**SP MEDICAL GROUP**

Documentação

Sumário

[1. Resumo 3](#_Toc71092221)

[2. Descrição do projeto 3](#_Toc71092222)

[3. Modelagem de Dados 3](#_Toc71092223)

[Modelo Conceitual 3](#_Toc71092224)

[Modelo Lógico 3](#_Toc71092225)

[Modelo Físico 3](#_Toc71092226)

[Cronograma 3](#_Toc71092227)

[Trello 4](#_Toc71092228)

[4. Back-End 4](#_Toc71092229)

[Funcionalidades 5](#_Toc71092230)

[Sistema Web 5](#_Toc71092231)

[Perfis de usuário: 5](#_Toc71092232)

[Funcionalidades: 5](#_Toc71092233)

[Sistema Mobile 5](#_Toc71092234)

[Perfis de usuário: 5](#_Toc71092235)

[Funcionalidades: 5](#_Toc71092236)

# Resumo

Esse documento apresenta as conclusões relacionadas à segunda sprint do segundo semestre, relacionada a API, bem como os conhecimentos aplicados no último projeto, o SPMED

# Descrição do projeto

SPMedical Group é uma empresa médica que utilizava de uma base de dados no Excel para armazenamento das informações, e que contou com o desenvolvimento de um banco de dados SQL desenvolvido por mim, que fui contratado pelo CEO da empresa novamente para desenvolver uma API utilizando de autorizações diferentes para certas pessoas cadastradas no banco.

# Modelagem de Dados

Modelagem de dados é uma prática de organização baseada numa estrutura de dados, de maneira com a qual possamos entender melhor como essas funcionam, como o banco de dados se comportará, os dados que serão incluídos, entre outras informações.

## Modelo Conceitual

O modelo conceitual explora as relações entre os objetos do banco de dados, bem como os tipos dessas relações, que podem ser “1:1”, “1:N”, “N:N”.

Diagrama

Descrição gerada automaticamente

## Modelo Lógico

Comparado com o modelo conceitual, que normalmente é o apresentado para o cliente, o modelo lógico é mais técnico, e busca aproximar mais ainda os dados com o banco de dados

Diagrama

Descrição gerada automaticamente

## 

## Modelo Físico

O modelo físico é a representação do banco de dados gerada por este, sua construção.

Diagrama

Descrição gerada automaticamente

## Cronograma

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 | Dia 5 |
| Modelo Conceitual | X |  |  |  |  |
| Modelo Lógico |  | X | X |  |  |
| Modelo Físico |  |  |  | X | X |

### Trello

https://trello.com/b/YjCjwnKE/spmed

# Back-End

O sistema Back-End foi desenvolvido a partir da linguagem de programação C#, numa arquitetura de **API** baseada no DOTNET 5.0, na **IDE** do Visual Studio. Essa API foi integrada a partir do **Repository Pattern**, e da utilização do **Entity Framework Core**, com a abordagem **database first**. As requisições e respostas da API se baseiam em **JSON** e a autenticação é feita a partir do **JWT**.

**API** – Application Programming Interface – Interface de Programação de Aplicativos.

**IDE** – Integrated Development Environment – Ambiente utilizado para desenvolvimento

**JSON** – JavaScript Object Notation – Notação de Objetos JavaScript.

**Repository pattern** – Padrão de esqueleto da aplicação que se baseia na ideia de domínios que representam entidades do banco, interfaces que introduzem métodos, repositórios que executam esses métodos, e controladores que recebem requisições do front-end executam repositórios.

**Entity Framework Core** – Mapeador de banco de dados baseado no .NET, que funciona com diversos tipos de bancos de dados, e facilitam a conexão entre o backend e o armazenamento de dados.

**Database first** – Abordagem do EF Core que se baseia no desenvolvimento c# de uma aplicação com o banco de dados já pronto.

**JWT** – JSON Web Token – Padrão de criação de dados baseados em JSON com assinatura e codificação que contém algum número de declarações.

Para a execução da aplicação, você necessitará do DOTNET 5.0, bem como um editor relacionado ao SQL Server, e o Microsoft Visual Studio.

Ao iniciar a API, importe a coleção (“SPMED.Manha.postman\_collection”) no aplicativo de requisições Postman e siga as instruções do Swagger (“http://localhost:5000/swagger/index.html”) para utilizar das funcionalidades

## Funcionalidades

Insira as funcionalidades que a API atende.

Por exemplo:

## 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);

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;

## Sistema Mobile

### Perfis de usuário:

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

### Funcionalidades:

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