**GOVERNO DO ESTADO DO RIO DE JANEIRO**

**SECRETARIA DE ESTADO DE CIÊNCIA E TECNOLOGIA**

**FUNDAÇÃO DE APOIO À ESCOLA TÉCNICA**

**FACULDADE DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO DE JANEIRO -**

**FAETERJ/PARACAMBI**

**CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS DE INFORMAÇÃO**

**TRABALHO DE CONCLUSÃO DE CURSO**

**MATHEUS RODRIGUES HILÁRIO E**

**MATHEUS DE SOUZA FELIX**

**SISTEMA DE MONITORAMENTO DE GÁS E QUALIDADE DO AR COM ESP8266**

**PARACAMBI, RJ**

**2021**

**MATHEUS RODRIGUES HILÁRIO E**

**MATHEUS DE SOUZA FELIX**

**SISTEMA DE MONITORAMENTO DE GÁS E QUALIDADE DO AR COM ESP8266**

Trabalho de Conclusão de curso (TCC) apresentado ao Curso Superior de Tecnologia em Sistemas de Informação da Faculdade de Educação Tecnológica do Estado do Rio de Janeiro como requisito parcial para obtenção do grau de Tecnólogo em Sistemas de Informação.

**Orientador:**Prof. Doutor Carlos Eduardo Costa Vieira.

**PARACAMBI, RJ**

**2021**

|  |
| --- |
| AR749 Hilário, Matheus Rodrigues Hilário  Felix, Matheus de Souza Felix  Sistema de monitoramento de gás e qualidade do ar com esp8266 /  Matheus Rodrigues Hilário.  Matheus de Souza Felix. - 2021  60 f.; 30 cm  Trabalho de Conclusão de Curso (Tecnólogo em Sistema de Informação) Fundação de Apoio à Escola Técnica do Estado do Rio de Janeiro. - Faculdade de Educação Tecnológica do Estado do Rio de Janeiro, campus Paracambi – FAETERJ/Paracambi, Rio de Janeiro, 2021. |

**MATHEUS RODRIGUES HILÁRIO E**

**MATHEUS DE SOUZA FELIX**

**SISTEMA DE MONITORAMENTO DE GÁS E QUALIDADE DO AR COM ESP8266**

Trabalho de Conclusão de Curso (TCC) apresentado ao Curso Superior de Tecnologia em Sistema de Informação da Faculdade de Educação Tecnológica do Estado do Rio de Janeiro como requisito parcial para obtenção do grau de Tecnólogo em Sistemas de Informação.

**TRABALHO DE CONCLUSÃO DE CURSO APROVADO EM 00/00/2021**

**BANCA EXAMINADORA**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Doutor Carlos Eduardo Costa Vieira

FAETERJ – Campus Paracambi

Orientador

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

FAETERJ – Campus Paracambi

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

FAETERJ – Campus Paracambi

A toda força de vontade de minha mãe e familiares que me ajudaram seja com uma palavra de incentivo ou com algum tipo de apoio.

Matheus Rodrigues Hilário

A toda minha família que sempre me ajudou com palavras e apoio em todo esse tempo.

Matheus de Souza Felix

**AGRADECIMENTOS – MATHEUS RODRIGUES HILÁRIO**

Agradeço primeiramente a Deus por me conceder essa honra de ser um membro com ensino superior na família.

Ao meu orientador Prof. Doutor Antônio Carlos de Oliveira pelo apoio e incentivo durante a escrita desse material.

A todos os professores da FAETERJ/Paracambi pelos ensinamentos e todos os outros que passaram pela minha vida sem eles não estaria aqui.

Ao pessoal da secretaria pelo auxilio.

A todos que de alguma forma contribuíram para a realização deste trabalho.

**AGRADECIMENTOS – MATHEUS DE SOUZA FELIX**

Agradeço a Deus por me ajudar a ser mais um membro da minha família com ensino superior.

Ao meu orientador Prof. Doutor Antônio Carlos de Oliveira pelo apoio e incentivo durante a escrita desse material.

A todos os professores que passaram em minha vida durante meus anos de estudo pelos ensinamentos passados.

A todos que de alguma forma ajudaram para a realização deste trabalho.

**RESUMO**

**HILÁRIO**, Matheus Rodrigues. **FELIX,** Matheus de Souza. SisGeCEEA Smart Things na sociedade: Sistema de Monitoramento de gás e Qualidade do ar com ESP8266. 2019. 60f. Trabalho de Conclusão de Curso(Curso Superior de Tecnologia em Sistemas de Informação).Faculdade de Educação Tecnológica do Estado do Rio de Janeiro campus, Paracambi – FAETERJ/Paracambi. 2021.

Vivemos na era digital, e com isso boa parte do que era feito a mão ou com sistemas manuais, sendo estes dependentes da intervenção humana, foi aos poucos migrando para a internet. Estamos totalmente sendo bombardeados por informações, independente de serem boas ou ruins, estamos indo para uma era onde tudo estará conectado um exemplo mais claro disso são os caminhões totalmente autônomos que já estão em funcionamento nos EUA e em outras partes do mundo. De fato isso abre novas experiências entre empresas e seus clientes, mas se isso já está acontecendo o porquê aqui no Brasil não podemos tirar proveito disso de um modo mais pessoal, digo isso com os olhos voltados para monitoramento de residências e ambientes. Sendo que ao passo que as tecnologias evoluem a distribuição dessas tecnologias tende a ficar mais fácil para todos, o lado do usuário dessas novas tecnologias fica a cada dia mais fácil e cada vez mais fácil a interação no nosso mundo. Esse trabalho propôs uma consulta rápida do usuário para saber o monitoramento de gás e a qualidade do ar em tempo real, para isso vamos usar as tecnologias Python e MySQL, e a plataforma Arduíno para construção do protótipo de monitoramento do ambiente, no qual o mesmo fará a medição e enviará os dados via protocolo HTTP para o sistema online no qual o usuário fará a consulta. Utilizaremos o Bootstrap para facilitar a construção do template, Python irá obter esses dados, gravará no banco de dados junto com a data de registro e exibirá isso para o usuário o qual o mesmo poderá se planejar para ajustes em caso de algum vazamento.

Palavras-chave: Monitoramento de Gás; Qualidade do ar; Arduíno; Python.

**ABSTRACT**

**RODRIGUES,** Matheus. **FELIX,** Matheus. Smart Things in society: Gas and Air Quality Monitoring System with ESP8266. 2019. 60f. Final Paper(Curso Superior de Tecnologia em Sistemas de Informação).Technological Education College ot The State of Rio de Janeiro, Paracambi campus – FAETERJ/Paracambi. 2021.

We live in the digital age, and with that a good part of what was done by hand or with manual systems, which were dependent on human intervention, was gradually migrating to the internet, we are totally bombarded by information, regardless of whether it is good or bad, we are going for an era where everything will be connected a more classic example of this is the fully autonomous trucks that are already in operation in the US and elsewhere, in fact this opens up new experiences between companies and their customers, but if this is already happening why here in Brazil we cannot take advantage of this in a more personal way, I say this with my eyes focused on monitoring homes and environments, whereas as technologies evolve the distribution of these technologies tends to become easier for everyone, the user side of these new technologies it becomes easier and more and more easy to interact in our world, this work proposed a quick consultation of the user io to know the gas monitoring and air quality in real time, for that we will use Python and MySQL technologies, and the Arduino platform to build the environment monitoring prototype, in which it will measure and send data via HTTP protocol to the online system in which the user will make the query. We will use Bootstrap to facilitate the construction of the template, Python will obtain this data, record it in the database together with the registration date and display it to the user, who can plan for adjustments in case of any leak.

Keywords: Gas Monitoring; Air Quality; Arduino; Python.

**L****ISTA DE FIGURAS**

Figura 1 – Logo IDE Arduíno …….……………………….…………...…………… 19

Figura 2 – Sensor MQ-135….……………………………….……………………… 20

Figura 3 - ESP8266………………………………………….……………………… 20

Figura 4 – Logo MySQL e MariaDB ……….…..………….……………………….. 23

Figura 6 – Logo HTML5 ……………………………...…….……………………… 27

Figura 7 - Inclusão da biblioteca ESP8266WiFi.h 01 …...……...…………………. 30

Figura 8 - Inclusão da biblioteca ESP8266WiFi.h 02 ……………….……………. 30

Figura 9 - Inclusão da biblioteca ESP8266WiFi.h 03 ……....……….……………. 31

Figura 10 - Inclusão da biblioteca ESP8266WiFi.h 04 ……...………..………….. 32

Figura 11 - Inclusão da biblioteca WiFiManager.h 01 …….………….…………… 32

Figura 12 - Inclusão da biblioteca WiFiManager.h 02 ……...…...………………… 33

Figura 13 - Inclusão da biblioteca WiFiManager.h 03 …….……….……………… 34

**LISTA DE TABELAS**

Tabela 1 - Preço dos Componentes do Protótipo ……….………………………. 40

**LISTA DE ABREVIATURAS E SIGLAS**

API *Application Programming Interface* – Interface de Programação de Aplicações

CRUD *Create, Read, Update, Delete* - Criar, Ler, Atualizar, Apagar.

CSS *Cascading Style**Sheets* – Folha de Estilo em Cascata.

