

Documentação

Sumário

1.	Resumo	3
2.	Descrição do projeto	3
3.	Banco de dados relacional	3
4.	Modelagem de dados	4
Modelo Conceitual.....		4
Modelo Lógico		5
Modelo Físico.....		5
Cronograma		6
Trello		7
5.	Back-End	8
Funcionalidades		14
6.	Sistema Web	14
7.	Sistema Mobile	16
8.	NoSQL	19

1. Resumo

Este é o documento que contém o que foi feito no projeto SP Medical Group e uma breve descrição para cada item.

2. Descrição do projeto

Fernando Strada solicitou ao desenvolvedor que criasse um sistema integrado web/mobile para a nova clínica médica chamada SP Medical Group, empresa de pequeno porte da região paulista do estado de São Paulo, atualmente se encontra nos moldes de pequeno porte e possuem uma pequena equipe de médicos especializados em algumas áreas. Com o iminente sucesso da clínica a transição das planilhas manuais para um sistema de software completo foi necessária.

Todas as modelagens e scripts para um banco de dados foram criados e estão disponibilizados para acesso assim como a solução API.

Projeto desenvolvido por Gustavo Borges de Souza

3. Banco de dados relacional

Bancos de dados são importantes para organizarmos e armazenarmos informações de maneira organizada e centralizadas, assim facilitando o acesso aos dados e tornando possível adquirir informações através de relacionamentos entre entidades/tabelas, quando houver.

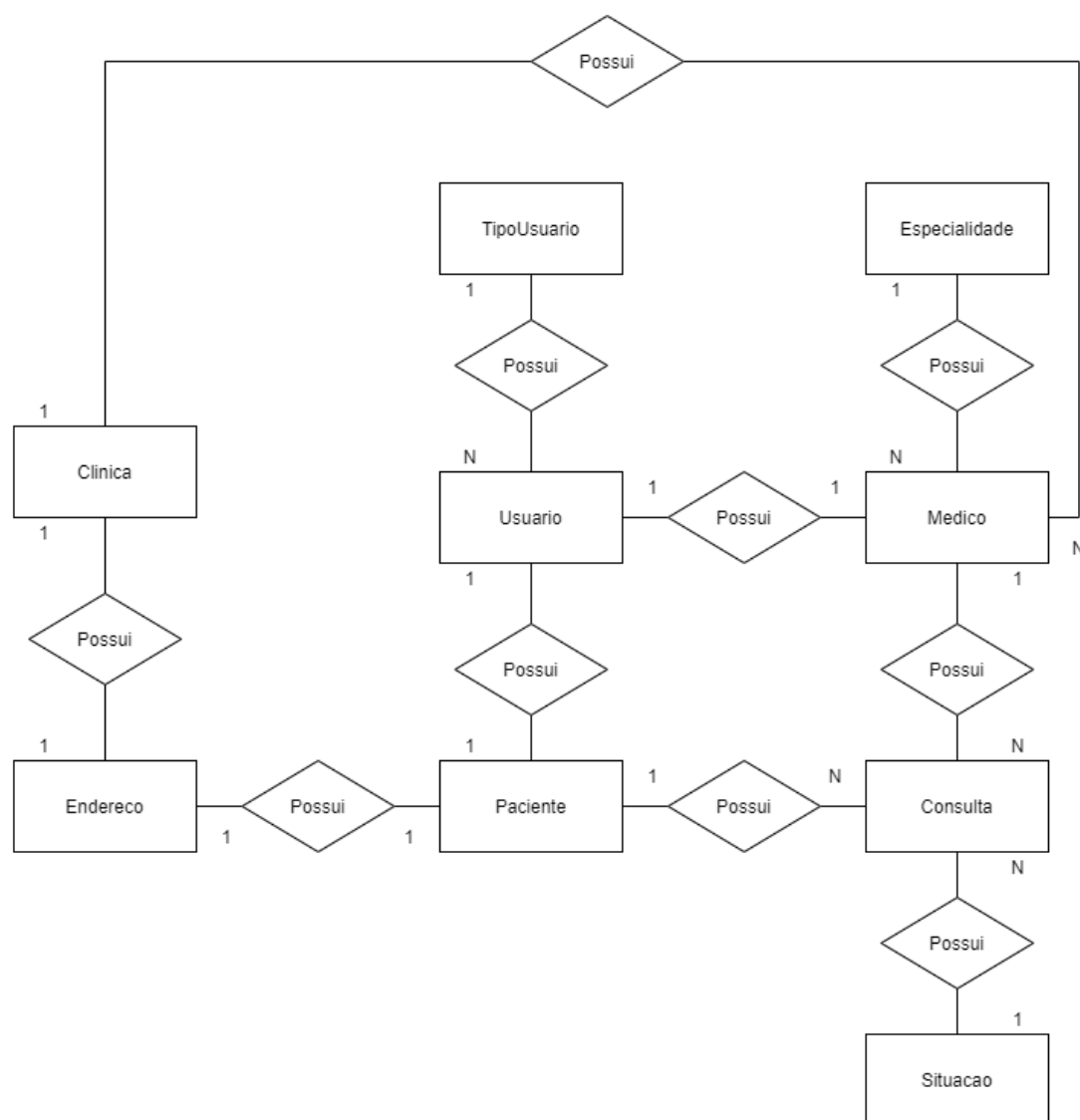
Nos casos em que estes relacionamentos existem, temos bancos de dados relacionais. Nestes a modelagem é feita de uma forma que estes tenham tabelas interligadas entre si, cada registro tem seus campos e é acompanhado por uma chave primária, que pode ser usada como estrangeira em outra entidade, formando a dita relação entre tabelas,

que se bem feita, além de evitar anomalias, deixa o banco sem redundância e menos lento.

4. Modelagem de dados

A modelagem de dados é uma etapa de extrema importância para projetos de desenvolvimento de softwares, em especial na área de bancos de dados. Nela temos tipos de relações (Cardinalidade), as entidades presentes e como o próprio nome sugere, um modelo do banco, seja esse qualquer um dos três possíveis, conceitual, lógico e físico.

Modelo Conceitual



Acima temos um diagrama que mostra a modelagem conceitual para um banco de dados, é a modelagem mais visual e simples de compreender, geralmente é a mostrada ao cliente. Abaixo as cardinalidades e relações explicadas:

