

UNIVERSIDADE FEDERAL DE SANTA CATARINA

Prontuário Familiar no e-SUS APS: Implementação de uma ferramenta para
qualificação do cuidado em saúde na Atenção Primária do SUS

Davi Felipe Becker Morfim

Florianópolis

2025

UNIVERSIDADE FEDERAL DE SANTA CATARINA

DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Prontuário Familiar no e-SUS APS: Implementação de uma ferramenta para
qualificação do cuidado em saúde na Atenção Primária do SUS

Trabalho de Conclusão do Curso de
Graduação em Sistemas de Informação do
Centro Tecnológico, da Universidade
Federal de Santa Catarina, como requisito
para à obtenção do título de Bacharel em
Sistemas de Informação, no segundo
semestre de 2025.

Orientador: Dr. Prof. Raul Sidnei Wazlawick

Comissão Examinadora

Prof. Dr. Raul Sidnei Wazlawick
Orientador

Me. Jades Fernando Hammes

Bel. Nicolas Elias Sant Ana

Agradecimentos

É com imensa satisfação que concluo este trabalho, e gostaria de expressar minha profunda gratidão a todos que tornaram sua execução possível.

Agradeço ao **Laboratório Bridge** por ter me concedido a honra e a oportunidade de desenvolver este Trabalho de Conclusão de Curso sobre um dos maiores e mais relevantes sistemas de saúde do Brasil. É sempre um privilégio poder contribuir diretamente para o aprimoramento da saúde pública nacional.

Primeiramente, minha sincera gratidão ao **Ricardo Luiz Camargo Prado**. Sua proatividade e visão foram cruciais; sem sua liderança na frente com o cliente e sua confiança em minha capacidade de desenvolver o **Prontuário Familiar**, este que é um dos módulos mais esperados pelo e-SUS APS, o projeto não teria sido concretizado.

Estendo meus agradecimentos à **Aline Effting Guedes**, expressando minha sincera gratidão por sua valiosa colaboração na concepção visual do módulo, garantindo que o projeto seguisse os melhores padrões de experiência do usuário e estivesse alinhado aos conceitos estabelecidos do e-SUS APS.

Agradeço também à **Equipe NP-Completo**, do Laboratório Bridge, por atuarem como revisores e futuros guardiões deste projeto, assegurando que o módulo fosse desenvolvido rigorosamente dentro dos padrões técnicos do e-SUS APS.

Da mesma forma, sou grato à **Equipe Royal Flush**, também do Laboratório Bridge, pela paciência, empatia e compreensão demonstradas, cujo apoio constante foi essencial para lidar com a intensidade do desenvolvimento deste trabalho.

A todos que, de alguma forma direta ou indireta, contribuíram para a realização e o sucesso deste projeto, o meu mais sincero e profundo obrigado.

Resumo

A Atenção Primária à Saúde (APS), sustentada pela Estratégia de Saúde da Família (ESF), exige uma visão holística e centrada na unidade familiar, um conceito que o Prontuário Eletrônico do Cidadão (PEC) do sistema federal e-SUS APS não abordava integralmente. Essa lacuna, que limitava a efetividade do cuidado longitudinal, representou o desafio central deste Trabalho de Conclusão de Curso. O objetivo principal foi implementar o módulo do Prontuário Familiar no e-SUS APS, fornecendo aos profissionais de saúde uma ferramenta para visualizar, de forma integrada, o contexto social e de saúde de toda a família.

Adotando metodologias de desenvolvimento ágil, como o Scrum e b_Thinking, e uma arquitetura moderna, a solução foi desenvolvida utilizando Java 17 com Spring Boot no *back-end* e React/TypeScript no *front-end*. A arquitetura empregou o GraphQL como *gateway* de dados, essencial para otimizar o consumo de informações e garantir a baixa latência nas consultas.

O Prontuário Familiar desenvolvido permite a visualização imediata da lista de todos os familiares, dos dados cadastrais do imóvel e do histórico de visitas domiciliares, cumprindo todos os Requisitos Funcionais e os rigorosos Requisitos Não Funcionais de desempenho e segurança. Como resultado, este trabalho entrega uma solução funcional, tecnologicamente robusta e alinhada às melhores práticas de software e políticas de saúde, com potencial para a modernização da gestão da informação e a qualificação do cuidado na APS brasileira.

Palavras-chave: Prontuário Familiar. e-SUS APS. Saúde Digital. Atenção Primária à Saúde. Estratégia de Saúde da Família. Desenvolvimento Ágil. Prontuário Eletrônico do Cidadão.

Abstract

Primary Health Care (PHC), supported by the Family Health Strategy (FHS), requires a holistic and family-centered approach, a concept that the Citizen's Electronic Health Record (PEC) of the federal e-SUS APS system did not fully address. This gap, which limited the effectiveness of longitudinal care, represented the central challenge of this Final Project. The main objective was to implement the Family Health Record module in e-SUS APS, providing healthcare professionals with a tool to visualize, in an integrated way, the social and health context of the entire family.

Adopting agile development methodologies, such as Scrum and b_Thinking, and a modern architecture, the solution was developed using Java 17 with Spring Boot on the back-end and React/TypeScript on the front-end. The architecture employed GraphQL as a data gateway, essential for optimizing information consumption and ensuring low latency in queries.

The developed Family Health Record allows for the immediate visualization of the list of all family members, property registration data, and the history of home visits, fulfilling all Functional Requirements and the rigorous Non-Functional Requirements of performance and security. As a result, this work delivers a functional, technologically robust solution aligned with best software practices and health policies, with the potential to modernize information management and improve the quality of care in Brazilian PHC.

Keywords: Family Health Record. e-SUS APS. Digital Health. Primary Health Care. Family Health Strategy. Agile Development. Citizen's Electronic Health Record.

LISTA DE FIGURAS

Figura 1 – Aba do Prontuário Familiar no cabeçalho do módulo do Cidadão.....	69
Figura 2 – Componente da composição familiar - todos os dados preenchidos.....	70
Figura 3 – Componente da composição familiar - dados mínimos.....	71
Figura 4 – Acompanhamento de vulnerabilidade.....	72
Figura 5 – Tab switcher.....	73
Figura 6 – Cabeçalho do imóvel - todos os dados preenchidos.....	74
Figura 7 – Cabeçalho do imóvel - dados mínimos.....	74
Figura 8 – Condições de moradia do imóvel - todos os dados preenchidos.....	75
Figura 9 – Condições de moradia - dados mínimos.....	76
Figura 10 – Histórico de visitas realizadas.....	77
Figura 11 – Histórico de visitas domiciliares - sem visitas.....	78
Figura 12 – Prontuário Familiar - cidadão não possui uma família.....	79
Figura 13 – Prontuário Familiar - cidadão mudou de território.....	79
Figura 14 – Prontuário Familiar completo com seção “Dados do imóvel”.....	80
Figura 15 – Prontuário Familiar completo com seção “Visitas realizadas”	81

SUMÁRIO

Resumo.....	4
1 INTRODUÇÃO.....	9
1.1 Objetivo Geral.....	10
1.2 Objetivos Específicos.....	10
1.3 Delimitações.....	11
1.4 Estrutura do Trabalho.....	11
2 FUNDAMENTAÇÃO TEÓRICA.....	13
2.1 O Sistema Único de Saúde (SUS).....	13
2.2 Atenção Primária à Saúde e a Estratégia e-SUS APS.....	13
2.3 O Prontuário Eletrônico e-SUS APS.....	15
2.3 A Estratégia de Saúde da Família.....	16
2.3 Prontuário familiar.....	17
2.4 A Triagem para o Risco de Insegurança Alimentar.....	18
2.5 Arquiteturas e Tecnologias do e-SUS APS.....	19
2.5.1 Interfaces de Programação de Aplicações (APIs).....	19
2.5.2 Padrões de Projeto de Software Relevantes.....	21
2.5.3 Java e Spring Boot (Back-end).....	21
2.5.4 Banco de Dados - Postgres e Oracle.....	22
2.5.5 TypeScript, React e Design Systems (Front-end).....	24
2.6 Metodologias de Desenvolvimento de Software.....	25
2.6.1 Desenvolvimento Ágil: Manifesto e Princípios.....	25
2.6.2 Scrum - Um framework ágil.....	26
2.6.3 Engenharia de Requisitos Ágil: O Foco nas Histórias de Usuário.....	28
2.6.4 O Processo de Descoberta de Produto utilizando o b_Thinking.....	28
3 TRABALHOS RELACIONADOS.....	30
4 METODOLOGIA E PROJETO DO PRONTUÁRIO FAMILIAR.....	31
4.1 Metodologia de Desenvolvimento.....	31
4.2 Levantamento e Análise de Requisitos.....	32
4.2.1 Épico do Prontuário Familiar.....	33
4.2.2 Requisitos Funcionais (RF).....	33
4.2.3 Requisitos Não Funcionais (RNF).....	35
4.3 Arquitetura do Sistema.....	37
4.3.1 Arquitetura do Back-end.....	37
4.3.2 Arquitetura do Front-end.....	38
4.3.3 Banco de Dados.....	39
5 IMPLEMENTAÇÃO DO PRONTUÁRIO FAMILIAR.....	41
5.1 Ambiente de Desenvolvimento e Ferramentas Utilizadas.....	41
5.2 Implementação do Back-end.....	42
5.2.1 Schemas (Camada de API).....	43
5.2.2 Query resolvers e Field Resolvers (Camada de API).....	44
5.2.2 Data Loaders (Camada de Acesso a Dados).....	47

5.2.3 Services (Camada de Negócio).....	48
5.2.3 Repositórios (Camada de Acesso a Dados).....	50
5.2.4 Implementação da Segurança.....	52
5.2.4.1 Herança e Padrões de Segurança.....	52
5.2.4.2 Controle de Acesso por Recurso.....	53
5.2.5 Considerações.....	53
5.2.6 Testes Automatizados.....	53
5.2.6.1. Testes Automatizados no Back-end.....	54
5.2.6.1.1. Testes de Unidade.....	54
5.2.6.1.2. Testes de integração.....	55
5.3 Implementação do Front-end.....	55
5.3.1 Estrutura de Roteamento.....	55
5.3.2 Tela Principal: ProntuarioFamiliarView.....	56
5.3.3 Componentes filhos.....	57
5.3.4 Outros componentes.....	62
5.3.5 Utilitários.....	66
5.3.6 Testes Automatizados no Front-end.....	66
6 APRESENTAÇÃO E ANÁLISE DO PRONTUÁRIO FAMILIAR DESENVOLVIDO.....	68
6.1 Apresentação das Funcionalidades Implementadas.....	68
6.1.1 Acesso ao Prontuário Familiar.....	69
6.1.2 Composição da família.....	69
6.1.3 Acompanhamento de vulnerabilidade.....	71
6.1.4 Botões Dados do imóvel e Visitas realizadas.....	73
6.1.5 Dados do imóvel - Informações de localização e contato.....	73
6.1.6 Dados do imóvel - Condições de moradia.....	74
6.1.7 Visitas realizadas.....	76
6.1.8 Estados de exceção.....	78
6.1.9 Visualização completa do Prontuário Familiar.....	79
6.2 Verificação do Atendimento aos Requisitos.....	81
6.2.1 Requisitos Funcionais.....	82
6.2.2 Requisitos Não Funcionais.....	84
6.3 Discussão das Escolhas de Projeto e Implementação.....	85
7 CONCLUSÃO.....	87
7.1 Síntese do Projeto e Alcance dos Objetivos.....	87
7.2 Principais Contribuições do Trabalho.....	88
7.3 Reconhecimento das Limitações.....	89
7.4 Trabalhos Futuros e Continuidade do Projeto.....	89
8 CONSIDERAÇÕES FINAIS.....	91
REFERÊNCIAS.....	92

1 INTRODUÇÃO

O trabalho está inserido no contexto da Saúde Digital e da Atenção Primária à Saúde (APS) no Brasil, especificamente no âmbito do Sistema Único de Saúde (SUS). O SUS, estabelecido em 1988, define a saúde como um direito de todos os cidadãos, sendo a APS o principal nível de atenção e promoção da saúde no país.

Para qualificar a gestão da informação na APS, o Ministério da Saúde lançou, em 2013, a Estratégia e-SUS APS, culminando na criação do Prontuário Eletrônico do Cidadão (PEC). O PEC é uma ferramenta de *software* essencial para a informatização das Unidades Básicas de Saúde (UBS), projetada para garantir o registro individualizado e a integração dos dados de saúde.

Paralelamente, a Estratégia de Saúde da Família (ESF) estabeleceu a família como o objeto central da atenção, enfatizando o cuidado integral. A incorporação do contexto familiar na tomada de decisão requer instrumentos adequados, sendo o Prontuário Familiar um instrumento que garante o registro e o acesso ágil às informações e ações de saúde de todos os membros de uma família.

O objetivo deste trabalho é, portanto, suprir uma lacuna no sistema e-SUS APS: a ausência de um prontuário eletrônico dedicado à unidade familiar. Não foram encontrados trabalhos correlatos que apresentassem a mesma proposta de desenvolvimento de um prontuário familiar eletrônico associado à estratégia de saúde da família do governo federal, o que demonstra a adesão da proposta ao quesito inovação.

Este documento detalha a fundamentação teórica, a metodologia de desenvolvimento, a arquitetura da solução, os resultados alcançados e as conclusões deste projeto.

1.1 Objetivo Geral

Implementar o módulo do Prontuário Familiar dentro do sistema e-SUS APS (Prontuário Eletrônico do Cidadão - PEC), fornecendo aos profissionais de saúde da Estratégia de Saúde da Família uma ferramenta para visualizar, de forma integrada e no contexto do atendimento, as informações sociais, clínicas e de acompanhamento da família do cidadão, visando qualificar o cuidado em saúde no âmbito da Atenção Longitudinal.

1.2 Objetivos Específicos

Para alcançar o objetivo geral, este trabalho se propõe a:

1. Desenvolver a funcionalidade de carregamento e exibição do Prontuário Familiar no contexto do cidadão atualmente em atendimento ou acompanhamento, permitindo a visualização da lista de todos os familiares cadastrados, condições de moradia e dados de apoio.
2. Apresentar dados essenciais dos familiares (como nome completo, idade, data de nascimento, CPF ou CNS, e grau de parentesco) e informações detalhadas sobre o imóvel de residência da família, incluindo endereço, dados cadastrais e condições de moradia (ex: situação, localização, destino do lixo, escoamento do banheiro).
3. Implementar a exibição do histórico de visitas domiciliares realizadas para a família e seus membros, incluindo data da visita, profissional que realizou e sua equipe, além dos motivos que levaram à visita e anotações realizadas.
4. Garantir a navegação direta e segura para o prontuário individual de qualquer familiar a partir da listagem, assegurando que o acesso seja validado pelo sistema de controle de acesso por recurso e conforme a Lei Geral de Proteção de Dados (LGPD).

5. Assegurar que o módulo atenda a requisitos de desempenho, carregando as informações cruciais (familiares, imóvel e histórico de visitas) com baixa latência perceptível (em menos de 0,5 segundos).

1.3 Delimitações

As delimitações estabelecidas para este projeto são:

1. Escopo Funcional (Módulo): O foco da implementação é a visualização das informações do Prontuário Familiar no e-SUS APS. Não está no escopo a criação de novas funcionalidades para edição, persistência de dados cadastrais de cidadãos ou famílias, que continuarão a ser gerenciadas pelos módulos já existentes.
2. Tecnologia: A implementação é restrita à arquitetura de *software* existente do e-SUS APS, utilizando Java 17 com o *framework* Spring Boot e GraphQL para o *back-end*, e React com TypeScript para o *front-end*, seguindo o padrão de arquitetura de blocos independentes e distinção inequívoca de responsabilidades.
3. Ambiente e Compatibilidade: O desenvolvimento e os testes garantem a compatibilidade com a base de dados (Postgres e Oracle) e com as duas versões mais recentes dos navegadores Google Chrome, Mozilla Firefox e Microsoft Edge, além de ser responsivo para telas com resolução mínima de 1024x768 píxeis.

1.4 Estrutura do Trabalho

O trabalho está estruturado em sete capítulos, conforme detalhado a seguir:

- **Capítulo 1 – Introdução:** Apresenta o contexto, a justificativa, o problema, o objetivo geral, os objetivos específicos e as delimitações do trabalho.

- **Capítulo 2 – Fundamentação Teórica:** Apresenta o contexto do SUS e da APS, detalha a Estratégia e-SUS APS e o Prontuário Eletrônico do Cidadão , além de abordar a Estratégia de Saúde da Família , o conceito de Prontuário Familiar e a Triagem para o Risco de Insegurança Alimentar. Aborda aspectos de acessibilidade e LGPD, a base tecnológica e as metodologias de desenvolvimento.
- **Capítulo 3 – Trabalhos Relacionados:** Descreve a pesquisa bibliográfica realizada, destacando a relevância e a inovação da proposta.
- **Capítulo 4 – Metodologia e Projeto:** Detalha a metodologia de desenvolvimento (Scrum e b_Thinking), o processo de Levantamento e Análise de Requisitos (Épico, Requisitos Funcionais e Não Funcionais), e o desenho da Arquitetura do Sistema.
- **Capítulo 5 – Implementação do Prontuário Familiar:** Descreve o ambiente de desenvolvimento, as ferramentas utilizadas e a implementação detalhada do *back-end* e do *front-end*.
- **Capítulo 6 – Apresentação e Análise do Prontuário Familiar Desenvolvido:** Expõe a solução final, verificando o atendimento aos requisitos funcionais e não funcionais, e discute as escolhas de projeto, contribuições e limitações.
- **Capítulo 7 - Conclusão:** Apresenta a síntese do trabalho, um resumo das principais contribuições, delimitações e trabalhos futuros.
- **Capítulo 8 – Considerações Finais:** Um breve texto sobre os aprendizados e considerações levantadas com a conclusão do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 O Sistema Único de Saúde (SUS)

Em 1988 foi instituído, pela Constituição Federal, o Sistema Único de Saúde (SUS), sendo regulamentado pelas Leis nº 8.080/1990 (BRASIL, 1990a) e nº 8.142/1990 (BRASIL, 1990b). Considerado por muitos como um marco na política social brasileira, o SUS estabelece a saúde como uma política de estado e um direito de todos os cidadãos. Seus princípios (Universalidade, Equidade e Integralidade, Regionalização, Hierarquização, Descentralização, Participação Social e Resolutividade) orientam a organização da rede de serviços e norteiam as ações de saúde em todo Brasil (MINISTÉRIO DA SAÚDE, [s.d.]c).

O SUS, para que possa oferecer serviços gratuitos para o cidadão Brasileiro, movimenta bilhões de reais anualmente. Segundo o Instituto de Estudos para Políticas de Saúde (IPES), “o orçamento federal da Saúde cresceu 30% entre 2022 e 2025, passando de R\$ 179,1 para R\$233,3 bilhões”, sendo que essa é somente a esfera federal. O custeio integral das ações é compartilhado com Unidades Federativas e Municípios e, em 2024, a União contribuiu com aproximadamente 42% do gasto público total, os municípios com cerca de 31,5% e as unidades federativas com cerca de 26,4%.

2.2 Atenção Primária à Saúde e a Estratégia e-SUS APS

A organização do SUS é formada por esferas e uma delas é a Atenção Primária à Saúde (APS), anteriormente conhecida como Atenção Básica. A APS é pautada como o primeiro ponto de contato dos cidadãos com o SUS, sendo o principal nível de atenção e promoção da saúde em território nacional.

Segundo o relatório Cobertura da Atenção Primária (PNS 2020-2023), o total de cidadãos brasileiros diretamente atendidos pela APS, em dezembro de 2023, foi de 162.269.112, o que correspondia a 79% da população do país à época, estimada pelo IBGE.

Sua definição oficial foi firmada pela Política Nacional de Atenção Básica, e consta como a seguinte (BRASIL, 2011):

“Caracteriza-se por um conjunto de ações de saúde, no âmbito individual e coletivo, que abrangem a promoção e a proteção da saúde, a prevenção de agravos, o diagnóstico, o tratamento, a reabilitação, a redução de danos e a manutenção da saúde, com o objetivo de desenvolver uma atenção integral que impacte na situação de saúde e autonomia das pessoas e nos determinantes e condicionantes de saúde das coletividades. A eficiência e a capacidade de resolução da APS são, portanto, fundamentais para o desempenho de todo o sistema”.

Em meados de 2013 o Ministério da Saúde, por meio da Secretaria de Atenção Primária à Saúde (SAPS), lançou a Estratégia e-SUS Atenção Primária (e-SUS APS), visando reestruturar e qualificar a gestão da informação na Atenção Primária. Esta iniciativa visa modernizar os processos de trabalho das equipes de APS, informatizando as Unidades Básicas de Saúde (UBS) e promovendo a coleta de dados individualizados e qualificados, essenciais para o planejamento, monitoramento e avaliação das ações em saúde (MINISTÉRIO DA SAÚDE, [s.d.]a).

