**NOMEPROJETO**

Documentação

Sumário

[1. Resumo 3](#_Toc71092221)

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

3 Banco de Dados relacional................................................................................................3

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

[Modelo Conceitual 3](#_Toc71092224)

[Modelo Lógico 4](#_Toc71092225)

[Modelo Físico 4](#_Toc71092226)

[Cronograma 5](#_Toc71092227)

[Trello .5](#_Toc71092228)

[4. Back-End 6](#_Toc71092229)

Cronograma......................................................................................................................6

[Funcionalidades 6](#_Toc71092230)

[Sistema Web 8](#_Toc71092231)

[Perfis de usuário: 8](#_Toc71092232)

[Funcionalidades: 8](#_Toc71092233)

[Sistema Mobile 8](#_Toc71092234)

[Perfis de usuário: 8](#_Toc71092235)

[Funcionalidades: 8](#_Toc71092236)

# Resumo

Projeto criado afim de auxiliar 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.).

# Descrição do projeto

Será criado um Sistema que ajudará a clínica a cadastrar e organizar os dados dos pacientes, médicos, consultas, etc.

# Banco de dados relacional

Um banco de dados relacional é um tipo de banco onde pontos de dados se relacionam entre si, através do modelo de relação é possível criar e representar bancos de dados de forma intuitiva e direta. Em um banco como esse cada registro da tabela precisa de um ID, as famosas chaves, sendo primária ou estrangeira. Graças a esse modelo se tem uma padronização de representar diversos dados de forma eficiente e de fácil acesso.

# Modelagem de dados

A modelagem de dados é a parte de desenvolvimento de sistemas que se encarrega de construir estruturas que permitam armazenar e recuperar informações em contextos específicos.

## Modelo Conceitual

## Diagrama Descrição gerada automaticamente

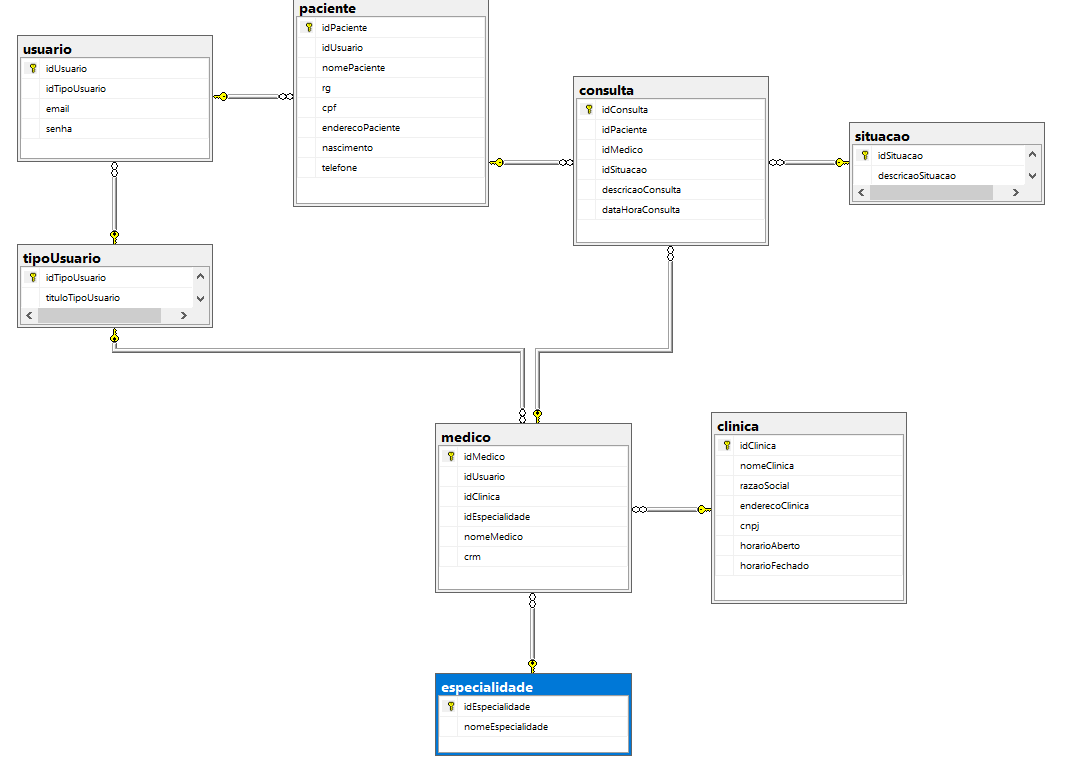
Modelagem Conceitual: Modelo mais simples de identificar as entidades e suas relações entre as tabelas, exemplo: 1:1, 1:N, N:1, N:N

