

UNIVERSIDADE DO OESTE DE SANTA CATARINA

CIÊNCIA DA COMPUTAÇÃO

GABRIEL CATANI

LUCAS KLAUCK

IGOR BERNARDI

EDUARDO TONKELSKI

TRABALHO FINAL INTEGRADO DE PII, ESI E BDII

CORONA PEEK

SÃO MIGUEL DO OESTE

2021

GABRIEL CATANI

LUCAS KLAUCK

IGOR BERNARDI

EDUARDO TONKELSKI

TRABALHO FINAL INTEGRADO DE PII, ESI E BDII

CORONA PEEK

Trabalho sobre o coronavírus.

Apresentado no Curso de Ciências da Computação.

Na matéria de Banco de Dados II, Programação II e Eng. De Software I.

SÃO MIGUEL DO OESTE

2021

## SUMÁRIO

1.....	Intodução.
1.1.....	Padronização-Finalidade.
2.....	Características.
3.....	Angular/Front-End.
4.....	Java/Back-End.
4.1.....	Spring-Boot..
4.2.....	Lombok.
5.....	Banco De Dados.
5.1.....	Postgresql.
6.....	Diagramas/UML.
6.1.....	Diagramas/Modelo Relacional.
7.....	Códigos/Scripts/SQL.
7.....	REFERÊNCIAS.

## 1. Introdução

Trabalho Integrado Interdisciplinar para desenvolvimento e aplicação dos conhecimentos das matérias de Banco de Dados II, Programação II e Eng. De Software I em uma aplicação WEB completa - Diagramas De Classe, Front-End, Back-End E Banco De Dados em formato CRUD.

### 1.1 Padronização - Finalidade

Sistemas De Login, Autenticação, Permissões, Cadastro De Usuário, Administradores, Empresas, Pacientes, Sintomas.

## 2. Características.

Aplicação desenvolvida em diversas linguagens e frameworks para melhor desempenho, listando alguns abaixo:

### 3. Angular / Front-End


- Angular é uma plataforma de aplicações web de código-fonte aberto e front-end baseado em TypeScript liderado pela Equipe Angular do Google e por uma comunidade de indivíduos e corporações. Angular é uma reescrita completa do AngularJS, feito pela mesma equipe que o construiu.
- O Angular é um poderoso framework que utiliza HTML e TypeScript para criar a interface com o usuário, ou seja, o front-end em aplicações web, desktop e dispositivos móveis.

Utilizamos o Angular em basicamente todo o Front-End devido ao seu excelente uso e estabilidade além de ser possível escalável futuramente para outros dispositivos


Outra vantagem do Angular é suas Bibliotecas De Componentes, dentre eles utilizamos duas, são elas: Angular Material e PrimeFace.


## - Login

Corona Peak



Cadastrar










Cadastrar

## -Dashboard.

Corona Peak

-  Início
-  Questionário paciente
- 
-  Detalhes da conta
-  Sair

### Relatórios dos dados coletados

Código e nome dos pacientes com idade ímpar que apresentam como sintoma febre, ordenado de forma ascendente por nome

Código do paciente	Nome do paciente
241	Roberto
221	Lucas
254	Gabriel Catani
231	Luiz Gustavo

Nome do paciente, nome da cidade que reside, de paciente que possuam mais de 60KG, residam nos municípios de Maravilha, Descanso, Pinhalzinho, Chapecó ou Itapiranga, que apresentam algum sintoma e que foram positivados com COVID. Ordenado de forma ascendente por nome da cidade e descendente por nome do paciente

Nome do paciente	Nome da cidade
Luiz Gustavo	Descanso
Gabriel Catani	Descanso
Roberto	Maravilha
Lucas Eduardo Klauck	Maravilha

## - Cadastro Perfil Paciente.

Corona Peak

Início

Questionário paciente

Detalhes da conta

Sair

Nome paciente

Data nascimento

Peso

0,0

Altura

0,0

Selecione uma cidade

Selecione uma empresa

Suspeito

Falso

Positivado

Falso

Doença respiratória

Falso

Selecione os sintomas

Salvar

## - Detalhes Da Conta.

Corona Peak

Início

Detalhes da conta

Sair

Detalhes da conta

Nome completo

E-mail

Nova senha

Data nascimento

Senha atual

Salvar

-Opções De Cadastro.

