

# Cuida Fácil

## Domain Driven Design Using Java

1TDSRH Gustavo Okada RM563428

07 de Novembro de 2025

## **Sumário**

1.	Objetivo e Escopo do Projeto .....	3
2.	Descrição das Funcionalidades .....	4
3.	Modelo de Entidade-Relacionamento (MER).....	13
4.	Diagrama de Classes.....	14
5.	Imagens do Sistema .....	15
6.	Tecnologias usadas .....	19
7.	Procedimentos para Rodar a Aplicação .....	19
8.	Testando API .....	20
9.	Considerações finais .....	20

# 1. Objetivo e Escopo do Projeto

O **Cuida Fácil** é uma solução digital desenvolvida para facilitar o agendamento e gerenciamento de consultas médicas em unidades de saúde. O sistema visa:

- **Otimizar** o processo de agendamento de consultas médicas
- **Centralizar** informações de pacientes, médicos, unidades e especialidades
- **Melhorar** a experiência do usuário através de uma API RESTful moderna
- **Integrar** um chatbot inteligente para suporte aos pacientes
- **Garantir** rastreabilidade através de protocolos únicos de atendimento

O escopo contempla um backend completo em Java utilizando Jakarta EE, com persistência em banco de dados Oracle, seguindo os padrões MVC e arquitetura em camadas.

## **2. Descrição das Funcionalidades**

### **1. Gestão de Pacientes**

- Cadastro completo com validação de CPF único
- Restrição de idade mínima (18 anos)
- Consulta por ID com tratamento de exceções
- Atualização e remoção de registros

### **2. Gestão de Médicos**

- Cadastro com CRM obrigatório
- Associação com especialidades
- Upload de foto de perfil (URL)
- CRUD completo

### **3. Gestão de Unidades de Saúde**

- Cadastro de endereços e horários de funcionamento
- Informações de contato (telefone, CEP)
- Imagens das unidades
- Localização geográfica

### **4. Gestão de Especialidades Médicas**

- Catálogo de especialidades disponíveis
- Descrições detalhadas
- Imagens ilustrativas
- Vinculação com serviços

### **5. Gestão de Serviços**

- Serviços médicos por especialidade
- Organização hierárquica
- Associação com consultas

## 6. Sistema de Consultas

- Agendamento com antecedência mínima de 2 horas
- Geração automática de protocolo único
- Status de acompanhamento (AGENDADA, REAGENDADA, CONCLUÍDA, CANCELADA)
- Tipo de atendimento (PRESENCIAL, TELECONSULTA)
- Consultas detalhadas com JOIN de todas as entidades relacionadas

## 7. Chatbot Inteligente

- Registro de interações com pacientes
- Identificação de intenções do usuário
- Respostas automáticas personalizadas
- Histórico de atendimentos

# PACIENTES

**Recurso:** /pacientes

*Operações*

- **GET /pacientes**
  - **Descrição:** Lista todos os pacientes cadastrados.
  - **Response:** List<PacienteTO> (200 OK)
- **GET /pacientes/{id}**
  - **Descrição:** Busca paciente por ID.
  - **Response:** PacienteTO (200 OK)
- **POST /pacientes**
  - **Descrição:** Cadastra novo paciente.
  - **Request Body:** PacienteTO
  - **Response:** PacienteTO (201 CREATED)
- **PUT /pacientes/{id}**
  - **Descrição:** Atualiza paciente existente.
  - **Request Body:** PacienteTO
  - **Response:** PacienteTO (200 OK)
- **DELETE /pacientes/{id}**
  - **Descrição:** Remove paciente.
  - **Response:** (204 NO CONTENT)

## **Regras de Negócio e Validações**

- **CPF** deve ser único (não permite duplicação).
- **Idade** mínima de 18 anos.
- **Campos Obrigatórios:** CPF, Nome, Data de Nascimento.

