

Documentação

SviGufo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Alameda Barão de Limeira, 539 – São Paulo/SP |  |  |  | sp.senai.br |

Sumário

1. [Resumo 2](#_Toc533767843)

[Objetivos](#_Toc533767844)

1. [Descrição do projeto 2](#_Toc533767845)

[Resumo do projeto](#_Toc533767846)

1. [Modelagem de Software 3](#_Toc533767847)

[Modelo Lógico](#_Toc533767848)

[Modelo Físico](#_Toc533767849)

[Modelo Conceitual](#_Toc533767850)

[Cronograma](#_Toc533767851)

1. [Funcionalidades 5](#_Toc533767852)

[Web](#_Toc533767853)

[Mobile](#_Toc533767854)

1. [Protótipos 6](#_Toc533767855)

[Web](#_Toc533767856)

[Mobile](#_Toc533767857)

1. [Front-End 7](#_Toc533767858)
2. [Mobile 8](#_Toc533767859)
3. [Arquitetura do Projeto 9](#_Toc533767860)
4. [Referências 10](#_Toc533767861)

[Links](#_Toc533767862)

[Livros](#_Toc533767863)

Resumo

Objetivos

Objetivo de se acostumar ao Banco de Dados, tendo o primeiro projeto, exercendo os aprendizados da primeira SPRINT

Descrição do projeto

Coloque aqui a descrição completa do projeto

Resumo do projeto

Coloque aqui um resumo sobre o projeto

Modelagem de Software

Coloque aqui brevemente o que é a modelagem de software

Modelagem de Software, é uma forma de como explicar o funcionamento, e como o sistema ou próprio software se desenha para ter um melhor aproveitamento.

Modelo Lógico

Modelo feito pelo excel, exercendo as funcionalidades correspondidas, contendo chaves primárias e estrangeiras.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Usuários |  |  |  |
| ID | Nome | E-mail | Senha | Tipo |
| 1 | EU | [e@](mailto:e@) | \*\*\* | 1 |
| 2 | ee | [a@](mailto:a@) | \*\*\* | 2 |
| 3 | qq | [q@](mailto:q@) | \*\*\* | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Médico |  |  |  |  |
| ID | ID\_Usuario | CRM | ID\_Especiliadade | |
| 1 | 3 | 44422 | 1 |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Pacientes |  |  |  |  |  |
| ID | Nome | CPF | RG | Endereço | DTNascimento | |
| 1 | Enzo | 111111 | 112212121 | Rua tal | 21/12/1990 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Agendamentos | |  |  |  |
| ID | ID\_Paciente | IDMedico | DT\_HORA | Status |
| 1 | 1 | 1 | 20/02/2000 08:00 | 2 |

|  |  |
| --- | --- |
| Especialidade | |
| ID | Nome |
| 1 | Psiquiatra |

|  |  |
| --- | --- |
|  | Status |
| ID | Status |
| 1 | Agendada |
| 2 | Realizada |
| 3 | Cancelada |

Modelo Físico

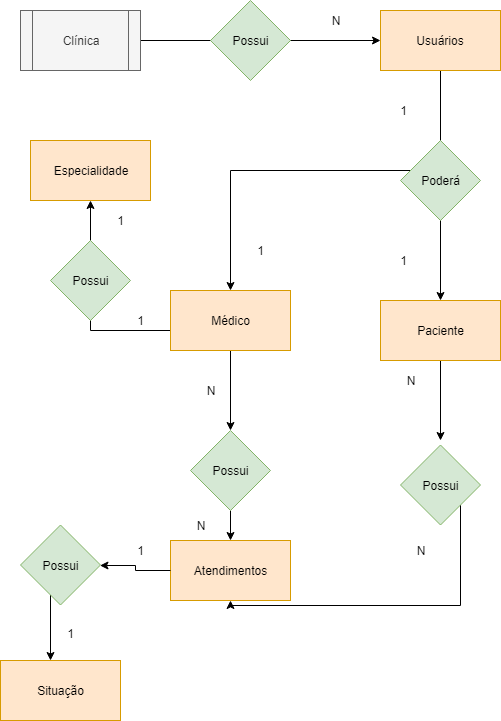
Insira aqui uma breve descrição sobre o modelo físico e a imagem exportada



­­­­­­­­­­­­­­­

Modelo feito pelo SQL, seguindo o script de Criação, contendo chaves primárias e estrangeiras

Modelo Conceitual



Modelo feito pelo site Draw.io, explicando o modelo lógico, contendo cardinalidade.

Cronograma

|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 | Dia 5 | Dia 6 | Dia 7 | Dia 8 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Modelo Lógico** | Início |  |  |  |  |  |  |  |
| **Modelo Físico** | Início |  |  |  |  |  |  |  |
| Modelo Conceitual |  |  |  |  |  |  |  |  |

Funcionalidades

Web

**- Cadastro de Usuários:** Só o Usuário com o tipo de Usuário Administrador, pode exercer essa funcionalidade. E ele pode cadastrar qualquer outro tipo de Usuário como: Usuário Comum, Médico ou até outro Administrador.

