

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 |
| 4. | Back-End..... | 4 |
| | Funcionalidades..... | 5 |
| | Sistema Web..... | 5 |
| | Perfis de usuário:..... | 5 |
| | Funcionalidades:..... | 5 |
| | Sistema Mobile :..... | 5 |
| | Perfis de usuário:..... | 5 |
| | Funcionalidades:..... | 5 |

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

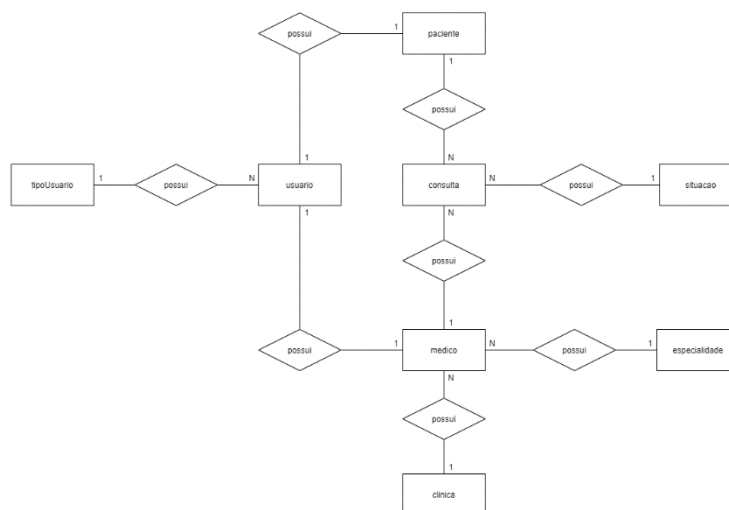
O banco de dados relacional é um banco de dados cuja a estrutura é modelada na forma de “tabelas”, que tenham associações/relações umas com as outras, esse tipo de banco é importante pois os dados são armazenados de forma mais organizada, segura e de relativamente fácil entendimento.

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.

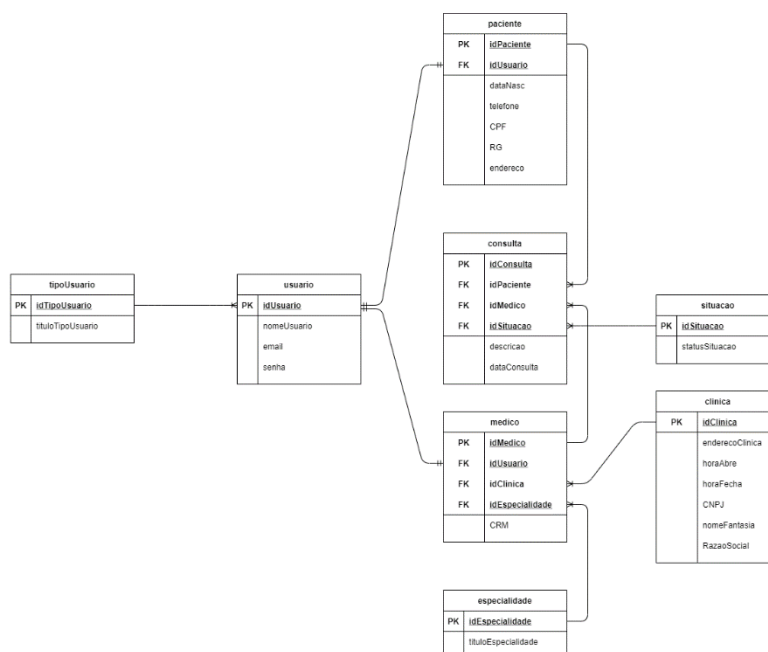
Modelo Conceitual

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.



