapor presente na atmosfera em relação à quantidade máxima que poderia existir nas mesmas condições de temperatura (ponto de saturação) (CGE, 2011).

Um ambiente com a umidade do ar controlada prevê que as hortaliças não sofram com a falta dela ou o excesso, pois ambos os fatores são pertinentes para sua sobrevivência. Uma

hortaliça crescendo em um ambiente com baixa umidade do ar pode apresentar sinais de desidratação, como folhas secas ou mortas. Enquanto seu excesso pode acarretar o desenvolvimento de doenças (BRANDÃO FILHO; CALLEGARI, 1999).

A tabela abaixo indica a faixa de umidade do ar ideal para desenvolvimento de algumas hortaliças em ambientes protegidos:

Tabela 2 – Umidade do Ar Ideal para Algumas Hortaliças

Hortaliça	Umidade do Ar Recomendada
Tomate	50 a 70%
Melão	65 a 75%
Pepino	70 a 90%
Pimentão	50 a 70%

Fonte: BRANDÃO FILHO; CALLEGARI. *Cultivo de hortaliças de frutos em solo em ambiente protegido*, 1999, adaptado.

No trabalho, optou-se por fazer um controle da umidade do ar para que a mesma se comporte na faixa dos 60% a 80% a partir do uso do sensor DHT22 e de um umidificador de ar acionado por um relé, que será ligado quando o nível de umidade do ar na estufa for menor do que o esperado.

2.2.3 Ventilação

A ventilação pode ser considerada como a troca de ar entre o meio externo e interno da estufa. O principal objetivo da ventilação é retirar a sobrecarga de calor de dentro da estufa. A importância de se fazer circular e renovar o ar interno se deve ao fato de evitar a presença e surgimento de pragas, controlando a umidade do ar na estufa. Para tanto, existem dois tipos de ventilações, a natural e a mecânica. A primeira forma de ventilação é caracterizada por aberturas laterais ou superiores existentes na estufa, o que permite a renovação gasosa interna. Já a ventilação mecânica, conta com um ventilador que será utilizado para as trocas gasosas (SOLERPALAU, 2015).

No trabalho, a ventilação da estufa será feita da forma natural e mecânica, sendo possível remover a tampa da estufa para uma ventilação natural. Porém, será também instalado um ventilador a ser ligado por um relé, para que seja realizada a troca de gases da estufa com o ambiente.

2.2.4 Luminosidade

A luz é fundamental para a sobrevivência das plantas, pois através do fenômeno da fotossíntese, as plantas transformam moléculas de água e dióxido de carbono em energia química e liberam simultaneamente o oxigênio, renovando e purificando o ar. Estas moléculas são, então, utilizadas pelas plantas em sua respiração (HOPKINS, 2006, p.4-8).

A luminosidade, em termos da agricultura, caracteriza-se pela quantidade de luz incidente nas plantas. Sendo este um fator importante não apenas para sua sobrevivência, mas também para seu desenvolvimento, já que o excesso – ou falta – podem gerar efeitos adversos em seu crescimento (Ibid., p.6).

No trabalho, optou-se pela instalação de uma lâmpada na estufa, para que em casos de baixa luminosidade, seja possível oferecer ao menos a luz artificial para a planta, tendo em vista a importância da luz em seu crescimento. Esta lâmpada será ativada por meio de um relé que será controlado pela plataforma Arduino, em conjunto com um sensor de luminosidade instalado na estufa.

2.2.5 Irrigação

A vida das plantas depende diretamente da água, elemento fundamental para o processo de fotossíntese e, consequentemente, para seu desenvolvimento.

Na produção de hortaliças, um sistema de irrigação pode funcionar de forma manual ou automática. O sistema manual exige um esforço contínuo por parte do usuário para seu bom funcionamento, enquanto que a automatização do processo de irrigação proporciona um maior controle e eficiência da quantidade de água a ser utilizada no cultivo.

Diferentes tipos de irrigação podem ser empreendidos em um ambiente protegido, tendo cada um deles vantagens e desvantagens. A irrigação por subirrigação inunda as bancadas ou o próprio piso em que os recipientes das plantas se encontram, mantendo as folhagens secas e podendo contribuir para uma menor incidência de doenças. Todavia, este método pode gerar um custo elevado. Outra forma de irrigação é por aspersão, no qual a água é liberada em forma de círculo ou semicírculo em torno da plantação, porém sua utilização acarreta desperdício de água elevado. A irrigação por nebulização é uma alternativa que fornece água em forma de névoa, proporcionando menor danos às plantas e melhor distribuição da água (BEZERRA, 2003).

Por outro lado, opta-se pela irrigação hidropônica na confecção deste projeto, que é uma forma de irrigação na qual as plantas crescem com suas raízes em um canal ou canaleta pelo qual atravessa, periodicamente, uma solução nutritiva composta por água e nutrientes, como mostra a figura 2.1. Esta solução oferece diversas vantagens na produção de hortaliças, quando comparada ao cultivo no solo, sendo elas: menor volume de água necessário, área para produção reduzida em cerca de dez vezes e um baixo uso de produtos defensivos agrícolas – como agrotóxicos – no cultivo. Além disso, as hortaliças possuem maior durabilidade, qualidade e produtividade (ABRANTES; SEIXAS FILHO, 2006).

No trabalho, foi utilizado o sistema sistema de técnica de cultura de fluxo laminar de nutrientes (NFT - *Nutrient Film Technique*) para a alimentação das plantas. Este sistema funciona de forma em que a solução nutritiva fica armazenada em um reservatório, sendo bombeada assim para a parte superior do canal ou canaleta de cultivo e retornando para o reservatório.

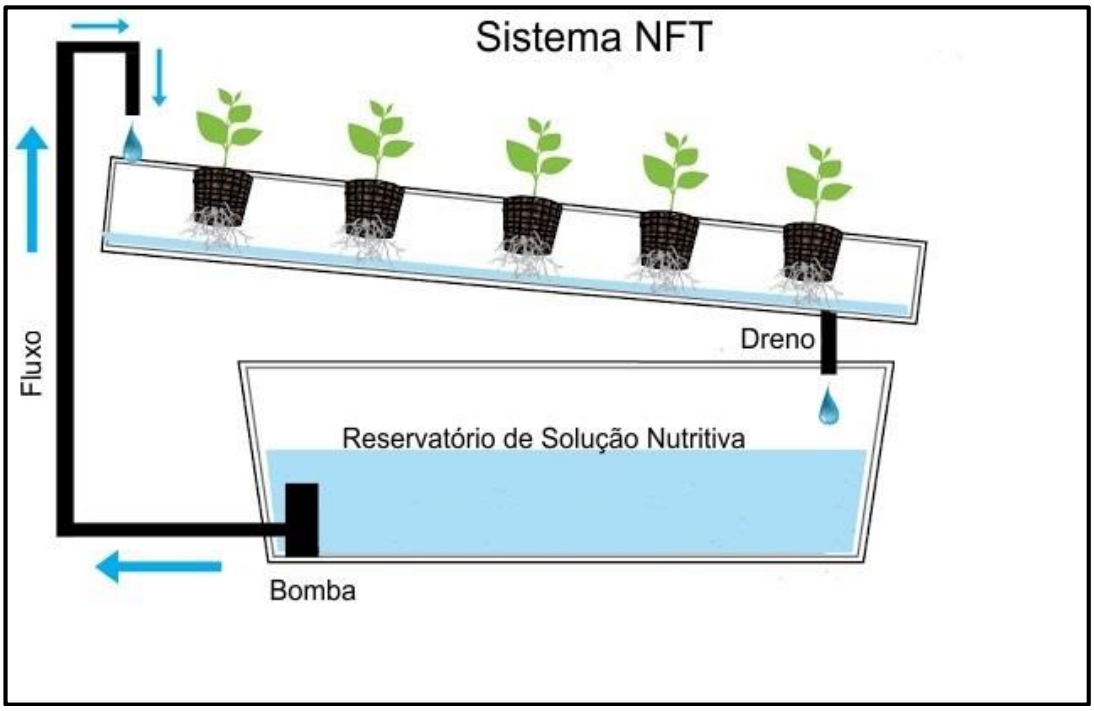


Figura 2.1 - Sistema NFT
Fonte: Adaptado de http://tudohidroponia.net/, 2015.

2.3 Plataforma Arduino

A plataforma Arduino foi criada na Itália, no Instituto de Design e Interação Ivera (*Interaction Design Institute Ivera*), escola voltada para a pesquisa na interação entre dispositivos digitais e sistemas (YARNOLD, 2015).

É uma placa de circuito montada com base nos microcontroladores ATmega8 ou ATmega168. Seu design atual oferece uma entrada USB para a configuração da memória flash da placa, além de uma série de pinos analógicos e digitais de entrada e saída, que são utilizados na criação de projetos (Ibid., p.3). O modelo denominado Arduino UNO, permite uma redução considerável nos custos de produção de um projeto, mantendo a eficiência do sistema.

Funcionando como um computador em miniatura, a plataforma Arduino é capaz de processar informações entre o próprio dispositivo e os componentes externos a ele conectados. Por meio da programação em seu ambiente de programação integrado (IDE - *Integrated Development Environment*), um *software* livre do próprio Arduino que é utilizado para criação de códigos baseados na linguagem C, é possível designar programas de computadores a serem instalados em sua memória *flash*, fazendo com que o Arduino realize o conjunto de instruções que foram delineadas no *software* (MCROBERTS, 2011).

Segundo McRoberts (2011), uma das maiores vantagens da plataforma Arduino é sua arquitetura de *hardware* e *software* abertas, possibilitando a utilização livre por qualquer pessoa ou propósito, de seus códigos, esquemas ou projetos (Ibid., p.24).

2.3.1 Arduino UNO

Arduino UNO é um dos vários tipos de Arduinos existentes na atualidade. Suas características consistem em:

> O Arduino UNO é uma placa baseada no microcontrolador ATmega168P, ele possui 14 pinos digitais de entrada e saída, 6 entradas analógicas, um cristal de quartzo de 16 MHz, conexão USB, um conector de energia, um botão ICSP (*In Circuit Serial Programming*) e um botão de reset. Ele contém tudo para suportar o microcontrolador; Simplesmente conecta-se a um computador por um cabo USB ou liga-se pelo seu conector de corrente alternada para corrente contínua (AC/DC - *Alternating Current to Direct Current*) de 9v (ARDUINO.CC, 2015, tradução nossa).

As especificações técnicas deste tipo de placa contemplam os seguintes dados apresentados na tabela 3:

Tabela 3 – Especificação Arduino UNO

Características Técnicas	Valores
Microcontrolador	ATmega328P
Tensão de operação	5V
Tensão de entrada (recomendada)	7-12V
Tensão de entrada (limite)	6-20V
Pinos digitais de entrada e saída (I/O - *input/output*)	14 (Dos quais 6 dão suporte a saída PWM)
Pinos digitais I/O PWM	6
Pinos de entrada analógica	6
Corrente contínua por pino I/O	20 mA
Corrente contínua por pino 3.3v	50 mA
Memória *Flash*	32KB (ATmega328P) dos quais 0.5 KB são utilizados pelo bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Velocidade de *Clock*	16 Mhz
Dimensões	68.6 x 53.4 mm
Peso	25 g

Fonte: Adaptado de www.arduino.cc, 2015.

A figura 2.2 mostra o Arduino *UNO*, utilizado no projeto:

Figura 2.2 - Arduino *UNO*
Fonte: www.arduino.cc, 2015.

O microcontrolador ATmega328P possui um conversor A/D (analógico/digital) de 10 bits de resolução para os pinos de entrada analógica, ou seja, é criado uma representação digital de até 1024 valores distintos para o sinal analógico de entrada (ARDUINO.CC). O Arduino *UNO* possui um regulador de tensão de 5V, o que permite uma alimentação variável de 7 a 12 Volts, sendo esta convertida a uma tensão constante de 5V. Será utilizado para alimentação do projeto uma fonte 220v, com uma porta USB de 5v para que seja alimentado o Arduino. Dos 14 pinos digitais de entrada/saída do Arduino, cinco são reservados para o *shield* de Ethernet (D13, D12, D11, D10 e D4), cinco (D9, D8, D7, D6 e D5) serão utilizados para o display e quatro (A0, A3, D2 e D3) serão para o *shield* de relés. Dos analógicos, 2 (A1 e A2) serão utilizados para os sensores e 1 (A4) será utilizado para o display.

Para este projeto, o *UNO* será responsável pelo recolhimento de informações dos sensores de luminosidade, umidade do ar e temperatura. Ao captar tais informações, serão enviados comandos para que sejam ativados os subsistemas conectados ao *shield* de relés, o umidificador, a ventoinha e a lâmpada. O sistema de irrigação funcionará sem nenhuma interferência dos sensores em um período pré-configurado.

A figura 2.3 apresenta o esquema elétrico do Arduino *UNO* na versão utilizada no projeto, em detalhes:

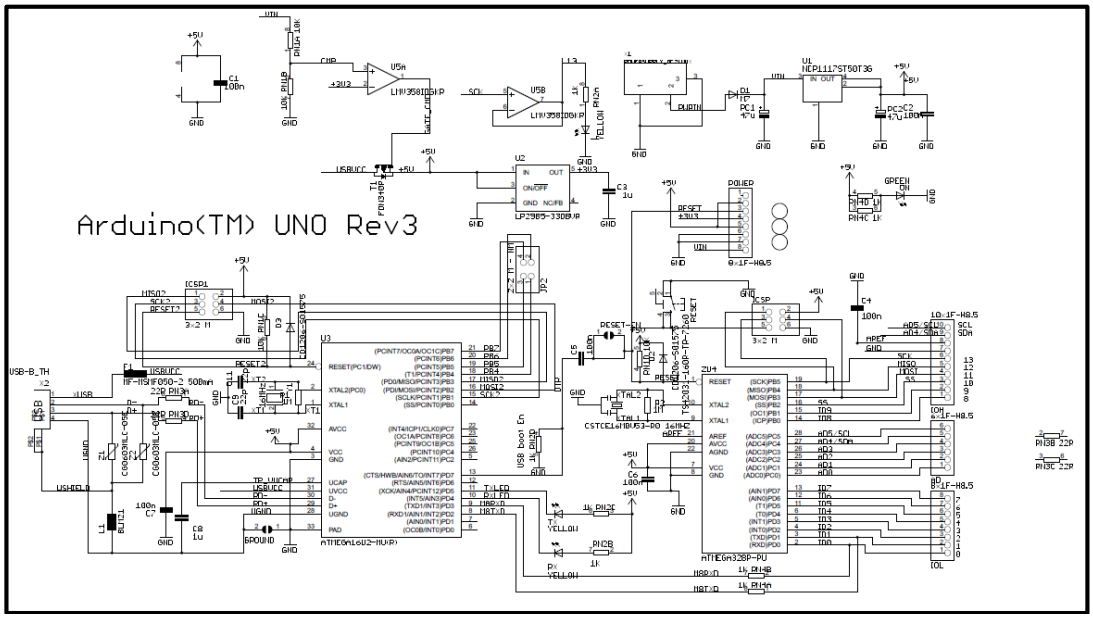


Figura 2.3 - Esquema elétrico do Arduino UNO Rev3
Fonte: https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf

O esquema elétrico mostrado na figura 2.3 também pode ser encontrado, ampliado, no Anexo A.

Os principais componentes do Arduino *UNO* são:

- 2 processadores: ATMega16U2 (U3) que é o processador USB e o ATMega328P (ZU4), que funciona como o processador principal. A razão disto é pelo fato do ATMega328P não suportar uma conexão direta com a porta USB. Desta forma, o ATMega16U2 converte os dados da USB para um sinal serial (UART), podendo este ser lido pelo processador principal;
- Conector USB (X2): é onde se conecta o cabo USB. Pode ser usado para transferência de dados e/ou de alimentação;
- Filtro e proteção USB (F1, Z1, Z2, L1, RN3A e RN3D): estes componentes, em conjunto, criam o chamado “filtro e proteção USB”. Os componentes Z1 e Z2 são varistores e têm como função proteger os pinos do ATmega16U2 (processador USB) contra descargas eletrostáticas. O componente F1 é um fusível, que tem como função proteger a porta USB do computador (PC - *Personal Computer*) caso ocorra um curto-circuito ou sobrecarga acidental na placa do Arduino. O componente L1 é um ferrite utilizado para a redução de possíveis ruídos que possam vir pela malha do cabo USB até o Arduino, bem como isolar o PC de ruídos gerados pelo Arduino. Já os componentes RN3A e RN3D formam uma rede de resistores, utilizados para

atenuar ruídos e picos de tensão que podem vir a ocorrer através do cabo USB, ajudando assim na proteção do processador;

- 2 Osciladores: o oscilador é responsável pela geração do pulso de *clock*. No Arduino *UNO*, o oscilador do ATMega16U2 foi implementado utilizando um cristal de 16MHz (Y1) como componente principal. Enquanto que o oscilador do ATMega328P foi feito com um ressonador cerâmico (Y2) de 16MHz;
- 1 Regulador linear de 5V: denominado U1 no circuito, este regulador é responsável por baixar a tensão da fonte e estabilizá-la em 5V, dissipando o resto em forma de calor;
- 14 pinos digitais de entrada ou saída (programáveis);
- 6 pinos de entrada analógica ou entrada/saída digital (programáveis);
- 1 Botão de reset: nomeado como “RESET” no esquemático, este botão fecha o contato dos pinos 1 e 2, fazendo com que ocorra um reset do processador.

2.3.2 IDE do Arduino

A plataforma Arduino utiliza um *software* próprio para o desenvolvimento de programas, conhecidos como s*ketch*. Os programas desenvolvidos no IDE do Arduino, quando compilados, podem ser enviados à placa para a execução do mesmo. Este ambiente, ainda, permite que os *softwares* criados sejam testados antes de serem enviados às placas (MCROBERTS, 2011, p.24).

