

**INSTITUTO FEDERAL DE SÃO PAULO  
CÂMPUS BOITUVA**

**ANGELO RODRIGO RIBEIRO DA SILVA**

**SOLUS, UM SOFTWARE PARA MONITORAMENTO  
DE PAINÉIS SOLARES**

**BOITUVA**

**2018**

ANGELO RODRIGO RIBEIRO DA SILVA

## **Solus, um software para monitoramento de painéis solares**

Documentação para o software desenvolvido como trabalho de conclusão de curso para análise e desenvolvimento de sistemas no Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, câmpus de Boituva.

INSTITUTO FEDERAL DE SÃO PAULO  
CÂMPUS BOITUVA

Orientador: Prof. Dr. Marcelo Figueiredo Polido.

BOITUVA  
2018

Silva, Angelo Rodrigo Ribeiro.

Solus: um software para monitoramento de painéis solares / Angelo Rodrigo Ribeiro da Silva. – Boituva, 2018.  
52 f. : il.

Orientador: Marcelo Figueiredo Polido

Monografia (Curso de Tecnólogo em Análise e Desenvolvimento de Sistemas) – Instituto Federal de São Paulo, Campus Boituva.

1. Nodejs. 2. Painéis solares. 3. Energia solar. 4. Arduino

I. Título.

ANGELO RODRIGO RIBEIRO DA SILVA

## **Solus, um software para monitoramento de painéis solares**

Documentação para o software desenvolvido como trabalho de conclusão de curso para análise e desenvolvimento de sistemas no Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, câmpus de Boituva.

BOITUVA, 4 de dezembro de 2018

Banca examinadora:

---

(Titulação, nome completo, instituição)

---

(Titulação, nome completo, instituição)

---

(Titulação, nome completo, instituição)

## **AGRADECIMENTOS**

Agradeço a meus amigos, que não me deixaram abalar nos momentos onde os prazos não pareciam ter fim, agradecimentos especiais aos meus amigos Cleiton Santos, Letícia Callistro, Wellington Sato, Dante Mesquita, Virginia Saucedo e Jaíne Santos, obrigado por entenderem minha ausência no processo de desenvolvimento do trabalho.

À minha família, por sempre fornecer a base que eu precisei para meus estudos.

E ao meu orientador Prof. Dr. Marcelo Figueiredo Polido, que auxiliou na construção dos protótipos e teve a disponibilidade e paciência para com o desenvolvimento deste trabalho.



## RESUMO

A aplicação desenvolvida baseia-se na dificuldade durante a análise de dados meteorológicos perante a prévia instalação e configuração de painéis solares. Este documento visa documentar um software com foco na captura e análise de dados meteorológicos, produzindo assim, resultados e informações necessárias para o usuário.

**Palavras-chave:** nodejs. painéis solares. energia solar. arduino.

## **ABSTRACT**

The application developed is based on the difficulty during the analysis of meteorological data before the previous installation and configuration of solar panels. This document aims to document a software focused on the capture and analysis of meteorological data, thus producing results and information necessary to the user.

**Keywords:** nodejs. solar panels. solar energy. arduino.



## LISTA DE ILUSTRAÇÕES

Figura 1 – Diagrama de caso de uso . . . . .	21
Figura 2 – Representação da arquitetura . . . . .	26
Figura 3 – Diagrama de classes . . . . .	27
Figura 4 – Diagrama de sequência . . . . .	30
Figura 5 – Diagrama de estado . . . . .	31
Figura 6 – Esboço da estação meteorológica . . . . .	34
Figura 7 – Protótipo da estação meteorológica . . . . .	35
Figura 8 – Tela de login . . . . .	46
Figura 9 – Dashboard . . . . .	46
Figura 10 – Resultados de estatísticas . . . . .	47
Figura 11 – Gráficos das estatísticas . . . . .	47
Figura 12 – Números dos gráficos das estatísticas . . . . .	48
Figura 13 – Listagem de estações meteorológicas . . . . .	48
Figura 14 – Modal de adição e edição dos dados de uma estação . . . . .	49
Figura 15 – Listagem de usuários . . . . .	49
Figura 16 – Modal de adição ou edição de usuário . . . . .	50
Figura 17 – Visualização de meu perfil . . . . .	50
Figura 18 – Edição de perfil . . . . .	51

## LISTA DE TABELAS

Tabela 1 – Consultar estatísticas . . . . .	23
Tabela 2 – Especificações do caso de uso consultar medidas . . . . .	23
Tabela 3 – Especificação do caso de uso manter usuários . . . . .	23
Tabela 4 – Especificação do caso de uso manter arduino . . . . .	24
Tabela 5 – Especificação do caso de uso editar perfil . . . . .	24
Tabela 6 – Especificação da Model base . . . . .	28
Tabela 7 – Especificação da Medida . . . . .	28
Tabela 8 – Especificação da Estação Meteorológica . . . . .	28
Tabela 9 – Especificação do Usuário . . . . .	29
Tabela 10 – Especificação do sensor de umidade e temperatura em relação a umidade . . . . .	36
Tabela 11 – Especificações do sensor de temperatura ambiente . . . . .	36
Tabela 12 – Especificações do sensor de luminosidade LDR . . . . .	36
Tabela 13 – Especificações do sensor de nível de chuva . . . . .	37
Tabela 14 – Especificações do sensor de capacidade solar . . . . .	37
Tabela 15 – Especificações do sensor de radiação uv . . . . .	37
Tabela 16 – Descrição das rotas da API . . . . .	42
Tabela 17 – Especificações das rotas de listagem e retorno de arduino . . . . .	42
Tabela 18 – Descrição dos campos adicionais . . . . .	42
Tabela 19 – Especificações das rotas de atualização e cadastro de arduino . . . . .	43
Tabela 20 – Especificações da rota exclusão de arduino . . . . .	43
Tabela 21 – Especificações das rotas de listagem e retorno de usuário . . . . .	43
Tabela 22 – Especificações das rotas de atualização e cadastro de usuário . . . . .	43
Tabela 23 – Especificações da rota de login de usuário . . . . .	44
Tabela 24 – Especificações da rota exclusão de usuário . . . . .	44
Tabela 25 – Especificações da rota de cadastro de medida . . . . .	44
Tabela 26 – Especificações da rota de consulta de estatísticas do arduino . . . . .	45

## **LISTA DE ABREVIATURAS E SIGLAS**

API	Application Programming Interface
HTTP	HyperText Transfer Protocol
IFSP Paulo	Instituto Federal de Educação, Ciência e Tecnologia de São
RESTFUL	Full Representational State Transfer

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>14</b>
<b>1.1</b>	<b>Objetivo</b>	<b>14</b>
<b>1.2</b>	<b>Justificativa</b>	<b>15</b>
<b>1.3</b>	<b>Estrutura do TCC</b>	<b>15</b>
<b>2</b>	<b>METODOLOGIAS</b>	<b>16</b>
<b>2.1</b>	<b>Programação funcional</b>	<b>16</b>
<b>2.2</b>	<b>Controle de qualidade</b>	<b>16</b>
2.2.1	Testes de interface	16
2.2.2	Testes unitários	16
<b>2.3</b>	<b>Diagramas UML 2.0</b>	<b>17</b>
<b>3</b>	<b>FERRAMENTAS UTILIZADAS</b>	<b>18</b>
<b>3.1</b>	<b>Astah</b>	<b>18</b>
<b>3.2</b>	<b>Javascript ES8</b>	<b>18</b>
<b>3.3</b>	<b>MongoDB</b>	<b>19</b>
<b>4</b>	<b>ENGENHARIA DE REQUISITOS</b>	<b>20</b>
<b>4.1</b>	<b>Cenários do sistema</b>	<b>20</b>
<b>4.2</b>	<b>Requisitos funcionais</b>	<b>21</b>
4.2.1	Detalhamento dos casos de uso	22
<b>4.3</b>	<b>Requisitos não funcionais</b>	<b>24</b>
4.3.1	Performance	24
4.3.2	Segurança	25
<b>4.4</b>	<b>Arquitetura macro</b>	<b>25</b>
<b>5</b>	<b>DIAGRAMAS DO SISTEMA</b>	<b>27</b>
<b>5.1</b>	<b>Diagrama de classes</b>	<b>27</b>
<b>5.2</b>	<b>Diagrama de sequência</b>	<b>29</b>
5.2.1	Controlador	30
5.2.2	Repositório	30
5.2.3	Modelo	31
<b>5.3</b>	<b>Diagrama de estado</b>	<b>31</b>
<b>6</b>	<b>ARQUITETURA DAS ESTAÇÕES METEOROLÓGICAS</b>	<b>33</b>
<b>6.1</b>	<b>Arduino</b>	<b>33</b>
<b>6.2</b>	<b>Sensores</b>	<b>35</b>

<b>6.3</b>	<b>NodeMCU</b> . . . . .	<b>38</b>
<b>7</b>	<b>ARQUITETURA DO SOFTWARE</b> . . . . .	<b>39</b>
<b>7.1</b>	<b>Servidor</b> . . . . .	<b>39</b>
7.1.1	Segurança . . . . .	39
7.1.2	Banco de dados . . . . .	40
7.1.3	Tratamento dos dados meteorológicos . . . . .	41
7.1.4	Análise dos dados . . . . .	41
7.1.5	Documentação da API . . . . .	41
<b>7.2</b>	<b>Cliente</b> . . . . .	<b>45</b>
7.2.1	Descrição das telas . . . . .	45
<b>8</b>	<b>CONCLUSÃO</b> . . . . .	<b>52</b>
	<b>REFERÊNCIAS</b> . . . . .	<b>53</b>

## 1 INTRODUÇÃO

A captação de energia elétrica no Brasil, é composta por diversas fontes primárias, dentre as quais, se destacam o uso de energia eólica, energia hidráulica e energia solar, analisadas as diversas fontes de energia, dentre as tais, a energia solar se destaca por sua baixa implicação a questões ambientais.

A energia solar, vem sendo cada vez mais amplamente utilizada no Brasil, tanto por empresas de pequeno a grande porte, quanto pelo mercado residencial, segundo a projeção da Associação Brasileira de Energia Solar Fotovoltaica (Absolar), tal forma de captação de energia representa 0,83% de toda a energia captada no país (KOLOSZUK; SAUAIA, 2018).

Mesmo com o amadurecimento da captação de energia solar no país, para a instalação de uma estação de captação de energia solar, é recomendado que seja feita uma prévia análise de dados no local, para que, resultados satisfatórios possam ser obtidos a mínimo prazo.

O trabalho desenvolvido visa fornecer uma ferramenta para auxiliar o monitoramento de painéis solares através da captura e análise de dados meteorológicos, visando a geração de informações necessárias para o usuário.

### 1.1 Objetivo

A eficiência energética de um painel solar está ligada diretamente a condição meteorológica atual, pois, uma vez que as condições não são satisfatórias, a eficiência tende a cair. Para que se possa obter melhor desempenho na captação de energia solar, as condições meteorológicas precisam ser previamente estudadas no local de instalação, pois erros de instalação e configuração podem trazer prejuízos financeiros.

O trabalho tem como objetivo fornecer uma ferramenta para captura, persistência e análise de dados meteorológicos, fornecendo a usuários, informações necessárias para o acompanhamento da eficiência energética de painéis solares.

Aproveitando-se do baixo custo de equipamentos eletrônicos para disponibilizar um software como um todo de baixo custo e de fácil instalação.

Para se atingir o objetivo especificado, foi construído um software de amostragem de dados ao usuário, realizando previamente um tratamento e análise de dados para que possam ser realizados estudos ante a instalação de painéis fotovoltaicos.

## 1.2 Justificativa

A justificativa do desenvolvimento do trabalho se dá pela iniciativa da instalação de uma usina solar no Instituto Federal de São Paulo, no Campus localizado em Boituva.

Foi implementado um laboratório de  $30m^2$ , composto por uma estação solarimétrica e um sistema de  $5kW$  com rastreador solar, entre outras tecnologias.