A Estratégia e-SUS APS foi formalizada, em suas diretrizes iniciais, por meio de normativas como a Portaria GM/MS Nº 1.412, de 10 de julho de 2013 (BRASIL, 2013), estabeleceu alguns requisitos para os sistemas de informação da APS:

- Individualização do registro: garantir o registro das informações de saúde de forma individualizada, permitindo o acompanhamento longitudinal do cidadão.
- Integração da informação: promover a interoperabilidade e a integração dos dados entre os diversos sistemas de informação da APS.
- Redução do retrabalho na coleta de dados: otimizar o processo de coleta, evitando a duplicação de registros em diferentes instrumentos ou sistemas.
- Informatização das unidades: estimular a adoção de tecnologias da informação e comunicação (TIC) nas UBS, com sistemas que contemplem os processos de trabalho e as boas práticas em saúde.
- Gestão do cuidado: oferecer ferramentas que apoiem os profissionais na gestão do cuidado, na tomada de decisão clínica e na coordenação das ações de saúde.
- Coordenação do cuidado: qualificar o uso da informação para aprimorar a coordenação do cuidado na Rede de Atenção à Saúde (RAS).

Tem-se como objetivo central da estratégia, portanto, fomentar um modelo de gestão da informação que apoie os municípios e os serviços de saúde na gestão efetiva da APS, e ofereça mecanismos de qualificação do cuidado ofertado aos usuários do SUS, tornando o SUS cada vez mais eletrônico e eficiente (MINISTÉRIO DA SAÚDE, [s.d.]a).

2.3 O Prontuário Eletrônico e-SUS APS

Como fruto da Estratégia e-SUS APS surge o Prontuário Eletrônico do Cidadão (PEC). Tal prontuário configura-se como uma das principais ferramentas de software disponibilizadas gratuitamente pelo MS (MINISTÉRIO DA SAÚDE, [s.d.]b) para a informatização das Unidades Básicas de Saúde (UBS), popularmente conhecidos como Postos de Saúde. O PEC foi criado como um sistema de informação que visa atender todos os requisitos supracitados para sistemas de informação da APS.

Para tanto, apresenta funcionalidades que incluem o registro de atendimentos clínicos, odontológicos, de procedimentos, atividades coletivas, dispensação de medicamentos, atualizações cadastrais, relatórios, suporte à decisão clínica, vacinação, acompanhamento de condicionalidades de programas sociais, agendamento, entre outras. O sistema busca organizar o fluxo de atendimento, qualificar os dados produzidos, otimizar o trabalho dos profissionais e fornecer informações para a gestão do cuidado e da unidade de saúde (CIELO; GOMES; JÚNIOR, 2021), possuindo uma forte interoperabilidade entre suas aplicações, instalações descentralizadas e o Centralizador Nacional de informações, que por sua vez, consolida dados brutos dos serviços prestados, para fomentar inteligência na elaboração de políticas públicas de saúde.

Não obstante, o PEC continua sendo plena e constantemente evoluído desde 2013 e já é utilizado por 87% das UBS no Brasil. Novas funcionalidades são constantemente mapeadas e desenvolvidas pela parceria SAPS/UFSC, através de concomitantes Termos de Execução Descentralizada que fomentam o trabalho

através do Laboratório Bridge e da Fundação Stemmer para Pesquisa, Desenvolvimento e Inovação (FEESC).

2.3 A Estratégia de Saúde da Família

Criada em 1998, 10 anos após a criação do SUS, a Estratégia de Saúde da Família torna-se a estratégia prioritária para expansão e consolidação da APS. (PAIM, 2001). Segundo Ramos (2008), os processos de assistência alinhados ao modelo de Saúde da Família devem ser estruturados por um conjunto de ações e atividades, no contexto da APS, e que, enquanto integradas e articuladas, promovem uma atenção integral à saúde, de tal forma que estruturam o trabalho das equipes.

Tais processos pautam-se em oferecer desde consultas clínicas ou de procedimentos, passando pela gestão do processo clínico longitudinal do cidadão, do processo familiar do mesmo (tendo a família como um sistema complexo, dinâmico e aberto), otimizando e qualificando a gestão prática clínica, provendo orientações à comunidade para proteção e promoção da saúde, além da integração entre todos os agentes da APS e outros. (GAETE, 2020)

Dentre os diversos objetivos e diretrizes específicos da ESF, destacam-se alguns mais relevantes para o desenvolvimento deste presente trabalho:

- Promover o vínculo e o acompanhamento da população, incentivando o cadastro qualificado, a realização de visitas domiciliares e o acesso equitativo aos serviços de saúde, e;
- Garantir a atenção integral à saúde da população adscrita, atendendo adequadamente a população vinculada, promovendo maior proximidade e resolutividade nos cuidados, ações de promoção, proteção, recuperação da saúde e prevenção de doenças, que assegurem a qualidade e humanização do cuidado.

Na Estratégia da Saúde da Família (ESF), a família é entendida como objeto da atenção e deve ser tratada “a partir do território onde vive, uma vez que é nele que ela constrói suas relações sociais, intra e extra familiares e também desenvolve sua luta cotidiana em busca da melhoria de suas condições de vida” (ATHAYDE; GIL, 2005, p. 14). Esse modo de atuar tende a fomentar a identificação de fatores de maior vulnerabilidade na família, permitindo a priorização de ações de cuidado a saúde que possam diminuir os riscos de agravos e garantir maior efetividade das ações clínicas e de assistência.

2.3 Prontuário familiar

A abrangência da Estratégia de Saúde da Família, segundo (SILVA *et al.*, 2013), tornou necessária a existência de instrumentos que fomentem os princípios que regem essa estratégia, centrados na incorporação do contexto familiar no processo de tomada de decisão.

Um desses instrumentos de trabalho é o Prontuário familiar, que garante o registro e acesso das informações e permite, de forma ágil, o acesso às ações já realizadas pela equipe de saúde da família, garantindo a continuidade do cuidado à saúde de seus membros. Ainda conforme (SILVA *et al.*, 2013):

“A medicina familiar tem como princípio centrar-se na atenção da pessoa como um todo. Para tanto, é necessário contar com equipes de saúde que vejam a doença no contexto da família, e que conheçam as premissas da orientação familiar na atenção primária (sic), com o propósito de realizar intervenções neste âmbito de atuação.”

A Secretaria de Estado da Saúde do Ceará (2005) define o Prontuário Familiar como sendo instrumento de integração das informações de saúde dos indivíduos, famílias e comunidades das áreas de atuação das equipes de saúde da família. Sua utilização adequada possibilita uma melhoria no acompanhamento da saúde dos cidadãos e na qualidade da atenção prestada, e deve ser composto de todas as fichas clínicas utilizadas pelos profissionais de saúde que atendem determinada família, disponibilizando uma gama de informações pertinentes ao

paciente e sua família capaz de facilitar aos profissionais envolvidos (médicos, dentistas, enfermeiros) o diagnóstico e a melhor conduta no plano tratamento destes pacientes.

2.4 A Triagem para o Risco de Insegurança Alimentar

Conforme a Política Nacional de Alimentação e Nutrição (PNAN), aprovada em 1999, entende-se que a alimentação e nutrição adequadas dos indivíduos são dois requisitos básicos para a eficaz promoção e a proteção da saúde, permitindo e fomentando assim o pleno desenvolvimento e a qualidade de vida das famílias acompanhadas.

Desde 2023 o nível de insegurança alimentar de uma família pode ser aferido no Prontuário Eletrônico e-SUS APS através do instrumento de Triagem para Risco de Insegurança Alimentar (TRIA). A TRIA é um instrumento validado academicamente, derivado da EBIA (Escala Brasileira de Medida Domiciliar de Insegurança Alimentar) cujo intuito é identificar o risco de insegurança alimentar no domicílio (CARVALHO et. al., 2022; POBLACION et al., 2021). A TRIA é composta de duas das 14 perguntas da EBIA, e tais perguntas são feitas durante as visitas domiciliares de acompanhamento.

As perguntas da TRIA são:

- I - Nos últimos três meses, os alimentos acabaram antes que você tivesse dinheiro para comprar mais comida?
- II - Nos últimos três meses, você comeu apenas alguns alimentos que ainda tinha, por que o dinheiro acabou?

Quando respondidas com “sim” por ao menos um dos indivíduos maiores de idade da família, na entrevista de acompanhamento, entende-se que esta família encontra-se em Risco de Insegurança Alimentar, segundo a Nota Técnica nº 30/2025 do Ministério da Saúde.

O Prontuário Eletrônico E-SUS APS por sua vez, permite aferir tal risco através de relatórios gerenciais, que possibilitam busca ativa de tais famílias fora do fluxo do atendimento, ou junto às informações clínicas exibidas na Folha de Rosto do prontuário individual do cidadão.

2.5 Arquiteturas e Tecnologias do e-SUS APS

O projeto de sistemas web modernos, como o e-SUS APS, é sustentado por uma estrutura composta de tecnologias, padrões e arquiteturas cujo objetivo é assegurar aspectos essenciais como escalabilidade, estabilidade, facilidade de manutenção e qualidade para o usuário. Na sequência, serão abordados os principais elementos que contribuem para essas características.

2.5.1 Interfaces de Programação de Aplicações (APIs)

As Interfaces de Programação de Aplicações (APIs), funcionam como um conjunto de protocolos e regras para a troca de informações entre softwares, permitindo que diferentes sistemas interajam e compartilhem dados. Essa comunicação ocorre sem que seja necessário conhecer os detalhes internos de cada sistema. Ao estabelecer protocolos claros para requisições e respostas, as APIs favorecem a interoperabilidade e a modularidade, facilitando tanto a integração entre partes de um mesmo sistema quanto entre plataformas distintas.

GraphQL

O e-SUS APS opera com uma arquitetura de comunicação híbrida, que estrategicamente combina a solidez do RESTful com a eficiência do GraphQL,

porém, este último, é atualmente a principal forma de comunicação entre o *back-end* e o *front-end*. O RESTful acaba ficando para operações específicas e geralmente assíncronas.

GraphQL é uma linguagem de consulta e um ambiente de execução de código aberto, concebido para superar as limitações do REST. Enquanto o REST utiliza múltiplos *endpoints* fixos para retornar estruturas de dados completas, o GraphQL centraliza todas as interações em um único *endpoint* (CHINA; JACKSON, [s.d.]). Sua principal inovação é a natureza declarativa da busca: o cliente define exatamente os campos e a estrutura dos dados que necessita em uma única *payload*, transferindo o poder de decisão sobre o formato da resposta do servidor para o cliente (AI FUTURE SCHOOL, [s.d.]). Esse modelo aumenta a eficiência, pois mitiga o *over-fetching* (receber dados desnecessários) e o *under-fetching* (exigir múltiplas requisições sequenciais para obter todos os dados) (GEEKSFORGEEKS, [s.d.]). No cerne da arquitetura está o Sistema de Tipos Forte (Schema), escrito em SDL, que atua como um contrato, validando rigorosamente as consultas antes da execução. Uma característica crucial é a introspecção, que permite aos clientes consultar o próprio *Schema* em tempo de execução, gerando documentação automática e facilitando o desenvolvimento de *tooling* avançado (F5, [s.d.]).

A execução das consultas GraphQL é orquestrada por funções especializadas chamadas Resolvers. Um Resolver é o componente que conecta o *Schema* às fontes de dados subjacentes (bancos de dados, microsserviços, etc.) (AWS, [s.d.]). Existem três papéis principais: o Query Resolver, que lida com operações de leitura de dados (*queries*); o Mutation Resolver, que garante a lógica para alterações de estado (escrita, modificação); e o Field Resolver, invocado para determinar o valor de um campo aninhado específico (GRAPHQL FOUNDATION, [s.d.]). Essa granularidade na execução permite que diferentes *Field Resolvers* acessem fontes de dados heterogêneas e as agreguem em uma única resposta unificada, fornecendo ao cliente uma visão coesa dos dados (GRAPHQL FOUNDATION, [s.d.]).

A execução independente dos *Field Resolvers* introduz o Problema N+1, um gargalo de desempenho que ocorre quando a busca de uma lista de \$N\$ itens aciona \$N\$ requisições adicionais ao banco de dados para buscar os dados

relacionados (GRAPHQL FOUNDATION, [s.d.]). Para contornar essa ineficiência, o uso do utilitário DataLoader é um padrão obrigatório (GRAPHQL FOUNDATION, [s.d.]). O DataLoader otimiza as operações de I/O por meio de dois mecanismos: Batching (Agrupamento) e Caching (Armazenamento em Cache). O Batching intercepta múltiplas chamadas assíncronas de *Field Resolvers* que ocorrem no mesmo ciclo de execução e as agrupa em uma única consulta otimizada para o banco de dados (por exemplo, usando `SELECT ... WHERE id IN (...)`), reduzindo drasticamente o número de *round trips* (APOLLO GRAPHQL, [s.d.]). O Caching armazena os resultados para a duração da requisição, eliminando buscas redundantes para o mesmo recurso. Dessa forma, o DataLoader mantém a composabilidade lógica do GraphQL, ao mesmo tempo que garante a eficiência máxima no acesso físico aos dados (PIXEL FREE STUDIO, [s.d.]).

2.5.2 Padrões de Projeto de Software Relevantes

Durante o processo de criação de software, os chamados Padrões de Projeto (Design Patterns) representam abordagens consolidadas e reutilizáveis para lidar com desafios recorrentes de arquitetura e design. Em vez de fornecerem implementações específicas ou trechos de código, esses padrões funcionam como guias conceituais que orientam a organização dos elementos de um sistema, promovendo soluções eficazes e adaptáveis. Ao adotar esses modelos, os desenvolvedores passam a compartilhar uma linguagem comum, o que facilita a comunicação técnica e contribui para o desenvolvimento de aplicações mais fáceis de manter, expandir e entender. (REFACTORING.GURU, [s.d.]).

2.5.3 Java e Spring Boot (Back-end)

O Java Spring Boot, é um framework de código aberto, adotado no desenvolvimento do e-SUS APS e afetando consequentemente o Prontuário Familiar, uma solução de

código aberto que visa tornar mais ágil e descomplicada a criação de aplicações Java dentro do ecossistema Spring. Seguindo o princípio de “convenção sobre configuração”, o Spring Boot automatiza grande parte da configuração inicial, disponibiliza servidores web integrados como Tomcat, Jetty ou Undertow, e realiza o gerenciamento de dependências, permitindo que os desenvolvedores concentrem seus esforços na implementação da lógica da aplicação.

A escolha do Spring Boot como framework se fundamenta em sua capacidade de acelerar o desenvolvimento de APIs seguras e robustas, sejam elas RESTful ou baseadas em GraphQL, variedade de documentação e uma comunidade ativa (SPRING, [s.d]). Seu ecossistema abrangente inclui módulos como Spring Security para autenticação e autorização, Spring Data para manipulação de dados, e integração facilitada com diversas tecnologias. Esses recursos tornam o back-end da aplicação mais escalável, confiável e simples de manter.

2.5.4 Banco de Dados - Postgres e Oracle

O e-SUS APS, dá suporte a dois tipos de bancos de dados. O PostgreSQL é um Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR) de código aberto, amplamente reconhecido pela sua estabilidade, robustez e estrita conformidade com os padrões SQL, incluindo a garantia de transações em total conformidade ACID (Atomicidade, Consistência, Isolamento e Durabilidade) (INSTACLUSTER, [s.d.]) Sua arquitetura fundamental é Objeto-Relacional, um modelo que não apenas suporta dados relacionais tradicionais, mas também permite o tratamento nativo de objetos complexos e a definição de tipos de dados personalizados (POSTGRESQL, [s.d.]).

A extensibilidade é um pilar chave do sistema. Ele fornece APIs definidas que permitem a rápida integração de novas funcionalidades pela comunidade e por desenvolvedores, como a criação de novos tipos de índices e o manejo de estruturas de dados específicas (POSTGRESQL [s.d.]). Em termos de escalabilidade, o PostgreSQL demonstrou ser altamente capaz, gerenciando ambientes de data warehousing com volumes que chegam a terabytes e petabytes de dados, além de suportar inúmeros usuários concorrentes (INSTACLUSTER, [s.d.]).

O Oracle Database é um SGBD proprietário que se consolidou como o padrão de mercado para ambientes corporativos de missão crítica, onde a prioridade é a máxima escalabilidade, segurança e alta disponibilidade(ORACLE, [s.d.]). O foco de sua arquitetura é a continuidade do negócio, e para isso, a empresa utiliza o framework Maximum Availability Architecture.⁵

O MAA serve como um guia estruturado para que as corporações implementem soluções que garantam a proteção de dados e a continuidade do serviço, mitigando riscos associados a tempos de inatividade (ORACLE, [s.d.]). O alto nível de resiliência e escalabilidade horizontal são garantidos por tecnologias proprietárias incluídas na licença Enterprise Edition, como o Oracle Real Application Clusters (RAC).⁵ O RAC utiliza tecnologia de cluster para distribuir cargas de trabalho entre servidores e fornecer resiliência contra falhas de instância. Além disso, o Oracle Sharding oferece disponibilidade extrema ao distribuir subconjuntos do banco de dados (shards) em servidores independentes, permitindo que a aplicação continue operando mesmo com a falha de um shard.

Apesar das diferenças em arquitetura e licenciamento (código aberto vs. proprietário), tanto o PostgreSQL quanto o Oracle Database oferecem excelente compatibilidade e são escolhas robustas dentro do ecossistema Spring Boot. O *framework* facilita a integração através de módulos como o Spring Data JPA, que utiliza o Hibernate por padrão.

Ambos os SGBDs possuem *drivers* JDBC maduros e bem mantidos (o Oracle, proprietário; o PostgreSQL, via *community*), permitindo que os desenvolvedores configurem rapidamente a persistência de dados. A abstração fornecida pelo Spring Data assegura que as aplicações possam interagir com qualquer um dos bancos de forma consistente, permitindo aos desenvolvedores focar na lógica de negócios, enquanto o *framework* cuida da tradução das operações para o dialeto SQL específico de cada SGBD.

2.5.5 TypeScript, React e Design Systems (Front-end)

Seguindo a *stack* utilizado no e-SUS APS pelo laboratório Bridge, para o desenvolvimento do projeto foi adotada a biblioteca React em conjunto com TypeScript. React, mantida pelo Facebook, é uma biblioteca JavaScript de código aberto voltada para a criação de interfaces interativas e reutilizáveis, com destaque para aplicações de página única (SPAs). Seu modelo de desenvolvimento é centrado em componentes, unidades independentes de código que encapsulam seu próprio estado e comportamento.

A escolha do React se deu, especialmente por sua habilidade de construir interfaces dinâmicas que respondem rapidamente às mudanças de estado, sem exigir o recarregamento completo da página. Isso é viabilizado pelo uso do DOM (Document Object Model) virtual, que otimiza as atualizações na interface.

A utilização do TypeScript ao desenvolvimento com React introduz uma tipagem estática ao JavaScript. Essa camada adicional de segurança permite identificar erros durante a fase de desenvolvimento, antes mesmo da execução, o que torna o processo mais confiável. Além disso, contribui para uma manutenção mais eficiente e para o entendimento mais rápido por parte da equipe de desenvolvimento.

A adoção de Design Systems tem se mostrado uma abordagem eficaz para garantir uniformidade visual, melhorar a experiência do usuário e agilizar o desenvolvimento de interfaces. No front-end, um design system funciona como um conjunto central de diretrizes, padrões e componentes reutilizáveis de UI, orientando tanto o design quanto a implementação de interfaces. No projeto, foi utilizado o BOLD Design System, criado pelo Laboratório Bridge da UFSC (LABORATÓRIO BRIDGE, [s.d.]a), como base para a construção dos elementos visuais da aplicação. Entre suas principais características estão o compromisso com a acessibilidade, dentro das especificações WCAG, o caráter de código aberto e a flexibilidade para personalizações, características pontuais que oferecem uma interface padronizada, acessível e adaptável.

2.6 Metodologias de Desenvolvimento de Software

A definição da metodologia de desenvolvimento é um fator determinante para o sucesso de um projeto de software, influenciando diretamente sua organização, execução e entrega. Cada abordagem metodológica propõe formas distintas de lidar com a gestão de equipes, definição de requisitos, controle de prazos e garantia da qualidade do produto. Em cenários que exigem maior capacidade de adaptação e resposta rápida a mudanças, as metodologias ágeis têm ganhado destaque por sua flexibilidade e foco na entrega contínua de valor.

2.6.1 Desenvolvimento Ágil: Manifesto e Princípios

O modelo Ágil de desenvolvimento de software propõe uma abordagem centrada na entrega contínua e progressiva de soluções funcionais, com foco na colaboração entre os envolvidos, na adaptação constante às mudanças e na busca pela satisfação do cliente. Essa filosofia foi consolidada em 2001 por meio do “Manifesto para o Desenvolvimento Ágil de Software”, que estabelece quatro valores essenciais (MANIFESTO PARA DESENVOLVIMENTO ÁGIL DE SOFTWARE, 2001):

- Valorizar pessoas e interações acima de ferramentas e processos.
- Priorizar software funcional em vez de documentação extensa.
- Incentivar a parceria com o cliente mais do que a rigidez contratual.
- Adaptar-se às mudanças ao invés de seguir planos inflexíveis.

