



**UNIVERSIDADE DO ESTADO DO AMAZONAS - ESCOLA SUPERIOR  
DE TECNOLOGIA**

**RENATO KLEHM RODRIGUES**

**SISTEMA DE MONITORAMENTO E ALERTA PARA CAIXAS TÉRMICAS  
UTILIZADAS NO TRANSPORTE DE PRODUTOS FARMACÊUTICOS**

Manaus  
2022

RENATO KLEHM RODRIGUES

SISTEMA DE MONITORAMENTO E ALERTA PARA CAIXAS TÉRMICAS  
UTILIZADAS NO TRANSPORTE DE PRODUTOS FARMACÊUTICOS

Monografia desenvolvida durante a disciplina de Projeto Final de curso II e apresentada à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas como pré-requisito para a obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Israel Gondres Torné, Dr

Manaus  
2022

**Universidade do Estado do Amazonas – UEA**  
**Escola Superior de Tecnologia - EST**

*Reitor:*

**André Luiz Nunes Zogahib**

*Vice-Reitor:*

**Kátia do Nascimento Couceiro**

*Diretor da Escola Superior de Tecnologia:*

**Ingrid Sammyne Gadelha Figueiredo**

*Coordenador do Curso de Engenharia Elétrica:*

**Israel Gondres Torné**

*Banca Avaliadora composta por:*

**Prof. Israel Gondres Torné, Dr (Orientador)**

**Prof. Fábio de Sousa Cardoso, Dr**

**Prof. Angilberto Muniz Ferreira Sobrinho, Dr**

*Data da defesa: 17/10/2022*

Rodrigues, Renato Klehm

Sistema De Monitoramento E Alerta Para Caixas Térmicas Utilizadas No Transporte De Produtos Farmacêuticos / Filipe Freire Diniz;[orientado por] Prof. Israel Gondres Torné, Dr – Manaus: 2022.

38 p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica).  
Universidade do Estado do Amazonas, 2022.

1. Sensor de umidade do solo. 2. Sensor de chuva. 3. Microcontrolador.  
4. Acionamentos.

I. Torné, Israel Gondres.

**CIP – Catalogação na Publicação**

RENATO KLEHM RODRIGUES

SISTEMA DE MONITORAMENTO E ALERTA PARA CAIXAS TÉRMICAS  
UTILIZADAS NO TRANSPORTE DE PRODUTOS FARMACÊUTICOS

Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro Eletricista.

Nota obtida: 8.7(oito vírgula sete).

Aprovada em: 17/10/2022

Área de concentração: Automação e Telecomunicações

BANCA EXAMINADORA:

Orientador: Prof. Israel Gondres Torné, Dr.

Avaliador 1: Prof. Fábio de Sousa Cardoso, Dr.

Avaliador 2: Prof. Angilberto Muniz Ferreira Sobrinho, Dr.

Manaus  
2022

## **DEDICATÓRIA**

Primeiramente a minha mãe, Cristiane da Silva Klehm, a qual tem me apoiado por todos estes anos com muita dedicação e companheirismo. E, sem a mesma eu não existiria...

Aos meus amigos e familiares, que estiveram presentes no decorrer dessa longa caminhada.

## **AGRADECIMENTOS**

Agradeço primeiramente a minha mãe, por ter sempre me incentivado e motivado a estudar.

Agradeço a minha família, por sempre estarem ao meu lado nesse processo.

Agradeço ao meu professor e orientador Israel Gondres Torné, por toda a sua ajuda durante a minha etapa de desenvolvimento do trabalho.

E também agradeço aos meus amigos e professores que me acompanharam durante todo este período de graduação, e que de uma forma ou outra, sempre contribuíram para que eu chegasse até aqui.

## RESUMO

Neste projeto foi desenvolvido o protótipo de um sistema de monitoramento e alerta em tempo real, aplicando o conceito de iot, para uma caixa térmica, que visa garantir que a cadeia de frio seja mantida durante a etapa de transporte que utilizam as caixas térmicas. Portanto, é um projeto bastante funcional, de baixo custo e de fácil manuseio, podendo também ser aplicado em outros sistemas que necessitam de monitoramento em tempo real. Este sistema possui as funções de: verificar a temperatura e umidade da caixa térmica, sua localização e emissão de alerta sonoro. Caso a temperatura da caixa térmica fique fora da faixa ideal, o sistema detecta e emite um alerta sonoro na caixa térmica e envia a temperatura, umidade e a localização dela para um site de monitoramento, permitindo que ações imediatas sejam realizadas visando preservar a carga transportada. É utilizado um circuito com o microcontrolador Arduino para automatizar este sistema, em conjunto com sensores e módulos. Realizou-se testes com o protótipo desenvolvido e os resultados obtidos mostraram-se bastante satisfatórios, atendendo aos objetivos deste trabalho. Portanto, conclui-se neste projeto que o protótipo funcionou adequadamente, atendendo aos requisitos essenciais de um sistema de monitoramento em tempo real. Com isto, aumentando a confiança no transporte de produtos farmacêuticos, garantido sua eficácia.

**Palavras-chave:** monitoramento em tempo real, sensor de temperatura e umidade, Arduino, IoT, caixas térmicas.

## **ABSTRACT**

This project was developed to be designed in a real-time monitoring and alert system, applying the iot concept, for a cooler, which aims to ensure that a cold chain is maintained during a transport stage that uses as coolers. Rather, it is quite an easy-to-use design and other functional systems, which can be used for real-time monitoring as well. This system has the location functions: temperature and temperature control of the cooler, its sound alert emission. If the box temperature monitoring falls into the ideal range, the system detects and emits a sound in the box and sends the temperature, humidity and protection to the actions of an alert location, preserving the immediate actions taken for the cargo transport. A circuit with the Arduino microcontroller is used to automate this system, together with sensors and modules. The results obtained were satisfactory and the tests were carried out with the proposed objective. Therefore, it is concluded that the project worked correctly, meeting the essential requirements of a real-time monitoring system. With this, increasing confidence in the transport of pharmaceutical products, guaranteeing their diligence.

**Keywords:** real-time monitoring, temperature and humidity sensor, Arduino, IoT, thermal boxes.



## LISTA DE FIGURAS

Figura 1 - Caixas térmicas. ....	15
Figura 2 - Módulo GSM GPRS. ....	18
Figura 3 - Módulo conectado a antena. ....	19
Figura 4 - Arduino Uno R3.....	21
Figura 5 - Ambiente de programação do Arduino.....	22
Figura 6 - Sensor DHT11 pinout. ....	23
Figura 7 - Módulo GPS NEO-6M pinout. ....	25
Figura 8 -Módulo I2C pinout.....	27
Figura 9 - Endereços do módulo I2C.....	27
Figura 10 - Display LCD 16x2 pinout.....	29
Figura 11 - Buzzer ativo pinout.....	30
Figura 12 - Diagrama em blocos do sistema. ....	33
Figura 13 - Conexão do sensor de temperatura no circuito.....	35
Figura 14 - Conexão do Buzzer no circuito.....	35
Figura 15 - Conexão do Display no circuito.....	36
Figura 16 - Conexão do módulo GPS no circuito. ....	37
Figura 17 - Conexão do módulo GSM GPRS no circuito. ....	38
Figura 18 - Criação da estrutura usuário no banco de dados.....	39
Figura 19 - Criação da estrutura GPRS Arduino no banco de dados. ....	39
Figura 20 – Comparação 1 de medição de temperatura do sistema e de um medidor de referência. ....	40
Figura 21 - Comparação 2 de medição de temperatura do sistema e de um medidor de referência. ....	40
Figura 22 - Informações do sistema recebidas no banco de dados.....	41
Figura 23 - Sistema montado no protoboard. ....	42
Figura 24 - Esquema em blocos do circuito. ....	42
Figura 25 - Tela de login do website. ....	43
Figura 26 - Tela de monitoramento do website.....	43
Figura 27 - Banco de dados preenchido com dados do sistema. ....	44
Figura 28 - Gráfico de monitoramento de temperatura e umidade.....	44
Figura 29 - Link do Google Maps para rastrear a caixa térmica. ....	44

## LISTA DE TABELAS

Tabela 1 - Especificações técnicas do módulo GSM GPRS SIM800L.....	19
Tabela 2 - Especificações técnicas do sensor DHT11.....	24
Tabela 3 – Especificações técnicas módulo GPS NEO-6M.....	25
Tabela 4 - Especificações técnicas do módulo I2C.....	27
Tabela 5 - Conexão dos componentes nas portas do Arduino.....	38

# SUMÁRIO

## Sumário

<b>1. INTRODUÇÃO .....</b>	<b>13</b>
1.1. PROBLEMÁTICA.....	15
1.2. OBJETIVOS.....	16
1.2.1. Objetivo Geral.....	16
1.2.2. Objetivos Específicos .....	16
<b>2. REFERENCIAL TEÓRICO .....</b>	<b>16</b>
2.1. REDES GSM GPRS.....	16
2.1.1. Módulo GSM GPRS SIM800L.....	17
2.2. IOT – INTERNET DAS COISAS.....	20
2.3. MICROCONTROLADORES .....	20
2.3.1. Arduino Uno .....	20
2.4. SENSORES DE TEMPERATURA E UMIDADE.....	22
2.4.1. Sensor de temperatura e umidade DHT11 .....	23
2.5. GPS .....	24
2.5.1. Módulo GPS NEO6M.....	24
2.6. DISPLAYS DIGITAIS.....	26
2.6.1. Módulo serial I2C.....	26
2.6.2. Display LCD 16x2.....	28
2.7. ALERTA SONORO AUTOMATIZADO .....	29
2.7.1. Buzzer Ativo.....	29
2.8. DESENVOLVIMENTO WEB.....	30
2.8.1. Front-end.....	30
2.8.2. Back-end.....	30
2.8.3. Banco de dados .....	31
<b>3. METODOLOGIA.....</b>	<b>32</b>
3.1. DIAGRAMA EM BLOCOS DO SISTEMA .....	32
3.2. DEFINIÇÃO DA LÓGICA PARA O CÓDIGO-FONTE DO ARDUINO.....	33
3.3. DEFINIÇÕES PARA OS CÓDIGOS-FONTES DO SITE.....	34
<b>4. IMPLEMENTAÇÃO .....</b>	<b>34</b>
4.1. MATERIAIS UTILIZADOS.....	34
4.2. MONTAGEM DO CIRCUITO .....	34
4.2.1. Sensor de temperatura e umidade .....	34
4.2.2. Buzzer ativo .....	35
4.2.3. Display LCD 16x2 com módulo I2C.....	36
4.2.4. Módulo GPS.....	36

4.2.5. Módulo GSM GPRS .....	37
4.3. PROGRAMAÇÃO DO ARDUINO .....	38
4.4. CRIAÇÃO DO WEBSITE .....	39
5.1. TESTE DO SENSOR E DISPLAY .....	40
5.2. TESTE DO BUZZER.....	41
5.3. TESTE DO MÓDULO GPS.....	41
5.4. TESTE DO MÓDULO GSM GPRS E DA COMUNICAÇÃO COM O BANCO DE DADOS.....	41
5.5. CIRCUITO MONTADO.....	41
5.6. WEBSITE E MONITORAMENTO .....	43
<b>CONCLUSÃO .....</b>	<b>45</b>
<b>REFERÊNCIAS BIBLIOGRÁFICAS .....</b>	<b>46</b>
<b>APÊNDICE A - CÓDIGO DO ARDUINO .....</b>	<b>49</b>

## 1. INTRODUÇÃO

A preocupação em manter os medicamentos com a mesma qualidade com que foi fabricado até chegar às mãos dos pacientes, a ANVISA promulgou portarias, RDC's (Resolução da Diretoria Colegiada) e leis para garantir o controle sanitário na produção, distribuição, transporte e armazenagem dos produtos farmacêuticos (MAIO; SILVA, 2014).

Além da legislação para que deve ser seguida pelos fabricantes e transportadores determinados pela ANVISA outro fator importante que necessita de uma atenção especial é a embalagem do produto, o acondicionamento, a quantidade a ser transportada, o meio de transporte utilizado para garantir com segurança a integridade do medicamento (MAIO; SILVA, 2014).

Dentro desse contexto de garantir a integridade do produto a ser transportado, surge o conceito de cadeia de frio.

A cadeia de frio (também conhecida por rede de frio e rede ou cadeia frigorífica) é todo o processo desde a concepção, passando pelo armazenamento, até ao transporte do produto, preservando todas as condições de refrigeração e garantindo a sua conservação (Tanabe et al, 2000).

Resumidamente a cadeia de frio é um ambiente de temperatura propícia que tem que ser mantida desde a saída da fábrica, no transporte e na chegada ao destino final. Sendo de extrema importância que a cadeia não seja quebrada para garantir a qualidade do medicamento ou imunizante recebido pelo paciente. Sendo as seguintes da cadeia de frio:

- Aquisição
- Recebimento
- Transporte
- Armazenamento
- Conservação
- Manuseio
- Distribuição

Sendo o foco deste trabalho garantir que a Cadeia de Frio seja mantida durante a etapa de transporte.

Alguns dos principais itens que demandam alto rigor quanto ao monitoramento de

temperatura na cadeia de frio são:

- Medicamentos;
- Vacinas;
- Suprimentos biológicos e farmacêuticos;
- Hemoderivados;
- Hemocomponentes;
- Amostras biológicas;
- Insulinas;
- Produtos veterinários.

As faixas de temperatura para a conservação de termos sensíveis variam conforme as características de cada produto e instruções do fabricante. As principais faixas aplicadas são:

- Temperatura ambiente: 15°C a 30°C;
- Resfriado: 8°C a 15°C;
- Refrigerado: 2°C a 8°C;
- Congelado: -10°C a -20°C;
- Ultracongelado: até -80°C.