Clínica: Possui vários médicos e um endereço, 1:N e 1:1

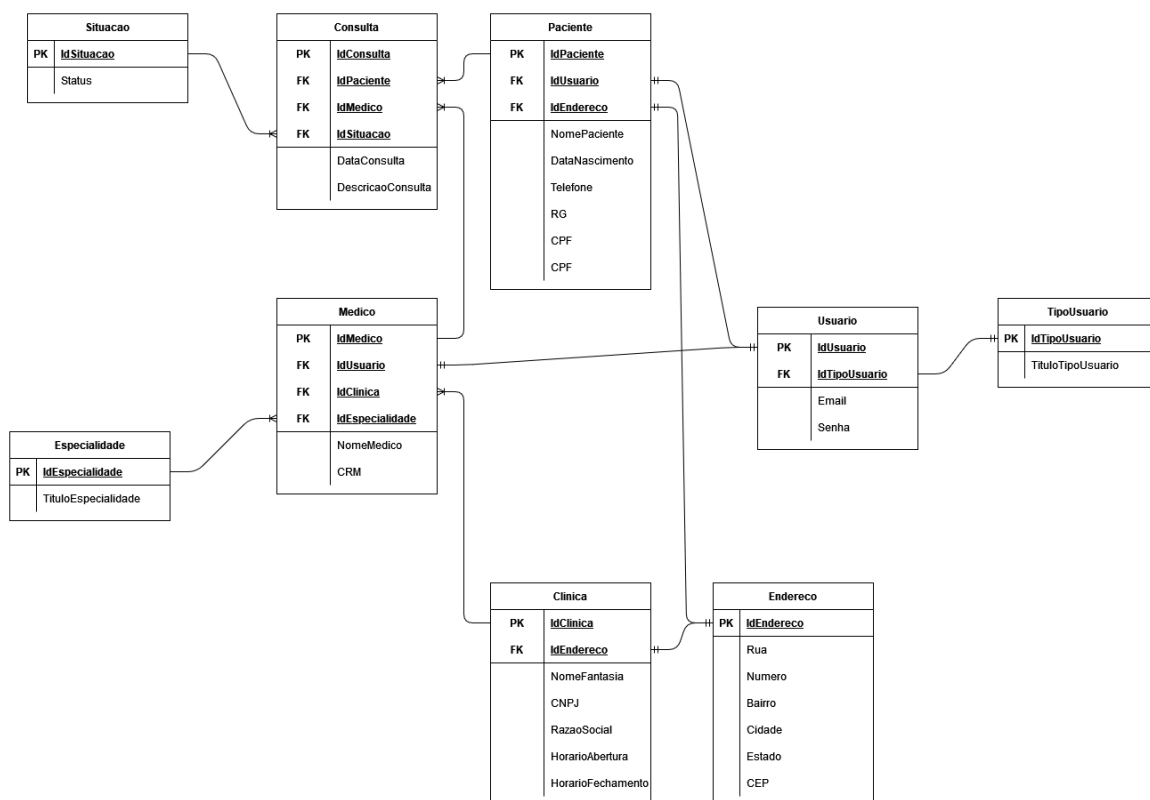
Usuários: Possuem Tipo, N:1

Paciente: Possui um usuário, um endereço e várias consultas, 1:1, 1:1 e 1:N

Médico: Possui um usuário, uma especialidade e várias consultas, 1:1, 1:1 e 1:N

Consulta: Possui uma situação, N:1

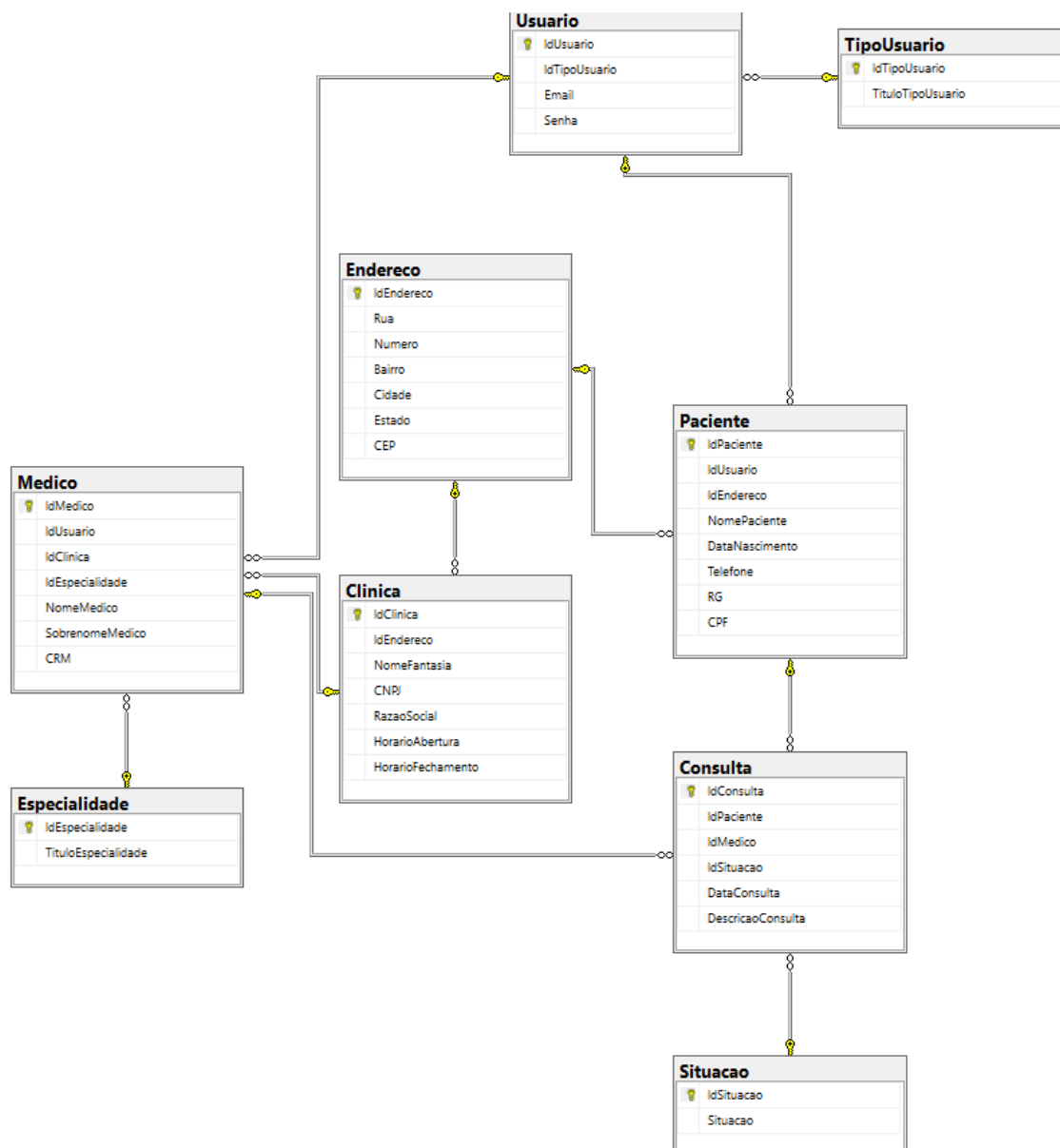
Modelo Lógico



Agora temos a modelagem lógica que adequa o modelo conceitual para a estrutura de um banco de dados, este já não é mostrado ao cliente e é visto somente pela equipe de desenvolvedores.

Neste tipo de modelagem já é possível ver todos os campos das tabelas/entidades, as chaves primárias (PK) e estrangeiras (FK) e quando necessário, a tabela de relação. A Consulta poderia ser considerada uma tabela de relação se não possuísse campos diferentes de PK e FK.

Modelo Físico



Enfim, o modelo físico. Este é a implementação do banco de dados, seja pela representação física no teste de mesa em planilhas, seja pela própria construção do banco. Nestes é possível observar todas as entidades, campos. Nos testes de mesa, em Excel, por exemplo, também podemos ver os registros e atributos.

