**SP MEDICAL GROUP**

Documentação

Sumário

[1. Resumo 3](#_Toc80311768)

[2. Descrição do projeto 3](#_Toc80311769)

[3. Banco de dados relacional 3](#_Toc80311770)

[4. Modelagem de dados 3](#_Toc80311771)

[Modelo Conceitual 3,4](#_Toc80311772)

[Modelo Lógico](#_Toc80311773) 4,5

[Modelo Físico 5](#_Toc80311774)

[Cronograma 5](#_Toc80311775),6

[Trello 6](#_Toc80311776)

# Resumo

Fernando Strada lhe contratou para desenvolver um sistema web/mobile integrado

onde seja possível realizar a gestão da clínica de forma automatizada e ter acesso fácil

aos dados sobre as informações de seus pacientes.

# Descrição do projeto

Uma nova clínica médica chamada SP Medical Group, empresa de pequeno porte que

atua no ramo da saúde, foi criada pelo médico Fernando Strada em 2020 na região da

Paulista em São Paulo. Fernando tem uma equipe de médicos que atuam em diversas

áreas (pediatria, odontologia, gastrenterologia etc.). A clínica SP Medical Group contratou você para desenvolver sua solução em software.

# Banco de dados relacional

Um banco de dados relacional é um banco de dados que possui tabelas que possamos

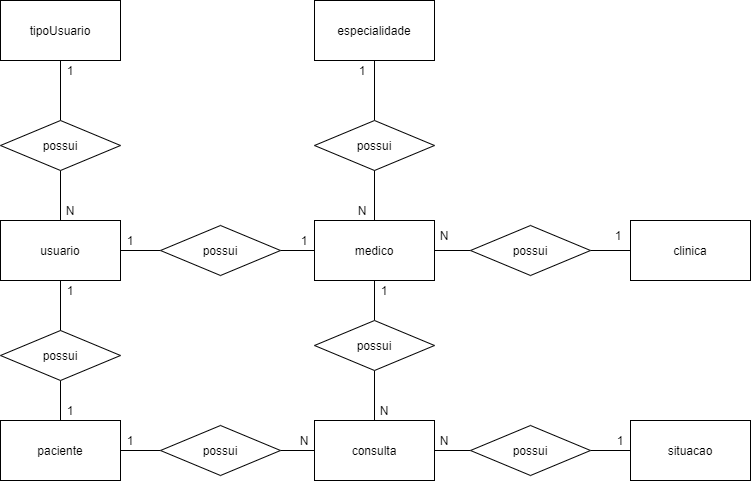
visualizar que possuem relações uma entre a outra, normalmente conectadas por id´s.

# Modelagem de dados

A Modelagem de dados tem o intuito de deixar o planejamento da criação do banco de dados mais visual e claro, nela nós vemos quais são as entidades, suas relações e até mesmo as futuras colunas das tabelas.

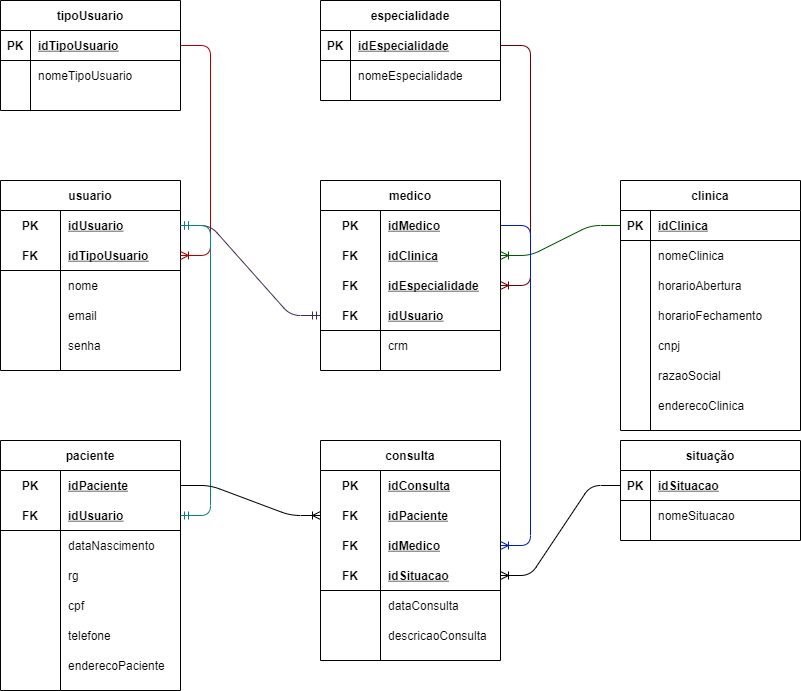
## Modelo Conceitual

Neste modelo conceitual do banco de dados do projeto SP Medical Groups estão todas as entidades necessárias e suas respectivas relações uma entre a outra.



