

# Documentação

## Sumário

1.	Resumo.....	3
2.	Descrição do projeto .....	3
3.	Banco de dados relacional .....	3
4.	Modelagem de dados.....	3
	Modelo Conceitual .....	3
	Modelo Lógico.....	4
	Modelo Físico .....	5
	Cronograma.....	5
	Trello .....	6
5.	Back-End.....	7
	Funcionalidades.....	8
	Sistema Web.....	5
	Perfis de usuário:.....	5
	Funcionalidades:.....	5
	Sistema Mobile :.....	14
	Perfis de usuário:.....	14
	Funcionalidades:.....	14
6.	Front-End.....	15
	Tecnologias.....	16
	Layout.....	17
7.	Mobile.....	18
	Tecnologias.....	18
	Layout.....	20
8.	Banco NoSql.....	22

## 1. Resumo

O documento a seguir é um informativo sobre a modelagem do banco de dados do gerenciador de consultas da SP Medical Group, onde brevemente descrevemos o projeto em si, o significado de banco de dados relacional, como funciona a modelagem, os 3 tipos de modelagem (Conceitual, Lógico e Físico) e organização dessa parte inicial do projeto.

## 2. Descrição do projeto

SP Medical Group é uma nova clínica criada por Fernando Strada no ano de 2020 para atuação no ramo da saúde. Por conta do sucesso da clínica, Fernando solicitou que fosse criado um sistema Web/Mobile para realização da gestão da clínica, de forma automatizada, e de fácil acesso aos dados das informações dos pacientes atendidos e dos médicos que trabalham na clínica.

## 3. Banco de dados relacional

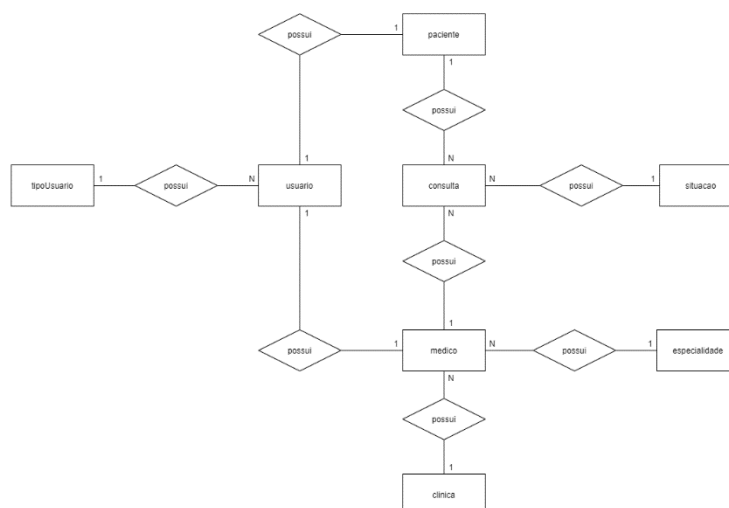
O banco de dados relacional é um banco de dados cuja a estrutura é modelada na forma de “tabelas”, que tenham associações/relações umas com as outras, esse tipo de banco é importante pois os dados são armazenados de forma mais organizada, segura e de relativamente fácil entendimento.

## 4. Modelagem de dados

A modelagem é o primeiro passo para a criação de um banco de dados relacional, a partir desse sistema visual, definimos as entidades, as relações que cada uma terá e a cardinalidade, facilitando a criação dos scripts, uma vez que já temos uma base de como irá funcionar, a modelagem está dividida em 3 tipos: conceitual, lógica e física, que serão explicadas a seguir.

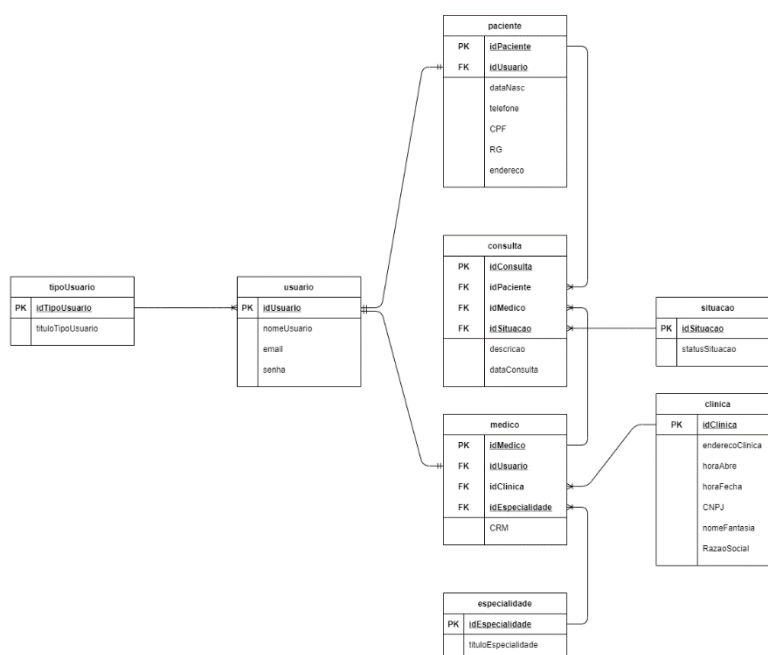
### Modelo Conceitual

Essa é uma modelagem mais simples, que informa as entidades do banco e a forma com a qual estão envolvidas, então usamos as cardinalidades de (1:1), (1:N), (N:1), (N:N) para relacionar cada tabela/entidade, dessa forma temos um visual mais “básico” de como o banco está armazenando seus respectivos dados.



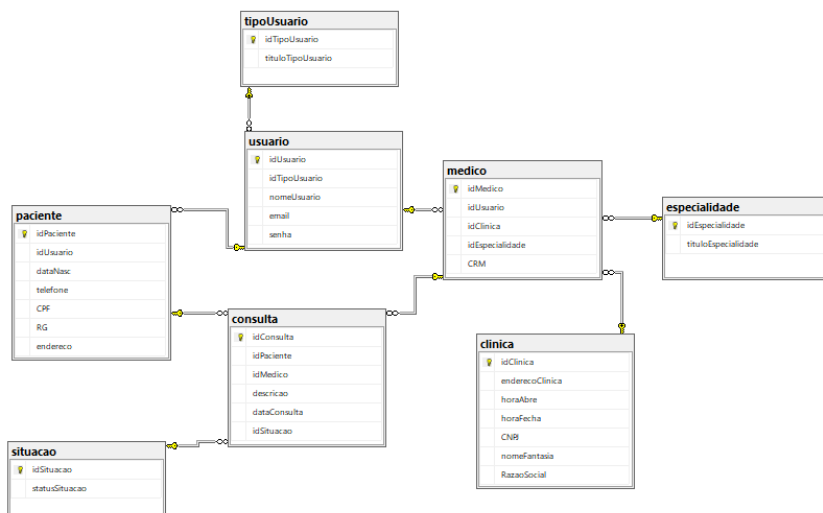
## Modelo Lógico

O modelo lógico é mais detalhado, com informação dos campos que estarão em cada entidade/tabela, assim como quais serão as chaves primárias (PK), que definem a tabela, e as estrangeiras (FK), que são as chaves que vem de relações com outras tabelas, conectando as e mostrando suas relações (com as respectivas cardinalidades).