O projeto de exploração de energia fotovoltaica proporcionou a criação do curso de instalador de sistemas fotovoltaicos, onde a ferramenta visa ser utilizada de forma a agrupar os dados capturados em diferentes pontos (IFSP, 2018).

## 1.3 Estrutura do TCC

O trabalho apresentado, primeiramente procura contextualizar acerca da atual situação da captação de energia solar, procurando explicar conceitos básicos de energia fotovoltaica e decorre sobre as dificuldades e cuidados entorno do assunto.

Subsequindo, são apresentadas as metodologias utilizadas no desenvolvimento do projeto, decorrendo acerca das ferramentas utilizadas e conceitos nos quais o projeto se baseia, no desenvolvimento do trabalho, em seguida, é apresentado ao usuário o projeto do sistema, onde questões teóricas acerca da implementação são discutidas, seguindo o desenvolvimento do projeto, se apresenta a arquitetura e documentação do software, detalhando questões internas do desenvolvimento do software e as justificativas pelas tecnologias utilizadas.

Por fim, são apresentadas as considerações finais do projeto, demonstrando os resultados obtidos, comparando os mesmos com os requisitos, procurando esclarecer se os objetivos foram atingidos ou não, e então, é feita uma projeção do projeto para o futuro, apresentando um caminho para o projeto de manutenção e continuidade do software.

## 2 METODOLOGIAS

### 2.1 Programação funcional

Foi utilizado para a elaboração da arquitetura do software, conceitos de programação funcional, sua maior característica é que, se uma expressão possui um valor bem definido, dada uma entrada de um dado, a ordem a qual o computador carregar a avaliação não afeta o resultado (BIRD; WADLER, 1988), ou seja, não importa qual a entrada, sempre teremos a mesma saída, assim, o software se torna mais previsível, aumentando também a testabilidade.

O conceito de uma função que sempre retorna o mesmo resultado dada uma entrada é chamado de conceito de função pura, outra característica de uma função pura, é que ela não deve interferir em contexto externo, dessa forma, pode-se garantir o controle de efeito colateral da aplicação, porém, ainda é necessário realizar operações de escrita em banco de dados e acesso a variáveis globais, isso é feito em funções impuras, que, uma vez identificadas, são separadas do resto da aplicação.

### 2.2 Controle de qualidade

Foram utilizadas, juntamente ao desenvolvimento do software, técnicas de controle de software orientadas ao teste, onde, a aplicação é submetida a seção de testes, tanto manuais como automatizados, foram utilizados testes de interface no sistema e testes unitários automatizados.

#### 2.2.1 Testes de interface

Para que fosse validada a funcionalidade do sistema como um todo, testes de interface no sistema foram aplicados, onde, foi escolhido um usuário leigo para aplicar testes no sistema, informando possíveis dificuldades na utilização e erros que são corrigidos até que se possa ser atingida uma mínima aceitação.

#### 2.2.2 Testes unitários

Segundo Beck, (2002, p.10), "Desenvolvimento orientado a testes, é uma maneira de gerenciar o medo durante a programação".

O conceito de testes unitários em desenvolvimento de software, é fornecer a uma unidade, que pode ser considerada como uma função ou uma instrução do sistema, valores esperados, valores esses, que são comparados com os retornos das funções, e que, uma vez não correspondendo, emitem um erro ao desenvolvedor, que



pode, com a visualização facilitada do problema, corrigir tal ponto de forma assertiva (BECK, 2002).

Tal conceito fora aplicado, para garantir, de forma unitária a qualidade e segurança do código fonte do sistema, garantindo assim, a qualidade da aplicação, desde os primeiros momentos de sua implementação.

### 2.3 Diagramas UML 2.0

Assim como definido por Guedes, (2018), a UML:

É uma linguagem 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. (p.17).

No trabalho apresentado, mesmo que não utilizados conceitos de orientação a objetos, a linguagem de modelagem UML foi utilizada como ferramenta para detalhar partes do software, descrevendo desde os casos de uso da aplicação, as funcionalidades do sistema.

## 3 FERRAMENTAS UTILIZADAS

### 3.1 Astah

Astah Community é um software para modelagem UML com suporte a UML 2, desenvolvido pela Change Vision, e disponível para diversos sistemas operacionais. Anteriormente conhecido por JUDE, um acrônimo de Java and UML Developers Environment (Ambiente para Desenvolvedores UML e Java) ([LIMA, 2016](#)).

O mesmo foi utilizado no projeto para a criação dos diagramas de descrição software, auxiliando no desenho dos casos de uso na engenharia de requisitos e nos diagramas de classe, diagramas de sequência, estado e implementação.

A utilização do software Astah, foi motivada pela sua confiabilidade na construção dos diagramas, fornecendo uma plataforma com redundâncias e por fornecer uma licença gratuita de sua versão profissional a universidades e estudantes de engenharia de software.

### 3.2 Javascript ES8

O Javascript é um linguagem de programação interpretada multiparadigma, aceitando estilos variados de programação como orientação a objetos, programação funcional e imperativa ([MOZILLA, 2018](#)). O Javascript é uma linguagem de programação primariamente desenvolvida para o lado do cliente, sendo executada no navegador do usuário, porém, com o advento do novo motor de processamento de javascript do Google Chrome, o v8, foi criado o nodejs, que é capaz de interpretar a linguagem javascript também no lado servidor ([NODEJS, 2018](#)).

Para a implementação do projeto, foi utilizada a linguagem de programação Javascript em sua versão 8, ou como definida pela especificação do Javascript, o EcmaScript 2017.

A utilização da linguagem Javascript, tanto para a construção da aplicação no lado servidor, quando para o lado cliente, foi determinada por sua alta performance e por proporcionar ferramentas que permitem a programação de forma funcional, detalhes acerca da utilização da linguagem são melhor definidos na seção acerca dos detalhes da implementação do software, na seção acerca da arquitetura de software, na seção [7](#).

### 3.3 MongoDB

O MongoDB é um banco de dados de documento com suporte a escalabilidade e flexibilidade para execução de consultas e indexação necessária ([MONGODB, 2018](#)).

O mesmo foi utilizado no projeto em sua versão 4.1, a justificativa de seu uso, se da pela sua facilidade ao se trabalhar com escrita massiva de informações, proporcionando a aplicação performance e disponibilidade.

## 4 ENGENHARIA DE REQUISITOS

Assim como descrito por [Pressman \(2011\)](#):

O amplo espectro de tarefas e técnicas que levam a um entendimento dos requisitos é denominado engenharia de requisitos. Na perspectiva do processo de software, a engenharia de requisitos é uma ação de engenharia de software importante que se inicia durante a atividade de comunicação e continua na de modelagem. Ela deve ser adaptada às necessidades do processo, do projeto, do produto e das pessoas que estão realizando o trabalho. (p.126).

Neste capítulo, há a utilização da engenharia de requisitos para auxiliar a documentação do projeto de software, descrevendo os cenários de utilização de sistema, onde é descrito o comportamento esperado pelo usuário através de entrevistas, após, utilizando os resultados obtidos com a modelagem através de cenários, os requisitos são definidos e são criados os contratos de uso do usuário com o sistema através de casos de uso, então, é descrita a arquitetura macro do software, procurando contextualizar acerca do desenho do projeto.

### 4.1 Cenários do sistema

Há diversas formas de se medir a qualidade de um software, porém, nenhuma delas representa melhor a qualidade do software do que a satisfação dos usuários. Se entendido como os usuários desejam interagir com o sistema, o que é esperado de entradas, processamentos e saídas, será possível a equipe construir um projeto mais objetivo e proveitoso ([PRESSMAN, 2011](#)).

Portanto, nesta seção são descritos os cenários do software através de uma descrição que foi construída utilizando conteúdos das entrevistas que foram feitas com as partes interessadas no projeto.

No Instituto Federal de São Paulo, no campus localizado em Boituva, o professor responsável pelo projeto descreve o sistema como uma aplicação capaz de capturar dados meteorológicos através de sensores próximos a painéis solares, ele necessita dessas informações sendo demonstradas através de gráficos, para que uma análise possa ser feita, além da disposição das informações através de gráficos, é esperado do sistema demonstrar ao usuário informações de mínimas e máximas das informações capturadas.

É desejado pelo usuário que o sistema seja capaz de lidar com diversas estações meteorológicas, reunindo os dados, portanto, o sistema necessita de um cadastro das estações meteorológicas, guardando informações e identificações.

Através do texto acima, se podem entender as necessidades do usuário para com o sistema, o cenário de software descrito é definido dentro dos requisitos nas seções seguintes.

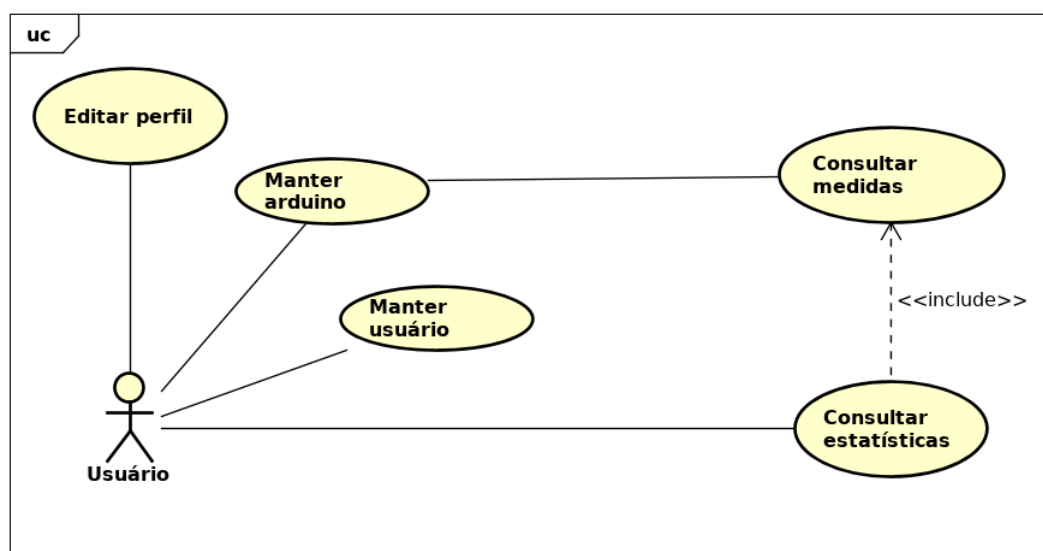
## 4.2 Requisitos funcionais

Os requisitos funcionais de um software são utilizados para descrever o que ele deve fazer, eles giram em torno do tipo de sistema desenvolvido, de seus usuários e da forma como uma equipe de desenvolvimento de software busca entender o que é necessário para criar um software que atenda as necessidades de um usuário. Ao serem descritas as funcionalidades básicas de um sistema, ela é feita de forma abstrata, para que os interessados no desenvolvimento do projeto possam entender o que será feito. Porém, os mesmos podem ser detalhados de forma específica, declarando suas entradas, saídas e exceções (SOMMERVILLE, 2011).

Foi utilizado para descrever os requisitos funcionais do sistema, a modelagem de requisitos através do diagrama de caso de uso, que busca visualizar, especificar e documentar o comportamento de um ator ao utilizar o sistema. (GUEDES, 2018).

O diagrama de caso de uso descrito na figura 1, resume os requisitos funcionais que foram definidos ao longo do desenvolvimento do projeto.

Figura 1 – Diagrama de caso de uso



Fonte: Produção do autor.

Através do diagrama do caso de uso contido na figura 1, é possível identificar apenas um ator na aplicação, que é o usuário do sistema, interessado em visualizar

informações geradas pela análise estatística dos dados capturados. O mesmo interage com a aplicação através da interface de usuário, realizando cadastros e consultas.

O requisito funcional **Consultar estatísticas** é o principal requisito do sistema, onde o usuário fará a filtragem dos dados, trazendo a ele as estatísticas geradas pela aplicação. O usuário irá utilizar de campos de entrada de dados para escolher qual a estação meteorológica ele deseja visualizar, a data inicial para realizar as consultas, a data final e o intervalo de tempo para cálculo das médias. Após a consulta das informações, o usuário irá obter como resultado as informações de mínimas e máximas das medidas e gráficos que o informam as médias para cada uma das propriedades meteorológicas.