A figura 2.4 mostra a interface IDE da plataforma Arduino:

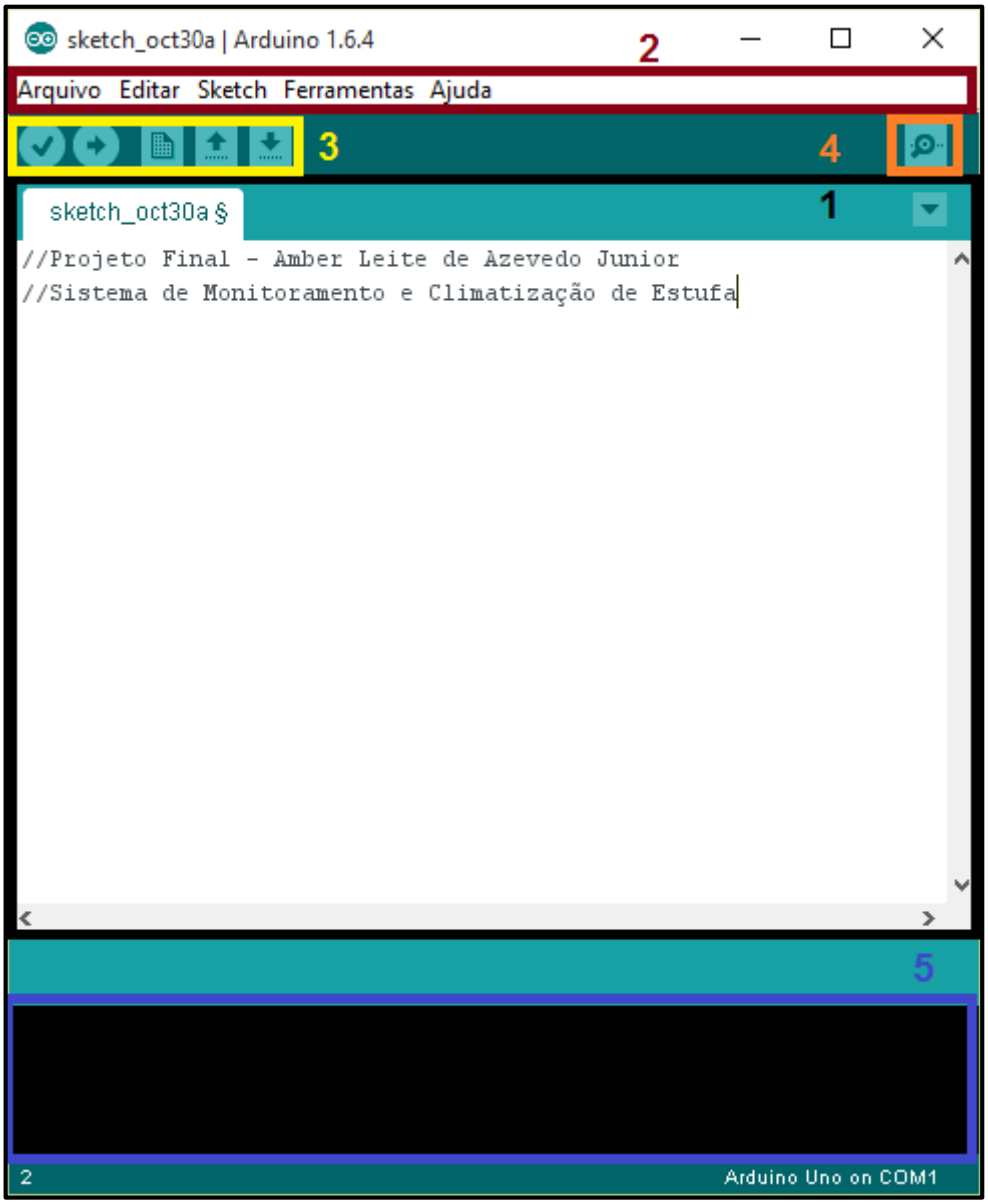


Figura 2.4 - IDE Arduino
Fonte: Foto elaborada pelo autor

A IDE do Arduino apresenta as seguintes funções, destacadas e numeradas na figura 4:

1. Área de programação: onde são escritos os códigos dos *softwares* criados para a plataforma Arduino;
2. Menu principal;
3. Botões de acesso rápido: indica atalhos para funções frequentemente usadas durante a programação de um programa para o Arduino. Da esquerda para direita é possível verificar a integridade do código, carregar o código no Arduino, abrir novo *sketch*, abrir um *sketch* salvo e, por fim, salvar o sketch apresentado na área de programação.

4. Exibir monitor serial: exibe os dados enviados e recebidos pelo Arduino em uma nova janela;
5. Barra de estado: descreve informações acerca do código no caso de sucesso ou falha das configurações.

A seguir, veremos os componentes externos que foram empregados no sistema, considerando as variáveis a serem exploradas para o controle e monitoramento da estufa desenvolvida neste projeto.

2.3.3 Bomba para Irrigação

O sistema NFT proposto necessita que, periodicamente, a solução nutritiva seja transportada pelas raízes das plantas. Para tal, será utilizada uma bomba submersa de água de 220l/h de vazão, que opera na finalidade de movimentar a água para que ocorra a troca de oxigênio entre ela e a atmosfera, renovando assim oxigênio da solução nutritiva utilizado no sistema NFT. A figura 2.5 abaixo mostra a bomba de água que será instalada no projeto:

Figura 2.5 - Bomba para Irrigação
Fonte: Foto elaborada pelo Autor

Segundo Braga (2009), um funcionamento adequado de um sistema hidropônico NFT envolve a solução nutritiva ser bombeada aos canais e escoar graças a gravidade, formando uma fina lâmina de solução de irriga as raízes.

2.3.4 Sensor de Umidade do Ar e Temperatura – DHT22

Para realizar as medições de umidade do ar e da temperatura na estufa proposta, será utilizado o sensor DHT22. Ele funciona com um sensor capacitivo de umidade e um thermistor

para medir o ar em volta dele, entregando suas leituras em forma de sinal digital para o pino de dados do Arduino.

Este sensor foi escolhido devido a sua facilidade de utilização e sua precisão nas medidas, sendo seu maior problema o fato de que seus dados só podem ser lidos a cada 2 segundos, problema este que foi contornado durante a criação do código.

As características dele são:

- Tensão de funcionamento de 3 a 5V e I/O;
- 2.5mA corrente máxima durante a conversão (pedindo dados);
- Bom para leituras de 0-100% de umidade com variação de 2~5%;
- Opera com faixas de leitura de temperatura de -40 a 80 ºC, com uma variação de ±0,5%;
- Limite de 0,5Hz de leitura (uma vez a cada 2 segundos);
- Tamanho: 27mm x 59mm x 13.5mm;
- 4 pinos com 0,1” de espaço.

Na figura 2.6 abaixo, o sensor DHT22 é mostrado:

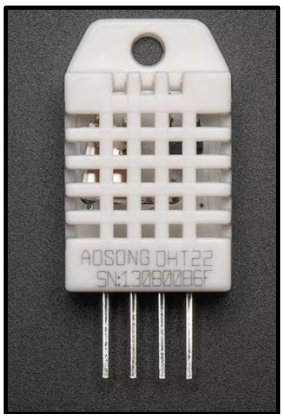


Figura 2.6 - Sensor DHT22
Fonte: Foto elaborada pelo autor.

Considerando a pinagem vista na figura acima, as configurações do sensor são feitas da seguinte forma:

- 1º pino é o VCC, que deve ser ligado no 5v do Arduino;
- 2º pino é o pino de dados, que deve ser ligado em uma porta digital do arduino;
- 3º pino é nulo, não precisando ser ligado;
- 4º pino é o GND do sensor, devendo ser ligado no GND do Arduino.

2.3.5 Sensor de Luminosidade – LDR

Para efetuar as medições de luminosidade da estufa, utilizou-se um sensor de luminosidade chamado de LDR (*Light Dependant Resistor*), que é um resistor cuja resistência varia de acordo com a incidência de luz. Em ambientes escuros, sua resistência é elevada. Porém, conforme a luz incide sobre ele, sua resistência diminui (MCROBERTS, 2014, p.114-115).

Na tabela 4 é utilizado um resistor de 10 kΩ (R1) como divisor de potencial, sendo possível assim constatar as variações nas tensões de saída em função da resistência do LDR, indicando a saída desde o mais escuro ao mais claro. :

Tabela 4 – Valores de Voult (tensão de saída) para um LDR com 5V como Vin (tensão de entrada)

R1	R2 (LDR)	Vout	Claridade
10 kΩ	100 kΩ	4,54 V	Mais escuro
10 kΩ	73 kΩ	4,39 V	25%
10 kΩ	45 kΩ	4,09 V	50%
10 kΩ	28 kΩ	3,68 V	70%
10 kΩ	10 kΩ	2,5 V	Mais claro

Fonte: McRoberts, 2014, p.115.

O sensor LDR, apesar de não ser um bom medidor de quantidade de luz no ambiente (lúmens), pode ser utilizado de maneira satisfatória para indicar a presença ou ausência de luz. No trabalho, ele permitirá que a leitura da luminosidade do ambiente em que a estufa está posicionada, sendo possível ligar uma lâmpada que ofereça luz artificial na ausência de luz natural.

A figura 2.7 mostra o LDR utilizado no trabalho:

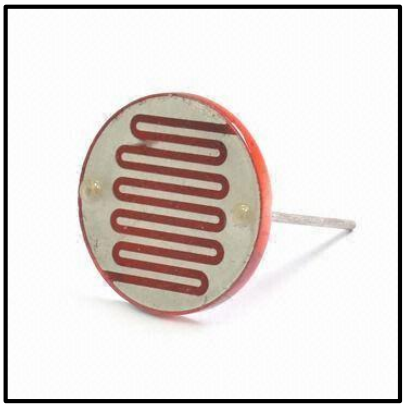


Figura 2.7 – Sensor de Luminosidade (LDR)
Fonte: Foto elaborada pelo autor

2.3.6 Shield de Relés

Relé é uma chave eletrônica que liga ou desliga quando uma pequena tensão é aplicada a seus terminais de controle. Os relés utilizados no trabalho, por exemplo, são ativados ao receber uma carga de 5V, permitindo que sejam ligados os dispositivos utilizados no controle do microclima da estufa. Relés eletrônicos foram utilizados por possuírem uma rápida velocidade de comutação e, ao receber o sinal do Arduino, são ativados imediatamente, possibilitando que a carga de 220V necessária para o funcionamento dos aparelhos utilizados no projeto seja provida.

A figura 2.8 mostra o *shield* de relés que foi utilizado no trabalho:

Figura 2.8 - *Shield* de Relés
Fonte: Foto elaborada pelo autor

A estrutura básica de cada um dos relés contidos no *shield* utilizado no trabalho consiste em três pontos de contato, sempre na mesma ordem, denominados de NO, Com e NC, que funcionam da seguinte forma:

- Normalmente Aberto (NO - *Normally Open*): É o primeiro ponto de contato do relé mostrado. Neste ponto é ligado o fio de força que vai até o equipamento que se deseja ligar. Quando o relé é ativado, é feita a conexão do ponto COM ao NO, fechando o circuito;
- Conexão Comum (Com - *Common Connection*): O ponto do meio recebe a carga da tomada (220V) e funciona como uma chave do relé que alterna entre NC e NO, de acordo com o estado do mesmo. Ao aplicar uma carga de 5V no relé, a conexão interna é trocada do ponto NC para o ponto NO;
- Normalmente Fechado (NC - *Normally Closed*): O último ponto opera sempre em contato com o ponto COM, mesmo com o relé desligado, proporcionando um circuito aberto até que o relé seja ativado.

No exemplo da figura 2.9, é possível observar as conexões a serem feitas entre a plataforma Arduino, um relé e um periférico qualquer (no caso do exemplo abaixo, uma lâmpada):

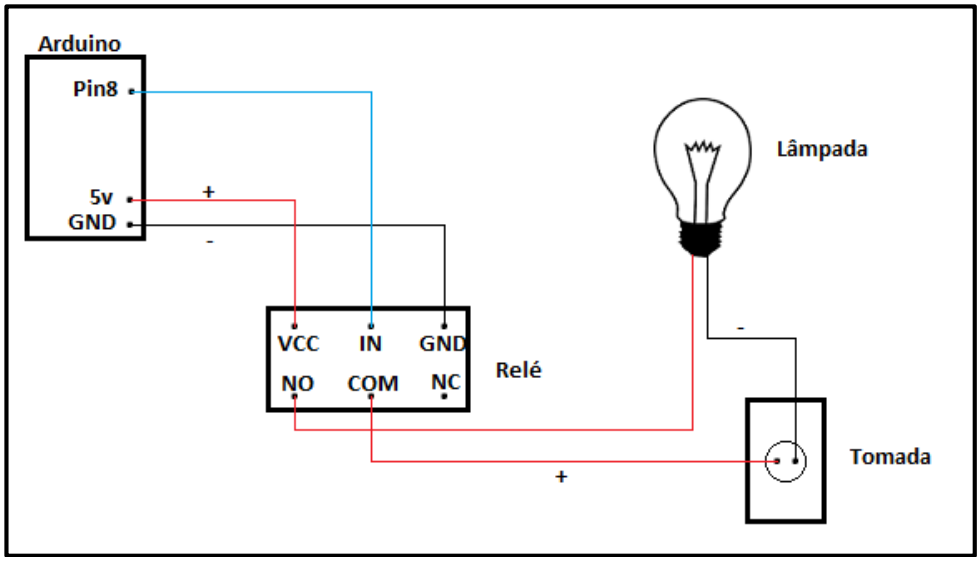


Figura 2.9 – Conexão relé com Arduino e uma lâmpada
Fonte: Foto elaborada pelo autor

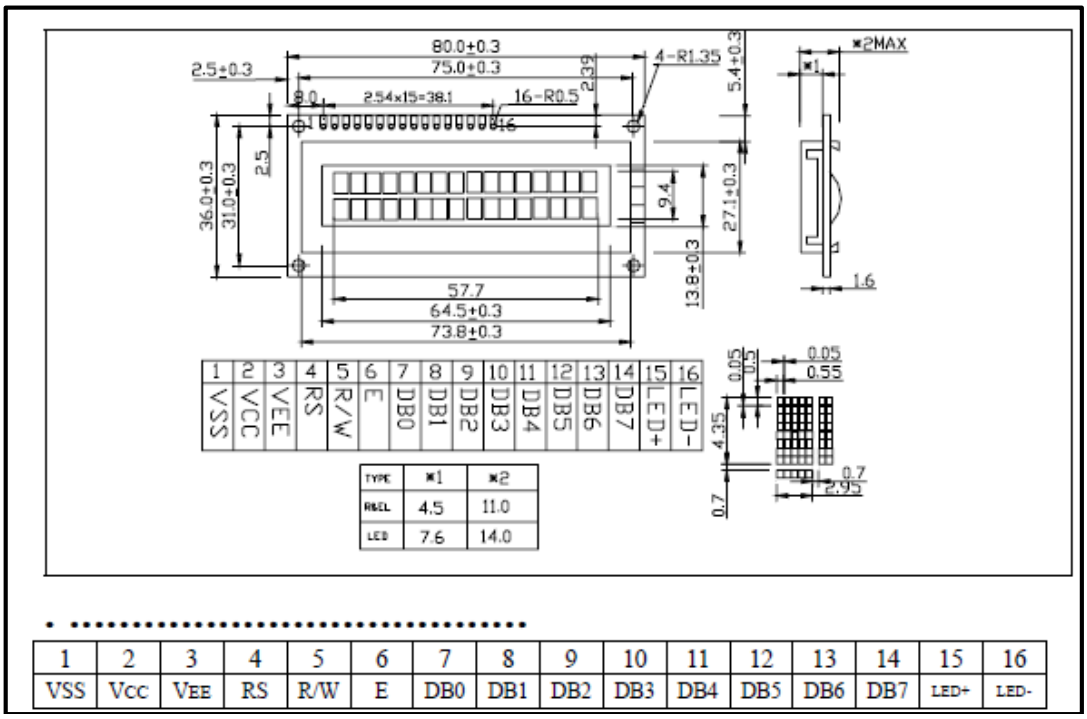
Seu funcionamento ocorre da seguinte forma: ao receber um sinal do pino 8, do Arduino, o relé troca a conexão interna do pino NC para o pino NO, ativando o circuito e permitindo, assim, que a carga da tomada chegue à lâmpada, acendendo-a. Esta operação será utilizada no

projeto para a ativação dos dispositivos utilizados na estufa, sendo eles o cooler, a lâmpada, o umidificador e a bomba de irrigação.

2.3.7 Visor LCD

O uso de um visor LCD 16x2 (16 colunas por 2 linhas) modelo JHD162A permite a visualização de informações dos sensores em tempo real. Será utilizado para tal, um visor LCD com tela azul e letra branca. A facilidade de sua configuração devido a compatibilidade com a biblioteca *LiquidCrystal* do Arduino foi um dos principais fatores de sua escolha.

A figura 2.10 é a representação técnica do display utilizado, com suas características principais e pinagens descritas a seguir:



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
VSS	Vcc	VEE	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LED+	LED-

Figura 2.10 – Características display JHD162A
Fonte: Huinfinito.com.br, 2015.

Sua pinagem é feita da seguinte forma:

- VSS – GND do visor;
- VCC – 5v do visor;
- VEE – Ajuste de contraste do visor;
- RS – Seletor de Registro (*Register Select*);
- R/W – Seletor de modo de leitura ou escrita (*Read/Write*);
- E – Permite que os registros sejam gravados (*Enable*);

- DB0 a DB7 – São os pinos de dados, os valores destes pinos (*high* ou *low*) determinam o que está sendo escrito/visto;
- LED+ e LED- – Controle da iluminação traseira do *display*;

No projeto, os 16 pinos mostrados na figura 10 são conectados da seguinte forma:

- Os pinos 1, 5 e 16 são ligados ao GND do circuito;
- O pino 2 é ligado no +5v do circuito, proporcionando a alimentação do visor;
- O pino 3 é utilizado para o ajuste de contraste. No projeto foram utilizados dois resistores, um de 10K (conectado ao +5v) e um de 100 Omhs (conectado ao GND) para tal;
- Os pinos 4, 6, 11, 12, 13 e 14 são os pinos conectados às portas digitais do Arduino e permitem que o visor opere no modo 4-bits. Os pinos 7, 8, 9 e 10 seriam utilizados caso fosse necessária uma configuração 8-bits no display;
- Os pinos 15 e 16 são, respectivamente, os pinos de +5v e GND da luz traseira do display. Foi instalado um resistor de 470 Omhs no pino 15 a fim de limitar a corrente.

A figura 2.11 mostra as conexões descritas:

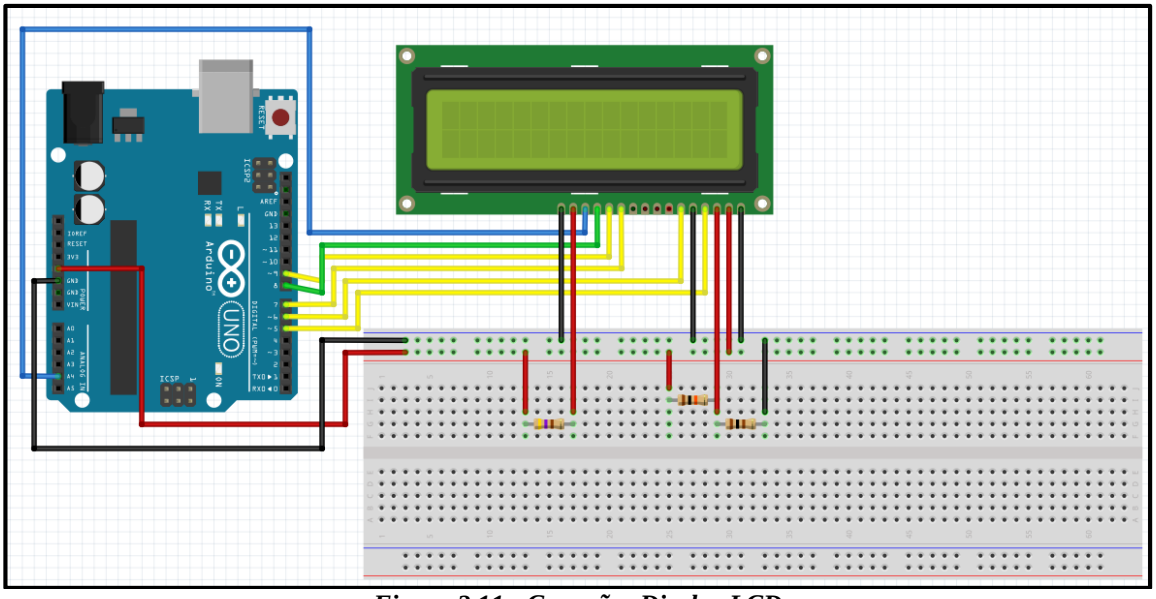


Figura 2.11 - Conexões Display LCD
Fonte: Foto elaborada pelo autor.