## Modelo Físico

Essa modelagem representa como as tabelas ficarão no banco de dados de uma maneira mais detalhada e clara, ainda pode ser testada/validada em modelos do excel antes de os dados serem realmente colocados no sistema por exemplo, abaixo temos a imagem de como é a tabela gerada pelo próprio SSMS.



## Cronograma

	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6
Modelo Conceitual	X					
Modelo lógico	X					
Modelo físico	X					
DDL	X					
DML		X				
DQL		X				
Configuração da API			X			
Domínios			X			
Interfaces e Repositórios				X	X	
Controladores					X	X
Hospedagem						X

Trello

<https://trello.com/b/b53OAdGE/sp-medgroup>

\_\_\_\_ SENAI . SP

TÉCNICO EM DESENVOLVIMENTO

## 1. Back-End

O sistema back-end do serviço da SP-Medical-Group foi desenvolvido por meio de uma API, usando a linguagem de programação C#, utilizando a ferramenta Microsoft Visual Studio.

API (Application Programmin Interface, ou Interface de Programação de Aplicativos) é onde acontece a magia da programação, por meio de um conjunto de padrões e instruções estabelecidos, e definindo as requisições (como fazer login, cadastrar, listar, atualizar e deletar os dados) e as respostas seguindo o protocolo HTTP (em específico nessa API, no recebendo no formato JSON), para que os usuários possam acessar essa aplicação e usufruir tudo que ela permite.

Foi utilizado o estilo de arquitetura **REST**, e o padrão de design Repository Pattern, com uso de domínios, interfaces, repositórios e controladores para se ter organização na API, também foi utilizado a **ORM EntityFramework Core** para se ter conexão com o banco de dados

A hospedagem foi feita por meio dos serviços da Microsoft Azure.

Também foi utilizado o software Postman para fazer requisições à aplicação (como fazer login, cadastrar, listar, atualizar e deletar os dados).

**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.

**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.

**ORM** – Object-Relational Mapping (Mapeamento Objeto Relacional) são um conjunto de classes que facilitam a conexão com o banco de dados (sem necessidade de códigos exagerados).

**EntityFramework** – O ORM que permite a conexão com o banco pelos usuários do .NET.

**.NET** – Plataforma para desenvolvimento de códigos.

**Domínios** – Representa as entidades do banco de dados.

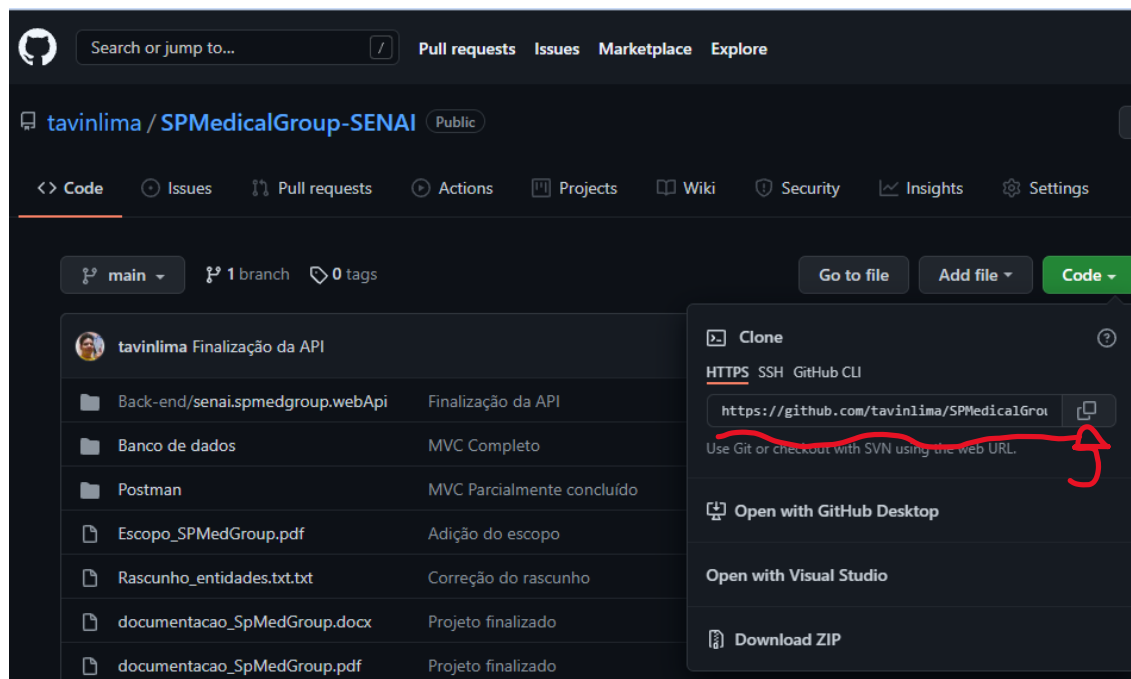
**Interfaces** – Define como vai ser feito.

**Repositórios** – Define o que vai ser feito.

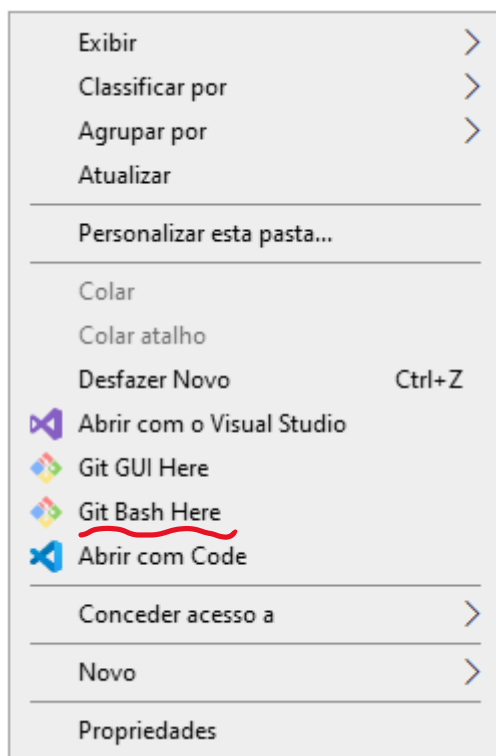
**Controladores** – Controla o fluxo de dados.

