

FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA

Natan Freitas de Moraes - RM564992 - 1TDSPI

Eduardo Batista Locaspi – RM561713 – 1TDSPI

Victor Alves Lopes – RM561833 – 1TDSPI

AxcessTech

São Paulo

2025

FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA

Natan Freitas de Moraes - RM564992 - 1TDSPI

Eduardo Batista Locaspi – RM561713 – 1TDSPI

Victor Alves Lopes – RM561833 – 1TDSPI

AxcessTech

Sprint 3 apresentado à Faculdade de Informática e Administração Paulista como requisito de nota para a avaliação da disciplina *Domain Driven Design using Java*, sob a orientação do Professor Henrique Dias Pastor.

São Paulo

2025

SUMÁRIO

1 DESCRIÇÃO DO PROJETO	3
1.1 Objetivo e escopo	4
2 DOCUMENTAÇÃO TÉCNICA.....	4
2.1 Descrição das funcionalidades	4
2.1.1 Funcionalidades gerais	5
2.2 Tabela de EndPoints	5
2.2.1 ResourceAreaFuncionario (“/funcionario”)	5
2.2.2 ResourceCadastro (“/cadastro”){.....	5
2.2.3 ResourceLogin (“/login”)	6
2.3 Telas do Sistema	6
2.4 Modelo de Entidade-Relacionamento (MER).....	8
2.4.1 Modelo geral	9
2.4.2 Modelo aplicado no sistema Java	9
2.5 Diagrama de Classes.....	10
3 COMPLEMENTARES	11
3.1 GitHub do Projeto.....	11

1 DESCRIÇÃO DO PROJETO

1.1 Objetivo e escopo

Tendo em vista a problemática apresentada pelo Hospital das Clínicas para alunos da Faculdade de Informática e Administração Paulista, o grupo AxcessTech decidiu desenvolver uma aplicação que diminua a taxa de absenteísmo para menos de 10%. O objetivo é desenvolver uma aplicação que facilite a utilização e navegação de usuários em domínios do HC. Por isso, esse sistema serve de apoio a um front-end capaz de realizar cadastro e login de paciente e funcionários, e permite a edição e remoção de paciente do banco de dados, por funcionários.

2 DOCUMENTAÇÃO TÉCNICA

2.1 Descrição das funcionalidades

2.1.1 Funcionalidades gerais

Para o cumprimento das exigências do instituto, o programa conta com funcionalidades voltadas para pacientes, como o sistema de login e cadastro facilitado e filtrado.

Para os funcionários, foi desenvolvido o sistema de login e cadastro para entrar na área do funcionário, o sistema de gerenciamento de pacientes, que permite apagar ou atualizar todas as informações de pacientes.

O CRUD está integrado por meios das funcionalidades descritas: Cadastros (create), visualização de dados dos pacientes (read), atualizações dos dados dos pacientes (update) e a exclusão completa de pacientes (delete)

2.2 Tabela de EndPoints

A seguir, estão os métodos EndPoints desenvolvidos para o projeto, separados pela classe Resource respectiva.

Todo o projeto está em produção no Render, localizado por: “<https://challenge-sprint4-java-2025.onrender.com>”

