Documentação da API — 4ª Sprint

Nome da Solução: ConectaReab — Clínica Médica

Integrantes:

• Gabriel Costa Solano, RM562325, 1TDSPV

• Kaiky Pereira Rodrigues Da Silva, RM564578, 1TDSPV

• Leandro Guarido, RM561760, 1TDSPV

Disciplina: Domain Driven Design using Java

Professor: Thiago Toshiyuki Izumi Yamamoto

2. Objetivo e Escopo do Projeto

Objetivo

O objetivo principal deste projeto é desenvolver o backend (API Restful) para um sistema de gestão de clínica médica. A solução foi projetada para organizar, simplificar e digitalizar o processo de atendimento. O sistema é focado na praticidade e na centralização das informações, permitindo o gerenciamento completo do cadastro de pacientes, do corpo clínico (médicos) e, o mais importante, do agendamento de consultas.

Escopo do Projeto (4ª Sprint)

O escopo desta entrega é a API Restful, que serve como a "porta de entrada" para o sistema e será consumida por um futuro frontend. A API está estruturada em três recursos principais, seguindo os princípios do Domain Driven Design (DDD): Pacientes, Médicos e Consultas.

Recursos:

• Pacientes: Endpoints para o gerenciamento completo do cadastro de pacientes.

• Médicos: Endpoints para o gerenciamento do cadastro de médicos e suas especialidades.

• Consultas: Endpoints para agendar, atualizar, listar e cancelar consultas, ligando um paciente a um médico.

3. Tabela de Endpoints (API Restful)

3.1. Recursos de Pacientes

Verbo HTTP	URI	Descrição da Ação	Códigos de Status Esperados
POST	/pacientes	Cadastra um novo paciente.	201 (Created): Sucesso. 400 (Bad Request): Dados inválidos (@Valid). 500 (Internal Server Error): Erro no servidor.
GET	/pacientes	Lista todos os pacientes cadastrados.	200 (OK): Sucesso. 500 (Internal Server Error): Erro no servidor.
GET	/pacientes/{id}	Busca um paciente específico pelo ID.	200 (OK): Sucesso. 404 (Not Found): Paciente não encontrado. 500 (Internal Server Error): Erro no servidor.
PUT	/pacientes/{id}	Atualiza os dados de um paciente existente.	200 (OK): Sucesso. 400 (Bad Request): Dados inválidos (@Valid). 404 (Not Found): Paciente não encontrado. 500 (Internal Server Error): Erro no servidor.
DELETE	/pacientes/{id}	Remove um paciente do sistema.	204 (No Content): Sucesso (sem corpo). 404 (Not Found): Paciente não encontrado. 500 (Internal Server Error): Erro no servidor.

3.2. Recursos de Médicos

Verbo HTTP	URI	Descrição da Ação	Códigos de Status Esperados
POST	/medicos	Cadastra um novo médico.	201 (Created): Sucesso. 400 (Bad Request): Dados inválidos (@Valid). 500 (Internal Server Error): Erro no servidor.
GET	/medicos	Lista todos os médicos cadastrados.	200 (OK): Sucesso. 500 (Internal Server Error): Erro no servidor.
GET	/medicos/{id}	Busca um médico específico pelo ID.	200 (OK): Sucesso. 404 (Not Found): Médico não encontrado. 500 (Internal Server Error): Erro no servidor.
PUT	/medicos/{id}	Atualiza os dados de um médico existente.	200 (OK): Sucesso. 400 (Bad Request): Dados inválidos (@Valid). 404 (Not Found): Médico não encontrado. 500 (Internal Server Error): Erro no servidor.
DELETE	/medicos/{id}	Remove um médico do sistema.	204 (No Content): Sucesso (sem corpo). 404 (Not Found): Médico não encontrado. 500 (Internal Server Error): Erro no servidor.