### ***Exemplo de Request Body (POST)***

JSON

```
{  
  "cpf": "12345678901",  
  "nome": "João Silva",  
  "telefone": "11987654321",  
  "email": "joao@email.com",  
  "dataNascimento": "1990-05-15",  
  "cep": "01310100"  
}
```

# **MÉDICOS**

## **Recurso: /medicos**

### ***Operações***

- **GET /medicos:** Lista todos os médicos. (Response: List<MedicoTO>)
- **GET /medicos/{id}:** Busca médico por ID. (Response: MedicoTO)
- **POST /medicos:** Cadastra novo médico. (Request/Response: MedicoTO, 201 CREATED)
- **PUT /medicos/{id}:** Atualiza médico existente. (Request/Response: MedicoTO, 200 OK)
- **DELETE /medicos/{id}:** Remove médico. (Response: 204 NO CONTENT)

### ***Exemplo de Request Body (POST)***

JSON

```
{  
  "crm": "123456-SP",  
  "nome": "Dra. Maria Santos",  
  "urlImagemMedico": "https://example.com/foto.jpg"
```

}

## UNIDADES

### Recurso: /unidades

#### *Operações*

- **GET /unidades:** Lista todas as unidades. (Response: List<UnidadeTO>)
- **GET /unidades/{id}:** Busca unidade por ID. (Response: UnidadeTO)
- **POST /unidades:** Cadastra nova unidade. (Request/Response: UnidadeTO, 201 CREATED)
- **PUT /unidades/{id}:** Atualiza unidade existente. (Request/Response: UnidadeTO, 200 OK)
- **DELETE /unidades/{id}:** Remove unidade. (Response: 204 NO CONTENT)

#### *Exemplo de Request Body (POST)*

JSON

```
{  
    "cdUnidade": "UN-001",  
    "endereco": "Av. Paulista, 1000",  
    "telefone": "1133334444",  
    "horario": "08:00 - 18:00",  
    "cep": "01310100",  
    "urlImagemUnidades": "https://example.com/unidade.jpg"  
}
```

## ESPECIALIDADES

### Recurso: /especialidades

#### *Operações*

- **GET /especialidades:** Lista todas as especialidades. (Response: List<EspecialidadeTO>)

- **GET /especialidades/{id}**: Busca especialidade por ID. (Response: EspecialidadeTO)
- **POST /especialidades**: Cadastra nova especialidade. (Request/Response: EspecialidadeTO, 201 CREATED)
- **PUT /especialidades/{id}**: Atualiza especialidade. (Request/Response: EspecialidadeTO, 200 OK)
- **DELETE /especialidades/{id}**: Remove especialidade. (Response: 204 NO CONTENT)

#### ***Exemplo de Request Body (POST)***

JSON

```
{  
  "nome": "Cardiologia",  
  "descricao": "Especialidade médica focada no coração",  
  "urlImagemEspecialidades": "https://example.com/cardio.jpg"  
}
```

# **SERVIÇOS**

## **Recurso: /servicos**

### ***Operações***

- **GET /servicos**: Lista todos os serviços. (Response: List<ServicoTO>)
- **GET /servicos/{id}**: Busca serviço por ID. (Response: ServicoTO)
- **POST /servicos**: Cadastra novo serviço. (Request/Response: ServicoTO, 201 CREATED)
- **PUT /servicos/{id}**: Atualiza serviço existente. (Request/Response: ServicoTO, 200 OK)
- **DELETE /servicos/{id}**: Remove serviço. (Response: 204 NO CONTENT)

#### ***Exemplo de Request Body (POST)***

JSON

```
{  
  "nome": "Eletrocardiograma",  
  "idEspecialidade": 1
```

}

## CONSULTAS

**Recurso:** /consultas

*Operações*

- **GET /consultas:** Lista todas as consultas (detalhadas). (Response: List<ConsultaDetalhadaTO>)
- **GET /consultas/{id}:** Busca consulta detalhada por ID. (Response: ConsultaDetalhadaTO)
- **POST /consultas:** Agenda nova consulta. (Request/Response: ConsultaTO, 201 CREATED)
- **PUT /consultas/{id}:** Atualiza consulta existente. (Request/Response: ConsultaTO, 200 OK)
- **DELETE /consultas/{id}:** Cancela consulta. (Response: 204 NO CONTENT)

*Regras de Negócio e Domínio*

- **Agendamento:** Antecedência mínima de 2 horas.
- **Protocolo:** Geração automática no formato: CFYYYYMMDDHHMMSS.
- **Status Padrão:** AGENDADA.
- **Status Permitidos:** AGENDADA, REAGENDADA, CONCLUIDA, CANCELADA.
- **Tipos de Atendimento:** PRESENCIAL, TELECONSULTA.

