**SP Medical Group**

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 5](#_Toc71092226)

[Cronograma 5](#_Toc71092227)

5. Trello..........................................................................................................................................6

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

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

[Funcionalidades 6](#_Toc71092230)

7. UX-UI..........................................................................................................................................7

8. Front-end...................................................................................................................................8

9. Mobile........................................................................................................................................9

7. [Sistema Web 8](#_Toc71092231)

[Perfis de usuário: 8](#_Toc71092232)

[Funcionalidades: 8](#_Toc71092233)

8. [Sistema Mobile 8](#_Toc71092234)

[Perfis de usuário: 8](#_Toc71092235)

[Funcionalidades: 8](#_Toc71092236)

# Resumo

Documentação sobre o projeto SP Medical Group, contendo os detalhes do processo de criação e desenvolvimento da aplicação.

# Descrição do projeto

Projeto criado afim de auxiliar nova clínica médica chamada SP Medical Group, empresa de pequeno porte que atua no ramo da saúde, criada pelo médico Fernando Strada em 2020 na região da Avenida Paulista em São Paulo. Fernando tem uma equipe de médicos que atuam em diversas áreas (pediatria, odontologia, gastrenterologia etc.).

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

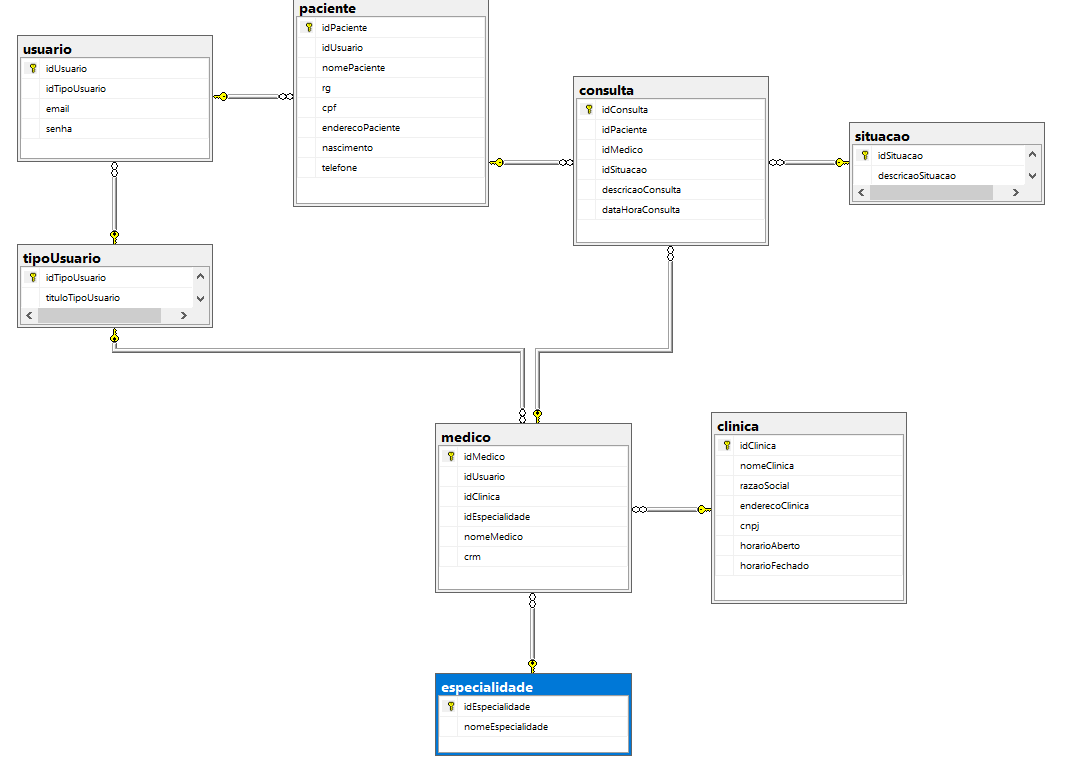
Modelagem Conceitual: Modelo mais simples de identificar as entidades e suas relações entre as tabelas, exemplo: 1:1, 1:N, N:1, N:N

## 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

Ambiente utilizado para organizar o projeto de acordo com tarefas, etapas, links úteis e a data de realização de cada processo.

[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:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

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

Tela de computador com texto preto sobre fundo branco

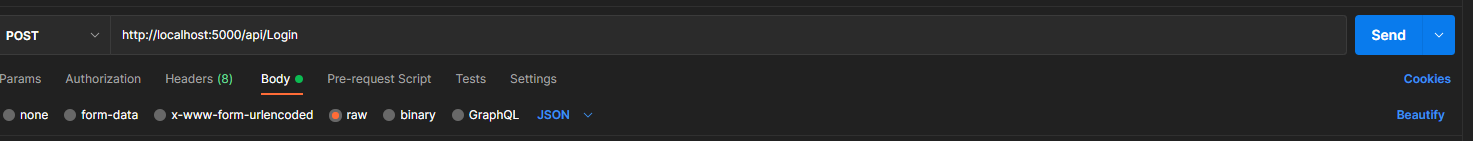
Descrição gerada automaticamente

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

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

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 |

# UX-UI

Descrição sobre o processo de criação de layouts:

## O que é UX-UI?

UX (User Experience) é lado do design que se relaciona com as emoções e experiências dos usuários, sendo exclusivo na relação do usuário com o produto ou serviço. O objetivo é oferecer um serviço que garanta uma boa experiência, gerando satisfação e evitando frustrações no usuário, fidelizando o cliente.

UI Design (User Interface) é área do Design que efetua a elaboração e criação do meio que o usuário interage e controla um determinado dispositivo. Esse dispositivo pode ser um sistema, software, produto, serviço, aplicativo, etc. O objetivo principal dessa área é conseguir elaborar uma interface onde o usuário consiga realizar o que ele pode, quer e espera realizar. Tudo isso de modo simples, fácil e intuitivo.

## Figma

O Figma foi o software utilizado para a construção das interfaces do projeto, tanto os layouts web e mobile, sendo focado no desenvolvimento de sistemas de design gráfico, prototipagem de interface gráfica de usuário e desenvolvimento de UI/UX.

[SpMedGroup – Figma](https://www.figma.com/file/1M6dVUlEnmDQXz8XBUYYif/SpMedGroup)

# Front-end

## O que é Front-end?

# Mobile

## O que é Mobile?

# 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;

Testar sistema Web:

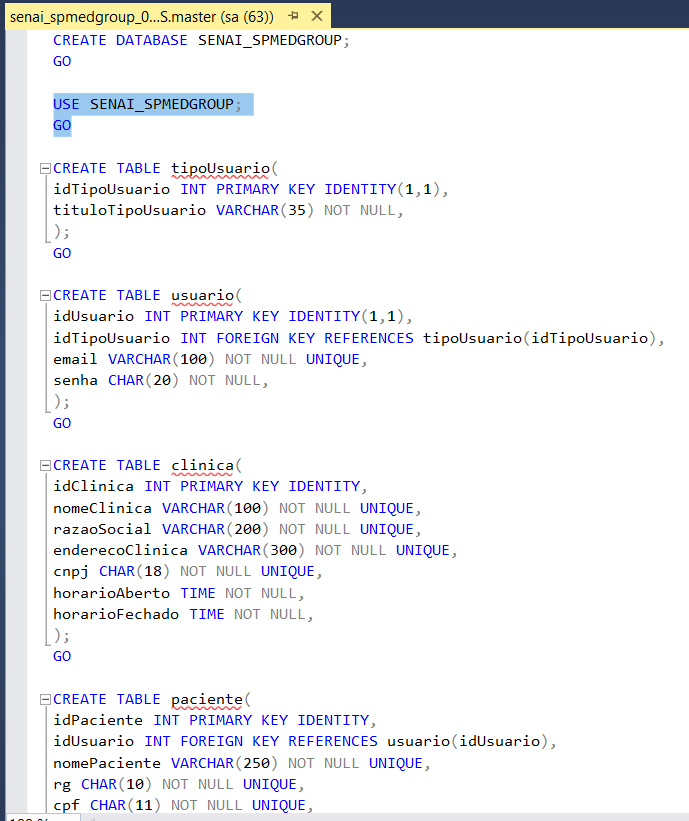
**Banco de dados (Banco Relacional)**

1°Passo:



Primeiramente é preciso ter instalado o SQL Server Management Studio (SSMS). Depois de instalado e configurado será preciso criar o banco de dados chamado SENAI\_SPMEDGROUP, para fazer isso abra o arquivo senai\_spmedgroup\_01\_DDL que está na pasta de banco de dados, depois selecione o comando create database e aperte F5.