**Passo a passo para fazer uma requisição na API, por meio do software Postman:**

**1º passo** - Copie o link em destaque de dentro do repositório do GitHub:

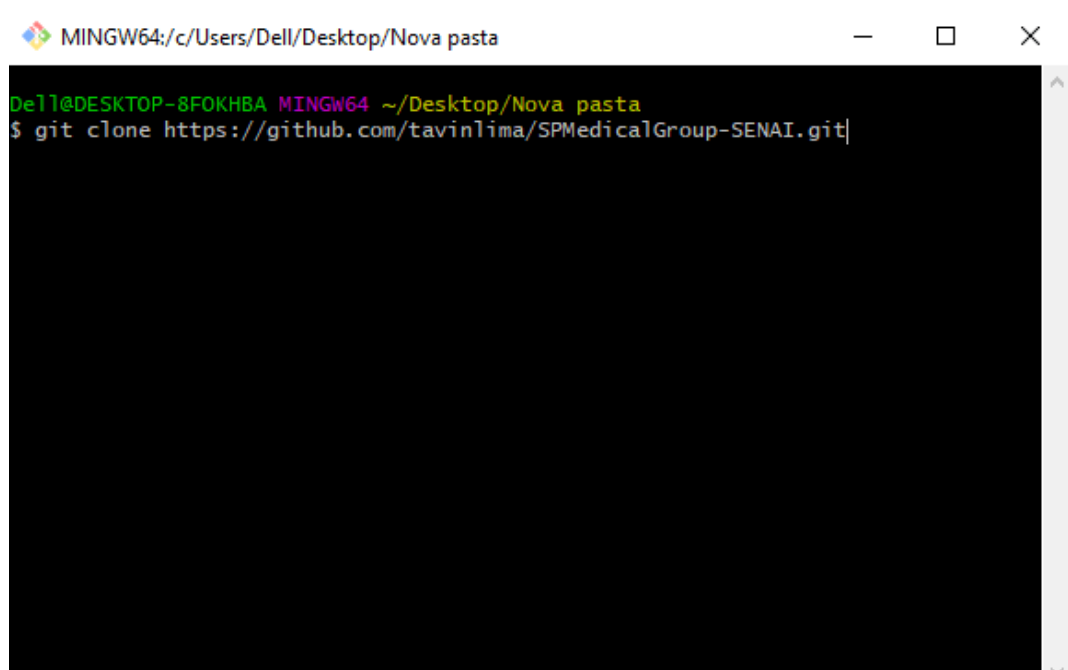


**2º passo** - Clique com o botão direito dentro de uma pasta vazia e clique em “Git Bash Here”:



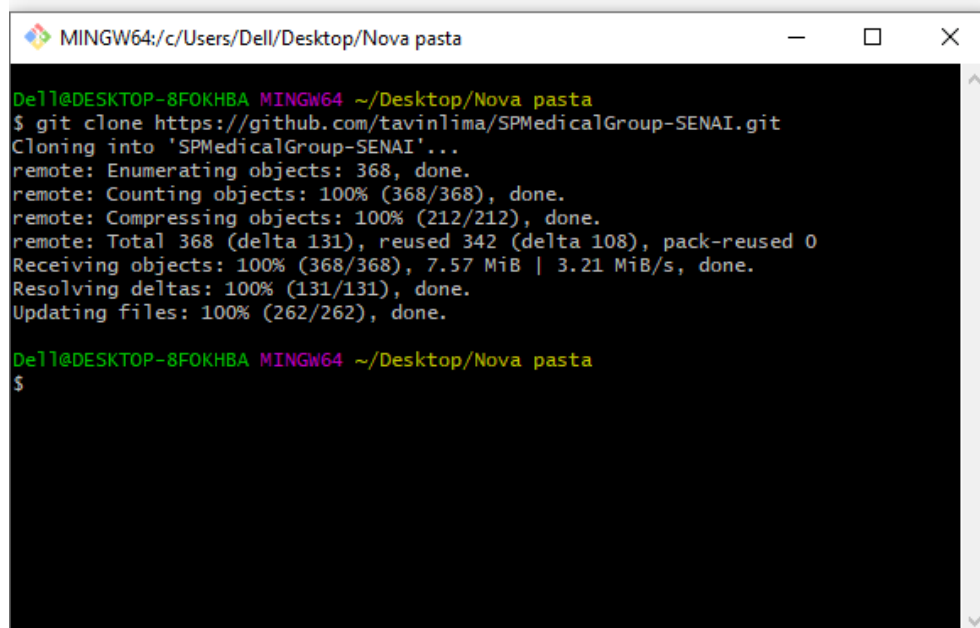


**3º passo** – Quando abrir essa tela, digite “git clone <link do repositório>” e aperte enter:



```
MINGW64:/c/Users/Dell/Desktop/Nova pasta
Dell@DESKTOP-8FOKHBA MINGW64 ~/Desktop/Nova pasta
$ git clone https://github.com/tavinlima/SPMedicalGroup-SENAI.git
```

.git	29/09/2021 21:35	Pasta de arquivos	
Back-end	29/09/2021 21:35	Pasta de arquivos	
Banco de dados	29/09/2021 21:35	Pasta de arquivos	
Postman	29/09/2021 21:35	Pasta de arquivos	
documentacao_SpMedGroup	29/09/2021 21:35	Documento do Mi...	113 KB
documentacao_SpMedGroup	29/09/2021 21:35	Microsoft Edge P...	230 KB
Escopo_SPMedGroup	29/09/2021 21:35	Microsoft Edge P...	417 KB
Rascunho_entidades.txt	29/09/2021 21:35	Documento de Te...	1 KB



```
MINGW64:/c/Users/Dell/Desktop/Nova pasta
Dell@DESKTOP-8FOKHBA MINGW64 ~/Desktop/Nova pasta
$ git clone https://github.com/tavinlima/SPMedicalGroup-SENAI.git
Cloning into 'SPMedicalGroup-SENAI'...
remote: Enumerating objects: 368, done.
remote: Counting objects: 100% (368/368), done.
remote: Compressing objects: 100% (212/212), done.
remote: Total 368 (delta 131), reused 342 (delta 108), pack-reused 0
Receiving objects: 100% (368/368), 7.57 MiB | 3.21 MiB/s, done.
Resolving deltas: 100% (131/131), done.
Updating files: 100% (262/262), done.

Dell@DESKTOP-8FOKHBA MINGW64 ~/Desktop/Nova pasta
$
```

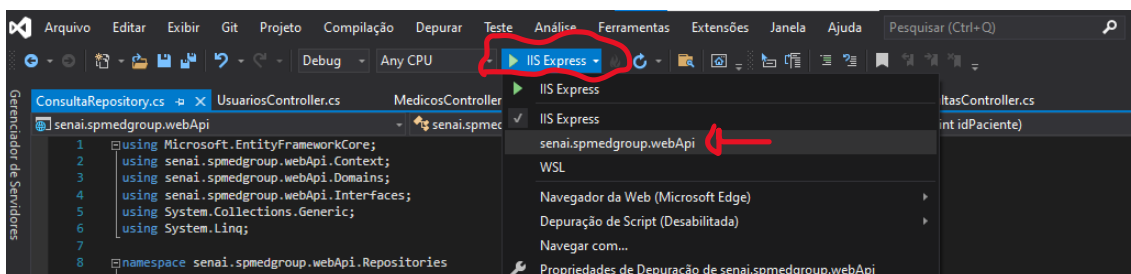
**4º passo** – Clique na pasta de “Back-End” -> “senai.spmedgroup.webApi”.