Cronograma

Dia	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Organização/Trello	X			X				x		x				x			
Modelo Conceitual	X																
Modelo Lógico	X																
Modelo Físico	X																

Script DDL		X														
Script DML		X														
Script DQL			X													
Documentação			X		X						X				X	
Hospedagem				X	X											
Criação da solução			X													
Documentação Swagger				X												
Autenticação				X												
Implementação dos métodos				X	X	X										
Testagem com Postman				X	X	X										
Criação layouts de baixa fidelidade							X									
Criação layouts de alta fidelidade								X	X							
Criação da aplicação web									X							
Página home									X							
Página login										X						
Página Consultas											X	X				
Implementação funcionalidades										X	X	X				
Criação da aplicação móvel													X			
Tela login													X	X		
Navegação mobile													X	X		
Tela Consultas															X	
Tela Perfil															X	
Ícone e splashscreen																X

Trello

<https://trello.com/b/j02qLOIV/sp-medical-group>

5. Back-End

A parte do **backend** ou servidor do projeto foi desenvolvido utilizando como **IDE** o Visual Studio versão Community, criada uma **API** integrada em **repository pattern** usando abordagem **database first** com o **ORM** Entity Framework Core. Para maior segurança foram utilizados o estilo de arquitetura **REST** e os **JWTs** para fazer as requisições no protocolo **HTTP**, com respostas em **JSON**, para que assim diversos dispositivos possam acessar o sistema sem preocupação de linguagem.

Backend = Servidor do sistema, processa os dados e roda por trás da interface de uma aplicação

IDE = Integrated Development Environment - É um ambiente para desenvolvimento integrado, possui ferramentas que apoiam o desenvolvimento e agilizam o processo

API = Application Programming Interface - É uma interface de programação de aplicativos, um conjunto de padrões estabelecidos para acesso a um software. Basicamente o núcleo que concentra a lógica e regras de negócio.

Repository pattern = Padrão de desenvolvimento que tem como base Domains, Interfaces, Repositores e Controllers (Dominios, Interfaces, Repositorios e Controladores), são geralmente desenvolvidos nessa ordem e seguem o padrão de nomenclatura:

Domain: nomeClasse - Representação do banco de dados

Interface: InomeClasseRepository - Contratos, dizem o que deve ser feito

Repository: nomeClasseRepository - Define como vai ser feito a partir da interface

Controller: nomeClassesController(nome da classe no plural) - recebem uma requisição, diretamente ligados ao front-end

ORM = Object Relational Mapping - Mapeador de objeto relacional, framework ou conjunto de códigos que permite a conexão com o banco sem necessidade de escrever códigos como string de conexão.

Database First = Banco de dados primeiro. Abordagem que parte de um banco de dados já existente, com tabelas e dados prontos para uso.

REST = Representational State Transfer – Um conjunto de regras que denomina recursos, métodos e verbos http e response status codes

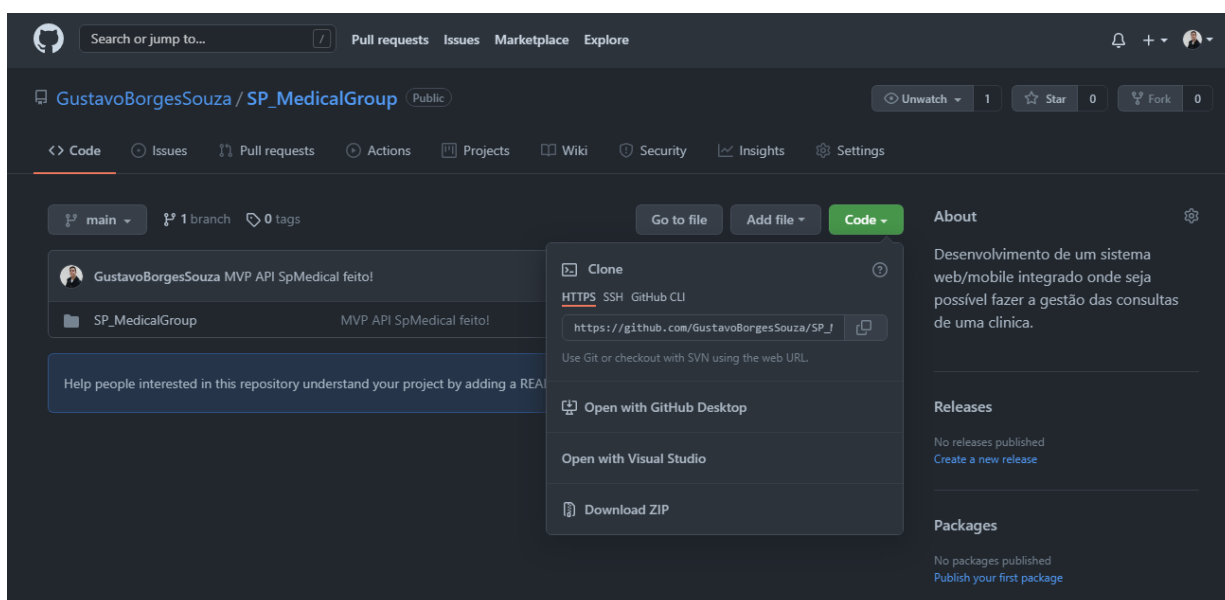
JWT= JSON Web Token - Token online que possui um conjunto de JSON, facilita a comunicação e segurança controlando os acessos através de autenticação e autorização.

HTTP = HyperText Transfer Protocol - Protocolo de Transferência de Hipertexto. Tem verbos e métodos http que são utilizados para definir a ação a ser executada pelo recurso.

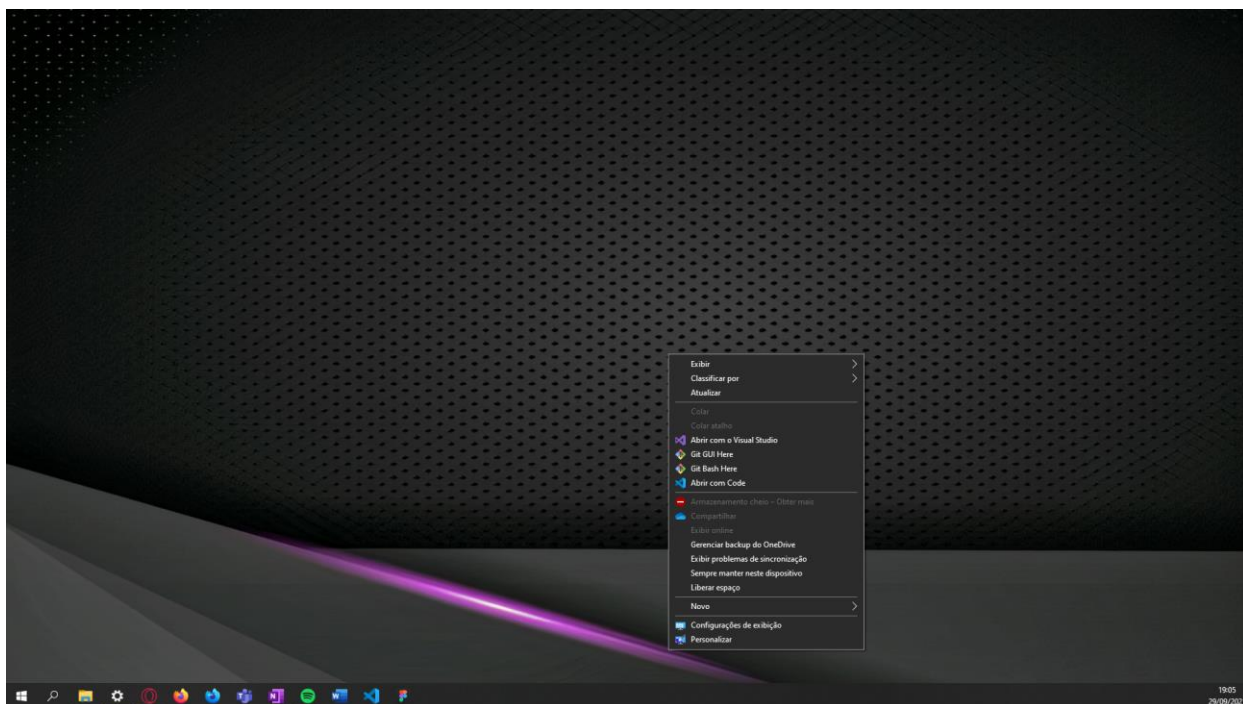
JSON= Javascript Object Notation - Notação de objeto javascript que geralmente é usada para requisições, pode vir como objeto solo ou em array de objetos.