*Exemplo de Request Body (POST)*

JSON

```
{  
  "dataConsulta": "2025-11-20T14:30:00",  
  "status": "AGENDADA",  
  "tipoAtendimento": "PRESENCIAL",  
  "idPaciente": 1,  
  "idMedico": 1,  
  "idUnidade": 1,  
  "idEspecialidade": 1}
```

}

### ***Exemplo de Response (GET - Consulta Detalhada)***

JSON

```
{  
    "idConsulta": 1,  
    "protocolo": "CF20251115143000",  
    "dataConsulta": "2025-11-20T14:30:00",  
    "status": "AGENDADA",  
    "tipoAtendimento": "PRESENCIAL",  
    "idPaciente": 1,  
    "nomePaciente": "João Silva",  
    "idMedico": 1,  
    "nomeMedico": "Dra. Maria Santos",  
    "crmMedico": "123456-SP",  
    "idUnidade": 1,  
    "nomeUnidade": "UN-001",  
    "enderecoUnidade": "Av. Paulista, 1000",  
    "idEspecialidade": 1,  
    "nomeEspecialidade": "Cardiologia"  
}
```

## **CHATBOT**

### **Recurso: /chatbot**

#### ***Operações***

- **GET /chatbot:** Lista todos os atendimentos. (Response: List<ChatbotTO>)
- **GET /chatbot/{id}:** Busca atendimento por ID. (Response: ChatbotTO)
- **POST /chatbot:** Registra novo atendimento. (Request/Response: ChatbotTO, 201 CREATED)
- **PUT /chatbot/{id}:** Atualiza atendimento. (Request/Response: ChatbotTO, 200 OK)
- **DELETE /chatbot/{id}:** Remove atendimento. (Response: 204 NO CONTENT)

## **Regras de Negócio e Domínio**

- **Data/Hora:** Registro automático se não informado.
- **Vinculação:** Obrigatória com paciente (idPaciente).

### **Exemplo de Request Body (POST)**

JSON

```
{  
  "idPaciente": 1,  
  "intencaoUsuario": "agendar_consulta",  
  "textoResposta": "Consulta agendada com sucesso para o dia 20/11 às 14h30"  
}
```

A documentação agora está formatada em um estilo de texto/Markdown, mais adequado para documentos técnicos, e o título foi corrigido.

Gostaria de detalhar ou gerar o mesmo formato de docum

### **Regras de Negócio:**

- Antecedência mínima de 2 horas para agendamento
- Geração automática de protocolo no formato: CFYYYYMMDDHHMMSS
- Status padrão: AGENDADA
- Status permitidos: AGENDADA, REAGENDADA, CONCLUIDA, CANCELADA
- Tipos de atendimento: PRESENCIAL, TELECONSULTA

