**Projeto SP Medical Group**

Documentação

Sumário

[1. Resumo 3](#_Toc71092221)

[2. Descrição do projeto 3](#_Toc71092222)

[3. Modelagem de Dados 3](#_Toc71092223)

[Modelo Conceitual 3](#_Toc71092224)

[Modelo Lógico 3](#_Toc71092225)

[Modelo Físico 3](#_Toc71092226)

[Cronograma 3](#_Toc71092227)

[Trello 4](#_Toc71092228)

[4. Back-End 4](#_Toc71092229)

[Funcionalidades 5](#_Toc71092230)

[Sistema Web 5](#_Toc71092231)

[Perfis de usuário: 5](#_Toc71092232)

[Funcionalidades: 5](#_Toc71092233)

[Sistema Mobile 5](#_Toc71092234)

[Perfis de usuário: 5](#_Toc71092235)

[Funcionalidades: 5](#_Toc71092236)

# Resumo

Documentação onde será mostrado informações sobre o projeto, como tecnologias, diagramas, funcionalidades, etc.

# Descrição do projeto

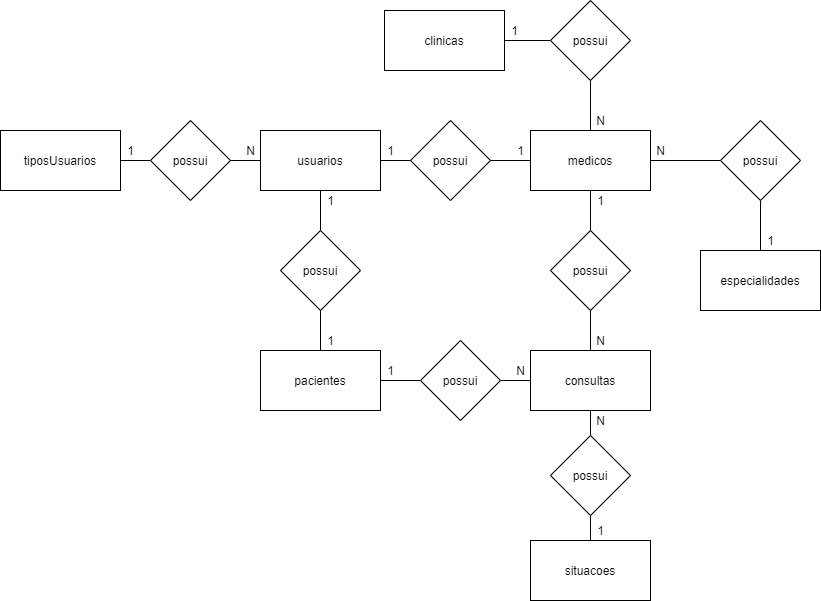
Sistema para gerenciar um hospital, onde é possível cadastrar pacientes, médicos, consultas, e clínicas. Um médico terá todas as funcionalidades necessárias para ver suas consultas e adicionar descrição a elas, já o paciente pode ver suas consultas, um administrador pode mudar o status das consultas além de poder modificar os dados dela, de um médico e de um paciente.

# Modelagem de Dados

A modelagem de dados em sistema relacional, é quando criamos as tabelas e suas ligações e representamos graficamente, em diagramas, que podem ser exportadas em imagens e até em planilhas no caso de um modelo físico.