2°Passo:



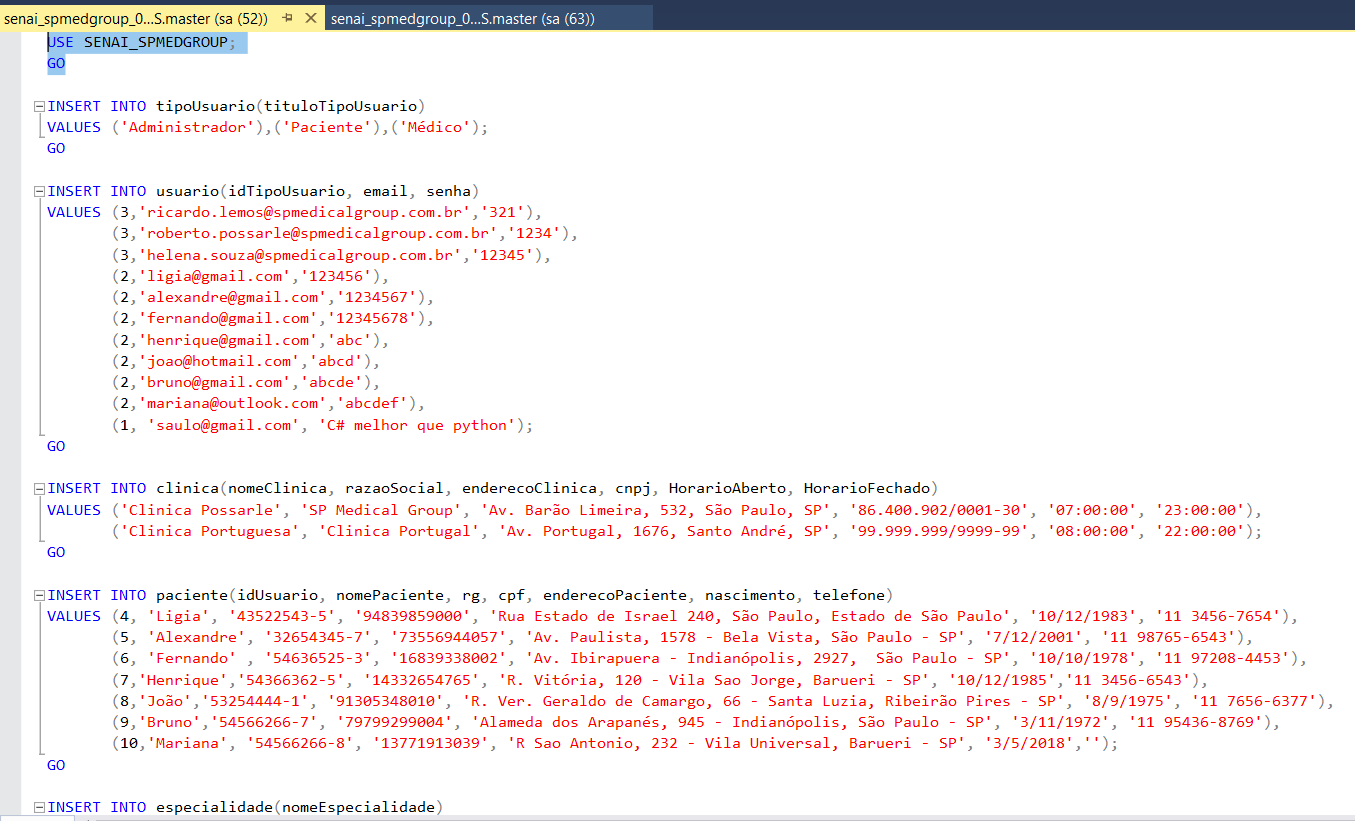
Selecione o comando use para usar o banco criado.

3°Passo:



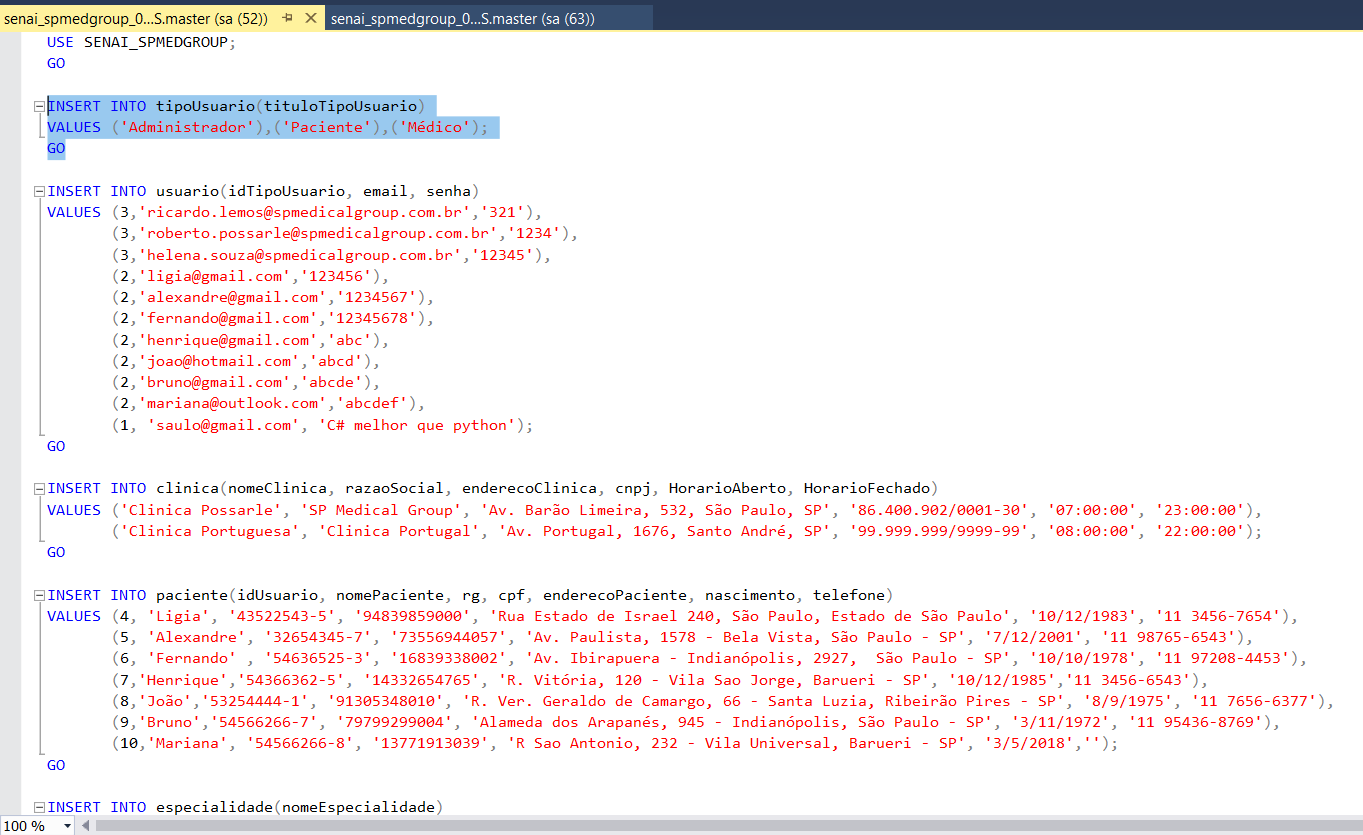
Crie todas as tabelas do mesmo jeito que foi criado o banco de dados e o use. Selecione e crie uma tabela de cada vez para não gerar nenhum conflito.

4°Passo:



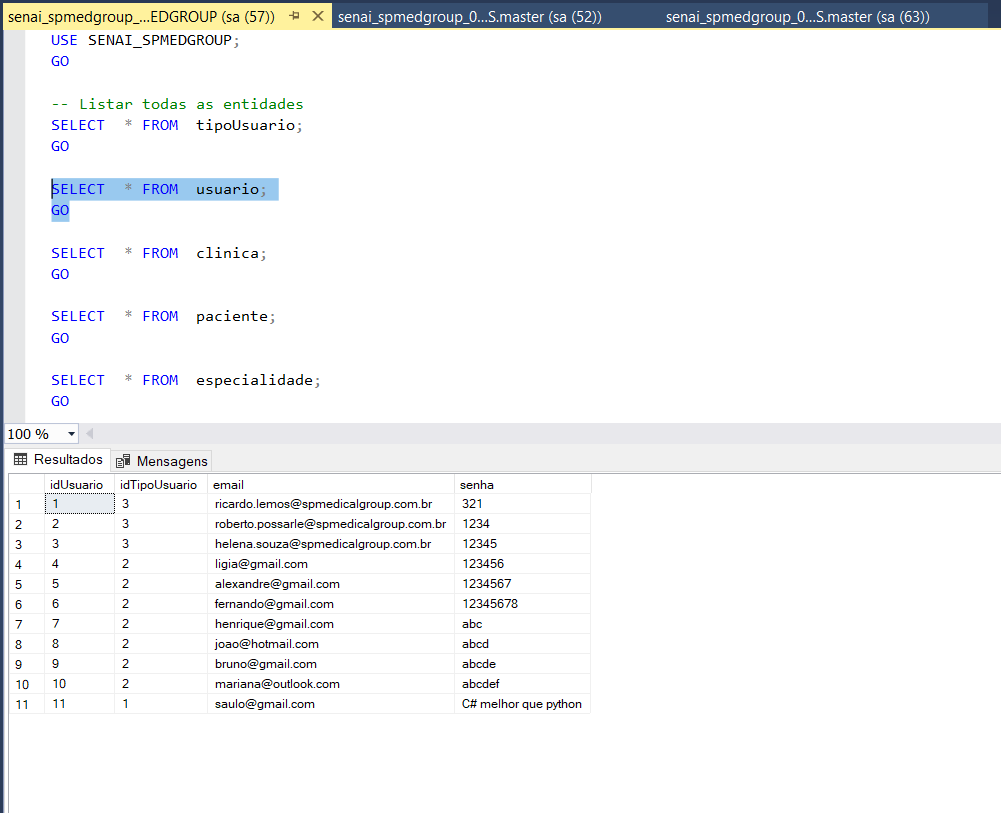
Abra o arquivo DML que está na mesma pasta e selecione o use.