Dessa forma, este trabalho possui o intuito de desenvolver um sistema que realize o monitoramento da temperatura e umidade das caixas térmicas utilizadas no transporte dos produtos, emita um alerta sonoro na própria caixa, para que se alguém estiver próximo possa tomar as devidas providências e também informe a uma central de monitoramento a localização e as condições de temperatura e umidade na caixa para que eles possam agir caso ocorra algum problema no transporte.

Para desenvolver este sistema, será utilizado o microcontrolador Arduino em conjunto com um sensor de temperatura e umidade, um módulo gps para obter a localização da caixa, um módulo GSM GPRS para realizar a comunicação da caixa com a central de monitoramento, um buzzer para emitir um alerta sonoro na caixa e um display LCD para visualizar as informações de temperatura e umidade. O sistema será então integrado a caixa térmica na sua fabricação, na Figura 1, tem-se o exemplo de uma caixa térmica convencional que pode ser utilizada para inserir o projeto.

Figura 1 - Caixas térmicas.



Fonte: Incoterminas (2021).

### ***1.1.PROBLEMÁTICA***

Necessidade de monitorar as condições de transporte relacionadas às especificações de temperatura, acondicionamento, armazenagem e umidade do medicamento utilizando instrumentos calibrados. sendo necessário fazer esse monitoramento em tempo real para que ações possam ser realizadas para salvar a carga transportada, garantindo que a cadeia de frio não foi comprometida.

No Brasil, na Resolução da diretoria colegiada - RDC nº 430, a Anvisa faz a exposição dos requisitos de boas práticas de distribuição, armazenagem e transporte de medicamentos determinado que são obrigações das empresas que realizam o transporte de medicamentos realizar:

- I. Dispor do manifesto de carga transportada com a previsão de desembarque a bordo do veículo transportador;
- II. Monitorar as condições de transporte relacionadas às especificações de temperatura, acondicionamento, armazenagem e umidade do medicamento utilizando instrumentos calibrados;
- III. Aplicar os sistemas passivos ou ativos de controle de temperatura e umidade que sejam necessários à manutenção das condições requeridas pelo registro sanitário ou outras especificações aplicáveis;
- IV. Fornecer ao contratante todos os dados relativos às condições de conservação durante o transporte, bem como durante a armazenagem em trânsito;

- V. Prover acesso restrito aos medicamentos;
- VI. Receber e entregar medicamentos somente às empresas devidamente autorizadas e licenciadas para as atividades relacionadas.

Dessa forma, seguindo as orientações descritas pela ANVISA, este trabalho irá focar em resolver a problemática exposta nos itens II e IV na etapa de transporte nas caixas térmicas.

## ***1.2.OBJETIVOS***

### **1.2.1. Objetivo Geral**

Desenvolver uma aplicação de IoT responsável por realizar o monitoramento e alerta de temperatura, umidade e localização em tempo real visando garantir que a cadeia de frio seja mantida.

### **1.2.2. Objetivos Específicos**

Visando atingir o objetivo principal, alguns objetivos específicos são necessários:

- Desenvolver um website para realizar o monitoramento em tempo real das caixas térmicas.
- Desenvolver um protótipo que realizará a coleta dos dados da caixa térmica e envie para o servidor de monitoramento.
- Armazenar os dados recebidos para possibilitar futuras análises caso seja necessário validar a eficácia do produto transportado.

## **2. REFERENCIAL TEÓRICO**

Tendo a problemática e a nossa hipótese de solução para este, faz-se necessário conhecer quais tecnologias, conceitos e ferramentas serão necessárias para realizar o desenvolvimento deste projeto, será exposto então os seguintes tópicos: Redes GSM GPRS, Internet das Coisas, microcontroladores, sensores de temperatura e umidade, GPS, displays digitais e alerta sonoro automatizado.

### ***2.1.REDES GSM GPRS***

GSM, que significa Global System for Mobile Communications, é um sistema aberto, não proprietário. Seu grande diferencial é a capacidade de funcionamento em mais de 170 países. Ele consegue atingir áreas que a cobertura terrestre não alcança (BRAGHETTO *et al*,



2003).

O GSM difere da primeira geração de sistemas sem fio no uso de tecnologia digital e métodos de transmissão TDM. A voz é codificada digitalmente via um único codificador, que emula as características da fala humana. (BRAGHETTO *et al*, 2003).

O serviço do GSM envolve alta velocidade, serviços de dados multimídia em sistemas sem fio (BRAGHETTO *et al*, 2003).

O GPRS (General Packet Radio Service - Serviço de Rádio Geral por Pacotes) é um novo serviço “novoice” que permite que a informação em forma de dados seja emitida e recebida através de uma rede de telefonia móvel. Ele complementa os atuais serviços de comutação por circuitos GSM (Global System for Mobile) e os serviços de envio de mensagens via rede celular denominado de SMS (Short Message System) (BRAGHETTO *et al*, 2003).

As redes GPRS foram desenvolvidas para suportar os serviços de dados, pois as mesmas foram criadas baseadas em transmissão por comutação de pacotes, diferentemente das GSM que ainda utilizam a comutação por circuitos. Na comutação por pacotes, como visto anteriormente, utiliza de forma mais eficiente a banda devido a transmissão do tráfego ser em rajadas o que é a característica dos serviços de dados (BRAGHETTO *et al*, 2003).

O GSM e o GPRS compartilham uma única base dinâmica e flexível, com várias características semelhantes entre si, como bandas, frequências, estrutura de frames e técnicas de modulação. No entanto, a cobrança pelo uso de GPRS é feita por quantidade de dados (Kbits) transmitidos enquanto no GSM é feita por tempo de conexão (segundos) (BRAGHETTO *et al*, 2003).

#### 2.1.1. Módulo GSM GPRS SIM800L

O Módulo GSM GPRS SIM800L é utilizado para integração de projetos de microcontroladores com redes de comunicação GSM e GPRS, através do Chip SIM800L, ele funciona em quad-band operando nas frequências GSM850MHz, EGSM900MHz, DCS1800MHz e PCS1900MHz.

O módulo opera com tensões de 3,7 a 5V e demanda uma alta potência, em decorrência disso é necessário utilizar uma fonte própria de alimentação, pois o Arduino não conseguirá fornecer energia suficiente para o módulo e os demais componentes. Possui interface serial USB-TTL permitindo que o módulo seja conectado diretamente a um microcontrolador, a operação do módulo com o microcontrolador é feito através de comunicação Serial UART.

Através de comandos AT é possível configurar a rede GPRS para envio e recebimento de SMS, chamadas através de um cartão SIM, conexão com a internet, dessa forma, possibilitando o envio de dados a longa distância, abrindo uma enorme possibilidade de projetos referentes a transmissão de dados e voz à distância.

O módulo funciona em conjunto com uma antena helicoidal que deve ser conectado na placa, também deve ser conectado um sim card no soquete de cartão SIM. Quando o módulo estiver em funcionamento um led indicativo acenderá.

Na Figura 2, é visto que o módulo possui 7 pinos de conexão, sendo eles:

Pino 1 – 5V/4V, responsável pela alimentação do módulo.

Pino 2 – GND, responsável pela conexão do terra.

Pino 3 – VDD, responsável pela alimentação do módulo.

Pino 4 – SIM\_TXD, responsável por enviar os dados do módulo.

Pino 5 -SIM\_RXD, responsável pelo recebimento de dados pelo módulo.

Pino 6 – GND, responsável pela conexão do terra.

Pino 7 – RST, responsável por resetar o módulo.

Figura 2 - Módulo GSM GPRS.



Fonte: Acoptex (2018).

Na Figura 3 tem-se o módulo conectado a antena, responsável pela comunicação com as estações de rádio base.

Figura 3 - Módulo conectado a antena.



Fonte: Acoptex (2018).

Na tabela 1, tem-se algumas especificações técnicas do módulo, pode-se ser visto que o módulo funciona em quad-band, possui uma faixa operação de temperatura ideal para o projeto, funciona nas tensões de 3,7V a 5,0V.

Tabela 1 - Especificações técnicas do módulo GSM GPRS SIM800L.

<b>Especificações Técnicas</b>	
<b>CI</b>	<b>SIM800L</b>
<b>Frequências</b>	EGSM900, DCS1800, GSM850, PCS1900
<b>Conector</b>	Para antena externa U.FL
<b>GPRS data downlink</b>	85.6kbps (máx.)
<b>GPRS data uplink</b>	85.6 kbps (máx.)
<b>Suporte</b>	PAP (password authentication protocol) para conexões PPP
<b>Protocolo</b>	TCP/IP embutido
<b>Serial</b>	1200bps à 115.200bps
<b>Slot</b>	MicroSIM
<b>Antena</b>	SMA e pigtail
<b>Temperatura de operação</b>	-40 - +85°C
<b>Tensão de operação</b>	3,7V a 5,0V
<b>Dimensões</b>	40 x 27.5 x 5.5MM

Fonte: Próprio Autor.

## ***2.2.IOT – INTERNET DAS COISAS***

A Internet das Coisas, em poucas palavras, nada mais é que uma extensão da Internet atual, que proporciona aos objetos do dia-a-dia (quaisquer que sejam), mas com capacidade computacional e de comunicação, se conectarem à Internet. A conexão com a rede mundial de computadores viabilizará, primeiro, controlar remotamente os objetos e, segundo permitir que os próprios objetos sejam acessados como provedores de serviços. Estas novas habilidades, dos objetos comuns, geram um grande número de oportunidades tanto no âmbito acadêmico quanto no industrial. Todavia, estas possibilidades apresentam riscos e acarretam amplos desafios técnicos e sociais (SANTOS *et al*, 2016).

Novos desafios surgem enquanto são criadas novas aplicações para IoT. Os dados providos pelos objetos agora podem apresentar imperfeições (calibragem do sensor), inconsistências (fora de ordem, outliers) e serem de diferentes tipos (gerados por pessoas, sensores físicos, fusão de dados). Assim, as aplicações e algoritmos devem ser capazes de lidar com esses desafios sobre os dados. Outro exemplo diz respeito ao nível de confiança sobre os dados obtidos dos dispositivos da IoT e como/onde pode-se empregar esses dados em determinados cenários. Deste modo, os desafios impostos por essas novas aplicações devem ser explorados e soluções devem ser propostas para que a IoT contemple as expectativas em um futuro próximo (SANTOS *et al*, 2016).

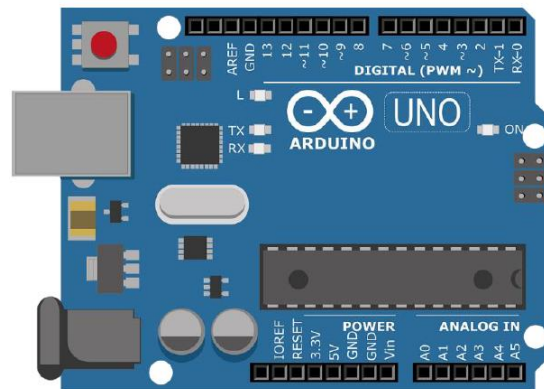
## ***2.3. MICROCONTROLADORES***

Um microcontrolador é um dispositivo composto de um processador, memórias e periféricos de entrada e saída. É importante salientar a diferença entre um microcontrolador e um microprocessador. Enquanto o microcontrolador tem autonomia para funcionar, precisando apenas de um programa e dispositivos de entrada e saída, o microprocessador é apenas a unidade central de processamento, sendo necessário a conexão com memórias, barramentos de entrada e saída etc (ZELENOVSKY, MENDONÇA, 2005).

### **2.3.1. Arduino Uno**

A placa **Arduino** é muito parecida com um computador de pequeno porte, sendo composta pelo microcontrolador, memória RAM, memória secundária (memória flash), clock e comunicação USB entre outras funcionalidades. Na Figura 4 temos o modelo mais popular dos Arduinos que é o Uno R3. (OLIVEIRA *et al*,2018).

Figura 4 - Arduino Uno R3.



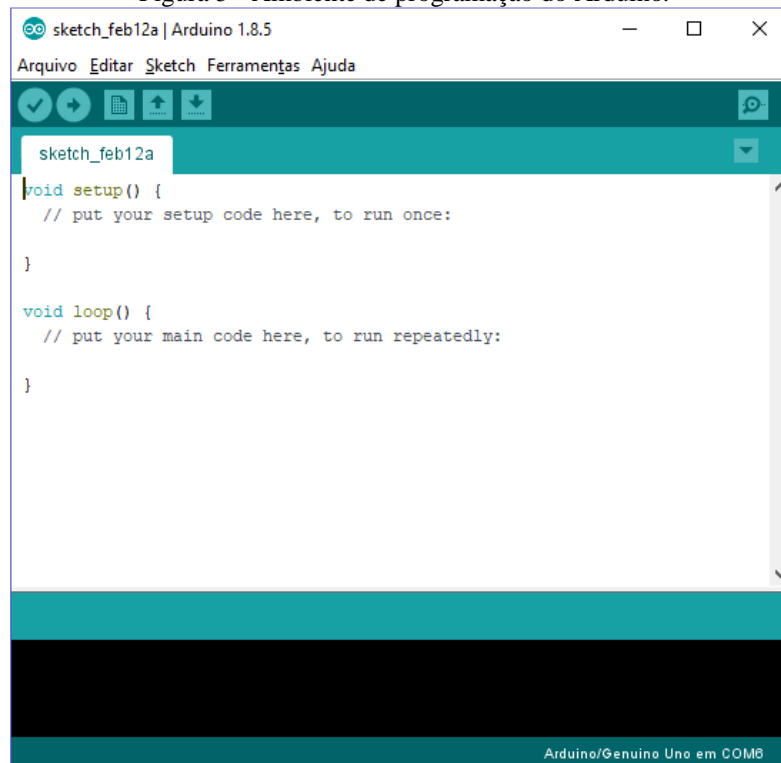
Fonte: Oliveira (2018, p.14).

