**SP MEDICAL GROUP**

Documentação

Sumário

[1. Resumo 3](#_Toc83811210)

[2. Descrição do projeto 3](#_Toc83811211)

[3. Banco de dados relacional 4](#_Toc83811212)

[4. Modelagem de dados 4](#_Toc83811213)

[Modelo Conceitual 5](#_Toc83811214)

[Modelo Lógico 6](#_Toc83811215)

[Modelo Físico 7](#_Toc83811216)

[Cronograma 8](#_Toc83811217)

[Trello 8](#_Toc83811218)

[5. Back-End 9](#_Toc83811219)

[Funcionalidades 13](#_Toc83811220)

[Sistema Web 13](#_Toc83811221)

# Resumo

O seguinte documento buscará descrever as inúmeras perspectivas que integram o   
sistema de gerenciamento de consultas construído para o acesso facilitado aos dados da clínica fictícia SP Medical Group, desde a elaboração de diferentes tipos de modelagens de dados até a lógica implementada para se enrijecer as relações entre as diversas informações contempladas pela aplicação. Também serão descritos detalhadamente os aspectos compreendidos no back-end da aplicação, tal qual a sua incorporação no layout elaborado, destacando os objetivos quanto a sua navegabilidade e ao aprimoramento da experiência entre o usuário e a função administrativa do sistema desenvolvido.

# Descrição do projeto

A clínica médica chamada SP Medical Group, empresa de pequeno porte que atua no ramo da saúde, necessita de um sistema web/mobile integrado para gerenciar o fluxo de consultas em suas respectivas unidades, facilitando a recuperação dos dados dos prontuários e dos profissionais atuantes na clínica.

Portanto, para atender com diligência as necessidades supracitadas, o sistema em desenvolvimento contará com uma plataforma que atenderá a três grupos distintos, o setor administrativo da empresa, que terá posição arbitrária dentro do sistema, podendo agendar, listar e cadastrar os demais usuários, os médicos, que poderão verificar os detalhes dos seus atendimentos e incrementá-los com descrições, e os pacientes, que também poderão conferir suas respectivas consultas dentro do sistema.

Ademais, para se adequar a possíveis adversidades, o sistema também estará disponível para acesso em dispositivos móveis, facilitando o acesso e a visualização das consultas por parte dos médicos e pacientes. Efetuando a autenticação, ambos os tipos de usuário poderão apenas acessar informações superficiais, revisando os detalhes dos agendamentos a eles atrelados, enquanto as demais funcionalidades do sistema terão suporte apenas na versão web, visando a comodidade dos usuários que experimentam tarefas mais laboriosas com o sistema.

# Banco de dados relacional

Como visivelmente descrito na própria nomenclatura, um banco de dados relacional consiste no estabelecimento de relacionamentos entre os dados organizadamente armazenados de uma determinada aplicação. Normalmente representado através de tabelas, os bancos relacionais são compostos por colunas (campos) especificados com atributos e o tipo de dado que ali será alocado, além de possuir linhas (registros), onde de fato serão definidos os dados. Geralmente, os registros das tabelas são acompanhados por identificadores únicos denominados chaves primárias, que além de permitirem a construção de relações, possibilitam uma verificação rápida dos dados.

Os bancos de dados podem ser manipulados através de um Sistema de Gerenciamento de Banco de Dados (SGBD), programas responsáveis pelo acesso e organização dos dados.

A importância de um banco de dados evidencia-se principalmente no planejamento dos negócios de determinadas corporações, empresas e comércios que almejam a assertividade em suas decisões. Com isso, torna-se imprescindível a apuração dos dados fornecidos consensualmente pelos usuários e clientes para que se atendam as expectativas na prestação de serviços ou vendagem de produtos, priorizando o conforto dos que navegam por uma determinada plataforma. Além do mais, a implementação de um banco de dados corretamente estruturado possibilitará um melhor acompanhamento do funcionamento da empresa ou demais patrimônios, controlando os fluxos existentes e mantendo um relacionamento sincronizado entre os mais diversos setores da empresa, aprimorando a sua performance e reduzindo os riscos de que usuários indesejados acessem dados restritos, garantindo a integridade dos documentos.