☰

Corona Peak

🏠

Início

+

Cadastro

👤

⚙️

Detalhes da conta

🚪

Sair

Cidade

Empresa

Sintoma

Paciente

-Filtro Relatórios.

☰

Corona Peak

🏠

Início

+

Cadastro

👤

⚙️

Detalhes da conta

🚪

Sair

Nome paciente

Data nascimento

Peso

0,0

Altura

0,0

Selecione uma cidade

▼

Selecione uma empresa

▼

Suspeito

Falso

Positivado

Falso

Doença respiratória

Falso

Selecione os sintomas

▼

Salvar

Nome	Data nascimento	Peso	Altura	Cidade	Empresa	Suspeito	Positivado	Doença Resp.	Editar	Excluir
Lucas	22/01/2000	55	175	Romelândia - SC	Sysmo	SIM	SIM	NÃO		
Lucas Eduardo Klauck	03/12/2021	80	185	Maravilha - SC	EduCell	NÃO	SIM	NÃO		
Luiz Gustavo	02/12/2004	65	170	Descanso - SC	Sysmo	NÃO	SIM	NÃO		

## 4. Java / Back-End

Para o desenvolvimento do Back-End da aplicação foi escolhida a linguagem JAVA, apresentada na matéria de Programação II é extremamente poderosa para esse requisito, podemos controlar melhor a memória, gerenciamento, requerys e demais tarefas necessitadas pelo Back-End, também foram utilizados dois o frameworks muito poderosos Spring-Boot e Lombok.

### 4.1 Spring-Boot

Spring Boot é um projeto da Spring que veio para facilitar o processo de configuração e publicação de nossas aplicações. A intenção é ter o seu projeto rodando o mais rápido possível e sem complicação.

Ele consegue isso favorecendo a convenção sobre a configuração. Basta que você diga pra ele quais módulos deseja utilizar (WEB, Template, Persistência, Segurança, etc.) que ele vai reconhecer e configurar.

Você escolhe os módulos que deseja através dos starters que inclui no pom.xml do seu projeto. Eles, basicamente, são dependências que agrupam outras dependências. Inclusive, como temos esse grupo de dependências representadas pelo starter, nosso pom.xml acaba por ficar mais organizado.

Apesar do Spring Boot, através da convenção, já deixar tudo configurado, nada impede que você crie as suas customizações caso sejam necessárias.

O maior benefício do Spring Boot é que ele nos deixa mais livres para pensarmos nas regras de negócio da nossa aplicação.

### 4.2 Lombok

O Lombok é um framework para Java que permite escrever código eliminando a verbosidade, o que permite ganhar tempo de desenvolvimento para o que realmente é importante. Seu uso permite gerar em tempo de compilação os métodos getters e setters, métodos construtores, padrão builder e muito mais

## 5. Banco de Dados.



Para ser um projeto completo foi necessário utilizar um Banco de Dados para gerenciar as informações da aplicação, o escolhido foi o postgresql apresentado na matéria de Banco de Dados II.

O banco de dados é a organização e armazenagem de informações sobre um domínio específico. De forma mais simples, é o agrupamento de dados que tratam do mesmo assunto, e que precisam ser armazenados para segurança ou conferência futura.

É comum que empresas tenham diversas informações que precisam ser organizadas e disponibilizadas dentro do negócio para que sejam consultadas posteriormente pela equipe e pela gerência.

Por isso, é interessante ter um sistema de gerenciamento de banco de dados, SGBD, para conseguir manipular as informações e tornar a rotina da empresa muito mais simples.

Hoje, existem diversos tipos de SGBDs, e cada um é adequado para uma necessidade dos clientes. São os mais comuns: Oracle, DB2, MySQL, SQL Server, PostgreSQL e outros.