O Arduino Uno R3 apresenta 14 pinos que podem ser utilizados como entradas ou saídas digitais (pinos 0 a 13), sendo que os pinos 3, 5, 6, 9, 10 e 11 também podem utilizar Pulse Width Modulation (PWM) para gerar um conjunto de valores inteiros entre 0 e 255. Os pinos de A0 a A5 correspondem às entradas analógicas, que recebem uma tensão entre 0 e 5V e o produzem em uma escala de 0 a 1023. Também temos os pinos 3,3V, 5V e GND (Terra) permitem alimentar os componentes dos circuitos conectados ao Arduino. Possui um microprocessador ATmega328, com uma memória RAM de 2KB, memória Flash de 32KB e clock de 16MHz. (OLIVEIRA *et al*,2018).

O ambiente de desenvolvimento do Arduino pode ser gratuitamente baixado do site [www.arduino.cc](http://www.arduino.cc). Neste site e em muitos outros, também estão disponíveis as instruções para realizar a instalação em diversos sistemas operacionais, além de fóruns para tirar dúvidas e obter maiores informações. (OLIVEIRA *et al*,2018).

O ambiente de programação (Figura 5) é muito simples e intuitivo. Um programa, que no Arduino é chamado de **sketch**, apresenta duas funções básicas: **setup ()** e **loop ()**.(OLIVEIRA *et al*,2018).

Figura 5 - Ambiente de programação do Arduino.



Fonte: Oliveira (2018, p.15).

A função `setup ()` deverá conter o código que irá executar apenas uma vez, quando o sketch iniciar. Normalmente colocamos nesta função as definições iniciais do programa. (OLIVEIRA *et al*,2018).

A função `loop ()` irá executar continuamente as instruções que estão lá até que outro sketch seja carregado na memória “flash” do Arduino. É importante notar que no Arduino é possível armazenar e executar um sketch por vez, desta forma, sempre quando transferimos um sketch esse irá substituir o programa que estava anteriormente carregado na memória. Também observe que como o sketch fica armazenado na memória “flash”, que é permanente, mesmo quando desligamos o Arduino, o programa continua armazenado e irá entrar novamente em execução quando o Arduino for ligado novamente. (OLIVEIRA *et al*,2018).

#### **2.4.SENSORES DE TEMPERATURA E UMIDADE**

A medição da temperatura pode ser realizada através dos seguintes sensores elétricos / eletrônicos: resistências metálicas (RTD), termopares, termistores (NTC e PTC), sensores de silício, sensores de radiação. Em geral a maior dificuldade consiste em decidir entre RTDs e termopares, ou entre NTCs e sensores de silício. Os sensores de radiação têm aplicações mais definidas e por isso não entram em comparação com outros sensores no momento de escolha (COCOTA, 2005).

A umidade pode ser descrita como: Umidade relativa: pela razão de água em relação a pressão de saturação; Umidade específica: pela massa de água por unidade de massa de ar seco; Razão de volume: pela parte de vapor de água por milhões de partes de ar. Pelo método clássico, a umidade relativa é obtida nas cartas psicométricas e definida pela leitura de dois termômetros. Realizando a medida da temperatura de bulbo seco e a de bulbo úmido, encontra-se através de fórmulas a umidade relativa (COCOTA, 2005).

#### 2.4.1. Sensor de temperatura e umidade DHT11

O sensor de temperatura e umidade DHT11, é um sensor que permite realizar leituras de temperaturas que variam de 0 a 50 Celsius e de umidade que varia de 20 a 90%.

O elemento do sensor de temperatura é um termistor do tipo NTC e o sensor de umidade é do tipo HR202, o circuito interno faz a leitura dos sensores e envia os dados através de um sinal serial de uma via.

Na Figura 6, é visto que o módulo possui 4 pinos de conexão, sendo eles:

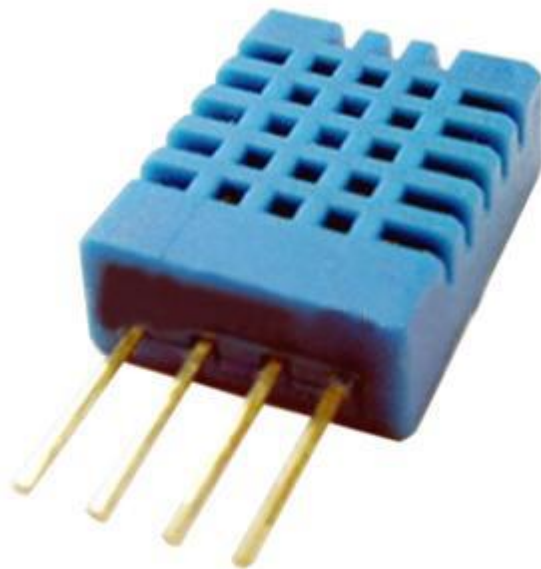
Pino 1 – VCC, responsável pela alimentação do sensor.

Pino 2 – Data, responsável por enviar os dados medidos pelo sensor.

Pino 3 – NC, pino que não é utilizado.

Pino 4 – GND, responsável pela conexão da terra.

Figura 6 - Sensor DHT11 pinout.



Fonte: Datasheet Mouser Electronics(2012).

Na Tabela 2 estão presentes algumas importantes especificações técnicas do sensor, que devem ser observadas para o desenvolvimento do projeto.

Tabela 2 - Especificações técnicas do sensor DHT11.

<b>Especificações Técnicas</b>	
<b>Modelo</b>	DHT11
<b>Faixa de medição de umidade</b>	20 a 90% UR
<b>Faixa de medição de temperatura</b>	0° a 50°C
<b>Alimentação</b>	3-5VDC (5,5VDC máximo)
<b>Corrente</b>	200uA a 500mA, em stand by de 100uA a 150 uA
<b>Precisão de umidade de medição</b>	± 5,0% UR
<b>Precisão de medição de temperatura</b>	± 2.0 °C
<b>Tempo de resposta</b>	2s
<b>Dimensões</b>	23 x 12 x 5mm (incluindo terminais)

Fonte: Próprio autor.

## **2.5.GPS**

O sistema de Posicionamento Global, conhecido por GPS (Global Positioning System) ou NAVSTAR-GPS (Navigation Satellite with Time And Ranging), é um sistema de rádio navegação desenvolvido pelo Departamento de Defesa dos Estados Unidos da América, visando inicialmente ser o principal sistema de navegação do exército americano (CARVALHO; ARAÚJO, 2009).

Esse sistema está programado para nos fornecer coordenadas bi ou tridimensionais de pontos no terreno, bem como a velocidade e direção com que nos deslocamos entre pontos. O GPS tem como objetivo auxiliar as atividades de navegação e realização de levantamentos geodésicos e topográficos. O sistema opera ininterruptamente, 24 horas por dia, independentemente das condições meteorológicas (muito embora essas condições possam provocar algum tipo de interferência na qualidade dos resultados do levantamento) (CARVALHO; ARAÚJO, 2009).

### **2.5.1. Módulo GPS NEO6M**

O módulo GPS NEO-6M é capaz de informar a localização exata do objeto em que o módulo esteja instalado, podendo informar dados referentes a latitude e longitude, data, hora e velocidade de deslocamento, ele funciona com uma tensão de trabalho de 3,3V a 5V

O módulo funciona em conjunto com uma antena, responsável por realizar a comunicação com os satélites.



Na Figura 7, é visto que o módulo possui 4 pinos de conexão, sendo eles:

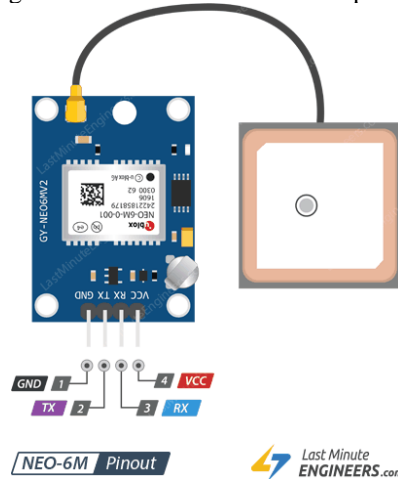
Pino 1 – GND, responsável pela conexão do terra.

Pino 2 – TX, responsável pelo envio de informações pelo módulo.

Pino 3 – RX, responsável pelo recebimento de informações pelo módulo.

Pino 4 – VCC, responsável pela alimentação do sensor.

Figura 7 - Módulo GPS NEO-6M pinout.



Fonte: Last Minute Engineers (2018).

Na Tabela 3 estão presentes algumas informações técnicas do módulo, importantes para o desenvolvimento do projeto.

Tabela 3 – Especificações técnicas módulo GPS NEO-6M

<b>Especificações Técnicas</b>	
<b>Modelo</b>	GPS6MV2
<b>Tensão de trabalho</b>	3,3 a 5V DC
<b>Informação</b>	Interface Serial de 3.3V
<b>Taxa de transmissão padrão</b>	9600
<b>Dimensões do módulo (CxLxA)</b>	30x23x4mm
<b>Dimensões da antena (CxLxA)</b>	25x25x8mm

Fonte: Próprio Autor.

## **2.6.DISPLAYS DIGITAIS**

Displays são dispositivos utilizados para apresentar ao usuário informações obtidas por determinado circuito. No universo do Arduino, são amplamente utilizados em projetos onde se faz necessária a leitura dos dados obtidos por um sensor ou qualquer outro sistema onde o usuário precisa visualizar as informações processadas pelo dispositivo. Se tratando do Arduino, existem basicamente três modelos de displays, que são utilizados com maior frequência pelos projetistas. São eles: Display LCD, Display OLED e display de 7segmentos (Jornal de Brasília, 2021).

### **2.6.1. Módulo serial I2C**

O protocolo I2C foi desenvolvido pela Philips visando realizar a conexão de diversos dispositivos utilizando apenas as duas linhas de dados (SDA e SCL) Serial Data e Serial Clock. A ideia é definir um endereço hexadecimal para cada dispositivo e no momento da comunicação somente o dispositivo solicitado responderá, dessa forma cada dispositivo receberá um endereço e a partir dele será feito o controle individual dos dispositivos. Na figura x é possível ver os pinos de seleção de endereço, que devem ser soldados para realizar a mudança de endereço, por padrão o módulo vem com o endereço 0x27, na tabela X é possível ver as demais configurações de endereço.

Para se utilizar um display lcd é necessário a conexão de muitos pinos, dessa forma em algumas situações pode-se não ter tantos pinos disponíveis, surge então o módulo serial I2C que serve para controlar o display utilizando apenas 2 pinos.

Na Figura 8, é visto que o módulo é composto por 16 pinos laterais que devem ser conectados paralelamente aos pinos do display e por outros 4 pinos de alimentação e controle, sendo eles:

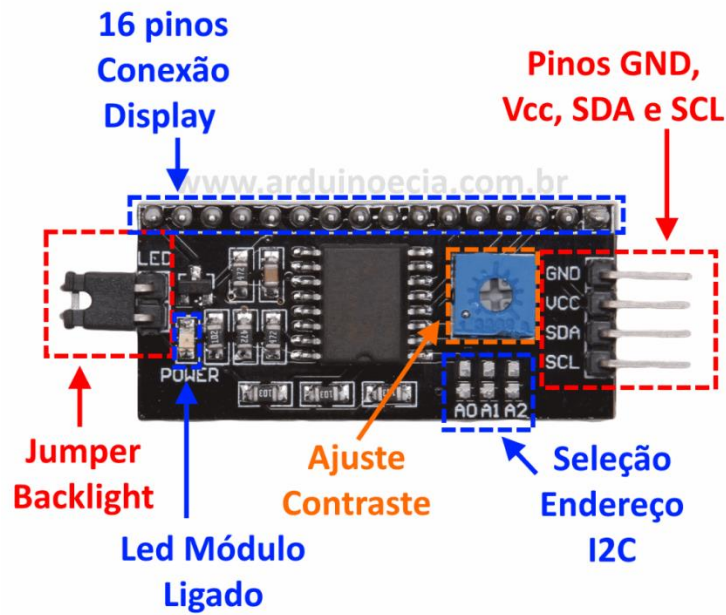
Pino 1 – GND, responsável pela conexão do terra.

Pino 2 – VCC, responsável pela alimentação.

Pino 3 – SDA, pino serial data.

Pino 4 – SCL, pino serial clock.

Figura 8 -Módulo I2C pinout.



Fonte: Arduino e Cia (2014).

Na Figura 9, pode-se ser visto os possíveis endereços dos módulos I2C.

Figura 9 - Endereços do módulo I2C.

Endereço	A0	A1	A2
0x20	0	0	0
0x21	1	0	0
0x22	0	1	0
0x23	1	1	0
0x24	0	0	1
0x25	1	0	1
0x26	0	1	1
0x27	1	1	1

Fonte: Arduino e Cia (2014).

Na Tabela 4, pode-se ser visto algumas informações técnicas do módulo I2C, importantes para realizar o desenvolvimento do projeto.

Tabela 4 - Especificações técnicas do módulo I2C.

<b>Especificações Técnicas</b>	
<b>Endereço I2C padrão</b>	0x20
<b>Compatibilidade</b>	Display LCD 16x2 20x4

<b>Tensão de operação</b>	5V
<b>Dimensões</b>	55x23x14mm
<b>Peso</b>	5g

Fonte: Próprio autor.