2.3.8 Ethernet Shield

O *Ethernet Shield* permite que o Arduino possua conexão a Internet por meio de um cabo de rede conectado entre o *shield* e um roteador. Esta placa tem a vantagem de possuir o formato empilhável que alguns *shields* apresentam, permitindo que ela seja montada em cima de alguns modelos de Arduinos, como o *UNO*, sem a necessidade de solda em suas conexões, conforme a figura 2.12 mostra:

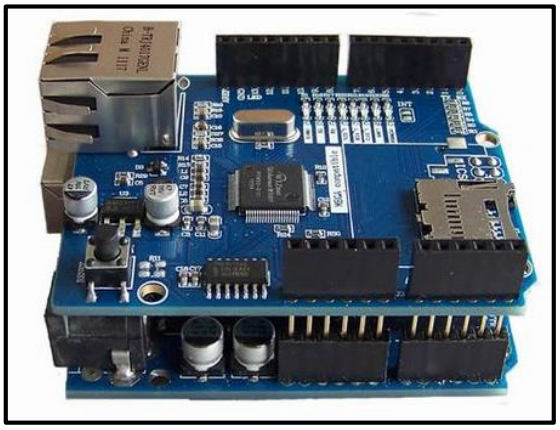


Figura 2.12 – Arduino* UNO *montado com* Ethernet *Shield
Fonte: Foto elaborada pelo autor.

O *shield* possui um controlador W5100 que permite a conexão da plataforma Arduino com a Internet e/ou dispositivos instalados na rede local. Este *shield* vem instalado com um soquete para leitura de cartão MicroSD, que pode ser utilizado no armazenamento de dados. A comunicação com a plataforma Arduino é feita por meio da interface SPI (*Serial Pheriferal Interface*), protocolo utilizado para a comunicação entre microcontroladores.

Seguem as descrições detalhadas do Ethernet *Shield* utilizado no trabalho:

- Conector RJ45;
- Soquete para cartão MicroSD;
- Controlador Ethernet W5100 com buffer interno de 16 k;
- Velocidade de conexão de 10/100 Mb;
- Pinos Reservados: 10 ao 13 para *SPI*;
- Tensão de operação de 3,3 – 5 V;
- Suporta até 8 conexões TCP/UDP simultâneas;
- Compatível com a biblioteca Ethernet do Arduino;
- Compatível com Arduino Uno (utilizado no projeto).

No projeto, o *Ethernet Shield* foi utilizado para enviar as informações provenientes dos sensores para um website (*ThingSpeak*), dispondo-as para o usuário em tempo real, o que possibilita a consulta do estado da estufa continuamente.

2.3.9 ThingSpeak

A plataforma *ThingSpeak* é um serviço aberto que oferece uma infraestrutura *web* e protocolo baseado em *http* para o envio e recebimento de dados gerados pela plataforma Arduino ou qualquer outro dispositivo com recursos para comunicação em rede. Os dados são armazenados no *website* e podem ser acessados publicamente ou privadamente, baseados na configuração do usuário.

Este serviço permite a criação de um canal de dados que contém oito campos capazes de comportarem qualquer tipo de dados, além de três campos para dados de localização e um para dados de *status*.

No projeto, a plataforma ThingSpeak será utilizada para criação de um canal chamado "Projeto Estufa" que receberá, armazenará e exibirá os dados gerados pela estufa em três campos: umidade do ar, temperatura e luminosidade. Esta informação será enviada por meio da conexão com a Internet do Arduino e disposta em forma de gráficos, conforme demonstra a figura 2.13:

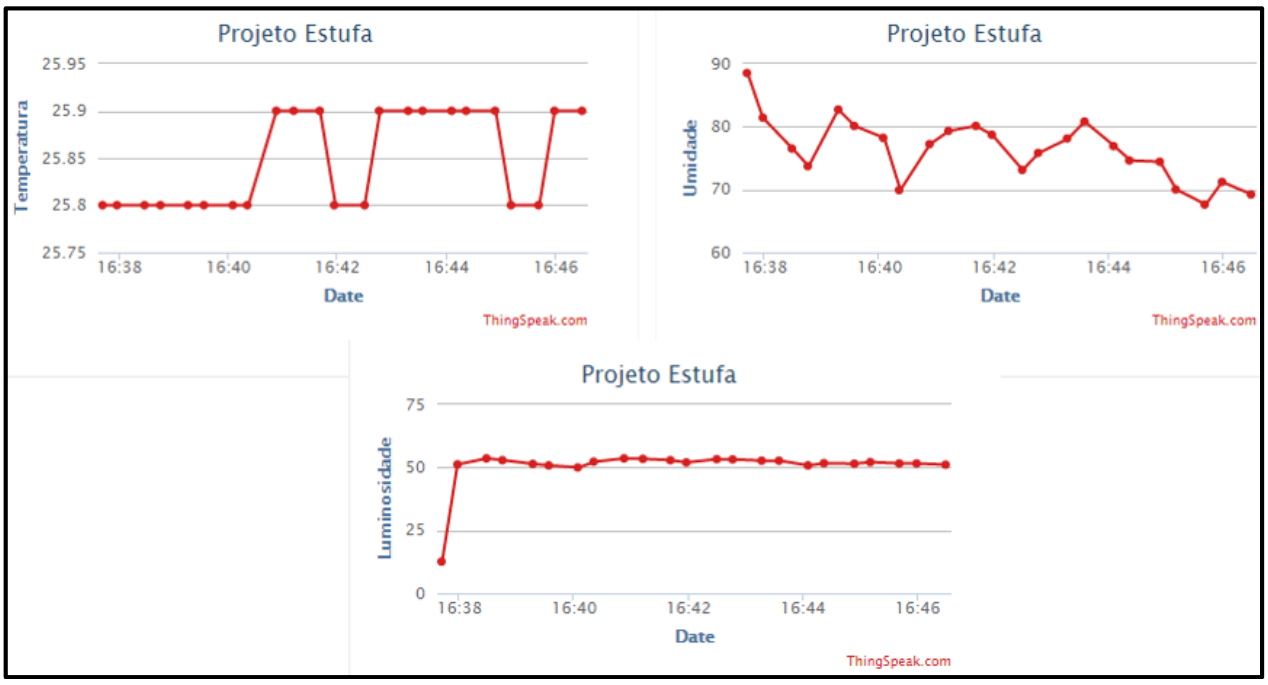


Figura 2.13 - Gráficos apresentados no ThingSpeak
Fonte: Foto elaborada pelo autor.

Na figura 2.13, percebe-se que a divisão das telas sobre os fatores que influenciam o crescimento das plantas permite organizar as informações, facilitando sua leitura, além de monitorar o funcionamento adequado da estufa, mesmo a distância.

2.4 Sistemas de Controle

Segundo DiStefano et al (1990), um sistema de controle é um arranjo de componentes físicos conectados ou relacionados de maneira a que comandem, direcionem ou regulem a si mesmos ou a algum outro sistema. Tendo em vista isso, pode-se afirmar que um sistema de controle pode ser designado a controlar o funcionamento de um processo à fim de gerar um resultado esperado.

Ogata (2003) define sistemas de controle em dois modelos: malha fechada e malha aberta. No controle de malha aberta, a saída não interfere na ação do sistema de controle. Sendo assim, o sinal de saída não é medido nem realimentado para comparação com o valor de entrada. Como este valor não é utilizado, cada entrada de referência corresponde a uma condição fixa de operação a ser executada, independente de quaisquer outra variável inerente ao processo. Um exemplo de um sistema de malha aberta é o que aciona um aparelho de microondas. Ao ser ligado, o microondas funcionará o tempo designado e seu sinal de saída não será medido, isto é, se esquentou o alimento de maneira satisfatória ou não. A figura 2.14 mostra um diagrama de blocos do funcionamento de um sistema de controle de malha aberta. Assim, é fundamental que em sistemas deste tipo determine-se com precisão os parâmetros da programação (processo) e condições do objeto ou ambiente de controle para que chegue até ele o sinal de saída adequado.

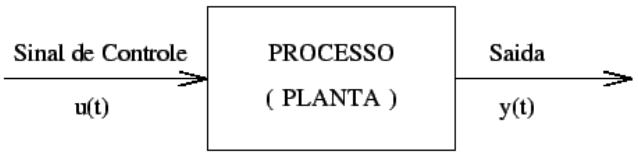


Figura 2.14 - Sistema de Malha Aberta
Fonte: Elaborado pelo Autor

Já em um sistema de malha fechada, ocorre a medição do valor de saída e este mesmo oferece um efeito direto no controle do sistema, comparando o valor obtido com o valor

esperado, e executando uma ação com base neste resultado. Um exemplo de funcionamento de um controle de malha fechada é um sistema de controle de temperatura do ambiente, como ocorre em estufas. É designado um valor de saída esperado e o sistema efetua comparações ao resultado obtido, em caso de divergência, pode-se executar uma ação de ligar o sistema de resfriamento ou o sistema de aquecimento do ambiente. O sinal gerado através da proporção de erro entre a resposta desejada (valor de referência r(t)) e a resposta real (valor de saída y(t)) resulta em uma sequência de operações em malha fechada, chamada de realimentação (Erro: e(t)). A equação (eq2.1) o define:

$$e(t) = r(t) - y(t) \qquad \text{(eq2.1)},$$

A partir do valor do erro, é possível efetuar ajustes na malha de realimentação buscando ajustar-se o erro, de modo a controlar o processo na busca da saída desejada. Na figura 2.15, é mostrado por meio de diagrama de blocos a representação de um sistema de controle de malha fechada

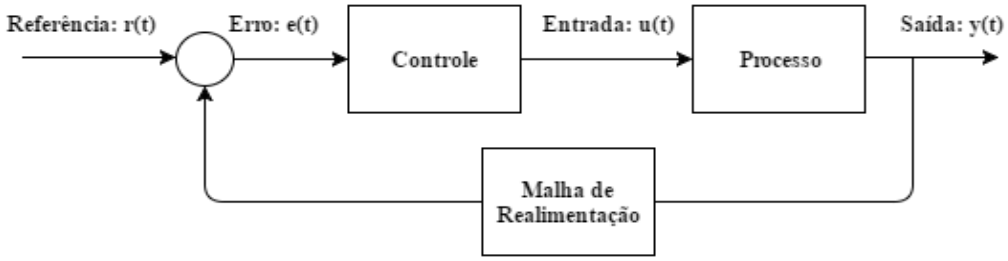


Figura 2.15 - Sistema de Malha Fechada
Fonte: Elaborado pelo autor

Neste projeto, serão utilizados ambos os sistemas de controle de malha, aberta e fechada. Como sistema de malha aberta, será utilizado o sistema de irrigação hidropônico, que funcionará com base na sua variável de tempo. Já como malha fechada, serão utilizados sistemas de controle da temperatura ambiente e da umidade do ar na estufa.

2.5 Visita Técnica

Para melhor entendimento das necessidades e atribuições no cultivo de hortaliças, foi feita uma visita técnica à Embrapa Hortaliças – DF, localizada no Gama – DF. O foco principal desta visita foi entender o funcionamento de uma estufa para o cultivo de hortaliças, bem como a importância do cultivo protegido e decisão do sistema de irrigação a ser utilizado.

O pesquisador em solos e nutrição de plantas, Juscimar Silva, e o pesquisador de irrigação e drenagem, Marcos Braga, apresentaram o funcionamento do sistema de irrigação

hidropônico e suas vantagens perante um sistema de irrigação do solo. Os fatores apresentados a seguir foram considerados para a escolha de utilização deste sistema:

- Limpeza – um sistema de irrigação de solo em ambientes fechados apresenta dificuldade de manejo devido a sua composição. A necessidade de adubação, por exemplo, pode acarretar mau cheiro. Por outro lado, o sistema hidropônico permite utilizar apenas a água e os nutrientes necessários para o desenvolvimento das plantas diretamente em seus caules, proporcionando a organização e a limpeza mais eficaz do espaço;
- Facilidade de manutenção – a vantagem de um sistema hidropônico é que ele precisa ter sua solução trocada somente de tempos em tempos, já que é possível utilizar uma mesma solução até que os nutrientes da mesma sejam absorvidos pela planta. Já um sistema de irrigação do solo precisa de um estudo aprofundado de seus componentes, de forma a avaliar sua composição e permitir observação constante de suas condições após o plantio, evitando a contaminação do solo.

Com as informações obtidas durante a visita técnica, foi então definido o sistema de irrigação proposto, por sugestão do Pesquisador de Irrigação e Drenagem Marcos Braga, que é a utilização de canos PVC e uma bomba de aquário para a montagem de um sistema hidropônico simples, similar ao sistema apresentado na figura 2.14, com adaptações para ser utilizado no projeto:

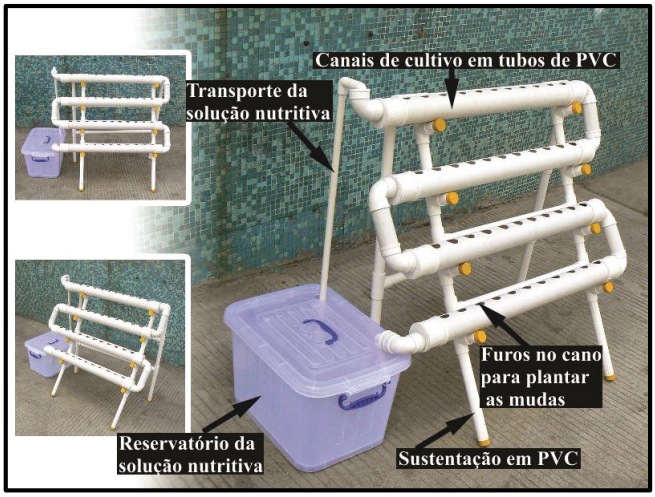


Figura 2.16 - Sistema Hidropônico Sugerido
Fonte: tudohidroponia.net, 2015.

Os aspectos técnicos e metodologia abordados neste capítulo são de fundamental importância para o desenvolvimento do projeto, conforme será visto nos próximos capítulos.

3 DESENVOLVIMENTO DO SISTEMA DE MONITORAMENTO E CLIMATIZAÇÃO DE ESTUFA DE PEQUENO PORTE EM UM CONTEXTO DOMÉSTICO

Neste capítulo será apresentado o desenvolvimento da solução proposta, bem como os detalhes de sua confecção e apresentação do que foi feito com base na aplicação dos conceitos teóricos adquiridos e apresentados no capítulo 2.

3.1 Apresentação Geral do Projeto Proposto

A proposta deste projeto é desenvolver um sistema de controle e monitoramento de uma estufa de pequeno porte, que possa ser utilizada para o cultivo de hortaliças em ambientes pequenos.

Com base no que foi apresentado no Capítulo 2, considerou-se que o cultivo de hortaliças envolve uma série de cuidados devido a natureza das plantas. Estes cuidados espelham-se principalmente em atributos fundamentais para o desenvolvimento saudável de plantas, como um clima favorável, uma irrigação adequada e luminosidade, que serão considerados na elaboração do sistema. Para tal, serão utilizados sensores de umidade do ar e temperatura (DHT22) e um sensor de luminosidade (LDR). Os modelos apresentados nas figuras 3.1 e 3.2 indicam o diagrama de controle que representam a lógica de funcionamento do sistema com uso dos sensores. São apresentados, em separado, para uma melhor compreensão.

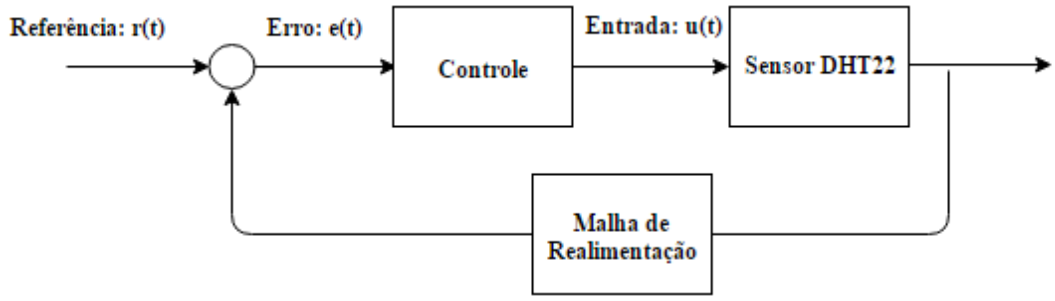


Figura 3.1 - Diagrama de Blocos DHT22
Fonte: Elaborado pelo autor

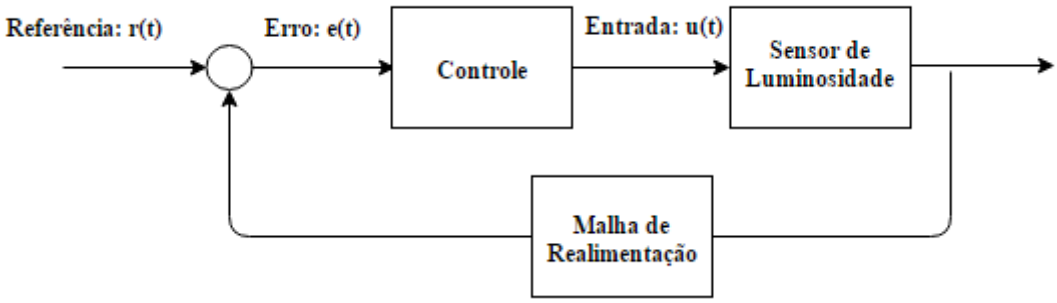


Figura 3.2 - Diagrama de Blocos LDR
Fonte: Elaborado pelo autor

O modelo representado nas figuras 3.1 e 3.2 mostram, de forma geral, o esquemático de como o projeto será montado, com a plataforma Arduino recebendo os dados dos sensores de umidade e temperatura (DHT22) e de luminosidade (LDR), atuando em cada um dos processos à fim de controlar os processos de temperatura e luminosidade da estufa. Será ainda utilizado um sistema de controle, independente das leituras dos sensores, para a irrigação hidropônica da estufa, que também será controlado pela plataforma Arduino. O sistema completo, contemplando os três sensores e o controle da irrigação, pode ser visto na figura 3.3. No caso a malha de realimentação, aplica-se, em cada caso, aos valores lidos na saída, em cada caso.