O requisito funcional representado pelo caso de uso **Manter usuário**, descreve as ações do usuário acerca do cadastro de usuários no sistema. Através desta funcionalidade, o usuário poderá, através da interface do usuário na web, utilizando tabelas, realizar o cadastro, consultas, atualização de informações e exclusão de usuários. Durante a interação do usuário com a persistência de dados de usuários no sistema, caso erros ocorram, mensagens de erro auxiliares serão mostradas ao usuário, o auxiliando a encontrar qual o erro no processo realizado.

O requisito funcional **Manter arduino**, é muito similar ao caso de uso anterior **Manter usuário**, e representa a persistência de informações acerca das estações meteorológicas no sistema, utilizando as quatro operações básicas de cadastro, consulta, atualização e exclusão de dados.

O requisito funcional de representado pelo caso de uso **Editar Perfil** representa a visualização e edição dos dados do usuário que está utilizando o sistema no momento, ele pode escolher editar suas informações, que exibe um formulário com os atuais dados preenchidos, onde o usuário pode realizar a alteração de suas informações.

#### 4.2.1 Detalhamento dos casos de uso

Utilizando de tabelas, este capítulo descreve de forma detalhada cada um dos casos de uso, apresentado as descrições, os atores relacionados, as pré condições, exceções e fluxos alternativos, conforme pode ser visualizado nas tabelas de 1 a 5.

Tabela 1 – Consultar estatísticas

<b>Realizar consultas com filtros</b>	
Descrição	Recebe do usuário os filtros para seleção das informações, então, realiza uma análise estatística dos dados requisitados e os exibe para o usuário utilizando listagens e gráficos.
Atores	Usuário
Pré-condições	Banco de dados em correto funcionamento Filtros corretos passados pelo usuário Listagem de estações meteorológicas retornando ao mínimo uma estação
Exceções e fluxos alternativos	Caso a seleção não retorne nenhuma informação ao usuário, uma mensagem de dados inexistentes é exibida Em caso de perda de conexão com banco de dados, exibe uma mensagem de erro ao usuário Caso não existam estações meteorológicas cadastradas, o filtro de seleção de estação meteorológica exibe um aviso ao usuário

Tabela 2 – Especificações do caso de uso consultar medidas

<b>Consultar medidas</b>	
Descrição	Consulta as medidas que estão persistidas no banco de dados para uma estação meteorológica, no software desenvolvido, atualmente este caso de uso é apenas acessado pela geração de estatísticas
Pré-condições	Credenciais de acesso ao banco de dados Banco de dados em correto funcionamento
Exceções e fluxos alternativos	Caso não existam medidas capturadas para a estação meteorológica, retorna uma lista vazia de dados

Tabela 3 – Especificação do caso de uso manter usuários

<b>Manter usuários</b>	
Descrição	Realiza uma listagem dos dados dos usuários atualmente registrados no sistema, fornecendo a possibilidade de cadastrar um novo usuário, atualizar os dados de um usuário ou excluir um usuário do sistema
Pré-condições	Credenciais de acesso ao banco de dados Banco de dados em correto funcionamento
Exceções e fluxos alternativos	Caso não existam usuários cadastrados no sistema, exibe uma mensagem de aviso ao usuário Para o cadastro ou atualização dos dados de um usuário, caso existam dados inválidos que foram dados como entrada, ao confirmar a operação, exibe uma mensagem de erro ao usuário

Tabela 4 – Especificação do caso de uso manter arduino

<b>Manter arduino</b>	
Descrição	Realiza uma listagem dos dados das estações meteorológicas atualmente registradas no sistema, fornecendo a possibilidade de cadastrar uma nova estação, atualizar os dados de uma estação ou excluir uma estação meteorológica do sistema
Pré-condições	Credenciais de acesso ao banco de dados Banco de dados em correto funcionamento
Exceções e fluxos alternativos	Caso não existam estações cadastradas no sistema, exibe uma mensagem de aviso ao usuário Para o cadastro ou atualização dos dados de uma estação, caso existam dados inválidos que foram dados como entrada, ao confirmar a operação, exibe uma mensagem de erro ao usuário

Tabela 5 – Especificação do caso de uso editar perfil

<b>Editar perfil</b>	
Descrição	Exibe os dados do usuário que está utilizando o sistema no momento, dando a possibilidade do usuário alterar os próprios dados
Pré-condições	Credenciais de acesso ao banco de dados Banco de dados em correto funcionamento
Exceções e fluxos alternativos	Caso na edição do perfil do usuário atual existam dados inválidos que foram dados como entrada, ao confirmar a operação, exibe uma mensagem de erro ao usuário

### 4.3 Requisitos não funcionais

Os requisitos não funcionais, como seu próprio nome demonstra, não estão ligados diretamente ao que um software deve fazer, porém, tais requisitos são considerados tão importantes quanto os requisitos funcionais, pois podem comprometer a usabilidade e funcionalidades de um sistema (SOMMERVILLE, 2011).

Comumente, requisitos não funcionais estão ligados a questões de disponibilidade de software, tempo de resposta ao usuário e outras partes que se comunicam com o sistema e segurança.

Uma das formas de determinar a arquitetura do sistema seguindo os requisitos não funcionais é restringir a forma como as funcionalidades primárias do sistema trabalham, trabalhando no projeto e na implementação do software de forma restrita dentro destes padrões.

Nas seções subsequentes os tópicos de disponibilidade, tempo de resposta e segurança do sistema são melhor detalhados.

#### 4.3.1 Performance

Para a boa usabilidade do usuário ao interagir com um sistema, um dos requisitos não funcionais mais importantes é o tempo de resposta de uma aplicação, pois,



dependendo da lentidão com a qual as informações são disponibilizadas, um usuário pode ter problemas e, em muitos casos, não utilizar o sistema por conta do tempo de resposta.

Na era da informação, a informação é trabalhada de forma quase instantânea, portanto, tal requisito não pode ser deixado em segundo plano ao se trabalhar no projeto de um software.

No sistema desenvolvido, todo o desenvolvimento do sistema foi pensando de forma a fornecer ao usuário um baixo tempo de resposta através do uso de técnicas de desenvolvimento de software focadas em alta disponibilidade, desde a implementação da arquitetura do software e de como o banco de dados é utilizado, até a forma como os dados são carregados na interface do usuário. Para atingir tal objetivo, optou-se por disponibilizar ao usuário o mínimo de informações necessárias para a interação com o sistema.

Detalhes acerca da forma como o sistema é implementado para fornecer ao usuário a melhor experiência possível desde sua implementação em baixo nível são melhor descritos na seção sobre a arquitetura do software, no capítulo 7.

#### 4.3.2 Segurança

Outro requisito muito importante para qualquer sistema é a segurança, tanto nas informações que ali estão contidas quanto no resultado final que é apresentado ao usuário, com a evolução da tecnologia, as pessoas passaram a estar muito mais próximas a sistemas de informação, colocando ali, informações sensíveis.

Neste projeto, para que se pudesse proporcionar ao usuário melhor segurança em seus dados e nas informações meteorológicas que são informadas, foram utilizadas técnicas de criptografia de informações, focando primariamente em informações que são sensíveis, e, para acesso ao sistema, técnicas de bloqueio de informações através de chaves foram utilizadas, detalhes acerca de como a segurança foi implementada são relatados na seção 7, onde a arquitetura do software proposto é apresentada.

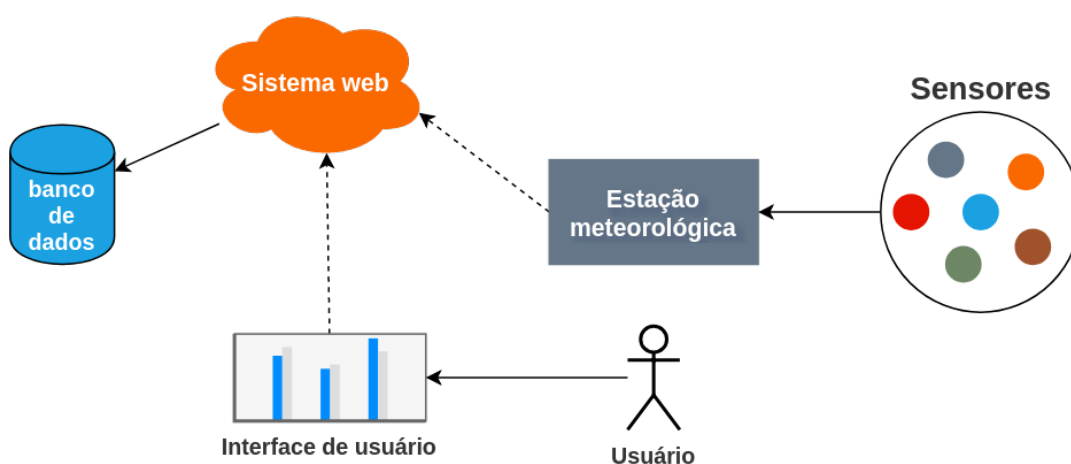
### 4.4 Arquitetura macro

A arquitetura de um software segundo [Sommerville, \(2011\)](#):

É o elo crítico entre o projeto e a engenharia de requisitos, pois identifica os principais componentes estruturais de um sistema e os relacionamentos entre eles. O resultado do processo de projeto de arquitetura é um modelo de arquitetura que descreve como o sistema está organizado em um conjunto de componentes de comunicação. (p.104).

Procurando descrever os casos de uso do sistema, optou-se por utilizar uma descrição da arquitetura macro do software através de figuras, descrevendo o sistema através de simples componentes, pode-se conferir a arquitetura através da figura 2

Figura 2 – Representação da arquitetura



Fonte: Produção do autor.

Através do desenho da arquitetura do software pode-se identificar as relações entre os diversos componentes do sistema, porém, de uma forma a qual as partes interessadas no projeto ainda podem interagir e entender o que está sendo proposto.

No diagrama, pode-se identificar os sensores conectados diretamente com a estação meteorológica (traço preenchido), enviando as informações capturadas para a estação meteorológica através de conexão web (traço pontilhado), que, após receber as informações as envia para o sistema Web, que as persiste em banco de dados.

O usuário, interage com o sistema diretamente através da interface que disponibiliza gráficos para que o acompanhamento possa ser realizando pelo usuário, a interface do usuário por sua vez, se comunica através de requisições web com o servidor.

Dessa forma, pode-se descrever a arquitetura do projeto, visando, atingir o objetivo definido através da engenharia de requisitos com um projeto agora melhor detalhado, tanto para as partes interessadas, quanto para a equipe de software, nas seções seguintes, é descrita de forma técnicas os componentes do sistema, através de diagramas UML.