PASSO A PASSO DE COMO EXECUTAR: (OBS: Necessário ter a versão 5.0 do .NET instalada, do git e um editor de consultas de banco como o ssms, recomendável usar o visual studio se possível)

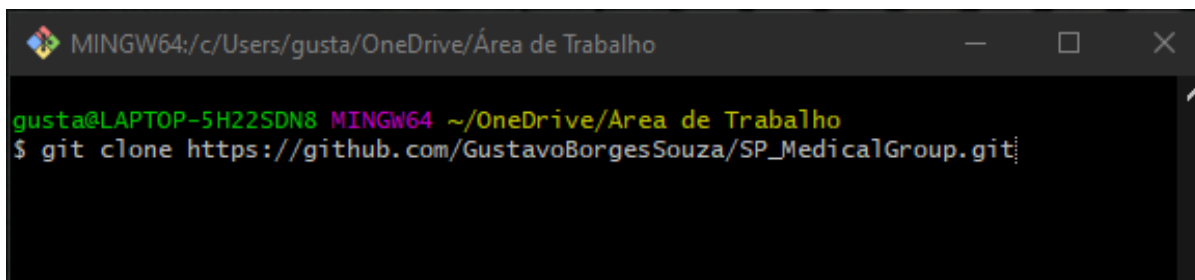
- 1- Acessar https://github.com/GustavoBorgesSouza/SP_MedicalGroup , clique em code e copie o link



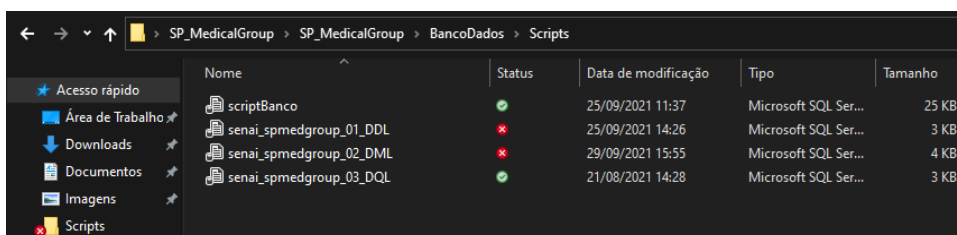
- 2- Clique com o botão direito na área de trabalho e selecione Git Bash here



- 3- Já logado com credenciais do git, digite o comando git clone + o link que foi pego anteriormente, como na imagem abaixo:



- 4- Siga o caminho SP_MedicalGroup -> SP_MedicalGroup -> BancoDados -> Scripts , como na imagem e execute os arquivos DDL, DML e DQL no seu editor de consultas, na respectiva ordem.



- 5- Após isso mude a string de conexão no SPMedContext

Padrão da string de conexão = "Data Source=NomeDoServidor; initial catalog=SPMedGroup_GBm; user Id=Seuld; pwd=senha" ou " Data Source=NomeDoServidor; initial catalog= SPMedGroup_GBm; integrated security=true"

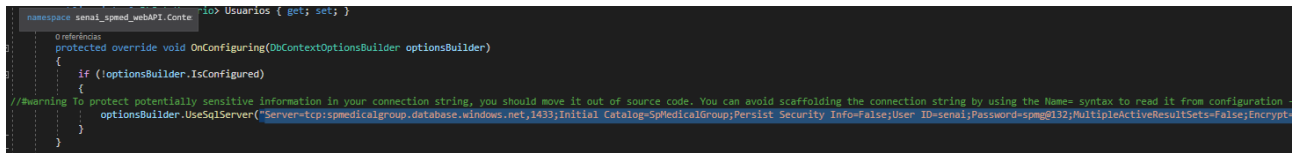
DataSource é o nome do seu servidor do banco

Initial catalog é o nome do banco

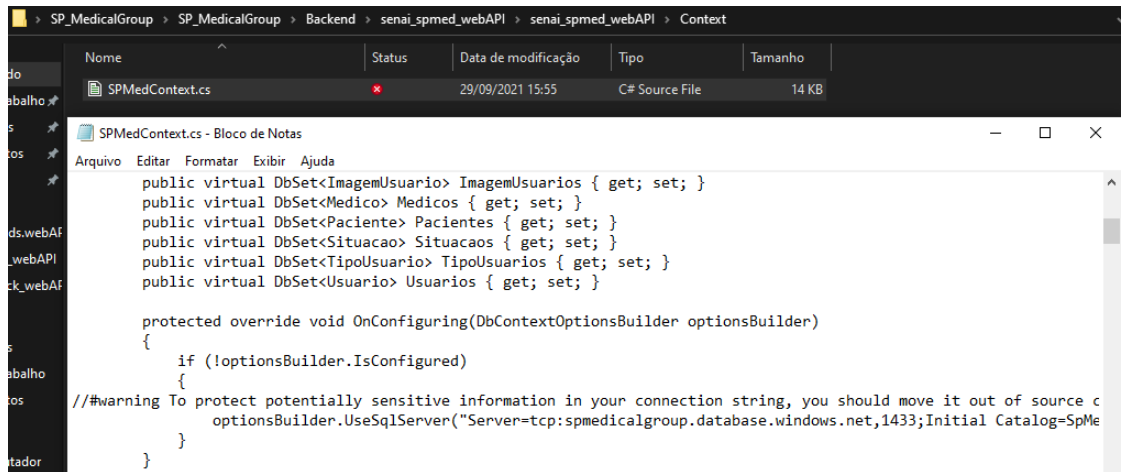
User Id e pwd são suas informações de login no servidor de banco

Integrated security usa as informações do usuário logado no windows

No VS :