HTML *Hypertext Markup Language* – Linguagem de Marcação de Hipertexto

HTTP *HyperText Transfer Protocol* – Protocolo de Transferência de Hipertexto.

IDE *Integrated Development Environment* - Ambiente de Desenvolvimento Integrado

IoT *Internet of Things* - Internet das Coisas

JS *JavaScript –* Linguagem de Programação JavaScript.

JSON *JavaScript Object Notation* – Notação de Objetos JavaScript

PHP *PHP Hypertext Preprocessor* – Pré-processador de Hipertexto PHP.

SGBD Sistema de Gerenciamento de Banco de Dados

SQL *Structured Query Language –* Linguagem Estruturada de Consulta.

USB Universal Serial Bus

UML *Unified Modeling Language* – Linguagem de Modelagem Unificada.

W3C *World Wide Web Consortium* – Mundo Largo em Rede Consórcio

**SUMÁRIO**

1. introdução…………………………………….……………………..…………….. 15

1.1 Objetivos…………………………………………………………………...16

1.1.1 Objetivos Gerais…………………………………………………….. 16

1.1.2 Objetivos Específicos……………………………………………….. 16

1.1.3 Justificativa…………………....……………………………………. 16

1.14 Metodologia……………………..…………………………………… 17

2. Tecnologias utilizadas………………………………..……………… 17

2.1 Plataforma Arduíno ……….……...……………………………………..... 17

2.1.1 IoT – Internet das Coisas …………………………………………..... 18

2.1.2 IDE Arduíno ……………………………………………………….... 19

2.1.3 Sensor MQ-135 …………….……………………………………...… 19

2.1.4 ESP8266 NODEMCU ………………………………………………... 20

2.2 MODELAGEM DE DADOS………………………….……………………………… 20

2.2.1 UML (Unified Modeling Language) ….……………………………..….. 20

2.2.2 Draw IO …………………………………………….…………………..….. 21

2.2.3 Diagrama de Classes e Diagrama de Caso de Uso ……………..…... 21

2.2.4 Modelo Entidade-Relacionamento ……………………………..…... 22

2.3 BANCO DE DADOS …………………………………………………….... 22

2.3.1 SQL (Structured Query Language)……….…………………………..…. 22

2.3.2 MySQL……………………………………………..……………………….… 22

***2.4 PYCHARM IDE.............................................................................................24***

2.5 LINGUAGEM DE PROGRAMAÇÃO PHP 7 E LINGUAGEM C (Embarcado)……...

2.5.1 PYTHON...............................................................................................

2.5.2 Linguagem C ………………………………………………………...

2.6 HOSPEDAGEM DO SITE …………………...………………….………………... 26

2.6.1 Xampp Server (Hospedagem local) ……………………………....……. 26

2.7 FRONT*-END DA APLICAÇÃO DO* SITE…………..……………..………………… 27

2.7.1 Linguagem de Marcação *HTML…………………………………….... 27*

2.7.2 CSS .……………………………………..……………………………………. 28

2.7.3 Bootstrap …………………………...………………………………... 28

2.7.4 JavaScript ……………………………………………………………. 30

2.7.5 Flask (framework web) …………………............................................. 30

2.7.6 JSON ………………………………………………………………… 31

***2.8 INTEGRAÇÃO COM O TELEGRAM BOOT…………………….…………………. 29***

2.8.1 Criando um boot no Telegram e obtendo o token de acesso…………... 29

2.8.2 Comunicando se com o boot do Telegram e visualizando dados……… 30

***2.9 ARQUITETURA TCP/IP E TECNOLOGIA WI-FI (802.11)……...………… 32***

2.9.1 Pilha TCP/IP …………………………………………………………… 32

2.9.2 Wi-Fi …………………………………………………………………… 34

***2.10 PLATAFORMA THINGSPEAK (PLOTANDO O GRÁFICO)…….………. 34***

***2.11 DETALHAMENTO DO MODELO DE TARIFAS …………..………….… 37***

2.11.1

**3. RESULTADOS E DISCUSSÃO ..……...………….......................…..…*.. 28***

3.1 Interfaces Gráficas ...........................................................................................…

3.1.2 Tela de Login……………………………………………………………..

3.1.3 Tela Inicial ……………………………………………………………….

3.1.4 Cadastro de Usuário …………………………………………………….

3.1.5 Cadastro de Pessoa ……………………………………………………..

3.1.6 Cadastro de Local ………………………………………………………

3.1.7 Consulta de Usuário ………………………………………………….. .

3.1.8 Consulta de Pessoa ……………………………………………………

3.1.9 Consulta de Local …………………………………………………….

3.1.10 Edição de Usuário

3.1.11 Edição de Pessoa

3.1.12 Edição de Local

**4. CONSIDERAÇÕES FINAIS ………………………………………………**

**5. REFERÊNCIAS ………………………………………………………….**

**1. INTRODUÇÃO**

A internet desde seus primórdios veio com um único e só objetivo que era conectar máquinas, centros militares e algumas instituições acadêmicas, devido ao medo de que uma bomba nuclear pudesse cair no centro de Nova Iorque, desde esses tempos muita coisa mudou, temos computadores mais potentes capazes de processamento absurdo de dados, celulares que antes você só usava para ligações se tornaram verdadeiros computadores portáteis, e a internet se tornou uma via digital onde quase tudo esta lá, desde comércio de entregas até compras e pagamentos *online*. Enfim é inegável que não pensar nesse século como a era da informação, estamos indo em direção onde os objetos e vestimentas estarão conectadas, mas para isso depende-se de uma infraestrutura adequada, novos padrões, normas e protocolos para que tudo funcione em conjunto.

Pensando nesse contexto de interconectividade, indo desde carros-autonomos, IA (inteligência artificial) e IoT (*Internet of Things* – Internet das Coisas) olhamos para nossas casas com o intuito de aplicar na vida real algo que fosse útil, rascunhamos então em uma particularidade e algo em que todos tenham contato no dia a dia, todas as pessoas tem contato continuo com gases fornecidos pelas empresas distribuidoras (gás residencial) ou no meio ambiente, sendo que o ninguém não possui total controle do quanto estamos em contato com esses gases e nem sabemos se estamos próximos de algum problema por conta desses contato com esses gases, pensando nisso o trabalho aqui proposto é de oferecer um sistema de monitoramento de gás e qualidade do ar, deixando o usuário mais consciente sobre o nível de gás no ambiente onde ele está e qual o impacto disso em relação a ele.

No projeto será composto por um *hardware* (plataforma Arduíno) que fará verificações da quantidade de nível de gás no ambiente, e outra parte será feita através de um *software* (Python + MySQL) no qual estará em contato direto com o *hardware* recebendo essas informações do nível de gás e mostrando qual o risco de explosão ou morte por sufocação.

* 1. **Objetivos**

**1.1.1 Objetivo Geral**

Este projeto tem como objetivo disponibilizar as informações de forma *online* (*web*) para o usuário que queiram saber o nível de gás no ambiente, para que caso o nível esteja alto o usuário possa verificar e corrigir o defeito, de forma que impactará em uma segurança contra possíveis vazamentos de gás.

**1.1.2 Objetivos Específicos**

* Criar um sistema *web* para exibir as informações do sistema de verificação do nível de gás.
* Saber o qual o nível de gás com o intuito de evitar riscos de explosão ou morte.

**1.2 Justificativa**

Dar uma maneira de comparar através do uso do protótipo o nível de gás em um ambiente e qual o nível de preocupação com os dados obtidos via sistema *web*, além de facilitar na compreensão os dados ali obtidos de forma clara e simples, sendo assim também possível consultar esses dados em tempo real e de forma remota, visto que os dados usam a infraestrutura de serviços da própria *web* e protocolos já em uso.

Sendo assim se justifica a necessidade de ter uma fonte de averiguação do nível de gás, visto que com níveis muito altos o risco de explosões e mortes por falta de O2 ou por inalação de CO2 são altos, tendo também uma forma de prevenir contra possíveis deteriorações do tempo em válvulas, encanamentos visto que com a menor mudança nos dados o usuário já ficaria sabendo de possíveis vazamentos.

**1.3 Metodologia**

Antes do desenvolvimento propriamente dito do sistema, pesquisamos sobre algum sistema para uso em residências, depois de algumas discussões chegamos à conclusão da criação de um sistema mais em conta financeiramente falando, mais leve para o usuário final e com peças fáceis de encontrar, seja na *internet* ou em alguma loja de eletrônica perto de casa.

As etapas posteriores para a configuração básica do sistema, foi o estudo de algumas tecnologias usadas na rede no caso os protocolos e tecnologias atuais, o uso da linguagem de programação *web* que é o Python e na questão de armazenagem dos dados usamos o MySQL.

Após isso vimos mais afundo como funciona o modelo TCP/IP que é fundamental nesse projeto, e as ferramentas necessárias para dar vida ao desenvolvimento da aplicação em si.

**2. TECNOLOGIAS UTILIZADAS**

Nessa etapa veremos sobre as tecnologias e conceitos básicos utilizados para a construção do protótipo.