## 5 DIAGRAMAS DO SISTEMA

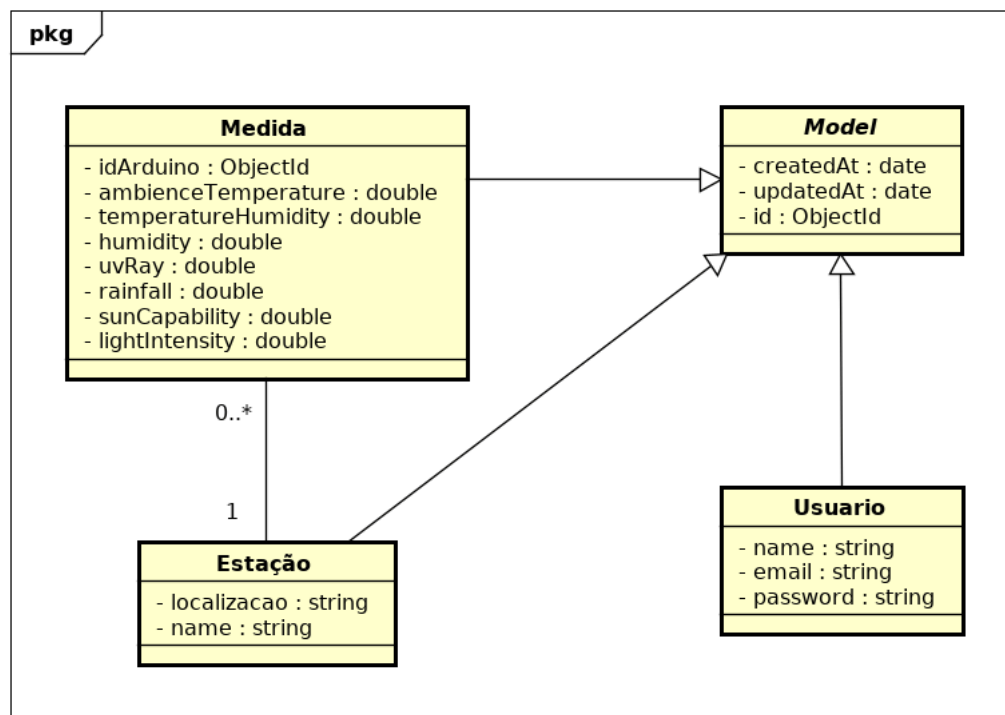
Foram utilizados para a descrição dos componentes da aplicação os diagramas UML, os diagramas de classe, de sequência e de estado descrevem o software nos capítulos seguintes.

### 5.1 Diagrama de classes

"Os diagramas de classe são usados no desenvolvimento de um modelo de sistema orientado a objetos para mostrar as classes de um sistema e as associações entre essas classes. Em poucas palavras, uma classe de objeto pode ser pensada como uma definição geral de um tipo de objeto do sistema."(SOMMERVILLE, 2011).

Como foram utilizados os conceitos de programação funcional na aplicação, não foram utilizados no sistema objetos para o encapsulamento dos comportamentos, como descreve a programação orientada a objetos, que a uml visa documentar, porém, o diagrama de classes ainda foi útil, pois, pode descrever as entidades do sistema.

Figura 3 – Diagrama de classes



Fonte: Produção do autor.

Como se pode verificar no diagrama descrito na figura 3 as entidades do sistema consistem na medida, que é a informação que foi capturada pela estação meteorológica, que pertence a uma estação, a entidade de estação meteorológica, que possui de nenhuma à muitas medidas capturadas, e a entidade de usuário, que representa um usuário cadastrado no sistema.

As propriedades de cada entidade são especificadas nas figuras 6 a 9, descartando seus comportamentos, pois, as mesmas são manipuladas através das camadas, que são descritas nas seções que descrevem o comportamento do sistema, através de diagramas de sequência e estado, nas respectivas seções 5.2 e 5.3.

Tabela 6 – Especificação da Model base

Model	
Descrição	É a model base da aplicação, que advem do framework utilizado pra mapeamento dos objetos para o banco de dados
Propriedades	
id	Identifica de forma única um objeto no banco de dados
createdAt	Data em que o objeto foi armazenado
updatedAt	Data em que alguma das propriedades do objeto foi alterada pela última vez

Tabela 7 – Especificação da Medida

Medida	
Descrição	Representada a medida que foi capturada pela estação meteorológica
Propriedades	
idArduino	Identificação do arduino que realizou a captura
ambienteTemperatura	O valor que foi capturado pelo sensor de temperatura ambiente
temperatureHumidity	Valor da temperatura capturada pelo sensor agregado de temperatura e umidade
humidity	Valor da umidade capturada pelo sensor agregado de temperatura e umidade
uvRay	Nível de radiação uv que foi medida
rainfall	Porcentagem de chuva capturada
sunCapability	Capacidade solar que foi medida
lightIntensity	Nível da intensidade de luz capturada

Tabela 8 – Especificação da Estação Meteorológica

Estação meteorológica	
Descrição	Representa uma estação meteorológica cadastrada no sistema
Propriedades	
location	Descrição de onde a estação meteorológica está posicionada
name	Nome de identificação da estação meteorológica

Tabela 9 – Especificação do Usuário

Usuário	
Descrição	Representa um usuário cadastrado no sistema
Propriedades	
name	Nome de identificação do usuário
email	E-mail que identifica o usuário e é utilizado para login no sistema
password	Senha do usuário ao acessar o sistema, a informação é criptografada

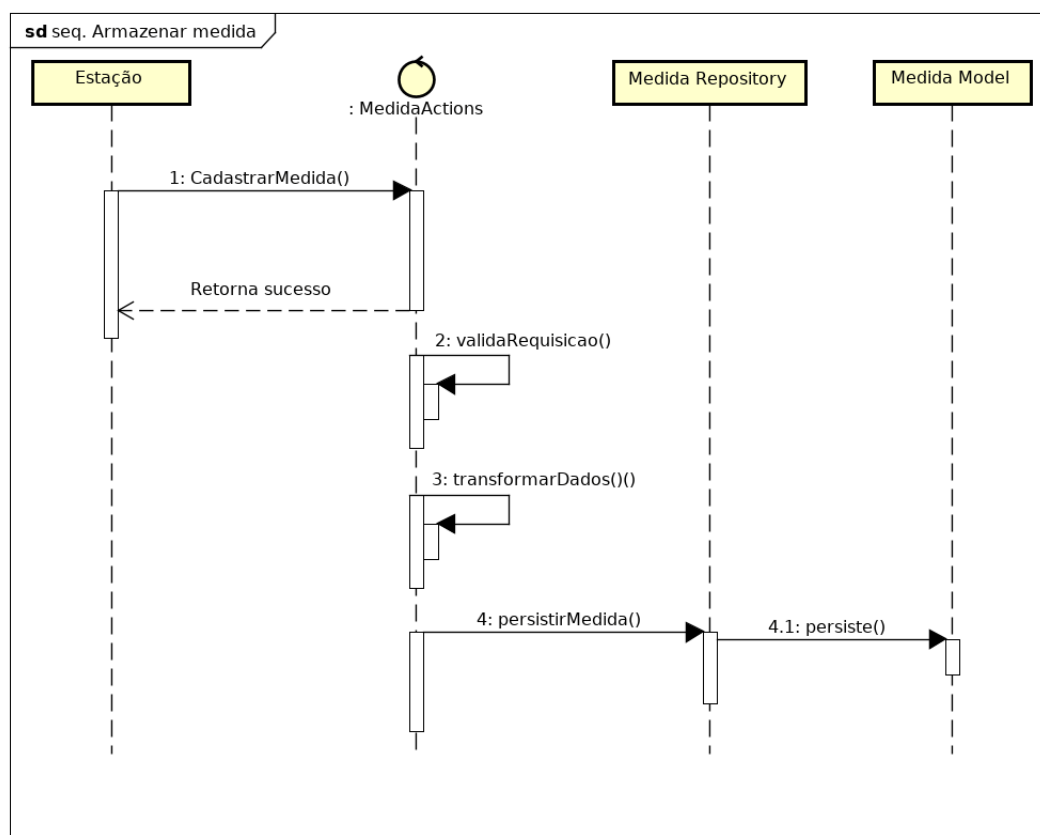
## 5.2 Diagrama de sequência

"Os diagramas de sequência em UML são usados, principalmente, para modelar as interações entre os atores e os objetos em um sistema e as interações entre os próprios objetos. A UML tem uma sintaxe rica para diagramas de sequência, que permite a modelagem de vários tipos de interação."(SOMMERVILLE, 2011).

Nesta seção, é apresentado o diagrama de sequência que representa o armazenamento de uma medida capturada por uma estação meteorológica, apresentando desde a saída dos dados, até a sua persistência em banco.

Para melhor aproveitamento da estação meteorológica e para que não haja necessidade de espera de uma resposta do software, utilizou-se da programação assíncrona para responder com sucesso a estação, os dados são enviados, e uma resposta de sucesso é emitido a estação, porém, os dados são processados em seguida, deixando livre a fila de envio na estação.

Figura 4 – Diagrama de sequência



Fonte: Produção do autor.

Como descrito no diagrama de sequência representado na figura 4 a aplicação é composta por 3 principais camadas, seguindo um modelo que são descritas nos capítulos seguintes.

#### 5.2.1 Controlador

Na camada de controle, após ser recebida uma mensagem da estação meteorológica, a mesma retorna sucesso para a estação, porém sem a confirmação das informações, que são logo em seguida, validadas, e então são tratadas utilizando as regras descritas na seção 7.1.3.

#### 5.2.2 Repositório

A camada de repositório trabalha como uma intermediária entre o controlador da aplicação onde a validação dos dados é feita e a camada de persistência da aplicação, ela trabalha verificando se os dados persistidos estão corretos e fornece um local comum para consulta e armazenamento das informações.

### 5.2.3 Modelo

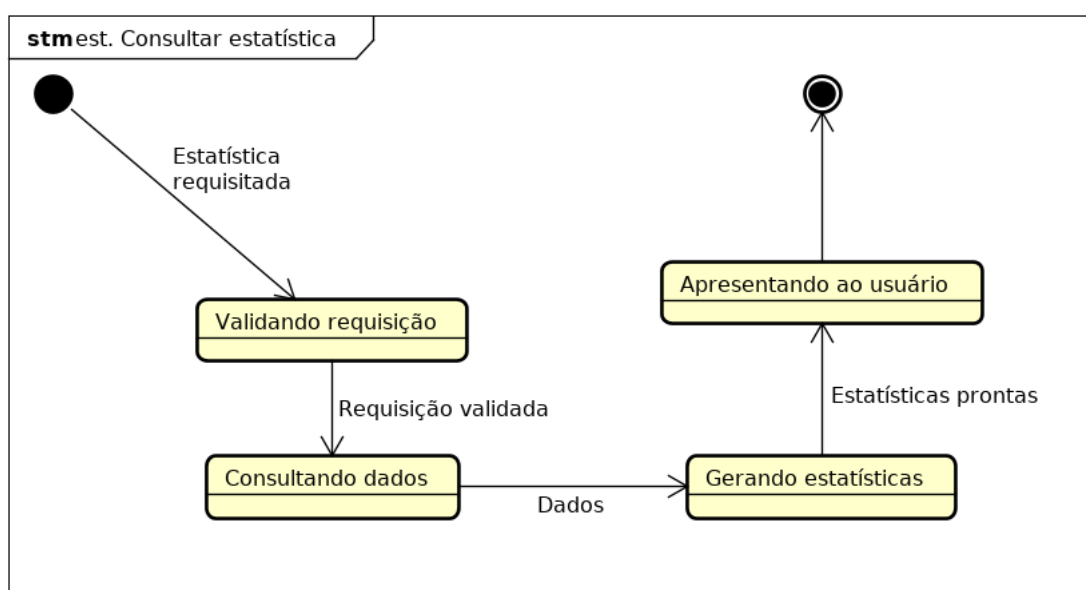
A camada de modelos, é composta pelas entidades do sistema, é a única camada da aplicação que possui acesso direto ao banco de dados, e é gerenciada através da ferramenta de comunicação com o banco de dados, abstraindo assim, o acesso ao baixo nível da persistência do software.

## 5.3 Diagrama de estado

"O diagrama de máquina de estados demonstra o comportamento de um elemento por meio de um conjunto finito de transições de estado, ou seja, uma máquina de estados."(GUEDES, 2018).

O diagrama de estado abaixo compreende a consulta de estatísticas na aplicação, demonstrando, o estado de uma requisição no sistema até sua resposta ao usuário.

Figura 5 – Diagrama de estado



Fonte: Produção do autor.

Na figura 5 pode-se identificar os estados como sendo a estatística requisitada ao sistema, a requisição sendo identificada como válida, então, os dados são encontrados no sistema e as estatísticas são apresentadas ao usuário.

No estado identificado como **Validando requisição**, a mesma é validada, identificando se o usuário realmente possui acesso ao sistema e se os filtros que foram

aplicados são também válidos, após os dados são consultados e passados para a geração de estatísticas, nos estados **Consultando dados** e **Gerando estatísticas**, respectivamente, por fim, no estado identificado como **Apresentando ao usuário** os gráficos são renderizados na tela, e o ciclo de vida da requisição termina.