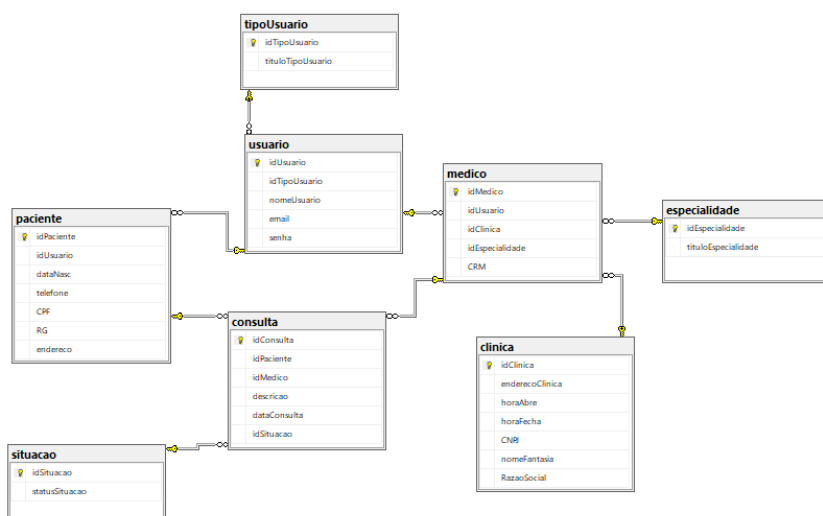
Modelo Lógico

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.

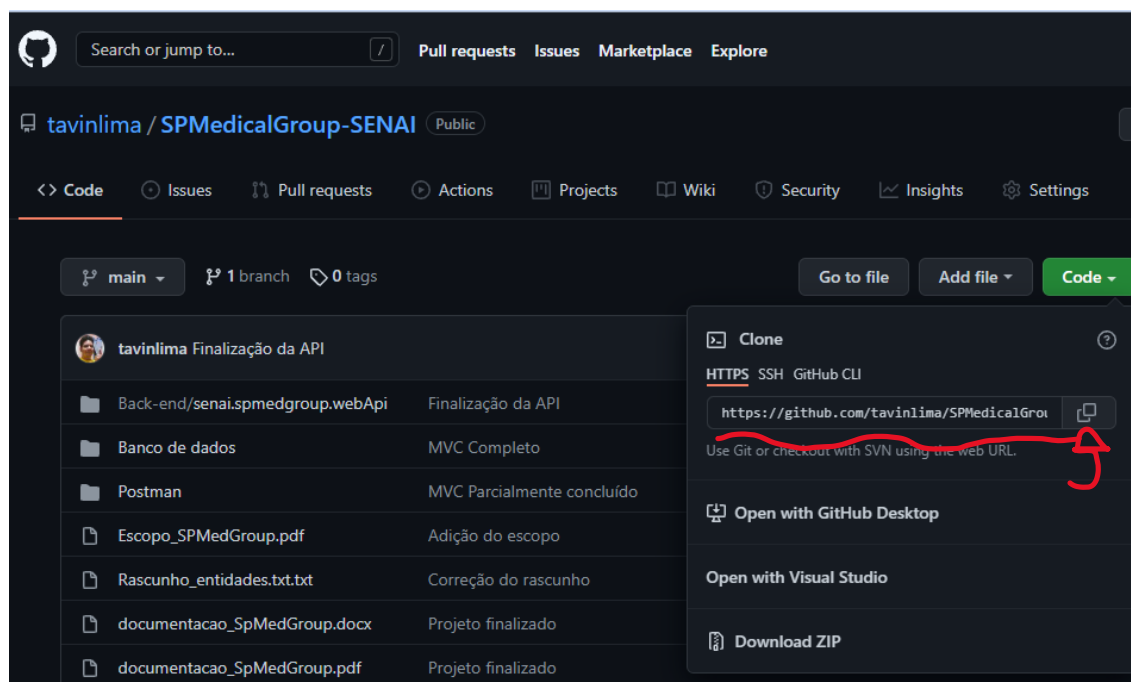
Interfaces – Define como vai ser feito.

Repositórios – Define o que vai ser feito.