### 5.1 Postgresql.

O banco de dados PostgreSQL é muito popular no mercado de tecnologia. Isso se deve ao fato da sua fácil integração com ferramentas e sistemas legados nas empresas.

Essa base funciona como um background para armazenar as informações geradas e processadas pelas aplicações e interfaces com usuários. Assim, consegue-se uma arquitetura de sistemas de TI escalável e flexível para as necessidades do seu negócio.

- As funções mais relevantes do Postgres são:
- heranças de tabelas;
- integridade de dados referencial via chaves estrangeiras;
- tipos de dados definidos pelo usuário;

- controle de concorrência multi versionado;
- recuperação de informações point-in-time;
- replicação assíncrona de dados;
- subconsultas;
- transações aninhadas via savepoints;
- controle de acesso aos dados; e
- tablespaces.

Os sistemas de banco de dados PostgreSQL viabilizam o padrão de arquiteturas Modelo-Visão-Controle (MVC) na infraestrutura de tecnologia das empresas. Com isso, é possível otimizar o processamento e a guarda das informações relevantes para o negócio. Trata-se de uma aplicação do conceito de APIs.

Confira agora as 3 vantagens do Postgres:

### 1. Extensibilidade

O sistema PostgreSQL permite a utilização de operadores, estruturas de dados, tipagens e métodos de acesso definidos pelo próprio programador. Tudo isso facilita o desenvolvimento e a manutenção dos softwares hospedados na infraestrutura da sua empresa.

### 2. Modularidade

Por meio dos stored procedures, o sistema PostgreSQL consegue criar funções específicas e otimizar a guarda e o processamento de informações. Para tanto, é preciso conhecer as sintaxes de PL/pgSQL, PL/Tcl ou PL/perl. Esses módulos integrados aumentam o desempenho das tecnologias do seu negócio como um todo.

### 3. Escalabilidade

Os bancos de dados PostgreSQL são de fácil configuração e replicáveis em outros sistemas da sua empresa. Com isso, é viável aproveitar os

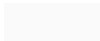
códigos-fonte dessas estruturas e otimizar o trabalho dos programadores e administradores de bases de dados.

Enfim, essas são as principais informações sobre os sistemas Postgres. São ferramentas que contribuem bastante para o trabalho colaborativo e segurança das informações da sua empresa. Vale lembrar também que as bases de dados PostgreSQL podem ajudar na adequação do seu negócio à Lei de Proteção de Dados Pessoais.

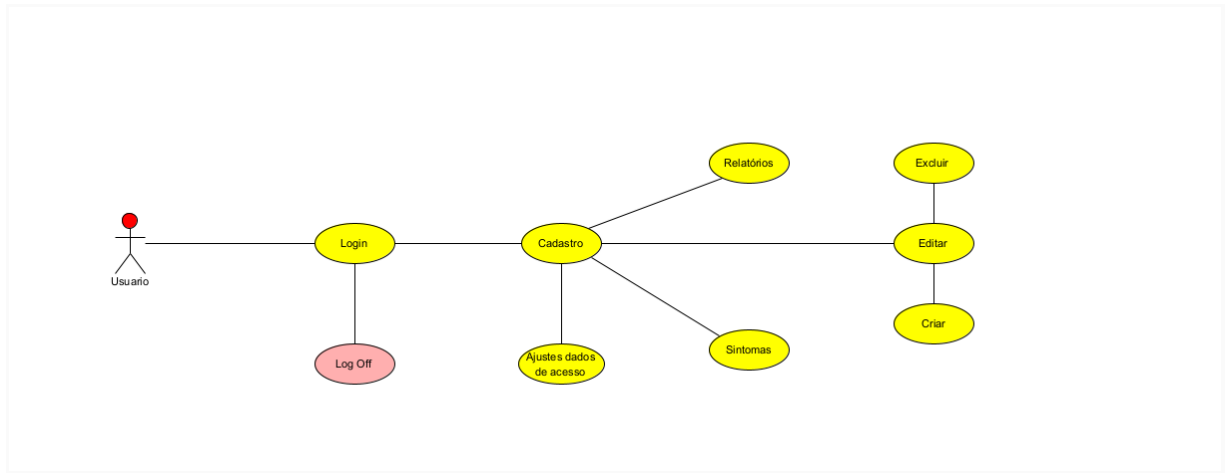
Muito bem, agora você já sabe como o PostgreSQL pode melhorar a produtividade do seu negócio. Se você gostou do texto, confira também este artigo sobre G Suite, que pode ajudar na segurança da informação da sua empresa.