3.3. Recursos de Consultas

Verbo HTTP	URI	Descrição da Ação	Códigos de Status Esperados

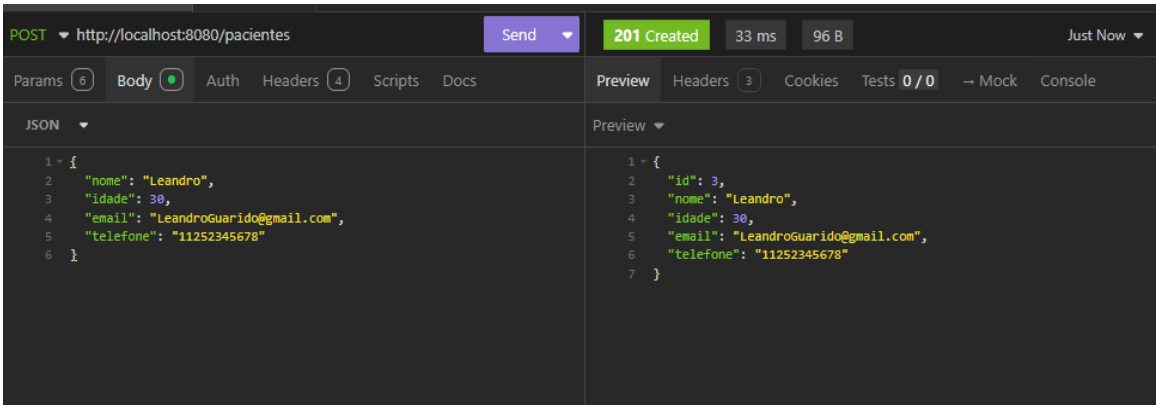
POST	/consultas	Agenda (cadastra) uma nova consulta.	201 (Created): Sucesso. 400 (Bad Request): Dados inválidos (@Valid). 500 (Internal Server Error): Erro no servidor.
GET	/consultas	Lista todas as consultas agendadas.	200 (OK): Sucesso. 500 (Internal Server Error): Erro no servidor.
GET	/consultas/{id}	Busca uma consulta específica pelo ID.	200 (OK): Sucesso. 404 (Not Found): Consulta não encontrada. 500 (Internal Server Error): Erro no servidor.
PUT	/consultas/{id}	Atualiza os dados de uma consulta existente.	200 (OK): Sucesso. 400 (Bad Request): Dados inválidos (@Valid). 404 (Not Found): Consulta não encontrada. 500 (Internal Server Error): Erro no servidor.
DELETE	/consultas/{id}	Cancela (remove) uma consulta do sistema.	204 (No Content): Sucesso (sem corpo). 404 (Not Found): Consulta não encontrada. 500 (Internal Server Error): Erro no servidor.

4. Protótipo – Testes de Funcionalidade (Postman)

4.1. Criação de Recurso (POST /pacientes)

Descrição: Teste de cadastro de um novo paciente. Um JSON com os dados do paciente é enviado no corpo da requisição POST.

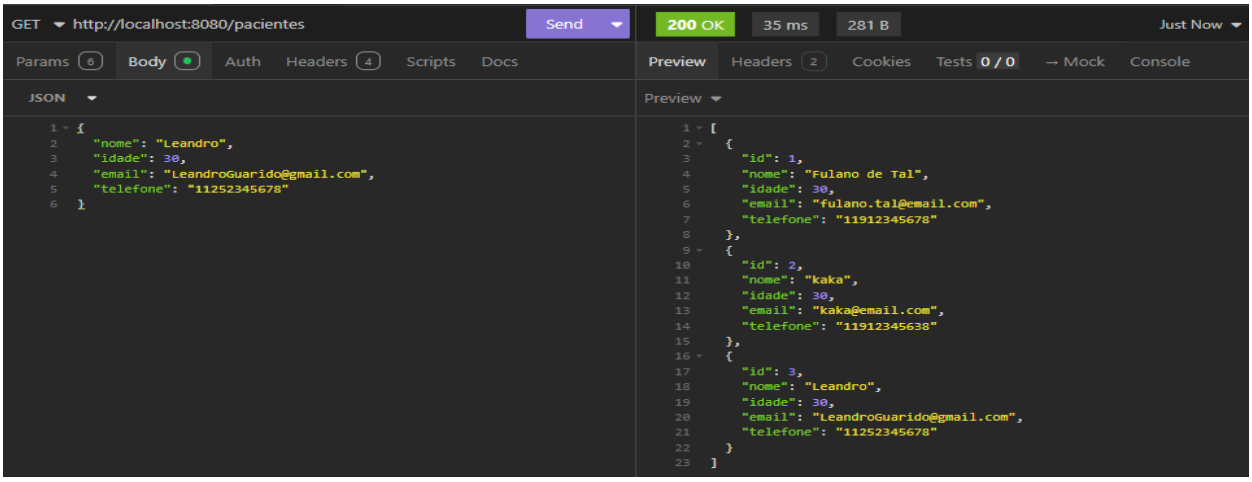
Resultado: A API processa os dados, salva no banco e retorna corretamente o status 201 Created, junto com o JSON do novo paciente criado (agora com um id).



