

HOSPITALTECH - Solução Digital Integrada para Gestão de Pacientes e Consultas

Pedro Henrique Luiz Alves Duarte – RM563405

Guilherme Macedo Martins – RM562396

Equipe: BDB

São Paulo – Novembro de 2025

1. Objetivo e Escopo do Projeto

O projeto HospitalTech tem como objetivo principal o desenvolvimento de uma API Restful completa e robusta, seguindo o padrão de arquitetura Domain Driven Design (DDD). O escopo da solução abrange o gerenciamento centralizado de entidades essenciais para um sistema de saúde, como Pacientes, Médicos, Especialidades e Consultas médicas. A solução é implementada em Java utilizando o framework Quarkus e se conecta a um banco de dados Oracle para persistência real dos dados. O foco é fornecer uma camada de serviço otimizada e escalável, capaz de suportar o fluxo de atendimentos e facilitar a integração com diversas aplicações front-end.

2. Breve Descrição das Principais Funcionalidades da Solução

A solução HospitalTech oferece uma API moderna e escalável, com uma arquitetura bem definida que separa as responsabilidades em camadas: Model (entidades de dados), DAO (acesso a dados), BO (regras de negócio) e Resource (endpoints da API).

Funcionalidade	Descrição
CRUD Completo	Operações de criação, leitura, atualização e exclusão para todas as entidades principais.
Persistência de Dados	Conexão com banco Oracle garantindo integridade e persistência.
API RESTful Padronizada	Endpoints com boas práticas REST e verbos HTTP adequados.
Documentação Interativa	Swagger UI para visualização e testes dos endpoints.
Tratamento de Exceções	Respostas JSON e status HTTP apropriados para erros.
CORS Habilitado	Permite consumo da API por diferentes domínios.
Deploy Funcional	Aplicação hospedada e funcional no Render Cloud.

3. Tabela de Endpoints (API RESTful)

Recurso	URI (Caminho)	Verbo HTTP	Descrição	Códigos de Status Esperados
Consulta	/consulta	GET	Lista todas as consultas.	200 OK
Consulta	/consulta	POST	Cria uma nova consulta.	201 Created, 500 Internal Server Error
Consulta	/consulta	PUT	Atualiza uma consulta existente.	200 OK, 404 Not Found
Consulta	/consulta/{id}	DELETE	Remove uma consulta pelo ID.	200 OK, 404 Not Found

Convênio	/convenio	GET	Lista todos os convênios.	200 OK
Convênio	/convenio	POST	Cria um novo convênio.	201 Created, 500 Internal Server Error
Convênio	/convenio	PUT	Atualiza um convênio existente.	200 OK, 404 Not Found
Convênio	/convenio/{id}	DELETE	Remove um convênio pelo ID.	200 OK, 404 Not Found
Endereço	/endereco	GET	Lista todos os endereços.	200 OK
Endereço	/endereco	POST	Cria um novo endereço.	201 Created, 500 Internal Server Error
Endereço	/endereco	PUT	Atualiza um endereço existente.	200 OK, 404 Not Found
Endereço	/endereco/{id}	DELETE	Remove um endereço pelo ID.	200 OK, 404 Not Found
Especialidades	/especialidades	GET	Lista todas as especialidades.	200 OK
Especialidades	/especialidades	POST	Cria uma nova especialidade.	201 Created, 500 Internal Server Error
Especialidades	/especialidades	PUT	Atualiza uma especialidade existente.	200 OK, 404 Not Found
Especialidades	/especialidades/{id}	DELETE	Remove uma especialidade pelo ID.	200 OK, 404 Not Found
Médico	/medico	GET	Lista todos os médicos.	200 OK
Médico	/medico	POST	Cria um novo médico.	201 Created, 500 Internal Server Error
Médico	/medico	PUT	Atualiza um médico existente.	200 OK, 404 Not Found
Médico	/medico/{id}	DELETE	Remove um médico pelo ID.	200 OK, 404 Not Found
Paciente	/paciente	GET	Lista todos os pacientes.	200 OK
Paciente	/paciente	POST	Cria um novo paciente.	201 Created, 500 Internal Server Error
Paciente	/paciente	PUT	Atualiza um paciente existente.	200 OK, 404 Not Found
Paciente	/paciente/{id}	DELETE	Remove um	200 OK, 404

			paciente pelo ID.	Not Found
Prontuário	/prontuario	GET	Lista todos os prontuários.	200 OK
Prontuário	/prontuario	POST	Cria um novo prontuário.	201 Created, 500 Internal Server Error
Prontuário	/prontuario	PUT	Atualiza um prontuário existente.	200 OK, 404 Not Found
Prontuário	/prontuario/{id}	DELETE	Remove um prontuário pelo ID.	200 OK, 404 Not Found
Usuário	/usuario	GET	Lista todos os usuários.	200 OK
Usuário	/usuario	POST	Cria um novo usuário.	201 Created, 500 Internal Server Error
Usuário	/usuario	PUT	Atualiza um usuário existente.	200 OK, 404 Not Found
Usuário	/usuario/{id}	DELETE	Remove um usuário pelo ID.	200 OK, 404 Not Found

4. Protótipo – Prints das Telas Implementadas (Swagger UI)



Figura 1 – Endpoints exibidos no Swagger UI (Captura de tela 2025-11-05 161139.png)