.git	29/09/2021 21:35	Pasta de arquivos	
Back-end	29/09/2021 21:35	Pasta de arquivos	
Banco de dados	29/09/2021 21:35	Pasta de arquivos	
Postman	29/09/2021 21:35	Pasta de arquivos	
documentacao_SpMedGroup	29/09/2021 21:35	Documento do Mi...	113 KB
documentacao_SpMedGroup	29/09/2021 21:35	Microsoft Edge P...	230 KB
Escopo_SPMedGroup	29/09/2021 21:35	Microsoft Edge P...	417 KB
Rascunho_entidades.txt	29/09/2021 21:35	Documento de Te...	1 KB
senai.spmedgroup.webApi	24/09/2021 15:07	Pasta de arquivos	

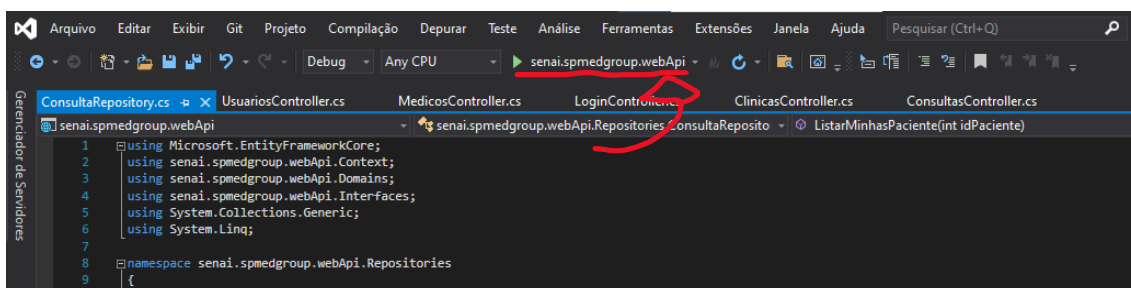
**5º passo** – Clique em “senai.spmedgroup.webApi.sln”, é a solução do arquivo. **SEMPRE** abra o arquivo pela solução, é onde você terá acesso a API.

.vs	24/09/2021 15:07	Pasta de arquivos	
senai.spmedgroup.webApi	29/09/2021 19:49	Pasta de arquivos	
senai.spmedgroup.webApi.sln	24/09/2021 15:07	Visual Studio Solu...	2 KB

**6º passo** – Troque “IIS Express” pelo nome da sua API, apertando na seta ao lado do nome.



**7º passo** – Clique no botão e aguarde.



**8º passo** – Se aparecer essa tela, a compilação deu certo.

```
C:\Users\Dell\Desktop\SENAI_SpMedGroup\Back-end\senai.spmedgroup.webApi\senai.spmedgroup.webApi\bin\Debug\net5.0\senai.spmedgroup.we...
info: Microsoft.Hosting.Lifetime[0]
      Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\Dell\Desktop\SENAI_SpMedGroup\Back-end\senai.spmedgroup.webApi\senai.spmedgroup.webApi
```

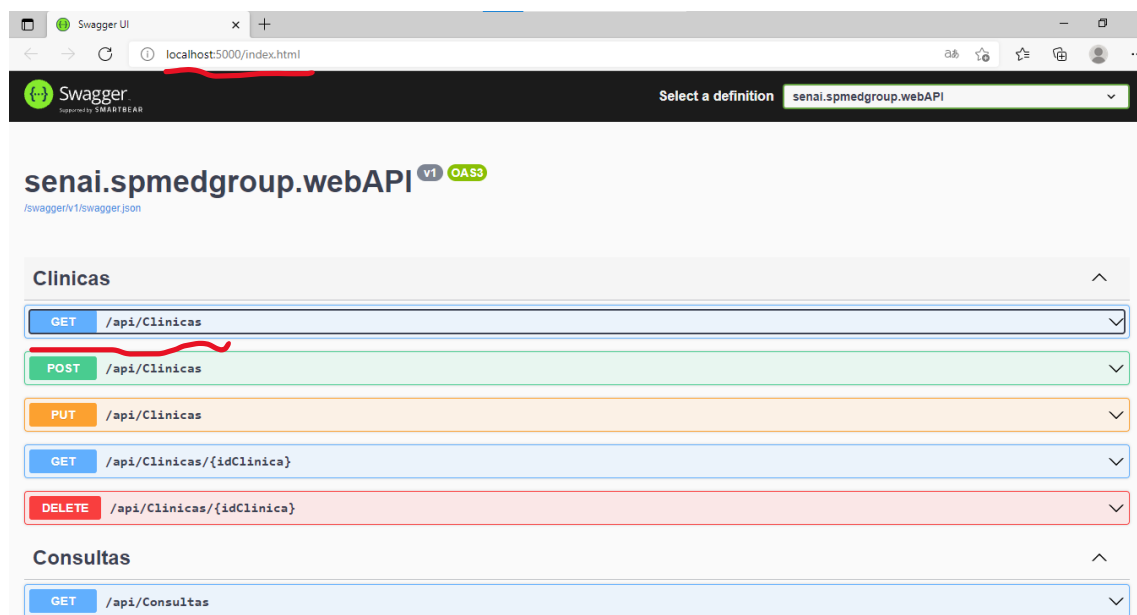
**9º passo** – Automaticamente, a API irá abrir essa tela no seu navegador padrão, a home é a documentação da sua API.

O link na barra de pesquisa até a primeira “/” é o que você vai colar no software postman para a requisição.

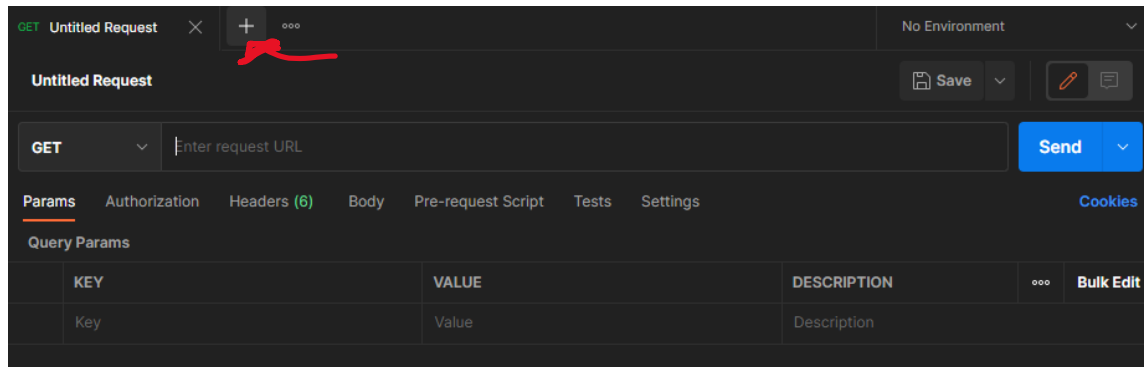
Por exemplo: “http://localhost:5000/”.

