RELATÓRIO TÉCNICO — Sistema de Gestão de Clínica (Frontend + Backend)

1. Visão Geral
O sistema de gestão de clínica é composto por um frontend desenvolvido em React e um backend em .NET 9, integrados via API RESTful. O objetivo é gerenciar pacientes, consultas, profissionais, serviços e relatórios de forma centralizada, com autenticação JWT, interface responsiva e feedback visual ao usuário.

2. Tecnologias Utilizadas
- Frontend: React 18, styled-components, react-router-dom, axios, react-icons.
- Backend: .NET 9, Entity Framework Core, SQL Server, JWT Authentication.
- Banco de Dados: SQL Server (local ou via Docker).
- Controle de versão: Git/GitHub.

3. Configuração do Ambiente
Frontend:
- Requisitos: Node.js v16+, npm ou yarn.
- Passos:
git clone https://github.com/CaioVAzeredo/front-end-gestao-clinica.git
cd front-end-gestao-clinica
npm install
Criar arquivo .env com REACT_APP_PORT apontando para a porta do backend.
npm start

Backend:
- Requisitos: .NET 9 SDK, SQL Server.
- Passos:
git clone https://github.com/cesaraugusto0/GestaoClinica.git
cd GestaoClinica
Configurar SQL Server local ou via Docker.
Executar scripts SQL presentes na pasta scripts.
Configurar appsettings.json com dados do banco e chave JWT.

4. Arquitetura
- Frontend consome API REST do backend para operações CRUD.
- Backend segue arquitetura em n camadas, separando responsabilidades em Controllers, Services, Repositories e Models.
- Banco de dados relacional no SQL Server, com entidades Cliente, Funcionário, Serviço, Agendamento e Endereço.

5. Funcionalidades Implementadas
Frontend:
- Login com autenticação JWT.
- Dashboard com indicadores.
- CRUD de clientes, funcionários, serviços e agendamentos.
- Agenda com validação de conflitos.
- Relatórios com filtros por período.
- Interface responsiva e estilizada com styled-components.

Backend:
- API RESTful com autenticação JWT.
- CRUD completo das entidades.
- Validações no lado do servidor.
- Integração com SQL Server.
- Geração de relatórios via consultas agregadas.

6. Estilização e Responsividade
- Uso de styled-components para CSS-in-JS.
- Layout adaptável a diferentes resoluções.
- Menu lateral recolhível no desktop e drawer no mobile.

7. Integração
- Comunicação via Axios no frontend, consumindo endpoints HTTPS do backend.
- Porta da API configurável via variável de ambiente no frontend.

8. Testes e Ajustes
- Testes manuais realizados para todas as funcionalidades principais.
- Ajustes visuais e de usabilidade implementados com base nos testes.

9. Documentação
- README.md atualizado com instruções de instalação e uso.
- Relatório técnico detalhando arquitetura, tecnologias e fluxo de integração.
- Apresentação final com prints das principais telas e funcionalidades.

10. Execução em Produção
Frontend:
npm run build
Deploy em servidor web ou serviço de hospedagem estática.

Backend:
Publicação com dotnet publish e hospedagem em servidor IIS, contêiner ou serviço em nuvem.

Desenvolvido por: Caio, Cesar, Matheus, Marçal e Marlon
Data: 09/08/2025