# Modelagem de dados

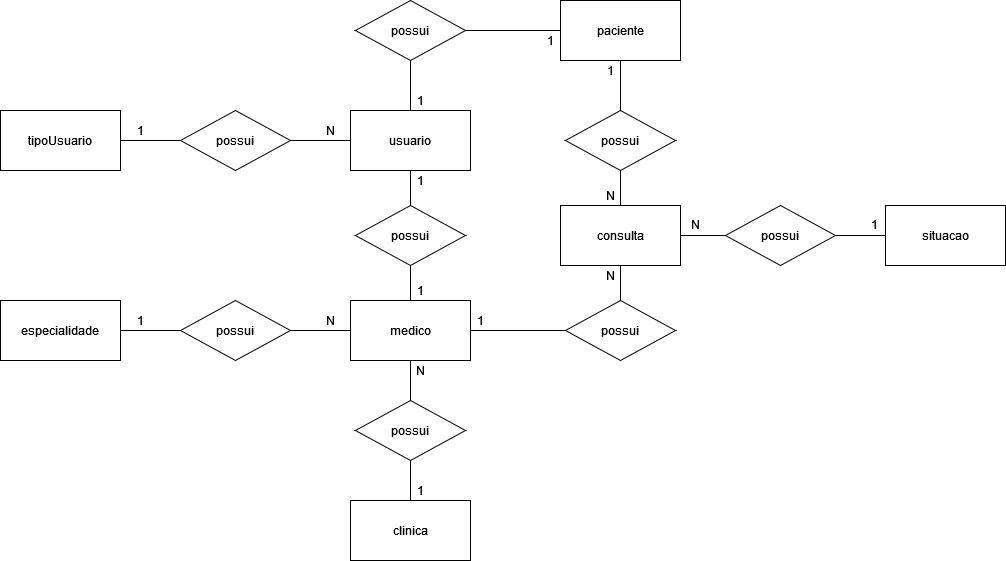
A modelagem de dados é a etapa do desenvolvimento de sistemas encarregada por   
demonstrar como os dados se comportarão dentro de uma aplicação, evidenciando suas   
relações e processos, imprescindível para a manutenção do negócio que se deseja   
implementar com a possibilidade de armazenar e recuperar informações de maneira   
padronizada. Para se compreender a modelagem de dados, há três maneiras de efetuá-la, seja através de um modelo conceitual, de um modelo lógico ou através de um modelo físico, entretanto, ambos partilham desses mesmos elementos básicos:

* **Entidade:** Conjunto de dados sobre determinada parte do sistema. As entidades   
  podem se caracterizar como fracas ou fortes de acordo com seu número de   
  relações.
* **Relacionamento:** Como descrito na própria nomenclatura, as entidades poderão   
  se relacionar entre si e permitem o compartilhamento de dados.

## Modelo Conceitual

Ao que se pode perceber nesta representação gráfica do sistema de armazenamento, toda sua composição é constituída por figuras geométricas simples que estabelecem relacionamentos entre si, responsáveis principalmente por expressar de maneira sucinta a regra de negócio que rege a aplicação, proporcionando uma rápida e fácil absorção de   
informações. Em outras palavras, a interatividade estabelecida nessa espécie de mapeamento contribuirá para o entendimento dos processos, justificando a necessidade, o comportamento e a aparência objetivada para seu estágio final.