5°Passo:



Selecione o comando insert que insira os dados na tabela, faça igual as tabelas e realize um insert de cada vez.

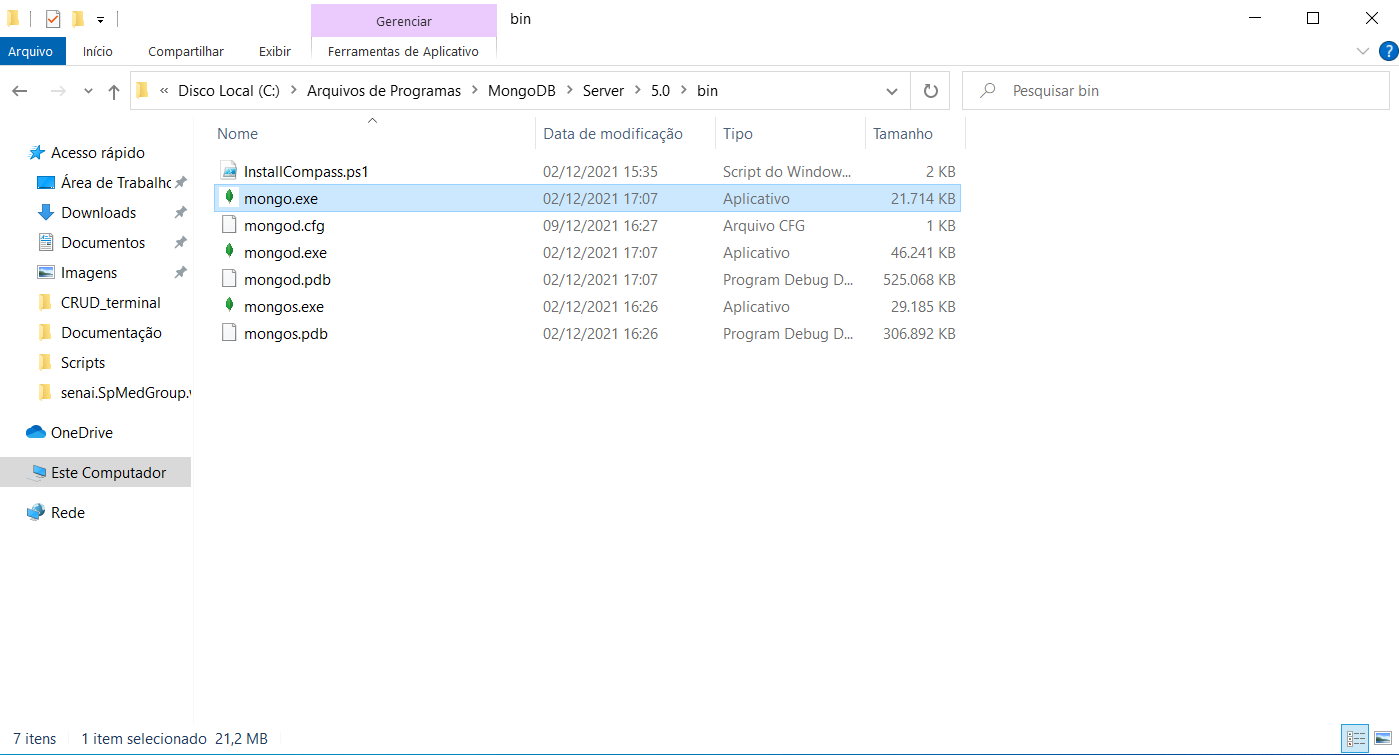
6°Passo:



Abra o arquivo DQL, use o banco SENAI\_SPMEDGROUP e use o comando select a fim de ter certeza de que as tabelas estão criadas com os dados corretos.

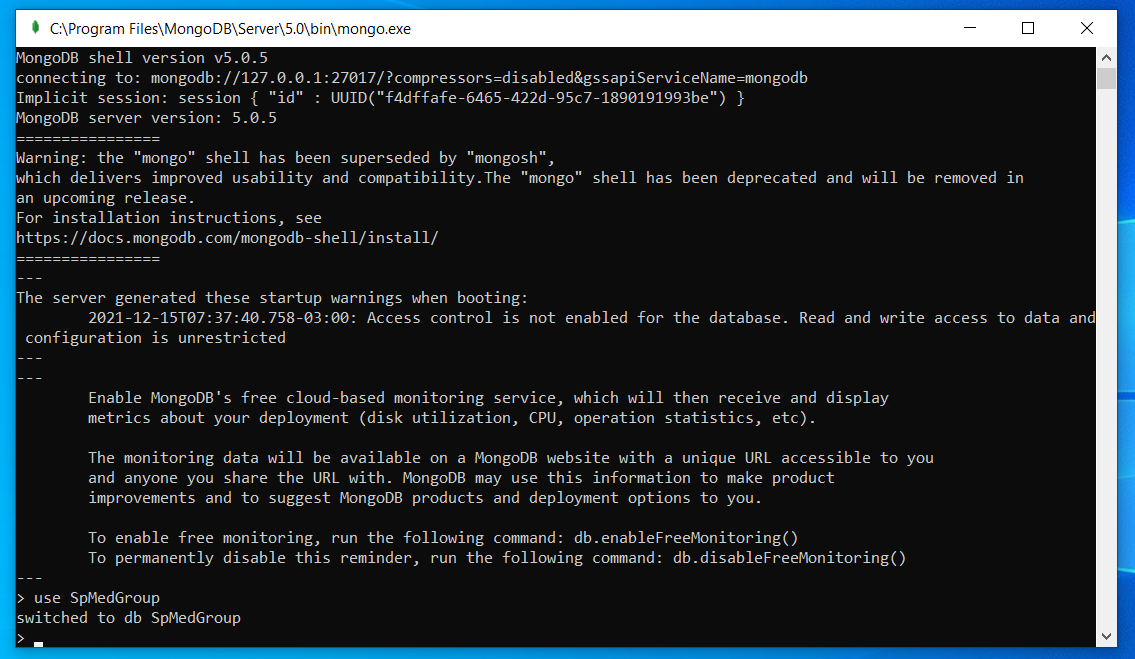
**Banco de Dados(NoSQL)**

1°Passo:



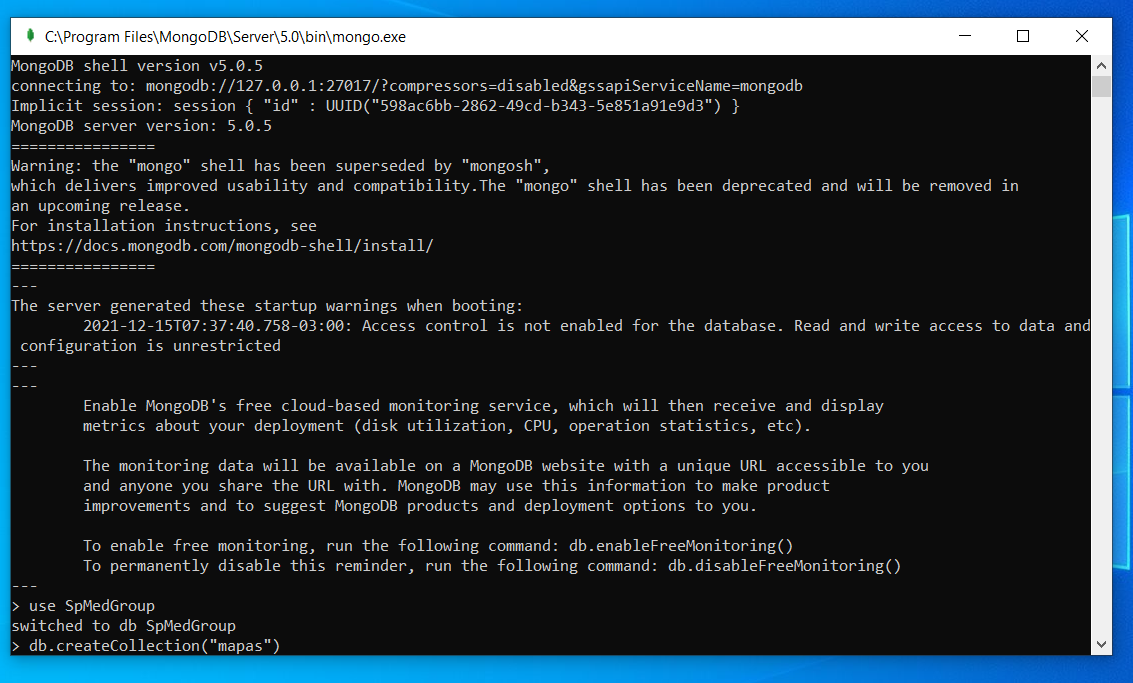
Necessário a instalação do MongoDB e MongoDB Community Server. Após instalado abra o mongo.exe na pasta Program Files\MongoDB\Server\5.0\bin.

2° Passo:



Depois de aberto o terminal, digite use SpMedGroup no terminal para criar o banco de dados.

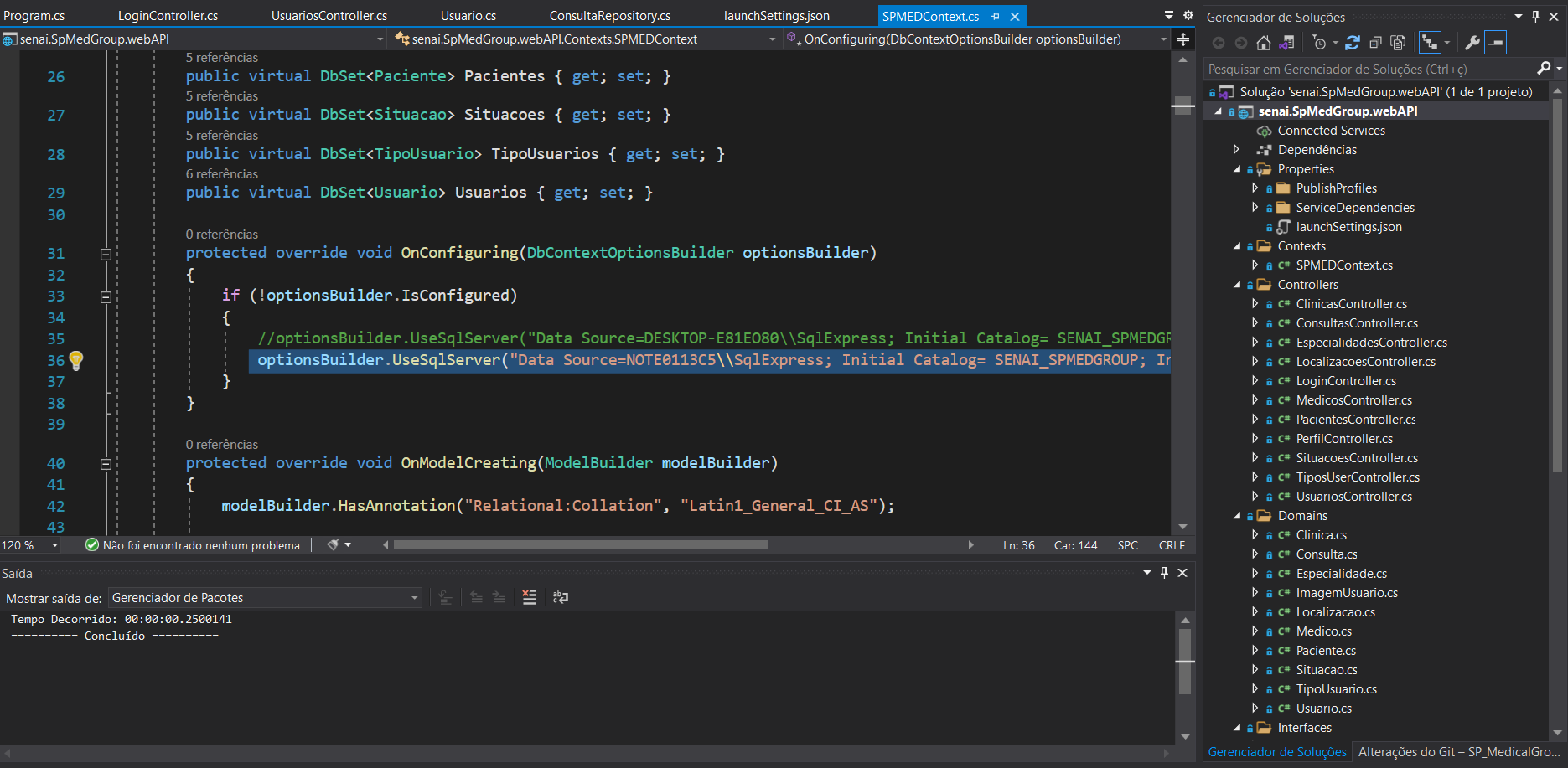
3° Passo:



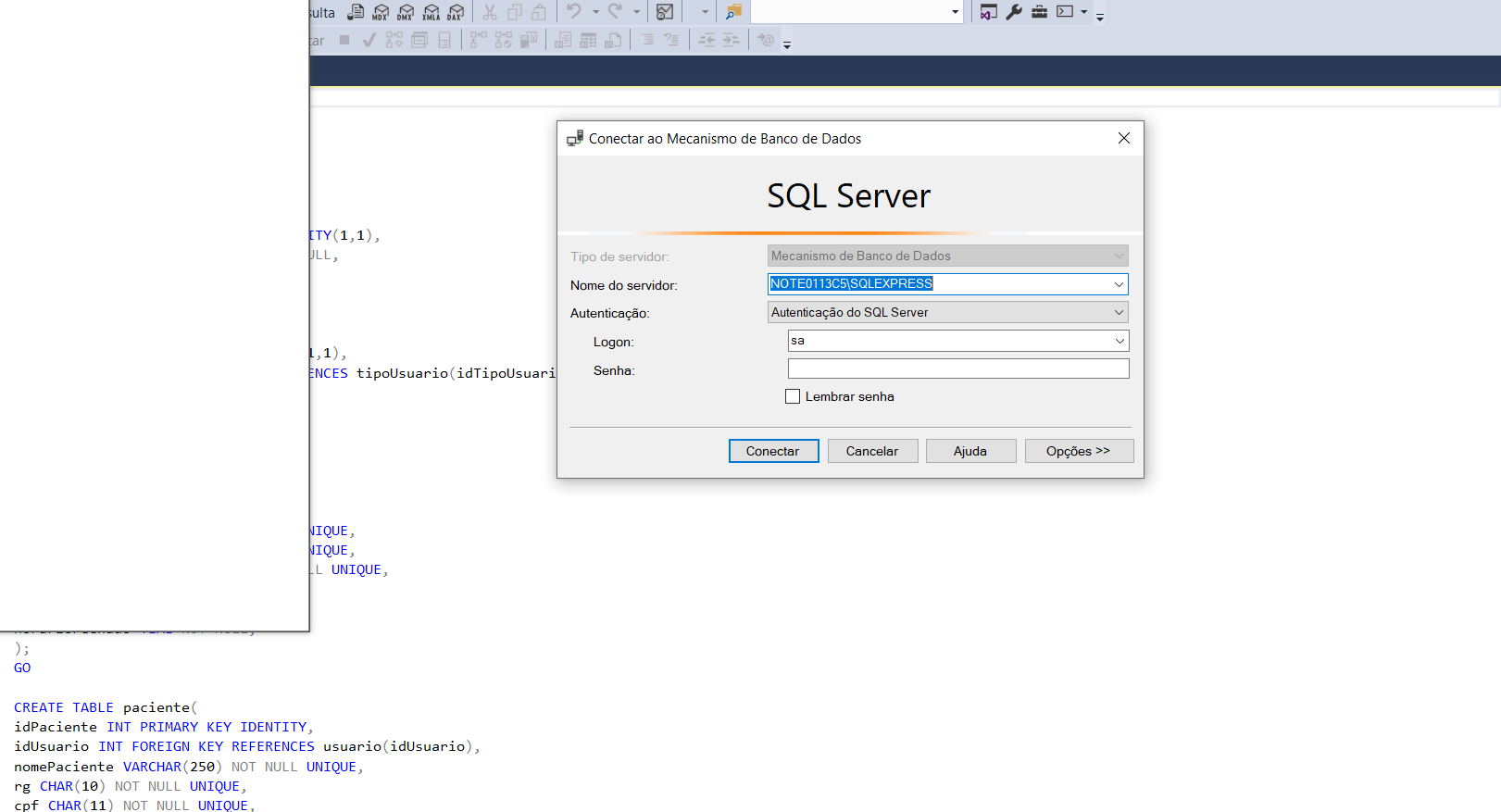
Digite o comando db.createCollection(“Mapas”), para criar de uma coleção no banco.

