**SP\_MEDICAL\_GROUP**

Documentação

Sumário

[1. Resumo 3](#_Toc90458368)

[2. Descrição do projeto 3](#_Toc90458369)

[3. Banco de dados relacional 3](#_Toc90458370)

[4. Modelagem de dados 3](#_Toc90458371)

[Modelo Conceitual 4](#_Toc90458372)

[Modelo Lógico 5](#_Toc90458373)

[Modelo Físico 6](#_Toc90458374)

[Cronograma 7](#_Toc90458375)

[Trello 7](#_Toc90458376)

[5. Back-End 8](#_Toc90458377)

[Descrição 8](#_Toc90458378)

[Passo a passo 8](#_Toc90458379)

[Funcionalidades 10](#_Toc90458380)

[Sistema Web 10](#_Toc90458381)

[Perfis de usuário: 10](#_Toc90458382)

[Funcionalidades: 10](#_Toc90458383)

[6. Aplicação WEB 11](#_Toc90458384)

[Descrição 11](#_Toc90458385)

[Passo a passo 11](#_Toc90458386)

[7. Mobile 12](#_Toc90458387)

[Descrição 12](#_Toc90458388)

[Passo a passo 12](#_Toc90458389)

[Perfis de Usuário 14](#_Toc90458390)

[Funcionalidades 14](#_Toc90458391)

[8. NoSQL 15](#_Toc90458392)

[Descrição 15](#_Toc90458393)

# Resumo

Este documento tem por função apresentar com detalhes o desenvolvimento do projeto SP Medical Group, dando destaque às principais etapas do processo, envolvendo e exibindo todas as atividades realizadas, a organização e o cronograma, além do planejamento e execução do Banco de Dados, API, Frontend e Mobile deste.

# Descrição do projeto

Seguindo ao escopo e às solicitações do cliente Fernando Strada, foi desenvolvido um banco de dados e uma aplicação API que atende a demanda de sua gestão de registros, assim armazenando de forma acessível e relacional as informações acerca de seus médicos, pacientes, administradores – usuários no geral -, consultas e especialidades. Para que isso fosse possível, as ferramentas como Trello, SQL Server e Visual Studio foram imprescindíveis neste processo e permitiram que as petições do cliente não fossem descartadas. Dessa forma, Strada possui um sistema mais otimizado e capaz de automatizar a administração da sua empresa.

# Banco de dados relacional x Banco de dados não-relacional

Um banco de dados relacional, como a nomenclatura já propõe, é um tipo de armazenamento organizado no qual os dados se relacionam entre si. Dessa maneira, o modelo relacional permite a conservação dos elementos de maneira mais intuitiva e próxima o que acarreta melhor acessibilidade. Visto que toda aplicação exige armazenamento, é de suma importância a presença e utilização de um banco de dados, pois a manipulação de dados se torna mais simplificada, além de garantir melhor precisão, eficiência e integridade do sistema em questão. Já o banco de dados não relacional possui uma estrutura mais flexível e dinâmica, sem o estabelecimento de linhas e colunas (tabelas) e armazena os dados através do modelo chave-valor (como no JSON, porém de forma binária - BSON).

# Modelagem de dados

A modelagem de dados se dá como primordial para um projeto de banco de dados, sendo seu principal objetivo representar as informações através do Modelo de Entidade Relacionamento (MER). Visto que os sistemas de banco de dados geralmente são relacionais, o acesso é mais rápido e eficaz, além de ser baseado em noções simplificadas e intuitivas, portanto é fundamental a organização nesta etapa.

## Modelo Conceitual

Diagrama

Descrição gerada automaticamenteO modelo conceitual costuma ser a primeira parte da modelagem (não obrigatoriamente), tem como objetivo traçar uma primeira organização mais simples do projeto, trazendo as classes e as respectivas cardinalidades que haverá entre elas.

## Modelo Lógico

A modelagem lógica é um protótipo mais aprofundado e mais sólido do modelo conceitual, trazendo uma visão mais ampla das necessidades do negócio, envolvendo as cardinalidades de forma mais dinâmica, as relações entre as chaves primárias e estrangeiras, além dos atributos de cada entidade.

Diagrama

Descrição gerada automaticamente

## Modelo Físico

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamenteA modelagem física pode ser entendida como uma esquematização mais específica do escopo, já apresentando os dados em modelo de tabela ou diagrama, assim simulando o banco de dados.