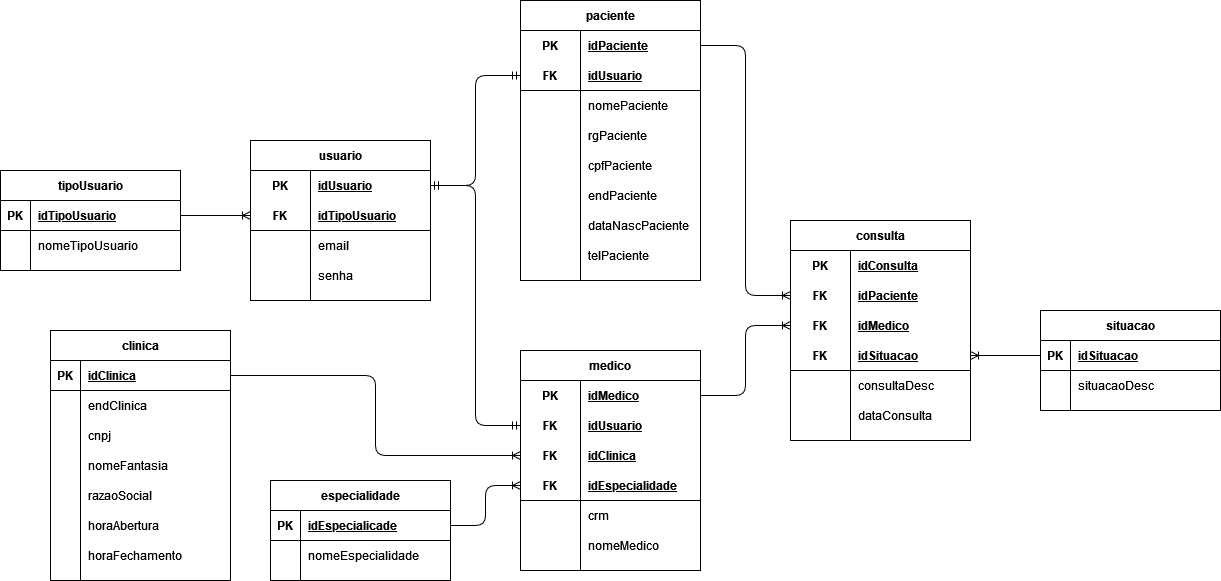
Como detalhado na representação, todos os usuários e seus respectivos cadastros ficaram armazenados na tabela **usuario**, que se relacionará com a tabela **tipoUsuario** que especificará as permissões dos mesmos. Com isso, esses usuários, conforme suas classificações, se dividirão em três grupos, os administradores, que não necessitarão de tratamento específico por apenas possuírem os dados de login, os pacientes, que possuirão os documentos relevantes e demais dados essenciais para o seu atendimento reservados na tabela **paciente**, os médicos, que além de possuírem sua documentação médica e nome armazenados na tabela **medico**, eles também se relacionarão com a tabela **clinica**, onde será determinada a unidade em que o profissional atende, e a tabela **especialidade**, que determinará a área de atuação do médico. As consultas, armazenadas e listadas na tabela **consulta,** serão as responsáveis por unir os dados do paciente que foi atendido e o médico que o atendeu, acrescentando também o horário da consulta, a descrição (opcional) e a situação, que se relacionará com os itens correspondentes na tabela **situacao**, que definirá o estágio em que se encontra o atendimento.



Modelo conceitual do banco de dados

## Modelo Lógico

O modelo lógico constitui uma representação mais técnica do projeto, trazendo conceitos mais próximos do que será implementado no banco de dados. Neste tipo de modelagem, o estabelecimento de padronizações já se torna crucial, tal qual a utilização dos campos (colunas) e como os dados deverão ser armazenados no banco. Nesta etapa já há a utilização das ditas chaves primárias (PK), utilizadas para distinguir diferentes   
registros dentro de uma mesma entidade, e as chaves estrangeiras (FK) utilizadas para estabelecer as relações construídas no modelo conceitual, permitindo a intersecção de   
dados de diferentes tabelas e facilitando na exibição dos mesmos. Isto é, uma chave que está em sua entidade de origem deverá ser única e exercerá o papel de índice, no   
entanto, quando compartilhada com uma segunda tabela, assumirá o papel de chave estrangeira e poderá se repetir dentro da tabela, tendo em vista que uma categoria poderá englobar vários registros (linhas).