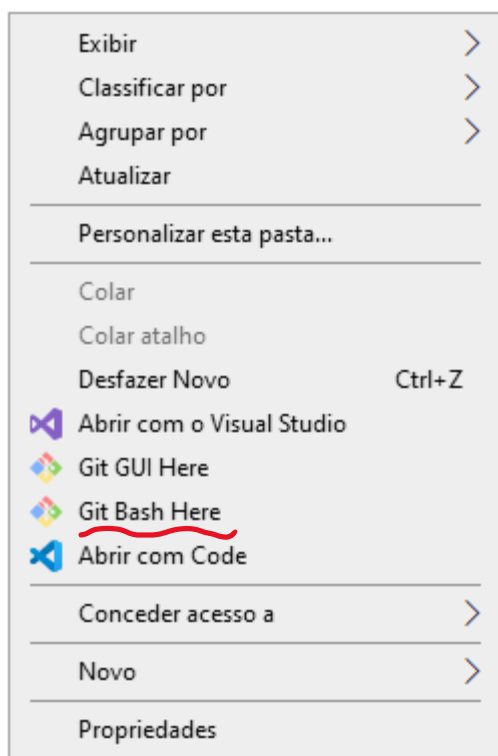
Controladores – Controla o fluxo de dados.

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

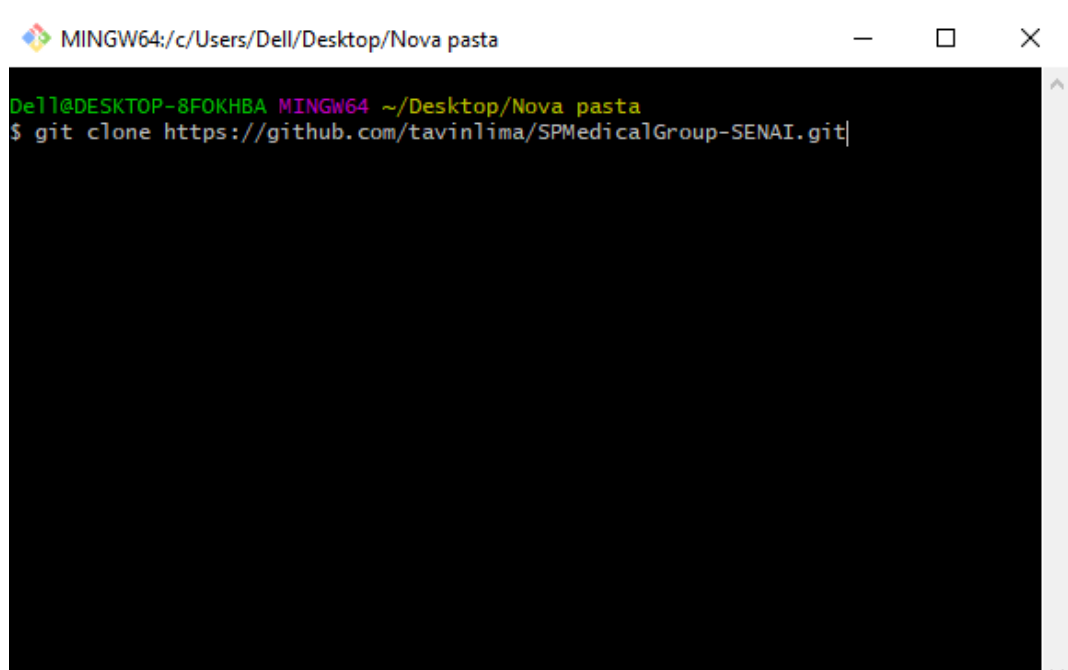
1º passo - Copie o link em destaque de dentro do repositório do GitHub:











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

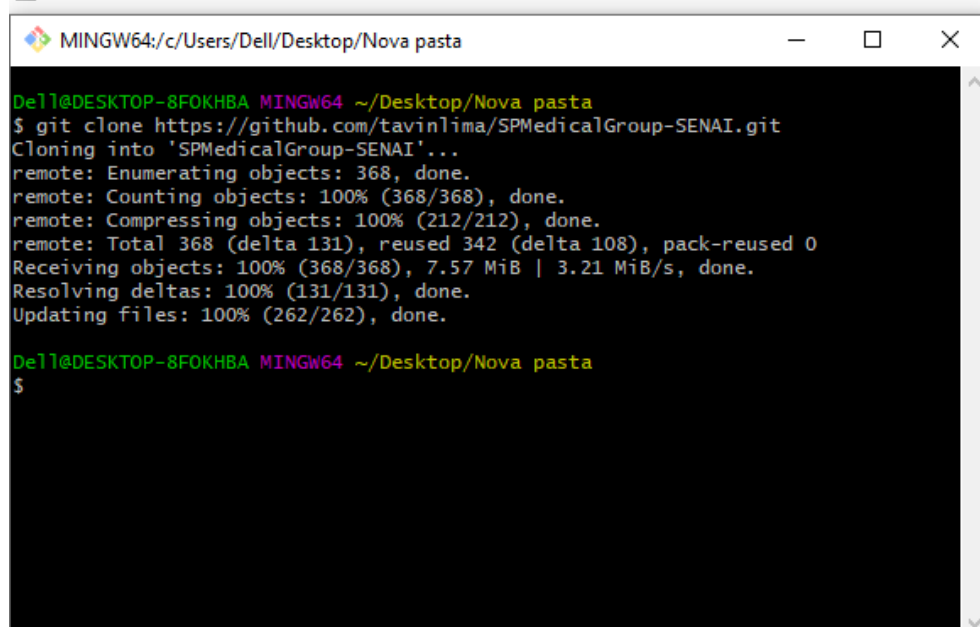


3º passo – Quando abrir essa tela, digite “git clone <link do repositório>” e aperte enter:



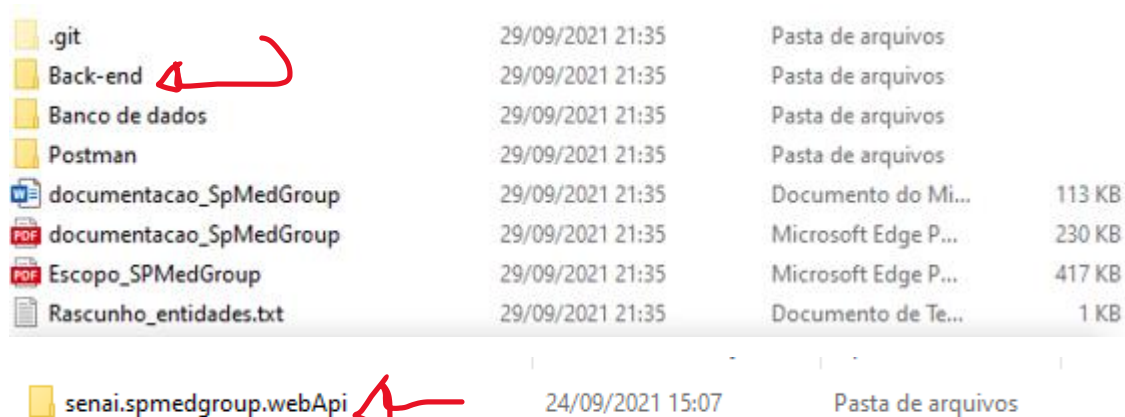
A terminal window titled "MINGW64:/c/Users/Dell/Desktop/Nova pasta" with standard window controls. The prompt is "Dell@DESKTOP-8FOKHBA MINGW64 ~/Desktop/Nova pasta". The command entered is "\$ git clone https://github.com/tavinlima/SPMedicalGroup-SENAI.git".

| | | | | |
|---|-------------------------|------------------|---------------------|--------|
|  | .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 |

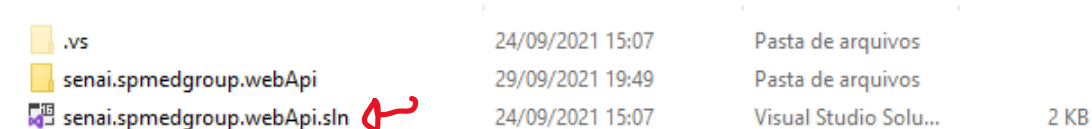


A terminal window titled "MINGW64:/c/Users/Dell/Desktop/Nova pasta" with standard window controls. The prompt is "Dell@DESKTOP-8FOKHBA MINGW64 ~/Desktop/Nova pasta". The command entered is "\$ git clone https://github.com/tavinlima/SPMedicalGroup-SENAI.git". The output shows the cloning process: "Cloning 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." The prompt returns to "\$".