**2.1 IoT – Internet das coisas**

A expressão IoT é utilizada para designar a conectividade e interação entre vários tipos de objetos do dia a dia, sensíveis à *internet*. Fazem parte desse conceito os dispositivos de nosso cotidiano que são equipados com “sensores capazes de captar aspectos do mundo real, como por exemplo temperatura, umidade e presença, e enviá-los a centrais que recebem estas informações e as utilizam de forma inteligente”. A sigla refere-se a um mundo onde objetos e pessoas, assim como dados e ambientes virtuais, interagem uns com os outros no espaço e no tempo. (MAGRANI, 2018, p. 44).

Do ponto de vista da normalização técnica, a IoT pode ser vista como uma infraestrutura global voltada para a era digital, permitindo serviços avançados por meio da interconexão de coisas (físicas e virtuais) com base nas tecnologias de informação e comunicação interoperáveis existentes e em constante evolução. (MAGRANI, 2018, p. 45).

**2.1.1 Plataforma Arduíno**

Para começar a falar do Arduino é preciso esclarecer que não se trata de um microcontrolador, mas, sim, de uma plataforma de desenvolvimento de sistemas embarcados de baixo custo aberta e livre.

As referências a Arduino consideram, normalmente, uma placa integrada com um microcontrolador e suas interfaces de entrada e saída, alimentação e comunicação. O módulo mais comum é o Arduino Uno, basicamente, um kit de desenvolvimento para iniciantes. Há diversos outros módulos Arduino, entre os quais, o Arduino Nano, módulo mais voltado para a produção de sistemas embarcados, além de diversos outros módulos compostos de sensores, atuadores e comunicação. (OLIVEIRA, 2017, p.48).

**2.1.2 IDE Arduíno**

# Uma das grandes vantagens da plataforma Arduíno está no seu ambiente de desenvolvimento, que usa uma linguagem baseada no C/C++, linguagem bem difundida, usando uma estrutura simples. Por isso, mesmo pessoas sem conhecimento algum em programação conseguem, com pouco estudo, elaborar programas rapidamente. (Mota, Allan. O que é Arduino e como funciona?. Disponível em < https://portal.vidadesilicio.com.br/o-que-e-arduino-e-como-funciona/ >).

# 2.1.3 Sensor MQ-135 (Sensor de Gás MQ-135 para Gases Tóxicos)

# O Sensor de Gás MQ-135 é um módulo capaz de detectar vários tipos de gases tóxicos como amônia, dióxido de carbono, benzeno, óxido nítrico, e também fumaça ou álcool. (Candido, [Gradimilo.](https://portal.vidadesilicio.com.br/author/gradimilo-candido/) Sensor de Gás MQ-135 e a Família de sensores MQ. Disponível em < https://portal.vidadesilicio.com.br/sensor-de-gas-mq-135/>).

Usando um sensor de gás MQ-135 é possível montar sistemas de monitoramento e alarmes para o controle da concentração de gás no ambiente, e com um *trimpot* na placa é permitido ajustes no nível de sensibilidade do sensor. Se outro sistema de sua preferência ou um microcontrolador for usado em conjunto, é possível acionar lâmpadas, sirenes, relés ou enviar alarmes pela *internet* quando um gás tóxico for detectado em níveis elevados.

**2.1.4 ESP8266 NODEMCU**

O microcontrolador ESP8266, produzido pela fabricante chinesa Espressif, é um microcontrolador de 32 bits que inclui um núcleo microprocessado Tensilica L106, que funciona na frequência-padrão de 80 MHz, podendo chegar a 160 MHz. O processamento da pilha de protocolos WiFi usa 20% da capacidade de processamento desse processador. Com isso, 80% dessa capacidade pode ser utilizada em aplicações do usuário. A memória disponível para os dados dos programas tem cerca de 50 kB, já descontado o espaço necessário para o padrão WiFi. A memória disponível para o programa principal é de 4 MB, em área acessível à atualização em funcionamento, também conhecida como OTA (Over-The-Air); além de mais 512 kB que não contam com esse recurso e só podem ser atualizados via cabo, em procedimento de atualização. (OLIVEIRA, 2017, p.51 e p. 52).

O ESP8266 foi escolhido como base para esse TCC devido a três fatores:

Preço: existem diversas placas para o desenvolvimento, das mais baratas produzidas em solo chinês e que podem ser compradas *online* ou placas vendidas em sites com domínio .br a preços mais acessíveis e um tempo de entrega menor, tornando um modelo de baixo custo, assim tornando-se um modelo viável para IoT e suas aplicações.

Módulos/Sensores: Por ser baseada na plataforma Arduíno ela possui compatibilidade com a maioria dos módulos/sensores que já se encontram disponível e de fácil acesso ao público, dando oportunidade para se testar novos produtos de forma rápida, simples e barata, tudo o que propôs o IoT.

Ambiente de Desenvolvimento: O ESP8266 possui um ambiente baseado na linguagem Lua, apesar de ser possível usar a Python, também podemos usar a IDE do Arduíno sendo essa usando como base a linguagem *Wiring* muito próxima do C trazendo muitos benefícios, pois esta IDE já vem sendo usada a mais tempo pela comunidade *Open-Source* e se encontra no estado maduro tanto em termos de correção de *bugs* como também possuem uma vasta gama de bibliotecas para os mais diversos tipos de projetos.

**2.2 Modelagem de dados**

**2.2.1 UML *(Unified Modeling Language)***

A UML - *Unified Modeling Language* ou Linguagem de Modelagem Unificada, é uma ferramenta visual utilizada para modelar *softwares* baseados no paradigma de orientação a objetos. É uma linguagem de modelagem de propósito geral que pode ser aplicada a todos os domínios de aplicação. Esta linguagem é atualmente a linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de *software*. (GUEDES. 2011, p. 19).

Foi usado como condutor para a criação dos dois diagramas existentes nesse trabalho: o diagrama de classes e o diagrama de caso de uso.

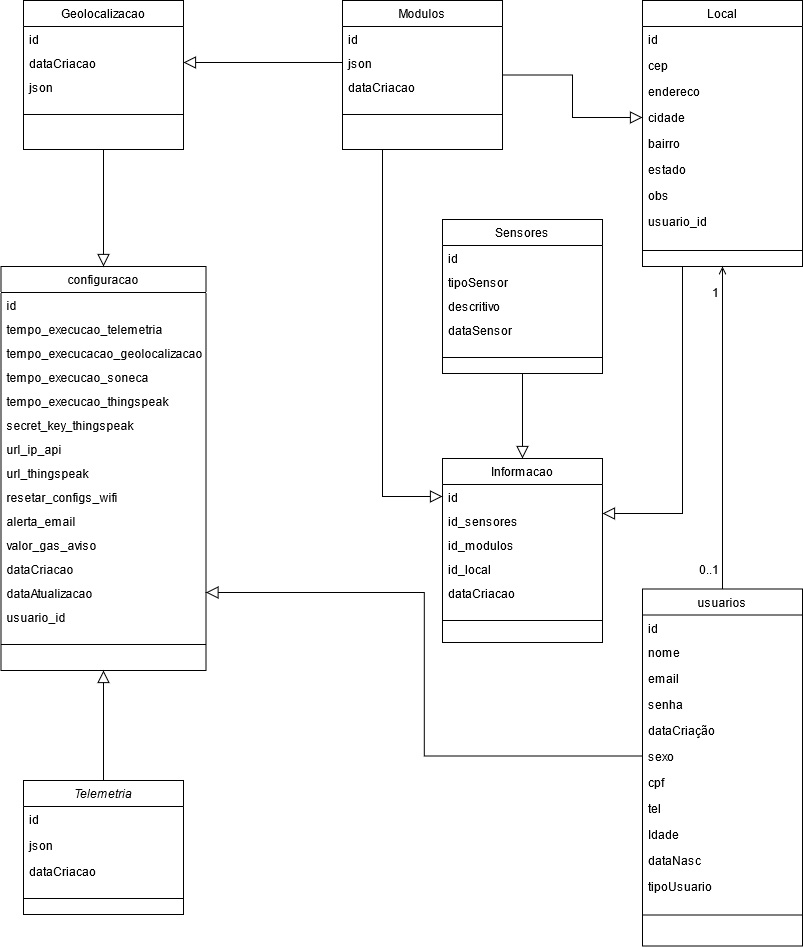
**2.2.2 *Draw.io***

*Draw.io* é um editor gráfico *online* onde é possível a construção de modelagens, diagramas e gráficos sem qualquer tipo de instalação de *softwares* no computador.

Os diagramas apresentados nesse trabalho foram feitos usando essa ferramenta.