### 2.6.2. Display LCD 16x2

O Display LCD 16x2 é um modelo utilizado em projetos onde se necessita uma interface homem-máquina, sendo composto por 16 colunas e 2 linhas com a escrita na cor branca e sua luz de fundo na cor azul para exibição de caracteres, letras e números de forma clara e nítida, melhorando a visibilidade para quem recebe a informação.

O Display LCD 16x2 utiliza o controlador HD44780, utilizado em toda indústria de LCD's como base de interface que pode ser operado em 4 ou 8-bits paralelamente.

Na Figura 10, pode ser visto que o display possui 16 pinos, sendo eles:

Pino 1 – GND, responsável pela conexão do terra

Pino 2 – VCC, responsável pela alimentação do LCD

Pino 3 – Vo, responsável pelo nível de contraste da tela

Pino 4 – Rs, responsável pela conexão ao microcontrolador para alternar entre registro e comando de dados

Pino 5 – R/W, responsável pela seção de leitura ou escrita de dados

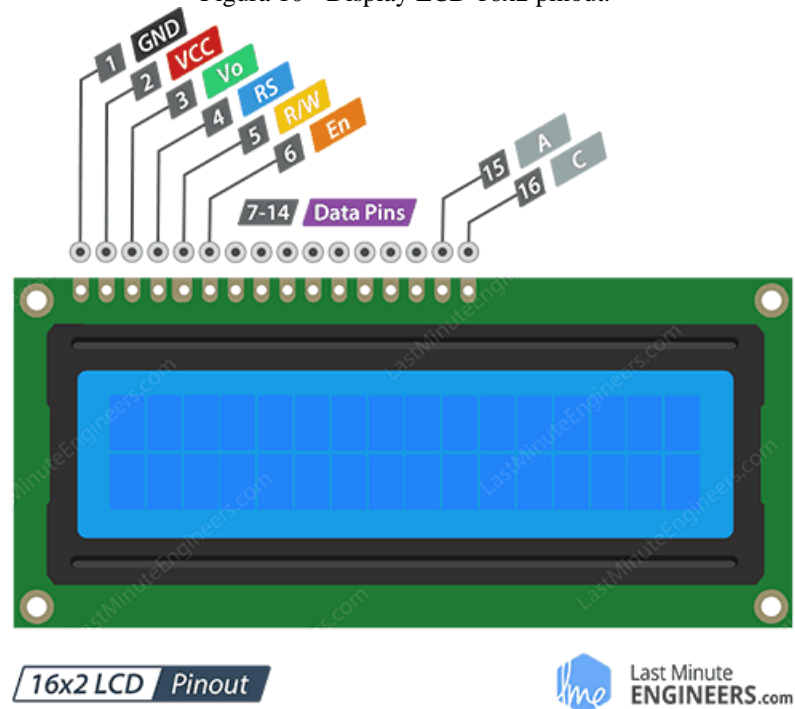
Pino 6 – En, responsável por habilitar o reconhecimento de dados

Pino 7 ao 14 – Data pins, responsável por receber os dados do microcontrolador, podendo operar em 4 ou 8 bits.

Pino 15 – A, terminal positivo do pino do led da luz de fundo

Pino 16 – C, terminal negativo do pino do led da luz de fundo

Figura 10 - Display LCD 16x2 pinout.



Fonte: Last Minute Engineers (2019).

## 2.7. ALERTA SONORO AUTOMATIZADO

O sistema de aviso sonoro automatizado é uma ferramenta que permite ao usuário emitir alertas ou colocar músicas automaticamente a partir de uma série de gatilhos. Ou seja, ele consegue propagar mensagens de áudio em horários pré-definidos ou quando alguma situação (como um acidente) é detectada. Assim, não há a necessidade de direcionar uma pessoa para a gestão da aplicação (Discabos, 2020).

### 2.7.1. Buzzer Ativo

O Buzzer Ativo é um alto-falante que serve para emitir sinais sonoros a partir da sua energização, não variando a frequência de emissão.

Na Figura 11 é visto que ele, possui dois pinos, sendo eles:

Pino 1 – 5V, responsável pelo controle do buzzer nos estados ligado/desligado

Pino 2 – GND, responsável pela conexão do terra

Figura 11 - Buzzer ativo pinout.



Fonte: Unit Electronics(2022).

## **2.8.DESENVOLVIMENTO WEB**

Desenvolvimento web é a área da tecnologia voltada à construção de sites, aplicativos, softwares, bancos de dados e quaisquer outras ferramentas que, de certa forma, constroem a internet como a conhecemos hoje. (ROVEDA,2020).

### **2.8.1. Front-end**

Programação front end é toda aquela que envolve a parte visível de um site; toda a parte com a qual o usuário pode interagir. Em outras palavras, sua interface externa. (ROVEDA,2020).

As principais linguagens front-end são HTML, CSS e JavaScript, mas as bibliotecas também são muito populares entre estes profissionais. Entre elas, as mais populares são React, jQuery e Node.js. (ROVEDA,2020).

### **2.8.2. Back-end**

Ao contrário de front-end, a programação back-end é aquela que envolve toda parte interna de um site, ou seja, tudo aquilo que diz respeito ao seu funcionamento e armazenamento de dados e informações. (ROVEDA,2020).

Quando um usuário interage com o site, por exemplo, ele faz isso através de sua interface visível, mas todas as informações inseridas ali serão processadas por um servidor, ou

armazenadas em um banco de dados, e é o desenvolvedor back-end quem está por trás de todos estes processos. As principais linguagens de programação back-end são Python, Java, PHP e Ruby. (ROVEDA,2020).

### 2.8.3. Banco de dados

Os bancos de dados, como o próprio nome sugere, são bancos virtuais que armazenam e registram informações e dados relacionados a pessoas ou coisas com o objetivo de fornecer maior eficiência e praticidade às suas leituras. (ROVEDA,2020).

Os bancos de dados são amplamente utilizados porque, de modo geral, tudo que é armazenado na internet está arquivado em um banco de dados, esteja ele instalado em uma máquina ou na forma de cloud computing -- ou armazenamento em nuvem. (ROVEDA,2020).

### 3. METODOLOGIA

O trabalho apresentado é uma pesquisa aplicada, e tem como objetivo a realização de pesquisa experimental sendo embasado em um material bibliográfico, de laboratório e de campo. Foram aplicados os procedimentos técnicos de pesquisa bibliográfica e experimental. É utilizado o método de abordagem hipotético-dedutivo e o método de procedimento monográfico em sua elaboração. Para coleta de dados é utilizada a observação direta intensiva e documentação indireta, e a análise e interpretação de seus dados será qualitativa.

Foram realizadas pesquisas bibliográficas na área de automação, microcontroladores, internet das coisas, sensores, desenvolvimento web. As pesquisas foram realizadas para coletar informações relevantes para realizar o desenvolvimento do sistema proposto neste trabalho.

A primeira etapa do desenvolvimento do projeto consistiu em montar o circuito do sistema e a programação do microcontrolador para automatizar o processo de envio de dados para a central de monitoramento.

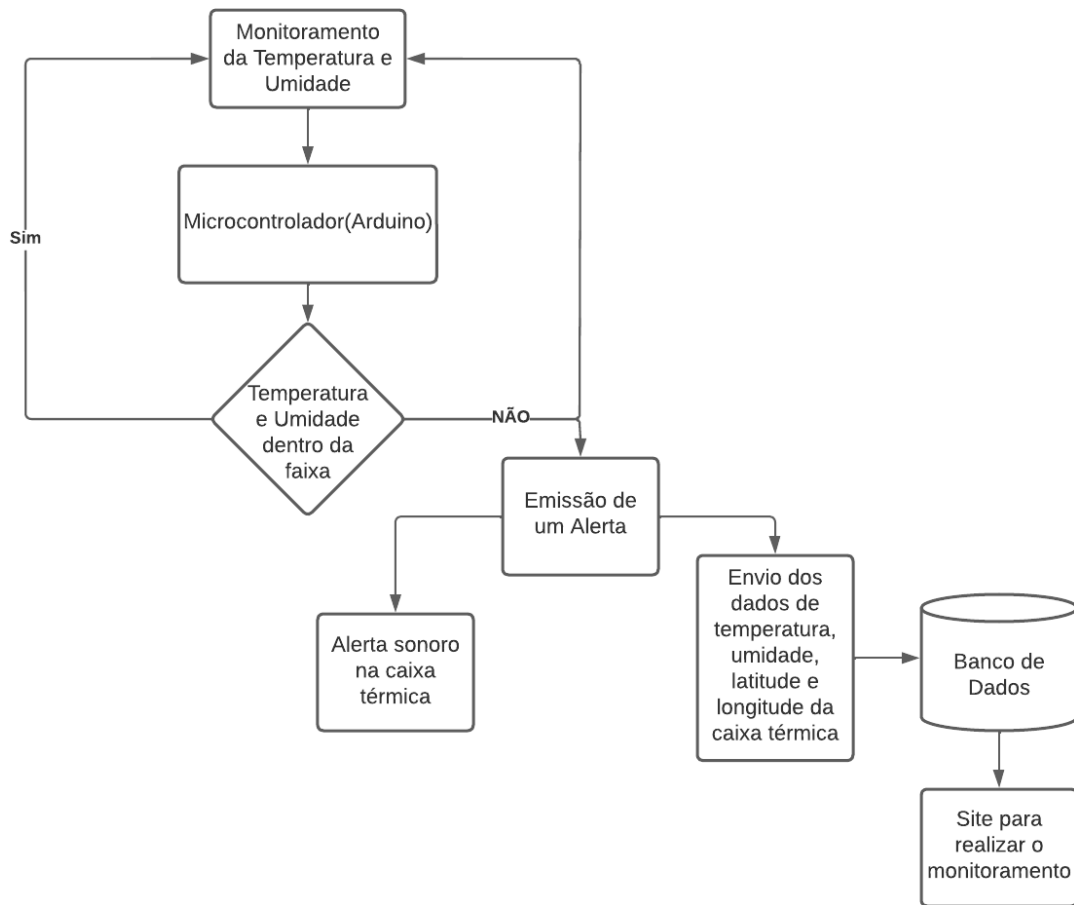
A segunda etapa foi referente ao desenvolvimento de um website para realizar o monitoramento de forma remota pela internet em tempo real dos dados enviados da caixa térmica pelo sistema de monitoramento.

#### ***3.1. DIAGRAMA EM BLOCOS DO SISTEMA***

Na Figura 12, tem-se o diagrama em blocos do sistema, o qual começa pelo monitoramento de temperatura e umidade utilizando o sensor dht11, em seguida estes dados são enviados para o Arduino, que verifica se estes estão dentro da faixa definida, se estiverem, o monitoramento continua normalmente, se não ocorre um alerta no qual acontece a emissão de um alerta sonoro pelo buzzer e o envio dos dados de temperatura, umidade, latitude e longitude da caixa térmica para um banco de dados e em seguida serão exibidos no website criado para realizar o monitoramento da caixa térmica, caso a temperatura e umidade não retornem para a faixa ideal, este processo de alerta continuará.



Figura 12 - Diagrama em blocos do sistema.



Fonte: Próprio autor.

### ***3.2.DEFINIÇÃO DA LÓGICA PARA O COÓDIGO-FONTE DO ARDUINO***

Para que se seja elaborado um software, é necessário definir parâmetros iniciais, visando facilitar o processo de programação, dessa forma tem-se que as variáveis de entrada serão a temperatura e umidade e para que o microcontrolador possa realizar ações referentes ao monitoramento delas, tem-se que a umidade é um dado complementar, não sendo definida uma faixa específica de alerta, logo será utilizado apenas uma faixa de temperatura para a emissão deste, será simulado o transporte de uma vacina que precisa ser mantida na faixa de temperatura refrigerada de 2 °C a 8° C. A temperatura deverá então ficar constantemente visível no display da caixa térmica para que o transportador possa vê-la e saber quais ações realizar. Caso a temperatura saia dessa faixa, o microcontrolador deverá ativar o buzzer para emitir um alerta sonoro e obter os dados de latitude e longitude do gps, juntamente com a temperatura e umidade do sensor e enviar estes dados para o banco de dados utilizando o módulo gsm gprs para conexão com a internet.

Dessa forma, tem-se o passo a passo que deve ser feito no código para que o sistema funcione adequadamente.

### ***3.3.DEFINIÇÕES PARA OS CÓDIGOS-FONTES DO SITE***

Para realizar os desenvolvimentos dos códigos do site, deve-se definir primeiramente o que será necessário, na parte do front-end, tem-se então a necessidade de uma página principal para realizar o login, uma outra página para realizar o monitoramento dos dados quando se estiver logado, dessa forma serão necessárias duas estruturas de dados, o primeiro responsável por guardar os usuários permitidos a entrar na parte de monitoramento e o segundo por armazenar os dados enviados pelo sistema.