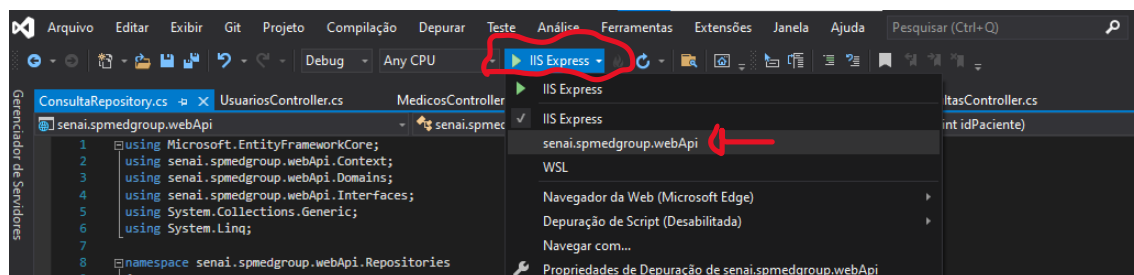
4º passo – Clique na pasta de “Back-End” -> “senai.spmedgroup.webApi”.



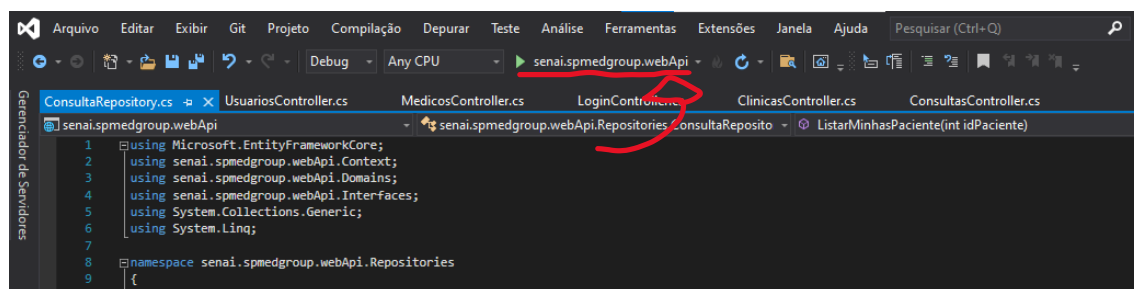
5º passo – Clique em “senai.spmedgroup.webApi.sln”, é a solução do arquivo. **SEMPRE** abra o arquivo pela solução, é onde você terá acesso a API.



