

FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO ACLIMAÇÃO

Aguinel Junior Bento da Silva - rm564857

Leonardo Saavedra - rm562229

Vitor Mendes da Silva RM - rm565376

1TDSA

Sistema de Teleconsultas – Hospital das Clínicas

Domain Driven Design using Java

São Paulo

2025

FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO ACLIMAÇÃO

Aguinel Junior Bento da Silva - rm564857

Leonardo Saavedra - rm562229

Vitor Mendes da Silva RM - rm565376

1TDSA

Sistema de Teleconsultas – Hospital das Clínicas

Domain Driven Design using Java

Sprint 3 apresentado à Faculdade de Informática e Administração Aclimação como requisito de nota para a avaliação da disciplina Domain Driven Design using Java, sob a orientação do Professor Rafael Desiderio

São Paulo

2025

SUMÁRIO

| | | |
|----------|----------------------------------------------|---|
| 1 | Objetivo e escopo do projeto | 4 |
| 2 | Descrição das funcionalidades..... | 4 |
| 3 | Protótipo do sistema..... | 5 |
| 4 | Modelo de Entidade-Relacionamento (MER)..... | 5 |
| 5 | Diagrama de classes..... | 6 |
| 6 | Tabela de Endpoints (API Restful)..... | 7 |

1. Objetivo e escopo do projeto

O projeto "Sistema de Teleconsultas – Hospital das Clínicas" tem como objetivo centralizar e digitalizar o processo de consultas médicas remotas. A proposta é simular o funcionamento de um sistema real, aplicando conceitos de Programação Orientada a Objetos (POO) e Domain Driven Design (DDD) em Java. A modelagem do sistema foi feita com base no domínio hospitalar, focando em atendimentos por teleconsulta.

2. Descrição das funcionalidades

Cadastro de Médicos

- Permite registrar médicos no sistema com dados pessoais, contato, CRM, especialidade e turno.
- Possibilita atualização, exclusão e consulta dos dados cadastrados.

Cadastro de Pacientes

- Permite registrar pacientes com informações pessoais, contato, dados clínicos e médico responsável.
- Possibilita atualização, exclusão e consulta dos dados cadastrados.

Agendamento de Consultas

- Permite marcar consultas para pacientes, definindo data, hora, tipo de consulta (Presencial ou Teleconsulta) e status (Pendente, Confirmado ou Cancelado).
- As consultas ficam associadas a um médico e a um paciente.

Gerenciamento de Teleconsultas

- Permite cadastrar teleconsultas, incluindo link, plataforma, senha, horário de início e fim e observações.
- Cada teleconsulta é associada a um médico responsável.

Controle de Consultas

- Permite cadastrar, atualizar, excluir e listar consultas, com informações de paciente, médico, tipo, data e status.
- Possibilita visualizar histórico de consultas de cada paciente.

Relacionamento Médico-Paciente

- Cada paciente possui um médico responsável, garantindo acompanhamento contínuo.
- Permite identificar rapidamente o médico de cada paciente em consultas e teleconsultas.

Lógica de Negócio Integrada

- Garantia de integridade entre médicos, pacientes e consultas.
- Validações de status, tipo de consulta e vínculos entre entidades

3. Protótipo do sistema

Agendar Atendimento

Clique na parte do corpo onde você sente dor para agendar!