Pela documentação, você escolhe qual requisição quer fazer e pega seu endpoint.

Por exemplo: “<http://localhost:5000/api/Clinicas>”.

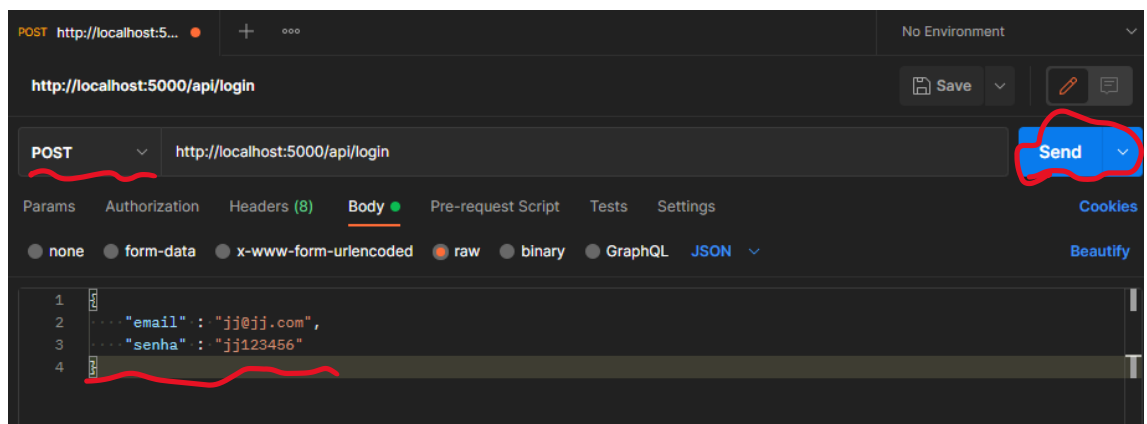


**10º passo** – Abra o postman, e aperte no “+” para fazer uma nova requisição.



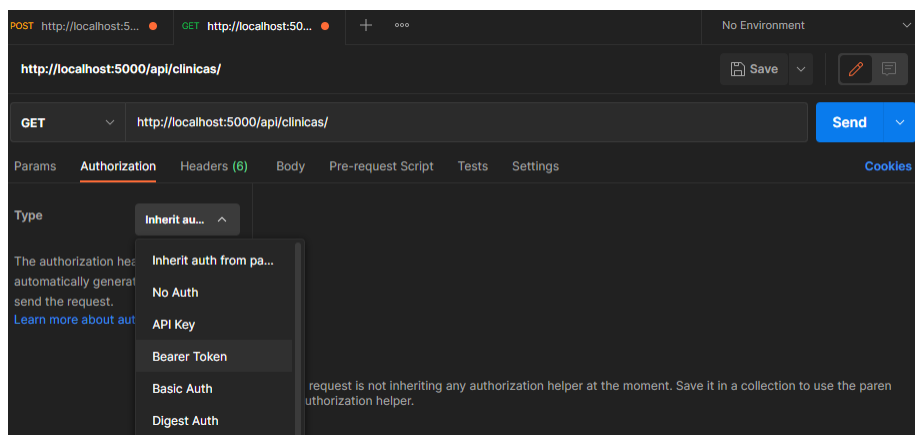
**11º passo** – Antes de tudo, altere a URL para “<http://localhost:5000/api/login>”, e seu verbo para “POST” faça login por meio de um usuário padrão, e aperte “SEND”.

Copie o token gerado, ele será importante para fazer a próxima requisição

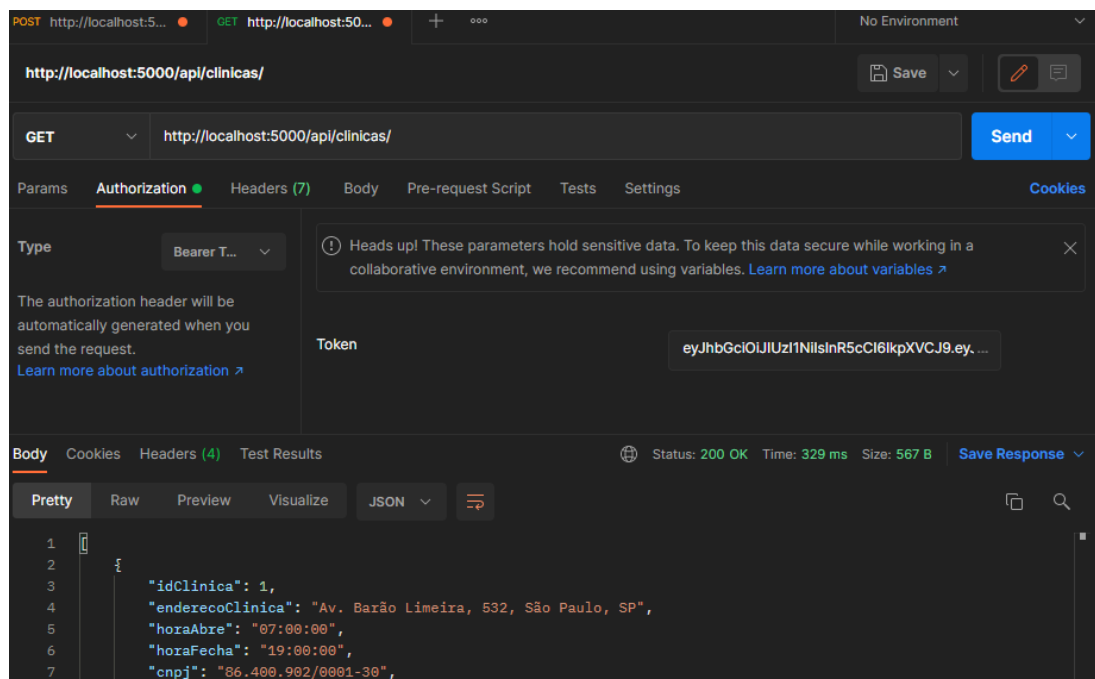


**13º passo** – A partir do token gerado, faça outra requisição e coloque na URL a requisição que deseja fazer.

Vá em “Authorization”, mude o tipo e selecione “Bearer Token”:



**14º passo** – Coloque o token, aperte em enviar e pronto! Sua requisição está pronta!



## Funcionalidades

### Sistema Web

#### 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;

### Sistema Mobile

#### Perfis de usuário:

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

#### Funcionalidades:

1. O **paciente** poderá visualizar suas próprias consultas;
2. O **médico** poderá ver as consultas (os agendamentos) associados a ele;

## 2. Front-End

O sistema front-end foi desenvolvido pela biblioteca JavaScript React, no editor de código VS Code.