### **Exemplo de Request Body (POST):**

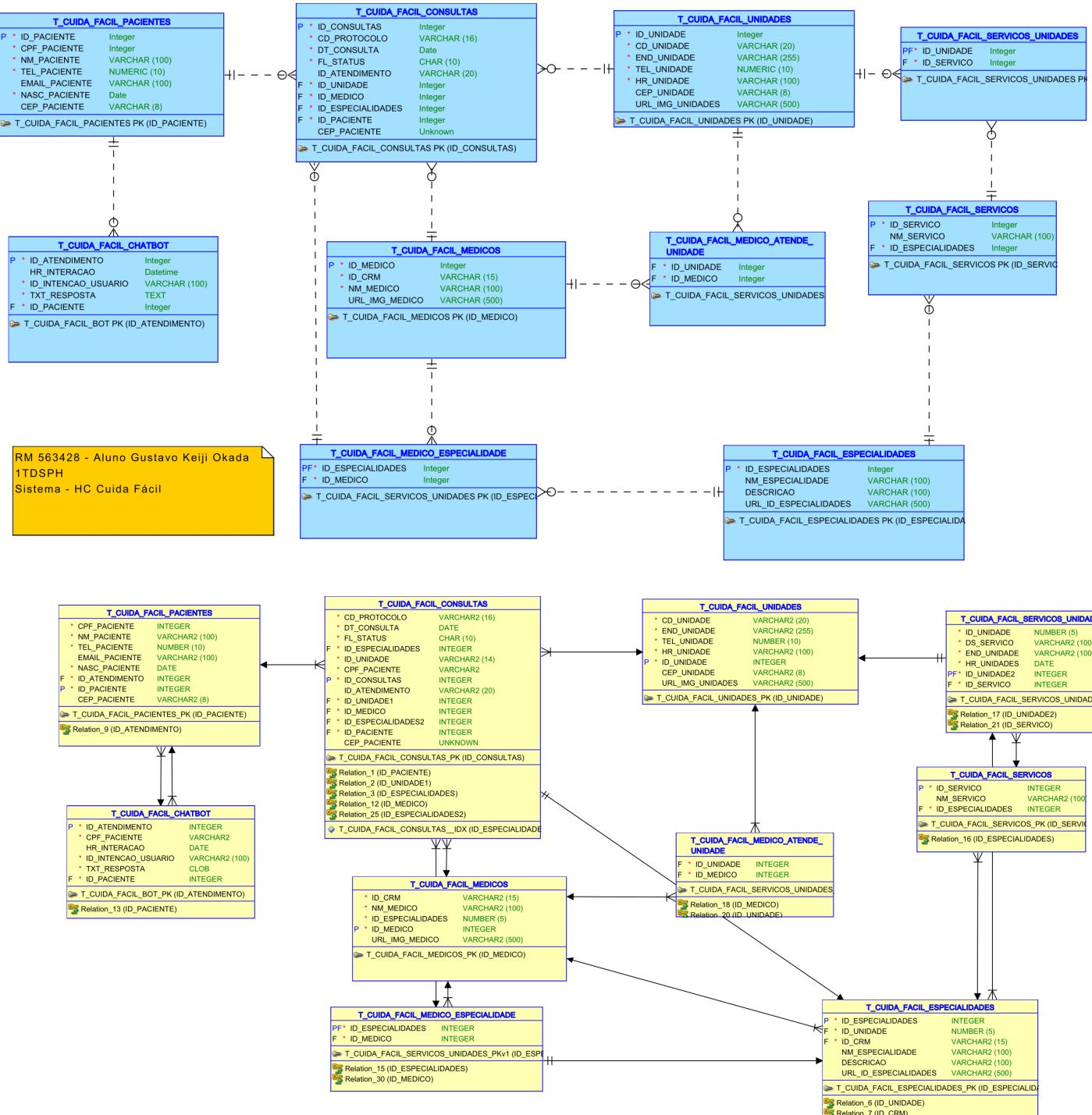
```
{  
  "dataConsulta": "2025-11-20T14:30:00",  
  "status": "AGENDADA",  
  "tipoAtendimento": "PRESENCIAL",  
  "idPaciente": 1,  
  "idMedico": 1,  
  "idUnidade": 1,  
  "idEspecialidade": 1  
}
```

### **Exemplo de Response (GET - Consulta Detalhada):**

```
{  
    "idConsulta": 1,  
    "protocolo": "CF20251115143000",  
    "dataConsulta": "2025-11-20T14:30:00",  
    "status": "AGENDADA",  
    "tipoAtendimento": "PRESENCIAL",  
    "idPaciente": 1,  
    "nomePaciente": "João Silva",  
    "idMedico": 1,  
    "nomeMedico": "Dra. Maria Santos",  
    "crmMedico": "123456-SP",  
    "idUnidade": 1,  
    "nomeUnidade": "UN-001",  
    "enderecoUnidade": "Av. Paulista, 1000",  
    "idEspecialidade": 1,  
    "nomeEspecialidade": "Cardiologia"  
}
```