## 6. Diagramas.

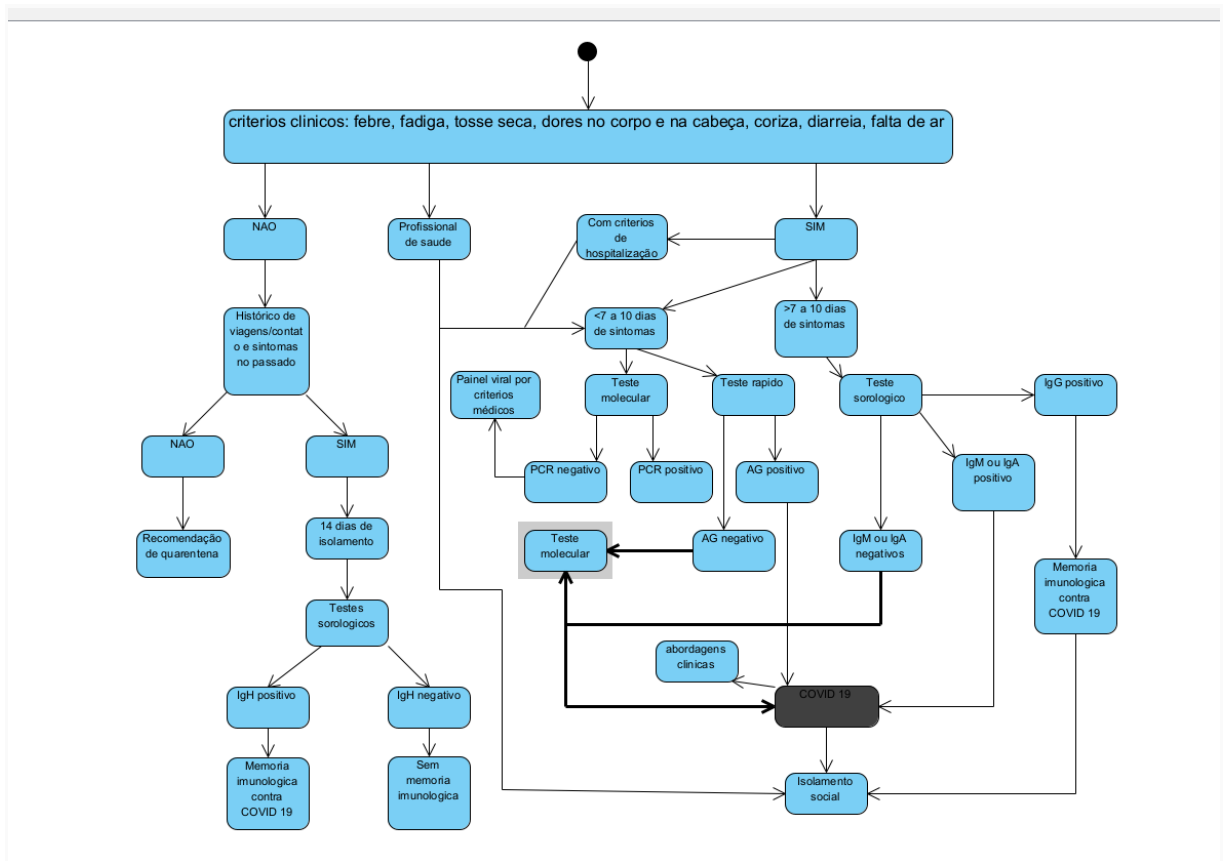
A UML – Unified Modeling Language (Linguagem de Modelagem Unificada), como o próprio nome já diz, é uma linguagem para modelagem de objetos do mundo real, usada para especificar, construir, visualizar e documentar um software. Em suma, uma modelagem UML oferece um “desenho” do software que se pretende desenvolver.



