

Saúde Digital

Integrantes: Lucas Barranha Giannini - RM:564508- 1TDSPW

Miguel Henrique Oliveira Dias – RM: 565492- 1TDSPW

João Victor Gomes de Souza – RM: 560907 – 1TDSPW

Sistema de gerenciamento de dados clínicos e Administrativos

=====

Sumário

1. Objetivo e Escopo do Projeto	[Página: 2]
1.1. Propósito Principal	[Página:2]
1.2. Limites de Atuação (Escopo)	[Página:2]
2. Descrição das Funcionalidades	[Página:3]
3. Diagrama de Classes	[Página:4]
4. Modelo de Entidade-Relacionamento (MER)	[Página:4]
5. Tabela de endpoints	[Páginas:5,6 e 7]
5.1 Passo a passo da execução.....	[Página:8]
6. Protótipo do sistema	[Páginas:9-13]

=====

1. Objetivo e Escopo do Projeto

1.1. PROPÓSITO PRINCIPAL

O sistema SAÚDE DIGITAL tem como objetivo principal centralizar e cadastrar as informações cruciais de pacientes, médicos, consultas, lembretes e seus respectivos parentes de contato.

O objetivo é fornecer uma interface simples e de fácil entendimento para a manipulação completa desses dados, garantindo a integridade entre as entidades, por exemplo, assegurando que uma Consulta só possa ser cadastrada se o Paciente e o Médico envolvidos já existirem, e assegurando que um Parente só exista se um Paciente esteja cadastrado no sistema, o “ator principal”.

1.2. LIMITES DE ATUAÇÃO (ESCOPO)

O sistema atua com um CRUD completo para todas as entidades do sistema (paciente, médico, parente, lembrete e consulta). Tem a integração com o banco de dados Oracle via JDBC, e com uma API, seguindo as práticas de um http. Foi utilizado QUARKUS para testar localmente a API através do Insomnia ou Postman. Existe a validação de foreigns keys (FK) e tratamento de exceções e padrões de respostas em JSON. Fora do escopo, não existe modelos Permite cadastrar, listar, buscar, atualizar e excluir pacientes, armazenando nome, CPF, data de nascimento e e-mail. gráficos e nem relatórios avançados, a listagem de dados é simples e funcional.

2. Descrição das Funcionalidades

A aplicação foi desenvolvida aplicando boas práticas de separação de camadas (Model, DAO, Service e Resource), e disponibilizando uma **API RESTful completa** para consumo externo (como no front-end React).

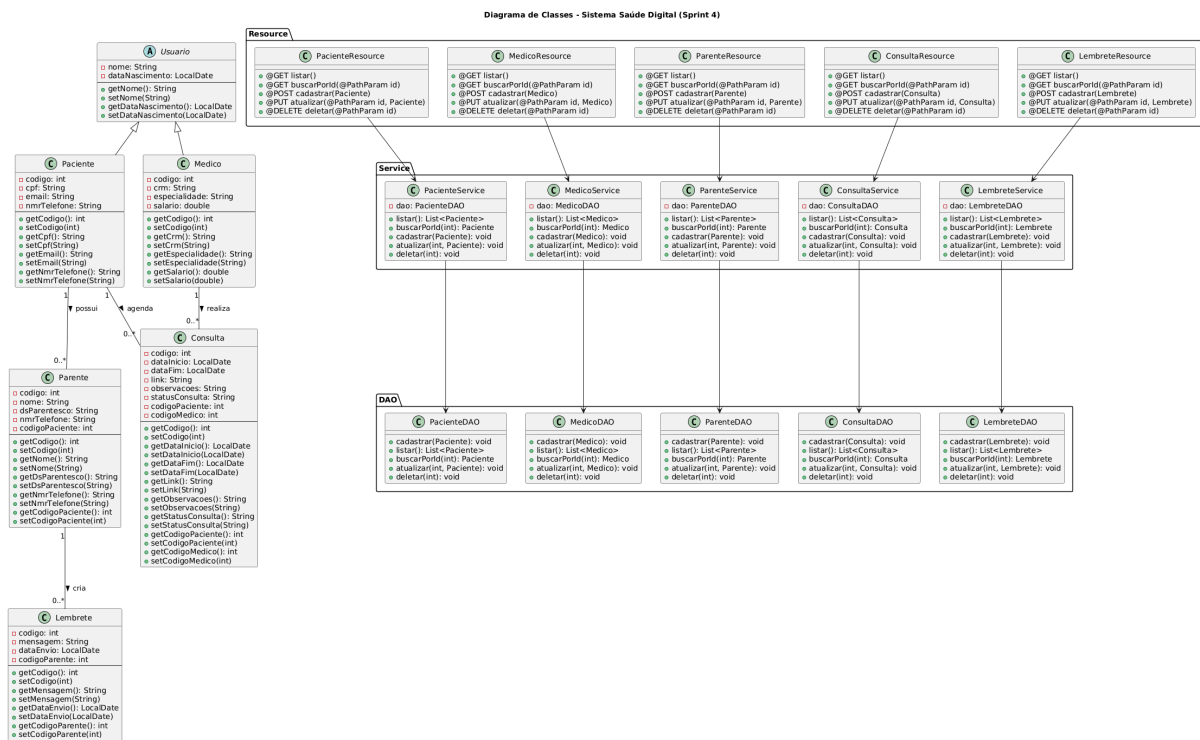
CRUD de pacientes - Permite cadastrar, listar, buscar, atualizar e excluir pacientes, armazenando nome, CPF, data de nascimento e e-mail.