## 6 ARQUITETURA DAS ESTAÇÕES METEOROLÓGICAS

Para a construção da arquitetura da estação meteorológica, foi utilizado como central o microcontrolador arduino, diversos sensores para captura das informações e a placa WiFi NodeMCU para conexão a internet e gerenciamento das requisições com o servidor.

### 6.1 Arduino

Como definido em seu site oficial: "O Arduino é uma plataforma eletrônica de código aberto baseada em hardware e software de fácil utilização. É destinado a qualquer pessoa que construa projeto interativos."([ARDUINO, 2018](#)).

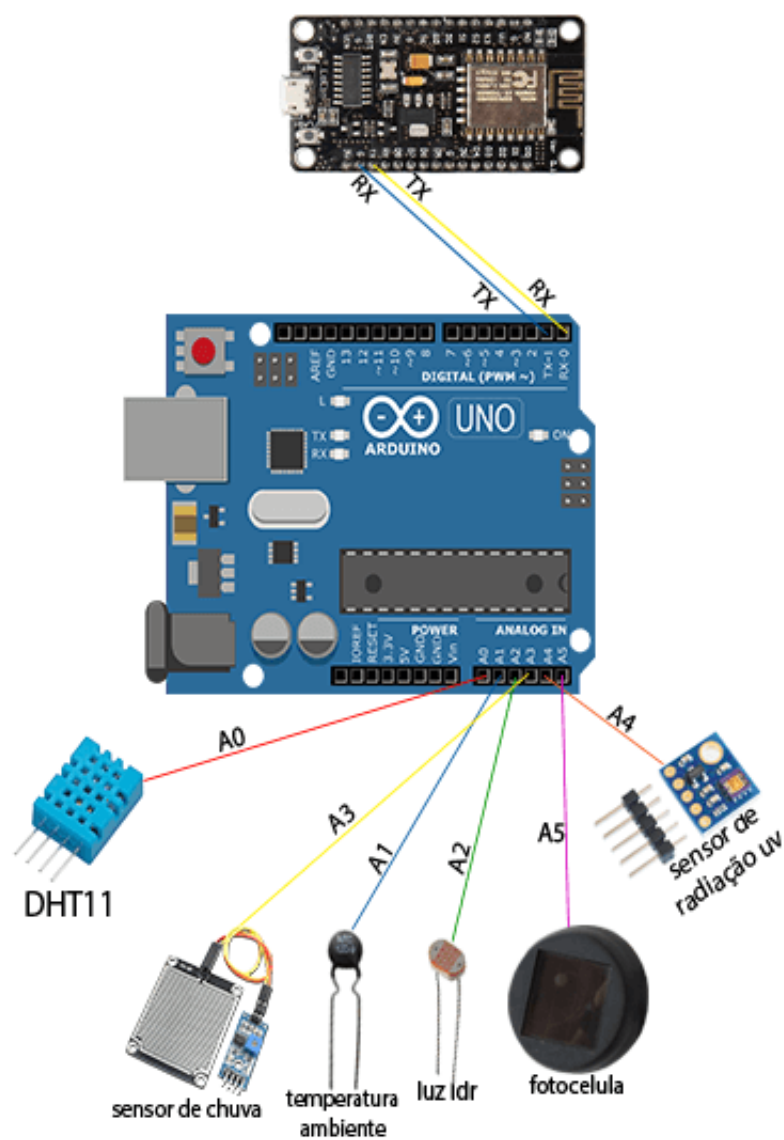
Ele fornece um controlador eletrônico único baseado no microprocessador AT-mega328, desenvolvido em C++, componentes eletrônicos oficiais para a construção de aplicações interativas e uma IDE (Ambiente de desenvolvimento integrado) para a construção de sistemas.

Para a construção da estação meteorológica descrita, foi utilizado arduino UNO, em sua versão contendo 6 portas analógicas, 6 portas digitais uma cpu com velocidade de 16 MHz e 32 KB de armazenamento interno.

Foram utilizados diversos sensores conectados as portas seriais do arduino para coleta de dados, após coletados os dados, o arduino monta a requisição que o servidor espera em formato JSON, o JSON é um formato de dados similar ao XML, porém, utiliza a sintaxe de objetos do javascript ([JSON, 2017](#)), para a construção do JSON no arduino não foi utilizada nenhuma biblioteca, por conta da quantidade de memória limitada do controlador.

Na figura 6, pode-se conferir o diagrama da estação meteorológica, demonstrando de forma resumida a ligação entre os sensores, o arduino e o nodeMCU.

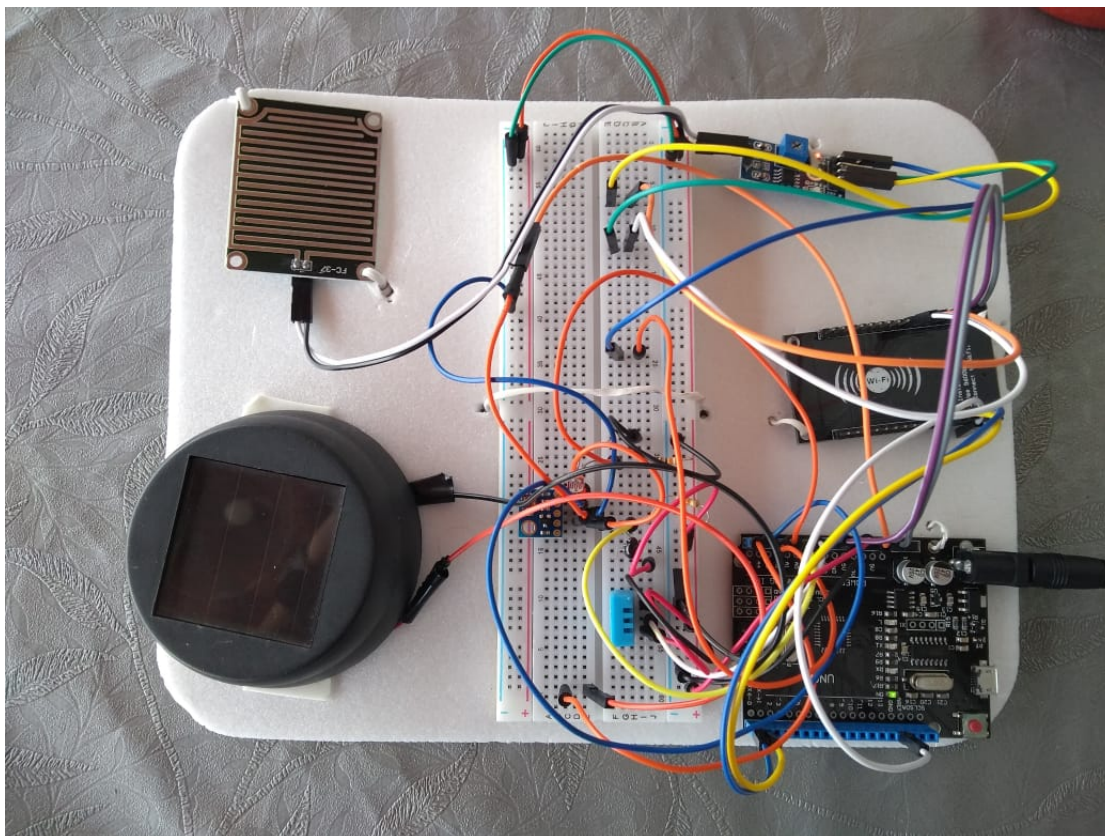
Figura 6 – Esboço da estação meteorológica



Fonte: Produção do autor.

Na figura 7 pode-se visualizar o protótipo construído.

Figura 7 – Protótipo da estação meteorológica



Fonte: Produção do autor.

## 6.2 Sensores

Para a captura das informações foram utilizados os sensores de temperatura, umidade do ar com temperatura em relação a umidade, sensor de chuva digital/analógico, sensor de intensidade luminosa, radiação UV e capacidade solar.

Os sensores utilizados se conectam com o arduino através de suas portas seriais, emitindo as informações nas portas seriais para que o arduino possa as capturar.

Abaixo, pode-se conferir um exemplo do JSON que é enviado ao servidor:

```
{
  "arduinoId": "5bd5182d0a39f500e611d5a5",
  "humidity": 46.41,
  "temperatureHumidity": 27.23,
  "ambienceTemperature": 29.35,
  "lightIntensity": 568.68,
  "rainfall": 325.13,
  "uvRay": 32.59,
  "sunCapability": 1.35
}
```

Os valores que foram enviados ao servidor e os sensores utilizados são melhor descritos nas tabelas de descrição 10 a 15.

Tabela 10 – Especificação do sensor de umidade e temperatura em relação a umidade

Sensor de umidade e temperatura relativa DHT11	
Descrição	Capturam os valores de umidade e temperatura em relação a umidade
Conexão	Se conecta com o arduino através da porta analógica 0 (A0)
Identificação dos valores	
Umidade	É representada por porcentagem (0 a 100%)
Temperatura	Capta a temperatura atual em °C

Tabela 11 – Especificações do sensor de temperatura ambiente

Sensor de temperatura ambiente Thermistor	
Descrição	Captura o valor da temperatura ambiente
Conexão	Se conecta com o arduino através da porta analógica 1 (A1)
Identificação dos valores	
Temperatura	Capta a temperatura ambiente atual em °C

Tabela 12 – Especificações do sensor de luminosidade LDR

Sensor de luminosidade LDR	
Descrição	Captura o valor da intensidade luminosa
Conexão	Se conecta com o arduino através da porta analógica 2 (A2)
Identificação dos valores	
< 100	Claridade Intensa
100 - 300	Claridade alta
300 - 500	Claridade acima da média
500 - 700	Claro
700 - 850	Pouca Claridade
850 - 1000	Muito Pouca Claridade
> 1000	Escuro

Tabela 13 – Especificações do sensor de nível de chuva

<b>Sensor de chuva</b>	
Descrição	Captura através do sensor digital se está chovendo (1) ou não (0) e através do sensor analógico de 0 a 4.5
Conexão	Se conecta com o arduino através da porta analógica 3 (A3)
Identificação dos valores	
< 2.00	Sem Chuva
2.00 – 3.00	Nuvens Esparsas
3.00 – 3.99	Nublado
4.00 – 4.50	Chuva
> 4.50	Chuva Intensa

Tabela 14 – Especificações do sensor de capacidade solar

<b>Sensor de capacidade solar</b>	
Descrição	Captura a capacidade solar utilizando fotocélula
Conexão	Se conecta com o arduino através da porta analógica 5 (A5)
Identificação dos valores	
< 1,00	Fraca
1,00 – 1,49	Média
1,50 – 1,99	Boa
>= 2,00	Ótima

Tabela 15 – Especificações do sensor de radiação uv

Sensor de radiação uv						
Descrição	Captura índice uv de forma analógica (0 - 240)					
Conexão	Se conecta com o arduino através da porta analógica 4 (A4)					
Identificação dos valores						
Índice UV	0	1	2	3	4	5
Voltagem (mV)	<50	227	318	408	503	606
Valor analógico	<10	46	65	83	103	124
Índice UV	6	7	8	9	10	11+
Voltagem (mV)	696	795	881	976	1079	1170
Valor analógico	142	162	180	200	221	240

Alguns dados foram tratados antes de serem armazenados em banco de dados, detalhes acerca da transformação de dados que foi aplicada são melhor abordados na seção 7.1.3.

### 6.3 NodeMCU

Para a conexão e envio dos dados para o servidor, foi utilizada a placa wifi NodeMCU ESP8266, conectada nas portas TX e RX.

A placa NodeMCU fornece possibilidade de conexão com rede wifi e roteamento wifi e pode ser programada em linguagem LUA ou utilizando a IDE do Arduino.

No projeto, o arduino se conecta com o NodeMCU através das portas TX e RX, onde o NodeMCU espera que os dados que foram capturados pelo Arduino são escritos em sua porta serial, então, os dados são enviados através de requisição HTTP para o servidor.