## Cronograma

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dia | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Modelo Conceitual | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Modelo Lógico | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Modelo Físico | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SCRIPT  DDL | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SCRIPT  DML |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| SCRIPT  DQL |  |  | X | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Desenvolvimento API |  |  |  |  | X | X | X | X |  |  |  |  |  |  |  |  |  |  |  |
| Testagem Postman |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |  |
| Design – Layout Baixa Fidelidade |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |
| Design – Layout Alta Fidelidade e Mobile |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |
| Aplicação Frontend Web |  |  |  |  |  |  |  |  |  |  | X | X | X |  |  |  |  |  |  |
| Aplicação Mobile |  |  |  |  |  |  |  |  |  |  |  |  |  | X | X | X | X |  |  |
| Banco NoSQL |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X | X |

### Trello

<https://trello.com/b/s2XG1BEH/2dmmedical-group>

# Back-End

## Descrição

Haja vista os paradigmas inseridos na elaboração de sistemas no cenário contemporâneo, se mostra necessário a utilização de arquitetura a partir de ferramentas e pacotes coerentes. Diante disso, o desenvolvimento do projeto SP Medical Group usou como instrumentos o Visual Studio para a criação de solução, a tecnologia API para a construção do código, utilizando como base determinados frameworks – foi aplicado o método de otimização ORM (Object-Relational Mapping) para se trabalhar com Entity Framework e DataBase First como forma de abordagem de dados.

Além disso, empregou-se padrões de implementação e regras de negócio através do REST (Representational State Transfer); Protocolo de Transferência de Hipertexto e verbos de requisição com HTTP (Hypertext Transfer Protocol); linguagem de resposta por meio de JSON (JavaScript Object Notation); documentação do projeto com Swagger e autenticação e autorização utilizando o modelo JWT ((JSON Web Token).

## Passo a passo

Primeiramente, para que seja possível executar a API e testar as requisições, é preciso que todos os elementos da solução e do Postman estejam devidamente organizados. A seguir:

1. Tela de computador com texto preto sobre fundo branco

   Descrição gerada automaticamenteSelecione o nome da solução na barra de ferramentas do Visual Studio para que o projeto possa ser executado.
2. No Postman insira o verbo da requisição (Get, Post, Delete, Put, Patch) e a Url. Se a requisição necessitar de um corpo, é recomendável que se faça o seguinte caminho: Body >> raw >> JSON

Texto

Descrição gerada automaticamenteCaso a aplicação ainda não tenha sido hospedada, a Url terá esse modelo: [http://localhost:5000/api/{nome](http://localhost:5000/api/%7bnome) da controller}

1. Em seguida, envie a requisição através do botão ‘SEND’ e teste:

Tela de celular com aplicativo aberto

Descrição gerada automaticamente

## Funcionalidades

Sendo o desenvolvimento do projeto SP Medical Group voltado para a organização de uma clínica médica e gestão de suas consultas, foi elaborado uma aplicação Web que realiza o **CRUD\*** de todas as entidades informadas – como Usuarios, Medicos, Pacientes, Consultas, etc – possuindo relação com Banco de Dados anteriormente criado com o intuito de formular um sistema moderno e completo, de acordo com as necessidades da empresa. Ademais, ainda é efetuado o login (autenticação - validação do usuário); autorização destes para determinadas funções a partir do seu tipo de usuário; executa a listagem de consultas de um determinado médico ou paciente; realiza o agendamento e cancelamento de consultas pelo administrador; como também implementa o upload de imagens para perfil de usuário.

**\*CRUD**

**C –** Create – Criar

**R –** Read – Leitura

**U –** Update – Atualizar

**D –** Delete - Deletar

## Sistema Web

### Perfis de usuário:

1. **Administrador**: Colaborador da área administrativa da clínica;
2. **Médico**: Colaborador que atua na área da saúde;
3. **Paciente**: Cliente da clínica;

### Funcionalidades:

1. O **administrador** poderá cadastrar qualquer tipo de usuário (administrador,

paciente ou médico);

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);

1. O **administrador** poderá cancelar o agendamento;
2. O **administrador** deverá informar os dados da clínica (como endereço, horário

de funcionamento, CNPJ, nome fantasia e razão social);

1. O **médico** poderá ver os agendamentos (consultas) associados a ele;
2. O **médico** poderá incluir a descrição da consulta que estará vinculada ao paciente

(prontuário);

1. O **paciente** poderá visualizar suas próprias consultas;

# Aplicação WEB

## Descrição

Considerando a formação sistemática e a arquitetura baseada em ferramentas e pacotes correlacionados e coesos utilizados no desenvolvimento do projeto SP Medical Group, fez-se necessário o uso de aplicativos como Node.js para execução de códigos em JavaScript fora de um navegador web; Visual Studio Code para elaboração das lógicas e códigos usando a biblioteca React JS, além do próprio Prompt de Comando (CMD) do computador para executar comandos de terminal.

## Passo a passo

Para executar a aplicação Web, é preciso ter a API em execução com a devida string de conexão do servidor do Banco de Dados.

Texto

Descrição gerada automaticamente Visto isso, atente-se às dependências do projeto: execute ‘npm i’ para trazer vinculações e pacotes antes instalados para que aplicação funcione de maneira correta.