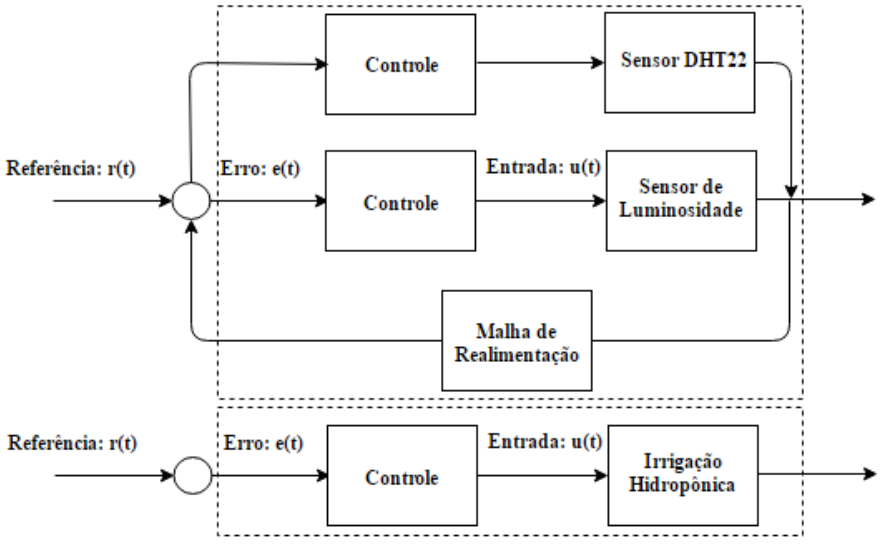


Figura 3.3 - Diagrama de Blocos (Exemplo)
Fonte: Elaborada pelo autor

Neste sistema proposto serão analisadas a temperatura, luminosidade e a umidade do ar no ambiente construído (uma estufa feita em material acrílico), por meio de sensores, a fim de demonstrar que o microclima deste espaço pode ser controlado de forma automática pelo

sistema aqui apresentado. Em um ambiente de dimensões diferentes das que serão aqui apresentadas, será necessário avaliar os parâmetros para tal condição e dada a aplicação do hardware aqui utilizado, poderá levar maior ou menor tempo para a obtenção dos parâmetros ideais. Assim, os dados obtidos neste projeto referem-se ao ambiente do tamanho da estufa projetada.

O controle será feito por uma placa de circuito, que com uma programação adequada, após a leitura e comparação dos valores, decidirá se aciona algum dos dispositivos instalados na estufa por intermédio do *shield* de relés de 4 canais, a saber:

- Uma ventoinha: será utilizada para controle da temperatura, sendo ligado quando a temperatura se elevar demais e desligado quando a temperatura chegar a um nível adequado;
- Um umidificador: será utilizado para auxiliar no controle da temperatura e para manter o ambiente com uma umidade do ar sempre acima de 60%;
- Uma lâmpada: será utilizada para prover luz artificial para as hortaliças, quando esta for insuficiente.

Este sistema, ainda, apresentará um sistema de irrigação hidropônico NFT, que consiste em imergir as raízes da planta em uma fluxo laminar de nutrientes para que estes sejam absorvidos. Isto será alcançado por meio de uma bomba d'água controlada pela mesma placa de circuito, que efetuará periodicamente a passagem da solução contida em um reservatório nas raízes das hortaliças. A programação para estas ações será apresentada nos próximos itens deste capítulo.

Os dados do sistema serão apresentados no visor LCD em tempo real, além de serem enviados por meio da *Internet* para a plataforma *ThingSpeak*, que exibirá os dados coletados para o usuário de maneira remota.

O desenvolvimento do projeto foi realizado em quatro partes:

1. Desenvolvimento da estufa: foram utilizadas placas de acrílico de 3mm para a construção de uma estufa capaz de comportar os dispositivos a serem utilizados para controle e monitoramento do clima da mesma, sendo eles um umidificador, um cooler e um soquete para lâmpada.
2. Desenvolvimento *software*: a programação do projeto compõe o desenvolvimento de um código capaz de controlar e monitorar o *hardware* do sistema, bem como enviar via Internet os dados coletados pelos sensores para a plataforma *ThingSpeak*.

3. Desenvolvimento de circuito: a placa desenvolvida integra a plataforma Arduino, o *Shield Ethernet*, o módulo de relés, os sensores e um visor LCD. A função da plataforma Arduino nesta placa é controlar todos estes dispositivos, bem como alimentá-los.
4. Desenvolvimento do protótipo: durante o desenvolvimento do protótipo foi necessário construir o sistema de irrigação hidropônico dentro da estufa, bem como transportar o circuito desenvolvido na protoboard para uma placa de fenolite. Foi também construída uma caixa, também em acrílico, que comportasse o visor LCD e o circuito, bem como suas entradas de alimentação e comunicação.
5. Testes: foi examinado o funcionamento da estufa, a fim de validar suas funcionalidades para melhor integração e eficácia do sistema.
6. Análise de Resultados: os resultados obtidos a partir dos testes efetuados na quinta etapa foram analisados e interpretados.

O diagrama de blocos da figura 3.1 mostra as etapas do desenvolvimento do projeto:

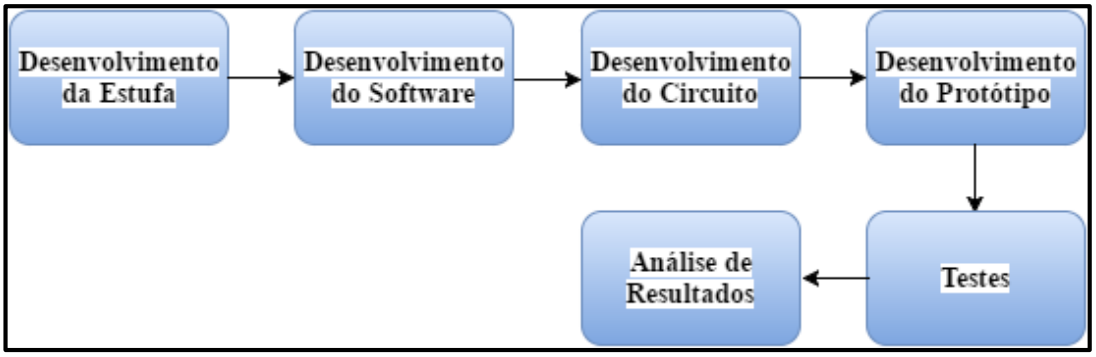


Figura 3.4 - Diagrama de blocos
Fonte: Elaborado pelo autor

A automação do projeto é controlada pela plataforma Arduino, que alimenta os sensores (luz, umidade do ar e temperatura) e converte suas grandezas físicas em sinais digitais, utilizando estas leituras para operar seus relés, responsáveis por ligar ou desligar os dispositivos da estufa, sendo eles a ventoinha, o umidificador e a lâmpada. O sistema de irrigação opera independente dos sensores, ligando e desligando periodicamente, também controlado pela plataforma Arduino. O visor LCD apresenta os dados provenientes dos sensores de forma suscinta, informando a temperatura dentro da estufa, sua umidade do ar, sua sensação térmica e seu nível de luminosidade.

Por fim, o *shield* de Ethernet é responsável por estabelecer uma conexão entre a plataforma Arduino e a Internet, possibilitando que os dados dos sensores sejam enviados para

a plataforma ThingSpeak, tornando possível ao usuário o monitoramento do funcionamento da sua estufa remotamente. A figura 3.2 representa o esquemático com a sequência das ações no funcionamento do sistema:

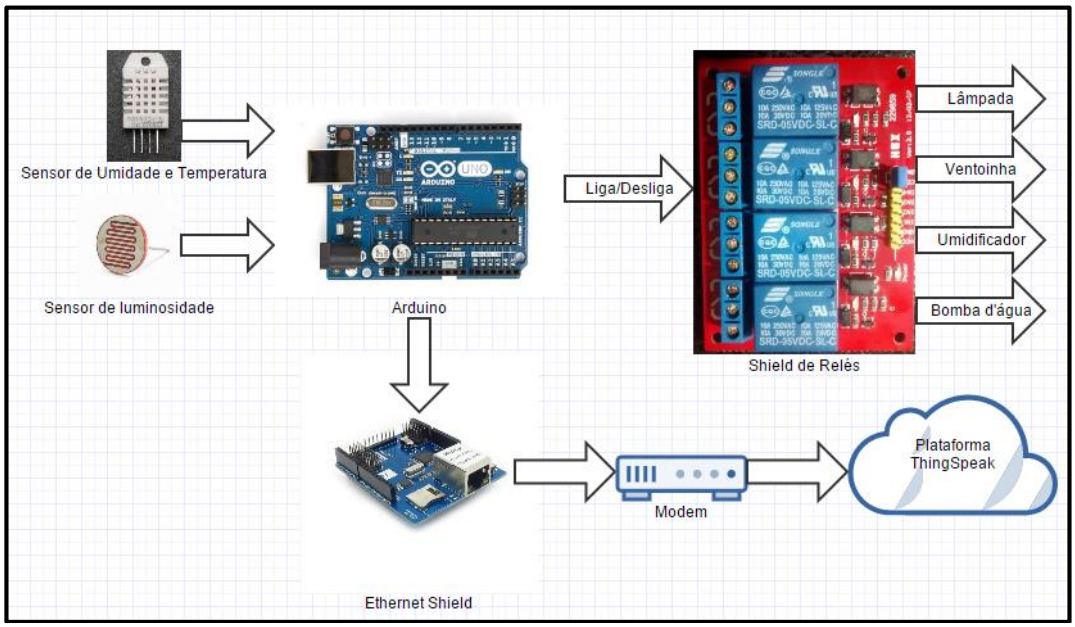


Figura 3.5 - Esquemático com a sequência de ações do Funcionamento
Fonte: Elaborado pelo autor

3.2 Construção da Estufa

A estufa servirá como ambiente protegido e controlado para o cultivo das hortaliças. Para sua construção, foram utilizadas placas de acrílico para a confecção de uma caixa de 60 cm de largura, 40 cm de altura e 30 cm de profundidade, com uma tampa removível para que possa ser feita a ventilação manual, caso necessário. Após a construção da caixa, foi instalada uma calha para a passagem dos fios por dentro da estufa, além de oferecer suporte aos sensores LDR e DHT22. A figura 3.3 mostra a caixa construída, já com o sistema hidropônico em sua posição:

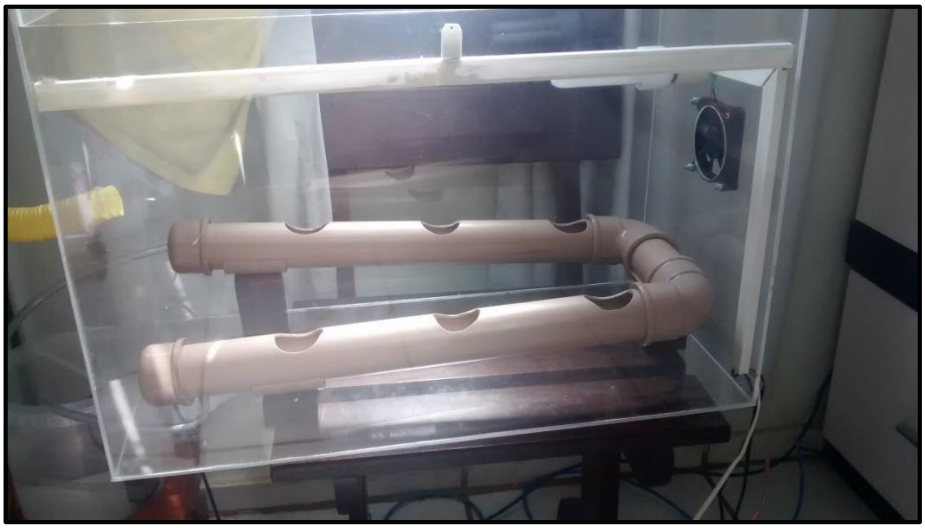


Figura 3.6 - Estufa construída
Fonte: Foto elaborada pelo autor

3.2.1 Dispositivos Instalados

Em conjunto com os sensores instalados na estufa, serão ativados dispositivos para o controle do ambiente, sendo eles: uma ventoinha, para o controle de temperatura; uma lâmpada, para que seja provida luz artificial quando baixa luminosidade; e um umidificador, para que o clima na estufa não fique muito seco e de forma a auxiliar o controle da temperatura. Será, ainda, incluso o sistema hidropônico construído dentro da caixa, com seus drenos necessários para o lado externo da estrutura.

3.2.1.1 Ventoinha

Para que fosse realizada a ventilação da estufa, foi instalada uma ventoinha de 12v em sua lateral. O acionamento e desligamento desta ventoinha são controlados pelo Arduino, que liga ou desliga o módulo de relé em que é suprida a energia para o seu funcionamento. A figura 3.4 mostra a ventoinha utilizada:

Figura 3.7 – Ventoinha
Fonte: Foto elaborada pelo autor

3.2.1.2 Lâmpada

Para a iluminação, foi instalado um soquete E27 (padrão brasileiro) de 220V. A energia suprida a este soquete também será controlada pelo Arduino que operará o módulo de relés. A lâmpada escolhida foi uma fluorescente eletrônica devido ao seu baixo custo e baixo consumo, além de não esquentar tanto quanto as incandescentes. A figura 3.5 mostra o soquete instalado na caixa, tal como a lâmpada a ser utilizada:

Figura 3.8 - Soquete E27 com Lâmpada
Fonte: Foto elaborada pelo autor

3.2.1.3 Umidificador

Para que fosse feito o controle da umidade do ar no ambiente da estufa, foi instalado um umidificador simples de 220V, cuja fonte de energia também é gerenciada pelo Arduino, por meio da operação do módulo de relés. Ele será acionado quando a umidade do ar na estufa baixar demais ou a temperatura subir demais. Ao chegar a valores adequados, ele será

desligado. A figura 3.6 mostra o umidificador utilizado no projeto, tal como o adaptador construído para que seu vapor seja direcionado para a estufa:

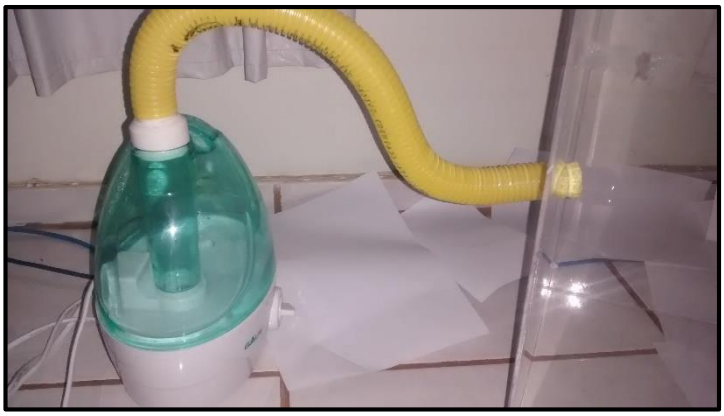


Figura 3.9 – Umidificador
Fonte: Foto elaborada pelo autor

3.2.2 Sensor de Temperatura e Umidade

Nesta etapa foi instalado o sensor de umidade e temperatura DHT22 ao Arduino, para que seja possível obter a temperatura e umidade do ar ambiente. O sensor DHT22 possui 4 pinos, como dito e mostrado no capítulo 2 (VCC, dados, nulo, GND). O pino VCC foi alimentado em 5V (jumper vermelho), o pino de dados foi conectado ao pino analógico A1 do Arduino (jumper amarelo), o pino nulo não tem necessidade de ser conectado e o pino GND foi ligado ao GND do próprio Arduino (jumper preto), conforme mostrado na figura 3.7 abaixo:

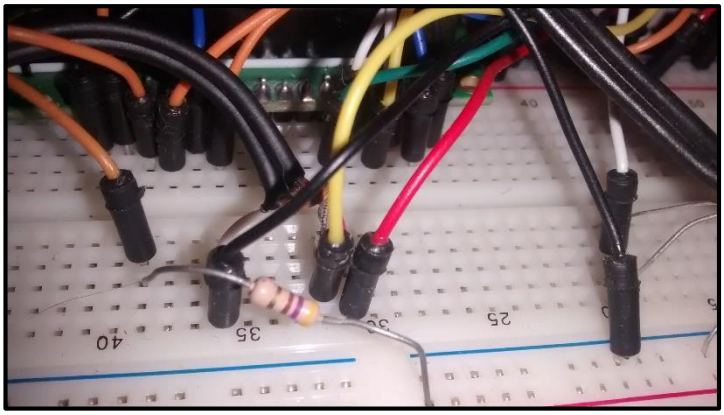


Figura 3.10 - Conexões DHT22
Fonte: Foto elaborada pelo autor

Em seguida, foram soldados três fios (vermelho, preto e branco) ao sensor DHT22. Estes fios são guiados pela calha instalada até a placa de circuito. A figura 3.8 mostra como ficaram as conexões no DHT22:

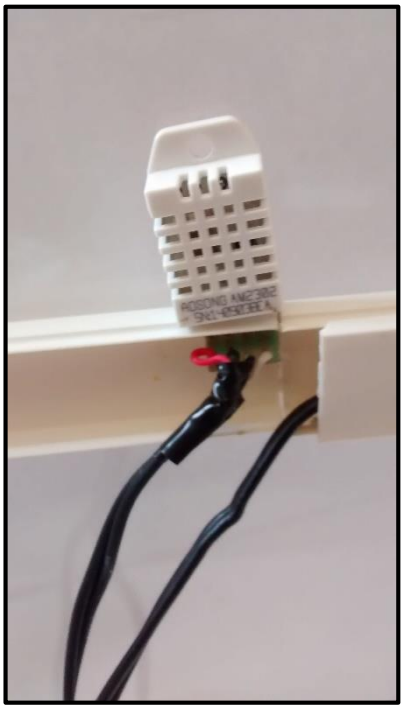


Figura 3.11 - Novas conexões DHT22
Fonte: Foto elaborada pelo autor

O fio branco corresponde ao GND. O fio vermelho está ligado ao VCC e, por fim, o fio preto ao pino de dados.

3.2.3 Sensor de Luminosidade

Para efetuar as leituras da luminosidade do local em que a estufa está posicionada, foi instalado um sensor de luminosidade (LDR), conforme mostra a figura 3.9:

Figura 3.12 - Sensor de Luminosidade Instalado
Fonte: Foto elaborada pelo autor

Este sensor realiza leituras periódicas da luminosidade incidente na estufa, enviando suas informações ao Arduino, para que o mesmo opere a lâmpada instalada.

3.2.4 Fonte de Alimentação

Nesta etapa, foi necessário a construção de uma fonte de alimentação para a plataforma Arduino e a ventoinha do sistema, tendo em vista que suas tensões não são compatíveis com a rede elétrica comum (110/220v). Para tal, foi construída uma fonte de alimentação de onda completa com *center-tape* (derivação central) com duas saídas, possibilitadas pelo uso de reguladores de tensão: uma saída de 12v (regulador U2 7812 - para a alimentação da ventoinha) e uma saída 5v (regulador U1 7805 - para a alimentação da plataforma Arduino). A figura 3.13 mostra o esquemático para a construção da fonte. Os dois capacitores C1 e C2 irão diminuir as flutuações da onda retificada, fornecida às cargas, indicadas por D1, U1 e U2, na figura.