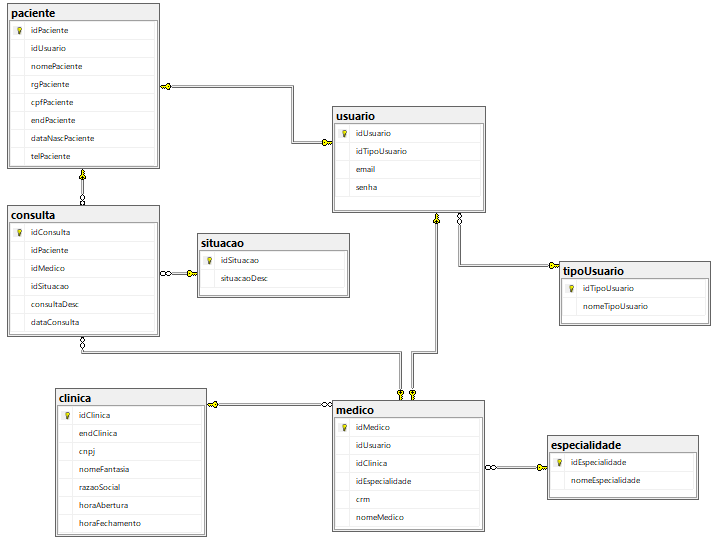


Modelo lógico do banco de dados

## Modelo Físico

Por fim, comparando aos supracitados, o modelo físico é o que mais se assemelha a versão composta em um banco de dados, isto é, trabalhando com o nível mais baixo de abstração de dados. Este modelo leva em consideração as limitações existentes no

Sistema de Gerenciamento de Banco de Dados (SGBD) escolhido para a sua confecção, além de usar como suporte os modelos construídos nos itens anteriores. Neste estágio também já são especificados os tipos de valores, os campos, tabelas, índices e todas os demais componentes que compõem a estrutura física do banco de dados. Resumidamente, o modelo físico representa uma pré-visualização do que logo   
será automatizado em algum SGBD.



Modelo físico do banco de dados

## Cronograma

|  |  |  |  |
| --- | --- | --- | --- |
|  | 19/08/2021 | 20/08/2021 | 23/08/2021 |
| Modelo Conceitual | X |  |  |
| Modelo Lógico | X |  |  |
| Modelo Físico | X |  |  |
| DDL | X |  |  |
| DML |  | X |  |
| DQL |  | X |  |
| Documentação |  |  | X |

### Trello

<https://trello.com/b/mz4GMGtv/sp-medical-group>

# Back-End

O Back-End é entendido como a parte responsável pelo funcionamento interno do sistema, que concilia toda a infraestrutura para o tratamento de dados, a segurança de informações, o gerenciamento das requisições do usuário e todos os demais processos que transcorrem por trás da aplicação. Para sua construção, foi utilizada a linguagem de programação orientada à objetos C#, juntamente a plataforma de desenvolvimento .NET Framework e da estrutura de aplicações web ASP.NET Core, este que possibilitou a criação da web API (Application Programming Interface – Interface de Programação de Aplicativos) do sistema.

As APIs podem ser concebidas como um conjunto de padrões e instruções estabelecidos para utilização do software, que utilizando o protocolo de comunicação HTTP (Hypertext Transfer Protocol – Protocolo de Transferência de Hipertexto) e as respostas e requisições no formato JSON (JavaScript Object Notation - Notação de Objetos JavaScript), universaliza o acesso ao sistema, considerando que independerá dos dispositivos utilizados para tal, sem preocupação com a linguagem utilizada.

