

Angelo Rodrigo Ribeiro da Silva

# **Solus**

Brasil

2018, v-0.0.1

Angelo Rodrigo Ribeiro da Silva

## **Solus**

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, campus de Boituva.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO  
PAULO – IFSP

Curso de Análise e Desenvolvimento de Sistemas

Orientador: Dr. Marcelo Figueiredo Polido

Brasil

2018, v-0.0.1

Angelo Rodrigo Ribeiro da Silva

Solus/ Angelo Rodrigo Ribeiro da Silva. – Brasil, 2018, v-0.0.1-  
26 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Marcelo Figueiredo Polido

Trabalho de conclusão de curso – INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA  
E TECNOLOGIA DE SÃO PAULO – IFSP

Curso de Análise e Desenvolvimento de Sistemas, 2018, v-0.0.1.

1. Palavra-chave1. 2. Palavra-chave2. 2. Palavra-chave3. I. Orientador. II. Univer-  
sidade xxx. III. Faculdade de xxx. IV. Título

# Resumo

O trabalho desenvolvido se baseia na dificuldade na análise de dados meteorológicos na instalação de painéis solares. Este documento visa descrever um software web focado em captar e descrever dados meteorológicos utilizando métricas de estatística.

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

# Abstract

The thesis developed is based on the difficulty in the analysis of meteorological data in the installation of solar panels. This document aims to describe a web software focused on capturing and describing meteorological data using statistics metrics.

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

# Lista de ilustrações

Figura 1 – Diagrama de caso de uso . . . . .	14
Figura 2 – Diagrama de classes . . . . .	17
Figura 3 – Diagrama de sequência . . . . .	18
Figura 4 – Diagrama de estado . . . . .	20
Figura 5 – Diagrama relacional do banco de dados . . . . .	22

# Lista de tabelas

Tabela 1	–	Especificações do caso de uso capturar dados meteorológicos . . . . .	15
Tabela 2	–	Especificações do caso de uso armazenar dados meteorológicos . . . . .	15
Tabela 3	–	Especificações do caso de uso realizar consultas com filtros . . . . .	15
Tabela 4	–	Especificações do caso de uso realizar análise estatística dos dados . . .	15
Tabela 5	–	Descrição das rotas da API . . . . .	24

# Lista de abreviaturas e siglas

API Application Programming Interface (em português Interface de Programação de Aplicações).

HTTP HyperText Transfer Protocol (em português Protocolo de transferência de hipertexto).

IFSP Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

MYSQL É um sistema de gerenciamento de banco de dados que utiliza a linguagem SQL

PHP Personal Home Page, uma linguagem de programação interpretada livre e voltada para a web.

RESTFUL API Full Representational State Transfer Application Programming Interface (em português é um serviço de API que segue a risca as definições da abstração REST)

SQL Structured Query Language (em português Linguagem de Consulta Estrutura)



# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>10</b>
	Introdução	10
1.1	Contextualização	10
1.2	Tema	10
1.3	Objetivo	10
1.4	Delimitação do Problema	10
1.5	Justificativa	11
1.6	Método	11
<b>2</b>	<b>DESCRIÇÃO GERAL DO SISTEMA</b>	<b>12</b>
	Descrição Geral do Sistema	12
2.1	Principais Envolvidos e suas Características	12
2.1.1	Usuários do Sistema	12
2.1.2	Desenvolvedores do Sistema	12
2.1.3	Tecnologias Empregadas	13
2.2	Regras de Negócio	13
<b>3</b>	<b>REQUISITOS DO SISTEMA</b>	<b>14</b>
	Requisitos do sistema	14
3.1	Requisitos funcionais	14
3.2	Requisitos não funcionais	16
3.2.1	Segurança	16
3.2.2	Disponibilidade	16
3.2.3	Performance	16
<b>4</b>	<b>DIAGRAMA DE CLASSES</b>	<b>17</b>
	Diagrama de classes	17
<b>5</b>	<b>DIAGRAMA DE SEQUÊNCIA</b>	<b>18</b>
	Diagrama de sequência	18
5.1	Controlador	18
5.2	Serviço de validação	18
5.3	Repositório	19