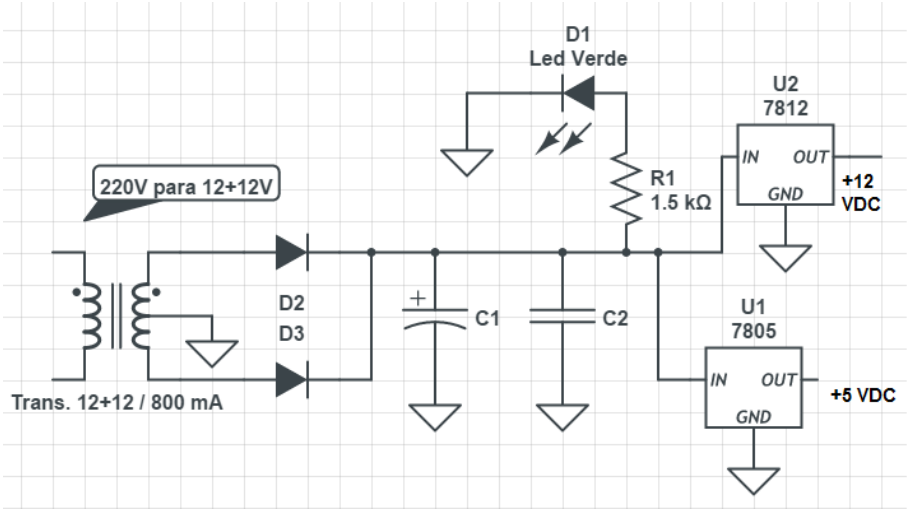


Figura 3.13 – Esquemático Fonte de Alimentação
Fonte: Elaborado pelo autor

3.2.5 Sistema de Irrigação Hidropônico

Para a construção do modelo hidropônico proposto, foram utilizados dois canos, dois joelhos e dois tampões de 50mm de diâmetro, em PVC, com furos de 10 centímetros de distância entre eles. Nestes orifícios encontram-se recipientes de plástico para comportar as

raízes das hortaliças, dando-as estabilidade e suporte de maneira que não sejam arrastadas pela solução aquosa.

Para o dreno da solução aquosa foram feitos dois furos, mostrados na figura 3.10, permitindo que fossem passadas duas mangueiras para condução da solução aquosa:

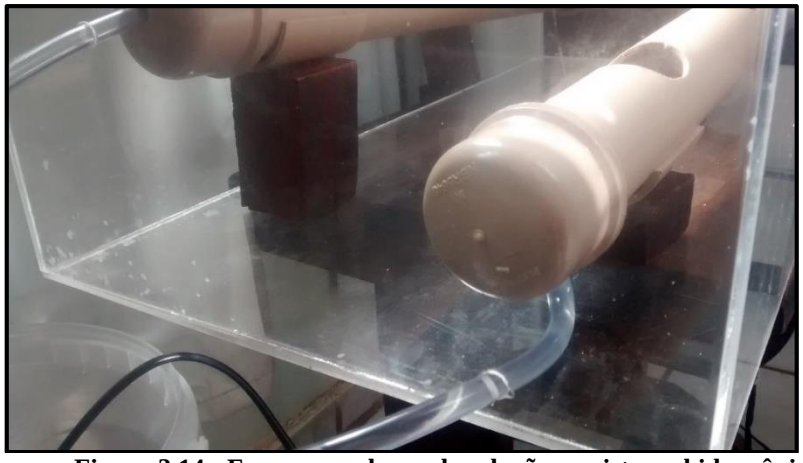


Figura 3.14 - Furos para dreno de solução no sistema hidropônico
Fonte: Foto elaborada pelo autor

A figura 3.11 mostra o sistema hidropônico montado dentro da estufa, em sua integridade:

Figura 3.15 - Sistema hidropônico montado dentro da estufa
Fonte: Foto elaborada pelo autor

3.3 Desenvolvimento do *Software*

O desenvolvimento do *sketch* (programa) do sistema proposto foi feito na IDE do Arduino, em sua própria linguagem, com o objetivo de suprir as seguintes demandas:

1. Capacidade de receber as informações dos sensores e interpretá-las, a fim de controlar os dispositivos que exercerão o controle do microclima da estufa;
2. Enviar as informações provenientes dos sensores por meio da Internet para a plataforma *ThingSpeak*, a fim de disponibilizar estas informações de maneira clara e organizada ao usuário.

A seguir será descrito de forma detalhada todas as etapas de construção do *software*, além de que o mesmo, em sua íntegra, será apresentado no apêndice A deste trabalho.

3.3.1 Inclusão das Bibliotecas

O *software* de desenvolvimento da plataforma Arduino possibilita o uso de Bibliotecas, funções pré-compiladas a fim de facilitar e padronizar a programação. A instalação padrão do *IDE* da plataforma Arduino é composta por diversas bibliotecas, dando ainda a possibilidade da instalação de outras. Para o desenvolvimento do sistema proposto foi necessário a utilização das seguintes bibliotecas:

- "DHT.h" - responsável pelo funcionamento do sensor de umidade e temperatura DHT22;
- "LiquidCrystal.h" - responsável pelo funcionamento do display;
- "SPI.h" e "Ethernet.h" - são responsáveis pela comunicação entre a plataforma Arduino e o *Ethernet Shield*, possibilitando a integração de ambos.

A figura 3.12, abaixo, mostra a inclusão das bibliotecas no código:

```
//Bibliotecas
#include "DHT.h" //Biblioteca DHT
#include <LiquidCrystal.h> //Biblioteca Display
#include <SPI.h> //Biblioteca do Shield Ethernet¹
#include <Ethernet.h> //Biblioteca do Shield Ethernet²
```

Figura 3.16 - Declaração de Bibliotecas
Fonte: Elaborado pelo autor

3.3.2 Preparação de Dispositivos

Cada dispositivo utilizado em conjunto com a plataforma Arduino necessita da sua própria configuração. Geralmente, essa configuração consiste em fixar qual porta o dispositivo está utilizando e define-se uma variável para tratar da comunicação de informações entre o dispositivo e a plataforma Arduino. No projeto foram apontadas quais portas seriam utilizadas para cada um dos dispositivos, criando suas respectivas variáveis correspondentes:

- Visor LCD: para a utilização do visor LCD utilizou-se a variável "LiquidCrystal" para definir quais portas da plataforma Arduino serão utilizadas pelo display, além de construir um array, a fim de utilizar o símbolo de grau (º) no visor, conforme mostrado na figura 3.13:

```
//Display
LiquidCrystal lcd(5, 6, 7, 9, 8, A4); /* Define os pinos que
serão ligados ao LCD */

//Array simbolo grau
byte grau[8] = { B00001100,
                 B00010010,
                 B00010010,
                 B00001100,
                 B00000000,
                 B00000000,
                 B00000000,
                 B00000000,
               };

//Fim do Display
```

Figura 3.17 - Código de Preparação do Display
Fonte: Elaborado pelo autor

- Sensor DHT22: para a configuração do sensor DHT22 foi necessário definir qual porta do Arduino receberia os dados do sensor, além do seu tipo de sensor instalado (pois a biblioteca abrange outros sensores de umidade) e inicializá-lo, conforme descrito na figura 3.14 abaixo:

```
//Sensor de Temperatura e Umidade
#define DHTPIN A1     // Pino em que o sensor de umidade e temperatura está conectado
#define DHTTYPE DHT22   // Tipo do sensor de umidade e temperatura: DHT 22  (AM2302)
// Inicializando o sensor DHT para um Arduino normal de 16 Mhz
DHT dht(DHTPIN, DHTTYPE);
//Fim do Sensor de Temperatura e Umidade
```

Figura 3.18 - Código de Preparação DHT22
Fonte: Elaborado pelo autor

- Sensor de Luminosidade: a configuração do sensor de luminosidade consiste em definir em qual pino (A2) ele está instalado, e criar uma variável (sensorLuz) para armazenar seu valor analógico. Para o projeto foi criada também uma variável de tratamento (luminosidade) para o valor obtido, conforme mostra a figura 3.15.

```
//Sensor de Luminosidade
int ldrPin = A2; //Pino do sensor
int sensorLuz; //Variável para armazenar o valor analógico
float luminosidade; //Variável para armazenar o valor da tensão após a conversão do valor analógico
//Fim do Sensor de Luminosidade
```

Figura 3.19 - Código de Preparação LDR
Fonte: Elaborado pelo autor

- *Shield* de Relés: para utilização do *Shield* de Relés, é necessário definir qual pino da plataforma Arduino cada relé irá utilizar, conforme figura 3.16:

```
//Relay
#define vent A3 //Ventilador ligado no pino 0
#define lamp A0 //Lâmpada ligada no pino 1
#define irri 2 //Sistema de Irrigação no pino 2
#define umid 3 //Umidificador ligado no pino 3
//Fim Relay
```

Figura 3.20 - Código de Preparação Relé
Fonte: Elaborado pelo autor

- *Shield* Ethernet: a preparação do *Shield Ethernet* consiste em definir seu endereço físico (*Mac Address*) e iniciá-lo, pois suas demais configurações já estão inclusas em sua biblioteca, conforme referenciado no capítulo 2. Porém, a versão da placa utilizada no projeto possui uma entrada de cartão SD, sendo necessário, assim, a definição de uma variável de controle (on/off) do cartão SD, para que não tenha riscos de falhas do *shield*. A figura 3.21 mostra este código:

```
//Início Ethernet
byte mac[] = { 0xDE, 0xAE, 0xBC, 0xED, 0xFE, 0x02 };
int sdcard = 4;
EthernetClient client; // Iniciar Client Ethernet
```

Figura 3.21 - Código de Preparação Shield Ethernet
Fonte: Elaborado pelo autor

- Plataforma ThingSpeak: a configuração da plataforma *ThingSpeak* ocorre a partir da definição do endereço de seu API (Application Programming Interface) na variável "thingSpeakAddress[]" e uma chave única na variável "writeAPIKey". Esta chave é dada após a criação do canal na plataforma *ThingSpeak* e serve como senha para o envio de informações, não devendo então ser revelada publicamente, para efeitos de segurança das informações. Em seguida, é necessário configurar o intervalo de tempo entre *updates* que, por definição da própria plataforma são de 16 segundos. As variáveis "lastConnectionTime", "lastCheck", "lastConnected" e "failedCounter" são utilizadas para controle da conexão e o código está mostrado na figura 3.22.

```
//Configurações ThingSpeak
char thingSpeakAddress[] = "api.thingspeak.com";
String writeAPIKey = "[redacted]";
const int updateThingSpeakInterval = 16 * 1000; // Intervalo de tempo para update no ThingSpeak
//Setup de Variáveis
long lastConnectionTime = 0;
long lastCheck = 0;
boolean lastConnected = false;
int failedCounter = 0;
```

Figura 3.22 - Código de Preparação ThingSpeak
Fonte: Elaborado pelo autor

3.3.3 Inclusão dos Sensores

O reconhecimento dos valores dos sensores é efetuado na estrutura "*void loop"* que realiza infinitamente um determinado processo ou até que se atinja uma condição pré-estabelecida. No projeto, esta estrutura se repetirá infinitamente, uma vez que o sistema deve funcionar realizando as leituras de forma contínua.

Para inclusão do sensor DHT22, foram definidas três variáveis ("h", "t" e "f"), respectivamente para os valores de umidade do ar, temperatura em graus celsius e temperatura em fahrenheit. Para o sensor LDR, foram definidas duas variáveis: "sensorLuz" para o valor de leitura analógico do LDR e "luminosidade" para que fosse feito o tratamento deste valor e sua devida calibração.

O sensor de umidade envia todos os seus dados de leitura por meio de seu pino de dados para o Arduino. Estes dados são interpretados por códigos já contidos na biblioteca referida. As funções "dht.readHumidity()", "dht.readTemperature()" e "dht.readTemperature(true)" são responsáveis, respectivamente, pela leitura dos valores de umidade do ar, temperatura em graus celsius e temperatura em fahrenheit. Em seguida são criadas as strings "temp" e "humid" para que estes valores sejam enviados à plataforma *ThingSpeak*, conforme descrito na figura 3.23 abaixo.

```
// Sensor de Umidade e Temperatura
float h = dht.readHumidity(); // Lê o valor da umidade
float t = dht.readTemperature(); // Lê a temperatura em Celsius
float f = dht.readTemperature(true);  // Lê a temperatura em Fahrenheit
// Criação de Strings
char t_buffer[10]; // Buffer da Temperatura
char h_buffer[10]; // Buffer da Umidade
String temp = dtostrf(t, 0, 5, t_buffer); // String da Temperatura
String humid = dtostrf(h, 0, 5, h_buffer); // String da Umidade
// Fim Sensor de Umidade e Temperatura
```

Figura 3.23 - Definições de Variáveis DHT22
Fonte: Elaborado pelo autor

As leituras do sensor de luminosidade são efetuadas pelo Arduino por meio de uma leitura analógica do sinal emitido na porta em que o sensor está conectado. Por este valor ser volátil entre 0 e 1023, dependendo da quantidade de luz inferida no LDR, foram criadas duas variáveis de tratamento. A variável "sensorLuz" recebe as leituras diretas do LDR, enquanto a variável "luminosidade" efetua o tratamento deste valor. A função "map()" é utilizada para transformar os valores identificados pela variável "sensorLuz" em uma escala de 0 a 1000. Em seguida, este valor é dividido por 10 para valores mais controlados. Posteriormente, é criada uma string "luz" para que estes valores sejam enviados à plataforma *ThingSpeak*, conforme evidenciado na figura 3.24.

```
//Sensor de Luminosidade
sensorLuz = analogRead(ldrPin);
sensorLuz = 1023 - sensorLuz;
luminosidade = map(sensorLuz, 0, 1023, 1000, 0);
luminosidade = (luminosidade / 10);
char l_buffer[10]; // Buffer da Luminosidade
String luz = dtostrf(luminosidade, 0, 5, l_buffer); // String da Lumi
//Fim Sensor de Luminosidade
```

Figura 3.24 - Definições de Variáveis LDR
Fonte: Elaborado pelo autor

3.3.4 Configuração Display LCD

O visor escolhido é compatível com a biblioteca “LiquidCrystal” da plataforma Arduino, o que facilita sua integração com o projeto e sua programação. Para configurá-lo, é necessário iniciar sua estrutura no *void setup()*. Para isso, é chamada a função “lcd.begin(16, 2)” que inicializa o visor nas configurações de 16 caracteres em 2 linhas de texto. Em sequência, são chamadas as funções “lcd.clear()” para que a tela seja limpa de quaisquer resquício de caracteres e a função “lcd.createChar(0, grau)”, que tem como objetivo permitir o uso do caractere de grau (º) criado na etapa de preparação. A figura 3.21 mostra a inicialização do visor:

```
lcd.begin(16, 2); //Inicializa LCD
lcd.clear(); //Limpa o LCD
lcd.createChar(0, grau); //Cria o caractere customizado com o símbolo do grau
```

Figura 3.25 - Código de Inicialização do Visor LCD
Fonte: Elaborado pelo autor

Para que seja possível escrever no display, é necessário informar a posição inicial do primeiro caractere por meio da função “lcd.setCursor”, seguido pela função “lcd.print()” para definir os dados a serem escritos. No projeto, estes comandos foram utilizados para apresentar as informações dos sensores de umidade do ar, temperatura e luminosidade. No caso do sensor de luminosidade foram criadas condições if{...} para que esta informação seja descrita de maneira nominal. Pelo fato deste visor comportar apenas duas linhas de informação por vez, foi necessário atribuir um atraso no tempo em que as informações aparecem por meio da função “delay()”. Experimentalmente, este intervalo ficou de 2000 milisegundos, que é um tempo suficiente para que o usuário visualize a informação do display. A figura 3.26 apresenta as configurações descritas:

```
// Display Informações
lcd.clear(); // Limpa a tela
lcd.setCursor(0, 0); // Escrevendo na Primeira Linha
lcd.print("Temp : "); // Temperatura
lcd.print(" ");
lcd.setCursor(7, 0);
lcd.print(t, 1);
lcd.setCursor(12, 0);
lcd.write((byte)0); //Mostra o simbolo do grau formado pelo arr

lcd.setCursor(0, 1); // Escrevendo na Segunda Linha
lcd.print("Umid : "); // Umidade
lcd.print(" ");
lcd.setCursor(7, 1);
lcd.print(h, 1);
lcd.setCursor(12, 1);
lcd.print("%");
delay(2000); //Intervalo para próxima informação
lcd.clear();

lcd.setCursor(0, 0); // Escrevendo na Terceira Linha
lcd.print("Luz : "); // Luminosidade
lcd.print(" ");
lcd.setCursor(7, 0);
if ( luminosidade >= 65.00 ) {
  lcd.print("Luz Alta");
  digitalWrite(lamp, LOW);
}
else {
  lcd.print("Luz Baixa");
  digitalWrite(lamp, HIGH);
}
```

Figura 3.26 - Código para display de informações no visor LCD
Fonte: Elaborado pelo autor

3.3.5 Configuração Ethernet Shield

Para a utilização do *Shield Ethernet*, é necessário inicializá-lo durante a preparação do programa, conforme apontado anteriormente, por meio da função “EthernetClient”. Foi criada, então, a função “inicioEthernet()” para conectar a plataforma Arduino com a Internet. Esta precisa ser chamada apenas uma vez, ficando contida no “void setup{...}” do código.