Esses valores são complementados por doze princípios que orientam o funcionamento das equipes ágeis, destacando práticas como entregas frequentes, comunicação direta, simplicidade nas soluções, autonomia dos times e melhoria contínua. A aplicação dessas práticas busca reduzir os riscos inerentes a projetos extensos e com requisitos voláteis, permitindo que as equipes respondam com agilidade a novas necessidades e ao retorno dos usuários.

2.6.2 Scrum - Um *framework* ágil

O Scrum configura-se como uma das estruturas ágeis mais difundidas e consolidadas no contexto da gestão de projetos e do desenvolvimento de produtos complexos. Diferentemente de metodologias prescritivas, o Scrum apresenta-se como um conjunto conceitual que estabelece papéis fundamentais, eventos, artefatos específicos e um conjunto de regras que orientam a implementação dos princípios do desenvolvimento ágil. Sua fundamentação teórica repousa sobre dois pilares centrais: o empirismo, que sustenta a ideia de que o conhecimento é construído a partir da experiência prática e da tomada de decisões com base em evidências, e o pensamento enxuto (*lean thinking*), cuja premissa é a eliminação de desperdícios e a concentração de esforços nas atividades que efetivamente agregam valor ao produto ou serviço (SCRUM.ORG, [s.d.]).

O framework Scrum é composto por um conjunto estruturado de elementos fundamentais que orientam a aplicação dos princípios ágeis no desenvolvimento de software. Esses elementos incluem papéis, eventos e artefatos, os quais interagem de forma coordenada para promover a entrega contínua de valor.

Abaixo são apresentados os principais papéis do Scrum:

- *Product Owner* (PO): Detém a responsabilidade de maximizar o valor do produto gerado pelo trabalho da equipe de desenvolvimento. Para isso, gerencia o *Product Backlog*, assegurando sua clareza, visibilidade e transparência para todos os envolvidos.
- *Scrum Master* (SM): Atua como facilitador do processo, promovendo e sustentando a aplicação do Scrum conforme definido em seu guia oficial. Exerce a função de líder servidor, auxiliando a equipe na compreensão dos fundamentos, práticas e valores do Scrum.
- Equipe de Desenvolvimento (*Development Team*): Composta por profissionais multifuncionais e auto-organizáveis, é responsável pela entrega de incrementos funcionais do produto ao final de cada ciclo de desenvolvimento (*Sprint*).

Abaixo são apresentados os principais eventos do Scrum:

- *Sprint*: Intervalo de tempo fixo, geralmente entre uma e quatro semanas, durante o qual é desenvolvido um incremento utilizável e potencialmente entregável do produto.
- *Sprint Planning*: Reunião inicial da *Sprint* destinada ao planejamento das atividades e definição dos objetivos a serem alcançados.
- *Daily Scrum*: Encontro diário de curta duração (aproximadamente 15 minutos), voltado à sincronização das tarefas e ao planejamento das próximas 24 horas.
- *Sprint Review*: Realizada ao término da *Sprint*, tem como finalidade a inspeção do incremento produzido e a eventual atualização do *Product Backlog*.
- *Sprint Retrospective*: Ocorre após a *Sprint Review* e antes da próxima *Sprint Planning*, sendo dedicada à reflexão sobre o processo e à proposição de melhorias para os ciclos subsequentes.

Abaixo são apresentados os principais artefatos do Scrum:

- *Product Backlog*: Lista priorizada e dinâmica de requisitos necessários para a evolução do produto, constituindo a principal fonte de demandas.
- *Sprint Backlog*: Subconjunto do *Product Backlog* selecionado para a *Sprint* atual, acompanhado de um plano de ação para atingir a meta estabelecida.
- Incremento: Representa a soma dos itens concluídos do *Product Backlog* durante a *Sprint*, agregando valor aos incrementos anteriores e compondo uma versão potencialmente liberável do produto.

O Scrum adota uma abordagem iterativa e incremental, com o propósito de aumentar a previsibilidade e reduzir os riscos associados ao desenvolvimento ao mesmo tempo, em que promove a entrega frequente de valor ao cliente.

2.6.3 Engenharia de Requisitos Ágil: O Foco nas Histórias de Usuário

No contexto das metodologias ágeis, o processo de elicitação, documentação e gerenciamento de requisitos apresentam características distintas em relação às abordagens tradicionais. Uma das técnicas amplamente utilizadas para representar funcionalidades sob a ótica do usuário final é a formulação de Histórias de Usuário (User Stories). Essas histórias consistem em descrições sucintas e objetivas de uma funcionalidade desejada, geralmente estruturadas segundo o modelo: “Como um <tipo de usuário>, eu quero <realizar alguma ação> para que <obtenha algum benefício>” (AGILE ALLIANCE, [s.d.]).

As Histórias de Usuário integram o Product Backlog — uma lista dinâmica e priorizada de requisitos conhecidos para o produto — e desempenham papel central na comunicação entre o Product Owner, a equipe de desenvolvimento e demais stakeholders. Essa abordagem favorece a orientação para o valor entregue ao usuário e, com frequência, é complementada por Critérios de Aceitação, os quais definem as condições necessárias para que uma história seja considerada concluída. Para garantir a qualidade e a clareza dessas histórias, recomenda-se que elas sigam o acrônimo INVEST, que contempla os atributos: Independente, Negociável, Valiosa, Estimável, Pequena e Testável.

2.6.4 O Processo de Descoberta de Produto utilizando o b_Thinking

Prévio ao desenvolvimento, tornou-se imprescindível a realização de um processo estruturado de descoberta de produto, com o objetivo de assegurar que a solução proposta estivesse alinhada tanto às demandas do negócio quanto às necessidades dos usuários finais. Para esse fim, foi adotada a abordagem b_Thinking, desenvolvida pelo Laboratório Bridge, a qual oferece uma estrutura metodológica voltada à ideação, validação e priorização de funcionalidades.

O processo foi conduzido com base nas etapas definidas pelo b_Thinking, conforme representado no fluxograma da metodologia. A aplicação prática dessa abordagem ocorreu da seguinte maneira:

As etapas iniciais do processo, Briefing e Imersão, foram fundamentais para a compreensão aprofundada das demandas dos usuários da atenção primária, bem como para o alinhamento com os objetivos estratégicos do projeto. Na fase de Definição, os conhecimentos adquiridos foram sistematizados, permitindo a delimitação do escopo principal da solução. Em seguida, a fase de Ideação possibilitou a formulação das primeiras hipóteses de solução, as quais foram posteriormente representadas por meio de protótipos. A Validação dessas propostas ocorreu em parceria com uma designer do Laboratório Bridge, assegurando a aderência às diretrizes de acessibilidade estabelecidas. Com as ideias validadas, as fases de Delimitação e Refinamento desempenharam papel crucial na conversão dos protótipos em histórias de usuário bem definidas e priorizadas, orientando o trabalho da equipe de desenvolvimento.

A implementação das funcionalidades foi conduzida por meio de ciclos iterativos de Sprint, característicos de abordagens ágeis. Ressalta-se, contudo, que o modelo adotado passou por adaptações: em virtude da ausência de contato contínuo com os usuários finais, as etapas de Avaliação da release e de Melhoria contínua foram realizadas com base em análises internas, e não por meio de validação direta pós-entrega, o que representa uma modificação no ciclo tradicional da metodologia aplicada (LABORATÓRIO BRIDGE, [s.d.]b).

3 TRABALHOS RELACIONADOS

Foi realizada uma pesquisa em bases de dados acadêmicas em busca de trabalhos equiparáveis ou correlatos à esse, para que fosse possível verificar o grau de relevância e inovação da ideia além de, claro, a pertinência do projeto.

O método de pesquisa pautou-se em realizar buscas que combinavam, alternadamente, os termos “prontuário”, “sistema”, “e-sus aps”, “família”, “familiar”, “saúde”, “pública” e “cuidado”.

Foram selecionados dezenas de trabalhos que inicialmente foram agrupados por título e, em seguida, filtrados pela leitura de seu resumo. Após a leitura dos trabalhos restantes, foi possível verificar elevada relevância e pertinência do objetivo desta monografia, porém todos os artigos encontrados que se adequaram à isso já foram citados na fundamentação teórica.

Não foi possível encontrar trabalhos correlatos que apresentassem a mesma proposta de desenvolvimento de um prontuário familiar eletrônico associado à estratégia de saúde da família do governo federal, o que demonstrou, pelo viés do sobrevivente, a adesão da proposta ao quesito inovação.

É de conhecimento tácito do autor e seus colegas de trabalho que aplicações privadas de Prontuário Eletrônico disponibilizam soluções equiparáveis, mas não foram encontrados materiais teóricos ou demonstrativos à respeito.

4 METODOLOGIA E PROJETO DO PRONTUÁRIO FAMILIAR

Esta seção foi feita para explicar a jornada de construção do projeto. Detalhando a metodologia que guiou o trabalho, como foi feito o levantamento e análise das necessidades do sistema e qual foi o desenho da arquitetura que foi escolhida.

4.1 Metodologia de Desenvolvimento

O desenvolvimento do Prontuário Familiar orientou-se por uma metodologia que priorizou a natureza iterativa e incremental com intensa aplicação de práticas ágeis, buscando garantir a entrega contínua de valor e uma alta capacidade de resposta a novos entendimentos e requisitos que surgissem durante o andamento do projeto. O arcabouço metodológico deste trabalho foi construído sobre a união do *framework Scrum* para a gestão do desenvolvimento e o método b_Thinking para a descoberta do produto.

A preferência por metodologias ágeis, tendo o *Scrum* como principal *framework* de gerenciamento, justifica-se pela particularidade do projeto, que envolvia a criação de uma solução inédita em um campo com especificidades significativas. O *Scrum* possibilitou a organização do trabalho em ciclos curtos e bem definidos, conhecidos como *Sprints* (com duração fixa de duas semanas), o que facilitou a inspeção regular do progresso e a adaptação do planejamento. O planejamento de cada *Sprint* visava a entrega de um incremento funcional do software, permitindo validações parciais de maneira ágil.

A gestão do projeto sob o modelo *Scrum* envolveu a criação e manutenção de um *Product Backlog*, que agrupava as funcionalidades e requisitos do sistema, priorizados e detalhados como Histórias de Usuário. No começo de cada *Sprint*, realizava-se a *Sprint Planning* para a seleção dos itens a serem desenvolvidos, seguida, ao final, pela *Sprint Review* para a demonstração das funcionalidades

concluídas e pela *Sprint Retrospectiva* para aprimoramento do processo de trabalho. Para o acompanhamento visual das tarefas e do fluxo de desenvolvimento dentro das *Sprints*, foi empregado um quadro *Kanban*.

Em paralelo à aplicação do gerenciamento ágil, a abordagem *b_Thinking* de descoberta de produto forneceu apoio robusto à fase inicial de concepção do sistema e à evolução contínua dos requisitos. Esta metodologia foi essencial para aprofundar o entendimento das necessidades dos usuários, permitindo a consolidação de um *épico* que definia o objetivo central do software. Este *épico*, por sua vez, foi desdobrado em *Histórias de Usuário* detalhadas, as quais formaram a base do *Product Backlog* e orientaram o desenvolvimento em ciclos iterativos. O *b_Thinking* colocou ênfase na ideação, na prototipação e na validação contínua, garantindo que o produto estivesse alinhado com as demandas reais dos profissionais de saúde e dos pacientes, estabelecendo um ciclo de aprendizado e ajustes constantes.

O controle do código-fonte e a gestão de versões foram executados por meio do Git e da plataforma GitHub, ferramentas que também facilitaram o rastreamento de tarefas (*issues*) e a revisão de código (*pull requests*). O processo de revisão de código demonstrou ser uma etapa crítica para a garantia da qualidade do projeto, sendo fortalecido pela mentoria e validação técnica de todo o código-fonte pela equipe NP-Completo. A contribuição da equipe, composta por desenvolvedores de referência no Laboratório Bridge, instituição com experiência notável em sistemas, como o próprio e-SUS APS, resultou em uma base de código visivelmente mais madura, consolidada e conforme as melhores práticas de mercado. A adoção dessa combinação de ferramentas e práticas metodológicas teve como propósito estabelecer um ciclo de desenvolvimento eficiente, transparente e adaptável.

4.2 Levantamento e Análise de Requisitos

O sucesso de um projeto de software está intrinsecamente ligado à rigorosa execução da etapa de levantamento e análise de requisitos, visto que ela delimita o

escopo e especifica as funcionalidades essenciais do sistema. No contexto do desenvolvimento do Prontuário familiar, este processo foi conduzido de maneira iterativa e evolutiva. Inicialmente, a compreensão aprofundada das necessidades dos usuários e do panorama da Atenção Primária à Saúde foi estabelecida por meio da abordagem de descoberta de produto *b_Thinking*, conforme detalhado na seção precedente.

As necessidades e os objetivos chave identificados foram sistematizados em um Épico central, que funcionou como um direcionamento estratégico de alto nível para o desenvolvimento. Posteriormente, este épico foi refinado e desmembrado em Histórias de Usuário pormenorizadas. Cada história representa uma capacidade ou um agrupamento de funcionalidades sob a ótica do usuário, especificando, de forma clara, os critérios para sua aceitação e validação.

4.2.1 Épico do Prontuário Familiar

O desenvolvimento do Prontuário familiar foi estrategicamente orientado pelo Épico a seguir, o qual condensa a visão central e de alto nível do produto:

- Épico:
 - “Eu, como profissional de saúde da Estratégia de Saúde da Família, quero visualizar informações sociais, clínicas e de acompanhamento da família do cidadão, durante o atendimento ou acompanhamento do mesmo, para qualificar o cuidado em saúde do mesmo no âmbito da Atenção Longitudinal.”

4.2.2 Requisitos Funcionais (RF)

Os Requisitos Funcionais (RF) especificam os comportamentos e as funcionalidades que o módulo do Prontuário familiar deve prover. Tais requisitos

foram diretamente mapeados a partir das Histórias de Usuário e de seus critérios de aceitação, servindo como base para a validação do produto. Os principais requisitos funcionais identificados e implementados são listados abaixo:

- **RF01 - Prontuário familiar no contexto do cidadão atual:** O sistema deve carregar e exibir o Prontuário familiar no contexto do cidadão atualmente selecionado ou em atendimento.
- **RF02 - Lista de familiares:** O sistema deve exibir uma lista de todos os cidadãos cadastrados na família do cidadão do contexto.
- **RF03 - Dados dos familiares:** O sistema deve apresentar o nome completo de cada familiar na lista, idade, data de nascimento, CPF ou CNS.
- **RF04 - Dados opcionais dos familiares:** O sistema deve mostrar o grau de parentesco de cada familiar em relação ao indivíduo responsável pela família, quando preenchido.
- **RF05 - Acompanhamento de vulnerabilidade social:** O sistema deve apresentar, na seção da composição familiar, se a família possui algum risco de insegurança alimentar e/ou faz parte do programa Bolsa família.
- **RF06 - Acesso a aba do Prontuário familiar:** O sistema deve permitir que o usuário visualize a aba dedicada ao Prontuário Familiar, sempre que o usuário tenha permissão e esteja em um contexto onde o prontuário do Cidadão pode ser visualizado.
- **RF07 - Navegação entre os prontuários dos familiares:** O sistema deve permitir que o usuário navegue para o prontuário do cidadão de qualquer familiar diretamente a partir da listagem.
- **RF08 - Dados do imóvel:** O sistema deve exibir uma seção específica contendo as informações detalhadas do imóvel onde o cidadão e sua família residem.
- **RF09 - Dados do endereço:** O sistema deve apresentar o endereço completo do imóvel, incluindo logradouro, número, bairro, CEP, entre outros.
- **RF10 - Condições de moradia do imóvel:** O sistema deve exibir dados sobre as condições de moradia(ex: situação de moradia, localização, destino do lixo, escoamento do banheiro).

- **RF11 - Lista das últimas visitas domiciliares:** O sistema deve exibir uma listagem do histórico de Visitas Domiciliares realizadas ao imóvel da família e seus moradores.
- **RF12 - Dados das visitas domiciliares:** O sistema deve apresentar, para cada visita listada, a data da visita, o profissional que realizou e sua equipe, além dos motivos que levaram o profissional a realizar a mesma.

4.2.3 Requisitos Não Funcionais (RNF)

A especificação dos Requisitos Não Funcionais (RNF) é fundamental para delinear os atributos de qualidade e as restrições operacionais do sistema, abrangendo critérios como segurança, desempenho e manutenibilidade. No contexto do Prontuário familiar, os seguintes RNFs foram priorizados e incorporados:

- **RNF01 - Desempenho:**
 - O módulo deve carregar a lista completa de familiares, dados do imóvel e histórico de visitas com latência mínima perceptível (em menos de 0,5 segundos), após a solicitação do usuário.
- **RNF02 - Segurança:**
 - O acesso à visualização do prontuário de cada familiar (RF06) deve ser validado pelo sistema de controle de acesso por recurso, garantindo que o profissional tenha permissão explícita para o acesso.
 - O tratamento de quaisquer dados de cidadãos deve seguir os princípios da Lei Geral de Proteção de Dados (LGPD), minimizando a coleta e exposição de dados e garantindo seu uso apenas para a finalidade do módulo.
 - Auditoria: O sistema deve registrar nas trilhas de auditoria do e-SUS APS o profissional que requisitou acesso ao prontuário familiar do cidadão do contexto, para fins de verificação e transparência aos usuários, sem expor dados sensíveis desnecessariamente.
- **RNF03 - Escalabilidade:**

- A arquitetura do módulo deve permitir o aumento da capacidade de processamento sem exigir alterações no código, como em casos que o cidadão possua mais famílias e com mais membros.

- **RNF04 - Manutenibilidade:**

- É essencial que o sistema seja concebido com uma arquitetura de blocos independentes, onde haja uma distinção inequívoca de responsabilidades (incluindo as áreas de interface, lógica de serviço e persistência de dados). O software precisa seguir padrões de design, como o *Repository* e o *Service*, para assegurar que os componentes tenham alta coesão e baixa dependência mútua, permitindo que ele seja facilmente atualizado e expandido no futuro.
- O código-fonte do módulo deve aderir a um padrão de codificação único e amplamente aceito pela equipe de desenvolvimento.
- Todas as alterações futuras em um requisito funcional devem ser implementadas e testadas sem causar regressões nas funcionalidades existentes.

- **RNF05 - Compatibilidade:**

- O módulo deve ser totalmente funcional e exibir o *layout* corretamente nas duas versões mais recentes dos navegadores Google Chrome, Mozilla Firefox e Microsoft Edge.
- O módulo deve ser responsivo, adaptando-se corretamente a telas com resolução mínima de 1024x768 píxeis, sem a necessidade de *scroll* horizontal para os elementos principais.

- **RNF06 - Usabilidade:**

- O layout deve utilizar cores e fontes que sigam o padrão de identidade visual do e-SUS APS, para manter a coerência visual com o restante do sistema.
- A navegação entre a lista de familiares e os seus respectivos prontuários (RF06) deve ser intuitiva e exigir o mínimo de cliques possível (idealmente um único clique).

4.3 Arquitetura do Sistema

4.3.1 Arquitetura do Back-end

O Prontuário Familiar foi concebido como uma funcionalidade do e-SUS APS. O *back-end* do sistema hospedeiro apresenta uma arquitetura robusta, com camadas bem estruturadas, com divisões de responsabilidade bem definidas e baseada em tecnologias contemporâneas de desenvolvimento. Essa integração permitiu que o novo módulo se beneficiasse da infraestrutura e dos padrões de engenharia de software existentes.

- Tecnologia Principal: Desenvolvido em Java 17 com Spring Boot, aproveitando seu ecossistema para criar serviços eficientes baseados em GraphQL. Esta abordagem garante a flexibilidade na comunicação e a otimização na recuperação de dados, além de utilizar o ecossistema Spring para o gerenciamento de transações e segurança.
- Padrões de Projeto: Os padrões de projeto Service Layer e Repository foram aplicados para garantir uma clara separação de preocupações. O primeiro concentrou a lógica de negócio, enquanto o segundo abstraiu a persistência de dados, conforme o referencial teórico que embasou o projeto.
- Containerização: O Docker foi adotado para garantir a consistência da camada de dados.
- Segurança: Além do Spring Boot Security, o e-SUS APS controla o acesso a *queries* e *mutations* do *back-end* através de recursos, um mecanismo que impede que usuário não autorizados acessem dados através da API.

4.3.2 Arquitetura do Front-end

O front-end do módulo Prontuário Familiar, a camada responsável pela interface com o usuário e pela apresentação dinâmica das informações, foi desenvolvido utilizando React em conjunto com TypeScript.