4.2. Listagem de Recursos (GET /pacientes)

Descrição: Teste de listagem de todos os pacientes cadastrados. Uma requisição GET é feita para o endpoint principal.

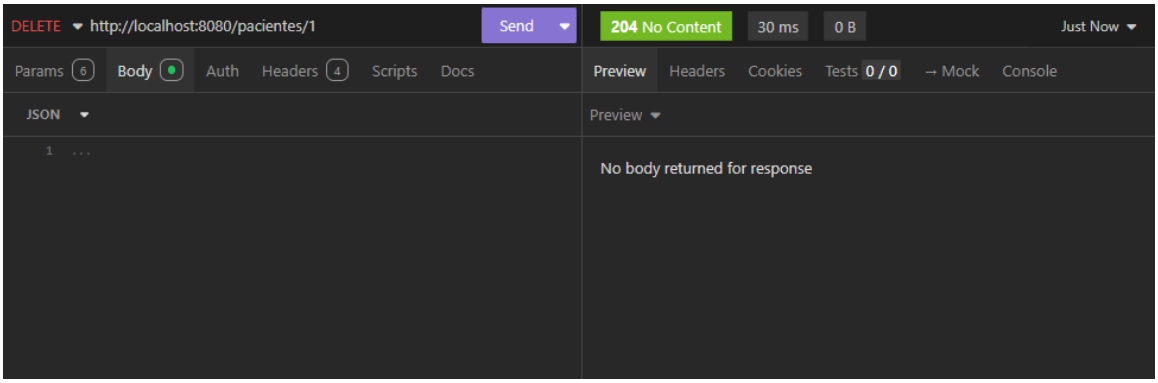
Resultado: A API consulta o banco de dados e retorna o status 200 OK, junto com uma lista em JSON contendo todos os pacientes que foram cadastrados.



4.3. Remoção de Recurso (DELETE /pacientes/{id})

Descrição: Teste de exclusão de um paciente. Uma requisição DELETE é enviada para a URL do recurso, especificando o id (ex.: /pacientes/1).

Resultado: A API localiza e remove o paciente do banco de dados e retorna o status 204 No Content.



5. Modelo de Entidade-Relacionamento (MER)

5.1. Script SQL (DDL)

O script abaixo define a estrutura completa do banco de dados.