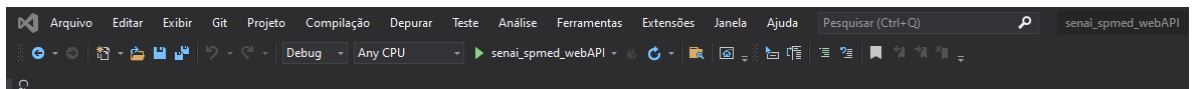


Sem o visual studio é preciso ir até o arquivo SPMedContext.cs e editar pelo bloco de notas:

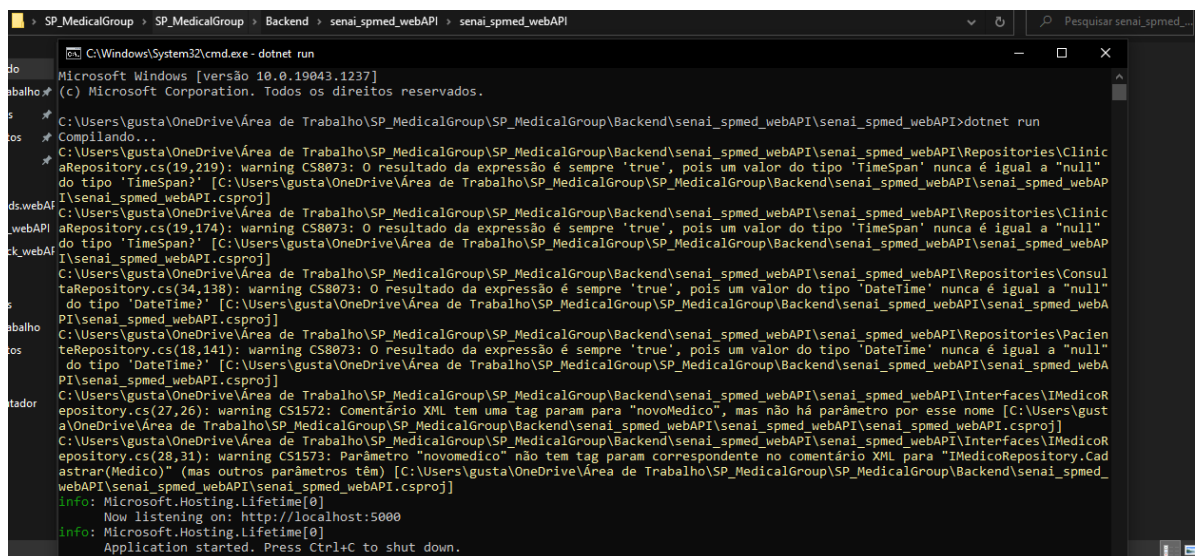


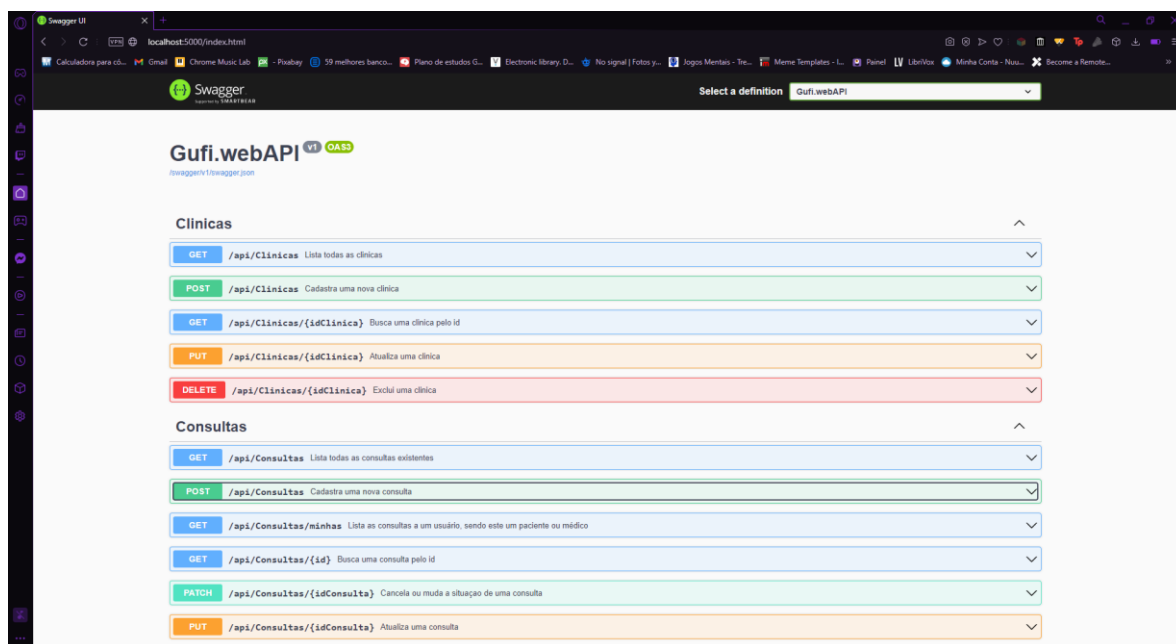
6- Compilar e executar o projeto

No VS basta clicar no botão verde de play:

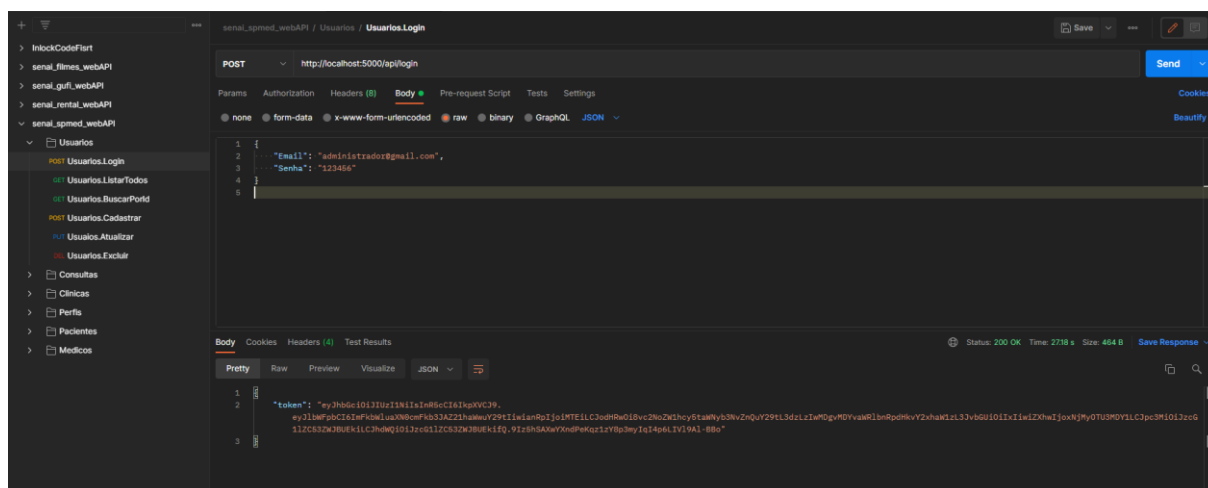


Sem o VS, dentro da pasta da solução digite cmd e no terminal de comando cmd escreva dotnet run, depois copie <http://localhost:5000> no seu navegador e veja a documentação de cada método no swagger