**API**

1°Passo:

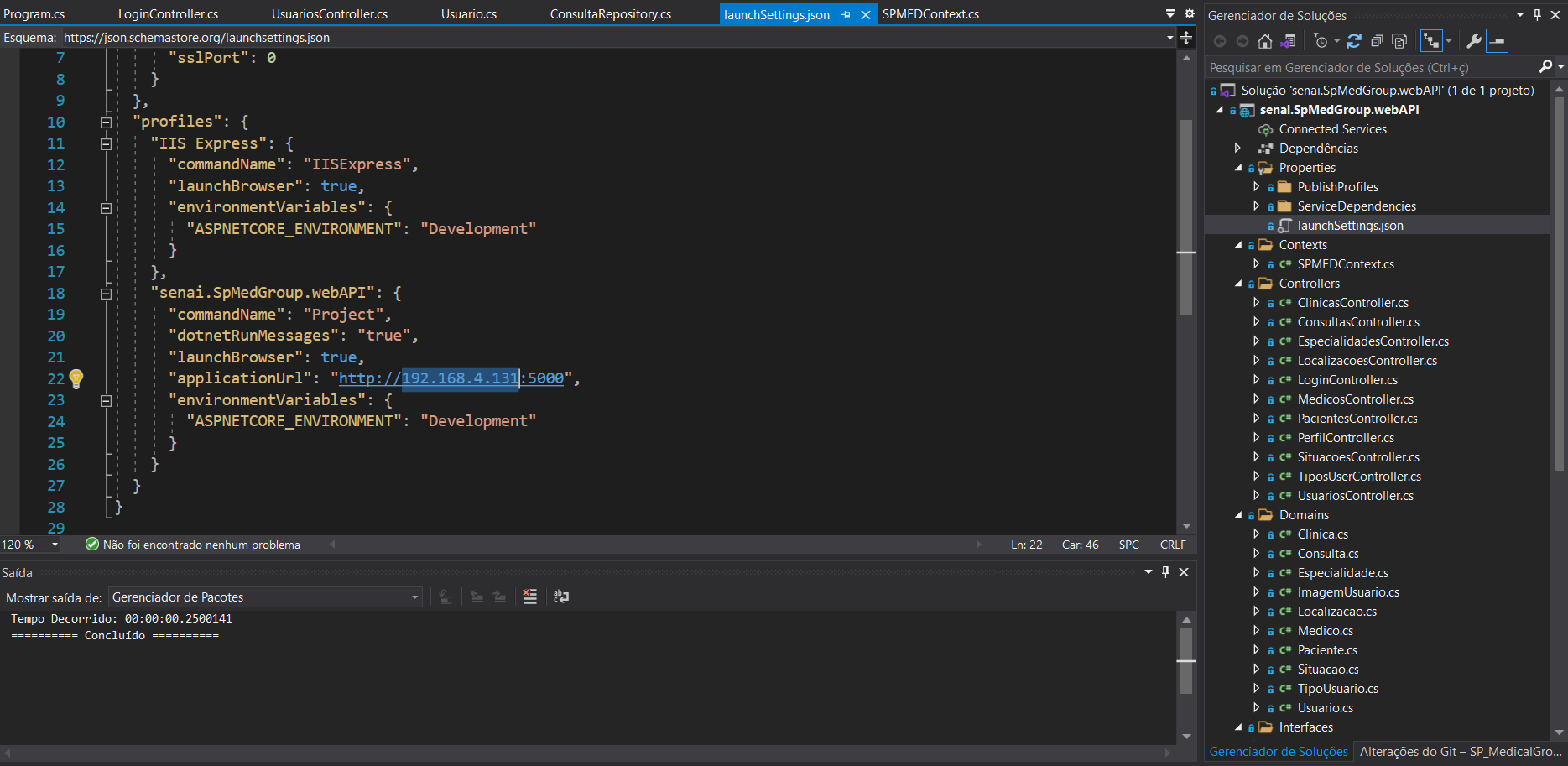


É necessário se ter instalado Microsot Visual Studio Community. Depois de instalado e configurado, abra o arquivo senai.SpMedGroup.webAPI.sln que está na pasta Back-end\senai.SpMedGroup.webAPI. No Visual Studio abra o arquivo SPMEDcontext.cs localizado na pasta contexts e mude o Data Source na optionsBuilder.

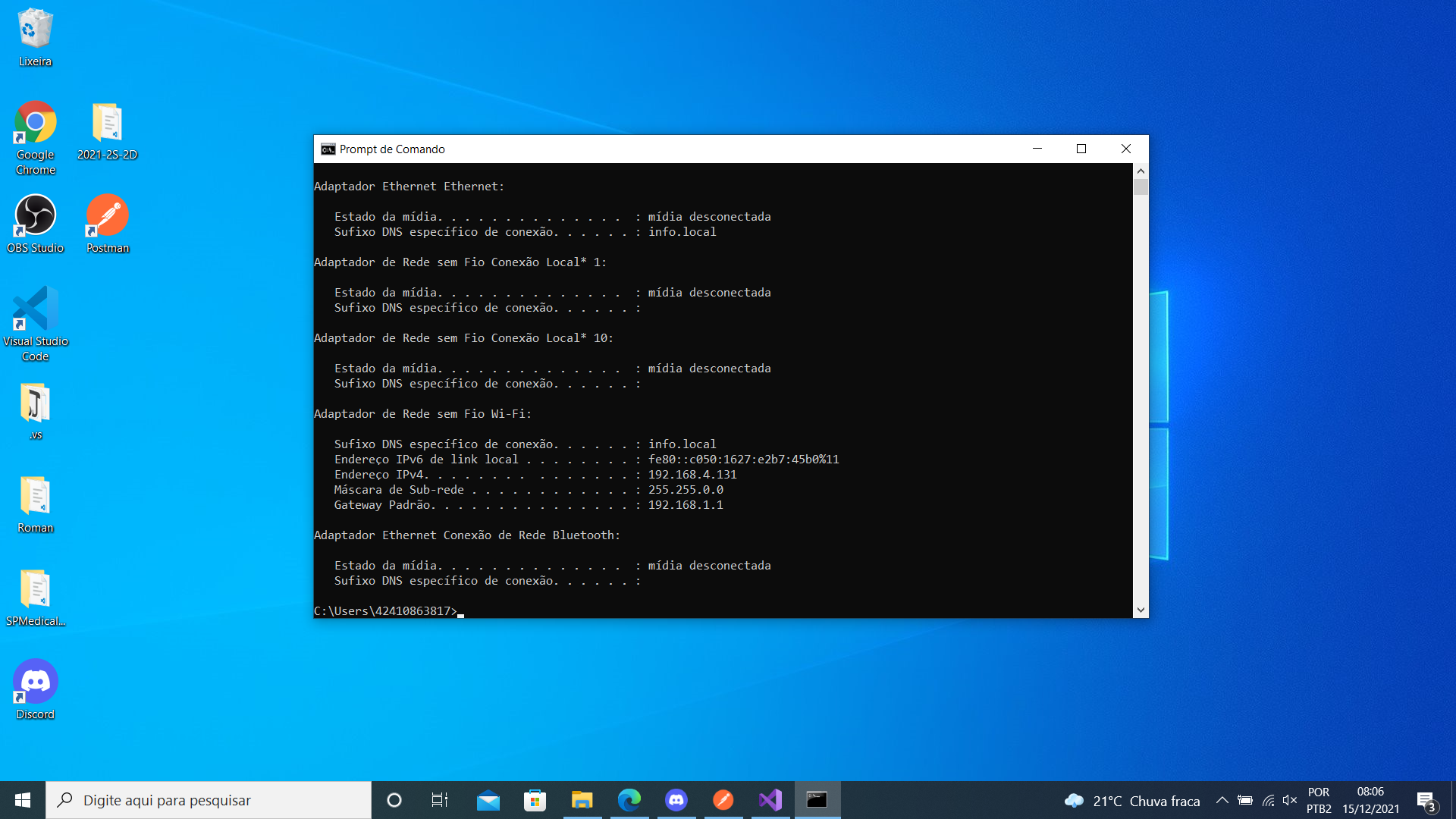


É possível ver qual é o Data Source no SSMS, assim que se abre o aplicativo há uma tela para se conecta, o nome do servidor é o Data Source. Não esqueça de colocar duas barras no Visual Studio antes do SQLEXPRESS, pois no SSMS se há apenas uma barra.