<b>6</b>	<b>DIAGRAMA DE ESTADO</b>	<b>20</b>
	Diagrama de estado	20
6.1	Validando requisição	20
6.2	Validando regras de negócio	20
6.3	Armazenando em banco de dados	21
<b>7</b>	<b>DESCRIÇÃO E ARQUITETURA DO BANCO DE DADOS</b>	<b>22</b>
	Descrição e arquitetura do banco de dados	22
7.1	Atores	22
7.2	Entidade de medida	22
7.3	Colunas de marcação de tempo	23
<b>8</b>	<b>DESCRIÇÃO E DOCUMENTAÇÃO DA API</b>	<b>24</b>
	Descrição e documentação da API	24
8.1	Métodos HTTP utilizados	24
8.2	Autenticação	24
<b>9</b>	<b>CONCLUSÃO</b>	<b>25</b>
	<b>REFERÊNCIAS</b>	<b>26</b>

# 1 Introdução

## 1.1 Contextualização

Na sociedade contemporânea, diversas preocupações quanto a captação de energia surgiram. Uma dessas preocupações é cada vez mais, buscar fontes renováveis de energia.

Atualmente, a energia solar vem mostrando seus benefícios, sendo pelo custo, que é mais baixo do que outras soluções e pela facilidade de instalação, que pode ser feita sem a necessidade de uma grande área reservada.

Devido ao avanço da captação de energia solar, desafios surgiram ao se estudar a melhor forma de se trabalhar com a energia captada.

## 1.2 Tema

O mote acerca do desenvolvimento desse trabalho, gira entorno do auxílio da tomada de decisão através do desenvolvimento de um software de análise e visualização de dados.

## 1.3 Objetivo

Este desenvolvimento tem como objetivo introduzir a questão da análise de dados meteorológicos e tomada de decisão acerca da instalação de painéis solares e documentar através de metodologias de descrição de software, a arquitetura e desenvolvimento de uma aplicação que visa auxiliar na resolução deste problema.

## 1.4 Delimitação do Problema

Não existe hoje uma forma prática, de baixo custo e precisa de realizar a análise de dados ante a instalação de painéis solares, visando, com uma massa de dados dispostos de maneira simples e organizada, auxiliar a tomada de decisão na instalação destes painéis, procurando através de variáveis como posição, ângulo entre outras, extrair o máximo de performance na captação de energia solar.

## 1.5 Justificativa

Existe um projeto de instalação de uma usina solar no IFSP, no campus localizado em Boituva, portanto, o tema do projeto foi escolhido, para que se possa através da análise prévia de dados, se encontrar a melhor configuração para os painéis fotovoltaicos, atingindo assim, uma maior captação de energia e aproveitamento de investimento.

## 1.6 Método

A metodologia de trabalho escolhida para este projeto, utiliza algumas convenções da metodologia SCRUM, porém, pelo tamanho limitado da equipe, o projeto foi trabalhado sendo ditado pela metodologia KANBAN. Para medirmos a qualidade do software, foi decidido optar pelo desenvolvimento da aplicação utilizando-se de técnicas de controle de qualidade orientadas ao teste.

## 2 Descrição Geral do Sistema

A arquitetura da aplicação desenvolvida pode ser resumida em dados sendo capturados através de sensores conectados a um microcontrolador arduino, serão captados dados de umidade do ar, temperatura e incidência de radiação solar.

Dados esses, que serão enviados através de requisições HTTP para uma API, serão armazenadas em banco de dados e então, será feita uma análise estatística dessas informações.

A interface do usuário final com a aplicação, será feita através de uma aplicação web, onde os dados analisados serão disponibilizados e o usuário fará consultas a essas informações.

### 2.1 Principais Envolvidos e suas Características

#### 2.1.1 Usuários do Sistema

O sistema visa atender especialistas que precisam realizar tomadas de decisão.

Isso inclui também, clientes que, antes de realizar a instalação de painéis solares, precisam analisar se o investimento será compensado. E também, empresas de instalação de painéis solares, que gostariam de fazer uma análise de viabilidade mudando local, angulo e fazendo outras pesquisas acerca da instalação ou da manutenção de painéis fotovoltaicos.