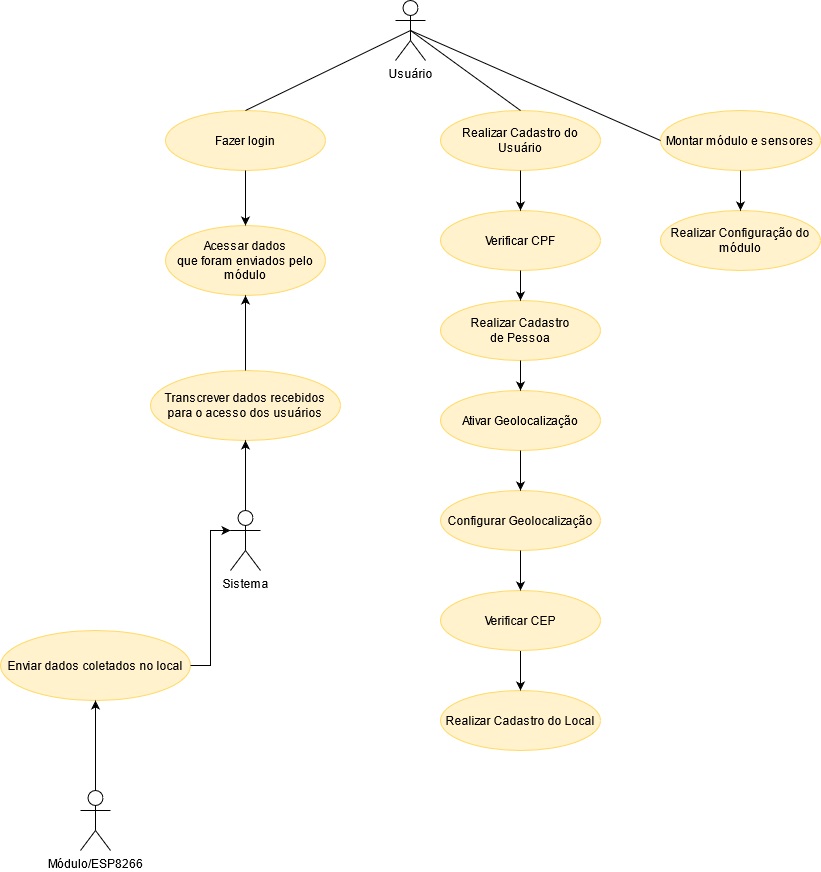
**2.2.3 Diagrama de Classes e Diagrama de Caso de Uso**

**2.2.3.1 Diagrama de Classes**

****

Fonte: Autoria Própria

**2.2.3.2 Diagrama de Caso de Uso**

****

Fonte: Autoria Própria

* 1. **SQL (*STRUCTURED QUERY LANGUAGE*)**

**SQL** é uma linguagem padrão para trabalhar com **bancos de dados relacionais**. Ela é uma linguagem declarativa e que não necessita de profundos conhecimentos de programação para que alguém possa começar a escrever ***queries***, as consultas e pedidos, que trazem resultados de acordo com o que você está buscando. (SILVEIRA, Paulo. O que é SQL?. Disponível em: <https://www.alura.com.br/artigos/o-que-e-sql>).

**2.3.1 MySQL**

O MySQL é um servidor e gerenciador de banco de dados (SGBD) relacional, de licença dupla (sendo uma delas *software* livre), projetado inicialmente para trabalhar com aplicações de pequeno e médio porte, mas hoje atendendo a aplicações de grande porte e com mais vantagens do que seus concorrentes. Possui todas as características que um banco de dados de grande porte precisa, sendo reconhecido por algumas entidades como o banco de dados *open source* com maior capacidade para concorrer com programas similares de código fechado, tais como SQL Server (da Microsoft) e Oracle. (MILANI, 2007, p.22).

**2.4 PyCharm IDE**

O PyCharm IDE é um software feito para desenvolvedores, com propósito de oferecer todas as ferramentas que um programador precisar para a construção de programas de forma produtiva usando Python.

**2.5 Linguagens Python e C**

**2.5.1 Python**

Python é uma linguagem de programação de alto nível — ou High Level Language —, dinâmica, interpretada, modular, multiplataforma e **orientada a objetos** — uma forma específica de organizar *softwares* onde, a grosso modo, os procedimentos estão submetidos às classes, o que possibilita maior controle e estabilidade de códigos para projetos de grandes proporções.

Por ser uma linguagem de sintaxe relativamente simples e de fácil compreensão, ganhou popularidade entre profissionais da indústria tecnológica que não são especificamente programadores, como engenheiros, matemáticos, cientistas de dados, pesquisadores e outros.

Um de seus maiores atrativos é possuir um grande número de bibliotecas, nativas e de terceiros, tornando-a muito difundida e útil em uma grande variedade de setores dentro de desenvolvimento *web*, e também em áreas como análise de dados, *machine* *learning* e IA. (ROVEDA, Ugo. O que é Pyhton, para que serve e por que aprender?. Disponível em: <https://kenzie.com.br/blog/o-que-e-python/>).

**2.5.2 Linguagem C**

A linguagem C foi criada por Dennis Ritchie nos laboratórios da Bell Telephone em 1972.

C foi criada com um propósito: ser usada no desenvolvimento de uma nova versão do sistema operacional *Unix*. A primeira versão do *Unix* utilizava *Assembly*. Então podemos dizer que desde o principio C foi uma linguagem criada por programadores para programadores.

A linguagem C é considerada de propósito geral, ou seja, é uma linguagem capaz de ser usada para praticamente qualquer tipo de projeto. É extremamente portável, um programa escrito em linguagem C pode ser facilmente usado em qualquer plataforma.

Utilizando linguagem C podemos criar sistemas operacionais, aplicativos de todos os tipos, *drivers* e outros controladores de dispositivos, programar microcontroladores, etc.

Além de toda essa flexibilidade, C é capaz de gerar programas extremamente rápidos em tempo de execução, possui uma sintaxe simples e poderosa, com instruções de alto nível.

A linguagem C influenciou de forma direta muitas linguagens como C++, Java, C#, Objective C, e muitas outras linguagens de programação têm sua sintaxe e estruturas influenciadas por C. (CASAVELLA, Eduardo. O que é Linguagem C?. Disponível em: <http://linguagemc.com.br/o-que-e-linguagem-c/>).

**2.6 Hospedagem do *site***

**2.6.1 Xampp**

Xampp é um *software* gratuito, que contém os principais servidores com código aberto, incluindo banco de dados para MySQL, Apache e suporte as linguagens de programação *web*. Nele é possível simular um servidor *web* de forma local sem transtornos.

**2.7 *Front-end* da aplicação do *site***

**2.7.1 HTML**

HTML é a sigla em inglês para *HyperText Markup Language*, que, em português, significa linguagem para marcação de hipertexto.

Hipertexto é todo texto inserido em um documento para a *web* e que tem como principal característica a possibilidade de se interligar outros documentos da *web* com uso dos nossos já conhecidos *links*, presentes nas páginas dos *sites* que estamos acostumados a visitar. Então, todo o conteúdo textual que você vê em uma página de um *site* é um hipertexto, assim como imagens, vídeos, gráficos, sons e conteúdos não textuais em geral são chamados de hipermídia. (SILVA, 2015, p. 19).

**2.7.2 CSS**

**CSS** (Cascading Style Sheets **ou Folhas de Estilo em Cascata)** é uma linguagem de estilo (en-US) usada para descrever a apresentação de um documento escrito em HTML ou em XML (incluindo várias linguagens em XML como SVG, MathML ou XHTML). O CSS descreve como elementos são mostrados na tela, no papel, na fala ou em outras mídias. (CSS. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/CSS>).

**2.7.3 Bootstrap**

Bootstrap é o mais popular *framework* JavaScript, HTML e CSS para desenvolvimento de *sites* e aplicações *web* responsivas e alinhadas com a filosofia *mobile first*. Torna o desenvolvimento *front-end* muito mais rápido e fácil. Indicado para desenvolvedores de todos os níveis de conhecimento, dispositivos de todos os tipos e projetos de todos os tamanhos. (SILVA, 2015, p. 20).

**2.7.4 JavaScript**

JavaScript é uma linguagem de programação que permite a você implementar itens complexos em páginas *web* — toda vez que uma página da *web* faz mais do que simplesmente mostrar a você informação estática — mostrando conteúdo que se atualiza em um intervalo de tempo, mapas interativos ou gráficos 2D/3D animados, etc. — você pode apostar que o JavaScript provavelmente está envolvido.

(O que é JavaScript. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript.html>).

**2.7.5 Flask (*framework web*)**

Flask é um *framework* pequeno para os padrões de *frameworks*, ele chega a ser chamado de "*microframework*". Apesar de pequeno o Flask, ele foi moldado para ser extensível e tem um núcleo de serviços básicos bem sólido, e suas extensões são capazes de entregar o restante que lhe falta na forma básica.

O Flask tem três dependências principais. Os subsistemas de roteamento, depuração e WSGI (*Web Server Gateway Interface*, ou Interface de *Gateway* de Servidor *Web*) são do Werkzeug (http://werkzeug.pocoo.org/); o suporte para *templates* é oferecido pelo Jinja2 (http://jinja.pocoo.org/), e a integração com a linha de comando pelo Click (http://click.pocoo.org). O autor de todas essas dependências é Armin Ronacher, autor do Flask. (GRINBERG, 2018, p. 23).

**2.7.6 JSON**

JSON (JavaScript *Object Notation* - Notação de Objetos *JavaScript*) é uma formatação leve de troca de dados. Para seres humanos, é fácil de ler e escrever. Para máquinas, é fácil de interpretar e gerar. Está baseado em um subconjunto da linguagem de programação JavaScript, *Standard* ECMA-262 3a Edição -Dezembro - 1999. JSON é em formato texto e completamente independente de linguagem, pois usa convenções que são familiares às linguagens C e familiares, incluindo C++, C#, Java, JavaScript, Perl, Python e muitas outras. Estas propriedades fazem com que JSON seja um formato ideal de troca de dados. (Introdução ao JSON. Disponível em: <https://www.json.org/json-pt.html>).