## **4. IMPLEMENTAÇÃO**

Neste capítulo será apresentado as etapas de desenvolvimento do sistema, tanto do circuito quanto dos códigos fontes

### ***4.1.MATERIAIS UTILIZADOS***

Durante a montagem do circuito eletrônico, utilizou-se os seguintes materiais:

- 1 Arduino Uno
- 1 Display LCD 16x2 com módulo I2C
- 1 Buzzer Ativo
- 1 Protoboard
- 1 Sensor de Temperatura DHT11
- 1 Módulo GSM GPRS SIM8001
- 1 Módulo GPS NEO6M
- 1 fonte de 5V

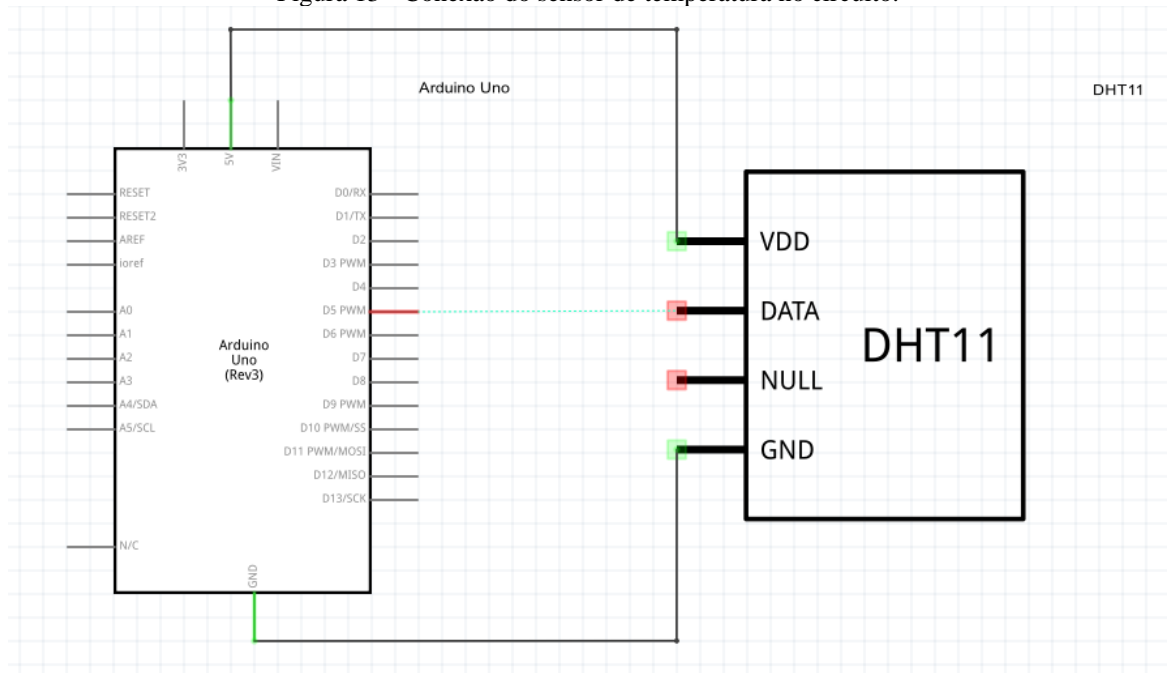
### ***4.2.MONTAGEM DO CIRCUITO***

Para começar o desenvolvimento do sistema, realizou-se primeiramente a montagem dos componentes no protoboard, para então realizar a programação no microcontrolador.

#### **4.2.1. Sensor de temperatura e umidade**

Conectou-se o pino 1(VCC) ao pino de alimentação de 5V do Arduino, o pino 2(DATA) ao pino digital 5 do Arduino e por fim o pino 4(GND) ao GND do Arduino, o resultado pode ser visto na Figura 13.

Figura 13 - Conexão do sensor de temperatura no circuito.

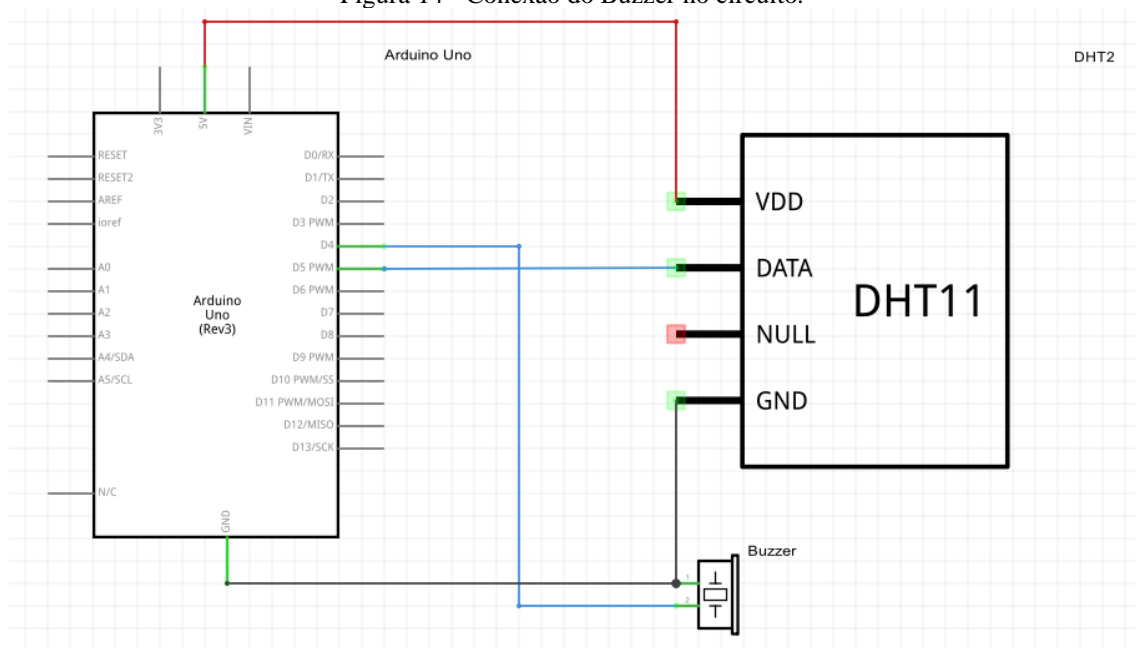


Fonte: Próprio autor.

#### 4.2.2. Buzzer ativo

Conectou-se o pino 1(POSITIVO) do buzzer ao pino digital 4 do Arduino e o pino 2(GND) ao pino de GND do Arduino, o resultado pode ser visto na Figura 14.

Figura 14 - Conexão do Buzzer no circuito.

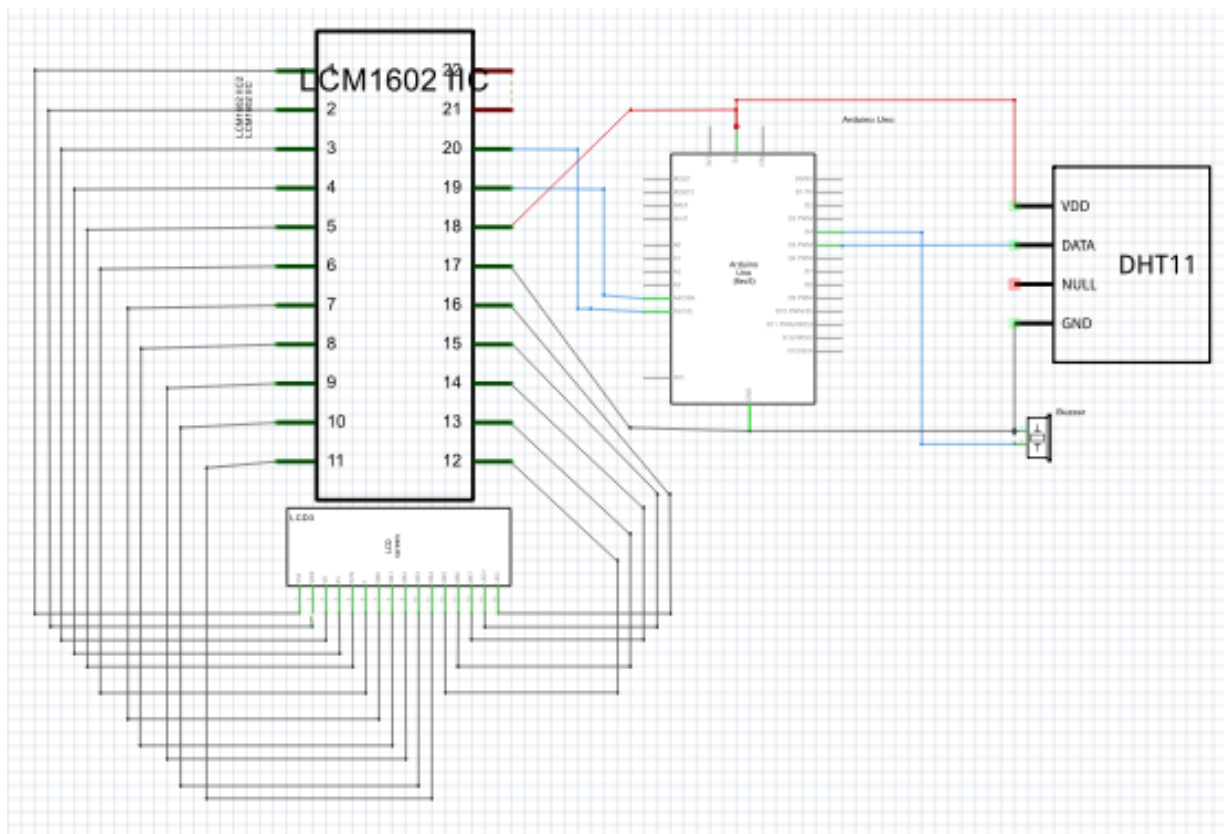


Fonte: Próprio autor.

#### 4.2.3. Display LCD 16x2 com módulo I2C

Conectou-se o pino 1(GND) do módulo I2C ao GND do Arduino, o pino 2(VCC) ao pino 5V do Arduino, o pino 3(SDA) ao pino analógico A4 do Arduino e o pino 4(SCL) ao pino analógico A5 do Arduino, o resultado pode ser visto na Figura 15.

Figura 15 - Conexão do Display no circuito.



Fonte: Próprio autor.

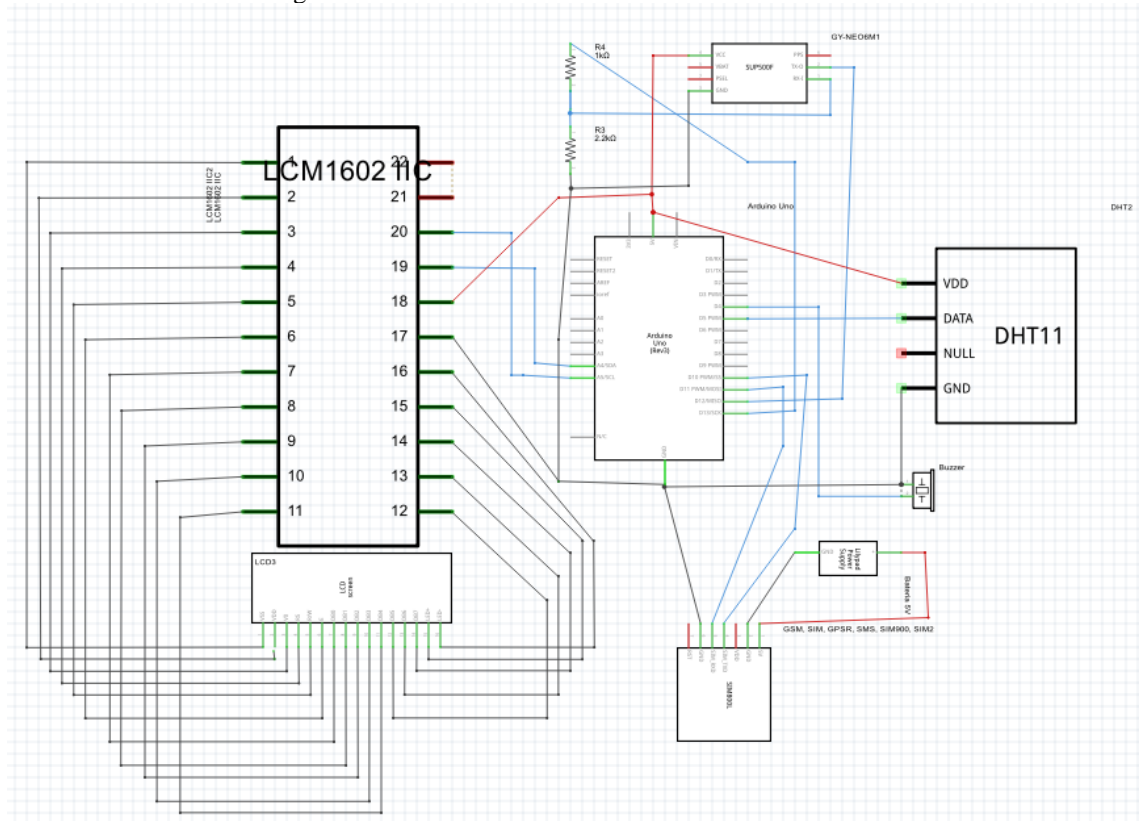
#### 4.2.4. Módulo GPS

Conectou-se o pino 1(GND) do módulo GPS ao pino GND do Arduino, o pino 2(TX) ao pino digital 12 do Arduino, no pino 3(RX), como visto na fundamentação teórica, não pode receber uma tensão de 5V, logo deve-se fazer um divisor de tensão para reduzir a tensão para 3.43V, utilizou-se dois resistores em série, sendo  $R_1=1k\Omega$  e  $R_2=2,2k\Omega$ , conectando-se o pino digital 13 do Arduino no  $R_1$  e o  $R_2$  ao terra, e ligou-se um jumper do nó ( $R_1,R_2$ ) ao pino 3(RX) do módulo GPS, por fim ligou-se o pino 4(VCC) ao pino de alimentação 5V do Arduino, o resultado pode ser visto na Figura 16.

Além das conexões dos pinos, o módulo necessita da antena para o devido



Figura 17 - Conexão do módulo GSM GPRS no circuito.



Fonte: Próprio autor.

### 4.3. PROGRAMAÇÃO DO ARDUINO

Para fazer a programação do microcontrolador, utilizou a plataforma Arduino IDE e a lógica definida anteriormente, juntamente com a Tabela 5, que facilita a visualização e o controle de cada componente do sistema conectado ao Arduino,

Tabela 5 - Conexão dos componentes nas portas do Arduino.