## 7 ARQUITETURA DO SOFTWARE

Segundo [Martin](#), um dos maiores pensadores sobre o assunto da qualidade de software na modernidade, em seu livro Clean Architecture (Arquitetura limpa): "O software foi inventado para ser soft. Pretende ser uma maneira de alterar facilmente o comportamento das máquinas. Se quiséssemos que o comportamento das máquinas fosse difícil de mudar, teríamos chamado de hardware". ([MARTIN, 2017](#)).

A arquitetura de software desenvolvida utiliza da separação dos componentes de software em pacotes que são exportados e podem ser utilizados em qualquer parte da aplicação, separando as responsabilidades do sistema e utilizando também os benefícios de técnicas do paradigma de programação funcional, foi possível isolar os pontos de efeitos colaterais da aplicação, ou seja, os pontos do software onde dados externos são alterados, como acesso a banco de dados, acesso a variáveis globais entre outros dados.

Para a construção, foi utilizado clássico modelo de cliente servidor, onde a aplicação do lado cliente tem a responsabilidade de renderizar o que o cliente vê (front end) e a aplicação no lado servidor tem a responsabilidade de aplicar regras de negócio, realizar armazenamento e análise dos dados solicitados (back end), no sistema construído, o lado cliente e o lado servidor se comunicam através de protocolo HTTP, utilizando o conceito restful no lado servidor que é descrito na seção [7.1](#) e o lado cliente utiliza a tecnologia SPA (Single Page Application), que é detalhada na seção [7.2](#)

### 7.1 Servidor

O lado do servidor foi construído utilizando NodeJS, um runtime para Javascript no lado servidor, foi disponibilizada para a aplicação uma API (Application Programming Interface) via HTTP para a comunicação tanto da interface do usuário quanto para o arduino, utilizando os métodos RESTful. Uma RESTful API é uma interface de acesso a aplicação que utiliza requisições http para a criação, atualização, consulta e exclusão de dados ([ROUSE, 2016](#)). No sistema desenvolvido, foi utilizado o formato de dados JSON.

A documentação da api pode ser conferida na seção [7.1.5](#).

#### 7.1.1 Segurança

A implementação da segurança do sistema foi feita através de técnica de autenticação de API utilizando JSON WEB TOKEN (JWT).

É enviado através do cabeçalho da requisição HTTP um token que guarda informações de acesso ao sistema, são armazenadas todas as informações do usuário no JSON WEB TOKEN com exceção de sua senha.

Segundo a RFC 7519 ([FORCE, 2015](#)) o JWT é uma maneira compacta e segura de transferir um objeto JSON entre duas partes, a listagem abaixo é um exemplo do token que é gerado:

```
{
  "exp": 1300819478,
  "data": {
    "_id": "8ab5x3c89fasd645261a3",
    "name": "John Doe",
    "email": "john.doe@host.com"
  }
}
```

Após o token ser construído, ele é assinado no servidor, ou seja, ele é criptografado através do algoritmo HS256 (HMAC 256), que utiliza uma senha para a assinatura do conteúdo ([ROUSE, 2010](#)), assim, o token só é decodificado pelo servidor, que é o único lugar no sistema que tem conhecimento de qual chave foi utilizada, caso ocorra um erro na decodificação do JWT, pode-se considerar o token inválido, negando então, o acesso do usuário ao sistema.

### 7.1.2 Banco de dados

O banco de dados utilizado foi o MongoDB, um banco de dados não relacional. O desenho do banco de dados é controlado pela aplicação, que define quais campos devem ser indexados e como os dados devem se "relacionar". Cada informação capturada fica armazenada dentro de uma coleção, utilizando o formato JSON.

A escolha de um banco de dados não relacional, foi motivada pela facilidade ao se trabalhar com esquemas de banco de dados maleáveis, assim, a estrutura do banco de dados pode ser facilmente modificada, ficando a cargo da aplicação. O banco MongoDB foi escolhido também pela sua facilidade ao trabalhar com escrita de dados concorrente, pois, as informações são, em primeiro instante processadas e armazenadas em cache, e já são retornadas para a aplicação, somente após, o MongoDB se encarrega de realizar a inserção dos dados em disco. Tais motivos, motivaram a escolha do banco de dados não convencional, pois, utilizando as técnicas descritas, ele se torna mais mutável e performática ao ser comparado com bancos de dados convencionais (relacionais).

Com todas as vantagens do banco de dados, a aplicação tira vantagem, se



beneficiando no que toca performance, disponibilidade e mutabilidade das informações.

### 7.1.3 Tratamento dos dados meteorológicos

Os dados que foram capturados pelas estações, tiveram seus valores mantidos, com exceção dos valores de intensidade de luz e quantidade de chuva, o valor de intensidade de luz precisou ser invertido para melhor visualização das informações.

Para a medição de intensidade de luz considerando  $x$  como o valor que foi capturado, com sua máxima identificada como 1024, a equação  $1024 - x$  é aplicada, invertendo o valor, caso o resultado seja negativo, o valor 0 é considerado em seu lugar. Para a quantidade de chuva, a mesma regra foi aplicada, invertendo o valor utilizando subtração, porém, neste caso, o valor também foi convertido para %, então a fórmula para obtenção do valor que será armazenado para quantidade de chuva será  $((1024 - x)/1024)100$ , novamente, dado resultado negativo, será considerada a porcentagem 0.

### 7.1.4 Análise dos dados

São retornados ao usuário os dados de médias, mínimas e máximas, por serem os valores mais significantes para o acompanhamento meteorológico na situação identificada. Para obtenção das médias dos valores, foi aplicada a função de agregação do MongoDB, reunindo as informações por intervalo de tempo e aplicando a função de média sobre o resultado.

### 7.1.5 Documentação da API

Na tabela 16 são detalhadas as rotas da api, documentando os métodos que foram utilizados, os endereços de cada uma das rotas e sua descrição.

Tabela 16 – Descrição das rotas da API

Método	Rota	Descrição
GET	/arduino	Lista os arduinos
GET	/arduino/:id	Retorna os dados de um arduino
POST	/arduino	Cadastra um novo arduino
POST, PATCH, PUT	/arduino/:id	Atualiza os dados de um arduino
DELETE	/arduino/:id	Deleta um arduino e suas medidas capturadas
GET	/user	Lista os usuários
GET	/user/:id	Retorna os dados de um usuário
POST	/user	Cadastra um novo usuário
POST, PATCH, PUT	/user/:id	Atualiza os dados de um usuário
DELETE	/user/:id	Deleta um usuário
POST	/user/login	Recebe as credenciais e retorna o token
POST	/measure	Cadastra uma medida capturada
GET	/statistic/:id	Retorna as estatísticas de dados do arduino

Nas tabelas 17 a 26, se pode conferir o detalhamento de cada uma das rotas da API, para todos os casos, os dados que forem retornados da API, são encapsulados na variável data dentro do corpo de resposta da requisição HTTP, no caso de listagem de dados, a variável data é um vetor que armazena as entidades listadas e no retorno de uma única entidade, a variável data é um objeto contendo os dados que foram retornados.

Juntamente com os dados, o campo success é retornado, quando verdadeiro, a requisição foi processada com sucesso, caso contrário, a requisição foi processada sem sucesso. Para todos os retornos, considere também os campos adicionais informados na tabela 18.

Tabela 17 – Especificações das rotas de listagem e retorno de arduino

GET /arduino e GET /arduino/:id		
Saída		
Propriedade	Tipo	Descrição
name	String	Nome da estação
location	String	Localização da estação

Tabela 18 – Descrição dos campos adicionais

Campos adicionais		
Propriedade	Tipo	Descrição
_id	String	Identificador da entidade no banco de dados
createdAt	String	Data de inserção do registro no banco de dados
updatedAt	String	Data da última atualização em dados do registro

Tabela 19 – Especificações das rotas de atualização e cadastro de arduino

<b>POST /arduino e POST /arduino/:id</b>		
Entrada		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome da estação
location	String	Localização da estação
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome da estação
location	String	Localização da estação

Tabela 20 – Especificações da rota exclusão de arduino

<b>DELETE /arduino/:id</b>		
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome da estação
location	String	Localização da estação

Tabela 21 – Especificações das rotas de listagem e retorno de usuário

<b>GET /user e GET /user/:id</b>		
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome do usuário
email	String	E-mail do usuário
password	String	Senha criptografada do usuário

Tabela 22 – Especificações das rotas de atualização e cadastro de usuário

<b>POST /user e POST /user/:id</b>		
Entrada		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome do usuário
email	String	E-mail do usuário
password	String	Senha do usuário
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome do usuário
email	String	E-mail do usuário
password	String	Senha criptografada do usuário

Tabela 23 – Especificações da rota de login de usuário

<b>POST /user/login</b>		
Entrada		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
email	String	E-mail do usuário
password	String	Senha do usuário
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome do usuário
email	String	E-mail do usuário
password	String	Senha criptografada do usuário
token	String	Token para autenticação no sistema

Tabela 24 – Especificações da rota exclusão de usuário

<b>DELETE /user/:id</b>		
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
name	String	Nome do usuário
email	String	E-mail do usuário
password	String	Senha criptografada do usuário

Tabela 25 – Especificações da rota de cadastro de medida

<b>POST /measure</b>		
Entrada		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
arduinoId	String	Id da estação
uvRay	Number	Medida de radiação UV
rainfall	Number	Medida da quantidade de chuva
sunCapability	Number	Nível de capacidade solar
humidity	Number	Porcentagem de umidade
ambienceTemperature	Number	Medida da temperatura ambiente
temperatureHumidity	Number	Medida da temperatura em relação a umidade
lightIntensity	Number	Medida da intensidade de luz

Tabela 26 – Especificações da rota de consulta de estatísticas do arduino

<b>GET /statistic/:id</b>		
Entrada		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
from	String	Data inicial
to	String	Data final
interval	String	Intervalo da seleção
Saída		
<b>Propriedade</b>	<b>Tipo</b>	<b>Descrição</b>
uvRay	Number	Medida de radiação UV
rainfall	Number	Medida da quantidade de chuva
sunCapability	Number	Nível de capacidade solar
humidity	Number	Porcentagem de umidade
ambienceTemperature	Number	Medida da temperatura ambiente
temperatureHumidity	Number	Medida da temperatura em relação a umidade
lightIntensity	Number	Medida da intensidade de luz

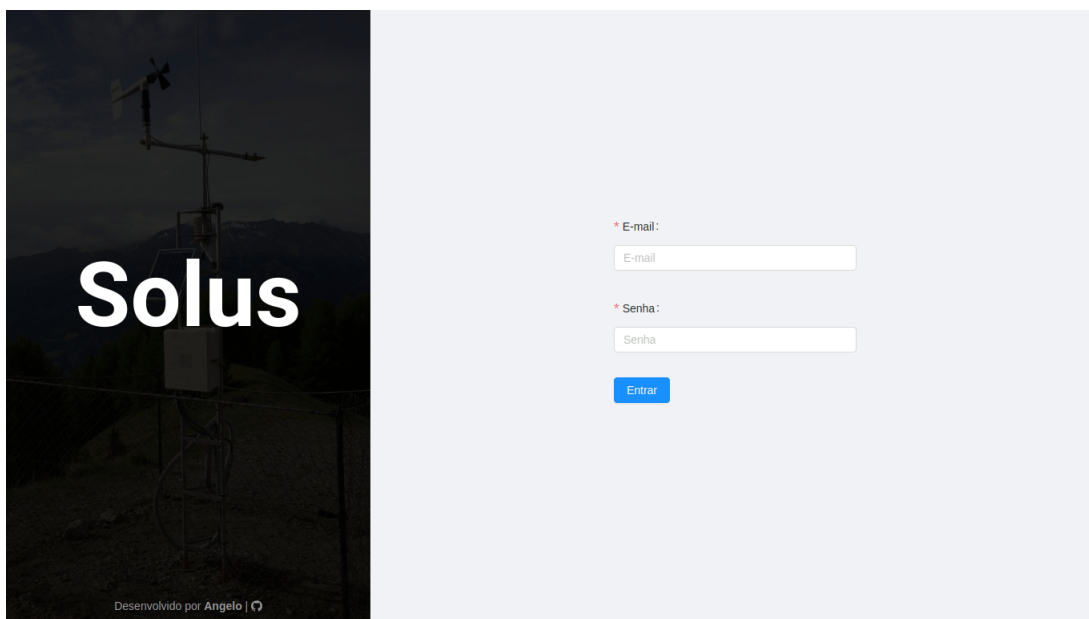
## 7.2 Cliente

Foi utilizado para a construção da aplicação lado cliente a biblioteca REACT, ela é uma biblioteca baseada em componentes e fornece, através da linguagem javascript diversas ferramentas para a construção de SPA's (Single Page Applications) (REACT, 2016). As SPA'S (Single Page Applications) são aplicações desenvolvidas para browser, utilizando javascript, que são capazes de fornecer ao usuário uma experiência sem atualização do navegador, de forma dinâmica.