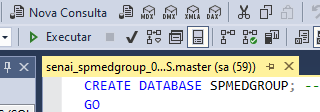
Neste projeto, foi utilizado o padrão de projeto **Repository Pattern**, separando os processos através de 5 camadas centrais: as **Domains** (Camada que representa as entidades (tabelas) do banco de dados); as **Interfaces** (Camada que define o que a API vai fazer, ou seja, o contrato da aplicação); os **Repositories** (Camada que define como serão feitas as funcionalidades previstas nas Interfaces); os **Contexts** (Camada que estabelece a comunicação com o banco de dados) e as **Controllers** (Camada que controla o fluxo de dados). Para se estabelecer uma comunicação ágil entre a aplicação e os dados contidos no banco, foi utilizado o Entity Framework Core para o mapeamento dos objetos.

**Executando a API:**

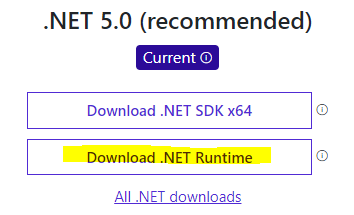
1. Primeiramente, clone o repositório disponível no link <https://github.com/Lucas-Araujo15/SpMedGroup> através do comando abaixo, executado pelo software de versionamento **Git**:



1. Antes de executar a API, é necessário criar o banco de dados em seu computador. Para isso, no repositório clonado, acesse a pasta **Banco de Dados**, em seguida a pasta **scripts** e abra em seu **SGBD** o arquivo **senai\_spmedgroup\_01\_DDL**. Feito isso, clique no botão **Executar**. Faça o mesmo para o arquivo **senai\_spmedgroup\_02\_DML**.



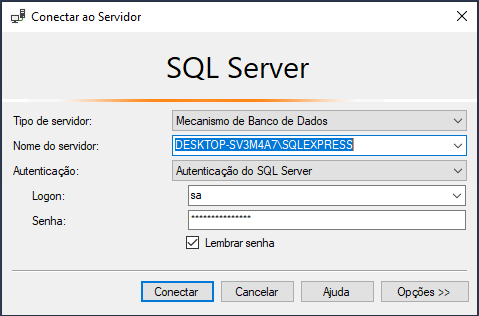
1. Com isso, é necessário também instalar o runtime do .NET 5.0 para rodar a aplicação. O link para a instalação está disponível em <https://dotnet.microsoft.com/download>



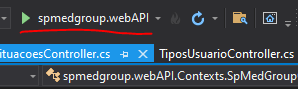
1. Instalado o runtime, agora é necessário realizar algumas alterações na aplicação para adequá-la a máquina local. Para isso, abra a pasta **Back-end**, em seguida abra a pasta **spmedgroup.webAPI** e abra a solução **spmedgroup.webAPI.sln** no **Microsoft Visual Studio.** Dentro da IDE, vá até a pasta **Contexts** e abra o arquivo **SpMedGroupContext.cs**. Na linha 36 do código, estará estabelecido uma string para conexão com o banco de dados. Diante disto, é necessário substituí-la e colocar o nome do seu servidor (Data Source), seu login (user Id) e sua senha (pwd).



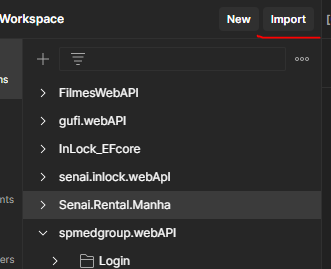
1. Para descobrir o nome de seu servidor, abra o seu SGBD e logo no início aparecerá uma janela para se conectar com seu servidor, demonstrando o seu nome. Para as demais informações, é só usar as credenciais que você utiliza para acessá-lo.



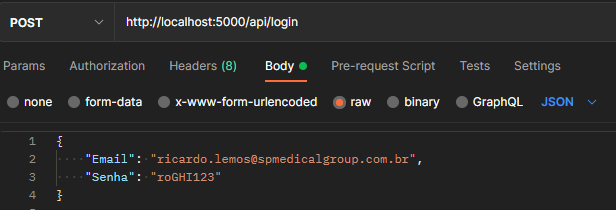
1. Substituída a string de conexão, retorne à solução aberta no Visual Studio e clique no botão de executar, sublinhado em vermelho na imagem.