2.2.1 ResourceAreaFuncionario (“/funcionario”)

Método: **listar()**

- Tipo: GET
- Path: “/lista-pacientes”
- Códigos de resposta: 200, 500

Método: **deletar (DTOPacienteDelete dtoPacienteDelete)**

- Tipo: DELETE
- Path: “/deletar”
- Códigos de resposta: 200, 500, 404

Método: **recuperaPorId(@PathParam("id") int id)**

- Tipo: GET
- Path: “/recupera-por-id/{id}”
- Códigos de resposta: 200, 500

Método: **atualizar(DTOPacienteFull pacienteFull)**

- Tipo: PUT

- Path: “/atualizar-paciente”
- Códigos de resposta: 200, 500

2.2.2 ResourceCadastro (“/cadastro”)

Método: **criar (DTOPacienteCompleto dadosCompleto)**

- Tipo: POST
- Path: “/criarContaPaciente”
- Códigos de resposta: 201, 500

Método: **criar (DTOFuncionarioCompleto dadosCompleto)**

- Tipo: POST
- Path: “/criarContaFuncionario”
- Códigos de resposta: 201, 500

2.2.3 ResourceLogin (“/login”)

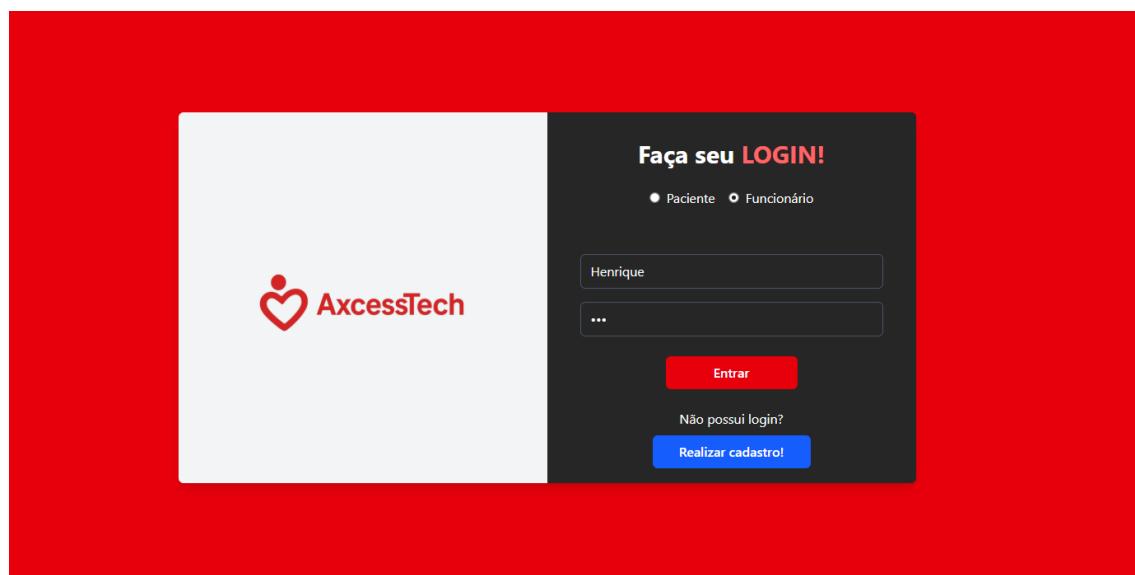
Método: **loginFuncionario (DTOContaFuncionario contaFuncionario)**

- Tipo: POST
- Path: “/paciente”
- Códigos de resposta: 200, 404, 500

Método: **loginPaciente (DTOContaPaciente contaPaciente)**

- Tipo: POST
- Path: “/funcionario”
- Códigos de resposta: 200, 404, 500

2.3 Telas do Sistema



Tela de Login, onde é aplicado o ResourceLogin.

Tela de cadastro, onde é aplicado o ResourceCadastro.

Lista de Pacientes

Nome	CPF	RG	Data de Nascimento	Grupo Sanguíneo	Ações
João	875.327.408-23	67.456.345-3	2025-02-09 00:00:00	AB+	Editar Excluir
Lucas	754.398.348-23	67.426.345-3	2025-02-12 00:00:00	AB+	Editar Excluir
Matheus	645.645.375-73	35.473.457-3	2008-04-07 00:00:00	O+	Editar Excluir

Pacientes cadastrados encontrados!

Tela da área do funcionário, onde é aplicada o método listar() e deletar() do ResourceAreaFuncionario.