2°Passo:

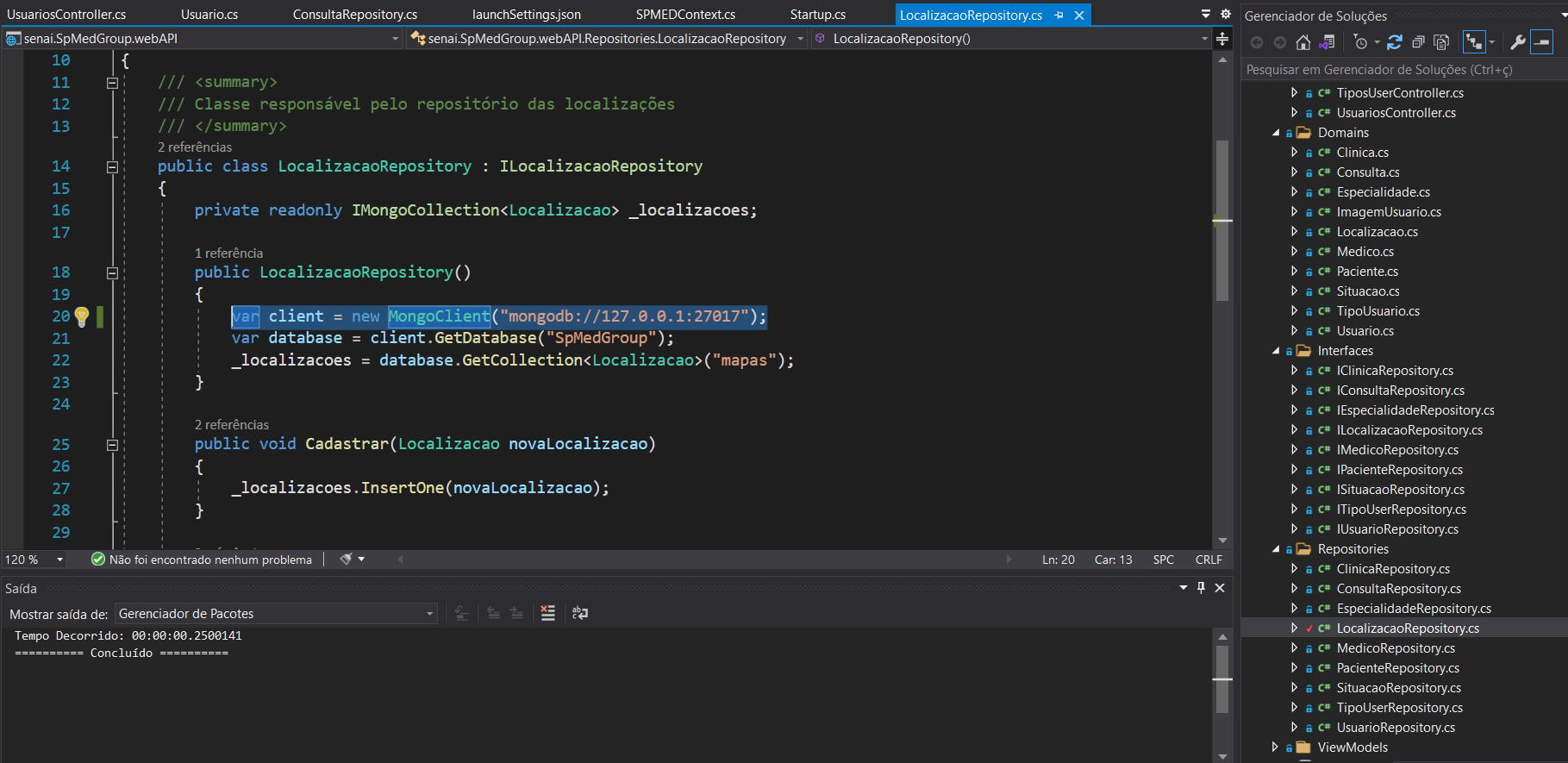


Abra o arquivo launchSettings na pasta Properties, na linha 22 mude o valor depois do http://, sendo que esse valor é IP da sua máquina.

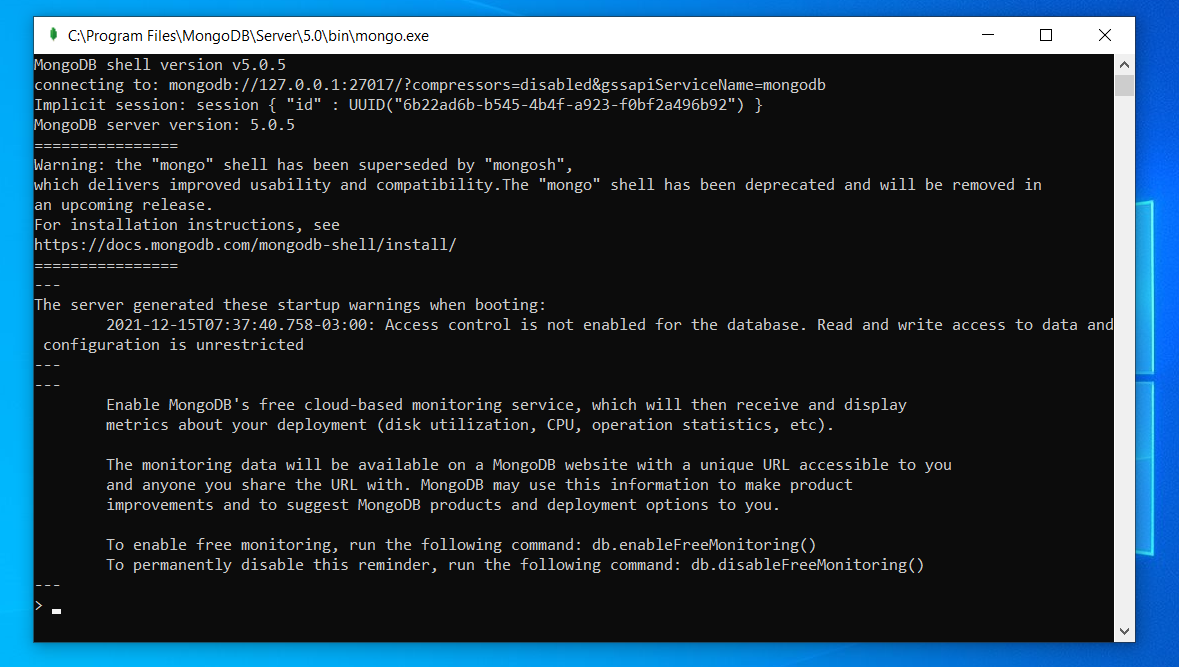


Para verificar o Ip abra o prompt de comando e digite ipconfig, o Ipv4 será o valor utilizado no lauchSettings.

3° Passo:



Por fim verifique se o servidor do mongodb, database e a coleção estão corretas e iguais a sua na linha 20 a 23 do arquivo LocalizacaoRepository na pasta Repositories.



É possível verificar o servidor no terminal do MongoDB na segunda linha.

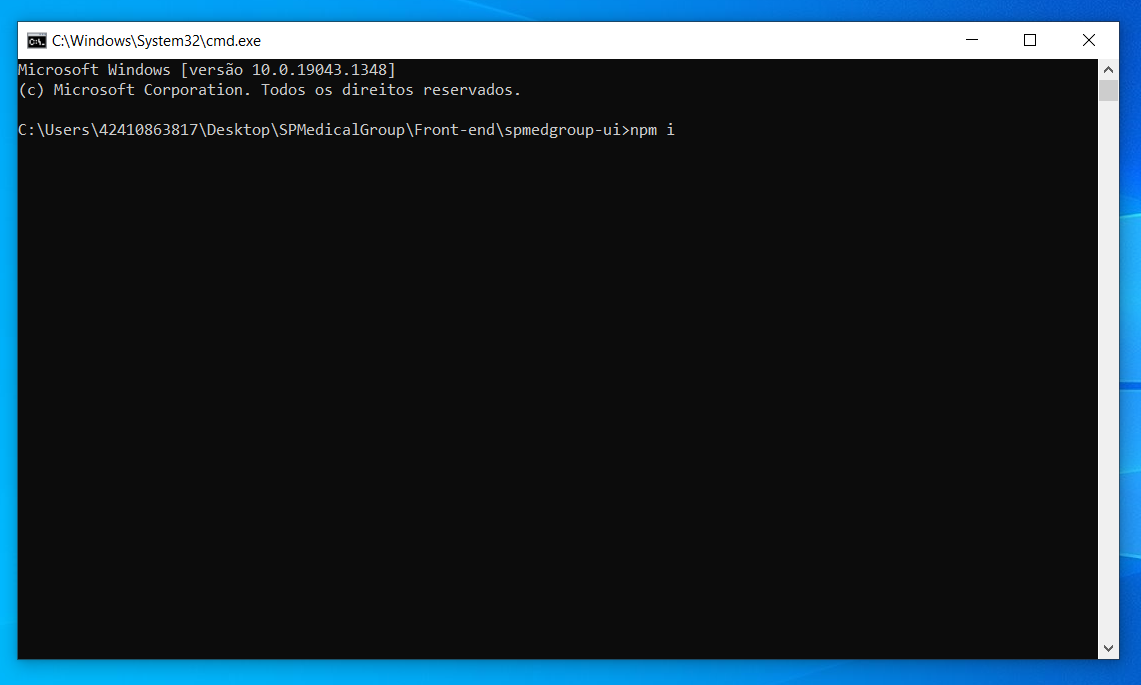
**Front-End**

1°Passo:



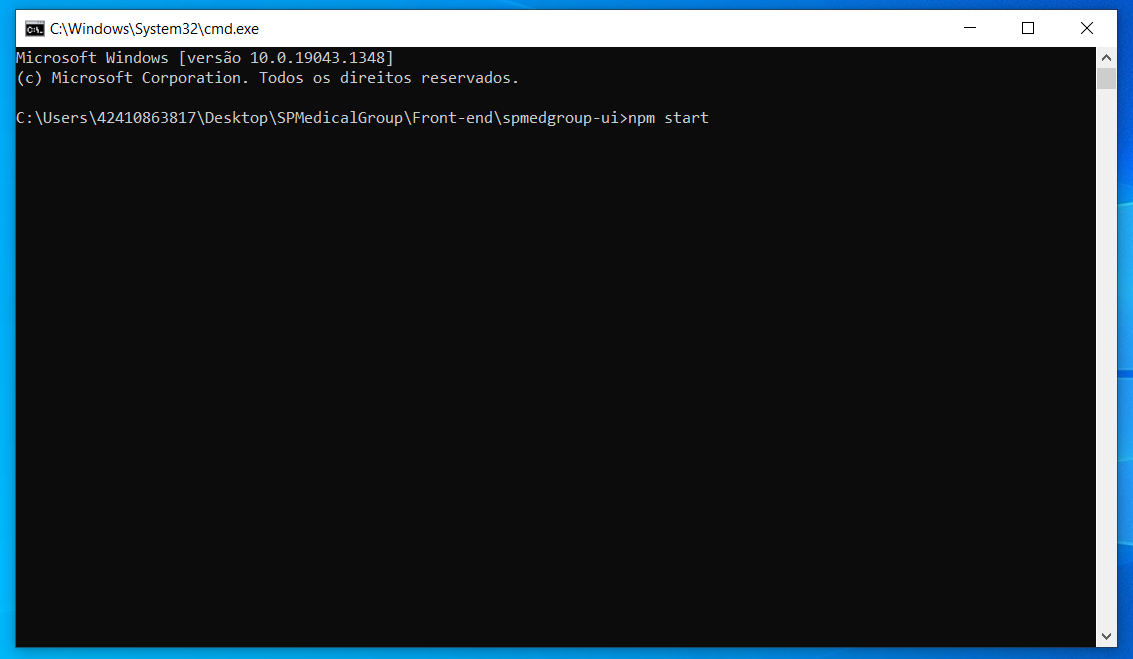
Abra a pasta SPMEDGROUP-UI no front-end e logo após abra o login.js, localizado no spmedgroup-ui\src\pages\login. Depois disso mudei mude a coloque o IPda sua máquina igual na API. É preciso mudar o IP em todas as requisições que queira testar nas outras páginas também.

2°Passo:

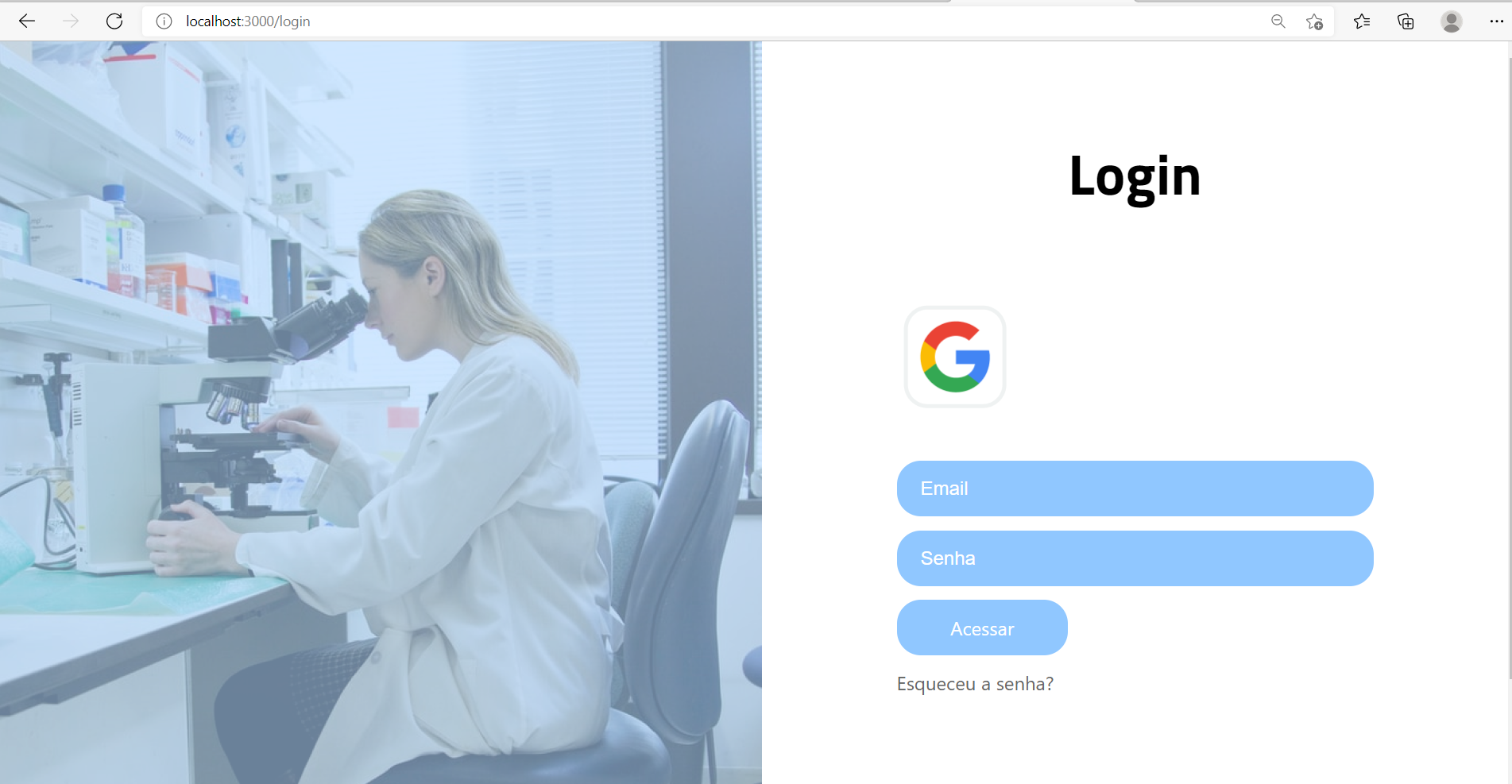


Abra o prompt de comando na pasta ou o terminal do Visual Studio e digite npm i para instalação dos pacotes necessários.

3°Passo:



Digite npm start logo após a conclusão da instalação dos pacotes, o projeto será iniciado!