COMPONENTE - PINO	CONEXÃO NO ARDUINO	TIPO
Sensor de temperatura e umidade - DATA	Porta digital 5	INPUT
Módulo GPS - TX	Porta digital 12	SERIAL
Módulo SM GPRS - TX	Porta digital 10	SERIAL
Módulo GPS - RX	Porta digital 13	SERIAL
Módulo GSM GPRX - RX	Porta digital 11	SERIAL
Buzzer - DATA	Porta digital 4	OUTPUT
Módulo I2C - SDA	Porta analógica A4	
Módulo I2C - SCL	Porta analógica A5	

Fonte: Próprio autor.

#### 4.4. CRIAÇÃO DO WEBSITE

Para realizar o monitoramento em tempo real dos dados enviados pelo sistema foi necessário criar um website, este contém duas páginas, sendo a página 1 de login e a página de 2 de monitoramento dos dados.

Em relação ao banco de dados, foram criadas duas estruturas, a primeira foi a “usuário”, responsável por guardar os dados de quem está permitido acessar o campo de monitoramento, a estrutura contém os campos ‘id’ que é um valor inteiro auto incrementado, ‘nome’, ‘e-mail’ e ‘senha’, sendo estes do tipo varchar com codificação utf8\_unicode, na Figura 18 pode-se ser visto estes campos criados no banco de dados.

Figura 18 - Criação da estrutura usuário no banco de dados.



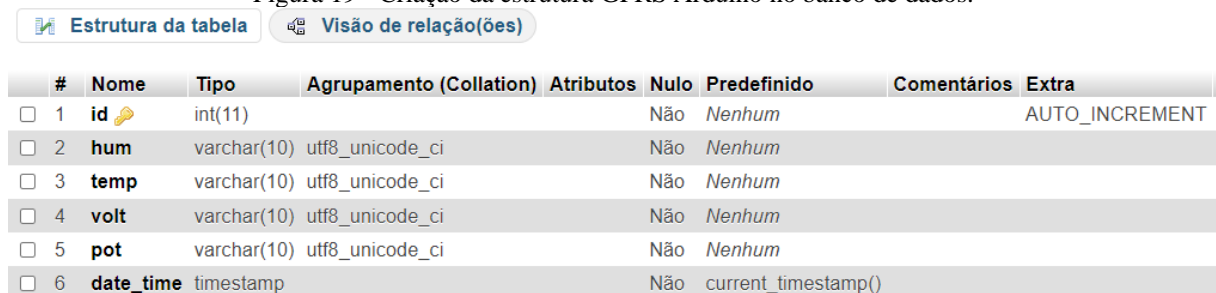
The screenshot shows a table structure view with two tabs: 'Estrutura da tabela' (selected) and 'Visão de relação(ões)'. The table has the following columns: #, Nome, Tipo, Agrupamento (Collation), Atributos, Nulo, Predefinido, Comentários, and Extra. The rows are:

#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra
1	id	int(32)			Não	Nenhum		AUTO_INCREMENT
2	nome	varchar(200)	utf8_unicode_ci		Não	Nenhum		
3	email	varchar(200)	utf8_unicode_ci		Não	Nenhum		
4	senha	varchar(32)	utf8_unicode_ci		Não	Nenhum		

Fonte: Próprio autor.

A segunda estrutura foi a “gprsArduinoTest”, responsável por guardar os dados enviados pelo sistema controlado pelo arduino, foram criados 6 campos, sendo eles o ‘id’ do tipo inteiro auto incrementado, a ‘hum’ responsável por armazenar a umidade medida, a ‘temp’ responsável por armazenar a temperatura medida, a ‘volt’ responsável por armazenar a latitude medida, a ‘pot’ responsável por armazenar a longitude medida e a ‘date\_time’ responsável por armazenar o horário que os dados foram recebidos, na Figura 19 pode-se ser visto a criação desta estrutura no banco de dados.

Figura 19 - Criação da estrutura GPRS Arduino no banco de dados.



The screenshot shows a table structure view with two tabs: 'Estrutura da tabela' (selected) and 'Visão de relação(ões)'. The table has the following columns: #, Nome, Tipo, Agrupamento (Collation), Atributos, Nulo, Predefinido, Comentários, and Extra. The rows are:

#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra
1	id	int(11)			Não	Nenhum		AUTO_INCREMENT
2	hum	varchar(10)	utf8_unicode_ci		Não	Nenhum		
3	temp	varchar(10)	utf8_unicode_ci		Não	Nenhum		
4	volt	varchar(10)	utf8_unicode_ci		Não	Nenhum		
5	pot	varchar(10)	utf8_unicode_ci		Não	Nenhum		
6	date_time	timestamp			Não	current_timestamp()		

Fonte: Próprio autor.

O site e o banco de dados foram hospedados em um servidor gratuito (br.000webhost.com) no qual foram hospedados os arquivos do site e a criação do banco de dados utilizando a ferramenta phpMyAdmin, disponível online no servidor.

## 5. TESTES E RESULTADOS OBTIDOS

### 5.1. TESTE DO SENSOR E DISPLAY

Para validar o funcionamento do sensor dht11 e do display, variou-se a temperatura ambiente e realizou-se a comparação da medida mostrada no display com um sensor de temperatura digital presente no celular.

Na Figura 20, tem-se que o sensor mede uma temperatura de 32.00 °C e o celular mede uma temperatura de 32,1° C, uma diferença de 0,31%.

Figura 20 – Comparação 1 de medição de temperatura do sistema e de um medidor de referência.



Fonte: Próprio autor.

Na Figura 21, tem-se o sensor medindo uma temperatura de 27.00° C e o celular medindo uma temperatura de 26,9° C, uma diferença de 0,37%.

Figura 21 - Comparação 2 de medição de temperatura do sistema e de um medidor de referência.



Fonte: Próprio autor.



### 5.2. TESTE DO BUZZER

Para validar o funcionamento do buzzer, simulou-se uma temperatura dentro da faixa estipulada de 2°C a 8°C, nesse caso o buzzer não foi ativado em seguida simulou-se uma temperatura acima de 8°C, ocasionando na ativação do buzzer e na emissão do alerta sonoro, estando então de acordo com o funcionamento esperado.

### 5.3. TESTE DO MÓDULO GPS

Para validar se o GPS estava funcionando corretamente, verificou-se as informações de latitude e longitude que ele informava, obteve-se a latitude igual -3.05329 e a longitude igual -59.98277. Verificou-se então no google maps se estes dados estavam de acordo com o local que o gps estava localizado através do link <http://maps.google.com/maps?q=-3.05329,-59.98277>. Os dados estavam corretos, demonstrando o pleno funcionamento do módulo.

### 5.4. TESTE DO MÓDULO GSM GPRS E DA COMUNICAÇÃO COM O BANCO DE DADOS

Para validar o funcionamento do módulo GPS GSM, simulou-se que a temperatura saiu da faixa estipulada de 2°C a 8°C, dessa forma os dados de temperatura, umidade, latitude e longitude deveriam ser enviados para o banco de dados através da comunicação feita utilizando o módulo conectado à rede móvel. Na Figura 22, pode ser visto que o banco de dados recebeu com sucesso os dados enviados.

Figura 22 - Informações do sistema recebidas no banco de dados.

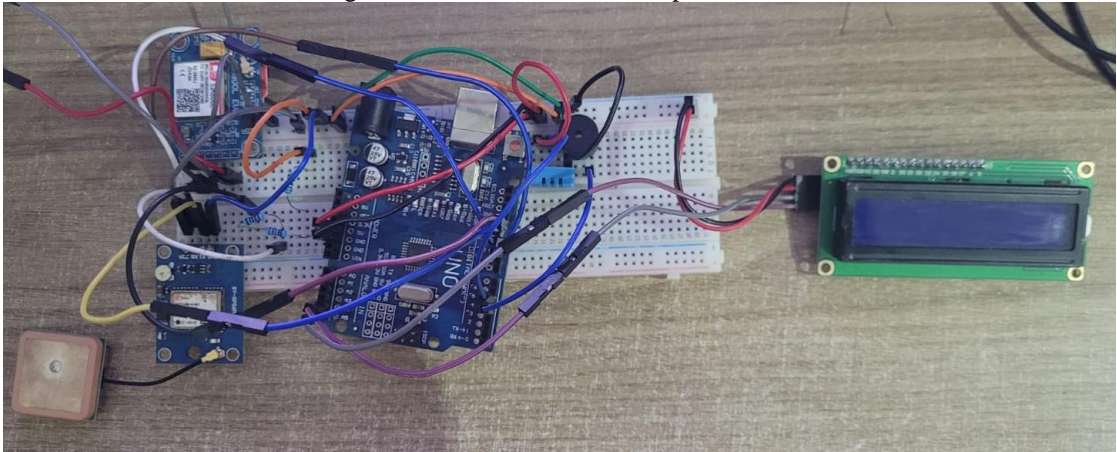
+ Opções			id	hum	temp	volt	pot	date_time	
<input type="checkbox"/>	Edita	Copiar	Apagar	24	56	32	-3.05329	-59.98277	2022-09-14 19:19:59
<input type="checkbox"/>	↑	<input type="checkbox"/> Marcar todos	Com os seleccionados:			Edita	Copiar	Apagar	Exportar

Fonte: Próprio autor.

### 5.5. CIRCUITO MONTADO

Na Figura 23, pode-se ser visto o protótipo montado com todos os componentes no protoboard.

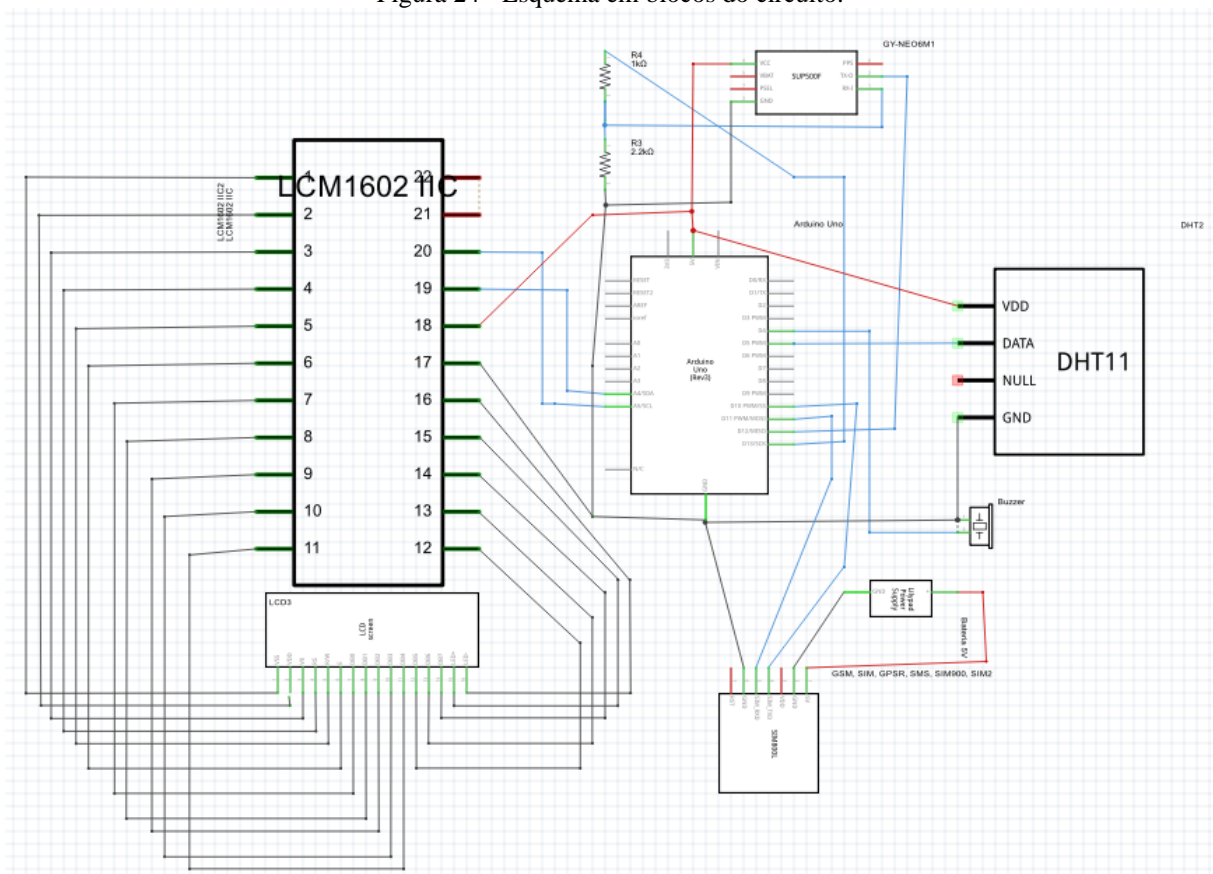
Figura 23 - Sistema montado no protoboard.



Fonte: Próprio autor.

Na Figura 24, pode-se ser visto o esquema em blocos do circuito, demonstrando todas as conexões feitas nos componentes.

Figura 24 - Esquema em blocos do circuito.