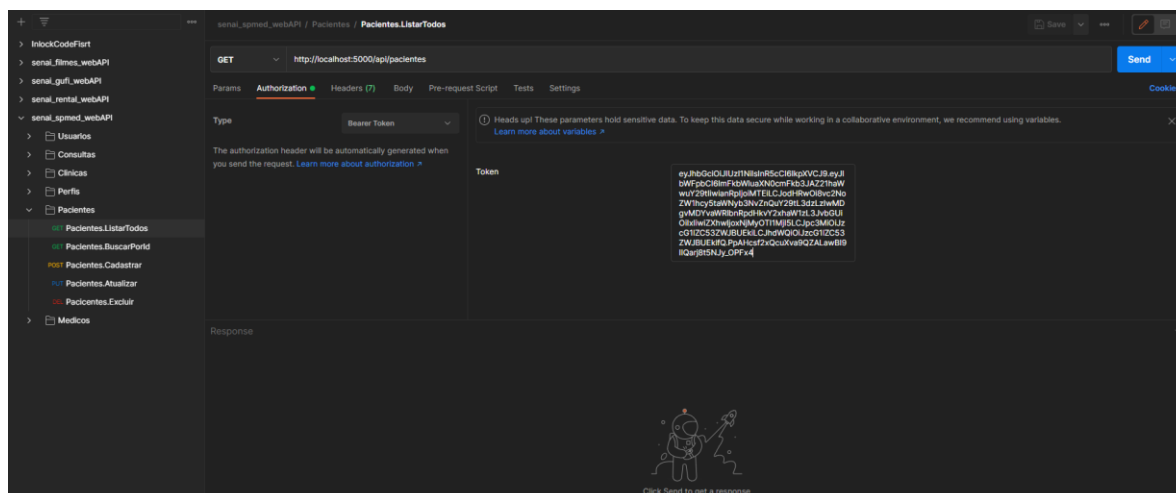




- 7- Abra a coleção exportada no postman, selecione o método de login na pasta de usuários e mande a requisição



- 8- Copie o token, o cole na Authorization/Bearer token da requisição selecionada, faça os ajustes necessários para os atributos se necessário e a envie clicando em send:



Funcionalidades

Funcionalidades que a API atende:

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:

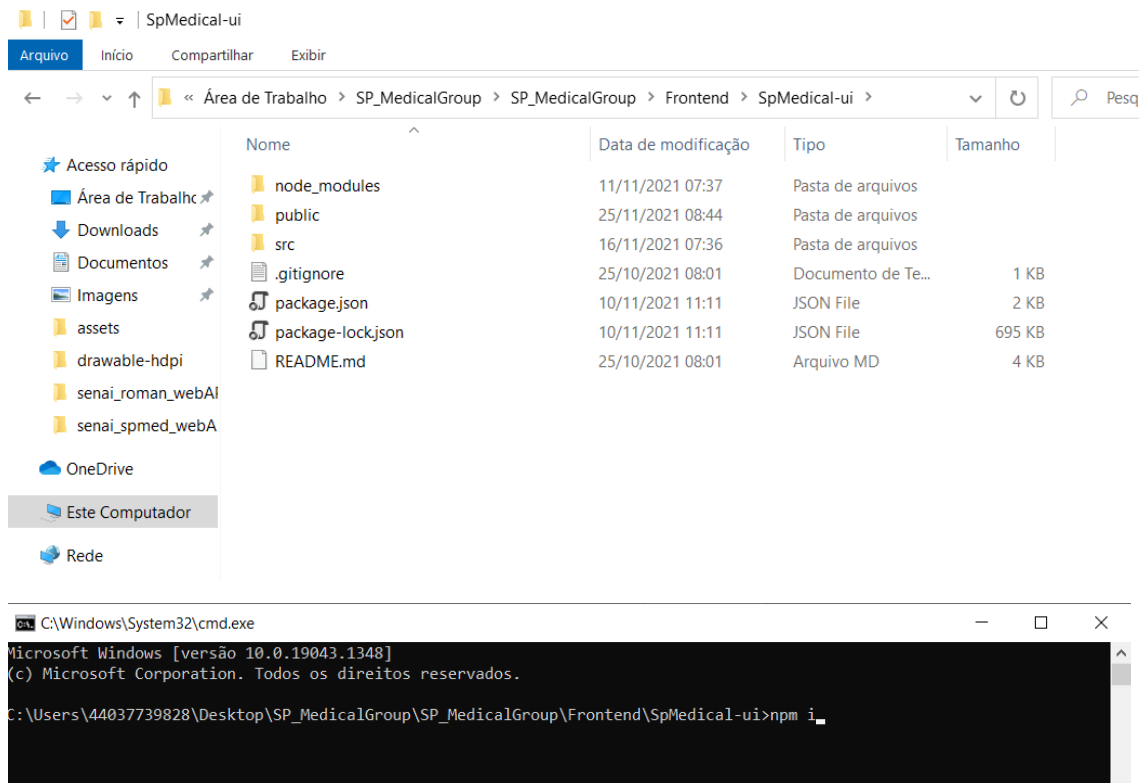
1. O administrador poderá cadastrar, mostrar, buscar, atualizar e excluir qualquer tipo de usuário (administrador, paciente ou médico);
2. O administrador poderá mostrar todas, buscar, atualizar, excluir e cadastrar uma consulta, onde será informado o paciente, data do agendamento e qual médico irá atender 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), além disso pode mostrar todas, buscar uma, atualizar e excluir;
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;
8. Qualquer usuário logado poderá salvar uma foto de perfil;

6. Sistema Web

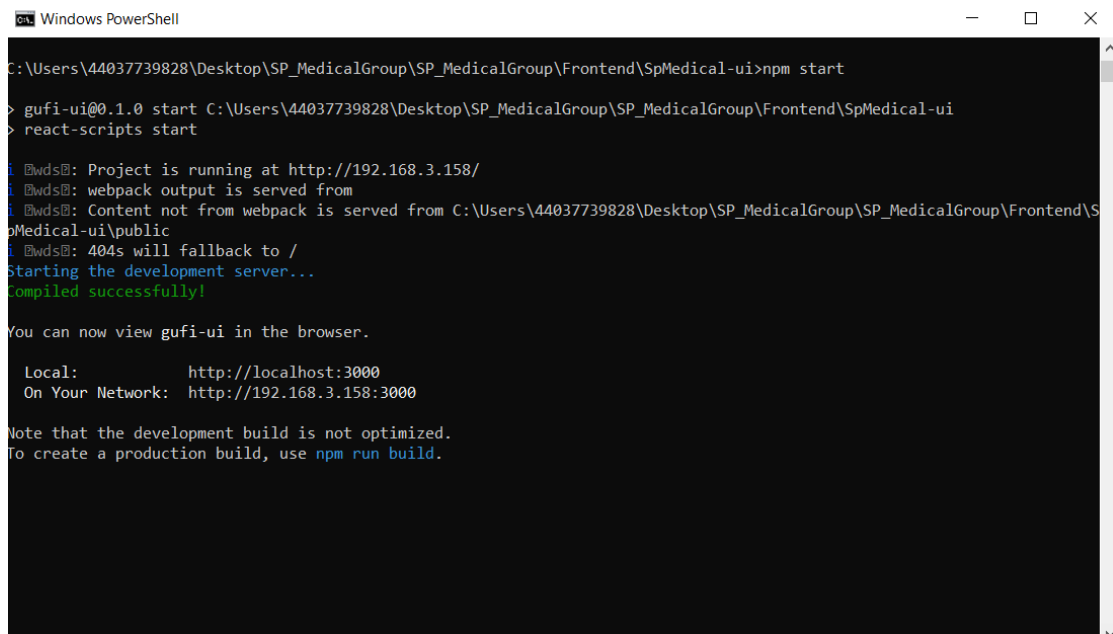
PASSO A PASSO DE COMO EXECUTAR O FRONT END COM REACT(OBS: TENHA A API E BANCO RODANDO NO SEU SISTEMA)