O React foi desenvolvido pela empresa Facebook, sendo uma biblioteca de código aberto para interfaces gráficas, com foco no uso de componentização para melhor aproveitamento de código.

O React também é quem se comunica com o sistema back-end, para que esse comunique com o banco de dados e possa fazer o sistema todo funcionar.

### Passo a passo:

Com a API rodando, abra o cmd na pasta em que se encontra a aplicação web;

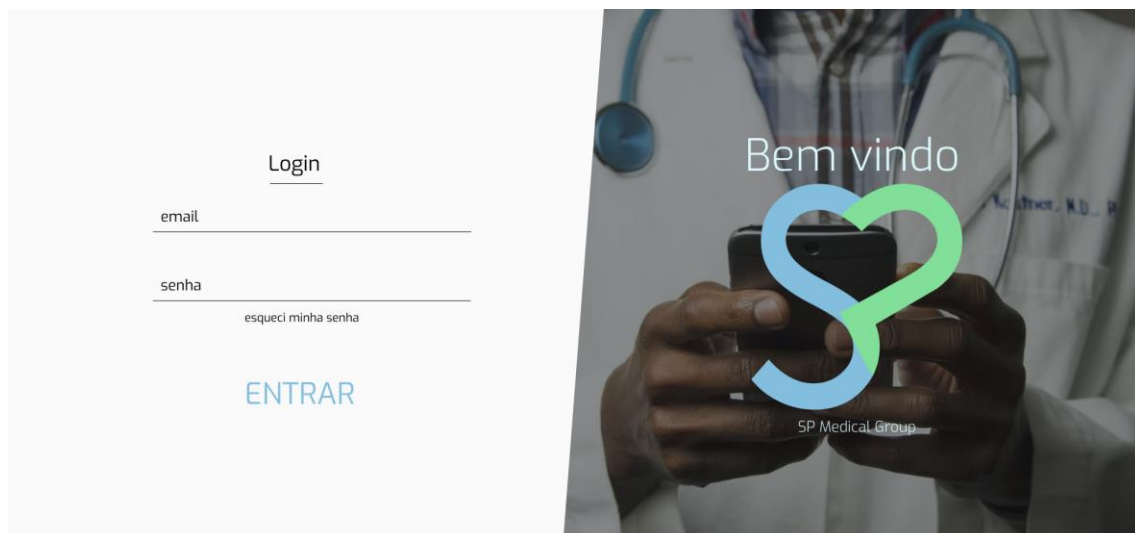
Digite “npm i” para instalar os pacotes necessários para rodar a aplicação;

Depois de tudo baixado, digite “npm start” para iniciar o sistema;

Seu computador abrirá o front-end da aplicação no seu navegador padrão, e já pode testar!

### Layout das Telas:

#### Tela de login



## Listagem e cadastro de consultas – ADM

**Consultas**

**Lista de consultas**

- 20/01/2020 15:00  
PACIENTE: MARIANA  
MÉDICO: HELENA STRADA  
SITUAÇÃO: REALIZADA  
REALIZADA  
ESPECIALIDADE: PEDIATRIA  
DESCRIÇÃO: CHECK-UP ANUAL
- 20/01/2020 10:00  
PACIENTE: ALEXANDRE  
MÉDICO: ROBERTO POSSARLE  
SITUAÇÃO: CANCELADA  
CANCELADA  
ESPECIALIDADE: PSIQUIATRIA  
DESCRIÇÃO: CONSULTA SEMANAL
- 07/02/2020 11:00  
PACIENTE: FERNANDO  
MÉDICO: ROBERTO POSSARLE  
SITUAÇÃO: REALIZADA  
REALIZADA  
ESPECIALIDADE: PSIQUIATRIA  
DESCRIÇÃO: CONSULTA MENSAL
- 06/02/2019 10:00  
PACIENTE: ALEXANDRE  
MÉDICO: ROBERTO POSSARLE
- 07/02/2019 11:00  
PACIENTE: HENRIQUE  
MÉDICO: RICARDO LEMOS
- 08/03/2020 15:00  
PACIENTE: MARIANA  
MÉDICO: HELENA STRADA

**NOVA CONSULTA**

PACIENTE:

MÉDICO:

DATA DA CONSULTA:

**Cadastrar consulta**

## Listagem e alteração de descrição de consultas – Médico

**Consultas**

**Lista de consultas**

- 07/02/2019 11:00  
PACIENTE: HENRIQUE  
MÉDICO: RICARDO LEMOS  
SITUAÇÃO: CANCELADA  
CANCELADA  
ESPECIALIDADE: ANESTESIOLOGIA  
DESCRIÇÃO: CIRURGIA DO APÊNDICE
- 09/03/2020 11:00  
PACIENTE: HENRIQUE  
MÉDICO: RICARDO LEMOS  
SITUAÇÃO: AGENDADA  
AGENDADA  
ESPECIALIDADE: ANESTESIOLOGIA  
DESCRIÇÃO: PARTO NORMAL

**BUSCAR CONSULTA**

DESCRIÇÃO:

**Adicionar Descrição**

SP MEDICAL GROUP 2021

## Listagem de consultas – Paciente

**Consultas**

**Lista de consultas**

- 04/10/2021 às 14:00  
PACIENTE: JUÁREZ SANTOS  
MÉDICO: ROBERTO CRISTO  
SITUAÇÃO: AGENDADA  
AGENDADA  
ESPECIALIDADE: OBSTRETA  
DESCRIÇÃO: PARTO NORMAL
- 04/10/2021 às 14:00  
PACIENTE: JUÁREZ SANTOS  
MÉDICO: ROBERTO CRISTO  
SITUAÇÃO: AGENDADA  
AGENDADA  
ESPECIALIDADE: OBSTRETA  
DESCRIÇÃO: PARTO NORMAL
- 04/10/2021 às 14:00  
PACIENTE: JUÁREZ SANTOS  
MÉDICO: ROBERTO CRISTO  
SITUAÇÃO: AGENDADA  
AGENDADA  
ESPECIALIDADE: OBSTRETA  
DESCRIÇÃO: PARTO NORMAL
- 04/10/2021 às 14:00  
PACIENTE: JUÁREZ SANTOS  
MÉDICO: ROBERTO CRISTO  
SITUAÇÃO: AGENDADA  
AGENDADA  
ESPECIALIDADE: OBSTRETA  
DESCRIÇÃO: PARTO NORMAL



### 3. Mobile

O sistema mobile foi desenvolvido pela biblioteca React-Native, com o editor de código VS Code e os testes foram feitos pelo emulador de Pixel 2 no Android Studio.