#### 2.1.2 Desenvolvedores do Sistema

Os envolvidos no desenvolvimento do projeto, são o orientador, Dr. Marcelo Polido, que ficará responsável pelos requisitos do sistema, ele irá coordenar o que será implementado e irá ditar as entregas incrementais. Também responsável por requisitos do projeto está o Professor Mario Pin, que será algo próximo de um Product Owner, ele será o primeiro cliente final da aplicação, irá utilizar o sistema para realizar análise de dados.

O planejamento e desenvolvimento do projeto, ficará por conta do aluno responsável pela defesa do mesmo, Angelo Silva.

O projeto é open source, ou seja, aberto para a comunidade no github, recebendo então, pequenas contribuições esporádicas de outros desenvolvedores ao longo do ciclo de vida do projeto.

### 2.1.3 Tecnologias Empregadas

A aplicação foi desenvolvida utilizando arduino para gerenciamento e captura dos dados utilizando requisições HTTP através das libraries do arduino. A conexão com a internet foi feita utilizando um arduino ethernet shield wifi, as informações são capturadas, é feita uma validação e formatação básica dos dados e então os mesmos são enviados para uma RESTFUL API construída com NODEJS.

O banco de dados utilizado foi o MYSQL, sendo servido a engine XtraDB, servido pela aplicação percona server.

## 2.2 Regras de Negócio

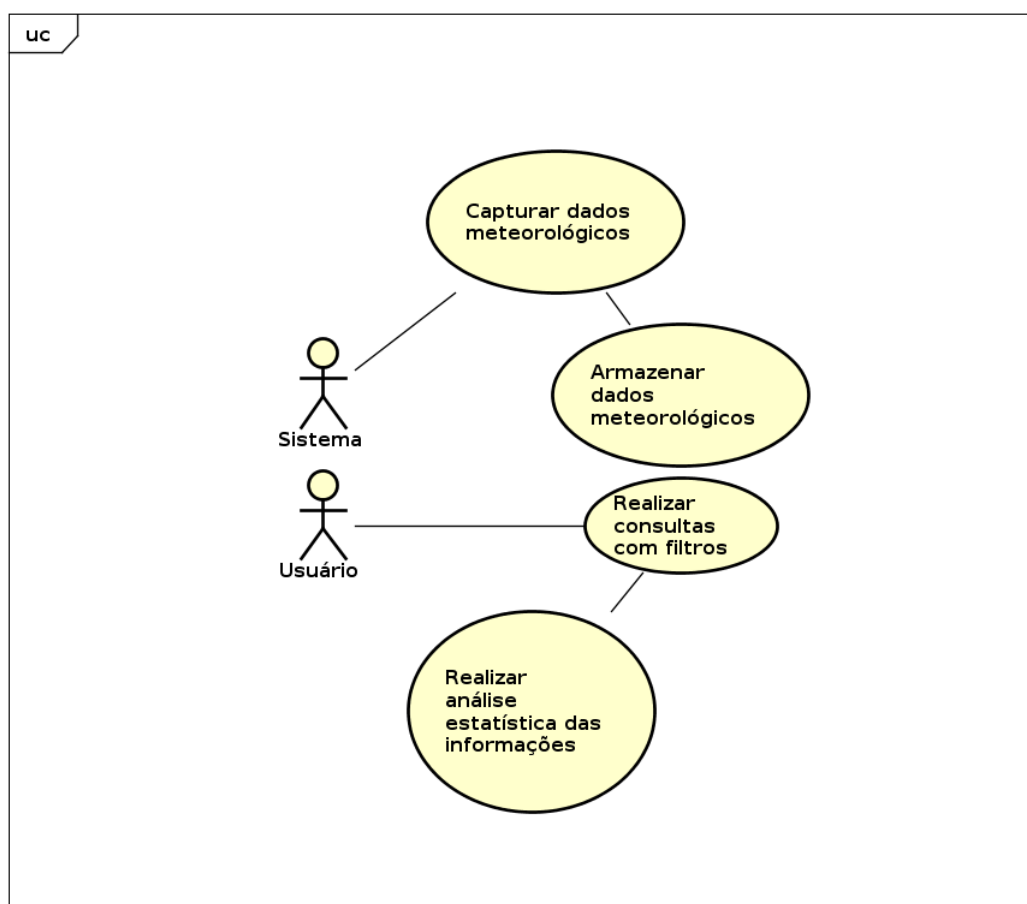
A maior parte das regras de negócio de sistema fica centralizada nos filtros, eles são quem valida os dados do sistema, definindo regras para a captura de dados. A regra de negócio de filtragem diz que os dados de umidade e temperatura não podem se diferenciar por 50% ou mais da média dos dados captados na ultima hora.