**2.7.7 API**

API é um conjunto de rotinas e padrões de programação para acesso a um aplicativo de *software* ou plataforma baseado na *web*. A sigla API refere-se ao termo em inglês "*Application Programming Interface*" que significa em tradução para o português "Interface de Programação de Aplicativos".

Uma API é criada quando uma empresa de *software* tem a intenção de que outros criadores de *software* desenvolvam produtos associados ao seu serviço. Existem vários deles que disponibilizam seus códigos e instruções para serem usados em outros sites da maneira mais conveniente para seus usuários. O [Google](https://canaltech.com.br/empresa/google/) Maps é um dos grandes exemplos na área de APIs. Por meio de seu código original, muitos outros sites e aplicações utilizam os dados do Google Maps adaptando-o da melhor forma a fim de utilizar esse serviço. (O que é API?. Disponível em: <https://canaltech.com.br/software/o-que-e-api/>).

Usando nosso sistema apresentado neste trabalho como exemplo, é possível a visualização da residência onde a parte física do projeto está alocada, usando o mapa do Google Maps dentro do próprio sistema. Nesse projeto estaremos usando as APIs: Google Maps, viaCEP, ip-API, Thinkspeak.

**2.7.8 Protocolo HTTP**

**HTTP** é um protocolo (*protocol*) que permite a obtenção de recursos, como documentos HTML. É a base de qualquer troca de dados na *Web* e um protocolo cliente-servidor, o que significa que as requisições são iniciadas pelo destinatário, geralmente um navegador da *Web*. Um documento completo é reconstruído a partir dos diferentes sub-documentos obtidos, como por exemplo texto, descrição do layout, imagens, vídeos, *scripts* e muito mais.(Uma visão geral do HTTP. Disponível em: **<**<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview>>).

**2.8 Arquitetura TCP/IP e tecnologia WiFi**

**2.8.1 Arquitetura TCP/IP**

Em virtude da complexidade dos sistemas envolvidos, os conceitos de redes de computadores são comumente divididos em camadas que encapsulam um conjunto de funcionalidades e oferecem serviços necessários à conectividade. A ISO (*International Organization for Standardization* – Organização Internacional de Padronização) propôs o modelo OSI com sete camadas para agrupar as funcionalidades de rede: Aplicação, Apresentação, Sessão, Transporte, Rede, Enlace e Física. Embora haja algumas implementações com o modelo OSI, na prática, o modelo TCP/IP, criado pelo Departamento de Defesa dos Estados Unidos, se tornou muito mais popular. Nesse modelo não existem as camadas de Sessão e Apresentação. Além disso, as funções das Camadas Enlace e Física não estão bem definidas; ficam a cargo de cada implementação. (OLIVEIRA, 2017, p.20).

No modelo TCP/IP, as camadas Aplicação, Transporte e Rede estão bem definidas. E o modelo considera que haja protocolos que façam a entrega dos pacotes na rede local, abaixo da camada de Rede. Para realizar essa função, existem dezenas de protocolos, com destaque para *Ethernet*, WiFi, PPP (*Point-to-Point Protocol* – Protocolo Ponto a Ponto) e as redes de celulares 2G/3G/4G, que se tornaram populares com o desenvolvimento da internet. (OLIVEIRA, 2017, p.21).

**2.8.2WiFi**

O padrão IEEE 802.11 ficou internacionalmente conhecido e popularizado como WiFi, acrônimo para *Wireless Fidelity*, em alusão à *HiFi*, termo usado para amplificadores de áudio de alta qualidade. Sua proposta, de conectar dispositivos em redes locais sem fio, sempre foi questionável em vários aspectos, mas isso não impediu que esse padrão se tornasse tão popular quanto a *internet*. Apresenta várias versões (a/b/g/n/ac) e tem, ainda, muitos problemas a ser solucionados para que não seja substituído por outra tecnologia que atenda melhor à evolução de Internet das Coisas. (OLIVEIRA, 2017, p.27 e p.28).

Na pilha de protocolos TCP/IP, o padrão WiFi ocupa a parte de baixo, respondendo pelas funções de subcamada física, acesso ao meio e ligação de dados. Em princípio, os dispositivos conectados à uma mesma rede WiFi local poderiam se conectar diretamente, sem a necessidade das demais camadas TCP/IP, mas, na prática, não existem implementações populares de protocolos de comunicação WiFi sem a pilha TCP/IP, o que justifica considerar WiFi totalmente integrado ao TCP/IP. (OLIVEIRA, 2017, p.28).

**2.9 Plataforma ThingSpeak**

*ThingSpeak.com* é um serviço de IoT na nuvem que possibilita fazer registro de informações recebidas pelos dispositivos IoT, além de disponibilizar diversas análises e ações sobre esses dados. As análises incluem análise de dados por *softwares* matemáticos, visualização de gráficos indicativos do comportamento dos dados e plugins para criar arquivos HTML, *JavaScript* ou CSS personalizados. As ações incluem o envio de *tweeters*, que podem ser encaminhados para comandar outras plataformas, além de controles temporizados, reações a eventos e comandos via protocolo HTTP. (OLIVEIRA, 2017, p.163 p.164).

**2.10 Integração com o Telegram bot**

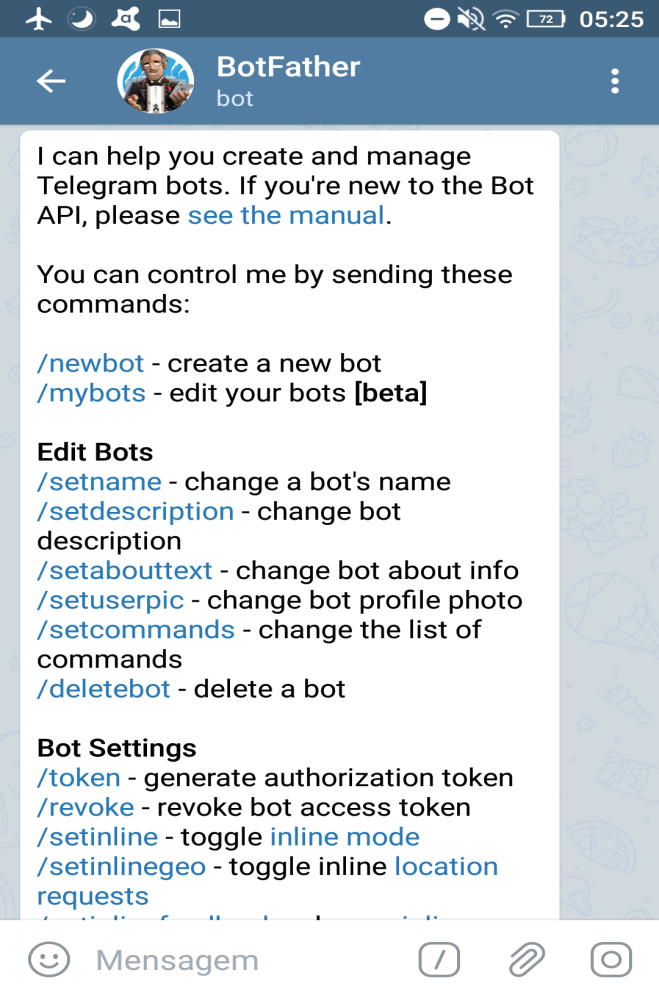
**2.10.1 Criando um bot no Telegram e obtendo o token de acesso**

**1° passo →** Abra o aplicativo do Telegram e no campo de busca digite “*BotFather*” .

****

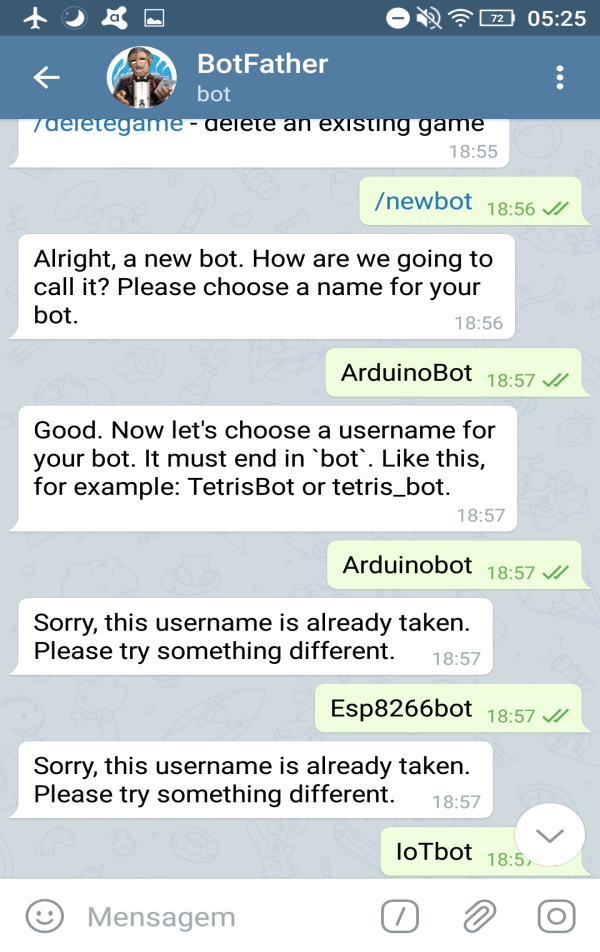
Fonte: Autoria Própria

**2° passo →** Ao adicionar o *BotFather* será apresentada a tela de boas vindas. Clique em *START* ou digite /*start* para começar, será apresentado um menu de comandos.