## Modelo Lógico

## Diagrama Descrição gerada automaticamente

Modelagem lógico: Modelo mais complexo para identificar chaves Primárias e Estrangeiras e saber seus ID e atributos

## Modelo Físico





Modelo Físico: Modelo que exemplifica com os dados em seus respectivos lugares da tabela, simulando um banco de dados

## Cronograma

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 | Dia 5 |
| Modelo Conceitual | X |  |  |  |  |
| Modelo Lógico | X |  |  |  |  |
| Modelo Físico | X |  |  |  |  |
| Scripts: DDL,DML e DQL |  | X |  |  |  |
| Documentação |  | X |  |  |  |

### Trello

[SP MEDICAL GROUP | Trello](https://trello.com/b/5053MZ5C/sp-medical-group)

# Back-End

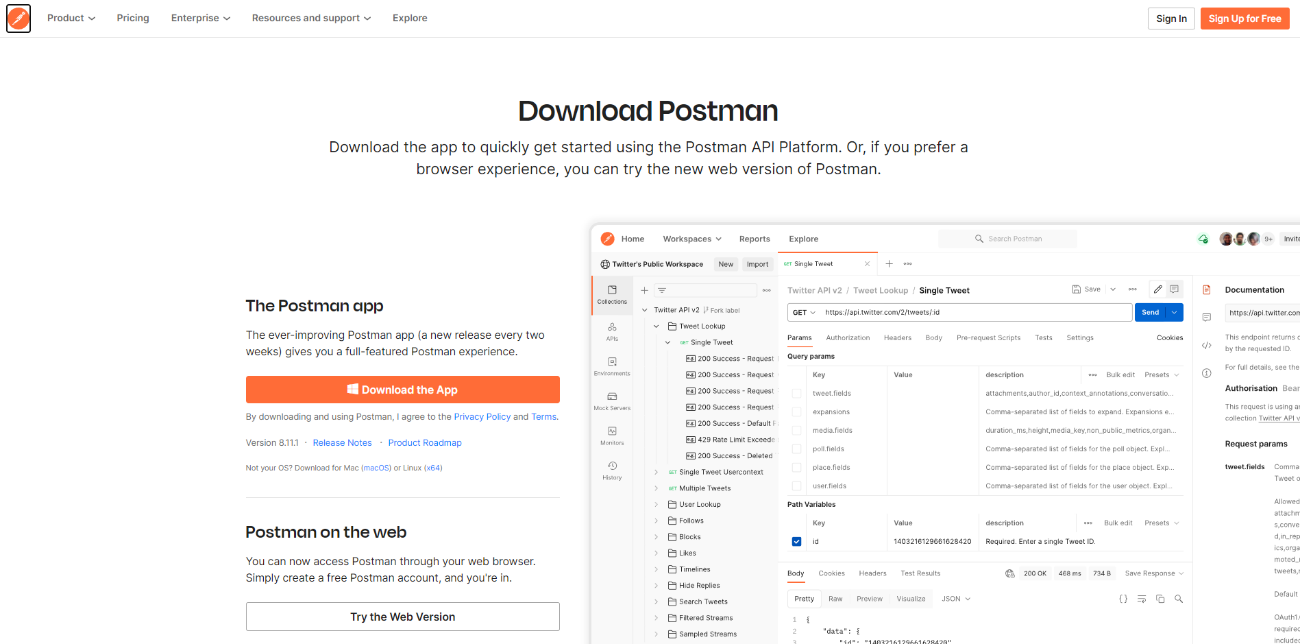
Descrição sobre a arquitetura utilizada para a criação do back-end.