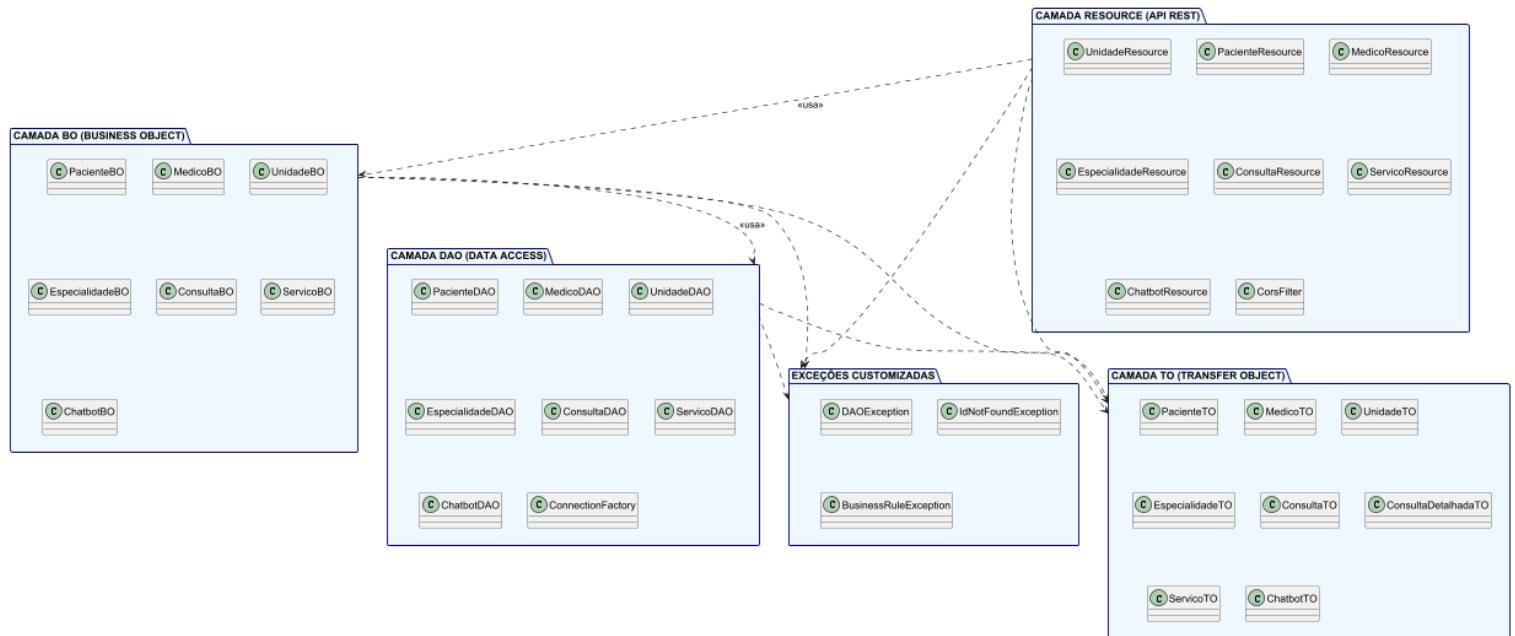
## **3. Modelo de Entidade-Relacionamento (MER)**

### **Tabelas do Banco de Dados Oracle**



## 4. Diagrama de Classes

### Arquitetura em Camadas



## Descrição das Classes Principais

### Camada Resource

- Responsabilidade:** Exportar endpoints REST, validar requisições, tratar exceções HTTP
- Anotações:** @Path, @GET, @POST, @PUT, @DELETE, @Produces, @Consumes
- Retorno:** Response com status HTTP adequado

### Camada BO (Business Object)

- Responsabilidade:** Implementar regras de negócio, validações, cálculos
- Exemplos de Regras:**
  - Validação de CPF único (PacienteBO)
  - Validação de idade mínima (PacienteBO)
  - Antecedência mínima para consultas (ConsultaBO)
  - Geração de protocolo automático (ConsultaBO)

### Camada DAO (Data Access Object)

- Responsabilidade:** Acesso direto ao banco de dados, executar SQL
- Métodos:** findAll(), findById(), save(), update(), delete()
- Padrão:** Uso de PreparedStatement para prevenir SQL Injection

### Camada TO (Transfer Object)

- Responsabilidade:** Representar dados transferidos entre camadas
- Validações:** Jakarta Bean Validation (@NotBlank, @NotNull, @Size, @Email, etc.)

## 5. Imagens do sistema



**Tela inicial:** Um menu inicial simples, com 3 opções em foco no centro da tela, Especialidades, Marcar Consulta e Unidades, note que a mensagem de “Bem-Vindo” informa um nome! É o nome do paciente cadastrado, que o sistema usa ao criar uma conta, o exato mesmo no Back-End! Caso contrário a mensagem é apenas “Seja Bem-Vindo ao Cuida Fácil como podemos te ajudar hoje?”.



[Voltar](#)

The image displays a grid of six cards, each representing a different medical unit or hospital. Each card includes a small thumbnail image of the facility's exterior, its name, address, phone number, operating hours, and a list of services offered.