A arquitetura desta camada foi estrategicamente projetada para ser componentizada, declarativa e reativa. A adoção de componentes visa a reutilização e facilita a manutenção, enquanto o uso do TypeScript promove maior segurança de tipos e a rastreabilidade de erros durante o desenvolvimento. O gerenciamento eficiente do estado foi priorizado, visando a escalabilidade e o desempenho da aplicação no e-SUS APS.

- **Tipagem Estática com TypeScript:** A utilização de TypeScript em todo o desenvolvimento do front-end permitiu a definição de interfaces e tipos para os dados e propriedades dos componentes, aumentando a robustez do código, auxiliando na detecção precoce de erros e melhorando a manutenibilidade e a compreensão da base de código.
- **Design System e Consistência Visual:** Conforme mencionado, a utilização dos componentes e diretrizes do *d* foi um pilar para garantir a consistência visual, a usabilidade e a aplicação de boas práticas de acessibilidade na interface.

Comunicação com o Back-end

O Prontuário Familiar adota o GraphQL como principal protocolo de comunicação entre o *front-end* e o *back-end*. Essa escolha arquitetural se fundamenta na necessidade de otimização do consumo de informações e na garantia de baixa latência nas consultas.

No *front-end* em React/TypeScript, a comunicação com o *back-end* é gerenciada pelo Apollo Client, uma biblioteca cliente de GraphQL. O fluxo de interação ocorre da seguinte forma:

- Definição das Consultas (.gql): As operações de busca (*Queries*), manipulação (*Mutations*) e atualização em tempo real são definidas em arquivos separados com a extensão “.gql”. Essa prática garante a clareza e a separação entre a definição da consulta e a lógica do componente.
- Geração de *Hooks*: O Apollo Client processa essas definições e as transforma em funções *hook* personalizadas do React (ex: *useQuery*, *useMutation*).
- Consumo nos Componentes: Nos componentes de interface, essas funções são invocadas diretamente, o que simplifica o gerenciamento do ciclo de vida das requisições, incluindo o estado de carregamento (*loading*), o tratamento de erros e o *caching* de dados.

Essa abordagem declarativa permite que o *front-end* solicite apenas os campos estritamente necessários para renderizar a interface, minimizando o tráfego de rede e acelerando o desenvolvimento de interfaces reativas.

4.3.3 Banco de Dados

O e-SUS APS foi projetado para ser compatível com um ambiente de dados heterogêneo, suportando dois tipos de bancos de dados relacionais (SGBDRs), PostgreSQL e Oracle Database, o por se encontrar dentro desse sistema, o módulo do Prontuário Familiar seguiu as mesmas definições.

Integração com o Back-end (Spring Boot e JPA)

A integração do *back-end* Java/Spring Boot com ambas as bases de dados é facilitada pelo uso do Spring Data JPA, que utiliza o Hibernate como provedor de persistência

A arquitetura do *back-end* segue o Padrão Repository , garantindo uma clara separação de responsabilidades. As classes anotadas com @Repository definem o contrato de acesso a dados, isolando a lógica de negócio. Esta lógica é concentrada na Camada de Serviço (classes @Service), que orquestra as operações de *back-end* e interage com os Repositórios(classes @Repository).

Essa abstração oferecida pelo Spring Data assegura que as aplicações possam interagir com o PostgreSQL ou o Oracle de forma consistente , permitindo aos desenvolvedores focar na lógica de negócio e garantindo a compatibilidade necessária para a arquitetura do e-SUS APS.

5 IMPLEMENTAÇÃO DO PRONTUÁRIO FAMILIAR

Este capítulo detalha a fase de construção do módulo Prontuário Familiar, traduzindo o projeto arquitetural e os requisitos especificados em código funcional. A implementação seguiu a metodologia ágil adotada, focando na integração plena com a arquitetura do e-SUS APS.

Nas seções a seguir, serão detalhados o ambiente de desenvolvimento e ferramentas utilizadas, o processo de implementação do *back-end*, que inclui a definição dos *resolvers* e a integração com a camada de serviços e acesso a dados. Em seguida, será abordada a implementação do *front-end*, destacando a estrutura em componentes React/TypeScript. Por fim, serão discutidos os desafios da implementação e soluções adotadas.

Ressalta-se que, devido à natureza proprietária do código-fonte do sistema e-SUS APS, a análise e a descrição detalhada de componentes, classes e da estrutura interna de código que transcende o módulo desenvolvido (Prontuário Familiar) será limitada. O detalhamento foca-se nas interfaces, nas regras de negócio e na arquitetura de integração, respeitando as restrições de propriedade intelectual do software.

5.1 Ambiente de Desenvolvimento e Ferramentas Utilizadas

A seleção estratégica de um ambiente de desenvolvimento produtivo e de ferramentas tecnologicamente adequadas revela-se crucial para otimizar a eficiência e assegurar a qualidade ao longo do ciclo de vida do software. Para a construção do Prontuário Familiar, foram adotadas ferramentas consolidadas no mercado, oferecendo um suporte robusto que perpassou as diversas fases do projeto, desde a codificação até o controle de versão e gerenciamento de build.

Para o desenvolvimento do back-end, a escolha recaiu sobre a Integrated Development Environment (IDE) IntelliJ IDEA Community Edition. Esta seleção foi

fundamentada em suas funcionalidades avançadas de inspeção de código, refatoração estruturada, depuração eficiente e sua nativa integração com o ecossistema Spring e ferramentas de automação de build, como o Maven. Por sua vez, no desenvolvimento do front-end, implementado em React com TypeScript, utilizou-se o editor de código Visual Studio Code (VS Code), reconhecido por sua leveza, extensibilidade e suporte robusto as tecnologias web contemporâneas, o que inclui a depuração de JavaScript/TypeScript e a integração direta com o Git.

A gestão do controle de versão do código-fonte foi realizada por meio do sistema distribuído Git, em conjunto com a plataforma de hospedagem de repositórios GitHub. Essa combinação viabilizou o monitoramento detalhado da evolução do código, fomentou a aplicação de boas práticas de desenvolvimento colaborativo (mesmo em um contexto de projeto individual), permitiu o gerenciamento organizado de branches para o desenvolvimento de funcionalidades e garantiu a rastreabilidade e a integridade das alterações efetuadas no projeto Prontuário Familiar, através do histórico de commits.

No que tange ao gerenciamento de dependências e à orquestração do ciclo de vida da aplicação back-end Spring Boot, foi empregado o Apache Maven. O Maven simplificou a administração das bibliotecas externas, a compilação do código-fonte, a execução dos testes automatizados e o empacotamento da aplicação, assegurando, assim, um processo de build padronizado e plenamente reprodutível.

5.2 Implementação do Back-end

A implementação do back-end do Prontuário familiar foi realizada em Java 17 com o *framework* Spring Boot e, GraphQL como linguagem de consulta e especificação de API conectando o back-end com o front-end, seguindo a arquitetura e os padrões de projeto definidos anteriormente. A seguir, detalha-se a implementação das principais camadas e componentes do back-end e da API.

5.2.1 Schemas (Camada de API)

Conforme conceituado na Fundamentação Teórica, o *Schema* GraphQL é o contrato que define o formato de todos os dados e operações disponíveis na API. É ele quem determina exatamente o que os clientes podem consultar e o que podem modificar. Para consultas o GraphQL disponibiliza um tipo raiz chamado *Query*, e para modificações um tipo raiz chamado *Mutation*.

Visto que o Prontuário Familiar foi implementado como um módulo que se integra ao sistema consolidado e-SUS APS, não foi necessária a criação de *Schemas* GraphQL totalmente novos. Em vez disso, optou-se pela reutilização de *Schemas* existentes na arquitetura, o que otimiza a consistência dos dados e a compatibilidade.

Os seguintes Schemas existentes foram mobilizados para a implementação:

- *Cidadao.graphqls*: É um dos maiores *schemas* do e-SUS APS, e é responsável por conter os *types*, *inputs*, *enums*, *Querys* e *Mutations* referentes ao contexto do cidadão.
- *Territorio.graphqls*: Este *schema* é responsável por conter os *types*, *inputs*, *enums*, *Querys* e *Mutations* referentes ao contexto do território, englobando informações relacionadas a imóvel e visitas domiciliares.
- *CidadaoNucleoFamiliar.graphqls*: Este *schema* possui apenas o *type* *CidadaoNucleoFamiliar*, responsável por definir os atributos referentes ao contexto do cidadão ao nível de família.
- *AnimaisImove.graphqls*: Este *schema* possui o *type* *AnimaisImovel* e o *input* *AnimaisImovelInput*. O tipo *AnimaisImovel* é empregado na recuperação de informações sobre os animais presentes no imóvel (operação de *Query*), enquanto o *input* é utilizado para a persistência e modificação desses dados (operação de *Mutation*).
- *CondicoesMoradia.graphqls*: Este *schema* possui o *type* *CondicoesMoradia* e o *input* *CondicoesMoradiaInput*. O tipo *CondicoesMoradia* é empregado na recuperação de informações sobre as condições de moradia do imóvel onde a família está presente (operação de *Query*), enquanto o *input* é utilizado na

persistência e modificação desses dados (operação de *Mutation*). Além disso, este *schema* possui uma variedade de *enums* que representam os valores possíveis para cada tipo de condição de moradia.

5.2.2 *Query resolvers* e *Field Resolvers* (Camada de API)

Conforme conceituado na Fundamentação Teórica, os endpoints do GraphQL atuam como a interface de entrada das requisições externas no e-SUS APS. No caso do back-end do Prontuário familiar, as requisições foram feitas exclusivamente através dos *Query Resolvers* e *Field Resolvers*, responsáveis por encaminhar as requisições para os serviços na camada de negócio.

Vale pontuar que devido à proposta do Prontuário familiar, não houve a necessidade de fazer a utilização de *types* do tipo *Mutation*. Os *query resolvers* utilizados no desenvolvimento do Prontuário familiar estão listados abaixo:

- *CidadaoQueryResolver*: As operações de *Query* definidas no *schema* *Cidadao.graphqls* são implementadas por este *Query Resolver*. Para o escopo do Prontuário Familiar, apenas os dois endpoints de consulta a seguir foram utilizados:
 - *cidadao*: Serve como o ponto de entrada do módulo do Prontuário familiar, aceitando o identificador do cidadão (id) como argumento. Sua função é buscar e retornar o objeto que representa o cidadão. É a partir desse objeto que os *field resolvers* atuam para buscar informações importantes para o módulo, como, por exemplo, as famílias. Quanto a lógica implementada nesse endpoint, é realizado apenas o controle de acesso e chamada para o método *getCidadao* do *CidadaoService*.
 - *cidadaos*: Foi projetado para atuar como ponto de acesso para recuperar as informações sócio-econômicas dos cidadãos presentes na família do cidadão do contexto atual. Este resolver recebe um objeto de entrada, contendo os identificadores dos familiares

(idCidadaoList) e retorna uma lista que incluem alguns dados dos familiares. Sua estrutura retorna a data de nascimento, se o cidadão é beneficiário do Bolsa Família e se está em Risco de Insegurança Alimentar. Quanto a lógica implementada nesse endpoint, é realizado apenas o controle de acesso, chamada para o método `getPage` do `CidadaoService` transformando o seu retorno em um objeto paginado.

- **TerritorioQueryResolver:** As operações de *Query* definidas no *schema* `Territorio.graphqls` são implementadas por este *Query Resolver*. Para o escopo do Prontuário Familiar, apenas o endpoint de consulta a seguir foi utilizado:
 - **visitas:** Este endpoint foi projetado para atuar como o ponto de acesso para a recuperação do histórico de visitas domiciliares feitas ao imóvel e seus moradores. Este resolver recebe um objeto de entrada, contendo obrigatoriamente o identificador do imóvel (`id`) e parâmetros de paginação e, opcionalmente, o desfecho da visita domiciliar. Ele retorna uma lista que inclui os dados de registro da visita. Sua estrutura retorna a data da visita, detalhes que motivaram a visita em si, o turno de realização e, de forma essencial, os dados completos da equipe e do profissional que a realizou. Além disso, o *resolver* fornece os dados de identificação do cidadão associado, garantindo um rastreamento completo das ações de saúde no domicílio. Quanto lógica implementada nesse endpoint, é realizado apenas o controle de acesso, chamada para o método `getVisitas` do `TerritorioService` transformando o seu retorno em um objeto paginado.

Os *field resolvers* utilizados no desenvolvimento do Prontuário familiar estão listados abaixo:

- **familias:** Implementado no tipo `Cidadao`, o *Field Resolver* `familias` é encarregado de resolver a totalidade dos dados relacionados à unidade familiar. Isso inclui os membros, o número do prontuário familiar, as informações do imóvel de residência, o profissional e a equipe de saúde responsáveis, entre outros dados correlatos.

- `cidadaosTerritorioBasico`: Associado ao tipo `Familia`, o *Field Resolver* `cidadaosTerritorioBasico` atua na recuperação dos dados essenciais dos membros da família vinculados ao território. Ele carrega atributos como nome, nome social, sexo, data de nascimento e o grau de parentesco, dentre outras informações relevantes.
- `cidadaoNucleoFamiliar`: Este *Field Resolver* foi implementado nos tipos `Cidadao` e `CidadaoTerritorioBasico`. No tipo `Cidadao`, ele é responsável por resolver o atributo `mudouSe`, que indica a situação de residência do indivíduo no território. Já no tipo `CidadaoTerritorioBasico`, o resolver busca o atributo `grauParentescoResponsavelFamiliar`, definindo o vínculo de parentesco do familiar com o responsável pela família.
- `imovel`: Implementado no tipo `Familia`, o *Field Resolver* `imovel` tem a finalidade de compor e resolver os dados estruturados relativos à residência. Este resolver é fundamental para a recuperação das informações cadastrais do imóvel, englobando detalhes como tipo de endereço, cep, logradouro, coordenadas geográficas (latitude, longitude), dados sociodemográficos, além de resolver tipos aninhados como tipo de logradouro, município, e outras informações correlatas. Adicionalmente, ele incorpora fragmentos essenciais como:
 - `CondicoesMoradiaFragment`: responsável por representar as condições de moradia do domicílio, por exemplo, tipo de domicílio, tipo de tratamento de água, se a residência possui energia elétrica, número de cômodos, entre outros.
 - `AnimaisDomicilioFragment`: responsável por representar as informações referentes aos animais presentes na residência, por exemplo, espécies de animais e quantidades
- `beneficiarioBolsaFamilia`: Associado ao tipo `Cidadao`, este *Field Resolver* atua na recuperação dos dados que identificam se os membros da família possuem vínculo com o programa Bolsa Família.
- `informacoesSocioEconomicasDW`: Associado ao tipo `Cidadao`, este *Field Resolver* atua na recuperação dos dados que identificam se os membros da família possuem algum Risco de Insegurança Alimentar.

5.2.2 Data Loaders (Camada de Acesso a Dados)

Conforme estabelecido na Fundamentação Teórica, o e-SUS APS emprega os *DataLoaders*, utilitários essenciais de otimização no ecossistema GraphQL, com o propósito de mitigar o problema de N+1 *queries*. Esses *loaders* se posicionam como uma camada de abstração entre os Field Resolvers e a Camada de Serviço (*Services*). É importante destacar que, seguindo o padrão de desenvolvimento do e-SUS APS, a implementação dos *loaders* é estritamente técnica, não incorporando regras de negócio.

Dada a sua principal funcionalidade de *batching*, os *loaders* são construídos com uma função central denominada *execute*, que aceita como parâmetro uma coleção (*Set*) de chaves (*Long*). Por convenção, essa função *execute* deve retornar um objeto do tipo *Map*, onde a chave corresponde ao identificador (*id*) recebido e o valor é o objeto de domínio solicitado pelo *Field Resolver* de origem. Logo, os *loaders* tem a função de invocar o serviço responsável pelo domínio trabalhado, e no máximo, realizar a transformação de objetos recebidos pelo service no *Map* solicitado pelo retorno.

A seguir, listamos os principais *DataLoaders* utilizados no desenvolvimento do módulo Prontuário Familiar:

- *InformacoesSocioEconomicasDWByCidadaoldLoader*: Usado pelo *field resolver* *informacoesSocioEconomicasDW*, ele invoca o método *getInformacoesSocioEconomicasDW* do *CidadaoService*.
- *BeneficiarioBolsaFamiliaByCidadaoldLoader*: Usado pelo *field resolver* *beneficiarioBolsaFamilia*, ele invoca o método *beneficiarioBolsaFamiliaByCidadaolds* do *CidadaoService*.
- *CidadaoNucleoFamiliarMaisRecenteByCidadaoldLoader*: Usado pelo *field resolver* *cidadaoNucleoFamiliar*, ele invoca o método *loadCidadaoNucleoFamiliarMaisRecente* do *CidadaoFciService*, e realiza o mapeamento entre o seu retorno e os identificadores dos cidadãos recebidos por parâmetro na função *execute*.

- **FamiliasByCidadaoLoader**: Usado pelo *field resolver* **familias**, ele invoca o método **getByCidadaoIds** do **FamiliaService**, e realiza o mapeamento entre o seu retorno e os identificadores dos cidadãos recebidos por parâmetro no método **execute**.
- **CidadaosTerritorioBasicoValidosByFamiliaLoader**: Usado pelo *field resolver* **cidadaosTerritorioBasico**, ele invoca o método **executeTerritorioBasico** do **CidadaoService**.
- **ImovelByIdLoader**: Usado pelo *field resolver* **imovel**, ele invoca o método **getCdsDomicilios** do **TerritorioService**, e realiza o mapeamento entre o seu retorno e os identificadores das famílias recebidas por parâmetro no método **execute**.
- **AnimaisByDomicilioLoader**: Usado pelo *field resolver* **animaisImovel**, ele invoca o método **getAnimaisByDomicilio** do **TerritorioService**.
- **FatoCidadaoPecByIdLoader**: Usado pelo *field resolver* **cidadaosPec**, ele invoca o método **getFatCidadaoPecByIds** do **FatoCidadaoPecService**.

5.2.3 Services (Camada de Negócio)

A Camada de Serviço encapsula a lógica de negócios do Prontuário Familiar, sendo responsável por orquestrar as operações de *back-end*. Ela interage com os Repositórios (responsáveis pela comunicação com o banco de dados) e executa as tarefas delegadas pelos *resolvers* da camada de API.

- **CidadaoService**: É o componente de serviço que encapsula a totalidade da lógica de negócios relativa à entidade Cidadão na arquitetura do e-SUS APS. Suas responsabilidades incluem a persistência, inativação, processamentos especializados e a recuperação de dados do cidadão mediante diferentes critérios de busca. Para o desenvolvimento específico do módulo Prontuário Familiar, foram mobilizados os seguintes métodos principais:
 - **getCidadao**: Chama o repositório responsável por executar a busca primária de uma única instância do Cidadão, utilizando o identificador do cidadão como parâmetro, e retorna o objeto completo.

- `executeTerritorioBasico`: Chama o repositório responsável por executar a busca de várias instâncias do objeto `CidadaoTerritorioBasico`, utilizando uma lista de identificadores de cidadãos como parâmetro, e retornando uma lista dos objetos. Este foi utilizado para buscar as informações dos familiares do cidadão do contexto.
- `getPage`: Chama o repositório responsável por executar uma busca que retorna uma lista paginada de objetos do tipo `Cidadao`. É através desse método que os familiares são carregados.
- `getInformacoesSocioEconomicasDW`: Realiza a recuperação de dados socioeconômicos do cidadão, acessando uma fonte de dados consolidada.
- `beneficiarioBolsaFamiliaByCidadaoIds`: Executa a consulta de status do cidadão para determinar se ele é beneficiário do programa Bolsa Família, utilizando uma coleção de identificadores como argumento.
- `FatoCidadaoPecService`: É o componente de serviço que encapsula a totalidade da lógica de negócios relativa à entidade `FatoCidadaoPec` na arquitetura do e-SUS APS. Para o desenvolvimento específico do módulo Prontuário Familiar, foi mobilizado apenas o seguinte método:
 - `getFatCidadaoPecByIds`: Responsável exclusivamente por buscar objetos do tipo `CidadaoFatoPec`, que representam o contexto do cidadão em uma visita domiciliar.
- `TerritorioService`: É o componente de serviço que encapsula a totalidade da lógica de negócios relativa às entidades pertencentes ao contexto do território, por exemplo, `Imovel` e `FatoVisitaDomiciliar`, que representam os imóveis e as visitas domiciliares na arquitetura do e-SUS APS. Para o desenvolvimento específico do módulo Prontuário familiar, foram mobilizados os seguintes métodos:
 - `getVisitas`: Chama o método `execute` do repositório `FatoVisitaDomiciliarPagingQuery`, responsável por executar uma operação de busca que retorna uma lista paginada de visitas domiciliares.
 - `getCdsDomicilios`: Chama o método `execute` do repositório `CdsDomicilioPagingQuery`, responsável por executar uma operação de busca que retorna uma lista paginada de imóveis.