## 3 Requisitos do sistema

### 3.1 Requisitos funcionais

Os requisitos funcionais do sistema são definidos pelas características para que um MVP possa ser entregue. Os requisitos são descritos no diagrama de caso de uso abaixo.

Figura 1 – Diagrama de caso de uso



Os atores do sistema foram definidos com o próprio sistema, que poderá realizar acesso a alguns casos de uso e o usuário final. Seguem abaixo as especificações dos casos de uso.

Tabela 1 – Especificações do caso de uso capturar dados meteorológicos

<b>Capturar dados meteorológicos</b>	
Descrição	Captura os dados meteorológicos através de sensores conectados ao microcontrolador arduino e os envia através de requisição HTTP para a API
Atores	Sistema
Pré-condições	Credenciais de acesso a API API em correto funcionamento
Exceções e fluxos alternativos	Em caso de perda de conexão com internet ou a API, armazena as informações temporariamente no arduino

Tabela 2 – Especificações do caso de uso armazenar dados meteorológicos

<b>Armazenar dados meteorológicos</b>	
Descrição	Após receber os dados captados pelo arduino, a api os valida e então, os armazena em banco de dados
Atores	Sistema
Pré-condições	Banco de dados em correto funcionamento Recebimento de dados através das rotas da API
Exceções e fluxos alternativos	Em caso de dados inválidos, a API os descarta Em caso de perda de conexão com banco de dados, a API os armazena em memória

Tabela 3 – Especificações do caso de uso realizar consultas com filtros

<b>Realizar consultas com filtros</b>	
Descrição	Recebe do usuário os filtros para seleção das informações, então, realiza uma análise estatística dos dados requeridos e os exibe para o usuário utilizando gráficos.
Atores	Usuário
Pré-condições	Credenciais de acesso ao banco de dados para realizar as consultas Banco de dados em correto funcionamento Filtros corretos passados pelo usuário
Exceções e fluxos alternativos	Caso ele não encontre dados na seleção, exibe uma mensagem de não encontrado Em caso de perda de conexão com banco de dados, exibe uma tela de erro ao usuário

Tabela 4 – Especificações do caso de uso realizar análise estatística dos dados

<b>Realizar análise estatística dos dados</b>	
Descrição	Realiza calculo estatístico de informações, retornando informações relevantes como média, moda e desvio padrão.
Atores	Usuário
Pré-condições	Credenciais de acesso ao banco de dados Banco de dados em correto funcionamento
Exceções e fluxos alternativos	Caso não existam dados para serem analisados, joga uma exceção de argumentos inválidos



## 3.2 Requisitos não funcionais

Os requisitos não funcionais do sistema complementam os requisitos funcionais, como melhorias para as especificações.

### 3.2.1 Segurança

O projeto precisa trabalhar de forma segura, então, esse requisito pede para que a API possua autenticação das aplicações clientes e que também, a aplicação web possua autenticação, para o usuário visualizar as informações e realizar consultas, ele precisa estar autenticado.

### 3.2.2 Disponibilidade

Para garantir uma melhor análise e fidelidade dos dados, o sistema precisa funcionar durante 24 horas por dia e 7 dias por semana, para isso, redundâncias precisam ser trabalhadas.