CRUD de médico - Gerencia informações de médicos, incluindo nome, CRM e especialidade.
--

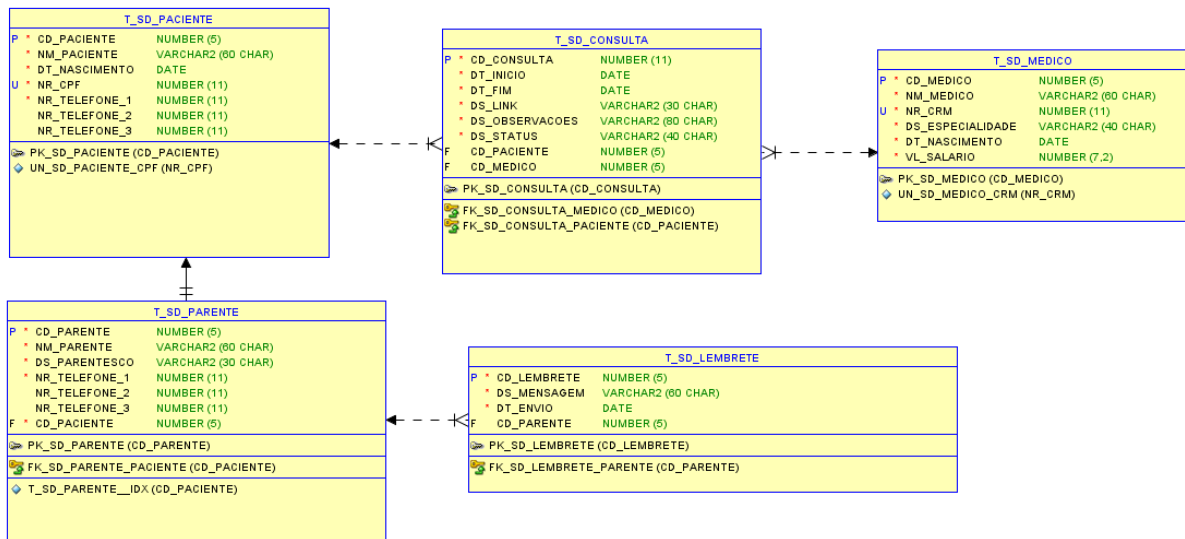
CRUD de parentes - Controla os contatos de parentes vinculados a pacientes, garantindo o relacionamento via chave estrangeira (codigoPaciente).

CRUD de lembretes - Permite criar, visualizar, atualizar e excluir lembretes associados aos parentes, com mensagem e data de envio.
CRUD de consultas - Gerencia o agendamento de consultas, relacionando médico e paciente existentes, com validação de integridade.
Camada DAO - Responsável pela persistência dos dados no banco Oracle, utilizando JDBC.
API restful - Disponibiliza endpoints REST para todas as entidades, permitindo comunicação com front-end e ferramentas como Insomnia ou Postman.
Tratamento de exceções - Inclui respostas HTTP apropriadas (200, 201, 404, 500) para cada operação, conforme o resultado da requisição.
Conexão com o banco de dados - Realizada via JDBC, através de application.properties e DataSource configurados.
Integração com Front-End - API exposta para consumo no front-end, possibilitando operações diretas sobre pacientes, lembretes e consultas.