- `getAnimaisByDomicilio`: Chama o método `execute` do repositório `AnimaisByDomicilioQuery`, responsável por executar uma operação de busca que retorna um mapa que possui como chave o identificador do imóvel e, como valor, uma lista de objetos contendo informações sobre os animais presentes no imóvel.
- **FamiliaService**: É o componente de serviço que encapsula a totalidade da lógica de negócios relativa a entidades pertencentes ao contexto da família, por exemplo, `Familia`, que representa as famílias na arquitetura do e-SUS APS. Para o desenvolvimento específico do módulo `Prontuário familiar`, foi mobilizado apenas o seguinte método:
 - `getByCidadaoIds`: Chama o método `execute` do repositório `FamiliasByCidadaoQuery`, responsável por executar uma operação de busca que retorna um mapa que possui como chave o identificador do cidadão e, como valor, uma lista de objetos contendo informações sobre as famílias relacionadas ao cidadão.
- **CidadaoFciService**: É o componente de serviço que encapsula a totalidade da lógica de negócios relativa a entidades pertencentes ao contexto da ficha de cadastro individual, que representa o cidadão no modelo de negócio do CDS. Na arquitetura do e-SUS APS, a entidade utilizada foi a `CidadaoNucleoFamiliar`. Para o desenvolvimento específico do módulo `Prontuário familiar`, foi mobilizado apenas o seguinte método:
 - `loadCidadaoNucleoFamiliarMaisRecente`: Chama o método `execute` do repositório `CidadaoNucleoFamiliarMaisRecenteByCidadaoQuery`, responsável por executar uma operação de busca que retorna uma lista de objetos contendo informações sobre o núcleo familiar no qual o cidadão do contexto está inserido.

5.2.3 Repositórios (Camada de Acesso a Dados)

A camada de acesso a dados foi implementada utilizando Spring Data JPA, que simplifica a interação com os banco de dados Postgres e Oracle mediante ao

uso das interfaces de repositório. Serão detalhadas a seguir as classes que implementam o Padrão Repository na arquitetura do sistema. Anotadas com `@Repository` do Spring, essas classes definem o contrato de acesso aos dados, isolando a lógica de negócio da Camada de Serviço.

- `CidadaosQuery`: Implementa regras de negócio referentes a operações de buscas pelos cidadãos. Também é disponibilizado diversos tipos de filtragem dos dados. Para o Prontuário Familiar foi usado apenas o filtro por lista de identificadores de cidadão.
- `CidadaoByIdQuery`: Executa uma busca simples, carregando a entidade `Cidadao` pelo identificador recebido por parâmetro.
- `FatoVisitaDomiciliarPagingQuery`: Implementa regras de negócio referentes a operações de busca por visitas domiciliares. Tem como parâmetro obrigatório o identificador do imóvel e permite a filtragem pelo desfecho da visita domiciliar. Para o Prontuário Familiar foi buscado apenas visitas com desfecho Realizado.
- `FamiliasByCidadaoIdQuery`: Implementa regras de negócio referentes a operações de busca por famílias. Tem como parâmetro obrigatório uma lista de identificadores de cidadãos e, de forma opcional, um filtro relacionado ao status da família. Para o Prontuário familiar foi usado apenas o filtro obrigatório.
- `CidadaoNucleoFamiliarMaisRecenteByCidadaoIdQuery`: Implementa regras de negócio referentes a operações de busca por núcleos familiares. Tem como parâmetro obrigatório uma coleção de identificadores de cidadão e retorna uma lista de objetos da entidade `CidadaoNucleoFamiliar`.
- `AnimaisByDomicilioIdQuery`: Implementa regras de negócio referentes a operações de busca por animais presentes no domicílio. Tem como parâmetro obrigatório o identificador do imóvel e retorna uma lista de objetos da entidade `AnimaisDomicilio`.
- `CdsDomicilioPagingQuery`: Implementa regras de negócio referentes a operações de busca por imóveis. Tem dois parâmetros obrigatórios, o primeiro é o identificador do usuário autenticado e, já que o e-SUS APS não permite que o usuário busque por imóveis que não fazem

parte da sua localidade. O segundo, um objeto que permite o usuário filtrar a *query* por diversos parâmetros opcionais. Para o Prontuário Familiar foi utilizado apenas o primeiro parâmetro.

- *FatoCidadaoPecByldQuery*: Implementa regras de negócio referentes a operações de busca por cidadãos consolidados, utilizados nas visitas domiciliares. Recebe como parâmetro o identificador do cidadão e retorna um objeto da entidade *FatoCidadaoPec*.
- *InformacoesSocioEconomicasDWByCidadaoldQuery*: Implementa regras de negócio referentes a operações de busca pelas informações sócio-econômicas do cidadão, englobando os dados sobre segurança alimentar e participação no programa Bolsa Família. Recebe como parâmetro uma coleção de identificadores do cidadão e retorna um mapa onde a chave é o respectivo identificador do cidadão, e como valor um objeto do tipo *InformacoesSocioEconomicasDWDto*. Vale destacar que essa *query* busca em bases de dados consolidadas e leva em consideração apenas os dados dos últimos seis meses partindo da data atual.

5.2.4 Implementação da Segurança

5.2.4.1 Herança e Padrões de Segurança

A implementação do módulo Prontuário Familiar ocorreu dentro do core do sistema e-SUS APS, que já possui sua infraestrutura de segurança estabelecida. Dessa forma, não houve a necessidade de desenvolver ou configurar novas camadas de segurança para o módulo.

O Prontuário Familiar herda e reutiliza integralmente o sistema de segurança do framework Spring Security, padrão do e-SUS APS. Essa abordagem assegura que o novo módulo opere conforme as políticas de autenticação e autorização já vigentes no sistema.

5.2.4.2 Controle de Acesso por Recurso

Embora a autenticação e as configurações globais de segurança sejam herdadas, o controle de acesso no nível de recurso é garantido no Prontuário Familiar através da autorização por endpoint.

Isso significa que cada novo endpoint exposto pelos *Query* e *Mutation Resolvers*, necessários para o acesso aos dados do Prontuário Familiar, foi configurado para exigir um conjunto específico de permissões. Este controle de acesso por recurso garante que apenas usuários devidamente autenticados no sistema e com os recursos corretos, podem interagir com as operações do GraphQL.

Dessa forma, a implementação de segurança do módulo focou-se em aplicar, mapear e garantiu que as restrições de autorização existentes para os endpoints do *Schema GraphQL*.

5.2.5 Considerações

Como dito anteriormente, o e-SUS APS é um sistema consolidado, e no que tange o desenvolvimento do *back-end* do Prontuário Familiar, buscou-se seguir as melhores práticas de desenvolvimento com Spring Boot, com clara separação de responsabilidades entre as camadas, um robusto mecanismo de segurança e sempre respeitando os padrões de projetos de desenvolvimento exigidos pelo laboratório Bridge.

5.2.6 Testes Automatizados

A garantia da qualidade e a corretude do *software* são pilares inegociáveis na Engenharia de Software moderna. No contexto do desenvolvimento do Prontuário

Familiar, a adoção de testes automatizados foi uma prática imperativa, utilizada para a verificação sistemática do comportamento de todos os componentes do sistema, facilitando a execução segura de refatorações e permitindo a identificação ágil de regressões ao longo do ciclo de desenvolvimento.

5.2.6.1. Testes Automatizados no *Back-end*

A principal estratégia de verificação automatizada empregada no *back-end* consistiu na combinação de Testes de Unidade e Testes de Integração.

5.2.6.1.1. Testes de Unidade

O objetivo primordial dos Testes de Unidade é validar a funcionalidade de segmentos ínfimos e isolados do código — tipicamente métodos ou classes específicas — assegurando que cada unidade atue estritamente conforme o comportamento especificado. Para a escrita, estruturação e execução desses artefatos de teste, foram utilizadas as seguintes ferramentas especializadas:

- JUnit 5: Foi empregado como o *framework* de teste base para a plataforma Java. O JUnit 5 viabilizou a estruturação dos casos de teste, a execução de asserções e a gestão do ciclo de vida completo do processo de testes.
- Mockito: *Framework* de *mocking* utilizado em Java, fundamental para a criação de objetos simulados (*mocks*) das dependências de uma classe sob teste. O uso de *mocks* é essencial para promover o isolamento da unidade em avaliação e o controle do comportamento de suas dependências.

5.2.6.1.2. Testes de integração

Complementarmente, foram desenvolvidos Testes de Integração com o objetivo de verificar a comunicação correta entre diferentes componentes do *back-end*, como as camadas de Service e Repository, e, principalmente, a interação com a base de dados. Essa camada de testes garante que os componentes, quando combinados, operem sinergicamente, validando fluxos de trabalho que exigem a persistência e a recuperação de dados do sistema Prontuário Familiar.

5.3 Implementação do Front-end

Como dito anteriormente, o desenvolvimento do Prontuário Familiar se dá dentro de um sistema consolidado e com arquitetura definida em todos os níveis e camadas. O front-end do módulo Prontuário Familiar foi desenvolvido utilizando as tecnologias fundamentais React e TypeScript. Esta escolha arquitetural visou a criação de interfaces de usuário componentizadas, declarativas e reativas, utilizando a segurança de tipos oferecida pelo TypeScript para mitigar erros e aumentar a robustez do código.

Crucialmente, a camada de apresentação aderiu integralmente ao Bold Design System, garantindo a coerência visual e a familiaridade do usuário com os padrões de interface do e-SUS APS. Por fim, a gestão de dependências do projeto foi realizada através do gerenciador de pacotes Yarn.

5.3.1 Estrutura de Roteamento

Dado que o Prontuário Familiar opera como um módulo acessado via uma aba no contexto do prontuário do cidadão, a implementação inicial focou na

definição da estrutura de roteirização. O primeiro passo foi a criação do componente `ProntuarioFamiliarView` (inicialmente um *dummy component*). Em seguida, foram definidas as novas rotas do módulo utilizando funcionalidades da biblioteca `react-router-dom` e `react-router`, vinculadas aos componentes de cabeçalho do módulo do Cidadão e do módulo dos Atendimentos.

Uma etapa crucial dessa implementação foi a integração do componente interno do e-SUS APS, `CheckPermission`. Este componente é utilizado para aplicar o controle de acesso por recurso, verificando se o usuário possui as permissões necessárias para acessar a rota e a aba do Prontuário Familiar. Essa camada de verificação é essencial para prevenir o acesso indevido ao módulo por manipulação direta da URL.

5.3.2 Tela Principal: `ProntuarioFamiliarView`

O componente `ProntuarioFamiliarView` é a view principal do módulo, recebendo como propriedade essencial o identificador do cidadão, utilizado para contextualizar a busca.

A lógica de funcionamento é iniciada pelo *hook* `useProntuarioFamiliarQuery`, gerada pelo `Apollo-Client`, que busca os dados da família do cidadão, empregando a política de *fetch cache-first* para otimizar o desempenho, pois ela prioriza a leitura de dados a partir do cache local da aplicação. Caso o dado solicitado não esteja presente, ou seja, considerado obsoleto no cache, ele é então buscado na fonte remota (servidor). O componente utiliza o *hook* `useState`, do React, para gerenciar o estado da navegação local, que controla a alternância entre a visualização dos dados do imóvel e o histórico de visitas domiciliares.

A renderização do módulo é estritamente condicional e segue uma ordem de prioridade de exceção:

1. Cenários de Exceção e Loading: A view verifica duas variáveis que indicam se o cidadão possui família e se ele ainda reside no território onde UBS se

encontra. Se o cidadão não possuir famílias ou se o loading dos dados estiver em andamento, o componente SemFamiliaContainer é renderizado.

2. Renderização Padrão (Família Válida): Se o cidadão possui ao menos uma família, o componente itera sobre a lista para renderizar, em cada iteração, os seguintes elementos na interface:

- O componente FamiliaSection, responsável pelo cabeçalho e listagem de membros.
- O componente ProntuarioFamiliarButtonGroup, que gerencia o estado da view entre os dados do imóvel e as visitas.
- Renderização Alternada: Com base no estado do ProntuarioFamiliarButtonGroup, a view exibe condicionalmente a seção de dados do imóvel (utilizando VisualizarImovelHeaderContent e CondicoesMoradiaSection) ou a lista de visitas (utilizando ProntuarioFamiliarVisitasListSection).

Dessa forma, o ProntuarioFamiliarView consolida toda a lógica de decisão e a orquestração dos componentes filhos.

5.3.3 Componentes filhos

O detalhamento a seguir apresenta os componentes filhos do ProntuarioFamiliarView, organizados de acordo com sua sequência de renderização condicional.

SemFamiliaContainer

Este componente é o elemento de interface dedicado a gerenciar a visualização de exceção do módulo, sendo renderizado quando o ProntuarioFamiliarView detecta a ausência de um vínculo familiar válido. Ele recebe

como propriedades *flags booleanas* que indicam se o cidadão reside no território, possui família ou se a *query* está carregando, e isso é utilizado para determinar o *status* da exibição.

A lógica do componente, encapsulada em uma função que , prioriza os seguintes cenários de forma condicional, quando suas *flags* são verdadeiras:

- Loading State: Renderiza um componente interno do e-SUS APS chamado LoadingIndicator, ele é exibido para informar o usuário sobre o carregamento assíncrono dos dados.
- Mudança de Território: Exibe a mensagem de desvinculação territorial, conforme a Figura 13.
- Cidadão não possui família válida: O componente exibe as duas mensagens informativas conforme Figura 10.

O componente garante que o feedback visual seja coerente com o estado do registro do cidadão, mantendo o usuário informado sobre a impossibilidade de visualização completa do prontuário.

FamiliaSection

Este componente tem como função primária a estruturação e apresentação do núcleo familiar. O ProntuarioFamiliarView envia como propriedade uma coleção de objetos que representam família para dar início a orquestração dos dados e a lógica interna.

A lógica do componente é iniciada por uma função que processa o objeto da família para segregar o responsável familiar de seus dependentes. Essa segregação é fundamental para determinar a hierarquia de exibição na interface. O fluxo de renderização é o seguinte:

- **Cabeçalho da Família:** O componente `FamiliaSectionHeader` é o primeiro a ser renderizado. Ele recebe, de forma explícita, todos os dados sumarizados da família (nome do responsável, número do prontuário, renda, etc.) para compor a área superior da seção.
- **Vulnerabilidade:** O `FamiliaAcompanhamentoVulnerabilidadeSection` é renderizado logo em seguida. Ele recebe uma coleção de identificadores dos familiares para realizar a consulta condicional dos indicadores de vulnerabilidade social.
- **Listagem de Membros:** O componente renderiza o responsável e, subsequentemente, itera sobre a lista de dependentes. Cada membro é exibido em um `CidadaoFamiliaCard`.

O `FamiliaSection` atua, portanto, como um componente orquestrador, transformando os dados da API em uma visualização coesa e hierárquica do núcleo familiar.

ProntuarioFamiliarButtonGroup

O componente `ProntuarioFamiliarButtonGroup` atua como o principal controlador de visualização dentro do `ProntuarioFamiliarView`. Sua função é gerenciar a alternância (*toggle*) entre as duas grandes seções de dados secundários do prontuário: "Dados do Imóvel" e "Visitas Realizadas" (conforme ilustrado na Figura 5).

O estado do componente é controlado e manipulado pelo `ProntuarioFamiliarView`, este envia duas propriedades para o componente, uma *flag booleana* que indica o estado do agrupamento e uma função utilizada para manipular o estado do agrupamento (*hook useState*). Ao clicar em um dos botões, é invocado a função que manipula o grupo, alterando o estado global para 'DADOS_IMOVEL' ou 'VISITAS'.

A implementação usa os componentes `ButtonGroup` e `Button`, pertencentes ao `bold-ui`, framework desenvolvido em cima do *design system* Bold. A estilização é

definida de forma reativa através da propriedade `kind`. O botão que corresponde ao valor atual recebe o valor `'primary'`, o que resulta no seu destaque visual, indicando ao usuário qual seção está ativa.

Essa implementação garante que a alternância entre a visualização das informações do imóvel e do histórico de visitas seja coerente e controlada pelo Design System.

VisualizarImovelHeaderContent

O componente `VisualizarImovelHeaderContent` é um componente de visualização que recebe o objeto que representa o imóvel onde a família reside, e sua função principal é renderizar o cabeçalho da seção de “Dados do imóvel” (conforme a Figura 6).

O `VisualizarImovelHeaderContent` usa alguns componentes disponíveis no *framework* `bold-ui`, estes são:

- `HFlow`, `VFlow`: São componentes de layout que atuam como agrupadores e gerenciadores de fluxo para seus componentes filhos. O `HFlow` (Horizontal Flow) define o fluxo de renderização na direção horizontal (linha) e o `VFlow` (Vertical Flow) Define o fluxo de renderização na direção vertical (coluna). Esses componentes são cruciais para o alinhamento, espaçamento e a composição estrutural da interface do Prontuário Familiar.
- `Tag`: É um elemento de interface gráfica utilizado para a categorização, identificação ou para sinalizar metadados concisos e importantes sobre um objeto ou estado. No contexto do cabeçalho do imóvel, ele exibe o tipo do imóvel (ex: casa/domicílio, apartamento).
- `Text`: Componente de básico para renderização de elementos textuais.

Este componente executa uma lógica de renderização condicional, que é necessária devido à natureza opcional e hierárquica dos dados de endereço e contato do imóvel. Essencialmente, o componente funciona como um agrupador de

decisões sobre o que deve ou não ser exibido, utilizando os componentes citados acima, funções de formatação de texto e estilização.

CondicoesMoradiaSection

O componente `CondicoesMoradiaSection` recebe os objetos referentes as condições de moradia e animais, e é responsável pela visualização estruturada dos dados socio sanitários do imóvel, conforme ilustrado na Figura 8 e Figura 9.

A renderização é organizada em Cells do Grid e utiliza o componente `InfoLabel` do Bold Design System para exibir cada dado. Possui a seguinte lógica de tratamento de dados:

- Agrupamento: Campos relacionados, como os de Saneamento e Animais de Estimação, são agrupados em HFlow e Cell.
- Enums e padronização: O componente traduz os valores Enum brutos recebidos da API para labels legíveis em português, usando objetos *lookup* em estruturas de *Record*.
- Gestão de null state: Utiliza a propriedade “placeholder='Não informado'” do `InfoLabel` para tratar a ausência de dados, garantindo a consistência com o padrão de sinalização de *null value* do módulo.
- Ícones: As seções internas do componente são iniciadas com ícones SVG para otimizar a clareza visual.

ProntuarioFamiliarVisitasListSection

O componente é responsável pela visualização do histórico de visitas domiciliares (seção "Visitas realizadas"). Ele consulta dados utilizando o identificador do imóvel e aplica filtros de negócio essenciais, como restringir a busca apenas às

visitas que tiveram o desfecho de "Realizada" e limitar o retorno às três visitas mais recentes.

A lógica de renderização é simples: exibe um indicador de carregamento durante a busca; se houver resultados, ele itera sobre o conjunto de visitas para apresentar o `ProntuarioFamiliarVisitaDomiciliarCard` de cada visita. Caso não haja visitas concluídas, é exibida uma mensagem informativa “Ainda não existem visitas associadas a esta família” em itálico.

5.3.4 Outros componentes

Esta seção é dedicada à análise dos componentes auxiliares que, embora não sejam renderizados diretamente pelo `ProntuarioFamiliarView`, desempenham um papel instrumental na orquestração dos dados, na lógica de layout ou na apresentação visual das informações. Adicionalmente, serão detalhados os arquivos utilitários desenvolvidos, que encapsulam funções de transformação e formatação de dados essenciais para o *front-end* do Prontuário Familiar.

CidadaoFamiliaCard

O componente é renderizado pelo `FamiliaSection`, recebe o objeto `cidadao` e é estruturado para exibir as informações principais do indivíduo de forma hierárquica. Ele é o responsável por identificar o cidadão exibindo obrigatoriamente o nome, sexo, idade, data de nascimento e o número de identificação primário (CPF ou CNS), em `HFlows`. Quando o cidadão é o responsável familiar, ele utiliza uma `Tag` contendo o texto “Responsável familiar” para destacá-lo. Quando o cidadão é o dependente, ele exibe também a informação de Parentesco com o responsável, que possui o *placeholder* “Não informado”.

O componente `CidadaoFamiliaCard` possui uma propriedade opcional que permite a injeção de um componente de ação. É através desta propriedade que o `FamiliaSection` insere o `VisualizarCidadaoButton`. Este botão, representado por um ícone de lupa, tem a função específica de redirecionar o usuário diretamente para o prontuário individual do respectivo cidadão, facilitando a transição de contexto e o aprofundamento da análise clínica.

FamiliaAcompanhamentoVulnerabilidadeSection

O `FamiliaAcompanhamentoVulnerabilidadeSection` é renderizado pelo componente `FamiliaSection`. Sua responsabilidade é determinar se o núcleo familiar possui risco social e, em caso positivo, renderizar a view de alerta. Ele recebe os identificadores de todos os membros da família e realiza uma consulta otimizada para buscar os respectivos status de vulnerabilidade individual.