****

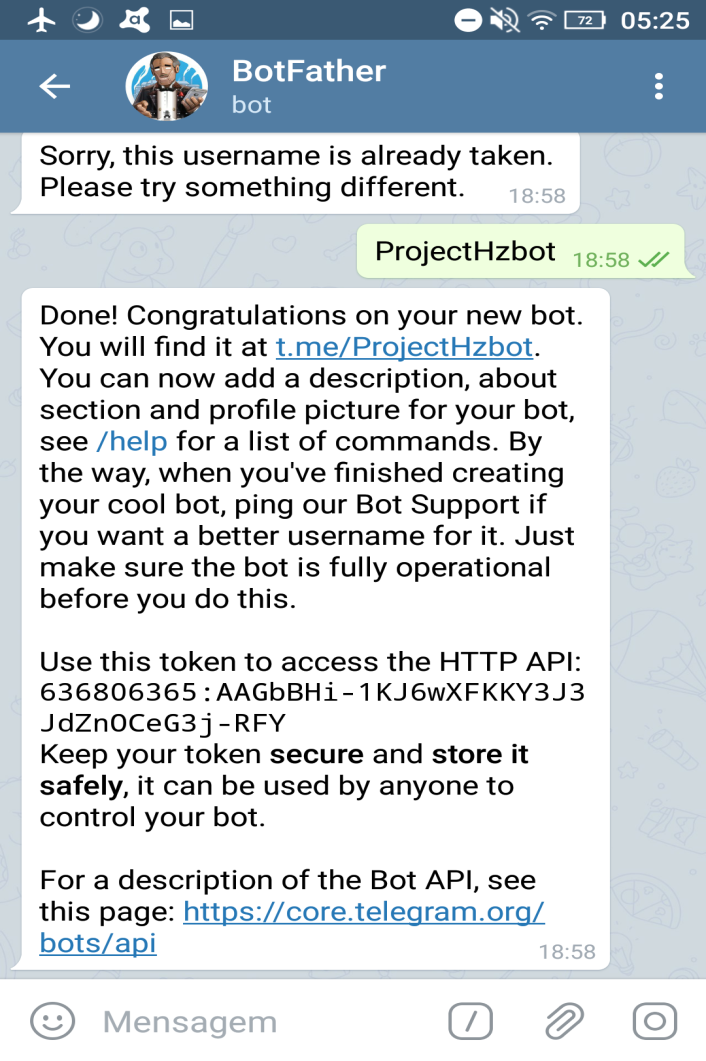
Fonte: Autoria Própria

**3° passo** → Digite /*newbot* para criar um novo bot, logo em seguida aparecerá uma tela de confirmação dizendo se é possível cadastrar seu *bot* com esse nome.



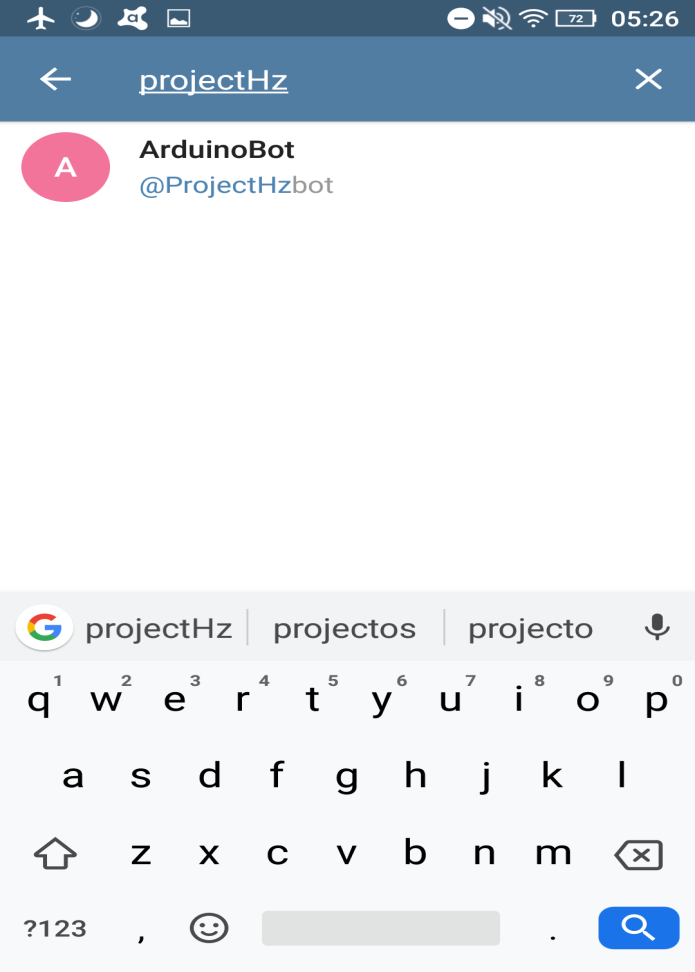
Fonte: Autoria Própria

**4° passo** → Se o nome do seu *bot* for aceito, aparecerá uma tela com seu *token* a HTTP API. Esse *token* é de acesso único e uma etapa de segurança quando formos nos conectar a HTTP API no servidor do Telegram.



Fonte: Autoria Própria

**5° passo** → Aparecerá o nome do seu *bot* na tela inicial do aplicativo.



Fonte: Autoria Própria

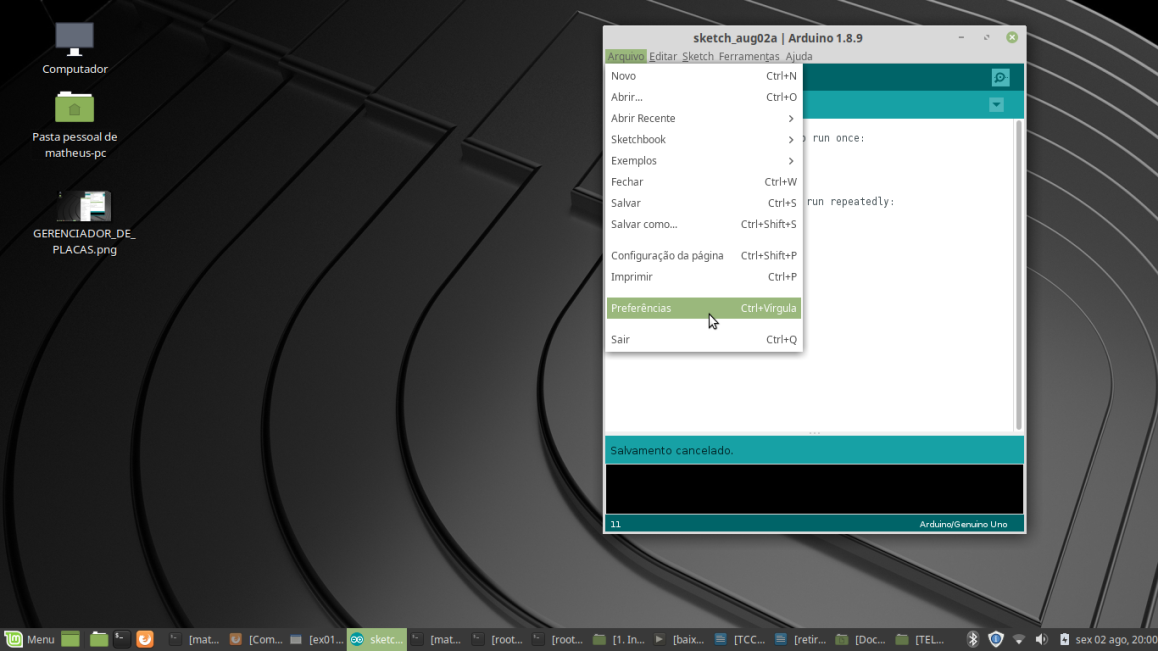
**2.10.2 Comunicando se com o bot do Telegram e visualizando dados**

Agora que o *bot* foi configurado você poderá instruí-lo conforme sua necessidade. Para informações sobre quais comandos são validos digite ‘opção’ ou 0 para exibir o menu.