- HC-PINHEIROS**  
Endereço: Rua Dr. Ovídio Pires de Campos, 225 - Cerqueira César, São Paulo  
Telefone: (11)2661-0000  
Horário: Seg-Sex: 7h-19h  
Serviços:
  - Consultas Eletivas
  - Consultas
  - Exames
  - Day Clinic
- ICH-CENTRAL**  
Endereço: Av. Dr. Enéas Carvalho de Aguiar, 255 - Cerqueira César, São Paulo  
Telefone: (11)2661-5000  
Horário: 24h  
Serviços:
  - Emergência 24h
  - Exames
- IMREA-VILA-MARIANA**  
Endereço: Rua Domingo de Soto, 100 - Jardim Vila Mariana, São Paulo  
Telefone: (11)5180-7800  
Horário: Seg-Sex: 6h30-20h  
Serviços:
  - Consultas Eletivas
  - Consultas
  - Exames
- INCOR**  
Endereço: Av. Dr. Enéas Carvalho Aguiar, 647 - Cerqueira César, São Paulo  
Telefone: (11)2661-5000  
Horário: 24h  
Serviços:
  - Emergência 24h
  - Exames
- ICR-HC**  
Endereço: Av. Dr. Enéas Carvalho de Aguiar, 647 - Cerqueira César, São Paulo  
Telefone: (11)2661-8500  
Horário: Atendimento 24h (Pronto-Socorro)  
Serviços:
  - Emergência 24h
  - Exames
- IOT-HC**  
Endereço: Rua Dr. Ovídio Pires de Campos, 333 - Cerqueira César, São Paulo  
Telefone: (11)2661-0000  
Horário: Seg-Sex: 7h-17h (Ambulatório)  
Serviços:
  - Consultas Eletivas
  - Consultas
  - Exames

**Tela especialidades e Unidades:** Ambos possuem relação direta com o Back-End/Banco de Dados! Criando cards dinâmicamente e atualizando a página baseado no que foi inserido, em Unidades, podendo se ver o nome da unidade, o seu endereço, horários de funcionamento e seus serviços! Em Especialidades, se define o nome da especialidade e sua função com uma descrição

The image shows a login form window titled "Login:". It contains fields for "CPF ou E-Mail" and "Senha", along with links for "Esqueci a minha senha" and "Não possui uma conta?". A "Confirmar" button is at the bottom. The background of the entire page has a light blue header bar with the word "Voltar" at the top right.

© 2025 CuidaFácil | Feito com empatia para você.

**Os sistemas de cadastro e login:** Essas duas funcionalidades operam baseado na interação entre a interface do Front-End e do Back-End, onde ocorre uma verificação de se o paciente existe mesmo no Banco de Dados, e caso contrário, no login se registra um novo paciente na tabela. Senha é o CEP do paciente + o dia de sua data de nascimento.

Voltar

### Agendar Consulta

Data da Consulta \*

Horário (07:00 - 18:00) \*

Especialidade \*

Selecione uma especialidade

Unidade de Atendimento \*

Selecione uma unidade

Tipo de Atendimento \*

Presencial  Teleconsulta

[Cancelar](#) [Confirmar Agendamento](#)