A lógica interna do componente é estruturada para determinar a exibição condicional do alerta. Após a conclusão do carregamento assíncrono dos dados, o componente processa o resultado para identificar a ocorrência de dois indicadores chave: se ao menos um membro está em risco de insegurança alimentar ou se ao menos um membro é beneficiário do programa social Bolsa família. Esta lógica é implementada com o uso de *hooks* essenciais do React. O *hook* `useState` é empregado para gerenciar o estado *booleano* que controla a renderização condicional da seção de Acompanhamento de vulnerabilidades. Complementarmente, o *hook* `useEffect` é utilizado para otimizar o ciclo de vida do componente, garantindo que o cálculo da necessidade de exibição do alerta seja executado apenas quando as dependências de dados (`loading` e os indicadores de risco) forem alteradas. Essa abordagem previne recálculos desnecessários e contribui para o desempenho geral do *front-end*.

O estado de exibição do componente é, então, reativamente atualizado. Somente quando a busca se encerra e pelo menos um dos indicadores de risco for confirmado, o componente `AcompanhamentoVulnerabilidadeCard` é renderizado.

Isso garante que a visualização final seja precisa, exibindo as labels e tooltips correspondentes ao risco detectado (conforme Figura 4 e Figura 5), e fornecendo a informação crítica para a equipe de saúde de maneira eficiente.

AcompanhamentoVulnerabilidadeCard

O componente AcompanhamentoVulnerabilidadeCard é um componente de apresentação que recebe uma flag *boolean* referente ao status do programa Bolsa família, a data da aplicação da TRIA e uma *string* que permite o uso de tooltips customizadas, traduzindo na visualização final, conforme detalhado na Figura 4.

A renderização interna utiliza a lógica de agregação vertical, com o componente VFlow, do bold-ui, e a renderização condicional para cada indicador:

- Beneficiário do Bolsa família: Esta seção é renderizada somente se a flag correspondente for verdadeira. Ela exibe a label em negrito, um ícone visual e o componente Infolcon, do e-SUS APS, para mostrar a *tooltip* informativa com a orientação operacional. No caso do Prontuário Familiar, foi uma *tooltip* customizada através da propriedade disponibilizada pelo componente.
- Risco de insegurança alimentar: Essa parte é renderizada somente se a data da aplicação da TRIA não for nula ou indefinida. Exibe a label em negrito, o ícone específico e a data de observação.

O componente utiliza classes CSS, tipicamente do Bold Design System, para garantir a hierarquia visual (negrito no título) e a consistência do design, isolando a lógica de formatação do restante da aplicação.

ProntuarioFamiliarVisitaDomiciliarCard

O componente `ProntuarioFamiliarVisitaDomiciliarCard` é o responsável por renderizar a visualização individual e detalhada de cada visita domiciliar no histórico da família, sendo chamado pelo `ProntuarioFamiliarVisitasListSection`. Este componente recebe o objeto que representa a visita domiciliar e atua como um cartão de informações. Sua estrutura e lógica incluem:

- **Cabeçalho Visual:** Renderiza um cabeçalho que utiliza dados de uma estrutura do tipo *Record* para exibir o título e o ícone associados ao tipo de visita. A data, o turno e uma borda colorida personalizada são exibidos, garantindo o destaque visual.
- **Motivo da visita e Detalhamento:** O motivo da visita é um dado composto, representado por quatro arrays de string diferentes. Para garantir a exibição coesa na interface, uma função de formatação é invocada para consolidar esses múltiplos objetos em uma única string legível. O componente utiliza o *hook* `useMemo` do React para a memoização do resultado dessa função de formatação. O uso do `useMemo` é uma estratégia de otimização de desempenho que assegura que o cálculo e a concatenação da string de motivo da visita não sejam reexecutados a cada renderização do componente, prevenindo processamento desnecessário e mantendo o desempenho da interface.
- **Renderização Condicional:** Exibe o campo "Cidadão visitado" e seus identificadores (CPF/CNS) somente se o tipo da visita for "Individual" ou "Indefinida", seguindo a regra de negócio definida para esse campo.
- **Informações da profissional:** Conclui o *card* exibindo os dados de quem realizou a visita, incluindo nome do profissional, CBO e a equipe/INE responsável.

Para a apresentação visual, a seção utiliza componentes base do Bold Design System:

- Para arquitetar o fluxo dos componentes de layout foi utilizado `HFlow` e `VFlow`
- O título principal da visita é renderizado com o componente `Heading`.
- Os dados detalhados da visita e o texto são exibidos pelo componente `Text`.

O ícone da visita é apresentado pela tag nativa `<img>`, garantindo a visualização da simbologia representativa do tipo da visita.

Dessa forma, o Card transforma os dados brutos da visita em uma visualização clara, formatada e hierarquizada, otimizando a leitura e a compreensão do histórico pelo profissional de saúde.

5.3.5 Utilitários

Para garantir a organização estrutural e promover o reuso de código no *front-end* do Prontuário Familiar, foram criados dois arquivos utilitários essenciais:

- Arquivo de Formatação (*Formatter*): Contém funções dedicadas à transformação de dados para exibição. A principal função desenvolvida realiza a concatenação de múltiplos *arrays* de *string* (representando os diversos motivos da visita domiciliar) em uma única *string* coesa, pronta para ser apresentada na interface do usuário. Esta centralização garante a uniformidade na exibição de dados compostos.
- Arquivo de Modelo (*Model*): Atua como um repositório de padronização, englobando elementos cruciais para a comunicação e tipagem do *front-end*. Ele contém constantes, Records, Enums e Interfaces que definem as estruturas de dados e regras de negócio utilizadas pelos componentes.

A criação destes arquivos reforça o princípio de separação de preocupações, isolando a lógica de apresentação e a padronização da tipagem da lógica dos componentes de interface.

5.3.6 Testes Automatizados no *Front-end*

Visando assegurar a corretude e a integridade da *interface* e da lógica de apresentação do usuário (desenvolvida em React/TypeScript), foi adotado o *framework* Jest.

- Jest: Utilizado como a principal ferramenta de teste para o *front-end*. O Jest viabilizou a escrita de Testes de Unidade para as principais funções utilitárias em TypeScript. Sua capacidade de isolamento e a API amigável foram cruciais para validar o comportamento dos componentes React, garantindo que a lógica de renderização, que foi muitas vezes gerado por essas funções utilitárias, atuassem conforme o esperado.

6 APRESENTAÇÃO E ANÁLISE DO PRONTUÁRIO FAMILIAR DESENVOLVIDO

Este capítulo se dedica à apresentação do Prontuário Familiar em sua forma final, detalhando as funcionalidades implementadas e a interface com o usuário. Adicionalmente, será realizada uma análise crítica sobre como os requisitos funcionais e não funcionais, definidos na fase de projeto, foram atendidos pela solução desenvolvida para o contexto do Prontuário Familiar.

6.1 Apresentação das Funcionalidades Implementadas

A interface do Prontuário Familiar foi desenvolvida com o objetivo primordial de equilibrar a simplicidade de uso com a riqueza e completude das informações clínicas e sociais.

Para garantir a coerência e a baixa curva de aprendizado, o *front-end* aderiu integralmente ao Bold Design System do e-SUS APS. Essa decisão assegura que o novo módulo utilize o mesmo conjunto de componentes visuais, padrões de interação e fluxo de trabalho já familiares aos usuários do sistema. A experiência do usuário foi priorizada para garantir que a navegação fosse intuitiva e rápida para os profissionais de saúde do e-SUS APS. Essa abordagem permite que o módulo sirva como uma fonte de informação abrangente, mas sem adicionar complexidade operacional à rotina do usuário.

A apresentação das telas do módulo será segmentada em subseções que acompanham a estrutura lógica e hierárquica do Prontuário Familiar. Essa organização facilita a compreensão da progressão da informação, iniciando pela composição da família, avançando para as informações domiciliares, detalhando o histórico de visitas domiciliares e, por fim, apresentando o Prontuário Familiar completo e seus respectivos cenários de exceção.

6.1.1 Acesso ao Prontuário Familiar

O acesso ao módulo Prontuário Familiar se realiza por meio do cabeçalho de contexto do cidadão. Este elemento de interface é um padrão de navegação presente em diversos módulos do e-SUS APS, o que garante a coerência da interface e a familiaridade do usuário. Na Figura 1, o cabeçalho é acessado a partir do Módulo do Cidadão e evidencia a nova aba "Prontuário Familiar" em destaque, servindo como o ponto de entrada principal.

Figura 1 - Aba do Prontuário Familiar no cabeçalho do módulo do Cidadão



Fonte: O autor (2025)

6.1.2 Composição da família

A Figura 2 apresenta o componente responsável por exibir as informações da composição familiar, ele é introduzido por um cabeçalho de identificação. Este cabeçalho utiliza o nome do responsável familiar como título principal, complementado, de forma opcional, pelo número do prontuário familiar.

O cabeçalho provê um resumo executivo das principais informações contextuais da família, conforme detalhado a seguir:

- **Equipe Responsável:** Apresenta o código INE (Identificador Nacional de Equipe) da Equipe de Saúde da Família responsável pelo acompanhamento longitudinal do núcleo familiar no território.

- **Renda Familiar:** Indica o valor total da renda familiar mensal, expresso em quantidade de salários mínimos, sendo um dado crucial para a avaliação socioeconômica.
- **Nº de Membros Declarados da Família:** Exibe o número total de membros que compõem o núcleo familiar, conforme declarado no cadastro do imóvel no e-SUS APS.
- **Reside Desde:** Informa o mês e o ano em que a família estabeleceu residência no imóvel atual, fornecendo um dado temporal relevante para a estabilidade territorial.

Na continuação da Figura 2, o cabeçalho é seguido por uma listagem detalhada dos membros que compõem o núcleo familiar, incluindo o cidadão em contexto. Cada indivíduo é apresentado em um *card* com informações cruciais para a identificação e o acompanhamento: nome completo, sexo, idade, data de nascimento e o identificador de cadastro (CPF ou CNS).

O responsável familiar é distinguido por uma tag visual destacada, garantindo sua rápida identificação na lista. Além disso, o vínculo de parentesco com o responsável é exibido exclusivamente para os membros que não são o próprio responsável.

A funcionalidade de navegação é implementada por um botão de *call-to-action* (identificado por um ícone de lupa) posicionado ao final de cada *card*. Ao ser acionado, este botão redireciona o usuário para o prontuário individual do respectivo cidadão, facilitando a transição de contexto e o acesso a informações mais detalhadas.

Figura 2 - Componente da composição familiar - todos os dados preenchidos

Família de Adão Gonçalves dos Santos Nº prontuário NumeroDoProntuarioFamiliar			
Equipe responsável ESF 1	Renda familiar Acima de 4 salários mínimos	Nº de membros declarados da família 2	Reside desde 12/2024
Responsável familiar Adão Gonçalves dos Santos Sexo Masculino 87 anos e 7 meses Nasceu em 20/03/1938 CNS 209178248000007			
Adbel Perao Sexo Feminino 35 anos e 5 meses Nasceu em 11/06/1990 CNS 200928173480006 Parentesco com o responsável Cônjuge / Companheiro(a)			

A Figura 3 ilustra a composição da família em um cenário de preenchimento mínimo, onde apenas os campos de cadastro obrigatórios da família foram efetivamente registrados. Em função da ausência de valores, os campos opcionais do cabeçalho, como, por exemplo, “Renda familiar” e “Reside desde”, são exibidos com o valor padrão “Não informado”, formatado em itálico. Esta formatação específica é um padrão de interface implementado nos cabeçalhos de vários módulos do e-SUS APS para sinalizar dados incompletos ou não inseridos.

No contexto da listagem de membros, o campo “Parentesco com o responsável” também assume o valor padrão "Não informado" quando ausente. Contudo, neste caso, a exibição mantém o padrão de escrita das demais informações do *card*, aderindo ao estilo definido para o conteúdo do núcleo familiar.

Figura 3 - Componente da composição familiar - dados mínimos

Família de Adão Gonçalves dos Santos			
Equipe responsável ESF 1	Renda familiar <i>Não informado</i>	Nº de membros declarados da família <i>Não informado</i>	Reside desde <i>Não informado</i>
Responsável familiar Adão Gonçalves dos Santos Sexo Masculino 87 anos e 7 meses Nasceu em 20/03/1938 CNS 209178248000007			
Adbel Perao Sexo Feminino 35 anos e 5 meses Nasceu em 11/06/1990 CNS 200928173480006 Parentesco com o responsável <i>Não informado</i>			

Fonte: O autor (2025)

6.1.3 Acompanhamento de vulnerabilidade

A Figura 4 detalha o componente de Acompanhamento de vulnerabilidade, uma seção crítica do Prontuário Familiar que consolida informações sobre o risco social da família. Este componente informa a presença de dois indicadores cruciais de vulnerabilidade: o Risco de insegurança alimentar e o status de Beneficiário do programa Bolsa Família.

Para garantir o destaque imediato da informação crítica, o componente de Acompanhamento de Vulnerabilidade foi posicionado estrategicamente entre o cabeçalho da família e a listagem de membros do núcleo.

Figura 4 - Acompanhamento de vulnerabilidade



Fonte: O autor (2025)

Este componente de Acompanhamento de vulnerabilidade opera sob lógica condicional e é exibido na interface somente se, no mínimo, um cidadão vinculado ao núcleo familiar possuir algum dos dois indicadores.

O status de Beneficiário do programa Bolsa Família é sinalizado por uma label destacada em negrito, seguida de um ícone de informação. Este ícone apresenta uma *tooltip* com a mensagem: "Um cidadão desta família consta como beneficiário do Bolsa Família na vigência mais recente importada. Verifique o acompanhamento das condicionalidades de saúde." A *tooltip* cumpre a função de orientação operacional para o profissional.

De modo análogo, para as famílias que apresentam o indicador de Risco de insegurança alimentar, é exibida uma label em negrito, complementada pela label "Observado em" e pela data de aplicação da Triagem de Risco de Insegurança Alimentar (TRIA), extraída da ficha de cadastro individual.

6.1.4 Botões Dados do imóvel e Visitas realizadas

Imediatamente após a seção de composição da família, é exibido um agrupamento de botões de alternância (*tab switcher*) contendo as opções "Dados do imóvel" e "Visitas realizadas" (conforme ilustrado na Figura 5). A função primordial deste grupo é comutar entre duas visualizações distintas de dados: a visualização das informações cadastrais e estruturais do imóvel e a visualização do histórico de visitas domiciliares efetuadas ao domicílio e seus residentes.

Figura 5 - *Tab switcher*



Fonte: O autor (2025)

6.1.5 Dados do imóvel - Informações de localização e contato

O cabeçalho da seção "Dados do Imóvel" (Figura 6) é iniciado por um componente de identificação e localização que exibe os atributos cadastrais essenciais da residência. Os campos apresentados são agrupados para fornecer uma visão completa do endereço e da acessibilidade do domicílio:

- **Dados de Endereço:** Incluem o tipo do logradouro, logradouro, número, tipo do domicílio, complemento do logradouro, bairro, cidade, UF e CEP.
- **Informações de Rastreamento:** Especificam a microárea de abrangência e os pontos de contato (telefone de contato e telefone residencial).
- **Localização:** Finalizam com as coordenadas de geolocalização do imóvel, essenciais para o planejamento territorial das equipes de saúde.

Figura 6 - Cabeçalho do imóvel - todos os dados preenchidos



Rua Lauro Linhares, 1600 Apartamento

Sala 901 - Torre Flora - Trindade - Florianópolis/SC - 88036-002

Microárea 01 Telefone de contato (48) 98899-8899 Telefone residencial (48) 3322-3322 Geolocalização —

Fonte: O autor (2025)

Conforme ilustrado na Figura 7, o tratamento de dados opcionais no cabeçalho das informações do imóvel segue duas lógicas distintas para otimizar a interface: campos como complemento do logradouro são omitidos quando vazios para reduzir a poluição visual, enquanto campos críticos como telefones e geolocalização são sinalizados explicitamente pela presença de um travessão (—), mantendo a coerência com a padronização de ausência de informação do sistema.

Figura 7 - Cabeçalho do imóvel - dados mínimos



Rua Lauro Linhares, 1 Casa/domicílio

Trindade - Florianópolis/SC - 88036-002

Microárea 01 Telefone de contato — Telefone residencial — Geolocalização —

Fonte: O autor (2025)

6.1.6 Dados do imóvel - Condições de moradia

Seguido da seção de informações de localização e contato, a Figura 8 ilustra a seção "Condições de Moradia". Este bloco é essencial por consolidar diversas informações sobre o saneamento e a estrutura física do imóvel.

Figura 8 - Condições de moradia do imóvel - todos os dados preenchidos

Situação de moradia/Posse de terra Alugado	Localização Urbana	Número de cômodos 6
 Material predominante Alvenaria com revestimento	 Tipo de acesso Pavimento	
 Abastecimento de água Rede encanada até o domicílio - Sistema de abastecimento público (municipal ou estadual)	 Água para consumo Clorada	
 Escoamento do banheiro Rede coletora de esgoto ou pluvial	 Destino do lixo Coletado	
 Possui eletricidade Sim	 Possui animais de estimação Quantidade 4 Tipos Gato Cachorro Pássaro	

Fonte: O autor (2025)

Os seguintes dados são exibidos por esta seção:

- **Situação de moradia/Posse de terra:** Define a natureza jurídica e de ocupação do imóvel (alugado, financiado, próprio, cedido, etc.), fornecendo contexto sobre a estabilidade habitacional da família.
- **Localização:** Indica a classificação da área de residência (urbana, rural, periurbana, etc.), dado crucial para o planejamento e a logística de atendimento da equipe de saúde.
- **Número de cômodos:** Determina o dimensionamento da residência em relação ao número de membros familiares, sendo um indicador de possível adensamento populacional.
- **Material predominante:** Classifica a qualidade da estrutura física do imóvel, influenciando nas condições de salubridade.
- **Tipo de acesso ao domicílio:** Descreve a via de acesso à residência, informação relevante para o planejamento das visitas domiciliares.
- **Abastecimento de água:** Detalha a principal forma de fornecimento de água para o domicílio.
- **Água para consumo:** Especifica a fonte da água utilizada para consumo humano, essencial para a avaliação do risco sanitário.
- **Escoamento do banheiro:** Informa a solução sanitária adotada para o descarte de dejetos.
- **Destino do lixo:** Indica como o lixo é gerenciado e descartado no domicílio.
- **Possui eletricidade:** Sinaliza a disponibilidade de energia elétrica na residência, um indicador básico de infraestrutura.

- **Possui Animais de Estimação, Quantidade e Tipos:** Apresenta a relação da família com a fauna doméstica, categorizando a quantidade e os tipos de animais, o que é relevante para a vigilância de zoonoses.

A maioria dos campos detalhados na seção Condições de Moradia são de preenchimento opcional. Dessa forma, foi implementada uma solução de para ausência desses valores, consistente com o padrão do Prontuário Familiar e do e-SUS APS: para os valores não registrados, a label “Não informado”, formatada em itálico, é utilizada como valor padrão de exibição. Esta abordagem de sinalização de ausência de informação garante a coerência visual em toda a interface do módulo, conforme evidenciado na Figura 9.

Figura 9 - Condições de moradia - dados mínimos

Situação de moradia/Posse de terra <i>Alugado</i>	Localização <i>Urbana</i>	Número de cômodos <i>Não informado</i>
 Material predominante <i>Não informado</i>	 Tipo de acesso <i>Não informado</i>	
 Abastecimento de água <i>Não informado</i>	 Água para consumo <i>Não informado</i>	
 Escoamento do banheiro <i>Não informado</i>	 Destino do lixo <i>Não informado</i>	
 Possui eletricidade <i>Não informado</i>	 Possui animais de estimação Quantidade <i>Não informado</i> Tipos <i>Não informado</i>	

Fonte: O autor (2025)

6.1.7 Visitas realizadas

O histórico de visitas realizadas é materializado em uma listagem que apresenta as três últimas visitas domiciliares registradas para o imóvel em questão ou para os seus respectivos residentes, conforme pode ser visualizado na Figura 10.