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
```

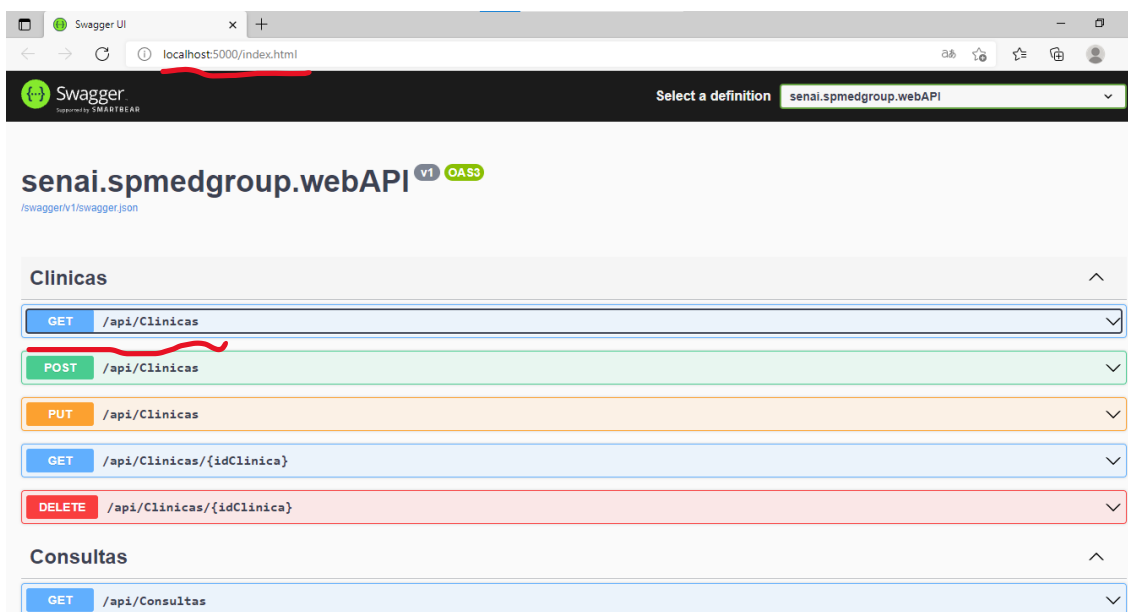
9º passo – Automaticamente, a API irá abrir essa tela no seu navegador padrão, a home é a documentação da sua API.

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

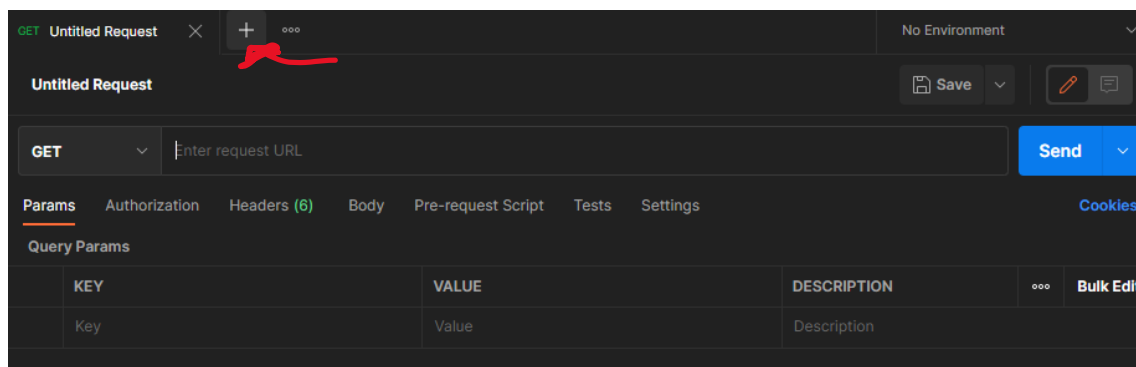
Por exemplo: “http://localhost:5000/”.

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

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

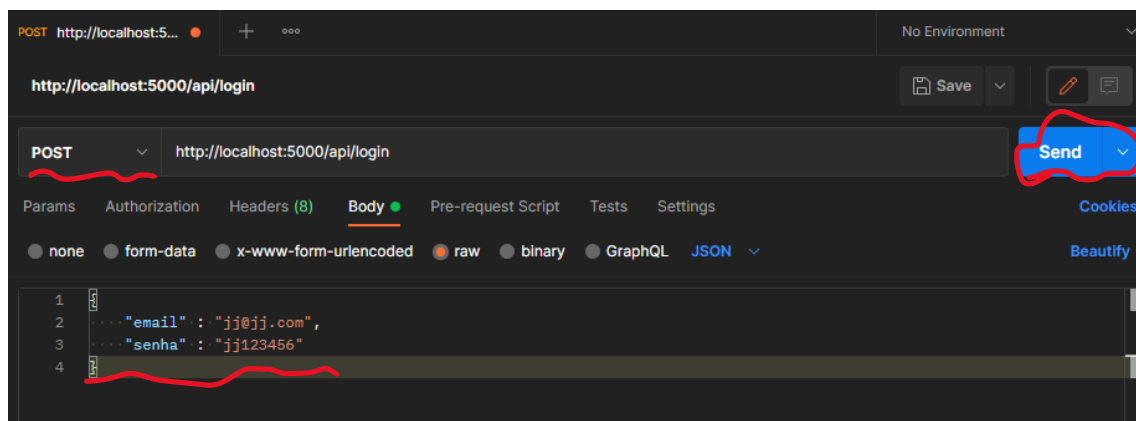