**//**  **Passo a Passo :** Usei o método DataBase First , onde é criado o Banco de Dados primeiro e depois o código. Já com o os domains definidos, criei a interface, para criar a tarefa Cadastrar. Que seria um void, porque ele não receberia um retorno por se tratar de um cadastro e no Parâmetro usei o Domain: Usuário. No Repositório , foi aonde eu fiz essa funcionalidade funcionar, usei o context , que já existe a função de adicionar ao banco de dados e a de salvar o banco. No Controller usei o HTTP POST, e chamei o método do Repositório. Coloquei um return OK() , caso funcionasse , e se ocorresse algum erro, o return BADREQUEST() me informaria. Como seria só o Administrador que pode Cadastrar o Usuário. Usei o JWT(JSON WEB TOKEN) que é uma forma de criptografar informações do Usuário. Com essas informações , o sistema já verifica se esse Usuário é um administrador ou usuário comum. Aí vem o Authorize , que autoriza ou não o Usuário a usar essa funcionalidade. O JWT fica na startup do projeto, com uma linha de comandos grande.

**- Cadastro de Agendamentos:** Só o Usuário com o tipo de Usuário Administrador, pode exercer essa funcionalidade. E ele deve informar o paciente, o médico (possuirá sua especialidade, como: Pediatria, Cardiologia etc.), a data de agendamento, e a situação do agendamento: se estiver agendada, realizada ou até mesmo cancelada, e a descrição sobre o agendamento: o médico se quiser pode colocar uma (não é obrigado).

**//**  **Passo a Passo :** Usei o método DataBase First , onde é criado o Banco de Dados primeiro e depois o código. Já com o os domains definidos, criei a interface, para criar a tarefa Cadastro de Agendamentos. . Que seria um void, porque ele não receberia um retorno por se tratar de um cadastro e no Parâmetro usei o Domain: No Repositório , foi aonde eu fiz essa funcionalidade funcionar, usei o context , que já existe a função de adicionar ao banco de dados e a de salvar o banco. No Controller usei o HTTP POST, e chamei o método do Repositório. Coloquei um return OK() , caso funcionasse , e se ocorresse algum erro, o return BADREQUEST() me informaria. Se o Médico quisesse colocar uma Descrição, criei o Método **– Atualizar Agendamento:** Que seria um void , ele não receberia retorno. E no Repositório , no context usei uma função que atualiza dados do Banco. No Controller usei o HTTP PUT.

**- Listar Clínica:** O administrador deverá mostrar os dados da Clínica: Endereço, CNPJ, Nome Fantasia e Razão Social da Clínica.

**//**  **Passo a Passo :** Usei o método DataBase First , onde é criado o Banco de Dados primeiro e depois o código. Já com o os domains definidos, criei a interface, para criar a tarefa Listar Clínica. Criei uma Lista de Clínicas, não usei nada de Parâmetro. No Repositório , do context usei uma função para listar e retornei o mesmo. No controller usei HTTP GET, e chamei o método do Repositório. Coloquei um return OK() , caso funcionasse , e se ocorresse algum erro, o return BADREQUEST() me informaria.

**- Listar Agendamentos:** O médico ou paciente podem ver quais agendamentos ele está associado. E o administrador poderá listar todos os agendamentos.

**//**  **Passo a Passo :** Usei o método DataBase First , onde é criado o Banco de Dados primeiro e depois o código. Já com o os domains definidos, criei a interface, para criar a tarefa Listar TODOS, Listar Médico e Listar Paciente.

**- Listar Médico ou Paciente:** No repositório do Paciente criei o seguinte Método: Procurar, que ele procura um Paciente pelo ID. No repositório do Agendamento , o método construído foi o de Listar Paciente , onde ele lista só os pacientes com o ID fornecido do Controller. E no Controller , quando um Usuário loga o JWT envia o ID dele para o Controller , e usei para listar só os pacientes logados.E foi usado o HTTP GET. Com o Médico fiz o mesmo.

**- Listar Todos:** Criei uma Lista de Agendamentos, não usei nada de Parâmetro. No Repositório , do context usei uma função para listar e retornei o mesmo. No controller usei HTTP GET, e chamei o método do Repositório.

**- Login:** O Usuário deverá informar o email e a senha correspondente ao cadastro dele. A autenticação é feita pelo JWT.

**//**  **Passo a Passo :** sei o método DataBase First , onde é criado o Banco de Dados primeiro e depois o código. Como teria que usar os dados do usuários tive que criar um ViewModel , criei a interface, para criar a tarefa Login . E no Repositório o método que usei foi o de Buscar, onde ele busca o email e senha do Usuário para verificar se existe. No controller do Login chamei o método do Repositório. E foi aí que usei as claims do JWT, que ajudou no resto do Projeto. Foi usado o HTTP GET. Controller usei o HTTP POST, e chamei o método do Repositório. Coloquei um return OK() , caso funcionasse , e se ocorresse algum erro, o return BADREQUEST() me informaria.

Mobile

Protótipos

Web

Mobile

Front-End

Mobile

Arquitetura do Projeto

Referências

Links

Livros