Figura 10 - Histórico de visitas realizadas

Dados do imóvel	Visitas realizadas
Últimas visitas	
 Visita indefinida 10/11/2025 (Manhã) Motivo da visita: Cadastramento / Atualização; Visita periódica; Pessoa com hipertensão Cidadão visitado: Dependente Familiar Um - 131.484.122-06 Sem anotação Realizado por: Davi Felipe Becker Morfim Agente comunitário de saúde ESF 1- 1010010001	
 Visita individual 10/11/2025 (Tarde) Motivo da visita: Cadastramento / Atualização; Orientação / Prevenção; Consulta; Exame; Vacina; Saúde mental Cidadão visitado: Responsável Familiar - 835533956330004 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Realizado por: Davi Felipe Becker Morfim Agente comunitário de saúde ESF 1- 1010010001	
 Visita Familiar 10/11/2025 (Tarde) Motivo da visita: Cadastramento / Atualização; Visita periódica; Ação educativa Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Realizado por: Davi Felipe Becker Morfim Agente comunitário de saúde ESF 1- 1010010001	

Fonte: O autor (2025)

Os seguintes dados são exibidos nessa seção:

- **Tipo da Visita:** Identifica se o foco da visita foi na família como um todo ou em um residente específico do imóvel.
- **Data e Turno:** Informa a data exata e o turno (matutino, vespertino ou noturno) em que a visita foi realizada.
- **Motivo da Visita:** Descreve a razão principal que levou o profissional de saúde a realizar a visita (ex: acompanhamento de gestante, busca ativa, monitoramento de condição crônica, etc.).
- **Cidadão Visitado:** Apresenta o nome completo do cidadão acompanhado, juntamente com seu identificador principal (CPF ou Cartão Nacional de Saúde – CNS).
- **Anotação:** Campo de texto livre destinado a observações adicionais e informações não estruturadas que o profissional de saúde julgar relevantes para o contexto da visita.
- **Realizada por:** Indica o profissional que executou a visita, exibindo seu nome, o Código Brasileiro de Ocupação (CBO), o nome da equipe e o INE a que pertence.

Uma vez que a visita domiciliar não é obrigatória, existem famílias que nunca a receberam. Nesses casos, o histórico de visitas domiciliares exibe em itálico a mensagem: "Ainda não existem visitas associadas a esta família" (conforme ilustrado na Figura 11).

Figura 11 - Histórico de visitas domiciliares - sem visitas



Fonte: O autor (2025)

6.1.8 Estados de exceção

A aba de acesso ao Prontuário Familiar é projetada para ser exibida no cabeçalho de contexto do cidadão independentemente da existência de um vínculo familiar válido. Essa decisão de design garante que a funcionalidade esteja sempre acessível para o usuário que necessite iniciar o cadastro familiar.

Na ausência desse vínculo, a interface (Figura 12) apresenta uma visualização condicional com duas mensagens de orientação, seguindo o padrão informativo do sistema:

- A principal condição é destacada pela label "Este cidadão ainda não está cadastrado em uma família", apresentada em negrito.
- A instrução para correção ou complemento do cadastro é fornecida em seguida: "Você pode realizar essa atualização através do Acompanhamento

do território", formatada em itálico, indicando o caminho de navegação necessário para vincular o cidadão a um núcleo familiar.

Figura 12 - Prontuário Familiar - cidadão não possui uma família



Fonte: O autor (2025)

Uma segunda variante de exceção na exibição do Prontuário Familiar está relacionada à mudança de território. Este cenário é disparado quando o registro do cidadão indica que ele se desvinculou da área de abrangência da equipe de saúde da família responsável pelo acompanhamento.

Neste caso específico, a lógica de negócio assume que o cidadão não pertence mais a um núcleo familiar ativo na área. A interface (Figura 13) exibe a seguinte mensagem de alerta e justificativa, formatada em negrito: "Esse cidadão não está mais vinculado ao território, e, por isso, não possui família cadastrada."

Figura 13 - Prontuário Familiar - cidadão mudou de território



Fonte: O autor (2025)

6.1.9 Visualização completa do Prontuário Familiar

As seções anteriores detalharam individualmente os componentes do Prontuário Familiar. Nesta seção, apresentamos a visualização completa do prontuário.

Devido à extensão das informações, e à forma como as seções "Dados do imóvel" e "Visitas realizadas" são exibidas, a visualização completa foi dividida para melhor detalhamento:

- A Figura 14 destaca o Prontuário Familiar incluindo a seção "Dados do imóvel".

Figura 14 - Prontuário Familiar completo com seção “Dados do imóvel”

Responsável Familiar | Sexo Masculino

30 anos e 5 meses | Nasceu em 13/05/1995

CPF — CNS 835533956330004

Nome da mãe —

Telefone (48) 98829-9864

Unidade responsável Unidade Básica de Saúde 1

Atualizar cadastro

Última atualização em 10/11/2025

Informações

Prontuário Familiar

Encaminhamentos

Vacinação

Unificações

Agendamentos

Família de Responsável Familiar | Nº prontuário 1234567890

Equipe responsável

ESF 1

Renda familiar

Acima de 4 salários mínimos

Nº de membros declarados da família

4

Reside desde

12/2022

Acompanhamento de vulnerabilidade

Beneficiário do programa Bolsa Família

Risco de insegurança alimentar Observado em 10/11/2025

Responsável familiar

Responsável Familiar | Sexo Masculino

30 anos e 5 meses | Nasceu em 13/05/1995

CNS 835533956330004

Dependente Familiar Um | Sexo Feminino

29 anos e 2 meses | Nasceu em 12/08/1996

CPF 131.484.122-06

Parentesco com o responsável Cônjuge / Companheiro(a)

Dependente Familiar Dois | Sexo Feminino

15 anos e 7 meses | Nasceu em 15/03/2010

CPF 777.495.761-85

Parentesco com o responsável Enteadado(a)

Dependente Familiar Três | Sexo Masculino

3 anos | Nasceu em 04/11/2022

CPF 717.196.307-12

Parentesco com o responsável Filho(a)

Dados do imóvel

Visitas realizadas

Rua Lauro Linhares, 1600

Casa/domicílio

Torre Flora - Sala 901 - Trindade - Florianópolis/SC - 88038-001

Microárea 01

Telefone de contato (48) 98829-9864

Telefone residencial (48) 3032-9457

Geolocalização —

Situação de moradia/Posse de terra

Alugado

Localização

Urbana

Número de cômodos

8

Material predominante

Alvenaria com revestimento

Tipo de acesso

Pavimento

Abastecimento de água

Rede encanada até o domicílio - Sistema de abastecimento público (municipal ou estadual)

Água para consumo

Clorada

Escoamento do banheiro

Rede coletora de esgoto ou pluvial

Destino do lixo

Coletado

Possui eletricidade

Sim

Possui animais de estimação

Quantidade 4

Tipos Gato | Cachorro | Pássaro

Fonte: O autor (2025)

- A Figura 15 destaca o Prontuário Familiar incluindo a seção "Visitas domiciliares".

Figura 15 - Prontuário Familiar completo com seção “Visitas realizadas”

Responsável Familiar | Sexo Masculino

30 anos e 5 meses | Nasceu em 13/05/1995

Telefone (48) 98829-9864

CPF — CNS 835533956330004

Nome da mãe —

Unidade responsável Unidade Básica de Saúde 1

Atualizar cadastro

Última atualização em 10/11/2025

Informações

Prontuário Familiar

Encaminhamentos

Vacinação

Unificações

Agendamentos

Família de Responsável Familiar | N° prontuário 1234567890

Equipe responsável
ESF 1

Renda familiar
Acima de 4 salários mínimos

N° de membros declarados da família
4

Reside desde
12/2022

Acompanhamento de vulnerabilidade

Beneficiário do programa Bolsa Família

Risco de insegurança alimentar Observado em 10/11/2025

Responsável familiar

Responsável Familiar | Sexo Masculino

30 anos e 5 meses | Nasceu em 13/05/1995

CNS 835533956330004

Dependente Familiar Um | Sexo Feminino

29 anos e 2 meses | Nasceu em 12/08/1996

CPF 131.484.122-06

Parentesco com o responsável Cônjuge / Companheiro(a)

Dependente Familiar Dois | Sexo Feminino

15 anos e 7 meses | Nasceu em 15/03/2010

CPF 777.495.761-85

Parentesco com o responsável Entead(a)

Dependente Familiar Três | Sexo Masculino

3 anos | Nasceu em 04/11/2022

CPF 717.196.307-12

Parentesco com o responsável Filho(a)

Dados do imóvel

Visitas realizadas

Últimas visitas

Visita indefinida | 10/11/2025 (Manhã)

Motivo da visita: Cadastro / Atualização; Visita periódica; Pessoa com hipertensão

Cidadão visitado: Dependente Familiar Um - 131.484.122-06

Sem anotação

Realizado por: Davi Felipe Becker Morfim | Agente comunitário de saúde | ESF 1- 1010010001

Visita individual | 10/11/2025 (Tarde)

Motivo da visita: Cadastro / Atualização; Orientação / Prevenção; Consulta; Exame; Vacina; Saúde mental

Cidadão visitado: Responsável Familiar - 835533956330004

Realizado por: Davi Felipe Becker Morfim | Agente comunitário de saúde | ESF 1- 1010010001

Visita Familiar | 10/11/2025 (Tarde)

Motivo da visita: Cadastro / Atualização; Visita periódica; Ação educativa

Realizado por: Davi Felipe Becker Morfim | Agente comunitário de saúde | ESF 1- 1010010001

Fonte: O autor (2025)

6.2 Verificação do Atendimento aos Requisitos

A seguir, é apresentada a avaliação detalhada do módulo do Prontuário Familiar desenvolvido, em relação ao atendimento dos Requisitos Funcionais (RF) e Não Funcionais (RNF) estabelecidos previamente na Seção 4.2.

6.2.1 Requisitos Funcionais

Todos os requisitos funcionais definidos para o projeto foram plenamente atendidos. A implementação pode ser verificada da seguinte forma:

- **RF01 - Prontuário familiar no contexto do cidadão atual:** Este requisito central foi implementado com sucesso. O e-SUS APS permite que ao visualizar o prontuário do cidadão, o usuário também consiga visualizar a família do cidadão do contexto. O componente `ProntuarioFamiliarView` engloba os componentes filhos que detalham as outras seções e exibem o prontuário familiar completo.
- **RF02 - Lista de familiares, RF03 - Dados dos familiares, RF04 - Dados opcionais dos familiares, RF05 - Acompanhamento de vulnerabilidade social:** Estes quatro requisitos foram implementados com sucesso. O Prontuário Familiar exibe uma lista com as famílias na qual o cidadão do contexto está inserido. Nessa lista é apresentado dados relativos à família, composição familiar, dados dos familiares e o parentesco com o responsável familiar. O componente `FamiliaSection` englobou os componentes responsáveis por exibir o cabeçalho com informações específicas da família, a seção que apresenta as informações econômicas e a listagem com os cidadãos que participam da composição familiar. Vale ressaltar que as informações sócio-econômicas serão apresentadas somente quando pelo menos um dos familiares possua alguma vulnerabilidade social.
- **RF06 - Acesso a aba do Prontuário familiar:** Este requisito foi implementado com sucesso. A aba usada para acessar o Prontuário Familiar fica disponível para qualquer usuário que possua permissão de acesso ao prontuário do cidadão. Ficando localizada em diversos módulos, como o cidadão, e nos vários tipos de atendimentos disponibilizados pelo e-SUS APS.
- **RF07 - Navegação entre os prontuários dos familiares:** Este requisito foi implementado com sucesso. Na listagem dos membros que compõe família do cidadão do contexto, mais especificamente no *card* representado pelo componente `CidadaoFamiliaCard`, foi disponibilizado o `VisualizarCidadaoButton`, um botão com ícone de lupa que redireciona o

usuário para o cidadão escolhido. Essa funcionalidade permite que o usuário navegue fluidamente entre os membros da família, não se prendendo apenas ao cidadão do contexto.

- **RF08 - Dados do imóvel, RF09 - Dados do endereço. RF10 - Condições de moradia do imóvel:** Estes três requisitos referentes às informações do imóvel foram implementados com sucesso. O módulo agora apresenta um detalhamento completo do local de moradia e suas condições. O Prontuário Familiar exibe todas as informações relativas ao imóvel e ao endereço da família em questão. O componente `VisualizarImovelHeaderContent` foi o responsável por englobar e apresentar o cabeçalho com os dados específicos do imóvel e sua localização. Também foram detalhadas as condições de moradia do imóvel, incluindo aspectos de saneamento, acesso à água, tipo de construção, entre outros. Esta seção é controlada pelo componente `CondicoesMoradiaSection`, que consolida e exibe estas informações cruciais para a avaliação de risco.
- **RF11 - Lista das últimas visitas domiciliares, RF12 - Dados das visitas domiciliares:** Os dois requisitos referentes ao histórico de visitas domiciliares foram implementados com sucesso, garantindo o acesso rápido e organizado aos dados da assistência prestada no domicílio. O Prontuário Familiar exibe uma lista cronológica das últimas visitas domiciliares realizadas pela equipe de saúde à família. Isso permite que o profissional identifique rapidamente a frequência e o histórico recente de acompanhamento. Na listagem, o sistema apresenta o detalhamento completo daquela ocorrência, incluindo data, profissional responsável, motivos da visita. Ambos os requisitos são consolidados no componente `ProntuarioFamiliarVisitasListSection`, responsável por gerenciar a exibição tanto da lista quanto o acesso aos dados completos de cada visita, integrando o histórico de acompanhamento ao Prontuário Familiar.

6.2.2 Requisitos Não Funcionais

- **RNF01 - Desempenho:** Qualitativamente, o sistema demonstrou excelente responsividade. O carregamento das informações do módulo do Prontuário familiar alta ocorrem de forma praticamente instantânea após o clique do usuário na aba do módulo, proporcionando uma experiência fluida e em tempo real. Quanto
- **RNF02 - Segurança:** A segurança do módulo Prontuário Familiar foi estabelecida por meio da herança integral da infraestrutura do sistema hospedeiro, garantindo que o tratamento de todos os dados do cidadão seguisse rigorosamente os princípios da Lei Geral de Proteção de Dados (LGPD), com foco na minimização da exposição de dados. O controle de acesso no nível de recurso é garantido pela autorização em cada endpoint do GraphQL; dessa forma, somente usuários devidamente autorizados têm acesso à aba e aos dados do Prontuário Familiar. Por fim, a transparência e a responsabilidade são asseguradas pelas trilhas de auditoria do e-SUS APS, onde o acesso ao prontuário é registrado como uma visualização auditada.
- **RNF03 - Escalabilidade:** O desenvolvimento do módulo do Prontuário Familiar atendeu plenamente ao requisito de Escalabilidade ao ser integrado ao sistema consolidado e-SUS APS. Isso permitiu que o novo módulo fizesse uso da arquitetura de *back-end* robusta já existente e do *framework* Spring Boot. A adoção dessas tecnologias e padrões de desenvolvimento favorece inerentemente a escalabilidade do sistema, garantindo que o Prontuário Familiar possa suportar eficientemente o aumento projetado no volume de dados e o crescimento da base de usuários, sem comprometer o desempenho da aplicação.
- **RNF04 - Manutenibilidade:** Este requisito foi atendido por meio de uma arquitetura modular com clara separação entre as camadas de *back-end* e *front-end*. Essa abordagem, combinada com a aplicação de padrões de projeto consolidados (como *Repository* e *Service Layer*), assegura que o sistema seja facilmente evoluível, simplificando futuras atualizações, correções e expansões do Prontuário Familiar.

- **RNF05 - Compatibilidade:** A interface de visualização principal do Prontuário Familiar foi testada e demonstrou compatibilidade e operação correta nas versões recentes dos principais navegadores web modernos, como Google Chrome e Mozilla Firefox. A interface principal com o usuário, o `ProntuarioFamiliarView`, se mostrou responsivo a resolução mínima de 1024x768 píxeis, sem distorcer seus componentes.
- **RNF06 - Usabilidade:** O requisito de Usabilidade (RNF06) foi plenamente atendido, primeiramente, ao assegurar que o *design* do Prontuário Familiar seguisse o padrão de identidade visual do e-SUS APS. Para isso, foram utilizadas as cores e fontes da paleta do Bold-UI, o *design system* desenvolvido pelo Laboratório Bridge, garantindo a coerência visual com o restante da aplicação. Em segundo lugar, a navegação entre a lista de familiares e seus respectivos prontuários, foi implementada de forma altamente intuitiva e eficiente. Por meio do componente `VisualizarCidadaoButton`, o usuário necessita de apenas um clique para acessar o prontuário individual do cidadão, otimizando significativamente o tempo de busca e a experiência de uso.

6.3 Discussão das Escolhas de Projeto e Implementação

A seleção da arquitetura e das tecnologias para o desenvolvimento do módulo Prontuário Familiar foi primordialmente guiada pelo princípio da coerência arquitetural e pela necessidade de integração plena com o sistema hospedeiro, o e-SUS APS. Em vez de introduzir uma *stack* tecnológica inteiramente nova, a decisão estratégica foi aderir e alavancar a robustez da plataforma existente, garantindo a manutenibilidade e a previsibilidade do projeto.

A escolha do *back-end* para o módulo Prontuário Familiar seguiu rigorosamente a premissa de integração com o sistema hospedeiro, o e-SUS APS, tornando o uso do Java com Spring Boot não apenas uma preferência, mas uma decisão estratégica de coerência arquitetural. A familiaridade que possuía com estas tecnologias foi um fator catalisador, permitindo que o projeto fosse conduzido com

maior agilidade e segurança desde o início. O maior benefício de se trabalhar com um sistema já consolidado, no entanto, reside na reutilização inteligente de diversos artefatos das várias camadas de *back-end*. Isso se traduz na capacidade de herdar e adaptar *endpoints* de *query resolvers*, além de reaproveitar serviços e repositórios já testados e validados em produção. Essa sinergia eliminou a necessidade de recriar a lógica de acesso a dados e de infraestrutura básica, acelerando exponencialmente o desenvolvimento e conferindo uma confiança intrínseca na robustez e na qualidade do código que suporta as informações clínicas críticas do Prontuário Familiar.

Para o *front-end*, o uso de React com TypeScript revelou-se altamente acertada. O React permitiu a criação de uma interface de usuário dinâmica, essencial para a visualização rica e hierárquica de informações complexas, enquanto o TypeScript adicionou tipagem estática, aumentando a segurança e a manutenibilidade. Mais importante, a adesão estrita ao *design system* Bold-UI, utilizado no e-SUS APS, garantiu o pleno atendimento ao requisito de Usabilidade, assegurando que o PF tivesse uma coerência visual imediata com o restante da aplicação e otimizando a experiência do usuário. Outro ponto crucial que elevou o valor da utilização do Bold-UI foi o *framework* que ele disponibiliza. Este *design system* já oferece uma vasta gama de componentes prontos que abstraem diversos comportamentos e padrões de interação comuns. Dessa forma, a equipe de desenvolvimento não precisou criar componentes básicos do zero, como *labels*, caixas (*boxes*), *inputs*, entre outros, permitindo que o foco fosse direcionado à lógica de negócio específica do Prontuário Familiar.

Em suma, as escolhas de projeto não buscaram a simplificação tecnológica a todo custo, mas sim a integração perfeita e a funcionalidade robusta. Ao construir o Prontuário Familiar sobre uma fundação técnica consolidada, o projeto garante que o módulo não apenas cumpra seus requisitos atuais, mas também possua alta longevidade e capacidade de adaptação para futuras expansões e integrações exigidas pelo Ministério da Saúde, posicionando-o como uma ferramenta valiosa para o acompanhamento longitudinal das famílias na APS.

7 CONCLUSÃO

Este Trabalho de Conclusão de Curso dedicou-se à concepção, desenvolvimento e análise do módulo Prontuário Familiar no sistema e-SUS APS. Este capítulo tem a finalidade de consolidar os resultados do trabalho, sintetizando a jornada de desenvolvimento detalhada nos capítulos precedentes. Ao longo do texto, foram explorados a fundamentação teórica, a metodologia de projeto, a arquitetura do sistema, os detalhes da implementação e a análise da solução finalizada.

O capítulo se inicia com a síntese do projeto e o alcance dos objetivos propostos, seguida pelo destaque às principais contribuições do trabalho para a área de Saúde Digital. Em um movimento de reflexão crítica, o texto prossegue com o reconhecimento das limitações identificadas durante o desenvolvimento. Por fim, são apresentadas sugestões para trabalhos futuros e a continuidade do projeto.

7.1 Síntese do Projeto e Alcance dos Objetivos

O presente Trabalho de Conclusão de Curso cumpriu integralmente seu objetivo principal de implementar o módulo do Prontuário Familiar no sistema e-SUS APS. O projeto foi motivado pela lacuna na funcionalidade do *Prontuário Eletrônico do Cidadão (PEC)*, que não abordava integralmente a visão holística e centrada na unidade familiar exigida pela Atenção Primária à Saúde (APS) e pela Estratégia de Saúde da Família (ESF).

A solução foi desenvolvida com sucesso, aderindo rigorosamente à arquitetura de software existente no e-SUS APS. O back-end foi implementado utilizando Java 17 com o framework Spring Boot, enquanto o front-end utilizou React em conjunto com TypeScript. A adesão ao BOLD Design System, desenvolvido pelo Laboratório Bridge da UFSC, foi uma decisão estratégica que otimizou significativamente o processo de desenvolvimento. Esta abordagem garantiu a coerência visual e o atendimento ao requisito de usabilidade, uma vez que permitiu a reutilização de uma vasta gama de componentes prontos, eliminando a necessidade de desenvolver elementos básicos da interface do zero. A arquitetura

empregou o GraphQL como *data gateway*, essencial para otimizar o consumo de informações e garantir a baixa latência nas consultas.