## Modelo Lógico

Neste modelo lógico do banco de dados do projeto SP Medical Groups estão todas as entidades necessárias e suas respectivas relações uma entre a outra.



## Modelo Físico

No modelo físico, já no sql server, é possível observar as tabelas e suas colunas já com informações.

(Print SSMS)

## Cronograma

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 |
| Modelo Conceitual | X |  |  |  |
| Modelo Lógico | X |  |  |  |
| Modelo Físico |  | X |  |  |
| Script  DDL |  | X |  |  |
| Script  DML |  | X |  |  |
| Script  DQL |  | X |  |  |
| Scaffold |  | X |  |  |
| Domains |  | X |  |  |
| Interfaces |  | X |  |  |
| Repositories |  |  | X |  |
| Controllers |  |  | X |  |
| Postman |  |  | X |  |

### Trello

<https://trello.com/b/bPVO6LT3/sp-medical-group>

# Back-End

O código do sistema foi desenvolvido através de uma API utilizando o Microsoft Visual Studio, sendo utilizada a linguagem de programação C#. Para a criação da documentação foi utilizado o Swagger. Para a proteção e criação de tokens foi utilizado o JWT (JSON Web Tokens).

**API** é um conjunto de padrões e instruções estabelecidos para utilização do software,

definindo as requisições e as respostas seguindo o protocolo HTTP, neste caso

expresso no formato JSON, para que seja possível acessar o sistema em diversos

dispositivos distintos sem a preocupação com a linguagem que será utilizada por estes.

Além disso, foi utilizado o estilo de arquitetura **REST**.

**Swagger** é uma linguagem de descrição de interface para descrever **APIs RESTful** expressas usando JSON. O Swagger é usado junto com um conjunto de ferramentas de software de código aberto para projetar, construir, documentar e usar serviços da Web **RESTful**.

O **JSON Web Token** é um padrão da Internet para a criação de dados com assinatura opcional e/ou criptografia cujo payload contém o JSON que afirma algum número de declarações. Os tokens são assinados usando um segredo privado ou uma chave pública/privada.

**API** – Application Programming Interface – Interface de Programação de Aplicativos.

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

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

**REST** – Representational State Transfer – Interface de Programação de Aplicativos.

**JWT** – JSON Web Tokens - O JSON Web Token é um padrão da Internet para a criação de dados com assinatura opcional e/ou criptografia cujo payload contém o JSON que afirma algum número de declarações.

## Passo a passo Postman

1 – Utilizando do Github e do Git, faça a clonagem do repositório;

2 – Abra a solução utilizando o Visual Studio;

3 – Rodando a solução em sua máquina, execute-a;

4 – Após a execução, será automaticamente aberto o Swagger, ele auxiliara-lo com os nomes dos **controllers** e com os caminhos que será necessários no Postman;

5 – Com a API em execução, entre no Postman e crie uma request/requisição;

6 – Na request, coloque o caminho : “[http://localhost:5000/api/login”](http://localhost:5000/api/consultas/minhas/paciente%20)  com o termo “POST”,

clique em “body”,”raw” e escolha a opção “JSON”, e então insira:

{

    "Email" : "adm@gmail.com",

    "Senha" : "adm123"

}

7 – Copie o token dado e crie outra request;

8 – Nesta nova request, insira o caminho : “http://localhost:5000/api/pacientes” com o termo “GET”, o resultado será a lista com todos os pacientes.

## Funcionalidades

Insira as funcionalidades que a API atende.

## Sistema Web

### Perfis de usuário:

1. Administrador: Área administrativa da escola;

2. Comum: Pode ser um espectador ou palestrante.

### Funcionalidades:

1. O administrador poderá cadastrar qualquer tipo de usuário;

2. O administrador poderá cadastrar os dados da instituição;

3. O administrador poderá cadastrar os tipos de eventos;

4. O administrador poderá cadastrar eventos livres ou restritos;

5. Qualquer usuário autenticado poderá ver todos os eventos cadastrados;

6. O usuário comum poderá ver os eventos que participará;

7. O usuário comum poderá se inscrever para assistir a um evento;

8. O administrador poderá aprovar a participação do espectador de um

determinado evento restrito;

## Sistema Mobile

### Perfis de usuário:

1. Administrador: Área administrativa da escola;

2. Comum: Pode ser um espectador ou palestrante;

Funcionalidades:

1. Qualquer usuário autenticado poderá ver todos os eventos cadastrados;

2. O usuário comum poderá se inscrever para assistir um evento;

3. O usuário comum poderá ver os eventos que participará;