React-Native é a biblioteca do React para aplicações mobile de IOS e Android de forma nativa.

Aplicativos nativos são desenvolvidos para plataformas específicas, onde se explora todo o potencial sistêmico, e na linguagem exclusiva para o sistema operacional (como o Swift para o IOS e o Kotlin para o Android), tendo um custo bem maior, mas acesso a funcionalidades do celular, como o GPS ou a câmera.

Já o híbrido é feito para que se possa fazer um mesmo código para ser usados nos dois sistemas (como o próprio React-Native), auxiliando o desenvolvedor e facilitando parte do trabalho. Entre suas vantagens estão ao custo mais baixo e a funcionalidade em todas as plataformas, e atualmente sua performance quase se iguala a de um nativo, mas ainda possui um design mais restrito.

#### **Passo a passo:**

(Você precisará ter um emulador de android baixado no seu computador ou um celular android conectado)

Assim como na aplicação web, digite “npm i” na pasta em que se encontra o sistema mobile para instalar os pacotes necessários para rodar a aplicação;

Logo em seguida, digite “npx react-native run-android”;

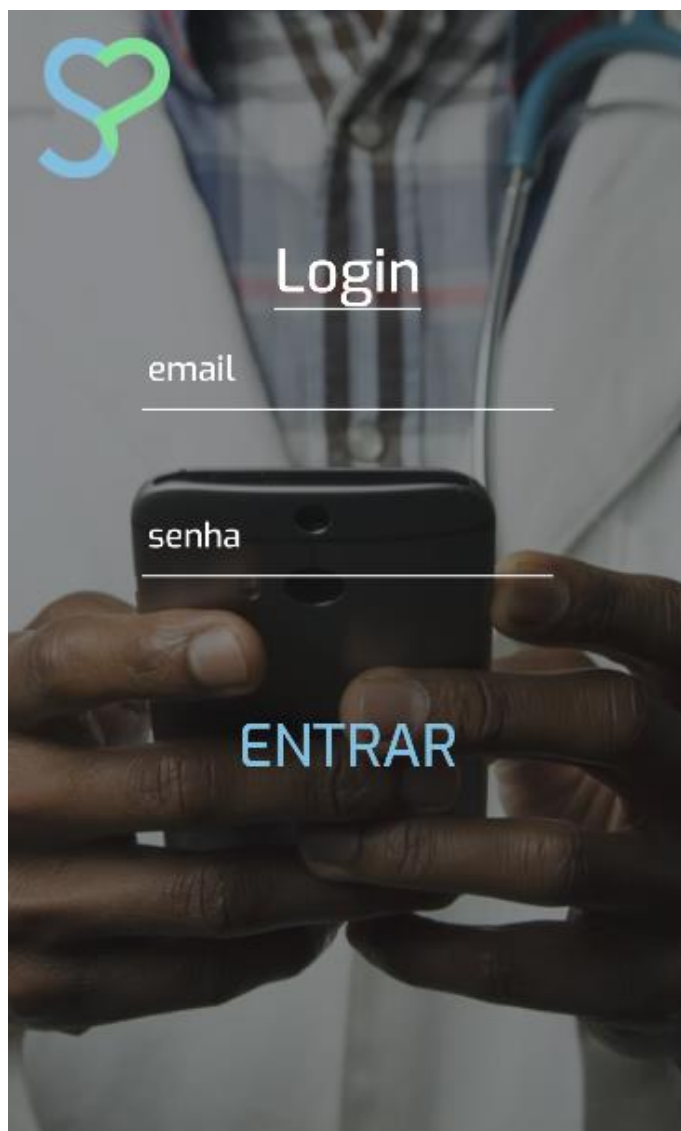
Automaticamente seu computador iniciará o emulador instalado anteriormente ou o app será instalado no celular conectado.

Caso tudo dê certo, a tela de login aparecerá e seu sistema já estará rodando!

(Lembre-se de alterar o caminho da API e do axios presente no sistema mobile para o seu endereço de IP)


Layout das telas:

Tela de login:



## Tela de listagem – Médico:

SAIR

 Lista de consultas

07/02/2019 11:00

MÉDICO:  
RICARDO LEMOS

PACIENTE:  
HENRIQUE

SITUAÇÃO:  
CANCELADA

ESPECIALIDADE:  
ANESTESIOLOGIA


DESCRIÇÃO:  
CIRURGIA DO APÊNDICE

09/03/2020 11:00

MÉDICO:  
RICARDO LEMOS

Tela de listagem – Paciente:

SAIR

 Lista de consultas

07/02/2019 11:00

MÉDICO:  
RICARDO LEMOS

PACIENTE:  
HENRIQUE

SITUAÇÃO:  
CANCELADA

ESPECIALIDADE:  
ANESTESIOLOGIA

DESCRIÇÃO:  
CIRURGIA DO APÊNDICE

09/03/2020 11:00

MÉDICO:  
RICARDO LEMOS

## 5. Banco de Dados NoSQL

O Banco de Dados não relacional utilizado foi o MongoDB, um software escrito em linguagem C++, em um formato "BSON" ou de "Binary JSON", bem parecido com o formato JSON convencional, mas codificado em linguagem binária.

Conceito: O banco de dados NoSQL se refere a um banco de dados não relacional de alto desempenho "Not Only SQL", ou seja, em que geralmente o SQL não é utilizado para consulta, sendo utilizados quando o banco de dados relacional não se faz útil ou necessário, como armazenar dados diretamente, sem que fiquem com campos "nulos" no banco.

Diferenças de um banco relacional para um não relacional:

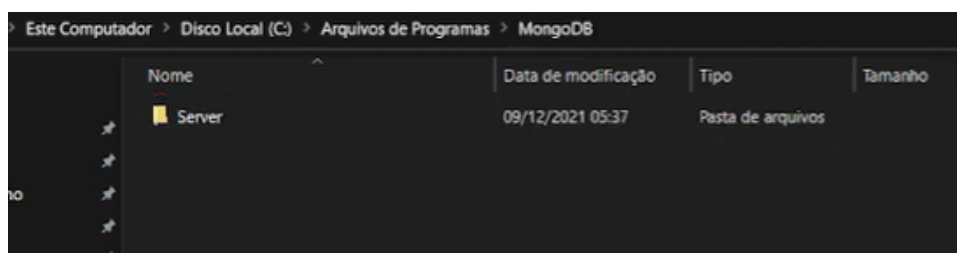
O principal é que não temos mais "tabelas" e "linhas" em bancos NoSQL, mas "collections" que armazenam os "documents" com os campos de dados.