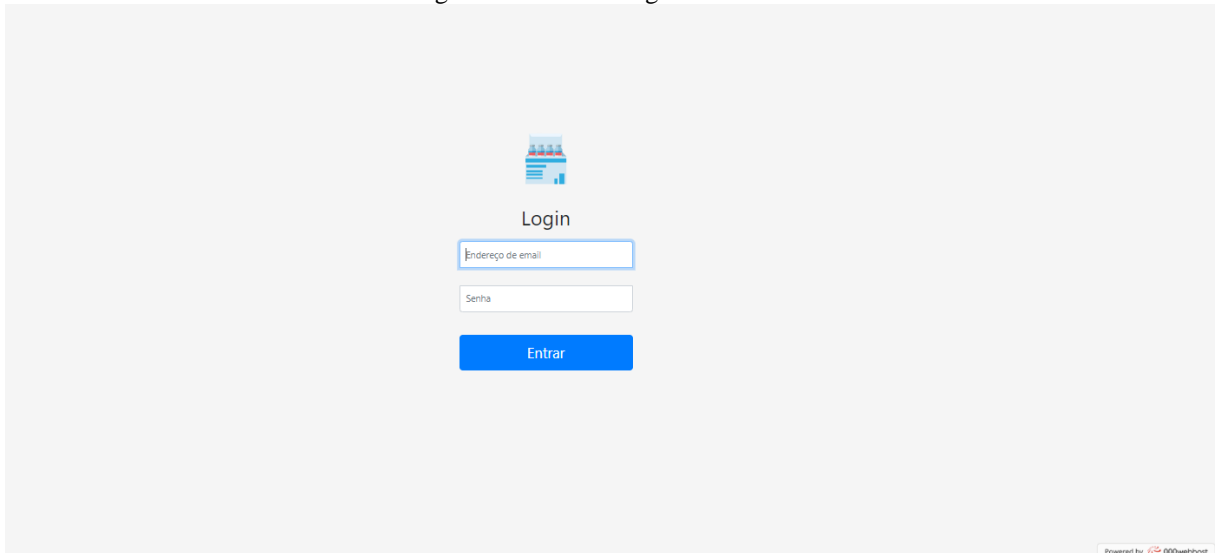


Fonte: Próprio autor.

## 5.6. WEBSITE E MONITORAMENTO

Na Figura 25, pode ser visto a tela inicial do website criado, sendo a tela de login.

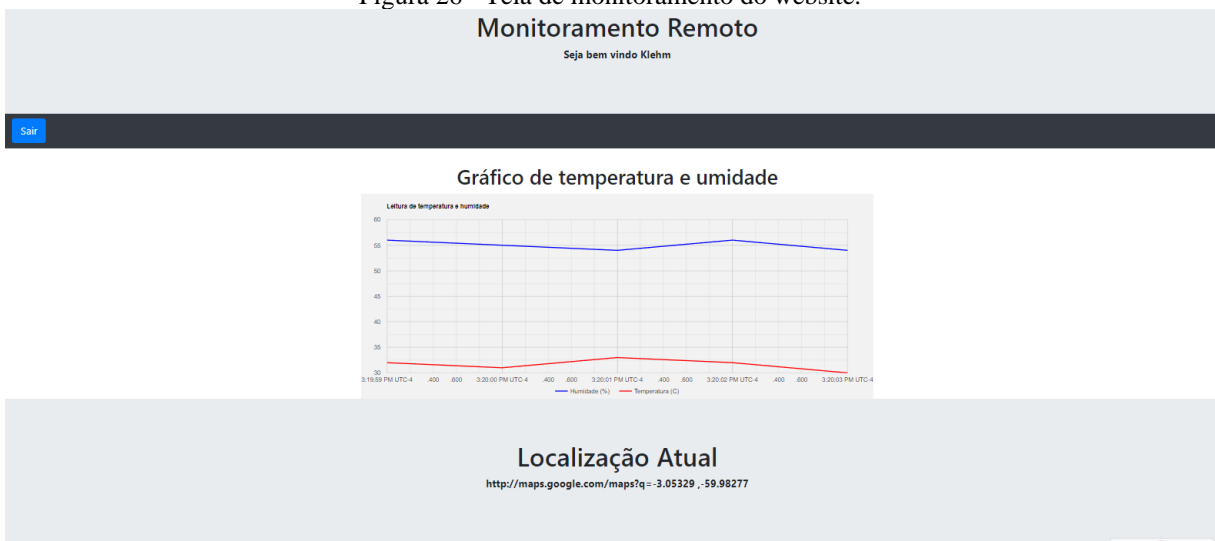
Figura 25 - Tela de login do website.



Fonte: Próprio autor.

Na Figura 26, tem-se a página de monitoramento dos dados da caixa térmica, composta por um gráfico para visualizar os dados de temperatura e umidade, e um link do maps informando a última localização da caixa térmica.

Figura 26 - Tela de monitoramento do website.


















Fonte: Próprio autor.

Na Figura 27, tem-se o banco de dados preenchido com a simulação de alguns valores de temperatura fora da faixa indicada de 2°C a 8°C, estes valores são exibidos na página de

monitoramento através do gráfico da Figura 28 e do link da Figura 29. A partir do gráfico é possível identificar por quanto tempo a caixa térmica ficou fora da faixa ideal, neste caso foram de cinco minutos, permitindo que posteriormente sejam realizadas análises para identificar se esse período de tempo prejudicou os produtos transportados.

Figura 27 - Banco de dados preenchido com dados do sistema.

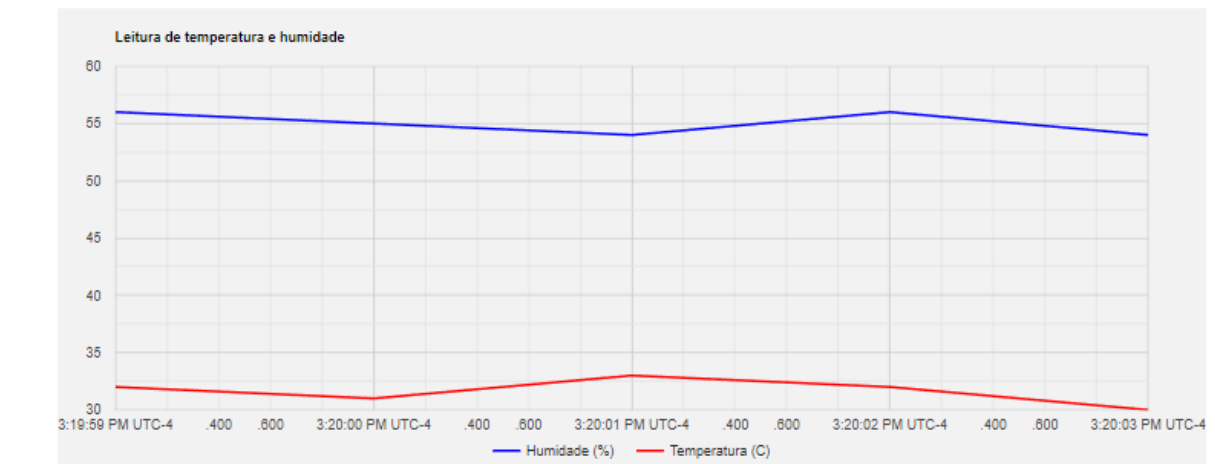
+ Opções

				id	hum	temp	volt	pot	date_time
<input type="checkbox"/>	 Edita	 Copiar	 Apagar	24	56	32	-3.05329	-59.98277	2022-09-14 19:19:59
<input type="checkbox"/>	 Edita	 Copiar	 Apagar	43	55	31	-3.05329	-59.98277	2022-09-14 19:20:00
<input type="checkbox"/>	 Edita	 Copiar	 Apagar	44	54	33	-3.05329	-59.98277	2022-09-14 19:20:01
<input type="checkbox"/>	 Edita	 Copiar	 Apagar	45	56	32	-3.05329	-59.98277	2022-09-14 19:20:02
<input type="checkbox"/>	 Edita	 Copiar	 Apagar	46	54	30	-3.05329	-59.98277	2022-09-14 19:20:03

Fonte: Próprio autor.

Figura 28 - Gráfico de monitoramento de temperatura e umidade.

## Gráfico de temperatura e umidade



Fonte: Próprio autor.

Figura 29 - Link do Google Maps para rastrear a caixa térmica.

**Localização Atual**

<http://maps.google.com/maps?q=-3.05329,-59.98277>

Fonte: Próprio autor.

## CONCLUSÃO

Este trabalho teve o intuito de desenvolver um sistema de monitoramento e alerta em tempo real para caixas térmicas, baseado na leitura de temperatura e umidade pelo sensor, visando garantir que a cadeia de frio seja mantida através de ações imediatas em decorrência dos parâmetros saírem da faixa ideal, caso estas ações não sejam possíveis, o sistema permite identificar por quanto tempo estes ficaram fora de sua faixa, permitindo analisar posteriormente se o produto transportado deve ser descartado ou ainda poderá ser utilizado. Portanto, a partir do desenvolvimento de um sistema responsável por realizar o monitoramento e alerta em tempo real e do desenvolvimento do website no qual será possível fazer esse monitoramento dos dados e a partir da análise das simulações e resultados, conclui-se que a problemática foi atendida com o desenvolvimento de um sistema funcional e eficiente.

Em relação a projetos futuros, fica a sugestão de um desenvolvimento de uma aplicação mobile capaz de realizar o monitoramento assim como a aplicação web, também o desenvolvimento de um sistema de controle de temperatura ativo que possa ser acionado visando retornar a temperatura para sua faixa ideal, além de um estudo de eficiência energética onde torne possível um sistema de alimentação com autonomia para todo o período do transporte.

## REFERÊNCIAS BIBLIOGRÁFICAS

Acoptex. **Basics Project 053a SIM800L GSM GPRS module.** <<https://acoptex.com/project/565/basics-project-053a-sim800l-gsm-gprs-module-at-acoptexcom/>> Acesso em: 20 de set. de 2022.

Arduino e Cia. **Como usar o módulo I2C com Arduino e display LCD 16x2.** <<https://www.arduinoecia.com.br/modulo-i2c-display-16x2-arduino/>> Acesso em: 20 de set. de 2022.

BANZI, Massimo. **Primeiros passos com o arduino.** São Paulo: Novatec, 2011.

BRAGHETTO, L. F. B; SILVA, S. C. DA; BRISQUI, M. L.; COSTA, P. **Redes GSM e GPRS.** 2003. 46f. Dissertação (Pós-Graduação em Redes Computadores) – Universidade Estadual de Campinas, Campinas. 2003. Disponível em <<http://www.braghetto.eti.br/files/Trabalho%20Final%20GSM.pdf>>. Acesso em: 07 abr 2022.

CARVALHO, Edilson; ARAUJO, Paulo. **Leituras cartográficas e interpretações estatísticas II – Aula 8 – Noções básicas de sistema de posicionamento global GPS.** Natal, RN: EDUFRN, 2009.

COCOTA, José. **Sistema de Monitoramento de Temperatura e Umidade.** Eneca. Florianópolis. 5f.[Out 2005]. Disponível em: <[http://professor.ufop.br/sites/default/files/cocota/files/eneca\\_2005\\_smtu.pdf](http://professor.ufop.br/sites/default/files/cocota/files/eneca_2005_smtu.pdf)>. Acesso em: 05 Abr 2022.

Discabos. **Aviso sonoro automatizado: para que serve e como funciona?.** Disponível em: <<https://discabos.com.br/blog/exemplos-de-projetos/aviso-sonoro-automatizado-para-que-serve-e-como-funciona/>>. Acessado em: 06 de abr de 2022.

G Brasche, B Walke – “[Concepts, services, and protocols of the new GSM phase 2+ generalpacket radio service](#)” - Communications Magazine, IEEE, 1997 - [ieeexplore.ieee.org](http://ieeexplore.ieee.org);

Incoterminas. Caixa térmica com termômetro embutido: entenda tudo sobre este assunto! <<https://www.incoterminas.com.br/post/caixa-t%C3%A9rmica-com-term%C3%B4metro-embutido-entenda-tudo-sobre-este-assunto>> Acesso em: 20 de set. de 2022.

J Cai, DJ Goodman - “[General packet radio service in GSM](#)” - Communications Magazine, IEEE, 1997 - [ieeexplore.ieee.org](http://ieeexplore.ieee.org);

Jornal de Brasília. **Displays: o que são, tipos e suas aplicações com o Arduino.** [26 ago 2021]. Disponível em: <<https://jornaldebrasil.com.br/nahorah/displays-o-que-sao-tipos-e-suas-aplicacoes-com-o-arduino/>>. Acessado em: 07 de abr de 2022.

Last Minute Engineers. **Interface ublox NEO-6M GPS Module with Arduino** <<https://lastminuteengineers.com/neo6m-gps-arduino-tutorial/>> Acesso em: 20 de set. de 2022.

Last Minute Engineers. **Interfacing 16x2 Character LCD Module with Arduino**<

<https://lastminuteengineers.com/neo6m-gps-arduino-tutorial/>> Acesso em: 20 de set. de 2022.

M Rahnema - “[Overview of the GSM system and protocol architecture](#)” - Communications Magazine, IEEE, 1993 - [ieeexplore.ieee.org](http://ieeexplore.ieee.org);

MAIO, Claudia; SILVA, Jose. **Armazenagem e distribuição de medicamentos na cadeia fria**. Latin American Journal Of Business Management. Taubaté. 19f. [18 Set. 2014]. Disponível em: <<https://www.lajbm.com.br/index.php/journal/article/view/209/114>>. Acesso em: 08 Abr 2022.

MCROBERTS, Michael. **Arduino básico**. São Paulo: Novatec, 2011.

MONK, Simon. **Programação com arduino: começando com sketches**, Porto Alegre: Bookman, 2013.

Mouser Electronics. **Datasheet - DHT 11 Humidity & Temperature Sensor**<<https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>> Acesso em: 20 de set. de 2022.

NOLETO, CAIO; **Arduino: o que é, para que serve e como começar a usar**. 2021. Disponível em: <<https://blog.betrybe.com/tecnologia/arduino-tudo-sobre/>>. Acesso em: 20 abr. 2022

OLIVEIRA, C. L. V. **Aprenda Arduino – Uma abordagem prática**. Duque de Caixas: Katzen Editora, 2018.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P.; NABARRO, C. B. M.; GONÇALVES, J. A. V. **Aprenda arduino: uma abordagem prática. Duque de Caxias**: Katzen, 2018. Disponível em: <http://www.fatecjd.edu.br/fatecino/material/ebook-aprenda-Arduino.pdf>. Acesso em: 22 ago. 2022.