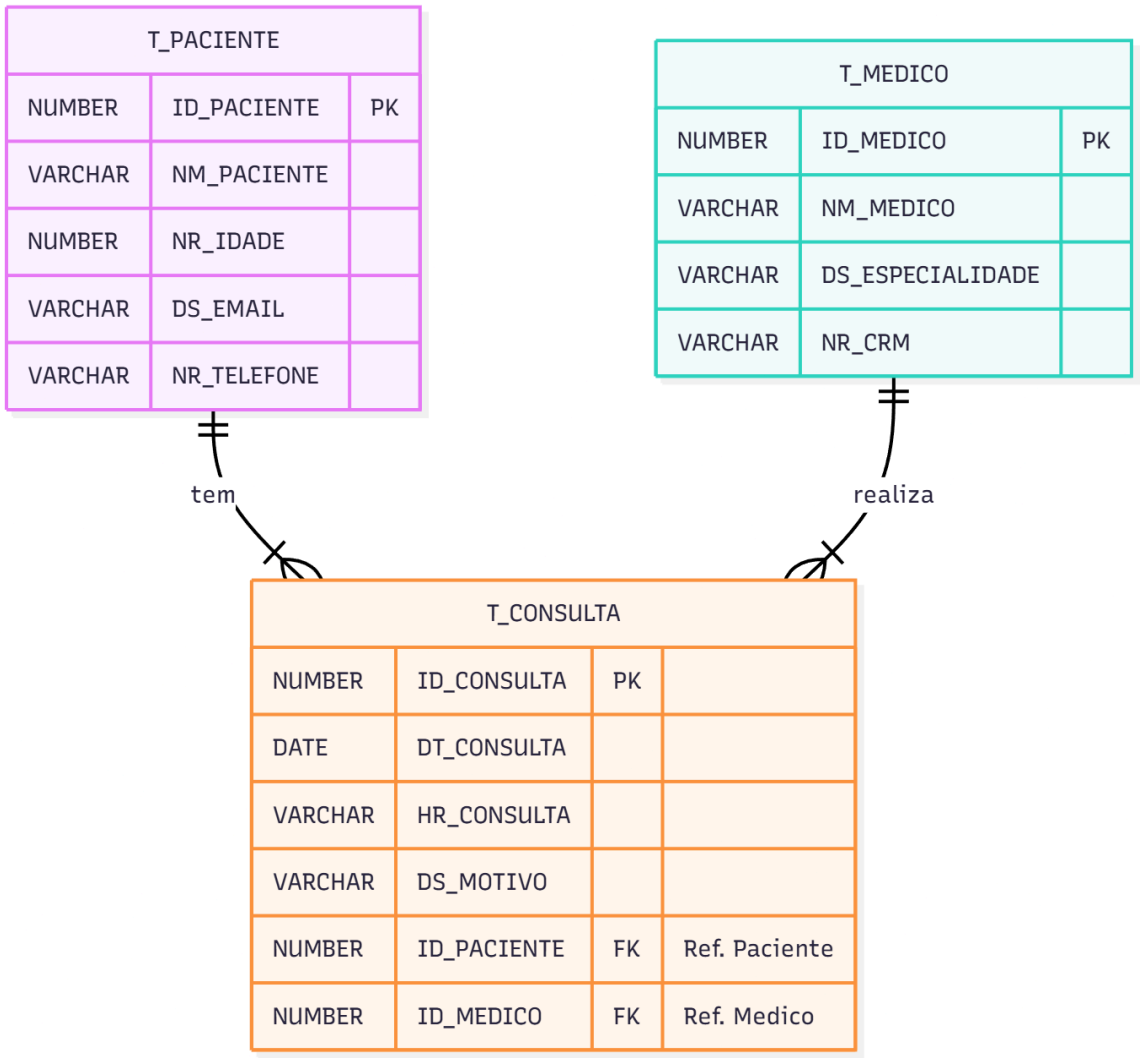
```
-- Tabela T_PACIENTE
CREATE TABLE T_PACIENTE (
  ID_PACIENTE NUMBER(10) GENERATED BY DEFAULT AS IDENTITY,
  NM_PACIENTE VARCHAR2(255) NOT NULL,
  NR_IDADE NUMBER(3) NULL,
  DS_EMAIL VARCHAR2(255) NOT NULL,
  NR_TELEFONE VARCHAR2(20) NULL,
  CONSTRAINT T_PACIENTE_PK PRIMARY KEY (ID_PACIENTE)
);

-- Tabela T_MEDICO
CREATE TABLE T_MEDICO (
  ID_MEDICO NUMBER(10) GENERATED BY DEFAULT AS IDENTITY,
  NM_MEDICO VARCHAR2(255) NOT NULL,
  DS_ESPECIALIDADE VARCHAR2(255) NOT NULL,
  NR_CRM VARCHAR2(50) NOT NULL,
  CONSTRAINT T_MEDICO_PK PRIMARY KEY (ID_MEDICO)
);

-- Tabela T_CONSULTA
CREATE TABLE T_CONSULTA (
  ID_CONSULTA NUMBER(10) GENERATED BY DEFAULT AS IDENTITY,
```

```
  DT_CONSULTA DATE NOT NULL,
  HR_CONSULTA VARCHAR2(10) NOT NULL,
  DS_MOTIVO VARCHAR2(500) NULL,
  ID_PACIENTE NUMBER(10) NOT NULL,
  ID_MEDICO NUMBER(10) NOT NULL,
  CONSTRAINT T_CONSULTA_PK PRIMARY KEY (ID_CONSULTA),
  CONSTRAINT T_CONSULTA_PACIENTE_FK
    FOREIGN KEY (ID_PACIENTE)
    REFERENCES T_PACIENTE (ID_PACIENTE),
  CONSTRAINT T_CONSULTA_MEDICO_FK
    FOREIGN KEY (ID_MEDICO)
    REFERENCES T_MEDICO (ID_MEDICO)
);
```

5.2. Diagrama MER (Visual)



6. Diagrama de Classes Atualizado

Estrutura das classes Java (POJOs) correspondentes às entidades do domínio (Paciente, Medico e Consulta) e suas associações.