Ajuda FAQ Integrantes Contato

Voltar

### Agendar Consulta

Data da Consulta \*

Horário (07:00 - 18:00) \*

Especialidade \*

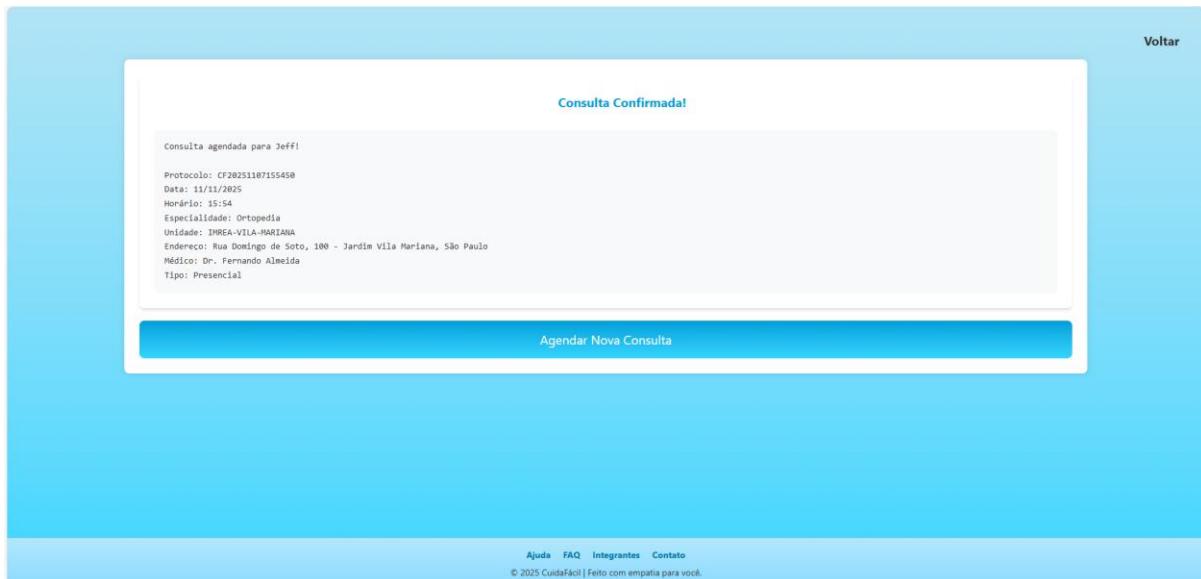
Selecione uma especialidade

Selecione uma especialidade

- [Cardiologia](#)
- [Ortopedia](#)
- [Pediatría](#)
- [Dermatología](#)
- [Neurología](#)
- [Ginecología](#)
- [Psiquiatría](#)
- [Oftalmología](#)
- [Urología](#)
- [Endocrinología](#)

[Ajuda](#) [FAQ](#) [Integrantes](#) [Contato](#)  
© 2025 CuidaFácil | Feito com empatia para você.

**Agendamento de Consultas:** As opções de especialidades, unidades e médicos todos são em tempo real baseados nos respectivos registrados dentro das suas tabelas no Banco de Dados que se comunica com o Back-End e Front-End!



```
{
  "idConsulta": 11,
  "protocolo": "CF20251107155450",
  "dataConsulta": "2025-11-12T15:54:00",
  "status": "AGENDADA",
  "tipoAtendimento": "PRESENCIAL",
  "idPaciente": 11,
  "idMedico": 107,
  "idUnidade": 3,
  "idEspecialidade": 2,
  "nomePaciente": "Jeff",
  "nomeMedico": "Dr. Fernando Almeida",
  "nomeUnidade": "IMREA-VILA-MARIANA",
  "nomeEspecialidade": "Ortopedia",
  "crmMedico": "CRM-RJ-78901",
  "enderecoUnidade": "Rua Domingo de Soto, 100 - Jardim Vila Mariana, São Paulo"
}.
```

**Tela especialidades e Unidades:** Ao marcar uma consulta ela é salva dentro da tabela, como visto nessa imagem do JSON visto em [cuida-facil-sprint-4-java.onrender.com/consultas](https://cuida-facil-sprint-4-java.onrender.com/consultas) e na tela de confirmação de consulta.

## 6. Tecnologias usadas

- Linguagem:** Java 17+
- Framework:** Jakarta EE (JAX-RS)
- Banco de Dados:** Oracle Database
- Validação:** Jakarta Bean Validation
- Servidor:** render.com (se for sem aplicação individual), caso contrário Apache Tomcat / Glassfish
- Build:** Maven
- Versionamento:** Git/GitHub

## **7. Procedimentos para Rodar a Aplicação**

### **7.1. Pré-requisitos**

- JDK 17 ou superior
- Maven 3.8+
- Oracle Database 19c ou superior

### **7.2. Configuração do Banco de Dados**

1. Criar as tabelas conforme scripts SQL fornecidos
2. Configurar variáveis de ambiente:

```
export DB_URL=" jdbc:oracle:thin:@oracle.fiap.com.br:1521:ORCL"
export DB_USER="seu_usuario"
export DB_PASSWORD="sua_senha"
```

### **7.3. Compilação para Deploy**

Utilizando o comando ‘mvn clean package’ Você pode obter um deploy do JAR limpo, gerado no servidor de aplicação

## **8. Testando a API**

Exemplo com curl: curl -X GET <http://localhost:8080/cuida-facil-sprint4>

Exemplo com render.com no Insomnia: <https://cuida-facil-sprint-4-java.onrender.com>

## **9. Considerações finais**

O sistema **Cuida Fácil** foi desenvolvido seguindo as melhores práticas de desenvolvimento Java, com arquitetura em camadas bem definida, tratamento robusto de exceções, validações em múltiplas camadas, API RESTful completa e funcional, código documentado e organizado, pronto para integração com front-end.

O projeto está 100% funcional e pronto para uso em produção após ajustes de segurança e performance.

Link do repositório Github: [Gdev3356/Cuida-Facil-Sprint-4-Java](https://github.com/Gdev3356/Cuida-Facil-Sprint-4-Java)