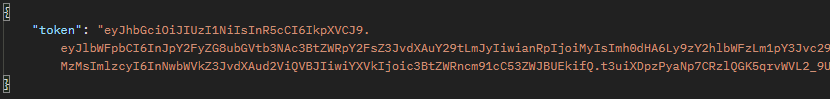
1. No software **Postman**, na sua área de trabalho, importe a coleção localizada na pasta **postman** no repositório clonado.



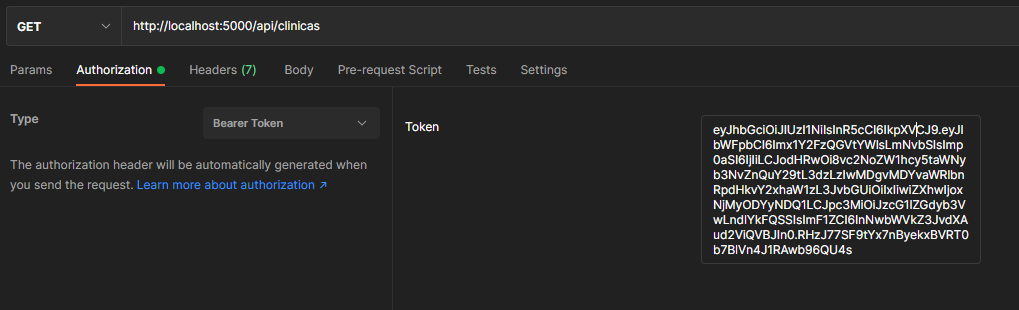
1. Após importar a coleção de requisições, é necessário se autenticar antes de testar a API. Para isso, dentro da coleção, vá na pasta **Login**, na requisição **Logar**, e execute-a passando no formato **JSON** uma das credenciais já armazenadas no banco. Para fornecer essas informações, clique em **Body**, depois em **raw** e altere o formato para **JSON**. Lembre-se que alguns tipos de usuário possuem restrições e não podem realizar determinadas requisições.



1. Após colocar as credenciais no formato correto, clique no botão **Send** para enviar a requisição. Em seguida, será retornado um token contendo suas informações de maneira codificada, prevenindo riscos de segurança. Nele estará definido o tempo de expiração de 30 minutos, sendo necessário a geração de outro token quando este tempo for atingido.



1. Para testar as demais requisições, copie o token gerado e na aba **Authorization** selecione o tipo de autenticação, que será o **Bearer Token**. No espaço a direita, cole o seu token, e caso não sejam necessárias outras informações para finalizar a requisição (atentar-se aos modelos disponibilizados na documentação Swagger apresentada ao executarmos a API), é só enviá-la.



## 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:** Cliente da clínica

#### Funcionalidades

1. O **administrador** poderá cadastrar, listar, atualizar e deletar as informações de uma clínica.
2. O **administrador** poderá cadastrar, listar, atualizar e deletar as informações das especialidades médicas da clínica.
3. O **administrador** poderá cadastrar, listar, atualizar e deletar as informações dos médicos da clínica.
4. O **administrador** poderá cadastrar, listar, atualizar e deletar as informações dos pacientes da clínica.
5. O **administrador** poderá cadastrar, listar, atualizar e deletar as possíveis situações de uma consulta.
6. O **administrador** poderá cadastrar, listar, atualizar e deletar os tipos de usuário existentes no sistema.
7. O **administrador** poderá alterar o status das consultas.
8. O **administrador** poderá cadastrar, listar e deletar os dados de um usuário.
9. Um **usuário autenticado** poderá atualizar suas credenciais e também colocar e ver sua imagem de perfil.
10. O **médico** poderá ver os agendamentos (consultas) associados a ele;
11. O **médico** poderá incluir a descrição da consulta que estará vinculada ao paciente (prontuário).
12. O **paciente** poderá visualizar suas próprias consultas;