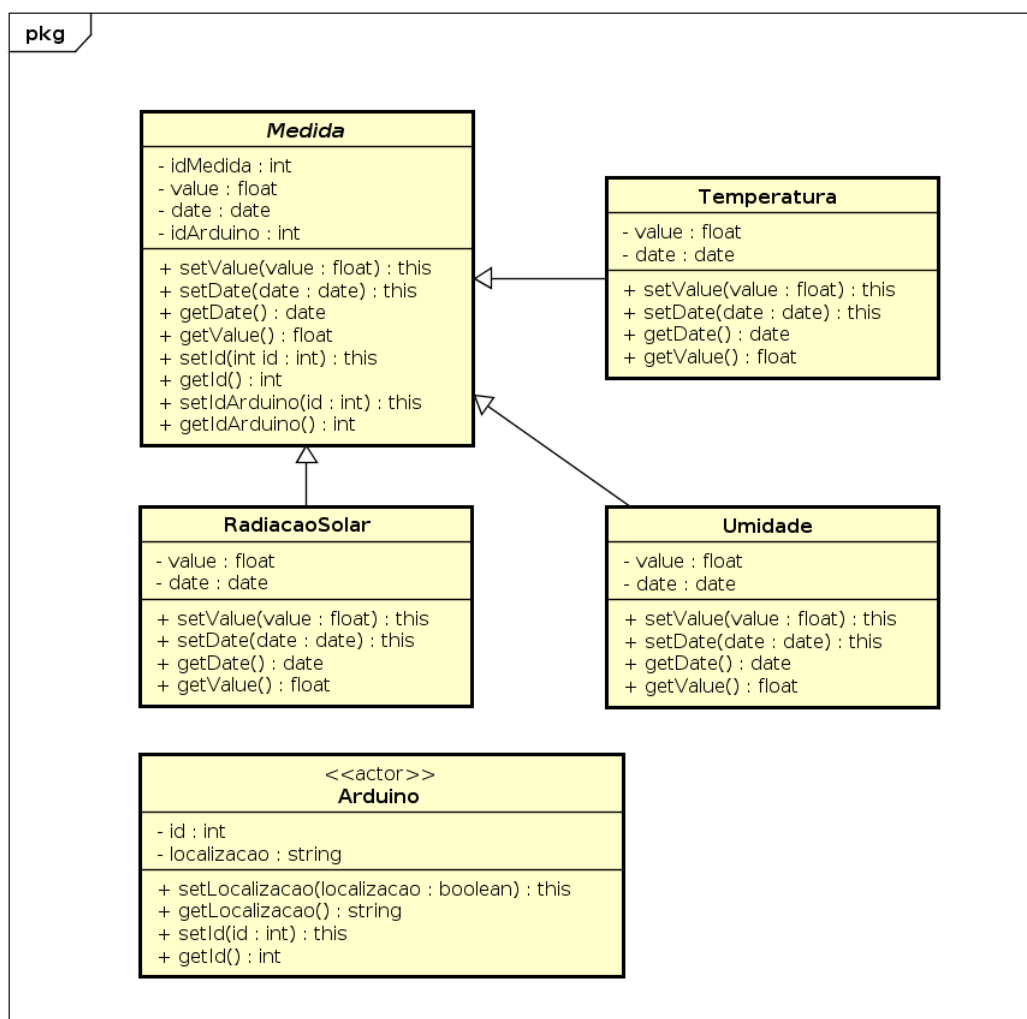
### 3.2.3 Performance

Outro requisito não funcional do sistema é a performance, as informações precisam ser processadas de forma rápida, problemas como lentidão no processamento podem acabar travando a entrada de dados no sistema.

## 4 Diagrama de classes

Como podemos verificar no diagrama descrito na figura 4 as entidades do sistema consistem na medida, que é a informação que foi capturada pelo arduino, uma entidade abstrata que é entendida pelas classes concretas das medidas, temos também uma entidade que representa o arduino, possuindo a localização do mesmo.