# Sistema Mobile

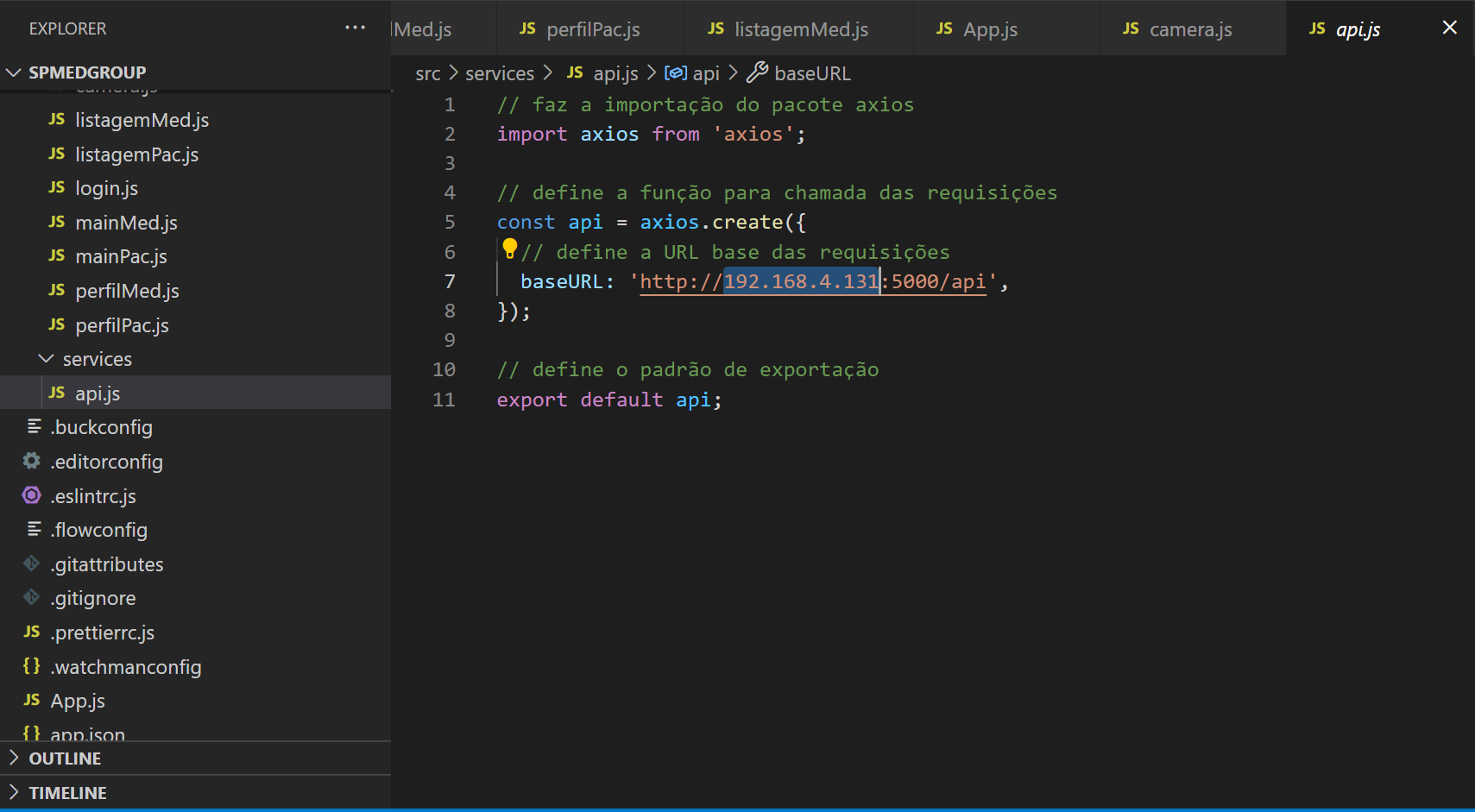
### Perfis de usuário:

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

### Funcionalidades:

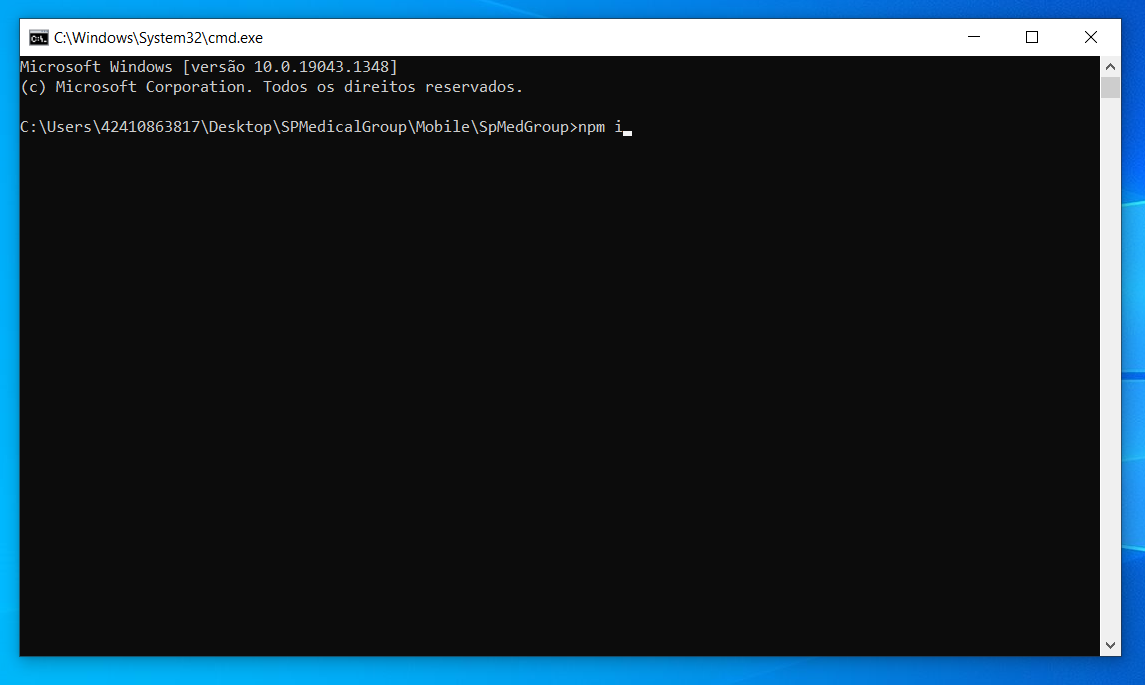
1. Qualquer usuário autenticado poderá ver todos as clínicas;
2. Somente o **administrador** poderá agendar a consulta para o paciente;
3. O usuário **paciente** poderá ver as consultas associados a ele;
4. O usuário **médico** poderá ver as consultas associados a ele;

1°Passo:



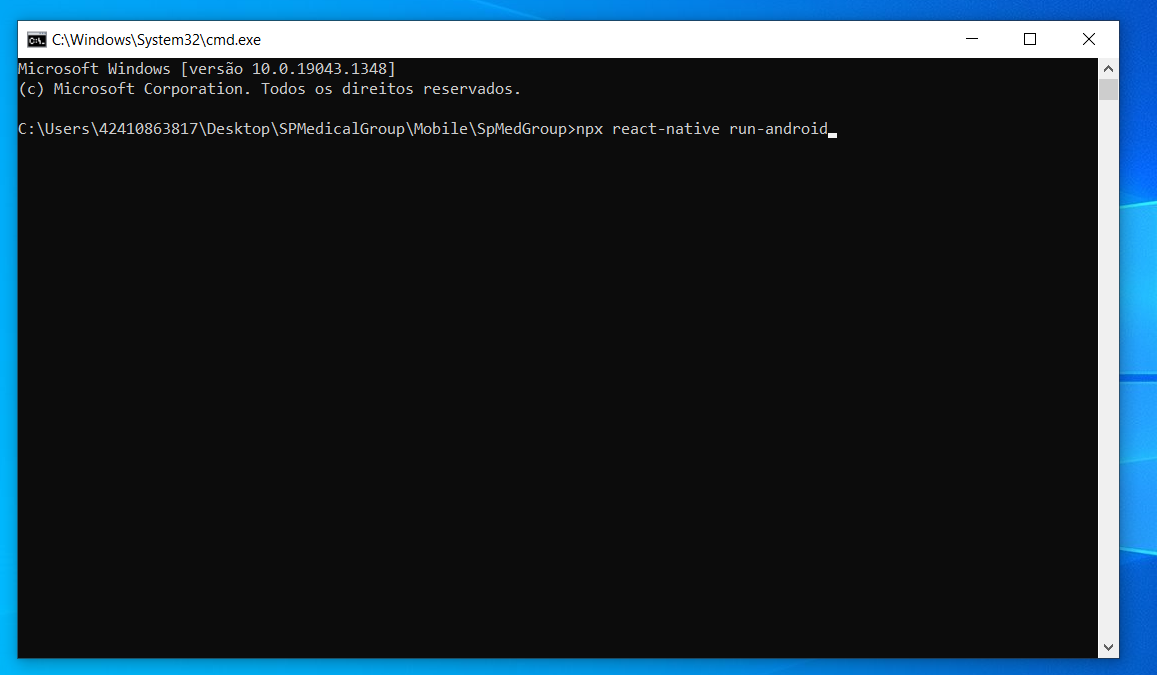
Abra o arquivo api.js localizado no SPMedicalGroup\Mobile\SpMedGroup\src\services e coloque o IP da sua máquina. Caso você não saiba o IP leia o 2°passo da API no Sistema Web explicando como é possível ver o IP.

2°Passo:



Abra o prompt de comando na pasta e digite npm i para instalação dos pacotes necessários.

3°Passo:



Digite npx react-native run-android logo após a conclusão da instalação dos pacotes, o projeto será iniciado!