**2.11** I**nstalação de bibliotecas na IDE Arduíno**

**2.11.1 Inclusão da biblioteca ESP8266WiFi.h**

**1° passo:** Abra a IDE Arduíno, clique em preferências.

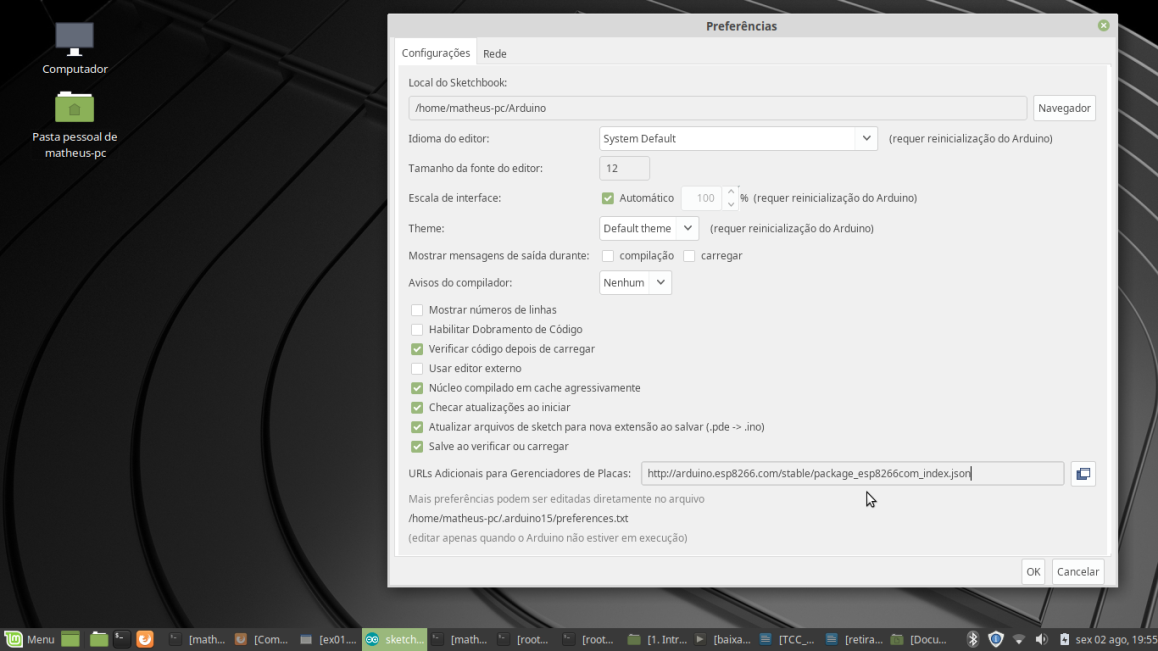


Fo

Fonte: Autoria Própria

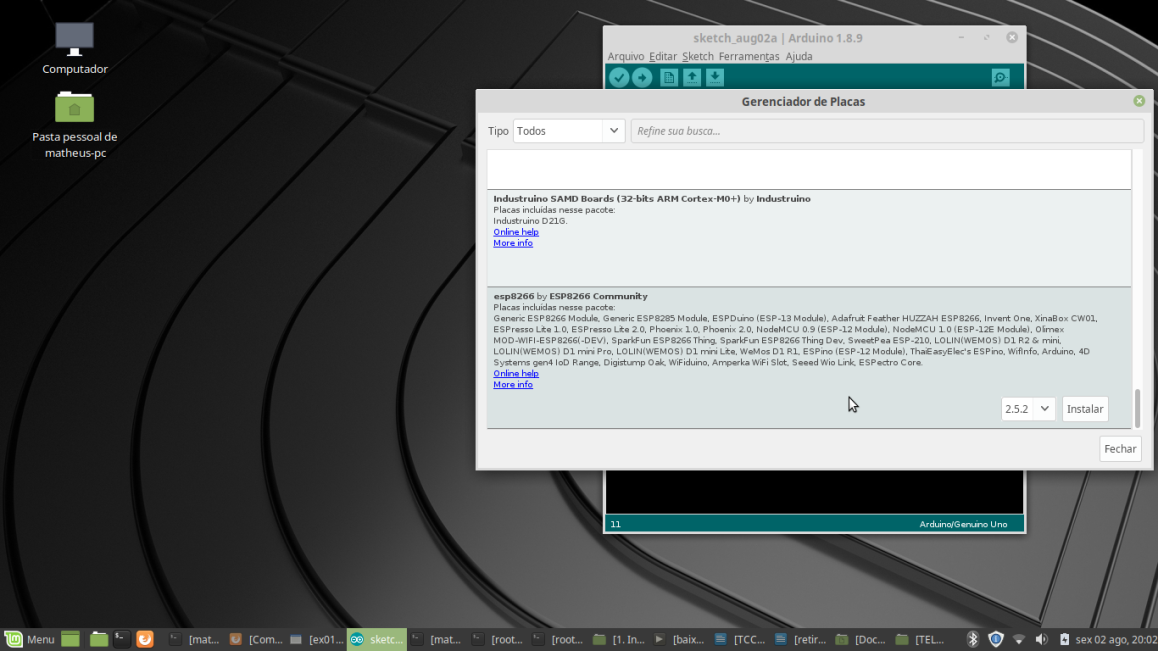
**2° passo:** Copie e cole o *link* abaixo, no campo URL Adicionais para Gerenciadores de Placas:

http://arduino.esp8266.com/stable/package\_esp8266com\_index.json



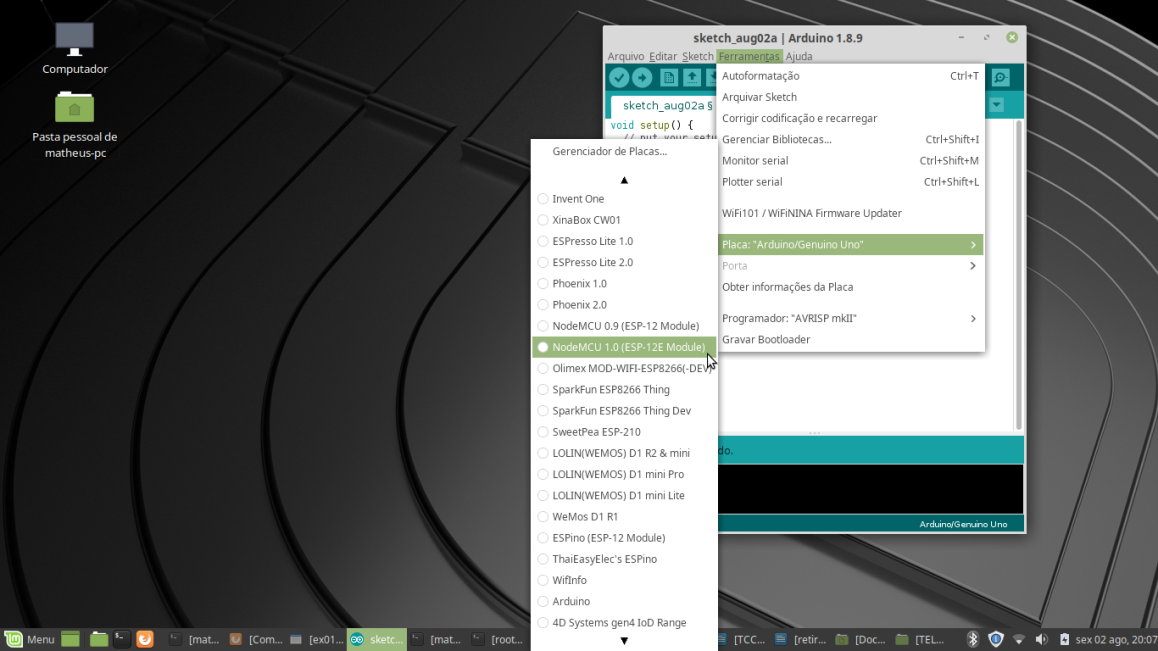
Fonte: Autoria Própria

**3° passo:** Clique em *OK*, você retornará para a tela principal do IDE Arduino, clique em Ferramentas → Placas → Gerenciador de Placas, use a barra de busca e filtre por esp8266 você encontrara o resultado igual a da imagem selecione a versão 2.5.2 ou mais recente da ESP8266 *by Community* e clique em instalar.