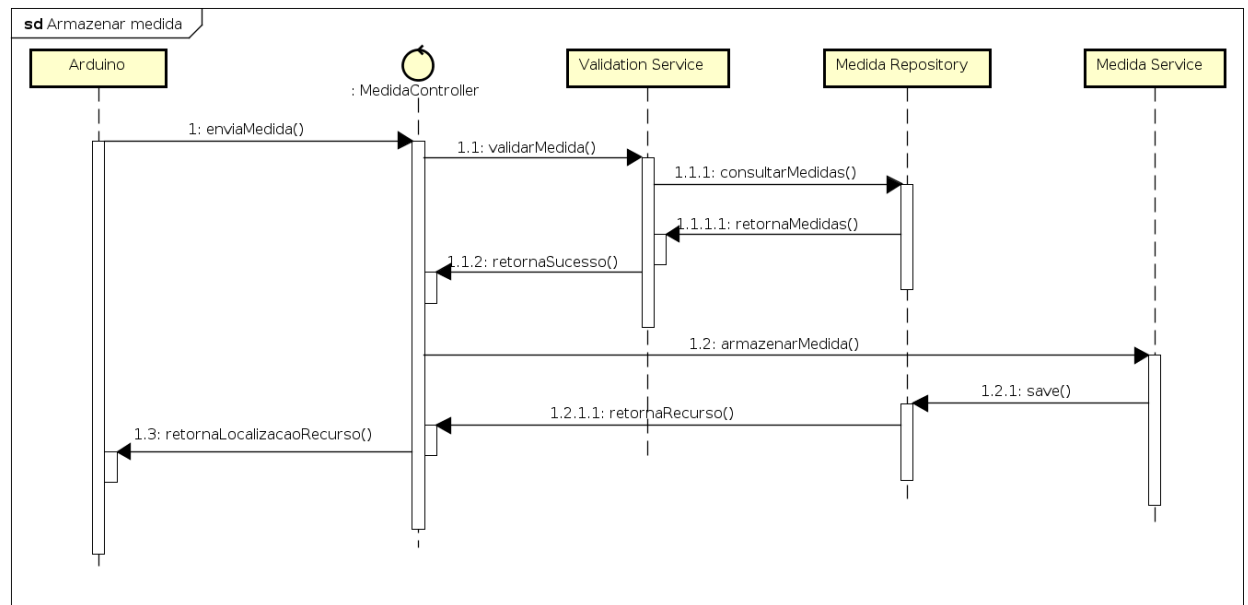
Figura 2 – Diagrama de classes



## 5 Diagrama de sequência

Apresentação do diagrama de sequencia do projeto.

Figura 3 – Diagrama de sequência



Como descrito no diagrama de sequência representado na figura 5 a aplicação é composta por 3 principais camadas.

### 5.1 Controlador

Na camada de controle, os dados são recebidos e passam por uma básica validação através, porém não sofrem a interferência das regras de negócio.

Nessa camada, os dados são recebidos e retornados ao cliente, é uma interface de acesso a aplicação, geralmente, essa camada recebe objetos de requisições HTTP.

### 5.2 Serviço de validação

No serviço de validação, regras de negócio são aplicadas a medida para verificar se a mesma é válida para ser armazenada.