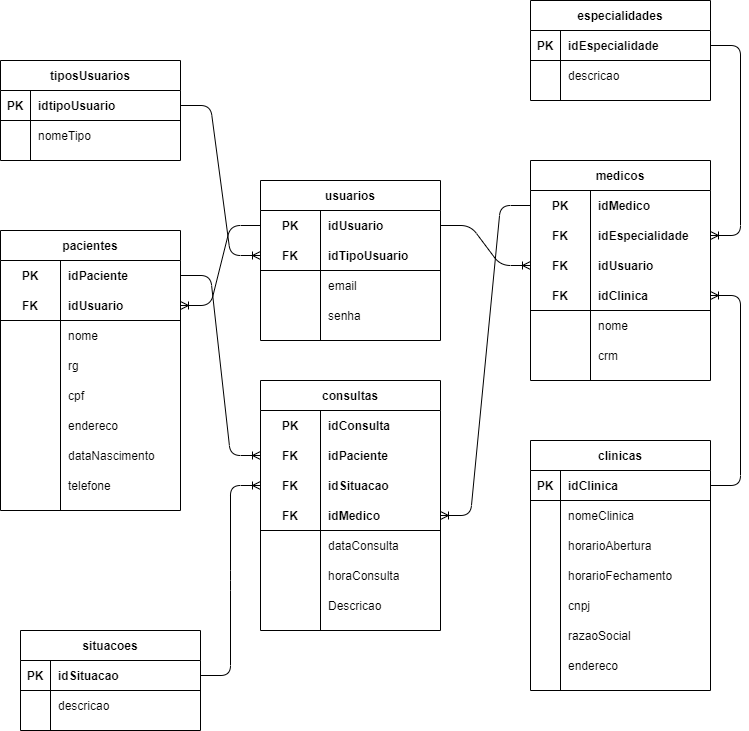
## Modelo Conceitual

O modelo conceitual é um diagrama mostrando entidades e relacionamentos, é mais simples e de fácil entendimento.



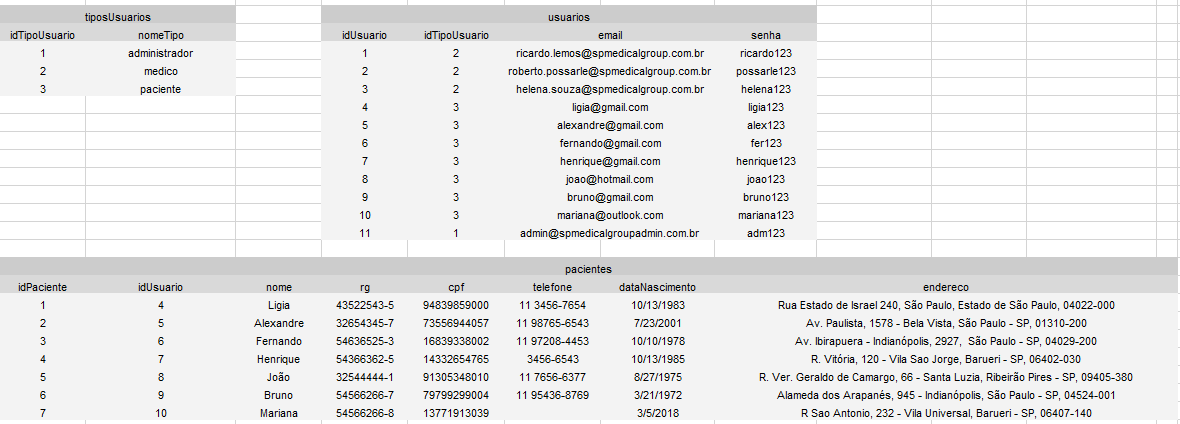
## Modelo Lógico

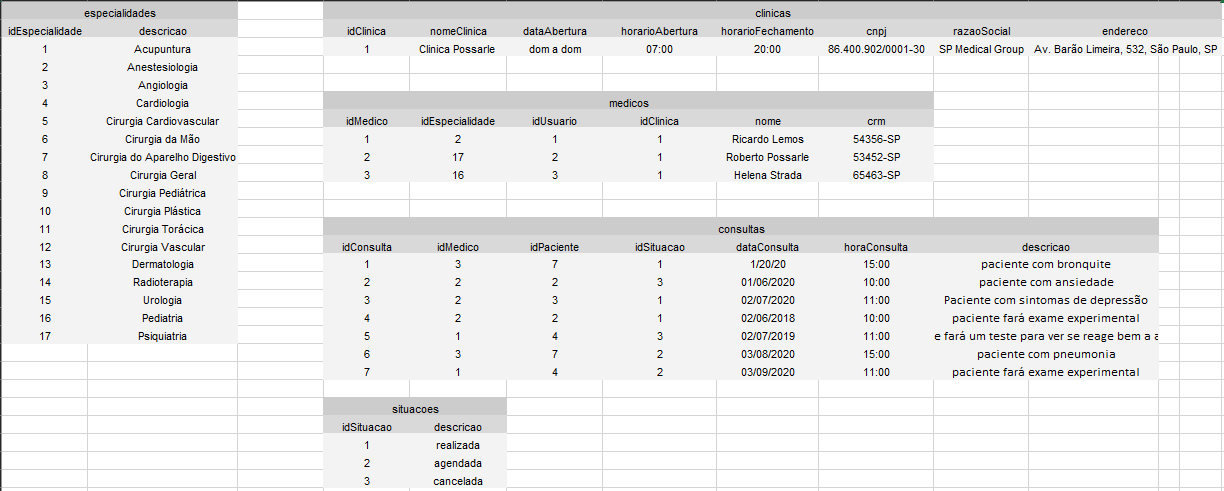
O modelo lógico é um modelo mais completo do diagrama, onde serão mostrados não apenas as entidades e relacionamentos, mas também as colunas/atributos de cada tabela, como primary keys e foreign keys.