Fonte: Autoria Própria

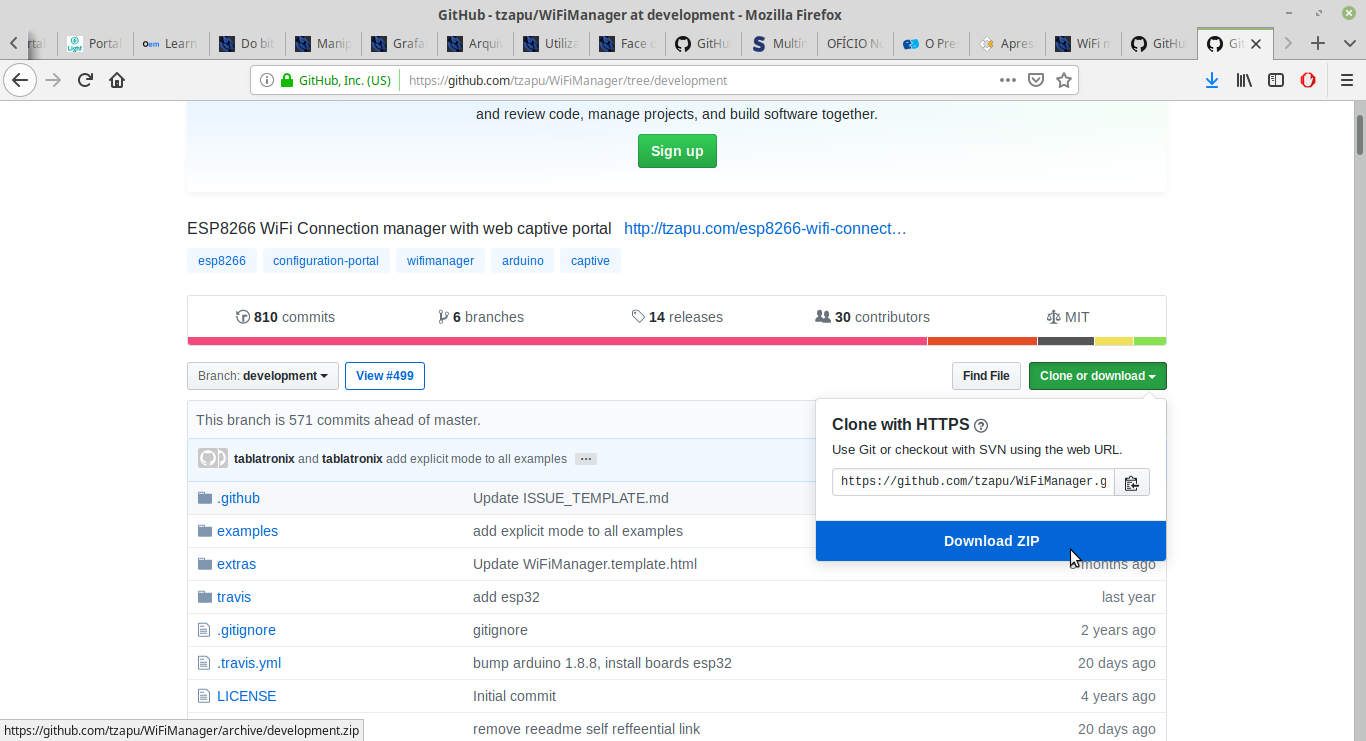
**4° passo:** Clique em fechar e em alguns minutos aparecerá em Ferramentas → Portas: NodeMCU1.0 (ESP-12E Module) selecione essa placa e a porta e comece a codar. Depois dessa configuração outras bibliotecas estarão disponíveis para serem utilizadas são algumas delas: WiFiClient.h, ESP8266WebServer.h, WiFiServer.h entre outras.



Fonte: Autoria Própria

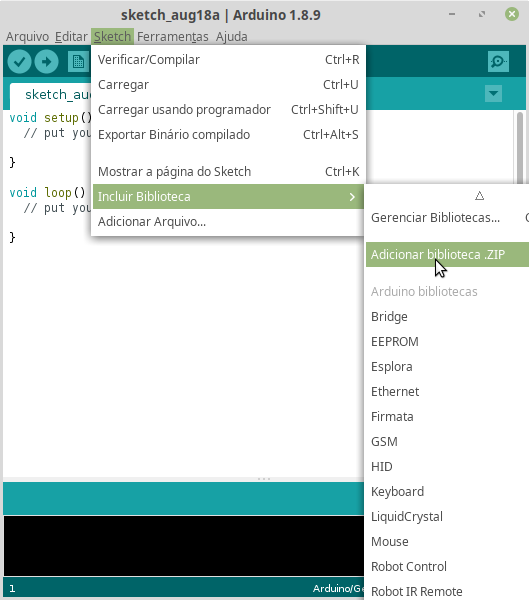
**2.11.2 Inclusão da biblioteca WiFiManager.h (Menu escolhe WiFi Cliente)**

**1° passo:** Vá em https://github.com/tzapu/WiFiManager clique em “Clone *or Download*” em seguida clique em “*Download* ZIP” em segundos será feito o *download* do ZIP.



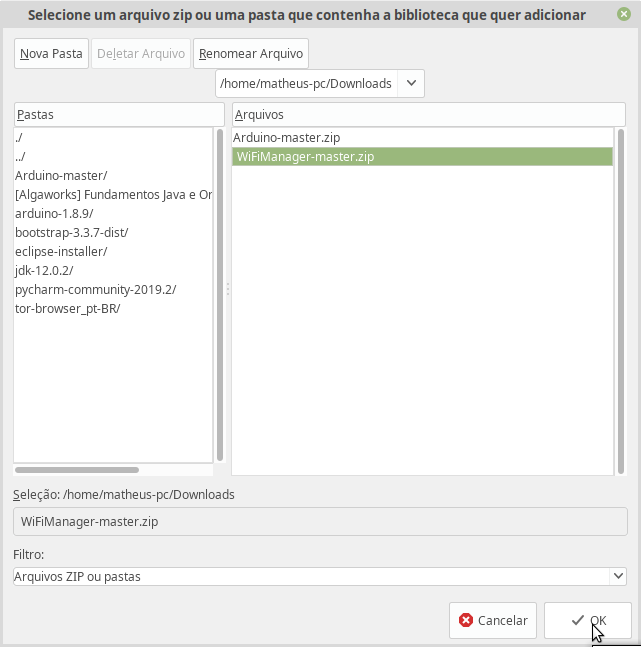
Fonte: Autoria Própria

1. **° passo:** Vá até a IDE Arduino, clique em “*Sketch*” → “Incluir Biblioteca” → “Adicionar biblioteca .ZIP” .



Fonte: Autoria Própria

**3 ° passo:** Depois selecione o diretório .ZIP “WiFiManager-master.ZIP” e clique em “OK” .



Fonte: Autoria Própria

**4° passo:** Agora você poderá incluir a biblioteca no seu projeto chamando dessa maneira → #include<WiFiManager.h> .

|  |  |  |  |
| --- | --- | --- | --- |
| **TABELA DE PREÇO DOS COMPONENTES DO PROTÓTIPO** | | | |
| **NOME DO COMPONENTE** | **PREÇO** | **QUANTIDADE** |  |
| ESP8266 NODEMCU V1 | R$ 45,00 | 1 |  |
| SENSOR SCT 013 (100A) | R$40,00 | 1 |  |
| PROTOBOARD 830 FUROS | R$12,00 | 2 |  |
| JUMPERS MACHO + FÊMEA | R$0,10 | 5 MACHOS + 5 FÊMEAS |  |
| RESISTORES DE 10K | R$0,15 | 2 |  |
| CAPACITOR ELETROLÍTICO 10pF | R$0,20 | 1 |  |
| RESISTOR DE 35 Ohms | R$0,10 | 1 |  |
| TOTAL | R$97,55 |  |  |

Fonte: Autoria Própria

**5. REFERENCIAS**

GUEDES, Gilleanes Thorwald Araujo. **UML 2: Uma Abordagem Prática.** 2° ed.São Paulo: Novatec Editora Ltda, 2011.

Oliveira, Sérgio de. **Internet das coisas com ESP8266, Arduíno e Raspberry PI**. São Paulo: Novatec Editora Ltda, 2017

Magrani, Eduardo. **A Internet das coisas**. Rio de Janeiro: Editora FGV, 2018.

Milani, André. **MySQL - Guia do programador**. São Paulo: Novatec Editora Ltda, 2007.

Silva, Maurício Samy. **Fundamentos de HTML5 e CSS3**. São Paulo: Novatec Editora Ltda, 2015.

Silva, Maurício Samy. **Bootstrap 3.3.5 Aprenda a usar o framework Bootstrap para criar layouts CSS complexos e responsivos**. São Paulo: Novatec Editora Ltda, 2015.

Grinberg, Miguel. **Desenvolvimento web com Flask Desenvolvendo aplicações web com Python**. Direitos de tradução. São Paulo: Novatec Editora Ltda, 2015.

Mota, Allan. **O que é Arduino e como funciona?** 2021. Disponível em <https://portal.vidadesilicio.com.br/o-que-e-arduino-e-como-funciona/>. Acesso em: 17 agosto. 2021.

Candido, [Gradimilo.](https://portal.vidadesilicio.com.br/author/gradimilo-candido/) **Sensor de Gás MQ-135 e a Família de sensores MQ** 2017. Disponível em < https://portal.vidadesilicio.com.br/sensor-de-gas-mq-135/>. Acesso em: 19 agosto. 2020.

Silveira, Paulo. **O que é SQL?** 2019. Disponível em: <https://www.alura.com.br/artigos/o-que-e-sql>. Acesso em: 20 outubro. 2020.

<<https://www.jetbrains.com/pt-br/pycharm/>>. Acesso em: 14 setembro. 2020.

Roveda, Ugo. O **que é Pyhton, para que serve e por que aprender?** 2020. Disponível em: <https://kenzie.com.br/blog/o-que-e-python/>. Acesso em: 25 setembro. 2021.

Casavella, Eduardo. **O que é Linguagem C?**. Disponível em: <http://linguagemc.com.br/o-que-e-linguagem-c/>. Acesso em: 17 julho. 2020.

**Introdução ao JSON**. Disponível em: <<https://www.json.org/json-pt.html>>. Acesso em: 04 janeiro. 2020

**O que é JavaScript**. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript.html>. Acesso em: 04 janeiro. 2020.

**CSS**. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/CSS>. Acesso em: 26 setembro. 2021.

**O que é API?**. Disponível em: <https://canaltech.com.br/software/o-que-e-api/> Acesso em: 13 março. 2021.

**Uma visão geral do HTTP**. Disponível em: **<**<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview>> Acesso em: 10 janeiro. 2020.