Editar Paciente

Identificadores

ID Login Paciente	ID Paciente	ID Pessoa
11	11	17

Dados de Login

Joao	...
------	-----

Dados Pessoais

João	875.327.408-23
67.456.345-3	09/02/2025
Masculino	Pós-Graduação Incompleto

Dados Médicos

Diabetes	AB+
123	42

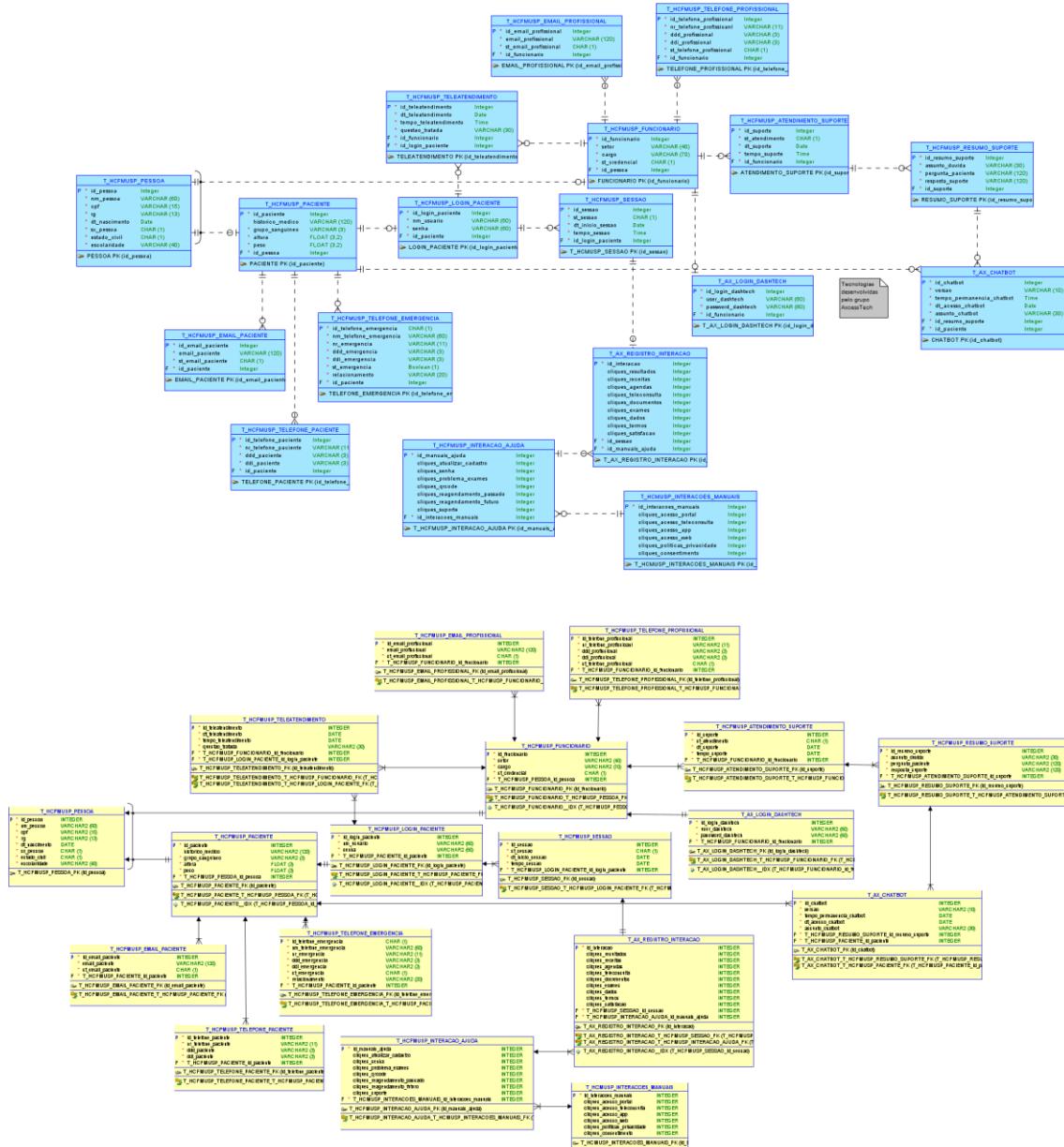
Ações

Salvar **Cancelar**

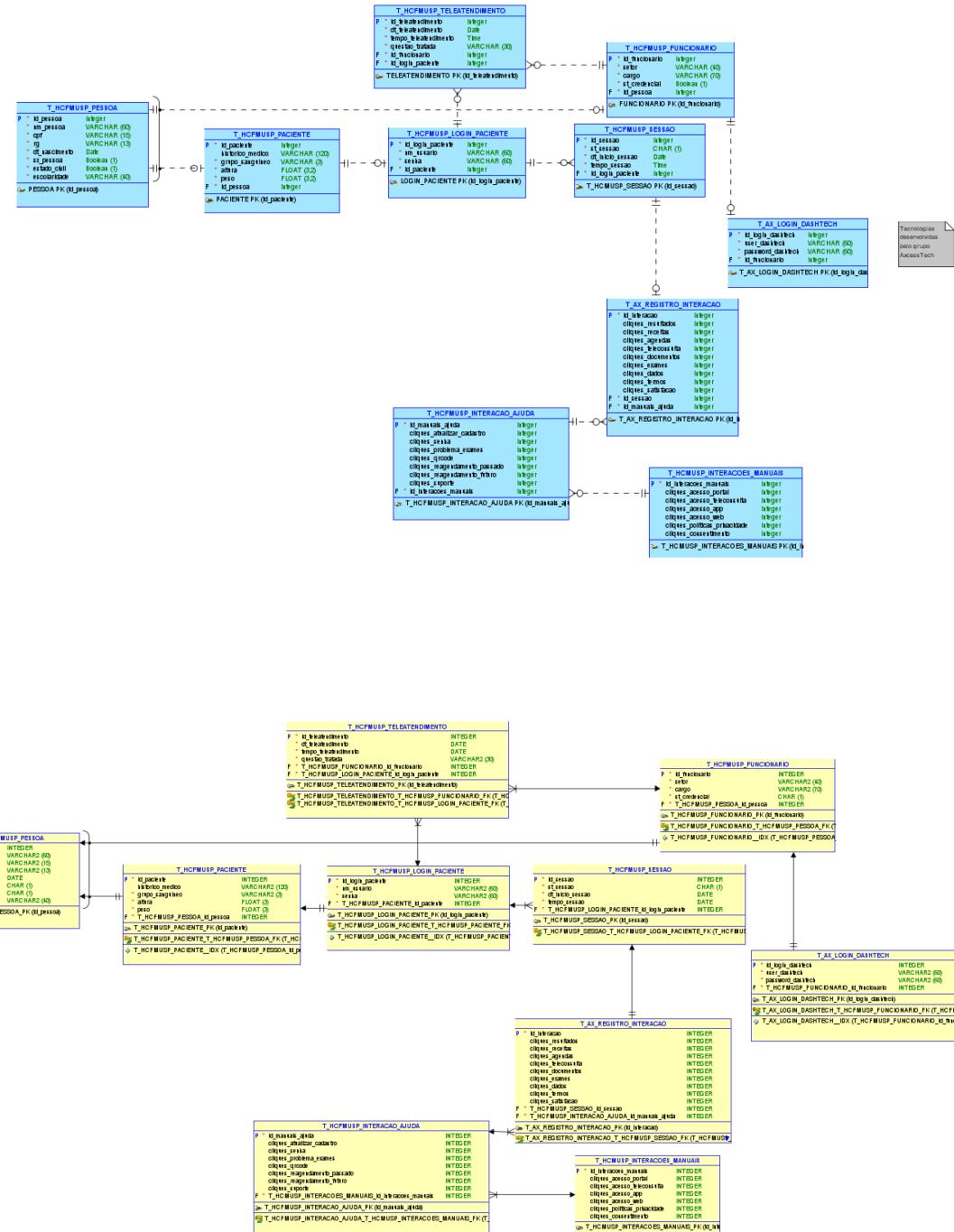
Tela de Editar Pacientes, que utiliza os métodos recuperarPorId() e atualizar(), do ResourceAreaFuncionario.