Endereco Resource			^
GET	/endereco		▼
PUT	/endereco		▼
POST	/endereco		▼
DELETE	/endereco/{id}		▼
Especialidades Resource			^
GET	/especialidades		▼
PUT	/especialidades		▼
POST	/especialidades		▼
DELETE	/especialidades/{id}		▼

Figura 2


Medico Resource			^
GET	/medico		▼
PUT	/medico		▼
POST	/medico		▼
DELETE	/medico/{id}		▼
Paciente Resource			^
GET	/paciente		▼
PUT	/paciente		▼
POST	/paciente		▼
DELETE	/paciente/{id}		▼

Figura 3

Prontuario Resource			^
GET	/prontuario		▼
PUT	/prontuario		▼
POST	/prontuario		▼
DELETE	/prontuario/{id}		▼
Usuario Resource			^
GET	/usuario		▼
PUT	/usuario		▼
POST	/usuario		▼
DELETE	/usuario/{id}		▼

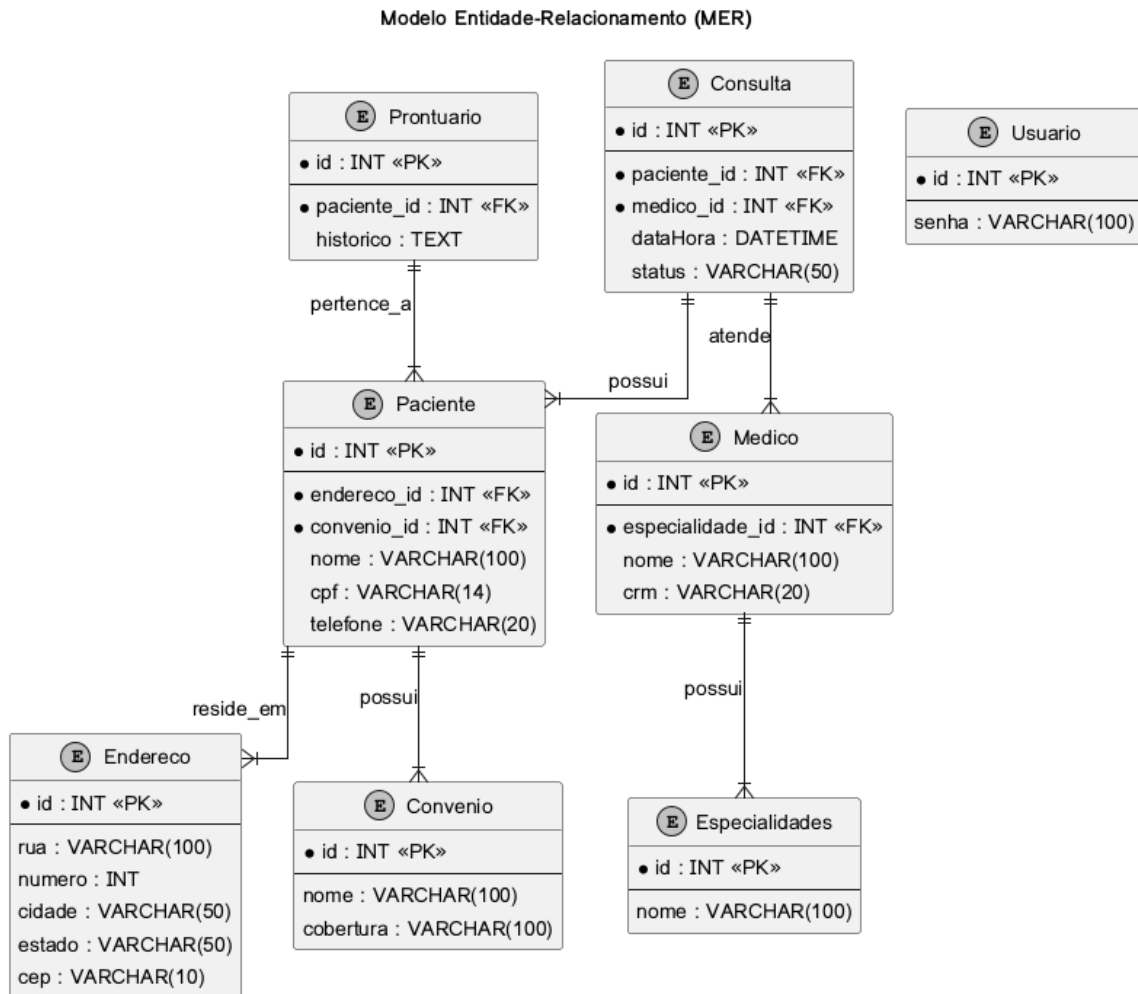
Figura 4

```
Consulta ▾ {  
  id                > [...]   
  paciente          Paciente > {...}  
  medico            Medico > {...}  
  dataHora          LocalDateTime > [...]  
  status            > [...]  
}
```

```
Convenio ▾ {  
  id                > [...]   
  nome              > [...]   
  cobertura         > [...]   
}
```

```
Endereco ▾ {  
  id                > [...]   
  rua               > [...]   
  numero           > [...]   
  cidade           > [...]   
  estado           > [...]   
  cep              > [...]   
}
```

5. Modelo Entidade-Relacionamento (MER)



6. Diagrama de classes