A função “inicioEthernet()” é iniciada desligando imediatamente o servidor por meio do comando “client.stop()”. Esta é uma medida preventiva para minimizar erros de conexão. Os comandos “Serial.println()” são utilizados com o propósito de visualizar por meio do monitor serial do Arduino se o sistema está funcionando. Em seguida é criada uma condição “if(...)” que testa se a conexão está funcionando ou não. Caso não esteja, é apresentada a

mensagem de erro “DHCP falhou, reinicie o Arduino”. Em caso de sucesso, é apresentada a mensagem de sucesso "Conectado a rede utilizando DHCP". A figura 3.27, abaixo, apresenta o código descrito:

```
//Conectar o Arduino na rede e obter um IP usando o DHCP
void inicioEthernet() {
  client.stop();
  Serial.println("Conectando o Arduino a rede...");
  Serial.println();
  delay(1000);
  if (Ethernet.begin(mac) == 0) {
    Serial.println("DHCP falhou, reinicie o Arduino");
    Serial.println();
  }
  else {
    Serial.println("Conectado a rede usando DHCP");
    Serial.println();
  }
}
```

Figura 3.27 - Código da Função inicioEthernet
Fonte: Elaborado pelo autor

3.3.6 Configuração ThingSpeak

A configuração da plataforma ThingSpeak se dá inicialmente no *website* da plataforma, criando uma conta ao clicar em cadastro (lado direito da tela), conforme mostra a figura 3.28:

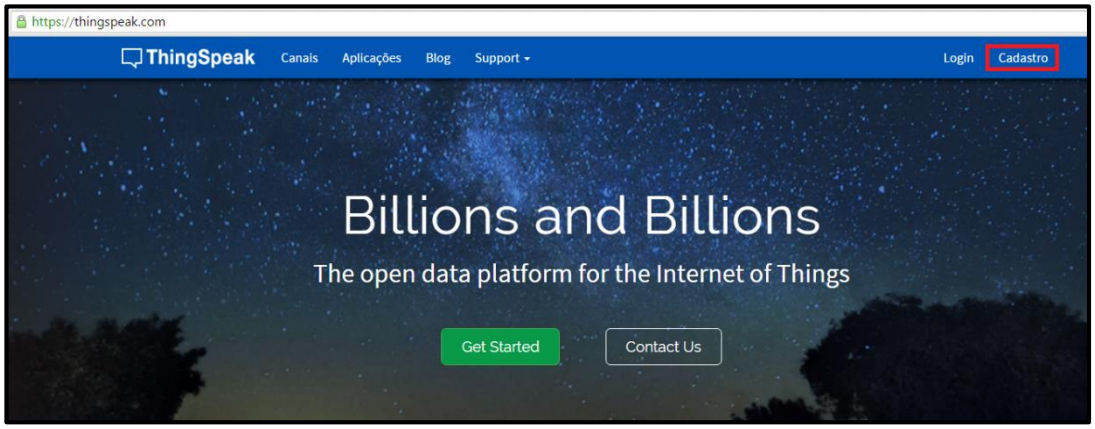


Figura 3.28 - Cadastro Plataforma ThingSpeak
Fonte: Elaborado pelo autor

Com o cadastro criado e autenticado, é necessário criar um canal para receber os dados, conforme descrito no capítulo 2 e evidenciado na figura 3.29:

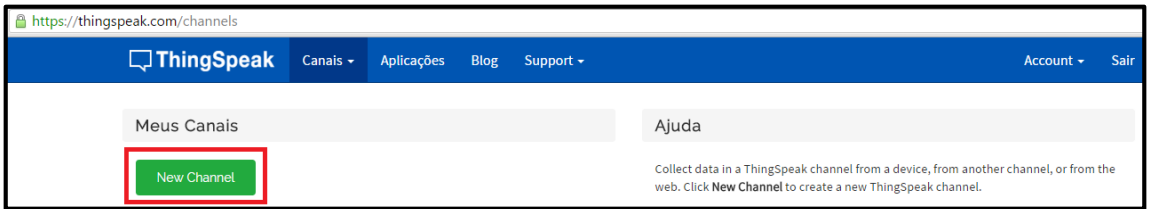


Figura 3.29 - Criação de Canal na Plataforma ThingSpeak
Fonte: Elaborado pelo autor

A criação do canal se porta de maneira bastante intuitiva, sendo necessário apenas marcar as *checkboxes* referentes ao número de campos que serão enviados para o canal. No caso do projeto, serão utilizados três campos de dados: temperatura, umidade do ar e luminosidade. O resto permanece desmarcado, pois não serão utilizados, conforme demonstra a figura 3.30:

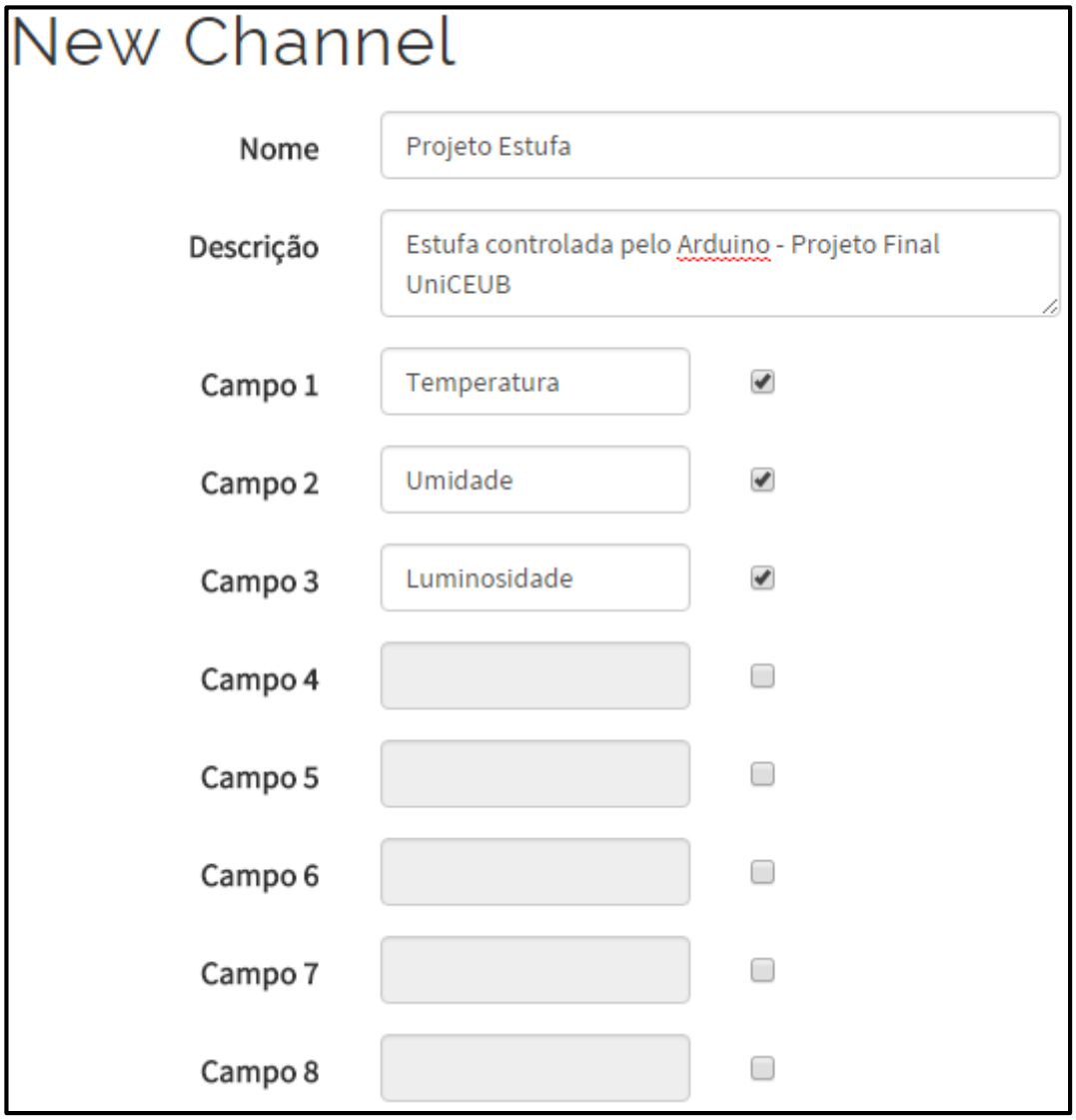


Figura 3.30 - Criação do Canal Projeto Estufa
Fonte: Elaborado pelo autor

Após sua criação, é possível ter controle do canal, configurando seu acesso como público (qualquer um com o link pode ter acesso aos dados) ou privado (apenas o detentor do canal tem acesso). A página permite, ainda, configurar novos campos, exportar e importar dados. Porém, o mais importante é o acesso à chave de escrita do canal, que servirá como uma senha para a comunicação do Arduino com a plataforma *ThingSpeak*. Esta chave fica contida no código do *software* e deve ser mantida em sigilo, para efeitos de segurança da informação. A seguir, a figura 3.31 mostra aonde buscar esta chave:

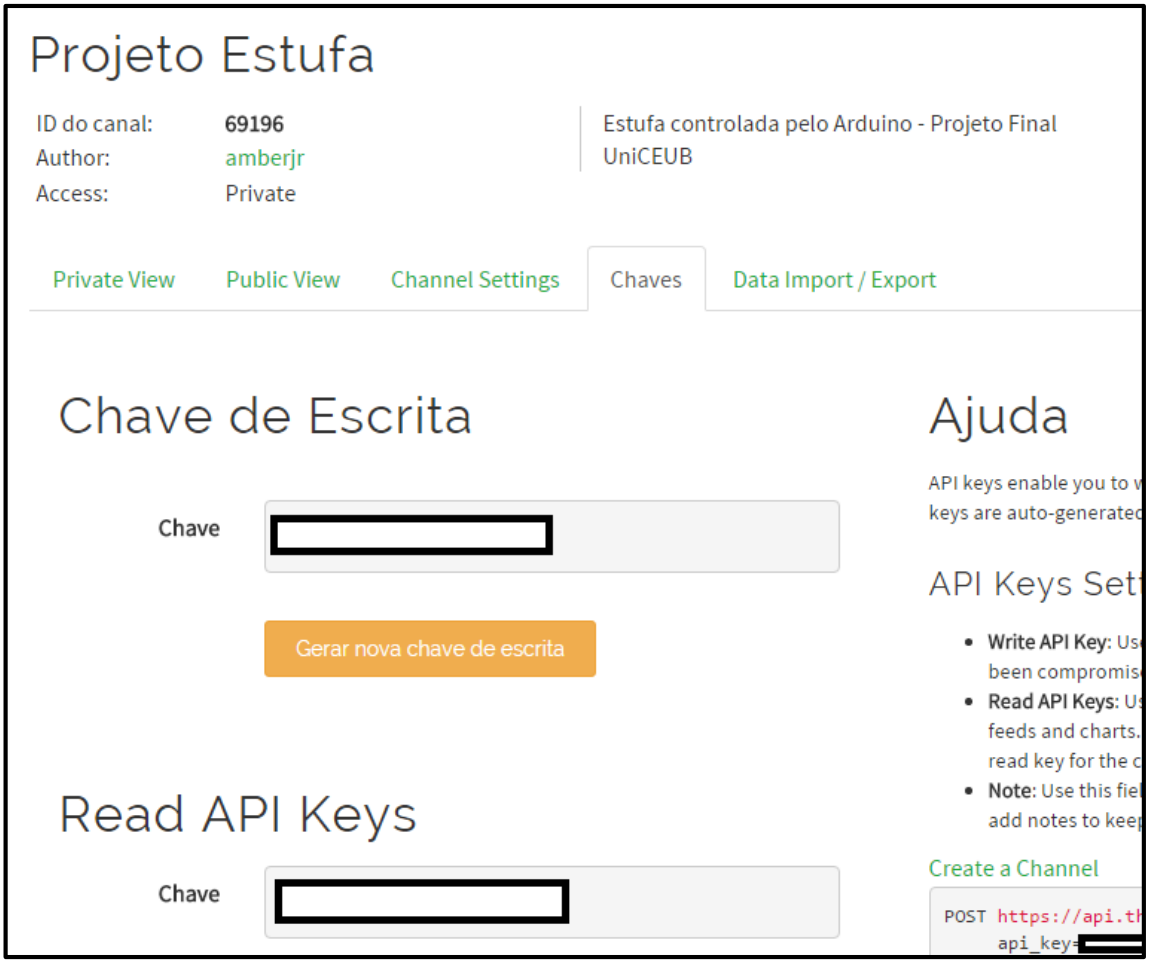


Figura 3.31 - Chave de Escrita ThingSpeak
Fonte: Elaborado pelo autor

Após designar esta chave de escrita no código de preparação do *ThingSpeak*, mostrado na figura 3.28, foi criada a função "updateThingSpeak" para efetuar o envio destas informações pelo método POST. A função é iniciada por uma condição de conexão. Caso o Arduino consiga se conectar ao canal designado e enviar as informações necessárias, o contador de falhas é zerado. Caso esta conexão falhe, o contador "failedCounter" é incrementado, a fim de que o

Shield Ethernet seja reiniciado após três tentativas de insucesso. Este código está mostrado na figura 3.32.

```
void updateThingSpeak(String tsData) { //Fazer update dos dados no ThingSpeak
  if (client.connect(thingSpeakAddress, 80)) {
    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: " + writeAPIKey + "\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(tsData.length());
    client.print("\n\n");
    client.print(tsData);
    lastConnectionTime = millis();
    if (client.connected()) {
      Serial.println("Conectando ao ThingSpeak...");
      Serial.println();
      failedCounter = 0;
    }
    else {
      failedCounter++;
      Serial.println("A conexão ao ThingSpeak falhou (" + String(failedCounter, DEC) + ")");
      Serial.println();
    }
  }
}
```

Figura 3.32 - Função updateThingSpeak
Fonte: Elaborado pelo autor

O chamado desta função se baseia em uma condição if(...), que realiza um teste para ver se o Arduino está conectado à Internet e, caso esteja, ela checa se o último *update* foi a 16 segundos atrás. Caso verdadeiro, novos dados são enviados pelas *strings* 'temp', 'humid' e 'luz', correspondentes a temperatura, umidade do ar e luminosidade. Esta condição é demonstrada na figura 3.33.

```
  //  ThingSpeak
  if (!client.connected() && lastConnected) {
    Serial.println("...desconectado");
    Serial.println();
    client.stop();
  }
  // Update do ThingSpeak
  if (!client.connected() && (millis() - lastConnectionTime > updateThingSpeakInterval))
  {
    updateThingSpeak("field1="+temp+"&field2="+humid+"&field3="+luz); // Envio das 3 String
  }
  // Checar se o Arduino precisa ser reiniciado (reinicia a Ethernet caso a conexão falhe m
  if (failedCounter > 3 ) {
    inicioEthernet();
  }
  lastConnected = client.connected();
 }
}
```

Figura 3.33 - Condição updateThingSpeak
Fonte: Elaborado pelo autor

3.3.7 Desenvolvimento da Placa do Circuito

A construção do protótipo em uma placa de circuito é a etapa final do desenvolvimento do projeto. Inicialmente, o projeto foi desenvolvido em uma *protoboard*, pois sua facilidade de manuseio é fundamental para testes iniciais em montagem. O projeto foi, então, transportado para uma placa de fenolite furada, pois este tipo de placa permite que sejam replicadas as mesmas ligações criadas em uma *protoboard*, devido aos seus designs similares, mostrados no comparativo da figura 3.34:

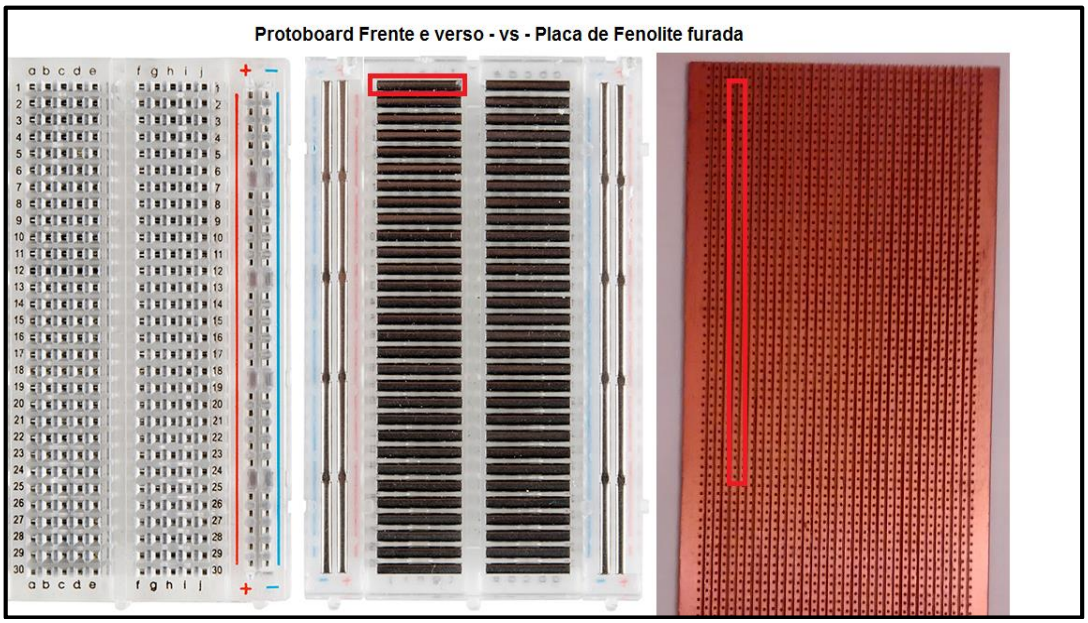


Figura 3.34 - Protoboard vs Placa de Fenolite Furada
Fonte: Elaborado pelo autor

O Arduino é o dispositivo mais importante da placa do circuito, pois ele controla todos os dispositivos do sistema, faz a comunicação com a plataforma *ThingSpeak* por intermédio do *Shield Ethernet*, recebe o sinal dos sensores e energiza o módulo de relés. Sendo assim, ele foi o primeiro dispositivo a ser fixado na placa de circuito. Para evitar que o conector do *Shield Ethernet* entrasse em contato com o Arduino, foi anexado uma borracha no topo do conector de alimentação conforme indica a figura 3.35:

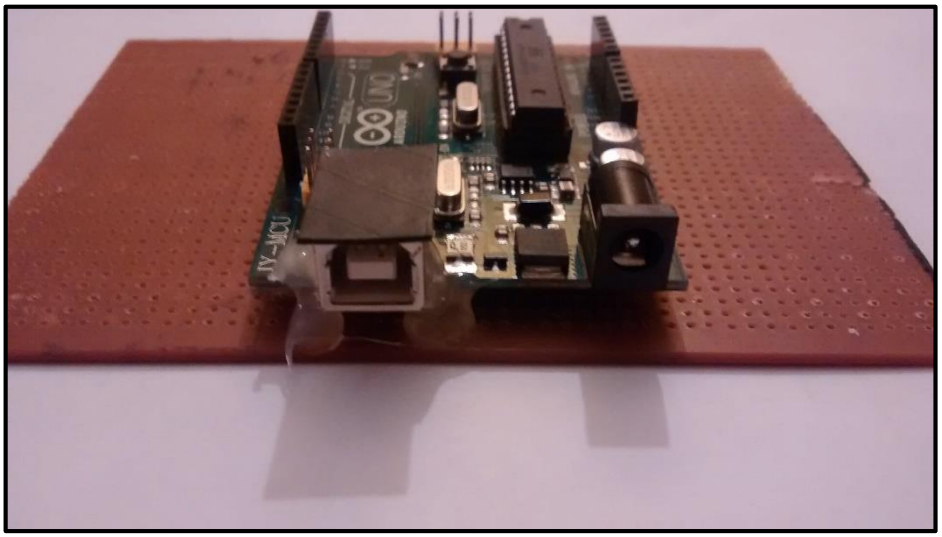


Figura 3.35 - Arduino fixado na placa de circuito
Fonte: Elaborado pelo autor

Para manuseio de uma placa de fenolite furada, é necessário efetuar uma raspagem dos contatos a serem isolados. Desta forma, após a definição de posição do Arduino, os contatos da placa de fenolite foram divididos em duas partes, simulando uma *protoboard*, conforme evidenciado na figura 3.36:

Figura 3.36 - Corte da placa fenolite
Fonte: Elaborado pelo autor

Para a conexão dos dispositivos, optou-se pela utilização de fios de jumper conectados ao Arduino e soldados na placa, para melhor manuseio. A figura 3.37 mostra todos os componentes instalados na placa:

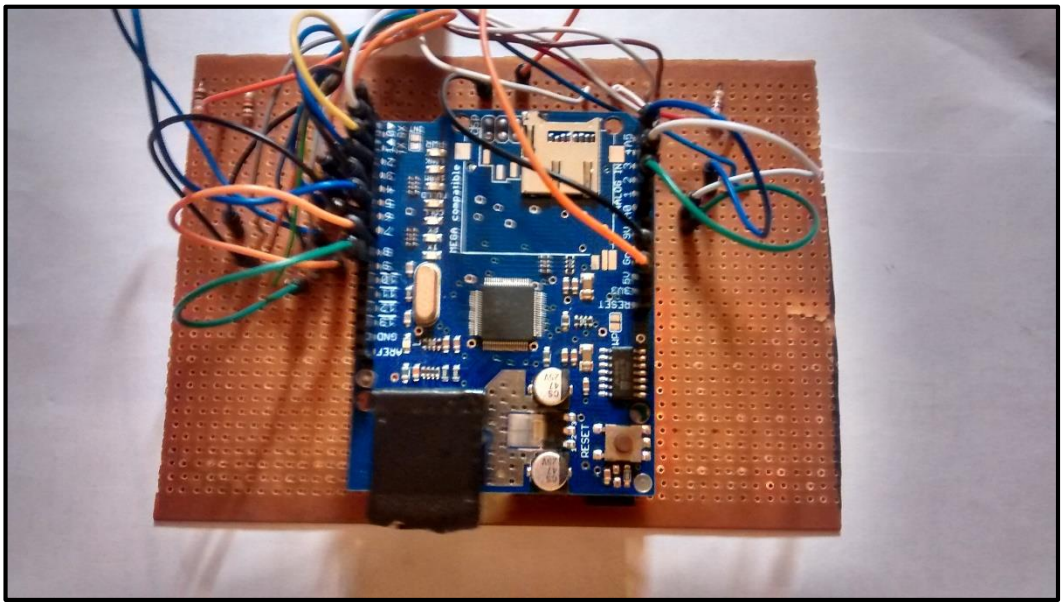


Figura 3.37 - Placa de circuito montada
Fonte: Elaborado pelo autor

A montagem final do protótipo deu-se a partir da construção de uma caixa, também de acrílico, a fim de comportar a placa de circuito e os fios necessários para a ligação de cada dispositivo. A figura 3.38 mostra o protótipo do sistema montado, com seus componentes instalados:

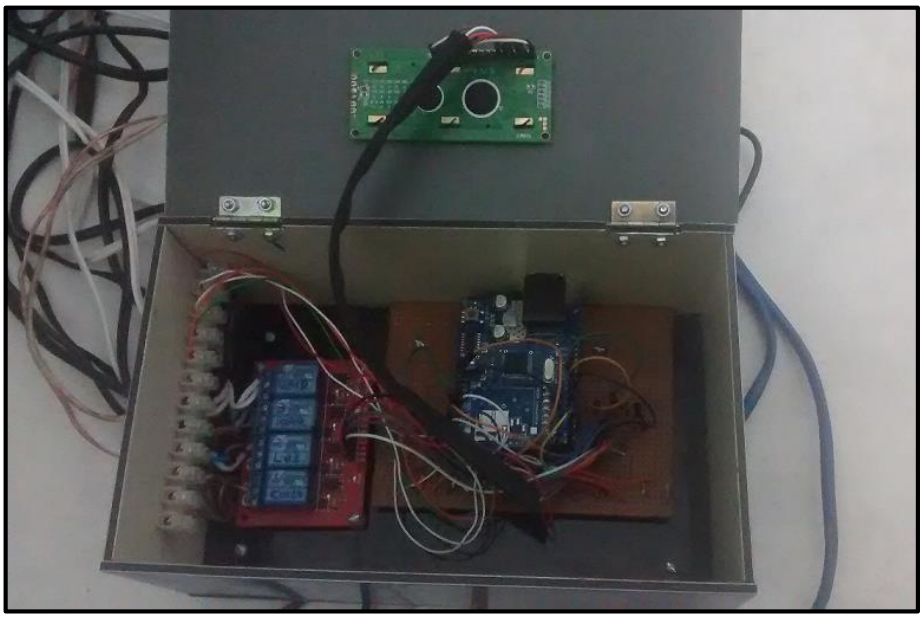


Figura 3.38 - Protótipo final
Fonte: Elaborado pelo autor

4 TESTES E RESULTADOS

Neste capítulo descrevem-se os testes realizados para verificar o funcionamento do projeto construído e seus resultados, com o objetivo de avaliar se o mesmo atende as especificações que foram propostas.

Para tal, foram testados os sensores e o *Shield Ethernet* utilizados a fim de verificar se os dados obtidos apresentam resultados satisfatórios, com comunicação confiável e sem falhas.

4.1 Cenário 1

O objetivo deste teste foi verificar o funcionamento do sistema com as funcionalidades do sensor DHT22 para medições de temperatura e umidade do ar. Com este intuito, o teste foi dividido em duas etapas: (A) Teste de Temperatura e (B) Teste de Umidade do Ar.

A. Teste de Temperatura

No teste de temperatura foi feito a leitura inicial do sensor DHT22. Em seguida, aproximou-se ar quente por meio de um soprador térmico e constatado um aumento de sua temperatura a cada dois segundos, conforme leitura programada no código configurado. A figura 4.1 mostra a evolução da temperatura ao longo do tempo:

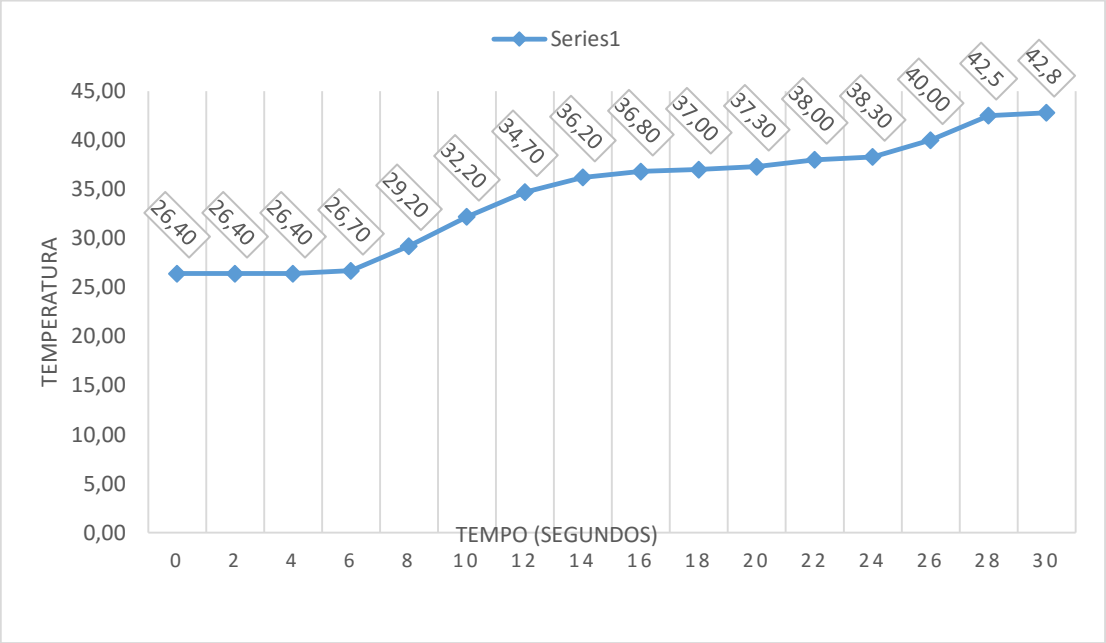


Figura 4.1 - Teste de Temperatura DHT22
Fonte: Elaborado pelo autor.

Em seguida, foi ligado o umidificador dentro da estufa a fim de constatar o declínio de temperatura. A figura 4.2 mostra os resultados obtidos:

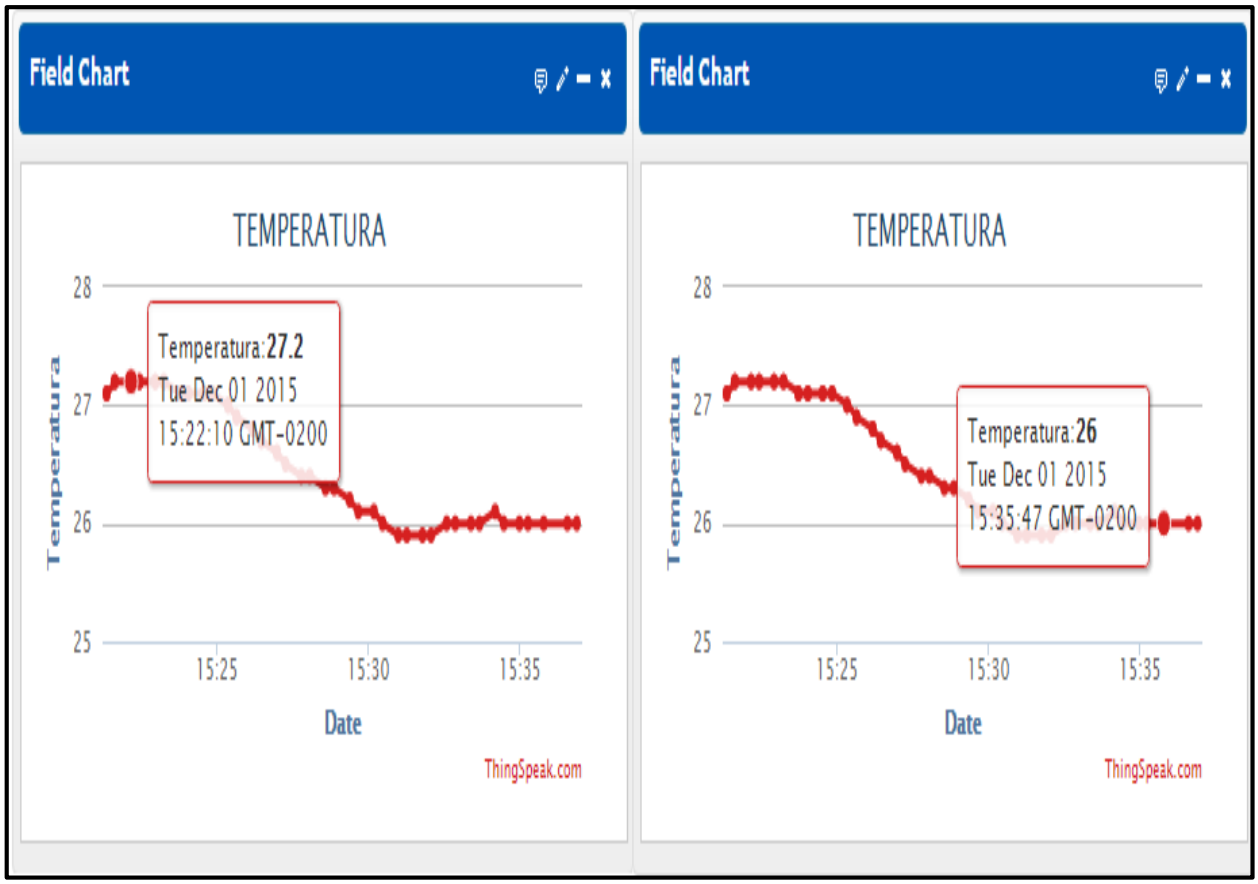


Figura 4.2 - Declínio de Temperatura DHT22
Fonte: Elaborado pelo autor.

Com estes testes, foi possível observar o funcionamento não apenas de leitura do sensor de temperatura, como também o efeito positivo ao ativar o dispositivo de controle da mesma – no caso do teste de declínio, a ventoinha – observando assim uma queda satisfatória da temperatura na estufa.

B. Teste de Umidade do Ar

Para o teste de umidade do ar foi ligado um umidificador próximo ao sensor, simulando a variação da umidade do ar relativa do local. A figura 4.3 abaixo mostra a curva de queda da umidade do ar dentro da estufa, evidenciando assim a efetividade do umidificador em manter a umidade do ar na estufa em parâmetros desejados.

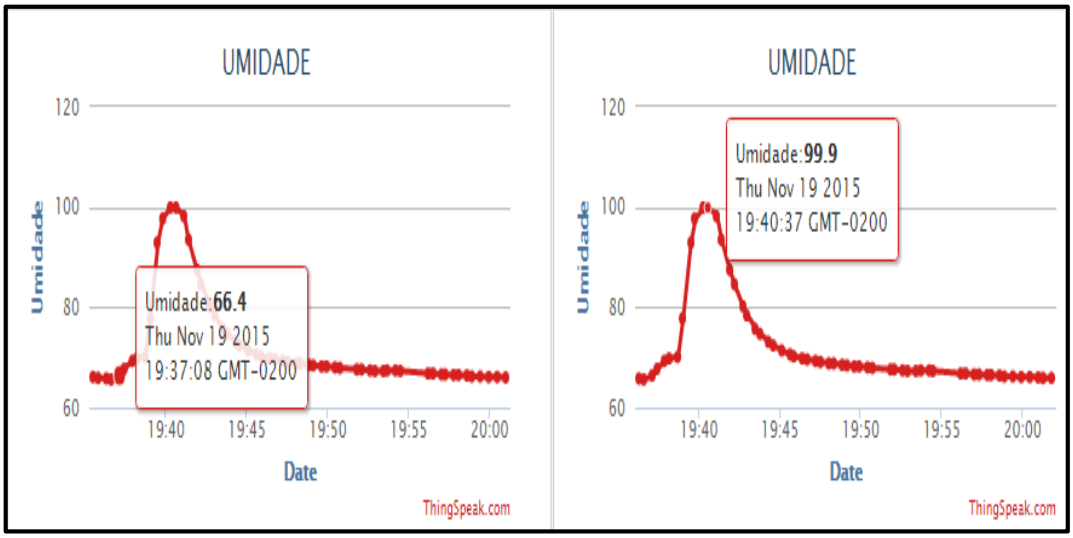


Figura 4.3 - Teste do Sensor DHT22 – Umidade
Fonte: Elaborado pelo autor

Para verificar os dados, foi realizado ainda o comparativo com os dados coletados pelo Instituto Nacional de Meteorologia (INMET) na data dos testes. O sensor apontou um valor de 66%, enquanto o INMET registrou um valor de 60% para toda região de Brasília. Esta diferença é aceitável no contexto do projeto, tendo em vista que o protótipo coleta informações de apenas um ponto local, enquanto os sensores do INMET coletam informações de diferentes regiões da cidade.

4.2 Cenário 2

O teste do sensor de luminosidade foi feito em um cômodo com uma lâmpada de 20W fluorescente. Foram feitos também testes durante um dia ensolarado, com o sensor totalmente coberto e com uma lanterna diretamente apontada no sensor. A tabela 5 mostra os resultados dos testes realizados:

Tabela 5 - Resultados dos testes LDR

Situação	Valor Luminosidade
Sensor Coberto	47,00
Lâmpada 20w ligada	69,20
Dia Ensolarado	96,70
Lanterna Apontada	98,30

Fonte: Elaborado pelo autor

4.3 Cenário 3

Este cenário foi utilizado para o teste de envio de informações com a plataforma *ThingSpeak*. Para tal, foi realizado o teste com a chave de escrita incorreta no código e, conforme esperado, nenhum dado foi recebido. A figura 4.4 mostra o código utilizado no teste, feito com a chave "TESTECENÁRIO3" no lugar da chave correta, com sua devida mensagem de erro.

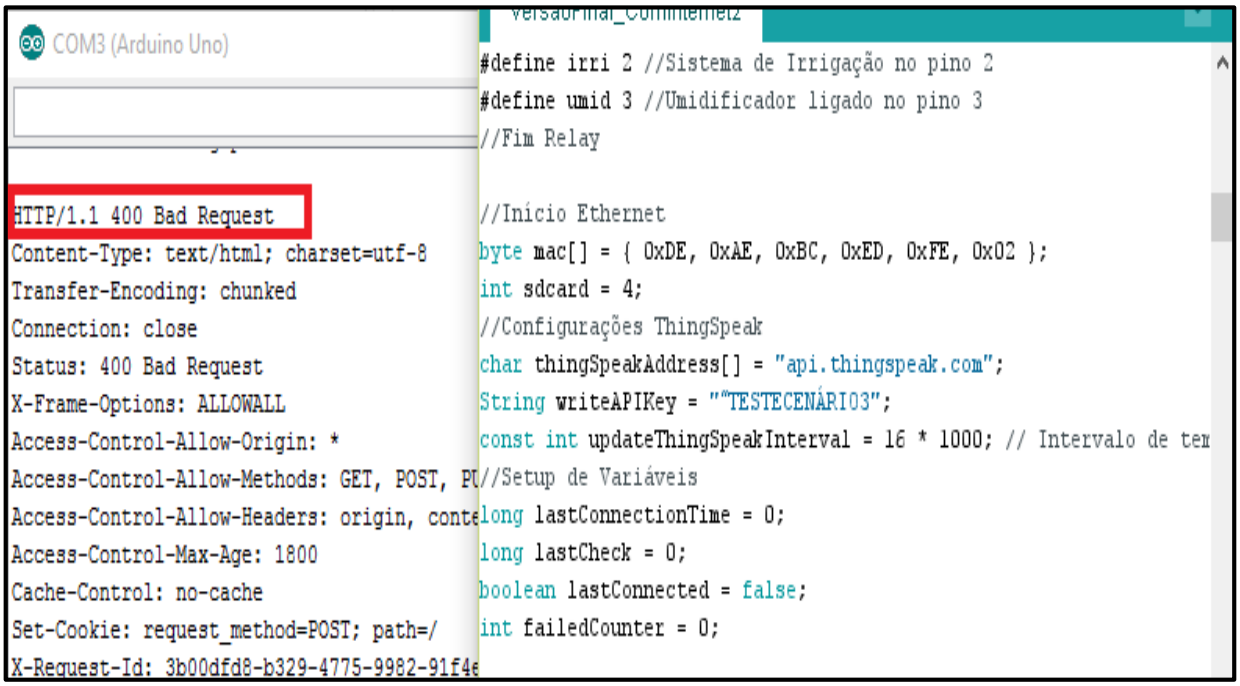


Figura 4.4 - Teste com APIKey errada
Fonte: Elaborado pelo autor

Com a conclusão deste teste, foi possível constatar uma segurança aceitável para a confiabilidade dos dados enviados à plataforma *ThingSpeak*, pois os dados só são incorporados no canal quando enviados com uma senha (*APIKey*) correta e única, que deve ser conhecida apenas pelo próprio usuário.

4.4 Análise de Resultados

Com estes três cenários de testes, foi possível constatar o funcionamento pleno do sistema proposto. Nos testes de temperatura e umidade, o tempo decorrido para o ajuste dos parâmetros da maneira desejada foi satisfatório, pois simulou condições naturais do mundo real, onde aumentos e declínios e temperatura ou umidade ocorrem de maneira gradativa.

O teste do sensor de luminosidade apresentou de maneira satisfatória a empregabilidade do sensor utilizado, pois este conseguiu reconhecer de maneira confiante a presença e ausência de luz no ambiente, possibilitando assim a utilização de uma luz artificial para suprir esta deficiência, quando necessário.

Já o teste da plataforma *ThingSpeak*, provou a confiabilidade desta para o registro dos dados enviados à esta, dando assim uma sensação de segurança ao usuário, pois enquanto sua senha for apenas de conhecimento próprio – como deve ser –, os dados enviados serão necessariamente os dados que espera-se serem enviados.