Os comandos também são diferentes, como o "INSERT", "SELECT", "UPDATE" e "DELETE" que dão lugar aos "insert/insertOne()/insertMany()", "find()", "update()" e "drop()" respectivamente.

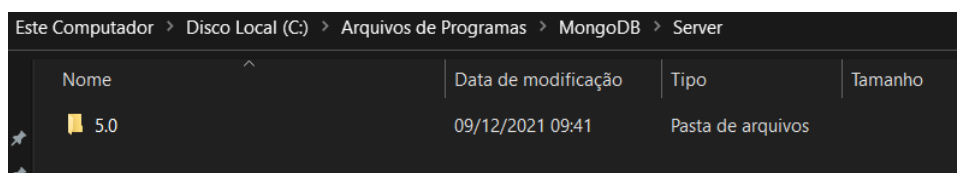
No mongoDb, as databases só aparecerão no registro quando tiver uma collection registrada, diferente do banco relacional, que aparecerá assim que for criada.

Enquanto o banco relacional é estritamente estruturado, baseado nas relações entre diferentes tabelas, no banco não relacional não há isso, sendo muito mais flexível no armazenamento.

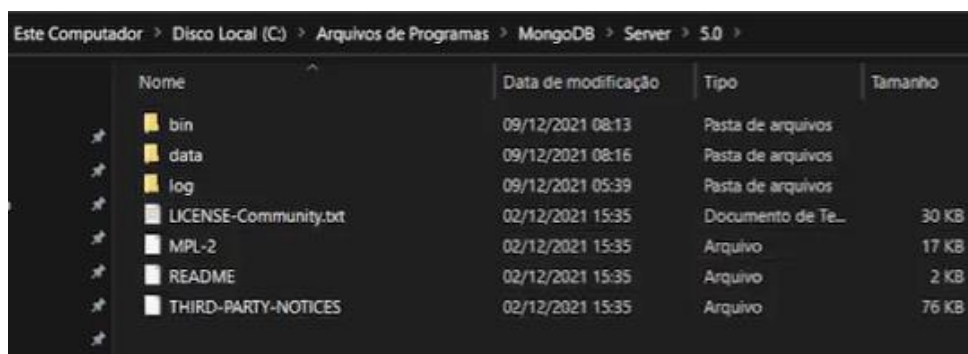
Depois de instalado, para abri-lo basta procurar pela pasta "MongoDB", dentro de "Arquivos de Programas" no Disco Local do seu computador.



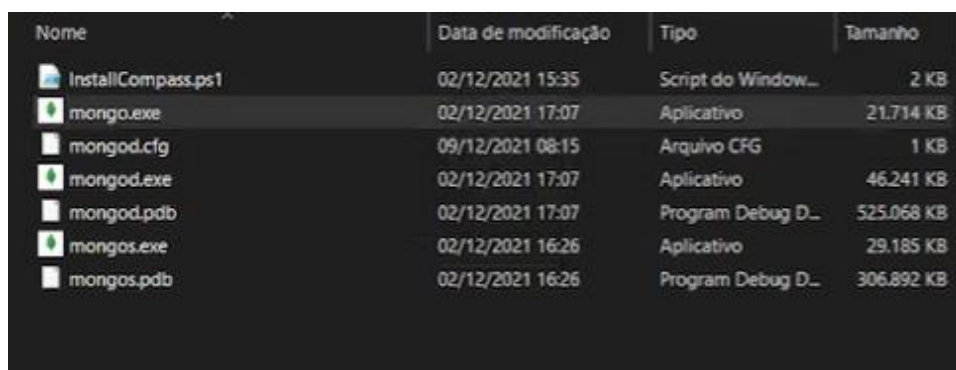
Clique na pasta "Server", logo em seguida na pasta com o nome da versão do Mongo que você baixou:



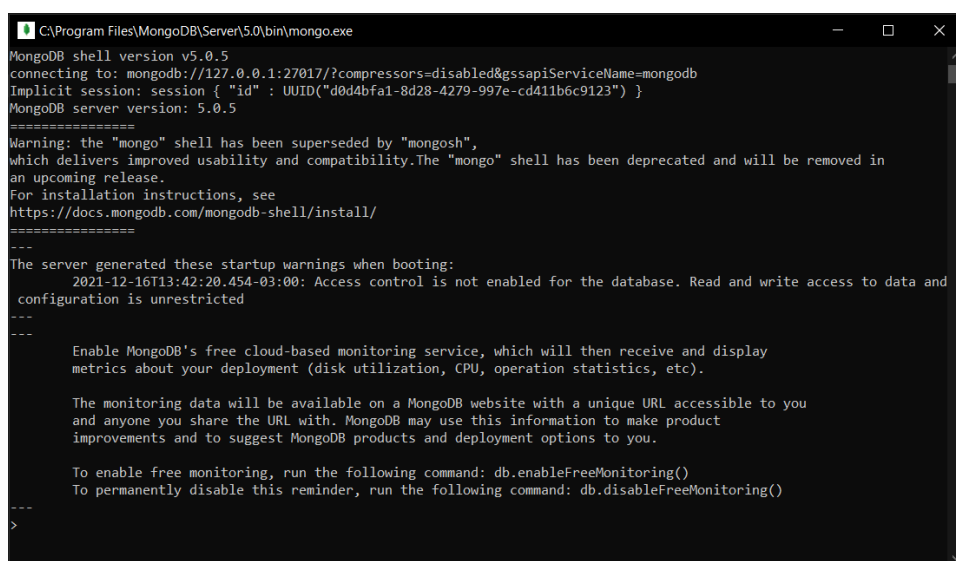
Abra a pasta bin:



E abra o arquivo “Mongo.exe”:



Se essa tela abrir, seu banco já está rodando:



E digite o comando “show dbs” para achar as databases registradas:

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility. The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
=====
The server generated these startup warnings when booting:
  2021-12-16T13:42:20.454-03:00: Access control is not enabled for the database. Read and write access to data and
  configuration is unrestricted
-----
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
  metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
  and anyone you share the URL with. MongoDB may use this information to make product
  improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----
> show dbs
admin            0.000GB
config           0.000GB
cursos           0.000GB
local            0.000GB
spmed_gustavo    0.000GB
spmed_manha      0.000GB
```

Para registrar novos dados no banco, abra o postman na URL

<http://localhost:5000/api/Localizacoes>, passando o token de Administrador no headers, com o seguinte texto:

```
{
  "latitude": "exemplo",
  "longitude": "exemplo"
}
```

E para visualizar no front-end seus registros, basta logar como administrador e abrir o arquivo html “mapa.html” dentro de “mapas” na pasta “pages” no vs code do front end.

Com os bancos relacional e NoSql, API, front-end e mobile funcionando, seu SP Medical Group está pronto para uso!