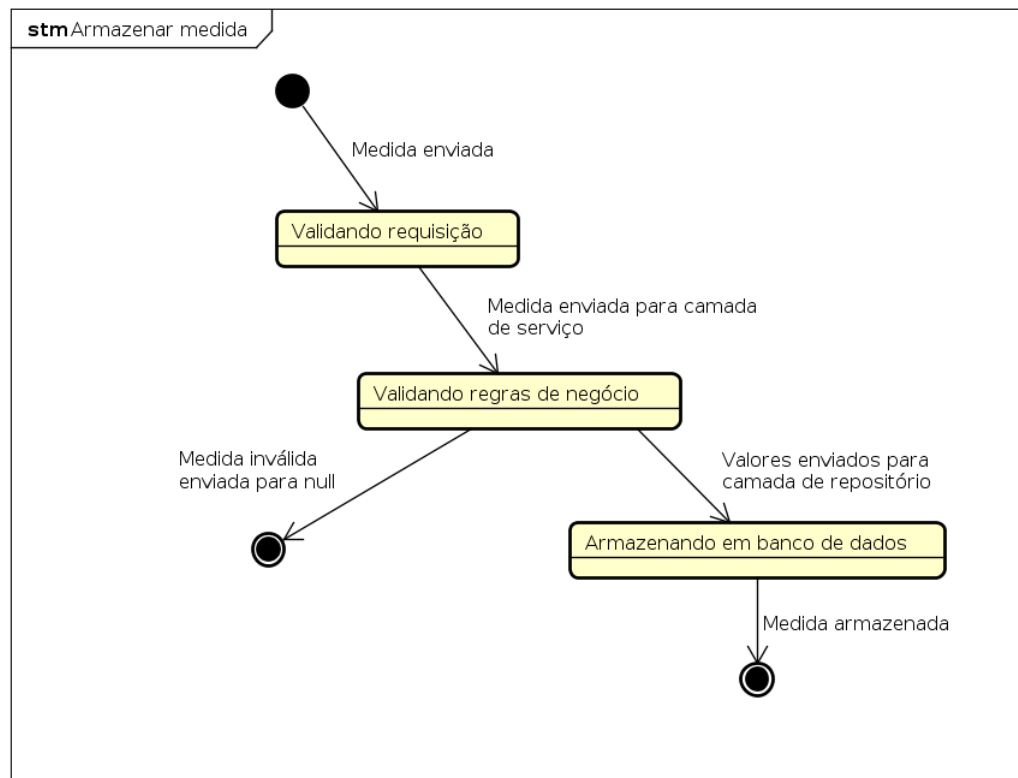
## 5.3 Repositório

Dentro da camada de repositório, são recebidos dados que, através de uma camada de infraestrutura são persistidos, retornando então, uma entidade.

## 6 Diagrama de estado

Apresentação dos estados de uma entidade de medida que foi capturada ao longo da interação com o sistema.

Figura 4 – Diagrama de estado



Conforme descrito na figura 6 os estados durante o armazenamento de uma de uma medida são:

### 6.1 Validando requisição

A autenticação do arduino na API foi feita e a requisição HTTP para o armazenamento de uma medida foi feita até a API. Nesse estado, a requisição está passando por uma validação básica, verificando os tipos e formato dos dados.

### 6.2 Validando regras de negócio

A medida passou pelo controlador e está sendo validada através dos filtros, onde as regras de negócio de validação estão verificando se aquela medida faz sentido, baseando-se

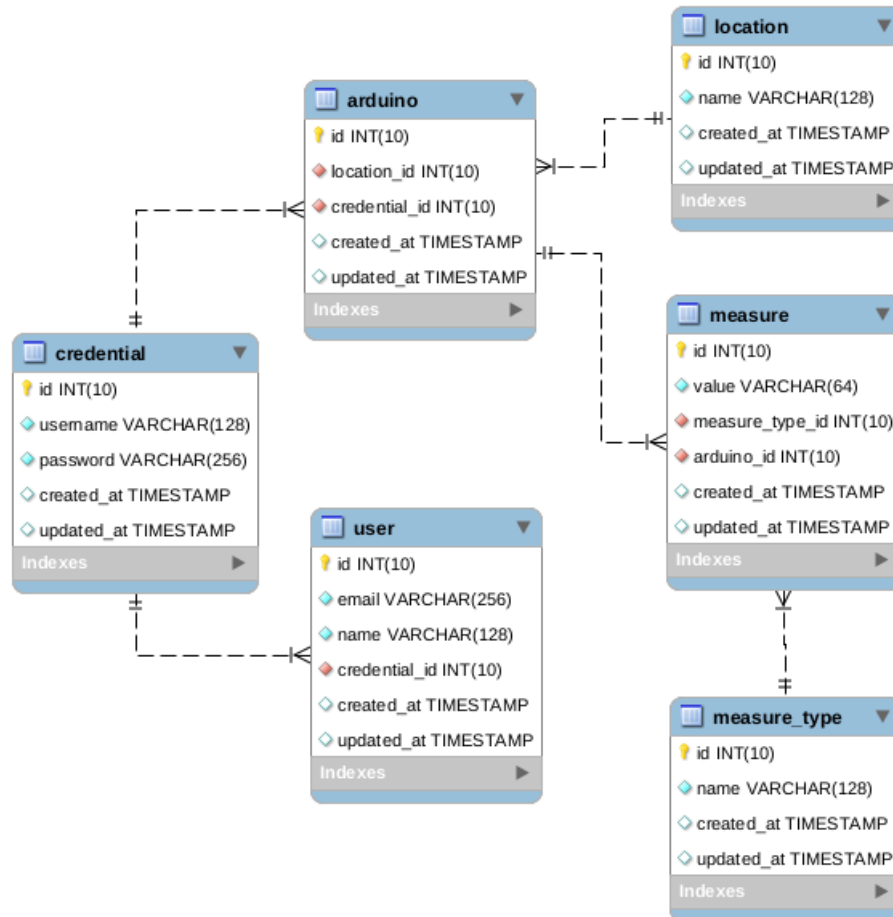
nas medidas captadas anteriormente.