Por fim, o projeto teve um custo total de R$ 601,75 (seiscentos e um reais e setenta e cinco centavos), onde o maior custo foi a construção da estufa em acrílico, que se equiparou ao valor de todos os materiais necessários para a parte de controle e monitoramento da estufa proposta. É possível optar por um material mais barato na confecção da estufa – como o vidro –, reduzindo ainda mais os seus custos de produção.

Tabela 6 - Custos do Projeto

Produto	**Custo**
Arduino UNO	R$ 55,90
Placa de Fenolíte Furada	R$ 5,00
Módulo Ethernet	R$ 22,90
Fios	R$ 3,80
Módulo de Relés	R$ 29,90
Sensor DHT22	R$ 34,99
Display LCD	R$ 17,99
LDR	R$ 1,40
Caixa em Acrílico	R$ 300,00
Umidificador	R$ 67,00
Soquete E27	R$ 3,75
Lâmpada Fluorescente 20w	R$ 10,99
Bomba de Água	R$ 23,90
Ventoinha 12v	R$ 5,50
Material Completo do Sistema	R$ 18,73
Total	**R$ 601,75**

Fonte: Elaborado pelo autor

5 CONCLUSÃO

Neste capítulo são descritas as conclusões referentes ao desenvolvimento do projeto, suas possíveis aplicações e sugestões de melhorias para o futuro.

5.1 Conclusões do Projeto

A idealização deste projeto teve como ponto principal a criação de um sistema automatizado, de fácil manutenção e que fosse capaz de apresentar uma forma alternativa de cultivo de hortaliças em ambientes residenciais à fim de suprir uma demanda crescente da população brasileira, por um produto confiável e bara. Neste contexto, o objetivo foi alcançado de forma satisfatória na forma da construção do sistema proposto, que não apenas apresenta resultados positivos de desempenho, como permite ao usuário que, ao escolher o tipo de cultivo desejado, consulte as referências de dados do cultivo da Embrapa e conforme estes dados, os valores de temperatura, umidade do ar e luminosidade possam ser ajustados de acordo.

Com a utilização da plataforma Arduino no projeto foi possível a construção de um protótipo que faça o uso de distintos sensores e dispositivos para alcançar o objetivo principal. Esta facilidade de integração de componentes permite que a proposta inicial possa ser expandida de maneira proporcional, possibilitando o uso do sistema proposto, com suas devidas adaptações, para locais de maior escala de produção.

Várias tecnologias são adaptáveis à plataforma Arduino, como o *Shield Ethernet* utilizado no trabalho, que estabelece uma conexão com a Internet, permitindo assim que informações possam ser transmitidas para a plataforma *ThingSpeak*. Outra característica importante que foi considerada na utilização da plataforma Arduino, através de testes de precisão e leitura de dados, foi seu tempo de resposta e eficiência para uma dada operação. Seu funcionamento como todo é visivelmente estável, pois suas falhas geralmente são recorrentes de erros no código, pois todos os seus dispositivos responderam de forma eficaz aos comandos realizados na IDE do Arduino.

5.2 Aplicações do sistema

O sistema proposto foi submetido a diversos testes que tornam possível verificar o funcionamento do Arduino e seus sensores, bem como alterações de leituras quando submetidos

a diferentes aplicações. Nestes há evidências da funcionalidade correta do sistema, atendendo então seu propósito inicial.

O sistema projetado informa os dados de umidade do ar, temperatura e luminosidade coletados por meio de seus sensores não apenas em tempo real para o usuário, mas também pela plataforma *ThingSpeak*, na Internet. Com isso, o usuário tem a possibilidade de verificar o estado de sua estufa a todo momento, além de armazenar o histórico ao longo de dias, meses ou até anos. O projeto pode ser utilizado, portanto, em residências particulares ou em quaisquer ambientes reduzidos voltados para a produção controlada de hortaliças hidropônicas para consumo próprio e/ou comercialização.

5.3 Sugestões para trabalhos futuros

Apesar de o projeto construído ter atendido às especificações estabelecidas, existem uma série de otimizações que podem ser feitas para ampliação do trabalho e melhorias de suas funcionalidades, dentre estas sugestões, pautam-se:

- Utilizar um módulo Wi-Fi para o envio dos dados obtidos, eliminando assim um dos diversos cabos que são utilizados no projeto. O custo será mais elevado, mas caso o usuário veja a necessidade de utilizar o sistema fora de casa, por exemplo, não será necessário o uso de um cabo de rede atravessando o quintal.
- Aplicar o sistema proposto em maior escala, possibilitando assim o cultivo de outros tipos de hortaliças além das folhosas, como hortaliças de frutos ou raízes.
- A solução utilizada no cultivo hidropônico tem um decaimento de nutrientes ao longo do tempo, conforme as plantas vão se alimentando. A instalação de um sensor de PH possibilitaria ao usuário saber exatamente quando trocar a solução, evitando a perda de solução ainda válida, o que ocorre quando esta é trocada baseando-se puramente em conhecimentos técnicos.
- Construção de um *software* integrado capaz de alterar os valores genéricos estabelecidos no projeto por valores específicos de cultivo para cada uma das hortaliças que possam ser cultivadas na estufa, permitindo assim o cultivo de hortaliças mais sensíveis, ou até outros tipos de plantas, como flores.

REFERÊNCIAS BIBLIOGRÁFICAS

ABRANTES, J.; SEIXAS FILHO, J.T. **A viabilidade da agricultura urbana através da hidroponia e do associativismo/cooperativismo**. Rio de Janeiro: Ampub Nobel, 2006.

Adafruit. Disponível em: <http://www.adafruit.com/products/385>. Acesso em: 12 set. 2015.

AGÊNCIA NACIONAL DE VIGILÂNCIA SANITÁRIA (Brasil). Resolução CNNPA nº 12, de 1978. **Diário Oficial [da] República Federativa do Brasil**, Brasília, 24 jul. 1978.

Arduino.cc. (s.d.). Disponível em: <http://arduino.cc/en/Main>. Acesso em: 07 ago. 2015.

BEZERRA, F. **Produção de Mudas de Hortaliças em Ambiente Protegido**. Fortaleza, Embrapa, 2003.

BRAGA, G. N. M. **Hidroponia**. [S.l.]: jun. 2009. Disponível em: < http://agronomiacomgismonti.blogspot.com.br/2009/06/hidroponia.html>. Acesso em: 24 ago. 2015.

BRANDAO FILHO, J.U.T.; CALLEGARI, O. **Cultivo de hortaliças de frutos em solo em ambiente protegido**. Informe Agropecuario, Belo Horizonte, v.20, n.200/201, p.64-68, set./dez. 1999.

CERMEÑO, Z. Serrano**. Construcción de Invernaderos**. Madrid: Grupo Mundi-Prensa, 2005.

DISTEFANO, J. J.; STUBBERUD, A. R.; WILLIAMS, I. J. Schaum's **Outline of Theory and Problems of Feedback and Control Systems**. 2. ed. New York: McGraw-Hill. 1990.

CGE: Centro de Gerenciamento de Emergências. **Umidade Relativa do Ar**, 2015. Disponível em: <http://www.cgesp.org/v3/umidade-relativa-do-ar.jsp>. Acesso em: 19 ago. 2015.

GONDIM, A. (Ed.). **Catálogo Brasileiro de Hortaliças**, Brasília, 2010. Disponível em: <http://uc.sebrae.com.br/sites/default/files/institutional-publication/pdf/catalogo_hortalicas.pdf>. Acesso em: 27 ago. 2015.

GUEDES, Í. M. R. **A agricultura brasileira precisa de uma revolução branca**. Disponível em: <http://scienceblogs.com.br/geofagos/2015/07/a-agricultura-brasileira-precisa-de-uma-revolucao-branca/>. Acesso em: 12 set. 2015.

GUEDES, Í. M. R. **Papel da pesquisa científica na produção de hortaliças no Brasil**. Disponível em: <http://scienceblogs.com.br/geofagos/2015/08/papel-da-pesquisa-cientifica-na-producao-de-hortalicas-no-brasil/>. Acesso em: 12 set. 2015.

HOPKINS, W. G. **Photosynthesis and Respiration**. [s.l.] Infobase Publishing, 2006.

Huinfinito. Disponível em: <http://www.huinfinito.com.br/>. Acesso em: 10 ago. 2015.

JENSEN, M. H.; MALTER, A. J. **Protected Agriculture: A Global Review**. [s.l.] World Bank Publications, 1995.

LUENGO, R. F. A.; CALBO, A. G. **Armazenamento de Hortaliças**. [s.l.] Embrapa, 2001.

MCROBERTS, M. **Arduino Básico**. São Paulo, Novatec, 2011.

NFT: Um tipo de Hidroponia. **Tudo Hidroponia**, [s.d.]. Disponível em: <http://tudohidroponia.net/nft-um-tipo-de-hidroponia/>. Acesso em: 30 set. 2015

OGATA, K. **Engenharia de Controle Moderno**. 4. Ed. São Paulo: Ed. Pearson Prentice Hall, 2003

PINHEIRO, C. **Produção em Ambiente Protegido em São Paulo**: Atuação da Secretaria de Agricultura Favoreceu a Expansão. CASA DA AGRICULTURA: Produção em Ambiente Protegido. Campinas, São Paulo, nº 2, p.40-40, abr./mai./jun. 2011.

SOLERPALAU. Disponível em: <http://www.solerpalau.pt/formacion_01_39.html>. Acesso em: 27 ago. 2015.

VILELA, N. **Produção de Hortaliças no Brasil**. Brasília, Embrapa, 2013.

YARNOLD, S. **Arduino in easy steps**. [s.l.] In Easy Steps, 2015.

APÊNDICE A – Código do Sistema de Controle da Estufa

```
//Projeto Final - Amber Leite de Azevedo Junior
//O código disposto é de um software construído para realizar as leituras dos sensores utilizados na estufa deste trabalho, apresentar estas informações no display, ativar atuadores por meio do funcionamento de relés, além de enviar as informações em forma de String por meio da internet à plataforma ThingSpeak.

//Bibliotecas – Inclusão de Bibliotecas
#include "DHT.h" //Biblioteca DHT
#include <LiquidCrystal.h> //Biblioteca Display
#include <SPI.h> //Biblioteca do Shield Ethernet¹
#include <Ethernet.h> //Biblioteca do Shield Ethernet²

//Display – Ativação do Display de Definição dos Pinos
LiquidCrystal lcd(5, 6, 7, 9, 8, A4); /* Define os pinos que
serão ligados ao LCD */

//Array simbolo grau
byte grau[8] = { B00001100,
                 B00010010,
                 B00010010,
                 B00001100,
                 B00000000,
                 B00000000,
                 B00000000,
                 B00000000,
               };

//Fim do Display

//Sensor de Temperatura e Umidade
#define DHTPIN A1     // Pino em que o sensor de umidade e temperatura está conectado
#define DHTTYPE DHT22   // Tipo do sensor de umidade e temperatura: DHT 22 (AM2302)
// Inicializando o sensor DHT para um Arduino normal de 16 Mhz
DHT dht(DHTPIN, DHTTYPE);
//Fim do Sensor de Temperatura e Umidade

//Sensor de Luminosidade
int ldrPin = A2; //Pino do sensor
int sensorLuz; //Variável para armazenar o valor analógico
float luminosidade; //Varíável para armazenar o valor da tensão após a conversão do valor analógico
//Fim do Sensor de Luminosidade

//Relay
#define vent A3 //Ventilador ligado no pino A3
#define lamp A0 //Lâmpada ligada no pino A0
#define irri 2 //Sistema de Irrigação no pino D2
```

```
#define umid 3 //Umidificador ligado no pino D3
//Fim Relay

//Início Ethernet
byte mac[] = { 0xDE, 0xAE, 0xBC, 0xED, 0xFE, 0x02 };
int sdcard = 4;
//Configurações ThingSpeak
char thingSpeakAddress[] = "api.thingspeak.com";
String writeAPIKey = "----------------";
const int updateThingSpeakInterval = 16 * 1000; // Intervalo de tempo para update no
ThingSpeak (Segundos * 1000 = interval)
//Setup de Variáveis
long lastConnectionTime = 0;
long lastCheck = 0;
boolean lastConnected = false;
int failedCounter = 0;

EthernetClient client; // Iniciar Client Ethernet
//Fim Ethernet

void setup() {
  Serial.begin(9600); //Inicializa a serial
  pinMode(ldrPin, INPUT); //Define o tipo de pino do sensor de luminosidade
  pinMode(vent, OUTPUT);
  pinMode(lamp, OUTPUT);
  pinMode(umid, OUTPUT);
  pinMode(irri, OUTPUT);
  lcd.begin(16, 2); //Inicializa LCD
  lcd.clear(); //Limpa o LCD
  lcd.createChar(0, grau); //Cria o caractere customizado com o símbolo do grau
  dht.begin(); //Inicia o sensor de umidade e temperatura
  inicioEthernet(); // Inicia a Ethernet no Arduino
  inicioIrrigacao();
  //Desativa o SD Card no módulo Ethernet para não dar problema de rede
  pinMode(sdcard, OUTPUT);
  digitalWrite(sdcard, HIGH);
}

void loop() {
  if (client.available()) {
    char c = client.read();
    Serial.print(c);
  }
  long now = millis();
  if (now - lastCheck > updateThingSpeakInterval) { //Caso o (horário atual - o horário
da última checagem) < (intervalo configurado), a rotina é iniciada
    // Sensor de Umidade e Temperatura
    float h = dht.readHumidity(); // Lê o valor da umidade
    float t = dht.readTemperature(); // Lê a temperatura em Celsius
    float f = dht.readTemperature(true);  // Lê a temperatura em Fahrenheit
```

```
// Criação de Strings
char t_buffer[10]; // Buffer da Temperatura
char h_buffer[10]; // Buffer da Umidade
String temp = dtostrf(t, 0, 5, t_buffer); // String da Temperatura
String humid = dtostrf(h, 0, 5, h_buffer); // String da Umidade
// Fim Sensor de Umidade e Temperatura

//Sensor de Luminosidade
sensorLuz = analogRead(ldrPin);
sensorLuz = 1023 - sensorLuz;
luminosidade = map(sensorLuz, 0, 1023, 1000, 0);
luminosidade = (luminosidade / 10);
char l_buffer[10]; // Buffer da Luminosidade
String luz = dtostrf(luminosidade, 0, 5, l_buffer); // String da Luminosidade
//Fim Sensor de Luminosidade

// Sensação Térmica
float hi = dht.computeHeatIndex(f, h);
float hic = (hi - 32) / 1.8; //Conversão para Celsius
char hic_buffer[10]; // Buffer da Sensação Térmica
String ster = dtostrf(hic, 0, 5, hic_buffer); // String da Sensação Térmica
// Fim Sensação Térmica

lastCheck = now; //Após as leituras, indica o tempo delas

// Display Informações
lcd.clear(); // Limpa a tela
lcd.setCursor(0, 0); // Escrevendo na Primeira Linha
lcd.print("Temp : "); // Temperatura
lcd.print(" ");
lcd.setCursor(7, 0);
lcd.print(t, 1);
lcd.setCursor(12, 0);
lcd.write((byte)0); //Mostra o simbolo do grau formado pelo array
if (t > 26) {
  digitalWrite(vent, HIGH);
}
else {
  digitalWrite(vent, LOW);
}

lcd.setCursor(0, 1); // Escrevendo na Segunda Linha
lcd.print("Umid : "); // Umidade
lcd.print(" ");
lcd.setCursor(7, 1);
lcd.print(h, 1);
lcd.setCursor(12, 1);
lcd.print("%");
if (h >= 80) {
  digitalWrite(umid, LOW);
```

```
            }
            else {
              digitalWrite(umid, HIGH);
            }
            delay(2000); //Intervalo para próxima informação
            lcd.clear();

            lcd.setCursor(0, 0); // Escrevendo na Terceira Linha
            lcd.print("Luz : "); // Luminosidade
            lcd.print(" ");
            lcd.setCursor(7, 0);
            if ( luminosidade > 65.00 ) {
              lcd.print("Luz Alta");
              digitalWrite(lamp, LOW);
            }
            else {
              lcd.print("Luz Baixa");
              digitalWrite(lamp, HIGH);
            }

            Serial.println(t);
            Serial.println(luminosidade);
            Serial.println(h);
            //  ThingSpeak
            if (!client.connected() && lastConnected) {
              Serial.println("...desconectado");
              Serial.println();
              client.stop();
            }
            // Update do ThingSpeak
            if (!client.connected() && (millis() - lastConnectionTime >
updateThingSpeakInterval))
            {
              updateThingSpeak("field1="+temp+"&field2="+humid+"&field3="+luz); // Envio
das 3 Strings para o ThingSpeak
            }
            // Checar se o Arduino precisa ser reiniciado (reinicia a Ethernet caso a conexão
falhe mais que 3x)
            if (failedCounter > 3 ) {
              inicioEthernet();
            }
            lastConnected = client.connected();
          }
        }
          void updateThingSpeak(String tsData) { //Fazer update dos dados no ThingSpeak
            if (client.connect(thingSpeakAddress, 80)) {
              client.print("POST /update HTTP/1.1\n");
              client.print("Host: api.thingspeak.com\n");
              client.print("Connection: close\n");
              client.print("X-THINGSPEAKAPIKEY: " + writeAPIKey + "\n");
```

```
            client.print("Content-Type: application/x-www-form-urlencoded\n");
            client.print("Content-Length: ");
            client.print(tsData.length());
            client.print("\n\n");
            client.print(tsData);
            lastConnectionTime = millis();
            if (client.connected()) {
              Serial.println("Conectando ao ThingSpeak...");
              Serial.println();
              failedCounter = 0;
            }
            else {
              failedCounter++;
              Serial.println("A conexão ao ThingSpeak falhou (" + String(failedCounter, DEC)
+ ")");
              Serial.println();
            }
          }
        }

        //Conectar o Arduino na rede e obter um IP usando o DHCP
        void inicioEthernet() {
          client.stop();
          Serial.println("Conectando o Arduino a rede...");
          Serial.println();
          delay(1000);
          if (Ethernet.begin(mac) == 0) {
            Serial.println("DHCP falhou, reinicie o Arduino");
            Serial.println();
          }
          else {
            Serial.println("Conectado a rede usando DHCP");
            Serial.println();
          }
        }
          void inicioIrrigacao() { //Setup de irrigação
            int segundo = 0;
            int minuto = 0;
            int cont = 1;
            double irrigar = (minuto / cont); //Variável para definir momento de irrigação
            static unsigned long ult_tempo = 0;
            int tempo = millis(); //Função millies retorna a quantidade de milissegundos desde
o início da execução do programa.
            if(tempo - ult_tempo >= 1000) {
              ult_tempo = tempo;
              segundo++;
            }
            if(segundo >= 60 ) {
              segundo = 0;
              minuto++;
```

```
   }
    if (irrigar = 1) {
    digitalWrite(irri, HIGH);
    cont++;
  }
  else {
    digitalWrite(irri, LOW);
  }
}
```

ANEXO A – Esquema Elétrico Arduino *UNO*