2.4 Modelo de Entidade-Relacionamento (MER)

2.4.1 Modelo geral



2.4.2 Modelo aplicado no sistema Java



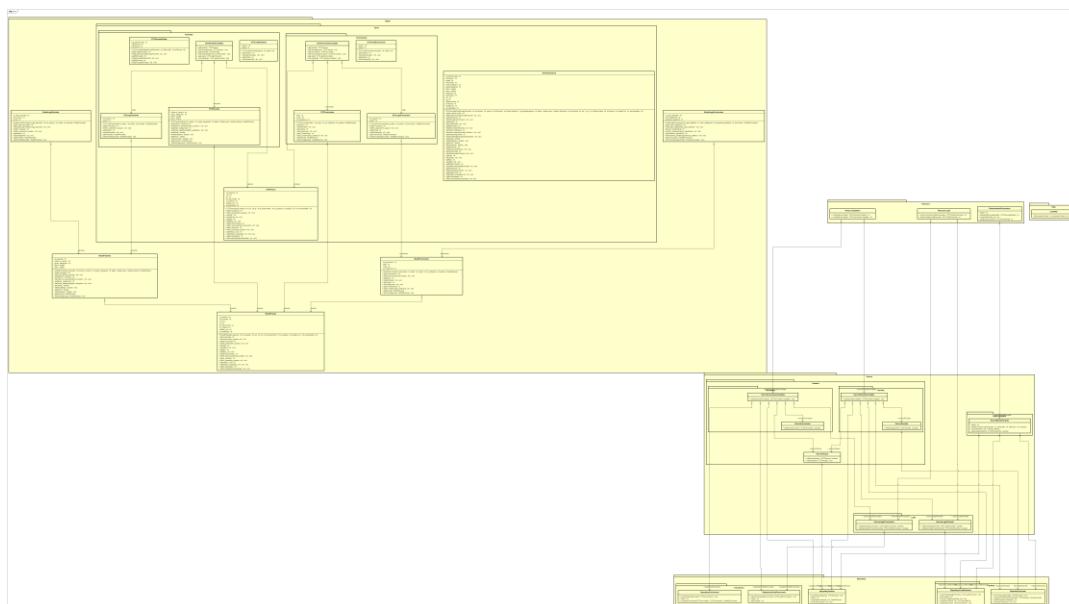
2.5 Diagrama de Classes

Para o desenvolvimento do projeto, um diagrama de classes UML foi criado com intuído de visualizar e estruturar como seriam realizadas as interações e funcionalidades presentes no programa.

Para isso, o “Astah”, um software especializado em modelagem, foi utilizado, desse modo, criando um ambiente com ferramentas semânticas para o desenvolvimento.

Os diagramas foram desenvolvidos por meio da importação automática no Astah.

Veja a diagramação abaixo:



Caso não seja possível a leitura, acesse o github e busque pelo arquivo “Class Diagram4.png”

Caminho no repositório: “code-with-quarkus/Class Diagram4.png”

3 COMPLEMENTARES

3.1 GitHub do Projeto

Link para o GitHub do projeto:
https://github.com/nfreitas2000/Challenge_Sprint4_Java_2025