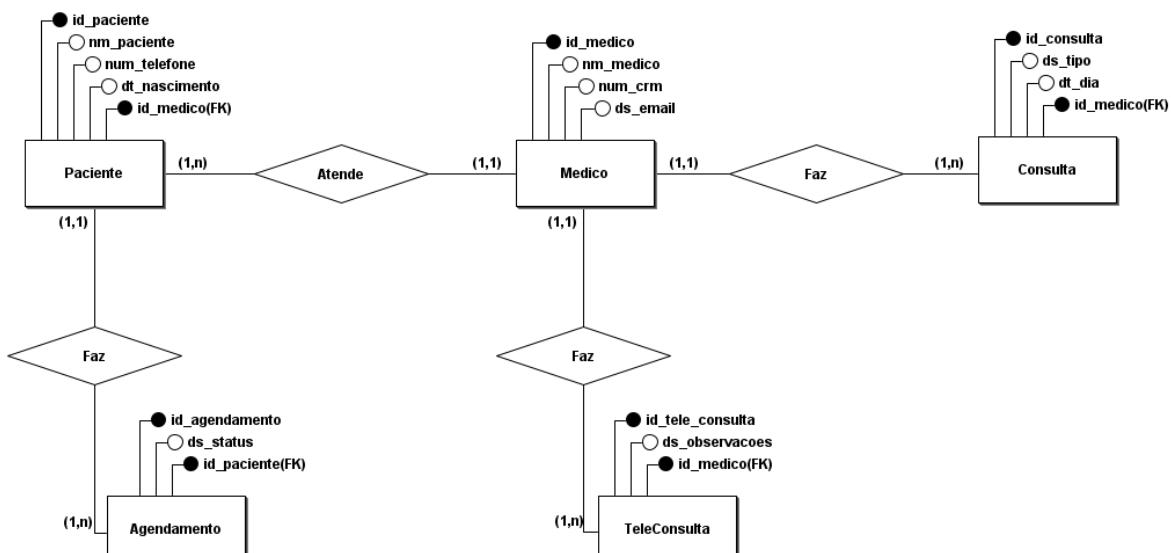


O protótipo foi desenvolvido para ilustrar a interação do usuário com a solução de teleconsulta digital. A interface principal é intuitiva, permitindo que o paciente descreva seus sintomas de maneira visual e detalhada, facilitando a comunicação com o médico durante a teleconsulta.

Tela Principal – Corpo Humano Interativo

- Descrição:
A tela apresenta um modelo de corpo humano na tela do computador. O usuário consegue clicar na região onde sente dor ou desconforto. Cada clique abre um pequeno formulário ou pop-up para detalhar o tipo de dor, intensidade, duração e outros sintomas relacionados.
- Objetivo:
Permitir que o paciente indique rapidamente as áreas afetadas e forneça informações detalhadas sobre sua condição sem a necessidade de digitar longos textos.

4. Modelo de Entidade-Relacionamento (MER)



5. Diagrama de classes

| Agendamento | Consulta | TeleConsulta |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> - id : int - status : String - paciente : Paciente <ul style="list-style-type: none"> + setId(id : int) : void + getId() : int + setStatus(status : int) : void + getStatus() : int + setPaciente(paciente : String) : void + getPaciente() : String | <ul style="list-style-type: none"> - id : int - data : LocalDate - tipo : String - medico : Medico <ul style="list-style-type: none"> + setId(id : int) : void + getId() : int + setData(data : LocalDate) : void + getData() : LocalDate + setTipo(tipo : String) : void + getTipo() : String + setMedico(medico : Medico) : void + getMedico() : Medico | <ul style="list-style-type: none"> - id : int - medico : Medico - observacoes : String <ul style="list-style-type: none"> + setId(id : int) : void + getId() : int + setMedico(medico : Medico) : void + getMedico() : Medico + setObservacoes(observacoes : String) : void + getObservacoes() : String |

| Medico | Paciente |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> - id : int - nome : String - email : String - crm : String <ul style="list-style-type: none"> + setId(id : int) : void + getId() : int + setNome(nome : String) : void + getNome() : String + setEmail(email : String) : void + getEmail() : String + setCrm(crm : String) : void + getCrm() : String | <ul style="list-style-type: none"> - id : int - nome : String - telefone : String - dataNascimento : LocalDate - medico : Medico <ul style="list-style-type: none"> + setId(id : int) : void + getId() : int + setNome(nome : String) : void + getNome() : String + setTelefone(telefone : String) : void + getTelefone() : String + setDataNascimento(dataNascimento : LocalDate) : void + getDataNascimento() : LocalDate + setMedico(medico : Medico) : void + getMedico() : Medico |

6. Tabela de endpoints (API Restful)

URIs
<http://localhost:8080/>

<http://localhost:8080/medicos>
<http://localhost:8080/pacientes>
<http://localhost:8080/agendamentos>
<http://localhost:8080/consultas>
<http://localhost:8080/teleconsultas>

Ao deletar ou atualizar é preciso inserir o id:
<http://localhost:8080/{nomedaclassse}/{id}>

exemplos:

<http://localhost:8080/medicos/1> - para mexer no médico cujo id é 1
<http://localhost:8080/consultas/23> - para mexer na consulta cujo id é 23

GET – POST – PUT – DELETE

Códigos de status de resposta esperados (tudo testando no Insumnia):

POST – 201 Created

PUT – 200 OK

GET – 200 OK

DELETE – 200 OK