### 7.2.1 Descrição das telas

Abaixo, pode-se conferir a documentação e descrição de cada uma das telas.

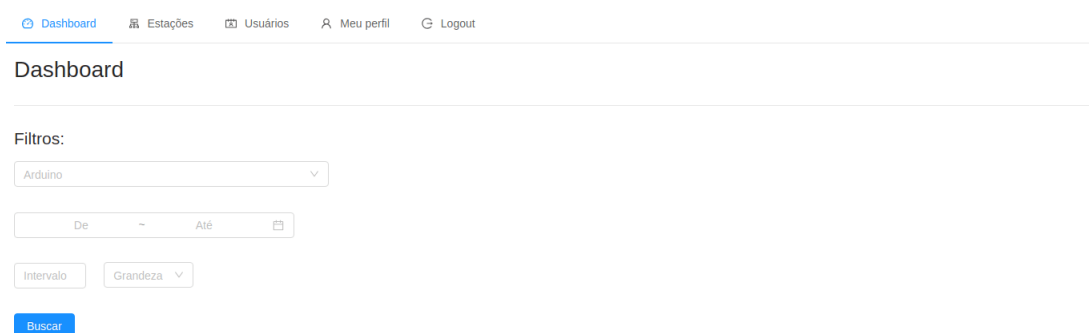
Figura 8 – Tela de login

A imagem mostra a tela de login do sistema Solus. À esquerda, há uma imagem de fundo escura com o nome "Solus" em branco e o texto "Desenvolvido por Angelo" no canto inferior esquerdo. À direita, há um formulário de login com campos para "E-mail" e "Senha", ambos com asteriscos vermelhos indicando obrigatoriedade. Abaixo dos campos, há um botão azul com o texto "Entrar".

Fonte: Produção do autor.

Na tela de login apresentada na figura 8, pode-se conferir o formulário de acesso ao sistema, requisitando o e-mail e senha do usuário e o botão de acesso, ao ser confirmado o login no sistema o usuário é levado até a tela principal do sistema.

Figura 9 – Dashboard

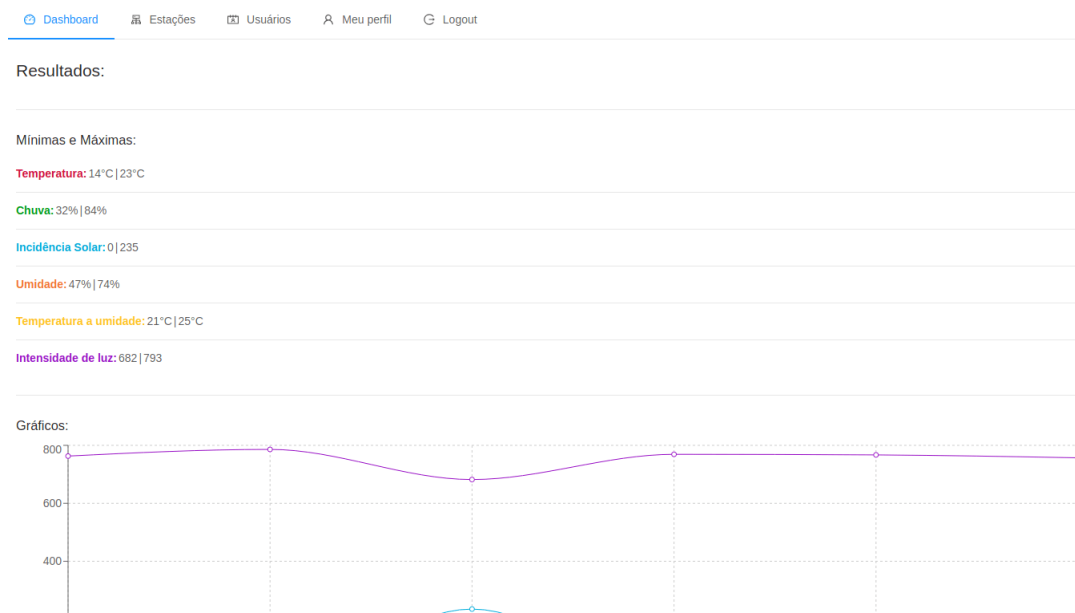
A imagem mostra o dashboard do sistema Solus. No topo, há uma barra de navegação com links para "Dashboard", "Estações", "Usuários", "Meu perfil" e "Logout". Abaixo, o título "Dashboard" é seguido por uma seção "Filtros:" com um menu suspenso para "Arduíno", campos para "De" e "Até" com um ícone de calendário, e botões para "Intervalo" e "Grandeza" com um menu suspenso. Um botão azul "Buscar" está na base dos filtros.

Fonte: Produção do autor.

Conforme pode-se conferir na figura 9 na tela inicial do sistema, pode-se conferir os filtros de dados, onde ao se escolher as informações de opção da estação mete-

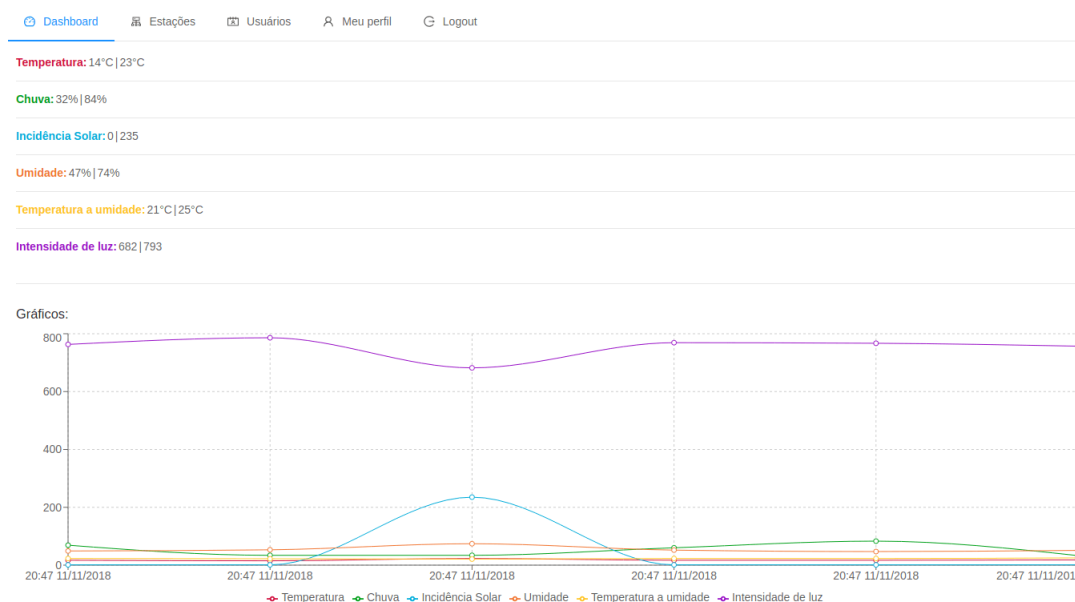
orológica desejada, data inicial e data final de captura e qual o intervalo de tempo e confirmar a seleção de estatísticas, os resultados das figuras 10 a 12 são exibidos.

Figura 10 – Resultados de estatísticas



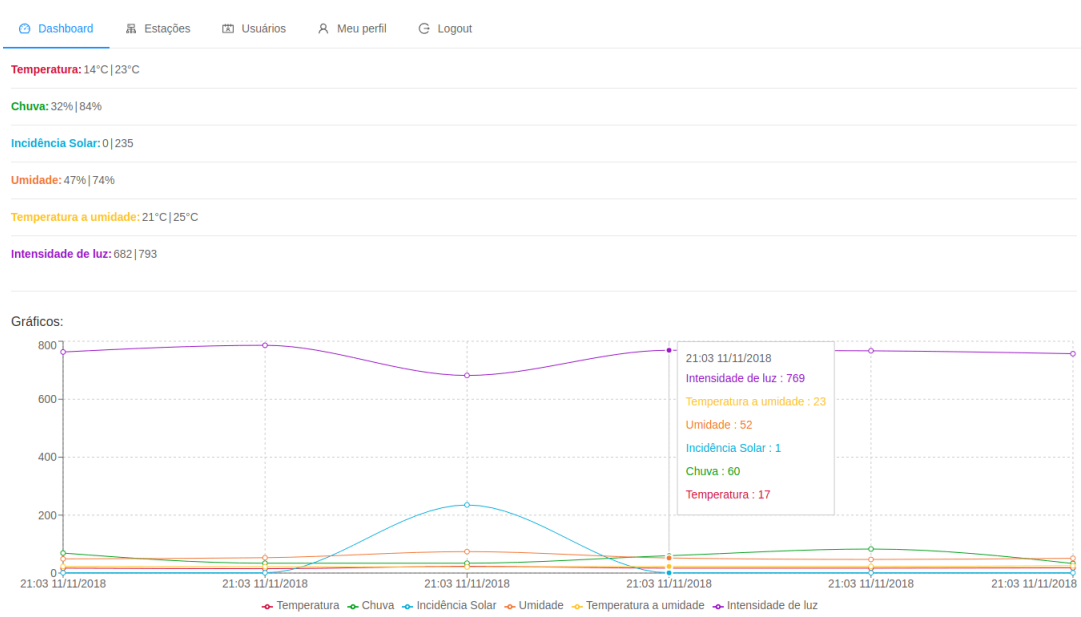
Fonte: Produção do autor.

Figura 11 – Gráficos das estatísticas



Fonte: Produção do autor.

Figura 12 – Números dos gráficos das estatísticas



Fonte: Produção do autor.

São exibidas ao usuário as medidas de mínimas e máximas dos dados e os gráficos das médias que foram calculadas, conforme figuras 10 a 12.

Figura 13 – Listagem de estações meteorológicas

Dashboard | **Estações** | Usuários | Meu perfil | Logout

### Estações meteorológicas

[Adicionar nova](#)

Nome	Local	Criado em	Última modificação	Ações
teste	teste	11/11/2018 19:33	11/11/2018 19:33	<a href="#">Editar</a> <a href="#">Deletar</a>

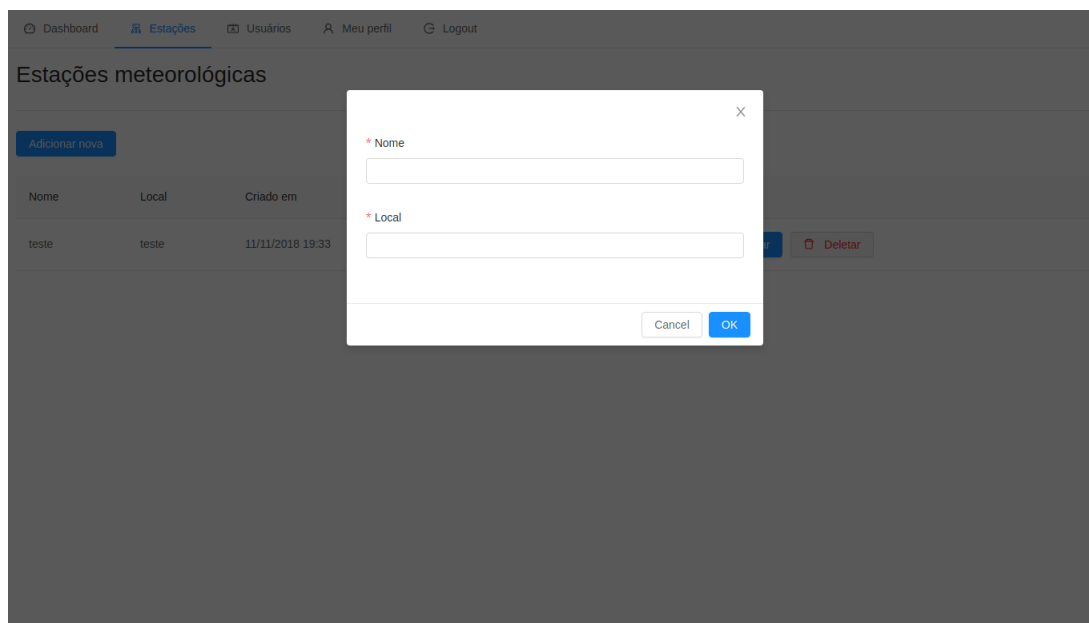
Fonte: Produção do autor.