3.Diagrama de Classes



4. Modelo de Entidade Relacionamento(MER)



5.Tabela de endpoints

URI	Método HTTP	Descrição	Status Esperado
/pacientes	GET	Lista todos os pacientes cadastrados.	200 OK
/pacientes/{id}	GET	Retorna um paciente específico pelo seu ID.	200 OK, 404 Not Found
/pacientes	POST	Cadastra um novo paciente no sistema.	201 Created, 400 Bad Request
/pacientes/{id}	PUT	Atualiza os dados de um paciente existente, pelo id.	200 OK, 404 Not Found
/pacientes/{id}	DELETE	Remove permanentemente um paciente pelo id.	204 No Content, 404 Not Found

URI	Método HTTP	Descrição	Status Esperado
/medicos	GET	Lista todos os médicos cadastrados.	200 OK
/medicos/{id}	GET	Retorna os dados de um médico específico pelo ID.	200 OK, 404 Not Found
/medicos	POST	Cadastra um novo médico no sistema.	201 Created, 400 Bad Request
/medicos/{id}	PUT	Atualiza as informações de um médico pelo ID.	200 OK, 404 Not Found

/medicos/{id}	DELETE	Remove um médico cadastrado pelo ID.	204 No Content, 404 Not Found
---------------	---------------	--------------------------------------	-------------------------------

URI	Método HTTP	Descrição	Status Esperado
/consultas	GET	Lista todas as consultas agendadas ou realizadas.	200 OK
/consultas/{id}	GET	Retorna os detalhes de uma consulta específica.	200 OK, 404 Not Found
/consultas	POST	Agenda uma nova consulta (valida FK de Paciente e Médico).	201 Created, 400 Bad Request
/consultas/{id}	PUT	Atualiza as informações de uma consulta existente.	200 OK, 404 Not Found
/consultas/{id}	DELETE	Cancela uma consulta cadastrada.	204 No Content, 404 Not Found
/consultas/pacientes/{codigoPaciente}	GET	Retorna os detalhes de uma consulta de um paciente específico, informando o id do paciente	200 OK, 404 Not Found
/consultas/medicos/{codigoMedico}	GET	Retorna os detalhes de uma consulta de um médico específico, informando o id do medico	200 OK, 404 Not Found

URI	Método HTTP	Descrição	Status Esperado
/parentes	GET	Lista todos os parentes cadastrados.	200 OK
/parentes/{id}	GET	Retorna os dados de um parente específico.	200 OK, 404 Not Found
/parentes	POST	Cadastra um novo parente vinculado a um paciente (FK).	201 Created, 400 Bad Request

/parentes/{id}	PUT	Atualiza os dados de um parente existente.	200 OK, 404 Not Found
/parentes/{id}	DELETE	Remove um parente do sistema.	204 No Content, 404 Not Found
/parentes/pacientes/{codigoPaciente}	GET	Retorna os parentes vinculados a um paciente específico, dado pelo id do paciente;	204 No Content, 404 Not Found

URI	Método HTTP	Descrição	Status Esperado
/lembretes	GET	Lista todos os lembretes registrados no sistema.	200 OK
/lembretes/{id}	GET	Retorna um lembrete específico pelo ID.	200 OK, 404 Not Found
/lembretes	POST	Cadastra um novo lembrete vinculado a um parente (FK).	201 Created, 400 Bad Request
/lembretes/{id}	PUT	Atualiza a mensagem ou data de envio de um lembrete.	200 OK, 404 Not Found
/lembretes/{id}	DELETE	Exclui um lembrete do sistema.	204 No Content, 404 Not Found
/lembretes/parente{codigoParente}	GET	Retorna os lembretes vinculados a um parente específico, dado pelo id do lembrete;	204 No Content, 404 Not Found

5.1 PASSO A PASSO DA EXECUÇÃO:

Para o projeto funcionar, é necessário também estar com o projeto já no Quarkus. Acesse o quarkus, preencha Group e Artifact, adicione as extensões Rest, Rest Jackson e JDBC Oracle e extraia.

Configure o banco de dados. Na parte do código application.properties, coloque o nome de usuário e senha de seu banco para realizar a conexão. Verifique o arquivo application.properties se as seguintes dependências estão inseridas:

```
quarkus.datasource.db-kind=oracle
quarkus.datasource.username=User
quarkus.datasource.password=Password
quarkus.datasource.jdbc.url=jdbc:oracle:thin:@oracle.fiap.com.br:1521:orcl
quarkus.datasource.jdbc.driver=oracle.jdbc.OracleDriver
quarkus.http.port=8080
```

Após a realização disso, você já estará conectado com o seu banco. Em seguida, abra o sql developer, insira seu nome de usuário e senha e se conecte também. Feito isso, no código java, procure o arquivo tabela.sql. Nele, contém todos os drops e sequences necessários para “dar vida” ao banco. Passe-os para o developer e execute o script, criando todas as tabelas, sequences e atributos.