Texto

Descrição gerada automaticamenteApós isso, execute ‘npm start’ para rodar o projeto:

Logo, a página será aberta no navegador e para parar a execução é só dar ‘ctrl + c’ no CMD.

# Mobile

## Descrição

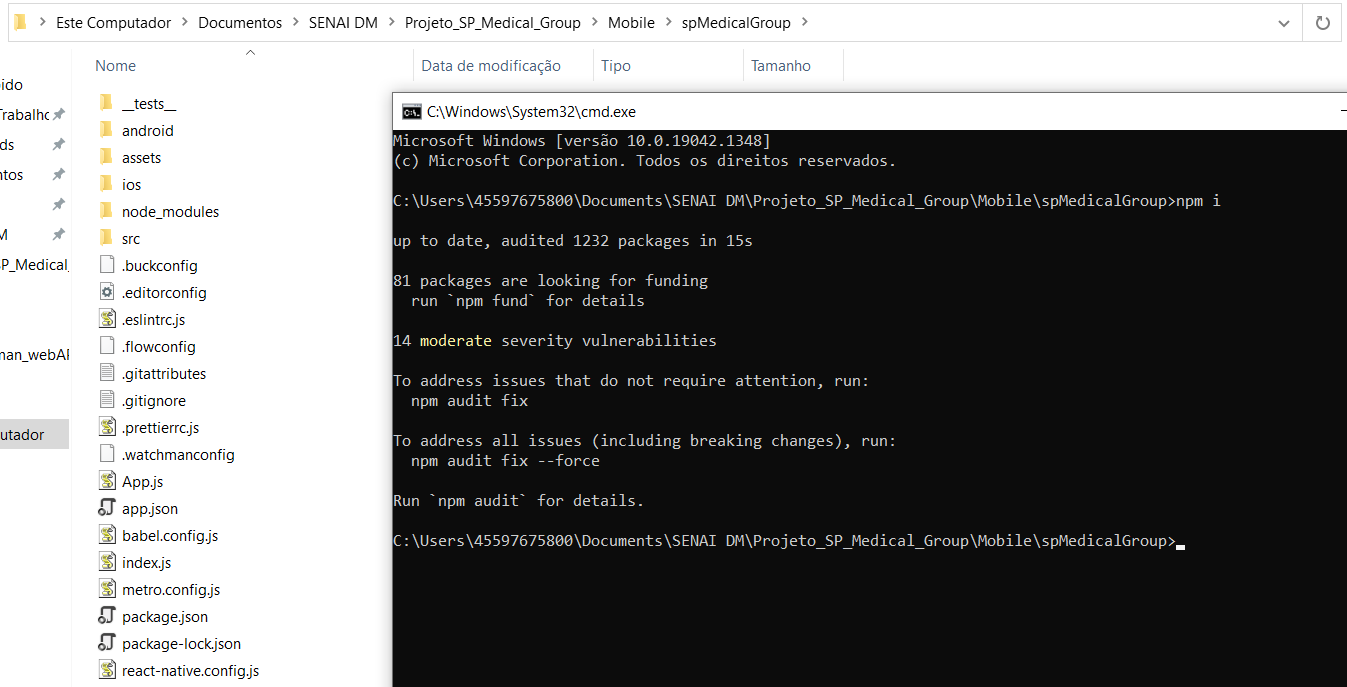
Levando em consideração as dinamicidades do cenário atual, é de suma importância o desenvolvimento de aplicações mobile, pois viabiliza meios mais acessíveis ao usuário e propõe maior praticidade. Sendo assim, de forma semelhante a aplicação web, o projeto SP Medical Group fez uso de aplicativos como Node.js para execução de códigos em JavaScript fora de um navegador web; Visual Studio Code para elaboração das lógicas e códigos utilizando React Native, Android Studio para simular as funções do dispositivo móvel, além do próprio Prompt de Comando (CMD) do computador para executar comandos de terminal e verificar informações necessárias (como o ip da máquina, por exemplo).