Na listagem de estações meteorológicas (figura 13), é exibida ao usuário uma tabela com as informações das estações meteorológicas cadastradas, há no topo da



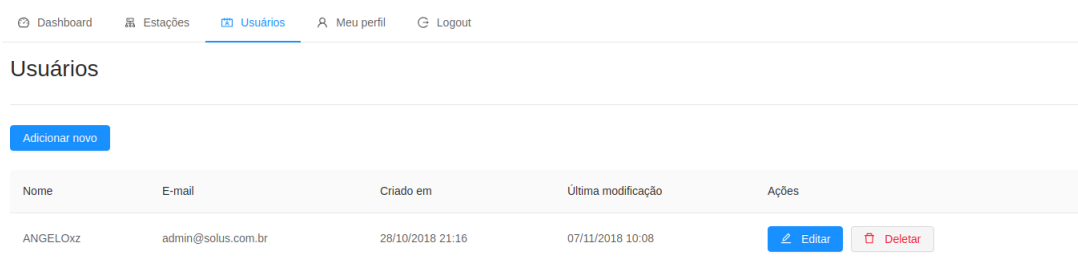
tabela a opção de adicionar uma nova estação meteorológica e para cada uma das linhas da tabela são exibidas as ações de editar ou excluir o registro de uma estação meteorológica, os modais de cadastro e edição podem ser conferidos na figura 14, porém, no caso da edição, o modal já abre com as informações preenchidas.

Figura 14 – Modal de adição e edição dos dados de uma estação



Fonte: Produção do autor.

Figura 15 – Listagem de usuários



Fonte: Produção do autor.

A tela de listagem de usuários apresentada na figura 15, segue o mesmo comportamento da tela de listagem de estações meteorológicas, porém, para o modal de cadastro de usuário (Figura 16), é adicionado o campo opcional para edição de senha e e-mail.

Figura 16 – Modal de adição ou edição de usuário

The screenshot shows a web application interface with a top navigation bar containing links for Dashboard, Estações, **Usuários**, Meu perfil, and Logout. The main content area is titled 'Usuários' and features a table with columns 'Nome' and 'E-mail'. The table contains one entry: 'ANGELOxz' with email 'admin@solus.com.br'. To the right of the table are 'Ações' buttons: 'Editar' and 'Deletar'. A modal form is open in the center, titled 'Usuários' with a close button (X). The form has three required fields: '\* Nome', '\* E-mail', and '\* Senha', each with a text input. At the bottom of the modal are 'Cancelar' and 'Salvar' buttons.

Fonte: Produção do autor.

Figura 17 – Visualização de meu perfil

The screenshot shows the 'Meu perfil' page. The top navigation bar is the same as in Figure 16, with 'Meu perfil' highlighted. The page title is 'Meu perfil'. Below the title is a section 'Meus dados' containing the following information: 'Nome: ANGELOxz', 'Email: admin@solus.com.br', 'Data de cadastro: 21:16 28/10/2018', and 'Última modificação: 10:08 7/11/2018'. At the bottom of this section is an 'Editar' button with a pencil icon.

Fonte: Produção do autor.

Na tela **Meu perfil**, conforme figura 17 são exibidos os dados do usuário que está utilizando o sistema no momento, caso o usuário abra a função de edição, é exibido o formulário conforme figura 18, que ao ser confirmado, é fechado novamente, exibindo somente suas informações já atualizadas.

Figura 18 – Edição de perfil

A imagem mostra a interface de usuário para a edição de perfil. No topo, há uma barra de navegação com links: Dashboard, Estações, Usuários, Meu perfil (destacado) e Logout. Abaixo, o título 'Meu perfil' é seguido por uma seção 'Meus dados' contendo três campos de texto: 'Nome' (com o valor 'ANGELOx'), 'E-mail' (com o valor 'admin@solus.com.br') e 'Senha' (com o valor 'Senha'). Abaixo dos campos, há dois botões: 'Salvar' (em azul) e 'Cancelar' (em cinza). No rodapé da seção, há um botão 'Editar' com um ícone de lápis.

Fonte: Produção do autor.

## 8 CONCLUSÃO

A captação de energia elétrica através de painéis fotovoltaicos depende diretamente das estações meteorológicas do local onde está instalada, portanto, dadas condições meteorológicas não satisfatórias, a quantidade de energia que é captada pelo painel solar tende a cair, uma estação solar precisa estar continuamente aquecida e recebendo incidência de radiação solar para que a eficiência energética seja maior.

Através da análise de dados utilizando métricas de mínima, máxima e média, foi possível visualizar mudanças climáticas que afetam a captação de energia solar, utilizando o sistema desenvolvido, o usuário pode tomar decisões acerca do posicionamento dos painéis solares, visando assim, mudanças que favorecem maior captação de energia solar através dos painéis fotovoltaicos.

O sistema desenvolvido é capaz de fornecer uma ferramenta ao especialista para análise e acompanhamento de informações importantes no acompanhamento de painéis solares, atingido assim seus objetivos.

Para projeto futuros, dadas as facilidades de mudança no software e na forma como os dados foram modelados, acrescentaria mais valor ao software, serem adicionadas novas medidas meteorológicas, enriquecendo mais a forma como o usuário é informado.

Por conta do alto custo dos sensores de velocidade do vento (anemômetro) e de quantidade de chuva em milímetros (pluviômetro), tais medidas não foram adicionadas no projeto e são medidas que trariam grandes avanços ao projeto. Funcionalidades geográficas como identificação das estações meteorológicas através de posicionamento em mapa seriam interessantes ferramentas para a análise meteorológica proposta.

## REFERÊNCIAS

ARDUINO. **Arduino: Introduction**. [s.n.], 2018. Disponível em: <<https://www.arduino.cc/en/Guide/Introduction>>. Acesso em: 03 nov. 2018. Citado na página 33.

BECK, K. **Test Driven Development By Example**. 1. ed. [S.l.]: Yahoo, 2002. Citado 2 vezes nas páginas 16 e 17.

BIRD, R.; WADLER, P. **Introduction to functional programming**. 1. ed. [S.l.: s.n.], 1988. Citado na página 16.

EVANS, E. **Domain Driven Design: Atacando as complexidades no coração do software**. 3. ed. [S.l.]: Dog Ear Publishing, 2014. Nenhuma citação no texto.

FORCE, I. E. T. **JSON Web Token**. [s.n.], 2015. Disponível em: <<https://tools.ietf.org/html/rfc7519>>. Acesso em: 03 nov. 2018. Citado na página 40.

GAMMA, E. et al. **Design Patterns: Elements of Reusable Object-Oriented Software**. 1. ed. [S.l.]: Addison-Wesley Professional, 1994. Nenhuma citação no texto.

GUEDES, G. T. A. **UML 2 - Uma Abordagem Prática**. 3. ed. [S.l.]: novatec, 2018. Citado 3 vezes nas páginas 17, 21 e 31.

IFSP. **1º Workshop do projeto de eficiência energética e P&D realizado entre o IFSP Câmpus Boituva**. [s.n.], 2018. Disponível em: <<https://btv.ifsp.edu.br/index.php/component/content/article/17-ultimas-noticias/2372-1-workshop-do-projeto-de-eficiencia-energetica-e-p-d-realizado-entre-o-ifsp-campus-boituva>>. Acesso em: 02 nov. 2018. Citado na página 15.

JSON. **ECMA-404 The JSON Data Interchange Standard**. [s.n.], 2017. Disponível em: <<https://www.json.org/>>. Acesso em: 03 nov. 2018. Citado na página 33.

KOLOSZUK, R.; SAUAIA, R. **Renováveis no Brasil: Maturidades diferentes para cada fonte exigem cuidados especiais**. [s.n.], 2018. Disponível em: <<http://www.absolar.org.br/noticia/artigos-da-absolar/renovaveis-no-brasil-maturidades-diferentes-para-cada-fonte-exigem-cuidados-especiais.html>>. Acesso em: 09 nov. 2018. Citado na página 14.

LIMA, D. de. **Modelos softwares com Astah Community**. [s.n.], 2016. Disponível em: <<https://www.techtudo.com.br/tudo-sobre/astah-community.html>>. Acesso em: 03 nov. 2018. Citado na página 18.

MARTIN, R. C. **Clean Architecture: A Craftsman's Guide to Software Structure and Design**. 1. ed. [S.l.]: Prentice Hall, 2017. Citado na página 39.

MARTINS, F. R. et al. **Projeto Sonda – Rede nacional de estações para coleta de dados meteorológicos aplicados ao setor de energia**. [S.l.]: Congresso Brasileiro de Energia Solar, 2007. Nenhuma citação no texto.

MONGODB. **What is MongoDB**. [s.n.], 2018. Disponível em: <<https://www.mongodb.com/what-is-mongodb>>. Acesso em: 03 nov. 2018. Citado na página 19.

MOZILLA. **Uma reintrodução ao JavaScript (Tutorial de JS)**. [s.n.], 2018. Disponível em: <[https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/A\\_re-introduction\\_to\\_JavaScript](https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/A_re-introduction_to_JavaScript)>. Acesso em: 03 nov. 2018. Citado na página 18.

NODEJS. **About Node.js**. [s.n.], 2018. Disponível em: <<https://nodejs.org/en/about/>>. Acesso em: 03 nov. 2018. Citado na página 18.

PRESSMAN, R. **Engenharia de Software: Uma Abordagem Profissional**. 7. ed. [S.l.]: AMGH, 2011. Citado na página 20.

QUEIROZ, A. **JavaScript assíncrono: callbacks, promises e async functions**. [s.n.], 2018. Disponível em: <<https://medium.com/@alcidesqueiroz/javascript-ass%C3%ADncrono-callbacks-promises-e-async-functions-9191b8272298>>. Acesso em: 30 out. 2018. Nenhuma citação no texto.

REACT. **What is React**. [s.n.], 2016. Disponível em: <<https://reactjs.org/tutorial/tutorial.html#what-is-react>>. Acesso em: 02 nov. 2018. Citado na página 45.

ROUSE, M. **Hash-based Message Authentication Code (HMAC)**. [s.n.], 2010. Disponível em: <<https://searchsecurity.techtarget.com/definition/Hash-based-Message-Authentication-Code-HMAC>>. Acesso em: 03 nov. 2018. Citado na página 40.

ROUSE, M. **RESTful API**. [s.n.], 2016. Disponível em: <<https://searchmicroservices.techtarget.com/definition/RESTful-API>>. Acesso em: 02 nov. 2018. Citado na página 39.

SADALAGE, P.; FLOWER, M. **NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence**. 1. ed. [S.l.]: Addison-Wesley Professional, 2012. Nenhuma citação no texto.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. [S.l.]: Pearson Education, 2011. Citado 5 vezes nas páginas 21, 24, 25, 27 e 29.