OLIVEIRA, Cláudia L. V., **Arduino Descomplicado: Como Elaborar Projetos De Eletrônica**, 2015, Erica.

Oliveira, Sergio. **Internet das Coisas com ESP8266, Arduino e Raspberry Pi**. São Paulo: Novatec, 2017.

RESOLUÇÃO DA DIRETORIA COLEGIADA - RDC Nº 430, DE 8 DE OUTUBRO DE 2020 – ANVISA.

ROVEDA, Ugo. **Desenvolvimento WEB: O que é e como ser um desenvolvedor WEB**. <<https://kenzie.com.br/blog/desenvolvimento-web/>> Acesso em: 20 de set. de 2022.

SANTOS, B. P. et al. **Internet das Coisas: da Teoria à Prática**. 2016. Disponível em: <<http://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf>>. Acessado em 05 de abr de 2022.

SimCom. **Datasheet – SIM800L Hardware Design V1.00**. <<https://datasheetspdf.com/pdf/989664/SIMCom/SIM800L/1>> Acesso em: 20 de set. de 2022.

SVERZUT, J.R. - “Redes GSM, GPRS, EDGE e UMTS – Uma evolução a caminho da terceira geração (3G)”, Ed Érica, 2005;

[TANABE](#), Shiguetoshi Celso; [CORTEZ](#), Luís Augusto Barbosa - *Perspectivas da cadeia do frio para frutas e hortaliças no Brasil* [em linha]. Nazaré, PA: Cabano Engenharia, 1998. [Consul. 16 Abr. 2008]. Disponível em: <<http://www.cabano.com.br/frio%20e%20frutas.htm>>. Acesso em: 08 Abr 2022.

Ublox. **Datasheet NEO-6M u-blox 6 GPS modules**. < <https://datasheetspdf.com/pdf-file/866235/u-blox/NEO-6M/1>> Acesso em: 20 de set. de 2022.

Unit Electronics. Buzzer 5V Ativo < <https://uelectronics.com/producto/buzzer-5v-activo/>> Acesso em: 20 de set. de 2022.

Zelenovsky R, Mendonça A (2005), **Microcontroladores: Programação e Projeto com a Família 8051**. MZ Editora, Rio de Janeiro-RJ.



## APÊNDICE A - CÓDIGO DO ARDUINO

```

// --- Bibliotecas
#include <dht.h> //biblioteca do sensor de umidade e temperatura
#include <Wire.h> //Biblioteca do I2c
#include <LiquidCrystal_I2C.h> //biblioteca do lcd
#include <SoftwareSerial.h>
#include <TinyGPS.h>

#include <String.h>
#include <SoftwareSerial.h>
#include "DHT.h"
#define OK 1
#define NOTOK 2
#define TIMEOUT 3

// --- Mapeamento de Hardware ---
#define dht_pin 5 //pino de sinal do dht11 ligado no digital 5
#define buzzer 4

#define endereco 0x27 // Endereços comuns: 0x27, 0x3F
#define colunas 16
#define linhas 2

#define GPS_RX 12
#define GPS_TX 13
#define GPS_Serial_Baud 9600

SoftwareSerial A6board (10, 11); // RX e TX

char end_c[2];
float voltage;
float denominator;
float umidade;

int signalPower = 0;
int avaregaPower = 0;

bool A6begin();
String A6read();
void reset();
int getSignalStrength();
bool sendToRemoteServerGSM(float hum, float temp, float volt, int pot);
String returnStringFromModemComand(String ATcommand, int responseDelay);
byte A6waitFor(String responsel, String response2, int timeOut);
byte A6command(String command, String responsel, String response2, int
timeOut, int repetitions);

// --- Declaração de Objetos ---
dht my_dht; //objeto para o sensor
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
TinyGPS gps;

```

```

SoftwareSerial gpsSerial(GPS_RX, GPS_TX);

// --- Variáveis Globais ---
float   temperatura = 0;    //armazena a temperatura em inteiro
int     temperaturaAnterior=0;
int     aux=0;
String  mensagem;
float   flat=0, flon=0;

void setup()

{

    // put your setup code here, to run once:
    A6board.begin(9600);           // the GPRS baud rate
    Serial.begin(9600);           // Serial baud rate
    end_c[0] = 0x1a;
    end_c[1] = '\0';

    lcd.init(); // INICIA A COMUNICAÇÃO COM O DISPLAY
    lcd.backlight(); // LIGA A ILUMINAÇÃO DO DISPLAY
    lcd.clear(); // LIMPA O DISPLAY

    lcd.print("Iniciando...");
    delay(1000); // DELAY DE 5 SEGUNDOS
    lcd.setCursor(0, 1); // POSICIONA O CURSOR NA PRIMEIRA COLUNA DA LINHA 2

    //lcd.noBacklight(); // DESLIGA A ILUMINAÇÃO DO DISPLAY
    //delay(2000); // DELAY DE 2 SEGUNDOS
    //lcd.backlight(); // LIGA A ILUMINAÇÃO DO DISPLAY
    //delay(2000); // DELAY DE 2 SEGUNDOS

    lcd.clear(); // LIMPA O DISPLAY
    //lcd.noBacklight(); // DESLIGA A ILUMINAÇÃO DO DISPLAY

    Serial.begin(GPS_Serial_Baud);
    gpsSerial.begin(GPS_Serial_Baud);

    pinMode(dht_pin, INPUT);
    pinMode(buzzer, OUTPUT);

} //end setup

//
=====
====
// --- Loop Infinito ---
void loop()
{

gpsSerial.listen();

//GPS
  bool newData = false;
  unsigned long chars;
  // Por um segundo, analisamos os dados do GPS e relatamos alguns valores-chave
  for (unsigned long start = millis(); millis() - start < 1000;)
  {

```

```

    while (gpsSerial.available())
    {
        char c = gpsSerial.read();
        if (gps.encode(c)) // Atribui true para newData caso novos dados
sejam recebidos
            newData = true;
    }
}
if (newData)
{

    gps.f_get_position(&flat, &flon);
    Serial.println(flat);
    Serial.println(flon);

}

A6board.listen();
//Leitura Temperatura
my_dht.read11(dht_pin);

temperatura = my_dht.temperature;
umidade = my_dht.humidity;
//temperatura = 10;

Serial.println("Teste");
temperatura=5;
voltage=300;
avaregaPower=32;

//Condição pra Ativa o Buzzer
if(temperatura<2.00 or temperatura>7.00){
    digitalWrite(buzzer,HIGH);

    if(aux==0){
        mensagem = "Perigo!! Temperatura da caixa esta fora da faixa!
Temperatura Atual:" + String(temperatura) + " Localizacao da caixa:
http://maps.google.com/maps?q=" +String(flat,6)+", "+String(flon,6);
        Serial.println("Sending Data..."); //Displays on the
serial monitor
        sendToRemoteServerGSM(umidade, temperatura, flat, flon); //
Envia para o servidor
        Serial.println(flat);
        Serial.println(flon);

        Serial.println("Data Sent.");
        delay(500);
        aux=0;
    }

}

if(temperatura>=20.00 and temperatura<=40.00){
    digitalWrite(buzzer,LOW);
    aux=0;
}

//Imprimir a temperatura no display

```

```

    if(temperaturaAnterior!=temperatura){
        lcd.clear();
        lcd.print("Temperatura:");
        lcd.setCursor(0, 1); // POSICIONA O CURSOR NA PRIMEIRA COLUNA DA
LINHA 2
        lcd.print(temperatura);
        lcd.write(223);
        lcd.print("C");
        temperaturaAnterior=temperatura;
    }

} //end loop

byte A6command(String command, String responsel, String response2, int
timeOut, int repetitions) {
    byte returnValue = NOTOK;
    byte count = 0;
    while (count < repetitions && returnValue != OK) {
        A6board.println(command);
#ifdef DEBUG
        Serial.print("Command: ");
        Serial.println(command);
#endif
        if ( A6waitFor(responsel, response2, timeOut) == OK ) {
            returnValue = OK;
        } else {
            returnValue = NOTOK;
        }
        count++;
    }
    return returnValue;
}

String A6read() {
    String reply = "";
    if (A6board.available()) { // Se algum dado estiver disponível
na serial,
        reply = A6board.readString(); // Realiza a leitura do dado
disponível na serial
    }
    return reply; // Retorna a String de resposta
}

byte A6waitFor(String responsel, String response2, int timeOut) {
    unsigned long entry = millis(); // Armazena o valor da variável
millis para saber quanto tempo estará dentro da função, relizar o timeout
    int count = 0; // Variável para o contador
    String reply = A6read(); // Realiza a leitura da resposta do
modem
    byte retVal = 99; // Variável para o retorno
    do {
        reply = A6read(); // Realiza uma primeira leitura
#ifdef DEBUG
        if (reply != "") {
            Serial.print((millis() - entry));
            Serial.print(" ms ");
            Serial.println(reply);
        }
#endif
    }
}
#endif

```

```

    } while ((reply.indexOf(response1) + reply.indexOf(response2) == -2) &&
millis() - entry < timeOut ); // Tenta localizar as strings de resposta
utilizando indexOf()
    if ((millis() - entry) >= timeOut) { // Confere se o tempo foi excedido
        retVal = TIMEOUT;
    } else {
        if (reply.indexOf(response1) + reply.indexOf(response2) > -2) retVal =
OK; // Tenta localizar novamente se as strings passadas como parametro
estão presentes
        else retVal = NOTOK;
    }
    return retVal;
}

bool sendToRemoteServerGSM(float hum, float temp, float volt, int pot) {
    String host = "klehm.000webhostapp.com";
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // VERIFICA O STATUS DA CONEXÃO IP
    A6command("AT+CGATT?", "OK", "yy", 200,
2); // VERIFICA SE ESTAMOS CONECTADOS A REDE
    A6command("AT+CGATT=1", "OK", "yy", 200,
2); // SE NÃO ESTIVER CONECTADO A REDE, CONECTA
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // VERIFICA O STATUS DA CONEXÃO IP
    A6command("AT+CGDCONT=1,\"IP\", \"claro.com.br\"", "OK", "yy", 200,
2); // CONECTA A REDE GPRS DA TIM BRASIL COM O APN timbrasil.br
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // STABELECE O LINK PDP - PACKEGE DATA
    PROTOCOL
    A6command("AT+CGACT=1,1", "OK", "yy", 100,
2); // ATIVA O CONTEXTO DO PDP (PACKET DATA
    PROTOCOL)
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // VERIFICA NOVAMENTE O STATUS DA CONEXAO
    A6command("AT+CIFSR", "OK", "yy", 200,
2); // SOLICITA O IP LOCAL
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // VERIFICA O STATUS DA REDE NOVAMENTE
    A6command("AT+CIPSTART=\"TCP\", \"\" + host + "\",80", "CONNECT OK", "yy",
250, 2); // INICIA A CONEXAO DO TIPO TCP, NA PORTA 80, E ESPERA A
    RESPOSTA "CONNECT OK"
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // SOLICITA O STATUS DA REDE NOVAMENTE
    A6command("AT+CIPSEND", ">", "yy", 100,
1); // ABRE A PORTA PARA ENVIAR OS DADOS
    ATRAVEZ DE UMA CONEXAO TCP

    delay(100);

    // Formata a URL, no realizando uma requisição GET, do protocolo HTTP
    (Tamanho máximo de 255 caracteres)
    A6board.print("GET /salvar.php"); // Envia uma solicitação do tipo
    GET no protocolo HTTP para a página salvar.php
    A6board.print("?hum="); // Formata a URL, parametro hum
    A6board.print(hum); // Carrega o valor da umidade
    A6board.print("&temp="); // Formata a URL, parametro temp
    A6board.print(temp); // Carrega o valor da temperatura
    A6board.print("&volt="); // Formata a URL, parametro volt
    A6board.print(volt); // Carrega o valor da tensão da
    bateria de alimentação do GPRS
    A6board.print("&pot="); // Formata a URL, parametro pot

```

```

    A6board.print(pot); // Carrega o valor da potencia do
sinal do rádio do GPRS
    A6board.print(" HTTP/1.1"); // Protocolo utilizado para
transferencia dos dados
    A6board.print("\r\n"); // Carrier return e new line
    A6board.print("HOST: "); // Formatação da url, parametro
HOST
    A6board.print(host); // Carrega a string com o
endereço do HOST
    A6board.print("\r\n"); // Carrier Return e new line
    A6board.print("\r\n"); // Carrier Return e new line
novamente. Necessário para o HTTP

#ifdef DEBUG
    Serial.print("GET /salvar.php");
    Serial.print("?hum=");
    Serial.print(hum);
    Serial.print("&temp=");
    Serial.print(temp);
    Serial.print("&volt=");
    Serial.print(volt);
    Serial.print("&pot=");
    Serial.print(pot);
    Serial.print(" HTTP/1.1");
    Serial.print("\r\n");
    Serial.print("HOST: ");
    Serial.print(host);

    Serial.print("\r\n");
    Serial.print("\r\n");
#endif

    A6command(end_c, "HTTP/1.1", "yy", 300, 1);
    unsigned long entry = millis();
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // SOLICITA O STATUS DA CONEXAO NOVAMENTE
    A6command("AT+CIPCLOSE", "OK", "yy", 150,
1); // FECHA A CONEXAO TCP IP
    A6command("AT+CIPSTATUS", "OK", "yy", 100,
2); // SOLICITA O STATUS DA CONEXAO NOVAMENTE
    delay(100);
#ifdef DEBUG
    Serial.println("-----FIM-----");
#endif
}

```