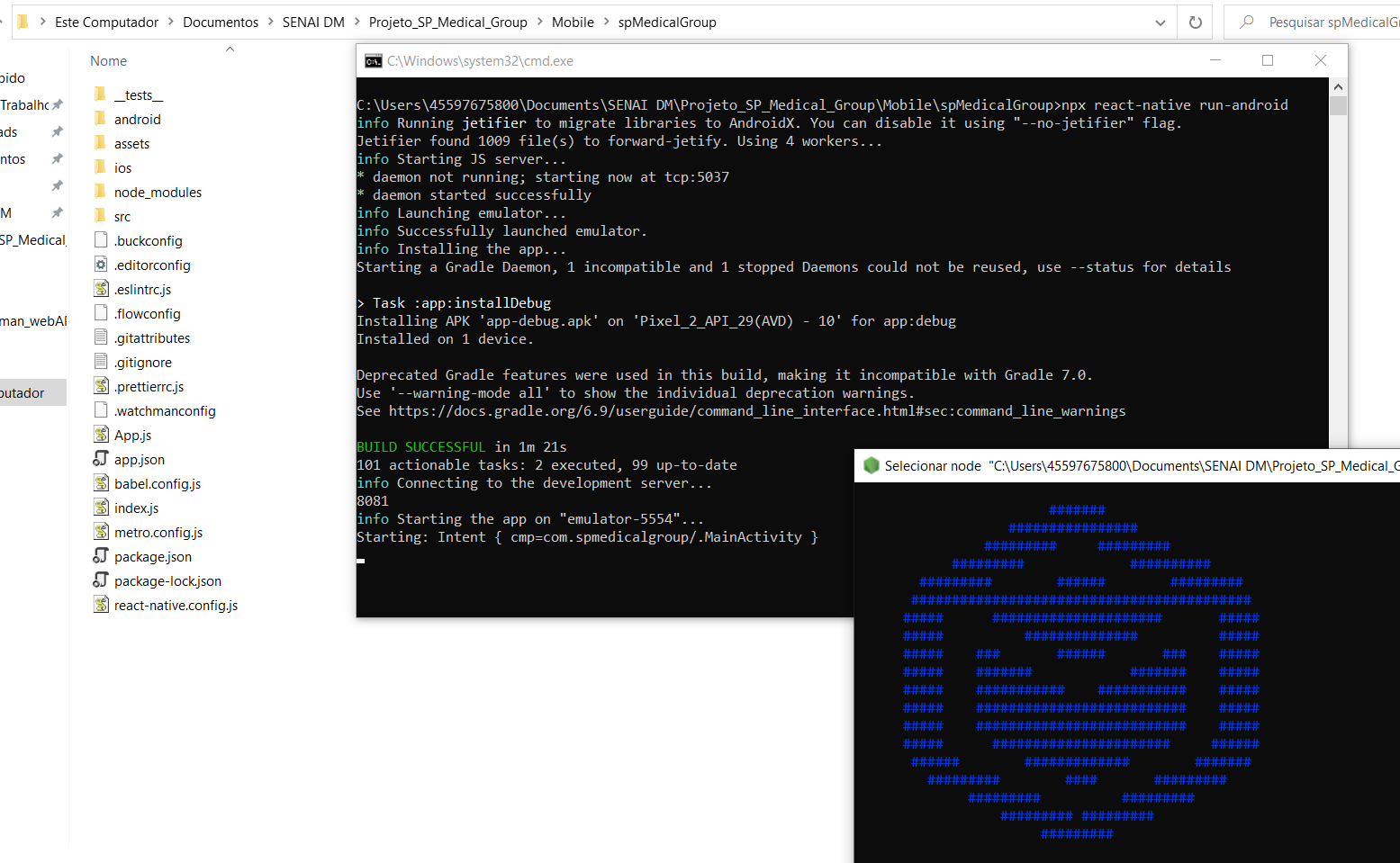
## Passo a passo

Para executar a aplicação, é preciso ter a API em execução com a devida string de conexão do servidor do Banco de Dados. É importante alterar a API configurando ip da máquina na pasta Properties. Caso não saiba, execute ipconfig no CMD (Prompt de Comando) e pegue o valor de IPv4, da seguinte forma:

Texto

Descrição gerada automaticamente

Em seguida, execute o comando ‘npm i’ para instalar todas as dependências (pacotes e documentos) necessárias para que o projeto funcione apropriadamente:

Logo, insira o comando ‘npx react-native run-android’ no CMD para executar o projeto:

Por fim, o emulador se iniciará e caso queira finalizar a execução é só dar ‘ctrl + c’ no CMD.



## Perfis de Usuário

1. **Médico**: Colaborador que atua na área da saúde;
2. **Paciente**: Cliente da clínica;

## Funcionalidades

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

# NoSQL

## Descrição

Tendo em vista as diversas funcionalidades do sistema e as exigências do cliente, foi empregado funcionalidades de Data Analytics visando levantar estatísticas acerca das especialidades de médicos mais frequentes de acordo com a localidade das clínicas. Para fazer a análise desses dados e implementar tais funções, foram utilizadas as ferramentas MongoDB (software de banco de dados orientado a documentos livre, de forma flexível e de código aberto), Visual Studio (para fazer a conexão com o banco e desenvolver lógicas), Visual Studio Code (para desenvolver lógicas e construir o mapa, identificando as localizações) e Postman (para realização de testes das requisições). Dessa forma, o sistema lista e cadastra o endereço, latitude e longitude do local, além de mostrar a especialidade da consulta.