## Modelo Físico

O modelo físico é onde tem os dados para a implementação inicial ao banco de dados.





## Cronograma

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Dia 1 | Dia 2 | | Dia 3 | Dia 4 | | Dia 5 |
| Modelo Conceitual | X |  | |  |  | |  |
| Modelo Lógico | X |  | |  |  | |  |
| Modelo Físico | X |  | |  |  | |  |
| Web API | X | X | | X | X | | X |
| Postman |  | |  |  | |  | X |
| Documentação |  | |  |  | |  | X |

### Trello

https://trello.com/b/LbRtGwxR/projeto-sp-medical-group

# Back-End

O código do sistema foi desenvolvido através de uma API utilizando o Microsoft Visual Studio como IDE. A linguagem de programação utilizada foi o C# e o framework para desenvolvimento web utilizado foi o Asp.Net Core. Foi utilizado o Postman para fazer requisições à API.

**API** é uma sigla para Application Programming Interface, ou Interface de Programação de Aplicação. Ela é o nosso back-end e roda por trás de tudo, essa API segue os protocolos HTTP, e ela recebe e response em formato JSON, ou seja, pode se comunicar com diversos sistemas independente da linguagem que estes utilizam.

Além disso, foi utilizado o estilo de arquitetura **REST**. Que segue alguns padrões como, nomear recursos de forma apropriada, utilizar os verbos corretos para cada recurso, utilizar status code, etc. Quando uma API segue todas essas regras, dizemos que ela é uma API Restful.

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

**JSON** – JavaScript Object Notation – Notação de Objetos JavaScript.

**REST** – Representational State Transfer – Transferência de estado representacional.

Como executar a API e fazer requisições.

### Pré requisitos:

Ter o DotNet Run Time e o Postman instalados.

Tutorial instalação DotNet SDK: <https://github.com/senai-desenvolvimento/2021-1S-2D/tree/main/apoio/tutoriais/05.dotnet%20sdk>

Instalação Postman, apenas baixar por este link e instalar:

<https://www.postman.com/downloads/>

### Como executar a API e utilizar:

Para executar a API, abrir a pasta da API onde tem uma pasta chamada program.cs, clicar no caminho da pasta no explorador de arquivos e digitar cmd e apertar enter, quando abrir o cmd, executar o **dotnet run.**

Quando a API subir ela vai abrir o navegador em uma página do Swagger com todos os endpoints e uma descrição do que cada um faz, além de ter como fazer requisições, mas para as requisições a ferramenta Postman será melhor.

Abra o Postman, localize a opção import, e escolha a pasta do Postman que está na pasta do projeto, com isso, a pasta irá aparecer no canto esquerdo, dentro dela terá outras subpastas e cada uma terá as requisições para testá-las. Atente-se para cada endpoints, pois a maioria tem restrições, alguns somente administrador pode fazer requisições, e outros somente médicos e pacientes. Por isso vai ser necessário fazer o Login no endpoint de login, você pode encontrar a requisição de login na pasta do Postman do projeto.

## Funcionalidades

Insira as funcionalidades que a API atende.

Por exemplo:

### Perfis de usuário:

1. **Administrador**: Para o colaborador da área administrativa da clínica;

2. **Médico**: Colaboradores que atuam na área da saúde;

3. **Paciente**: Clientes da clínica;

### Funcionalidades:

1. O **administrador** poderá cadastrar qualquer tipo de usuário (administrador,

paciente ou médico);

2. O **administrador** poderá agendar uma consulta, onde será informado o paciente,

data do agendamento e qual médico irá atender a consulta (o médico possuirá

sua determinada especialidade);

3. O **administrador** poderá cancelar o agendamento;

4. O **administrador** deverá informar os dados da clínica (como endereço, horário

de funcionamento, CNPJ, nome fantasia e razão social);

5. O **médico** poderá ver os agendamentos (consultas) associados a ele;

6. O **médico** poderá incluir a descrição da consulta que estará vinculada ao paciente

(prontuário);

7. O **paciente** poderá visualizar suas próprias consultas;