O módulo finalizado permite a visualização imediata de dados essenciais dos familiares, do imóvel e do histórico de visitas domiciliares, cumprindo todos os Requisitos Funcionais e os Requisitos Não Funcionais estabelecidos.

7.2 Principais Contribuições do Trabalho

Este trabalho entrega uma solução funcional e tecnologicamente robusta, destacando as seguintes contribuições:

- **Qualificação do Cuidado em Saúde:** A principal contribuição é a materialização de uma ferramenta que apoia a visão de cuidado centrada na família, elemento fundamental da ESF. Isso permite aos profissionais visualizar, de forma integrada, o contexto social e de saúde de toda a família, garantindo a continuidade e a qualificação do cuidado à saúde de seus membros.
- **Inovação e Pioneirismo Tecnológico:** O projeto demonstrou ser aderente ao quesito inovação, visto que não foram encontrados trabalhos correlatos com a mesma proposta de desenvolvimento de um prontuário familiar eletrônico associado à estratégia de saúde da família do governo federal.
- **Aplicação de Boas Práticas de Desenvolvimento:** O projeto demonstrou a aplicação de metodologias ágeis (Scrum e b_Thinking) e um rigoroso controle de qualidade, que incluiu a criação de testes automatizados (Unidade e Integração). A arquitetura utilizou padrões de projeto consolidados (Service Layer e Repository) e tecnologias contemporâneas como Java 17 com Spring Boot no back-end, React/TypeScript no front-end, e GraphQL como principal *data gateway*. Adicionalmente, a adesão ao BOLD Design System garantiu a usabilidade e a consistência visual da interface.
- **Robustez e Conformidade:** A solução adere rigorosamente aos princípios da Lei Geral de Proteção de Dados (LGPD) e ao controle de acesso por recurso,

garantindo a segurança necessária para um sistema que lida com informações clínicas sensíveis.

7.3 Reconhecimento das Limitações

Conforme discutido na análise do projeto (Capítulo 6), as principais limitações da implementação inicial são:

- **Escopo funcional restrito à visualização:** O foco da implementação foi estritamente a visualização das informações do Prontuário Familiar. Não está no escopo a criação de novas funcionalidades para edição ou persistência de dados cadastrais de cidadãos, ou famílias, que continuam a ser gerenciadas pelos módulos já existentes.
- **Validação Pós-Entrega:** Em virtude da ausência de contato contínuo com os usuários finais, as etapas de avaliação da *release* e de melhoria contínua da metodologia b_Thinking foram realizadas com base em análises internas, e não por meio de validação direta pós-entrega.

7.4 Trabalhos Futuros e Continuidade do Projeto

Para a evolução do módulo e a expansão de seu impacto, sugerem-se os seguintes trabalhos futuros, que visam aumentar a utilidade e a integração do Prontuário Familiar:

- **Gestão direta do Prontuário Familiar:** Integrar funcionalidades que permitam o profissional realizar a gestão da família diretamente no contexto do Prontuário Familiar. Este poderia realizar alterações no núcleo familiar, registrar informações de visitas domiciliares e até aplicar a Triagem para o Risco de Insegurança Alimentar (TRIA).

- **Aprofundamento dos Dados Clínicos e Hereditários:** Incluir a visualização de dados clínicos relevantes para o núcleo familiar, com ênfase em condições de saúde que possam ser hereditárias ou que apresentem alta prevalência no ambiente familiar.
- **Enriquecimento Socioeconômico:** Exibir informações socioeconômicas e de vulnerabilidade social mais detalhadas e consolidadas (além do status de Bolsa Família e Risco de Insegurança Alimentar), proporcionando uma compreensão mais completa dos determinantes sociais da saúde da família.
- **Geração de Insights de Risco:** Utilizar os dados agregados para gerar alertas ou insights automáticos sobre riscos que afetam o núcleo familiar, facilitando a priorização de ações preventivas pela equipe de saúde da família.
- **Genograma:** Um dos trabalhos futuros mais estratégicos é a integração do Genograma (representação gráfica da estrutura familiar, das relações e do histórico de saúde ao longo de gerações) no Prontuário Familiar. O Genograma é uma ferramenta essencial na avaliação clínica da família (MCGOLDRICK e GERSON, 2011), pois permite a rápida identificação visual de padrões de doenças hereditárias, relações complexas e o ciclo de vida familiar.

8 CONSIDERAÇÕES FINAIS

O presente trabalho representa a conclusão de um ciclo de desenvolvimento complexo, resultando na entrega de uma solução essencial para a qualificação do cuidado em saúde na Atenção Primária brasileira. A implementação do Prontuário Familiar no e-SUS APS é um exemplo da aplicação bem-sucedida de padrões modernos de engenharia de *software* (Spring Boot, GraphQL, TypeScript, Design System) em um sistema de escala nacional, garantindo a longevidade e a capacidade de adaptação do módulo para futuras integrações.

O sucesso da solução valida a premissa de que a tecnologia, quando alinhada às políticas de saúde e às metodologias ágeis (Scrum e b_Thinking), é um vetor de modernização da gestão da informação e de fortalecimento da Estratégia de Saúde da Família. O Prontuário Familiar, ao facilitar a visão holística do paciente no contexto familiar, apoia diretamente os princípios do SUS, reafirmando o compromisso com um sistema de saúde mais justo, resolutivo e centrado no cidadão e sua família. Espera-se que esta implementação sirva como referência e estímulo para novas iniciativas de Saúde Digital no Brasil.

REFERÊNCIAS

AI FUTURE SCHOOL. Introdução ao GraphQL para Desenvolvedores.: AI Future School, [s.d.]. Disponível em:

<https://www.ai-futureschool.com/pt/programacao/introducao-ao-graphql-para-desenvolvedores.php>. Acessado em: 30 out. 2025.

AGILE ALLIANCE. User Stories. [S.l.]: Agile Alliance, [s.d.]. (Glossary). Disponível em: <https://www.agilealliance.org/glossary/user-stories/>. Acesso em: 02 nov. 2025.

APOLLO GRAPHQL. Resolving a mutation successfully.: Apollo GraphQL, [s.d.]. Disponível em:

<https://www.apollographql.com/tutorials/lift-off-part4/05-resolving-a-mutation-successfully>. Acessado em: 30 out. 2025.

ATHAYDE, E.S.; GIL, C.R.R. Possibilidades do uso de genograma familiar no trabalho cotidiano dos médicos das equipes de saúde da família de Londrina. Espaço para a Saúde. Londrina, v. 6, n. 2, p. 13-22, jun. 2005.

AWS. How to Build GraphQL Resolvers for AWS Data Sources.: Amazon Web Services, [s.d.]. Disponível em: <https://aws.amazon.com/graphql/resolvers/>. Acesso em: 30 out. 2025.

CARVALHO, Renata Elyonara de Sousa et al. Validade do instrumento para triagem de domicílios em risco de insegurança alimentar em diversos estratos da população brasileira. Cadernos de Saúde Pública, v. 38, p. e00239521, 2022.

CHINA, Chrystal R.; JACKSON, Gita. ¿Qué es GraphQL?: IBM, [s.d.]. Disponível em: <https://www.ibm.com/mx-es/think/topics/graphql>. Acessado em: 30 out. 2025.

COBERTURA da Atenção Primária: PNS 2020 - 2023. PNS 2020 - 2023. 2025. Disponível em: <https://relatorioaps.saude.gov.br/cobertura/pns>. Acesso em: 18 set. 2025.

CIELO, Ana Claudia; GOMES, Igor de Carvalho; OLIVEIRA JÚNIOR, João Geraldo de (org.). e-SUS Atenção Primária à Saúde (e-SUS APS): manual de uso. 2021. Disponível em: <https://cgiap-saps.github.io/Manual-eSUS-APS/>. Acesso em: 19 set. 2025.

ESTRATÉGIA Saúde da Família. Disponível em:

<https://www.gov.br/saude/pt-br/composicao/saps/esf>. Acesso em: 18 set. 2025.

F5. Glossário: GraphQL.: F5, [s.d.]. Disponível em:

https://www.f5.com/es_es/glossary/graphql. Acessado em: 30 out. 2025.

GAETE, Rodrigo André Cuevas. **Informatização do Processo de Enfermagem na Atenção Primária à Saúde**. 2020. 386 f. Tese (Doutorado) - Curso de Pós-Graduação em Enfermagem em Saúde Pública, Escola de Enfermagem de Ribeirão Preto, Universidade de São Paulo, Ribeirão Preto, 2020. Disponível em: <https://www.teses.usp.br/teses/disponiveis/22/22133/tde-06072020-142744/publico/RODRIGOANDRECUEVASGAETE.pdf>. Acesso em: 18 set. 2025.

GEEKSFORGEEEKS. What Are Over-Fetching and Under-Fetching?: GeeksforGeeks, [s.d.]. Disponível em:

<https://www.geeksforgeeks.org/graphql/what-are-over-fetching-and-under-fetching/>.

Acessado em: 30 out. 2025.

GRAPHQL FOUNDATION. Solving the N+1 Problem with DataLoader.: GraphQL Foundation, [s.d.]. Disponível em: <https://www.graphql-js.org/docs/n1-dataloader/>. Acesso em: 30 out. 2025.

INSTACLUSTER. Complete guide to PostgreSQL: Features, use cases, and tutorial - Instacluster. Disponível em:

<https://www.instacluster.com/education/postgresql/complete-guide-to-postgresql-features-use-cases-and-tutorial/>. Acessado em: 08 nov. 2025.

LABORATÓRIO BRIDGE. BOLD: Design System. Florianópolis: Laboratório Bridge, UFSC, [s.d.]. Disponível em: <https://bold.bridge.ufsc.br/en/>. Acesso em: 01 nov. 2025.

MCGOLDRICK, Monica; GERSON, Randy. *Genogramas: Avaliação e Intervenção Familiar*. 3. ed. Porto Alegre: Artmed, 2011.

ORÇAMENTO do SUS cresceu 30% nos últimos 3 anos; quase metade da expansão foi apropriada por Emendas Parlamentares: Nota Técnica n. 38 analisa a evolução do gasto federal em saúde entre 2015 e 2025. Nota Técnica n. 38 analisa a evolução

do gasto federal em saúde entre 2015 e 2025. 2025. Notícia do portal IEPS - Instituto de Estudos para Políticas de Saúde. Disponível em:

<https://ieps.org.br/orcamento-do-sus-cresceu-30-nos-ultimos-3-anos-quase-metade-da-expansao-foi-apropriada-por-emendas-parlamentares/>. Acesso em: 18 set. 2025.

MAIS de 87% das Unidades Básicas de Saúde utilizam prontuário eletrônico e quase a totalidade tem acesso à internet. 2025. Disponível em:

<https://www.gov.br/saude/pt-br/assuntos/noticias/2025/junho/mais-de-87-das-unidades-basicas-de-saude-utilizam-prontuario-eletronico-e-quase-a-totalidade-tem-acesso-a-internet>. Acesso em: 19 set. 2025.

MANIFESTO PARA DESENVOLVIMENTO ÁGIL DE SOFTWARE. [S.l.: s.n.], 2001. Disponível em: <https://agilemanifesto.org/iso/ptbr/manifesto.html>. Acesso em: 07 nov. 2025.

MANUAL e-SUS APS: Capítulo 6 - Atendimento.. Capítulo 6 - Atendimento.. 2025. 6.4.2 Folha de Rosto. Disponível em:

https://sisaps.saude.gov.br/sistemas/esusaps/docs/manual/PEC/PEC_06_atendimentos/#642-folha-de-rosto. Acesso em: 05 out. 2025.

MINISTÉRIO DA SAÚDE. **30/2025-CGAN/DEPPROS/SAPS/MS**: NOTA TÉCNICA Nº 30/2025-CGAN/DEPPROS/SAPS/MS. Distrito Federal: Coordenação-Geral de Alimentação e Nutrição, 2025. Disponível em:

<https://www.gov.br/saude/pt-br/centrais-de-conteudo/publicacoes/notas-tecnicas/2025/nota-tecnica-no-30-2025-cgan-deppros-saps-ms>. Acesso em: 05 out. 2025.

ORACLE. Oracle® Database Database Reference 19c. Disponível em:

<https://docs.oracle.com/en/database/oracle/oracle-database/19/refrn/>. Acessado em: 08 nov. 2025.

ORACLE. High Availability and Data Protection – Getting From Requirements ... Disponível em:

<https://docs.oracle.com/en/database/oracle/oracle-database/23/haovw/ha-requirements-architecture.html>. Acessado em: 05 nov. 2025.

PAIM, J. S. Saúde da Família: espaço de reflexão e de contra-hegemonia. Interface - Comunicação, Saúde, Educação , v. 5, n. 9, p. 143–146, ago. 2001.

PIXEL FREE STUDIO. How to Use GraphQL for Efficient Data Fetching.: Pixel Free Studio, [s.d.]. Disponível em:
<https://blog.pixelfreestudio.com/how-to-use-graphql-for-efficient-data-fetching/>.
Acessado em: 30 out. 2025.

POBLACION, Ana et al. Validity of a 2-item screening tool to identify families at risk for food insecurity in Brazil. Cadernos de Saúde Pública, v. 37, p. e00132320, 2021

POSTGRESQL. About - PostgreSQL. Disponível em:
<https://www.postgresql.org/about/>. Acessado em: 08 nov. 2025.

REFACTORING.GURU. Padrões de Design. [S.l.]: Refactoring.Guru, [s.d.]. Disponível em: <https://refactoring.guru/design-patterns>. Acesso em: 01 nov. 2025.

SCRUM.ORG. What is Scrum? [S.l.]: Scrum.org, [s.d.]. Disponível em:
<https://www.scrum.org/resources/what-is-scrum>. Acesso em: 07 nov. 2025.

Secretaria de Estado da Saúde do Ceará. Metodologia de melhoria da qualidade em APS – PROQUALIS. 2a Ed. Fortaleza: Secretaria de Estado da Saúde do Ceará; 2005.

SILVA, Landwehrner Lucena da *et al.* Prontuário da família: coerência e comunicação na equipe de saúde da família. **Anais do Cbmfc**: 12º Congresso Brasileiro de Medicina da Família e Comunidade, Belém, v. 1, n. 12, p. 904-904, maio 2013. Disponível em:
<https://cmfc.emnuvens.com.br/brasileiro/article/view/733/729>. Acesso em: 05 out. 2025.

SPRING. Spring for GraphQL [s.d]a. Disponível em:
<https://docs.spring.io/spring-boot/reference/web/spring-graphql.html>. Acesso em: 01 nov. 2025.

Prontuário Familiar no e-SUS APS: Implementação de uma ferramenta para qualificação do cuidado em saúde na Atenção Primária do SUS

Davi Felipe Becker Morfim

Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brasil

***Abstract.** This paper describes the implementation of the Family Health Record module within the e-SUS APS system, the main software for Primary Health Care in Brazil. The solution addresses the lack of a family-centered view in the Electronic Health Record. Developed using Java, Spring Boot, and GraphQL, the module integrates social, clinical, and housing data, enabling longitudinal care focused on the family unit. The results demonstrate a functional tool compliant with performance and security requirements.*

***Resumo.** Este artigo descreve a implementação do módulo de Prontuário Familiar no sistema e-SUS APS, principal software da Atenção Primária à Saúde no Brasil. A solução aborda a lacuna de uma visão centrada na família no Prontuário Eletrônico do Cidadão. Desenvolvido com Java, Spring Boot e GraphQL, o módulo integra dados sociais, clínicos e habitacionais, viabilizando o cuidado longitudinal com foco no núcleo familiar. Os resultados demonstram uma ferramenta funcional e aderente aos requisitos de desempenho e segurança.*

1. Introdução

O Sistema Único de Saúde (SUS) tem na Atenção Primária à Saúde (APS) a sua principal porta de entrada, sendo a Estratégia Saúde da Família (ESF) o modelo prioritário para sua expansão e consolidação. Neste contexto, a família é entendida como o objeto central da atenção, devendo ser tratada a partir do território onde vive e de suas relações sociais.

Para apoiar a informatização das Unidades Básicas de Saúde (UBS), o Ministério da Saúde desenvolveu o Prontuário Eletrônico do Cidadão (PEC) como parte da estratégia e-SUS APS.

Embora o PEC seja uma ferramenta robusta para o registro clínico individualizado, existia uma lacuna sistêmica: a ausência de uma visão integrada e holística focada na unidade familiar. Essa limitação dificultava a efetividade do cuidado longitudinal preconizado pela ESF.

Este trabalho apresenta o desenvolvimento do módulo Prontuário Familiar, integrado ao e-SUS APS. O objetivo foi fornecer aos profissionais de saúde uma ferramenta para visualizar, de forma agrupada, a composição familiar, condições de moradia, vulnerabilidade social e histórico de visitas domiciliares, utilizando tecnologias modernas como GraphQL para otimização de dados.

2. Metodologia e Tecnologias

O desenvolvimento do sistema seguiu uma abordagem iterativa e incremental baseada no framework Scrum, combinada com a metodologia de descoberta de produto b_Thinking (do Laboratório Bridge/UFSC) para levantamento e validação de requisitos junto aos stakeholders.

2.1. Arquitetura do Sistema

O módulo foi construído sobre a arquitetura consolidada do e-SUS APS, garantindo escalabilidade e segurança.

Back-end: Utilizou-se Java 17 com o framework Spring Boot. A comunicação de dados foi implementada via GraphQL, o que permitiu mitigar problemas de over-fetching e under-fetching comuns em APIs REST, otimizando o carregamento de árvores de dados complexas (como a estrutura familiar e seus membros). Para resolver o problema de N+1 queries, foram utilizados DataLoaders para batching e caching das requisições ao banco de dados.

Front-end: A interface foi desenvolvida em React com TypeScript, garantindo tipagem estática e segurança no desenvolvimento. A identidade visual seguiu estritamente o Bold Design System, padrão oficial do e-SUS APS, assegurando usabilidade e acessibilidade.

Banco de Dados: A persistência de dados foi gerenciada via JPA/Hibernate, mantendo compatibilidade tanto com PostgreSQL quanto com Oracle, conforme requisitos do sistema nacional.

3. Implementação do Prontuário Familiar

O módulo foi estruturado para ser acessado diretamente a partir do contexto do cidadão em atendimento, permitindo uma transição fluida entre a visão individual e a familiar. As principais funcionalidades implementadas incluem:

3.1. Composição Familiar e Vulnerabilidade

O sistema exibe a lista completa dos membros da família, destacando o Responsável Familiar. Foi implementada uma lógica de verificação de vulnerabilidade social que alerta, visualmente, se a família possui integrantes beneficiários do programa Bolsa Família ou se foi identificada com Risco de Insegurança Alimentar através da triagem (TRIA).

3.2. Dados do Imóvel e Condições de Moradia

O módulo carrega e apresenta dados detalhados do domicílio, incluindo geolocalização, tipo de abastecimento de água, destino do lixo e material predominante da construção. Essas informações são cruciais para que a equipe de saúde avalie os determinantes sociais e ambientais que afetam a saúde do grupo.

3.3. Histórico de Visitas Domiciliares

Foi desenvolvida uma seção que lista cronologicamente as visitas domiciliares realizadas pela equipe de saúde. O sistema apresenta o motivo da visita, o desfecho, o profissional responsável e anotações clínicas, permitindo o monitoramento da assistência prestada no território.

4. Resultados e Análise

A solução atendeu a todos os Requisitos Funcionais (RF) e Não Funcionais (RNF) estabelecidos. Testes de desempenho demonstraram que o carregamento das informações (composição familiar, imóvel e visitas) ocorre com latência mínima perceptível (menos de 0,5 segundos), graças à eficiência dos resolvers GraphQL e dos DataLoaders implementados.

A segurança foi garantida através do controle de acesso por recurso (RBAC) e conformidade com a Lei Geral de Proteção de Dados (LGPD), assegurando que apenas profissionais autorizados visualizem os dados sensíveis da família. A interface, aderente ao Bold Design System, proporcionou uma experiência de usuário consistente e intuitiva.

5. Conclusão

A implementação do Prontuário Familiar no e-SUS APS preenche uma lacuna histórica na informatização da Atenção Primária no Brasil. Ao centralizar as informações do núcleo familiar, a ferramenta potencializa a capacidade das equipes de saúde de realizar um cuidado mais assertivo e contextualizado. Como trabalhos futuros, sugere-se a implementação da visualização gráfica via Genograma e a gestão direta de cadastros dentro do módulo.

References

LABORATÓRIO BRIDGE. BOLD: Design System. Florianópolis: Laboratório Bridge, UFSC, [s.d.].

MINISTÉRIO DA SAÚDE. Nota Técnica N° 30/2025-CGAN/DEPPROS/SAPS/MS. Brasília: Ministério da Saúde, 2025.

SILVA, L. L. et al. Prontuário da família: coerência e comunicação na equipe de saúde da família. Anais do 12º Congresso Brasileiro de Medicina da Família e Comunidade, 2013.

SPRING. Spring for GraphQL. [s.d.]. Disponível em: <https://docs.spring.io/spring-boot/reference/web/spring-graphql.html>.