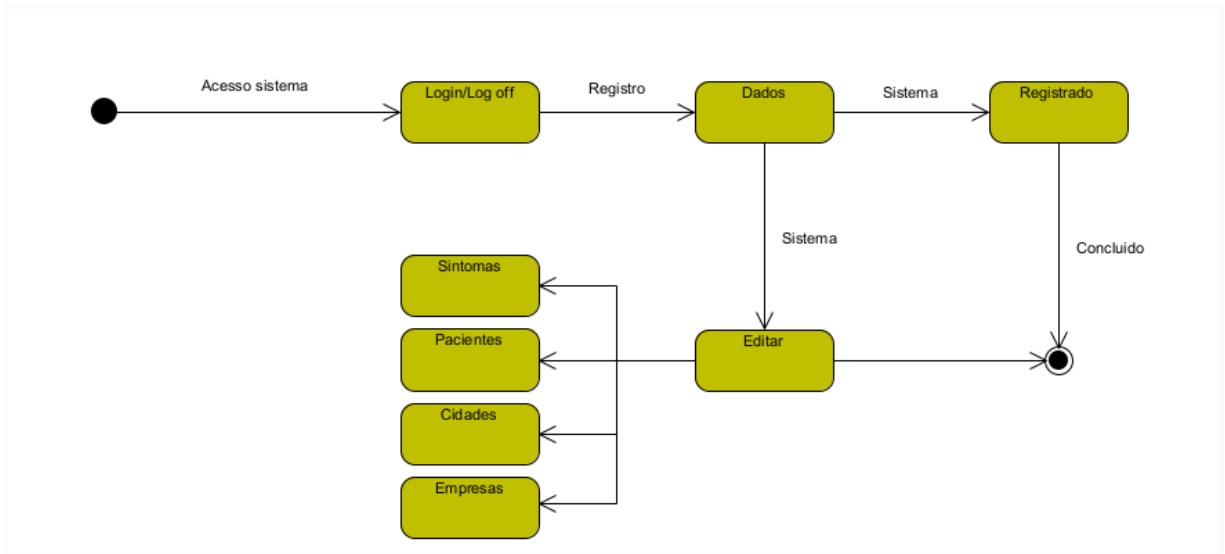
### Caso De Uso Administrador:



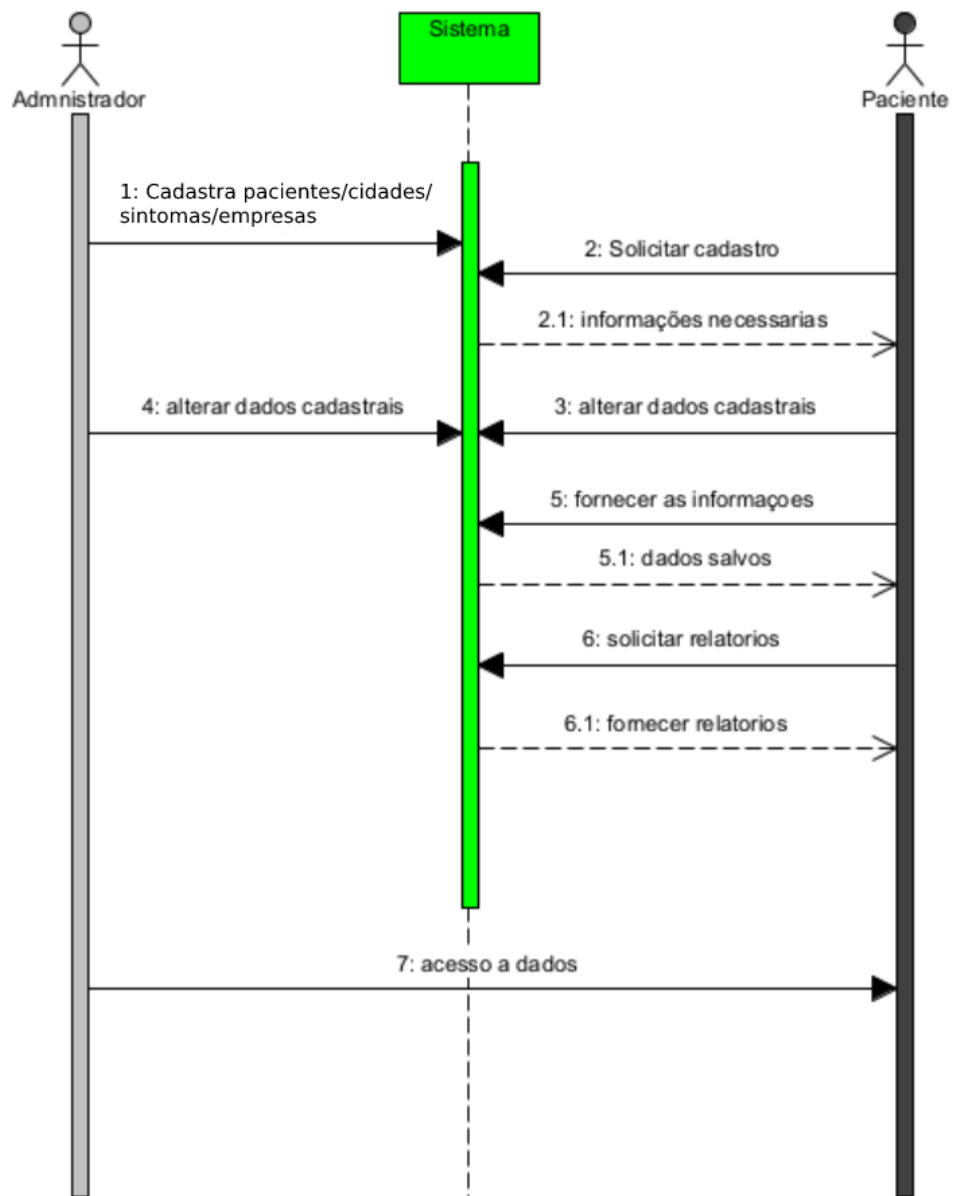
### Diagrama Atividade COVID-19:



## Diagrama De Estado Administrador:



## Diagrama De Sequência:



```
graph LR
    Admin[Administrador] --- UC1((Realizar cadastro de dados))
    UC1 -.->|<<include>>| UC2((Relatórios/cidades, pacientes, empresas, sintomas))
    UC3((Realizar edição de dados)) -.->|<<extend>>| UC1
    UC4((Login/Log Off)) -.->|<<extend>>| UC1
```

```

erDiagram
    tb_usuario ||--o{ tb_empresa : "trabalha em"
    tb_usuario ||--o{ tb_cidade : "reside em"
    tb_empresa ||--o{ tb_paciente : "empregado de"
    tb_cidade ||--o{ tb_paciente : "morador de"
    tb_paciente ||--o{ tb_sintoma : "apresenta"
    tb_paciente ||--o{ tb_questionario : "responde"
    tb_paciente ||--o{ tb_paciente_sintoma : "tem"
    tb_paciente_sintoma ||--o{ tb_sintoma : "sintoma"
    tb_paciente_sintoma ||--o{ tb_paciente : "paciente"
    tb_questionario ||--o{ tb_paciente : "paciente"
    
```

O diagrama de banco de dados relacional para o sistema de saúde apresenta as seguintes tabelas e atributos:

- tb\_usuario**:
  - id\_token** (integer(10))
  - tx\_token (varchar(255))
  - dt\_cadastro (timestamp)
  - id\_usuario (integer(10))
- tb\_empresa**:
  - id\_empresa** (integer(10))
  - tx\_cnpj (varchar(255))
  - tx\_nome (varchar(255))
  - dt\_abertura (timestamp)
  - tx\_fundador (varchar(255))
  - id\_cidade (integer(10))
- tb\_cidade**:
  - id\_cidade** (integer(10))
  - tx\_sigla (varchar(2))
  - tx\_nome (varchar(255))
  - tx\_estado (varchar(255))
- tb\_sintoma**:
  - id\_sintoma** (integer(10))
  - tx\_nome (varchar(255))
- tb\_paciente**:
  - id\_paciente** (integer(10))
  - tx\_nome (varchar(255))
  - dt\_nascimento (timestamp)
  - qt\_preso (double(10))
  - qt\_altura (double(10))
  - id\_cidade (integer(10))
  - id\_empresa (integer(10))
- tb\_paciente\_sintoma**:
  - cd\_paciente (integer(10))
  - cd\_sintoma (integer(10))
  - cd\_intensidade (integer(10))
- tb\_questionario**:
  - fl\_positivado (integer(10))
  - fl\_suspeito (integer(10))
  - fl\_obencansinistoria (integer(10))
  - cd\_paciente (integer(10))

As relações entre as tabelas são as seguintes:

- tb\_usuario** e **tb\_empresa**: Relação 1:N (um usuário pode trabalhar em uma ou mais empresas).
- tb\_usuario** e **tb\_cidade**: Relação 1:N (um usuário pode residir em uma ou mais cidades).
- tb\_empresa** e **tb\_paciente**: Relação 1:N (uma empresa pode empregar um ou mais pacientes).
- tb\_cidade** e **tb\_paciente**: Relação 1:N (uma cidade pode ter um ou mais moradores).
- tb\_paciente** e **tb\_sintoma**: Relação 1:N (um paciente pode apresentar um ou mais sintomas).
- tb\_paciente** e **tb\_questionario**: Relação 1:N (um paciente responde a um ou mais questionários).
- tb\_paciente** e **tb\_paciente\_sintoma**: Relação 1:N (um paciente tem um ou mais registros de sintomas).
- tb\_paciente\_sintoma** e **tb\_sintoma**: Relação 1:N (um sintoma é registrado em um ou mais registros de sintomas).
- tb\_paciente\_sintoma** e **tb\_paciente**: Relação 1:N (um registro de sintoma pertence a um ou mais pacientes).

## 7. Códigos/Scripts/SQL

```
app.component.spec.ts 9+ X
corona-peak > src > app > app.component.spec.ts > ...
Lucas Klauck, 2 weeks ago | 1 author (Lucas Klauck)
1 import { TestBed } from '@angular/core/testing';
2 import { RouterTestingModule } from '@angular/router/testing';
3 import { AppComponent } from './app.component';
4
5 describe('AppComponent', () => {
6   beforeEach(async () => {
7     await TestBed.configureTestingModule({
8       imports: [
9         RouterTestingModule
10      ],
11      declarations: [
12        AppComponent
13      ],
14    }).compileComponents();
15  });
16
17  it('should create the app', () => {
18    const fixture = TestBed.createComponent(AppComponent);
19    const app = fixture.componentInstance;
20    expect(app).toBeTruthy();
21  });
22
23  it(`should have as title 'projeto-shop'`, () => {
24    const fixture = TestBed.createComponent(AppComponent);
25    const app = fixture.componentInstance;
26    expect(app.title).toEqual('projeto-shop');
27  });
28
```



TS app.component.ts 2 X

corona-peak > src > app > TS app.component.ts > ...

Lucas Klauck, 15 hours ago | 1 author (Lucas Klauck)

```
1 import { Component, HostListener } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { AutenticacaoService } from '../service/autenticacao.service';
4 import { PacienteService } from '../service/paciente.service';
5 import { UsuarioService } from '../service/usuario.service';
6
```