1. Já tendo o repositório clonado, scripts do banco e api rodando, siga o caminho: SP_MedicalGroup -> SP_MedicalGroup -> FrontEnd -> SpMedical-ui, vá ao terminal e

digite o comando “npm i” para fazer a instalação de todas as dependências usadas no aplicativo, sejam outros pacotes ou arquivos necessários para que a aplicação funcione de forma correta.



2. Em seguida, quando a instalação for finalizada digite npm start para rodar os scripts



3. Pronto, agora você pode começar a navegar pelo site



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 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;

7. Sistema Mobile

PASSO A PASSO DE COMO EXECUTAR O MOBILE COM REACT NATIVE(OBS: TENHA A API E BANCO RODANDO NO SEU SISTEMA)

1. Troque a url da aplicação para seu Ip em vez de localhost:5000, para isso vá ao cmd da sua maquina e digite ipconfig e selecione o endereco ipv4:

```
Prompt de Comando

C:\Users\44037739828>ipconfig

Configuração de IP do Windows

Adaptador Ethernet Ethernet:

    Estado da mídia. . . . . : mídia desconectada
    Sufixo DNS específico de conexão. . . . . : info.local

Adaptador de Rede sem Fio Conexão Local* 1:

    Estado da mídia. . . . . : mídia desconectada
    Sufixo DNS específico de conexão. . . . . :

Adaptador de Rede sem Fio Conexão Local* 10:

    Estado da mídia. . . . . : mídia desconectada
    Sufixo DNS específico de conexão. . . . . :

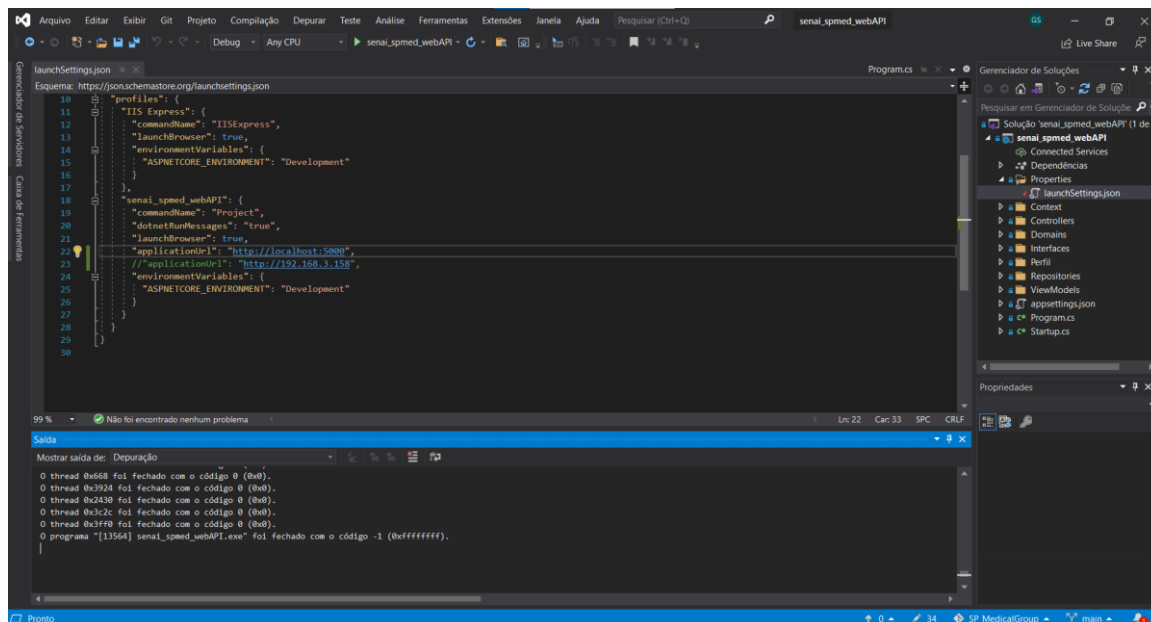
Adaptador de Rede sem Fio Wi-Fi:

    Sufixo DNS específico de conexão. . . . . : info.local
    Endereço IPv6 de link local . . . . . : fe80::1d57:572e:5db3:fc7d%18
    Endereço IPv4. . . . . : 192.168.3.158
    Máscara de Sub-rede . . . . . : 255.255.0.0
    Gateway Padrão. . . . . : 192.168.1.1

C:\Users\44037739828>
```

2. Coloque seu ip no applicationURL e comente o do localhost

Antes:

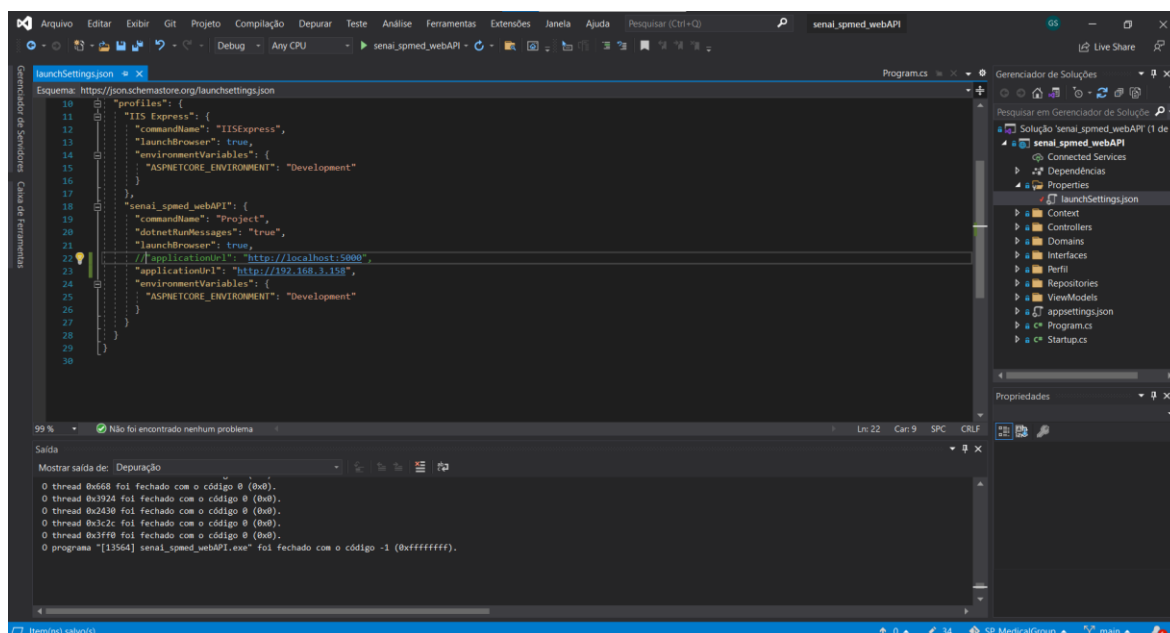


```
launchSettings.json
Esquema: https://json.schemastore.org/launchsettings.json
10  "profiles": {
11    "IIS Express": {
12      "commandName": "IISExpress",
13      "launchBrowser": true,
14      "environmentVariables": {
15        "ASPNETCORE_ENVIRONMENT": "Development"
16      }
17    },
18    "senai_spmed_webAPI": {
19      "commandName": "Project",
20      "dotnetRunMessages": true,
21      "launchBrowser": true,
22      "applicationUrl": "http://localhost:5000",
23      "applicationUri": "http://192.168.3.158",
24      "environmentVariables": {
25        "ASPNETCORE_ENVIRONMENT": "Development"
26      }
27    }
28  }
29
30

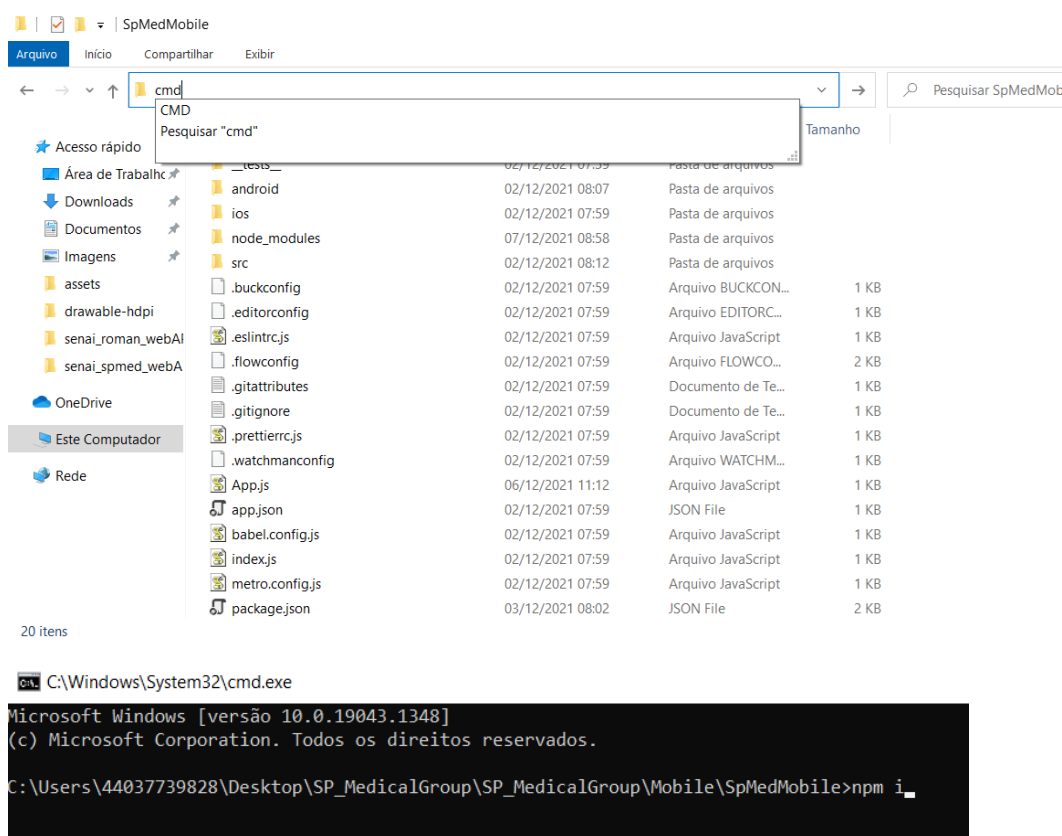
Program.cs
99 % Não foi encontrado nenhum problema
Ln: 22 Car: 33 SPC: CRLF

Mostrar saída de: Depuração
O thread 0x668 foi fechado com o código 0 (0x0).
O thread 0x3924 foi fechado com o código 0 (0x0).
O thread 0x2430 foi fechado com o código 0 (0x0).
O thread 0x3c2c foi fechado com o código 0 (0x0).
O thread 0x3ff0 foi fechado com o código 0 (0x0).
O programa "[13564] senai_spmed_webAPI.exe" foi fechado com o código -1 (0xffffffff).
```

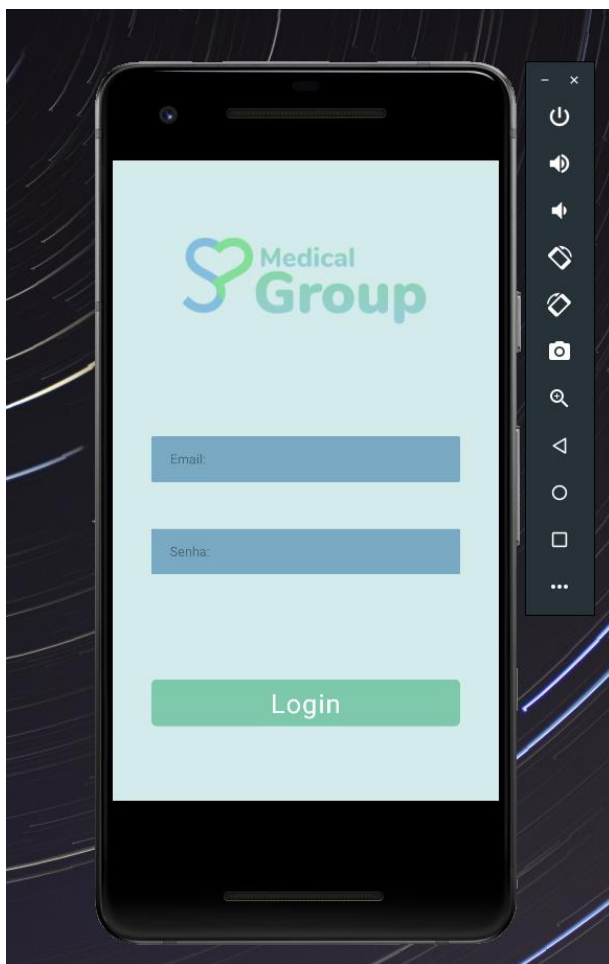
Depois:



3. Rode a API e siga o caminho: SP_MedicalGroup -> SP_MedicalGroup -> BancoDados -> Mobile-> SpMedMobile, abra o cmd e digite npm i:



4. Agora digite npx react-native run-android e use o aplicativo.



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;

8. NoSQL

Com o projeto finalizado, Fernando Strada requisitou a implementação de um sistema que permitisse aplicar data analytics, com isso foi criado um banco de dados não relacional, NoSQL(Not Only SQL). Os dados serão preenchidos com endereços dos pacientes para analisarem onde uma nova clínica pode ser construída e somente os administradores podem ver as informações seguindo a url a seguir no front end: <http://localhost:3000/analise>