## 6.3 Armazenando em banco de dados

A medida é válida e está sendo armazenada no banco de dados, após, ela é retornada a camada de serviço.

## 7 Descrição e arquitetura do banco de dados

Figura 5 – Diagrama relacional do banco de dados



### 7.1 Atores

Para representação dos atores foi definida uma tabela chamada credencial, que é utilizada para armazenar os dados de uma entidade que acessa a API. O arduino é representado por uma localização e por uma credencial, já o usuário possui nome e e-mail, também possuindo uma credencial de acesso.

### 7.2 Entidade de medida

A representação das medidas no banco de dados é definida pela tabela tipo da medida, que possui qual a medida que é capturada, temperatura, radiação solar, etc. Também há uma tabela chamada medida, que representa a entidade medida, que possui uma identificação do tipo da medida capturada, do arduino que fez a captura e do valor.

## 7.3 Colunas de marcação de tempo

Há também as colunas de marcação de tempo em cada uma das tabelas, que são o momento em que o registro foi criado e o momento em que o mesmo foi atualizado, caso o registro nunca tenha sofrido atualização, a coluna de atualização é igual a de criação.



## 8 Descrição e documentação da API

Podemos conferir abaixo a documentação das rotas da API.

Tabela 5 – 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	/arduino/:id	Atualiza os dados de um arduino
DELETE	/arduino/:id	Deleta um arduino e suas medidas capturadas
GET	/arduino:id/measure	Retorna as medidas capturadas por um arduino
POST	/arduino/:id/measure	Cadastra uma medida em um arduino
GET	/measure/:id	Retorna os dados de uma medida
POST	/authenticate	Retorna o Json Web Token de uma credencial
POST	/location	Cria uma nova localização
GET	/location/:id	Retorna os dados de uma localização
POST, PATCH	/location/:id	Atualiza os dados de uma localização
DELETE	/location/:id	Deleta uma localização

### 8.1 Métodos HTTP utilizados

Como podemos visualizar na tabela 5, o método POST fica designado a criar um recurso quando a requisição for enviada para uma rota sem a identificação. O método POST também é utilizado para atualizar um recurso quando a requisição for enviada para um recurso identificado. Para a atualização de um recurso também pode ser utilizado o método PATCH. Para a consulta de recursos é utilizado o método GET, o método DELETE é utilizado para excluir um recurso.

### 8.2 Autenticação

Uma exceção do método POST é na autenticação, os campos nome de usuário e senha são enviados para a API, que retorna um Json Web Token para que possa ser feita a autenticação de forma moderna.

## 9 Conclusão

Conclusão

## Referências

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

GAMMA RICHARD HELM, R. J. E.; VLISSIDES, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1. ed. [S.l.]: Addison-Wesley Professional, 1994. Nenhuma citação no texto.

MARTINS RICARDO ANDRÉ GUARNIERI, R. C. C. S. L. M. N. E. B. P. E. A. F. R.; THOMAZ, C. Projeto sonda – rede nacional de estações para coleta de dados meteorológicos aplicados ao setor de energia. Congresso Brasileiro de Energia Solar, n. 1, 2007. Nenhuma citação no texto.