Verifique o arquivo pom.xml. Ele precisa de algumas dependências para funcionar, entre elas o modelMapper

```
<dependency>
  <groupId>org.modelmapper</groupId>
  <artifactId>modelmapper</artifactId>
  <version>3.1.1</version>
</dependency>
```

Dependencia de validações

```
<dependency>
  <groupId>io.quarkus</groupId>
  <artifactId>quarkus-hibernate-validator</artifactId>
</dependency>
```

Abrindo o projeto no IntelliJ, vá em Run, edit configuration, selecione o ícone + e depois procure por Maven, logo em seguida em “name”: runquarkusEM, no diretório, selecione o do projeto e em commando use quarkus:dev e clique em ok. Feito isso, aperte no botão verde na parte superior do IntelliJ e depois de um tempo, gerará um link como <http://localhost:8080>. Copie esse link, coloque no Insomnia ou PostMan e siga os comandos que estão na tabela de endpoints para popular o projeto.

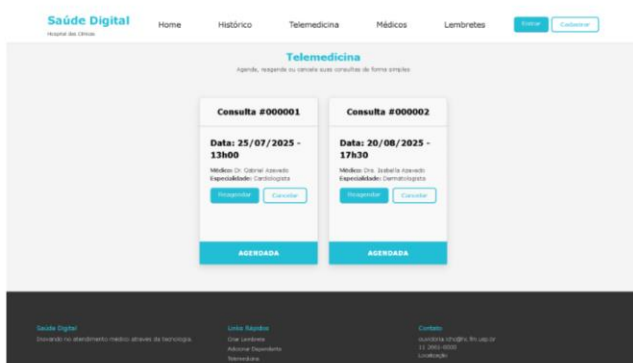
6. Protótipo do sistema



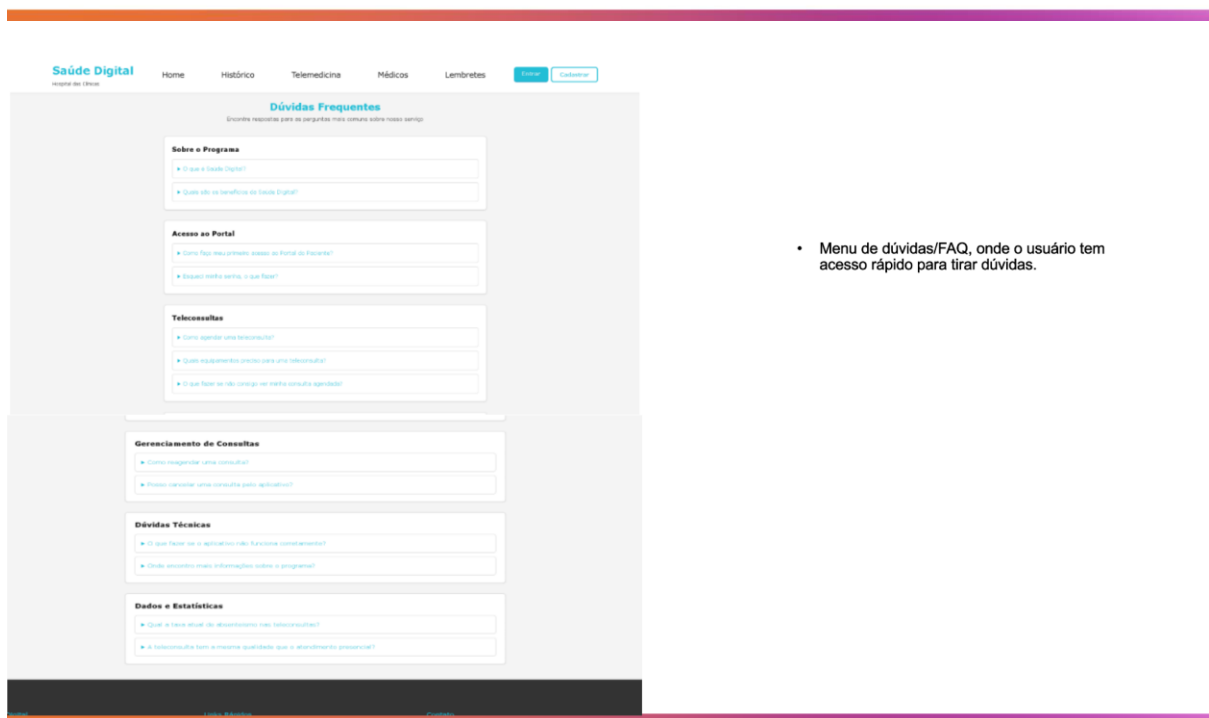
Menu principal/home do sistema, simples e intuitivo, dando boas-vindas ao usuário e apresentado a tela inicial com as opções de acesso e novas abas nos links rápidos



- Histórico de consultas, menu que apresenta todas as consultas que o usuário já realizou ou as consultas que foram canceladas, mostrando as datas do ocorrido.