Lucas Klauck, 15 hours ago | 1 author (Lucas Klauck)

```
7 @Component({
8   selector: 'app-root',
9   templateUrl: './app.component.html',
10  styleUrls: ['./app.component.scss'],
11 })
12 export class AppComponent {
13   readonly MOBILE = 500;
14
15   title = 'app-corona-peak';
16   mobile = false;
17   tipoAdministrador = false;
18   respondeuQuestionario = true;
19   estaLogado: boolean;
20
21   constructor(
22     private router: Router,
23     private autenticacaoService: AutenticacaoService,
24     private usuarioService: UsuarioService,
25     private pacienteService: PacienteService
26   ) {}
27
28   ngOnInit() {
29     this.pacienteService.respondeuQuestionario.subscribe(/
```

```
app.component.html X
corona-peak > src > app > app.component.html > div
Lucas Klauck, 15 hours ago | 1 author (Lucas Klauck)
1 <div *ngIf="estaLogado; else login"> Lucas Klauck, 2 weeks ago • Commit inicio de projeto
2
3   <p-toast position="top-center"></p-toast>
4
5   <mat-toolbar color="primary">
6     <button mat-icon-button class="botao-menu" (click)="menu.toggle()">
7       <mat-icon>menu</mat-icon>
8     </button>
9     <span>Corona Peak</span>
10  </mat-toolbar>
11
12  <mat-sidenav-container>
13    <mat-sidenav #menu [mode]="!mobile ? 'side' : 'over'" [opened]="!mobile" class="mat-elevation-z8">
14      <div class="menu">
15
16        <button mat-raised-button class="menu-button" matRipple routerLink="/" (click)="menu.close()">
17          <mat-icon>home
18          </mat-icon>
19          <span>Inicio</span>
20        </button>
21
22        <mat-divider></mat-divider>
23
24        <div *ngIf="tipoAdministrador">
25          <button mat-raised-button class="menu-button" matRipple routerLink="cadastro/">
26            <mat-icon>add_box
27            </mat-icon>
28            <span>Cadastro</span>
29          </button>
30
```

login.component.spec.ts 6 X

corona-peak > src > app > components > autenticacao > login > login.component.spec.ts > ...

Lucas Klauck, 2 weeks ago | 1 author (Lucas Klauck)

```
1 import { ComponentFixture, TestBed } from '@angular/core/testing';
2
3 import { LoginComponent } from './login.component';
4
5 describe('LoginComponent', () => {
6   let component: LoginComponent;
7   let fixture: ComponentFixture<LoginComponent>;
8
9   beforeEach(async () => {
10     await TestBed.configureTestingModule({
11       declarations: [ LoginComponent ]
12     })
13     .compileComponents();
14   });
15
16   beforeEach(() => {
17     fixture = TestBed.createComponent(LoginComponent);
18     component = fixture.componentInstance;
19     fixture.detectChanges();
20   });
21
22   it('should create', () => {
23     expect(component).toBeTruthy();
24   });
25 });
26
```

```
corona-peak > src > app > components > pagina-inicial > TS pagina-inicial.component.ts > ...
Lucas Klauck, 15 hours ago | 1 author (Lucas Klauck)
1 import { Component, OnInit } from '@angular/core';
2 import { DadosRelatorioDTO } from '../../../models/dados-relatorio-dto';
3 import { RelatorioService } from '../../../service/relatorio.service';
4 import { ToastService } from '../../../service/toast-service.service';
5
6 @Component({
7   selector: 'app-pagina-inicial',
8   templateUrl: './pagina-inicial.component.html',
9   styleUrls: ['./pagina-inicial.component.scss'],
10 })
11 export class PaginaInicialComponent implements OnInit {
12   dadosRelatorio = new DadosRelatorioDTO();
13
14   constructor(
15     private relatorioService: RelatorioService,
16     private toastService: ToastService
17   ) {}
18
19   ngOnInit(): void {
20     this.relatorioService.adquirirTodos().subscribe(
21       (dadosRelatorio) => (this.dadosRelatorio = dadosRelatorio),
22       () => this.toastService.addError('Erro ao adquirir relatórios!')
23     );
24   }
25 }
26
```

## 7. REFERÊNCIAS.

### GITHUB:

<https://github.com/lucasklauck2/coronapeak-api>

<https://github.com/lucasklauck2/corona-peak>

### PESQUISA:

<https://www.devmedia.com.br/primeiros-passos-com-o-angular-material/40409>

<https://digitalinnovation.one/artigos/como-usar-o-lombok-em-projetos-java>

<https://rockcontent.com/br/blog/banco-de-dados/>

<https://blog.algaworks.com/spring-boot/>