A API (Application Programming Interface) foi desenvolvida através do Visual Studio 2019 Community, famosa IDE da Microsoft que possibilita o desenvolvimento web.

A linguagem utilizada para criação do projeto foi o C#, uma linguagem desenvolvida pela própria Microsoft como parte da plataforma .NET. Como já dito a arquitetura utilizada foi a API, sendo um modelo prático e mais fácil de ser compreendido.

Para as requisições foi utilizado o protocolo HTTP (Hypertext Transfer Protocol), em formato JSON (JavaScript Object Notation), dessa forma é possível transferir dados de forma leve e compacta, não sendo necessário pegar todos os dados disponível para carregar páginas.

Passo a passo de como executar a API criada e testar as requisições no Postman:

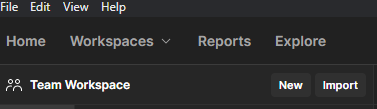


Primeiro Passo: Primeiramente instale a versão desktop do Postman, onde você irá testar as requisições.

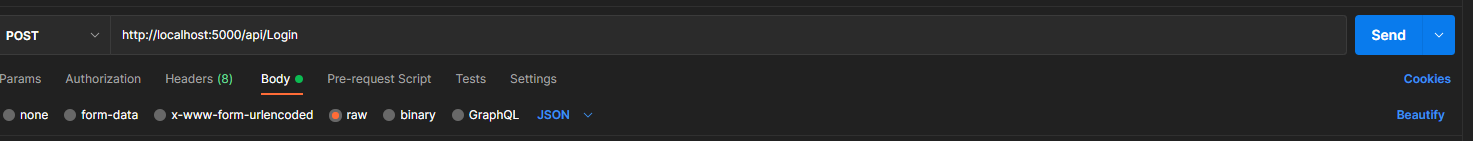
Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Segundo Passo: Logo após o download, use o comando git clone para clonar o repositório, conseguindo assim usar a coleção do Postman.



Terceiro passo: Depois disso clique no botão Import e faça o upload da coleção, que trará junto dela as pastas com as requisições.



Quarto passo: Por fim cliquei no botão send e teste todas as requisições.

## Cronograma

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 | Dia 5 |
| Domains | X |  |  |  |  |
| Interfaces | X |  |  |  |  |
| Repositories |  | X |  |  |  |
| Controllers |  |  |  | X |  |
| ViewModel |  |  | X |  |  |
| Swagger e Startup | X |  |  |  |  |
| Postman |  |  |  |  | X |

## Sistema Web

### Perfis de usuário:

1. **Administrador**: Área administrativa da clínica;
2. **Paciente**: Clientes da clínica;
3. **Médico**: Colaboradores que atuam na área da saúde;

### Funcionalidades:

1. O **administrador** poderá cadastrar qualquer tipo de usuário, sendo outro administrador, paciente ou médico;
2. O **administrador** poderá cadastrar os dados de uma consulta, como por exemplo a data do agendamento, médico, etc;
3. O **administrador** poderá alterar a situação da consulta, podendo cancelar ela, além disso tem a permissão para mudar a consulta, sendo realizada ou apenas agendada;
4. O **administrador** poderá cadastrar os dados das clínicas, como por exemplo endereço, nome, CNPJ, etc;
5. Qualquer usuário autenticado poderá ver todas as consultas relacionados a ele, sendo paciente ou médico;
6. O médico poderá inserir a descrição da consulta que está associada a um paciente;

## Sistema Mobile

### Perfis de usuário:

1. **Administrador**: Área administrativa da clínica;
2. **Paciente**: Clientes da clínica;
3. **Médico**: Colaboradores que atuam na área da saúde;

### Funcionalidades:

1. Qualquer usuário autenticado poderá ver todos as clínicas;
2. Somente o **administrador** poderá agendar a consulta para o paciente;
3. O usuário **paciente** poderá ver as consultas associados a ele;
4. O usuário **médico** poderá ver as consultas associados a ele;