- Menu Telemedicina, onde o usuário acessa as consultas que ainda não foram realizadas, com a opção de reagendar para um dia ou hora diferentes, ou cancelar as consultas.



- Menu de dúvidas/FAQ, onde o usuário tem acesso rápido para tirar dúvidas.

The screenshot shows the 'Contate' form on the Saúde Digital website. The header includes the logo 'Saúde Digital Hospital das Clínicas' and navigation links: Home, Histórico, Telemedicina, Médicos, Lembretes, and buttons for 'Login' and 'Cadastro'. The form itself is titled 'Contate' and has a subtitle 'Entramos aqui para tirar suas dúvidas, receber sugestões e melhorar cada vez mais nosso atendimento.' It contains input fields for 'Nome', 'E-mail', and 'Mensagem', followed by a 'Enviar Mensagem' button.

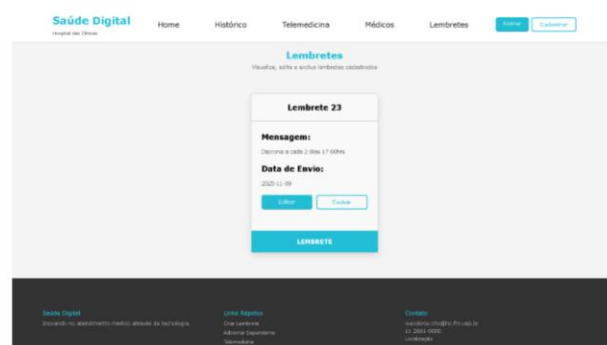
- Menu Fale Conosco, que permite ao usuário tirar dúvidas mais específicas ou dar sugestões de melhorias. Basta inserir a mensagem que deseja, se identificar e enviar. ido para tirar dúvidas.

The screenshot shows the 'Crie sua conta' form on the Saúde Digital website. The header is identical to the previous screenshot. The form is titled 'Crie sua conta' with the subtitle 'Preencha os dados abaixo para se cadastrar.' It includes input fields for 'Nome Completo', 'CPF', 'E-mail', 'Senha', and 'Confirme a senha', followed by a 'Cadastrar' button. A link 'Já tem uma conta? Clique aqui' is located at the bottom of the form.

- Menu de cadastro, onde o usuário cria a conta no site informando nome, cpf e email.



- Menu de médicos, onde posso ver os médicos cadastrados no sistema, esses criados no banco de dados/java.



- Menu de lembretes, onde é possível adicionar, editar e remover lembretes.

Parentes
Cadastrar, editar e visualizar os parentes cadastrados e em processo

Nome:

Parentesco:

Telefone:

ID Paciente:

[Cadastrar](#) [Cancelar](#)

Pedro Guilherme

Parentesco: Filho

Telefone: (11) 91234-5678

ID Paciente: 1

[Editar](#) [Excluir](#)

Ana Julia

Parentesco: Mãe

Telefone: (11) 98765-4321

ID Paciente: 2


[Editar](#) [Excluir](#)

- Menu de dependentes, onde é possível adicionar, editar e remover algum parente cadastrado.


Saúde Digital
Hospital das Clínicas

Home Histórico Telemedicina Médicos Lembretes [Novo](#) [Cadastrar](#)


Nosso Time
Conheça os profissionais responsáveis por desenvolver a Saúde Digital



João Victor Gomes de Souza
CRM: 345678 | Tatuá, SP | 11030-900
[Saúde Digital](#)



Lucas Barreto da Costa
CRM: 345679 | Tatuá, SP | 11030-900
[Saúde Digital](#)



Miguel Henrique de Oliveira Dias
CRM: 345680 | Tatuá, SP | 11030-900
[Saúde Digital](#)

Saúde Digital
Iniciando no atendimento médico através da tecnologia

Sobre Nós
Sua jornada
Nossa missão
Nossa visão
Nossa história
Nossa equipe

Contato
Av. Paulista, 1000 - São Paulo, SP
CEP: 01305-900
Fone: (11) 3456-7890

- E por fim, o menu de integrantes da equipe, com nosso github e linkedin.