10º passo – Abra o postman, e aperte no “+” para fazer uma nova requisição.



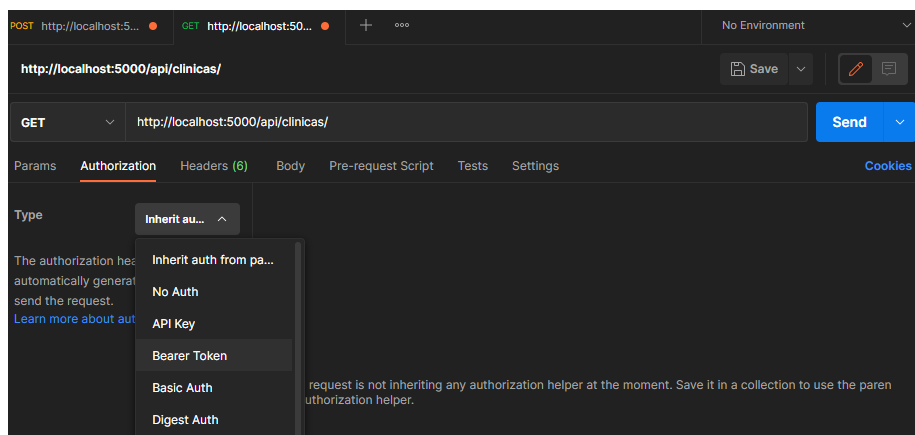
11º passo – Antes de tudo, altere a URL para “<http://localhost:5000/api/login>”, e seu verbo para “POST” faça login por meio de um usuário padrão, e aperte “SEND”.

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

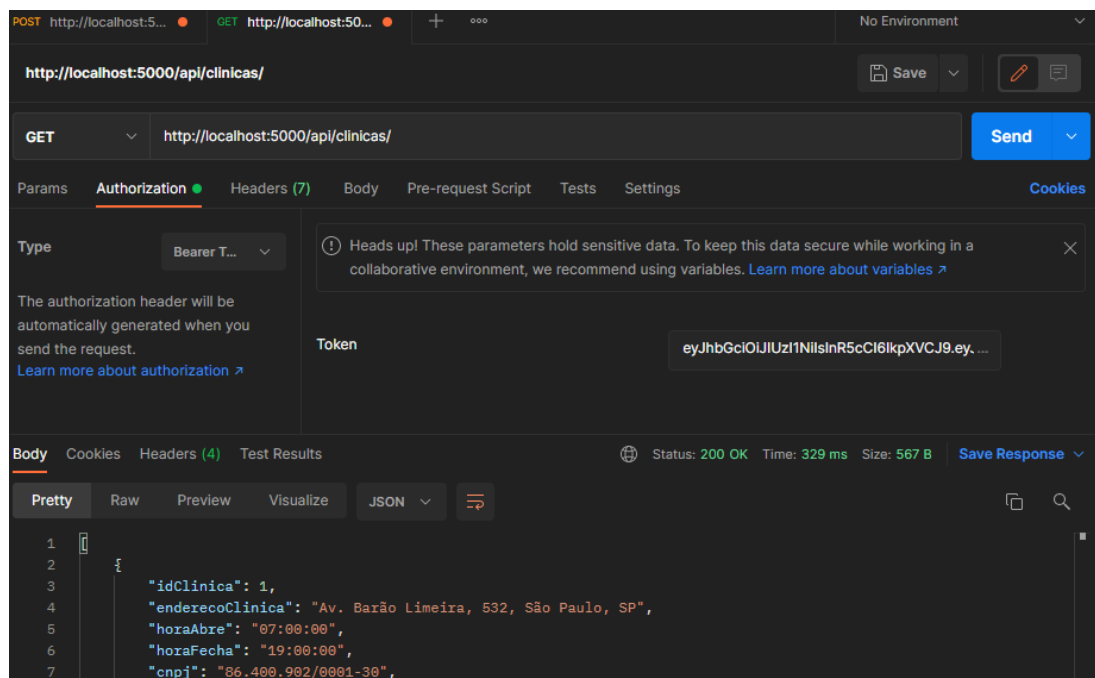


13º passo – A partir do token gerado, faça outra requisição e coloque na URL a requisição que deseja fazer.

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;