Instituto Federal de Educação, Ciências e Tecnologia Goiano
Campus Morrinhos
Tecnologia em Sistemas para Internet

RAFAEL DE SOUZA COSTA

PROJETO SAÚDE DO INTERIOR

DESENVOLVIMENTO E APLICAÇÃO DE UM SISTEMA VOLTADO À GESTÃO DA SAÚDE RURAL

MORRINHOS – GO
2025

RAFAEL DE SOUZA COSTA

PROJETO SAÚDE DO INTERIOR

DESENVOLVIMENTO E APLICAÇÃO DE UM SISTEMA VOLTADO À GESTÃO DA SAÚDE RURAL

Trabalho apresentado ao Instituto Federal Goiano Campus Morrinhos como requisito para conclusão do curso de Tecnologia em Sistemas para Internet.

Orientador: Prof. Marcel Melo

MORRINHOS – GO
2025

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema Integrado de Bibliotecas do IF Goiano - SIBi

Costa, Rafael de Souza
C837p Projeto Saúde do Interior: Desenvolvimento e Aplicação de um Sistema Voltado à Gestão da Saúde Rural / Rafael de Souza Costa. Morrinhos - Goiás 2025.

96f. il.

Orientador: Prof. Me. Marcel da Silva Melo.
Tcc (Bacharel) - Instituto Federal Goiano, curso de 0421176 - Curso Superior de Tecnologia em Sistemas para Internet - Morrinhos (especial) (Campus Morrinhos).
I. Título.

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO

PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610, de 19 de fevereiro de 1998, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano a disponibilizar gratuitamente o documento em formato digital no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

IDENTIFICAÇÃO DA PRODUÇÃO TÉCNICO-CIENTÍFICA

- ☐ Tese (doutorado)
- ☐ Dissertação (mestrado)
- ☐ Monografia (especialização)
- ☑ TCC (graduação)
- ☐ Artigo científico
- ☐ Capítulo de livro
- ☐ Livro
- ☐ Trabalho apresentado em evento
- ☐ Produto técnico e educacional - Tipo:

Nome completo do autor:
RAFAEL DE SOUZA COSTA

Matrícula:
2020104211710109

Título do trabalho:
PROJETO SAÚDE DO INTERIOR - DESENVOLVIMENTO E APLICAÇÃO DE UM SISTEMA VOLTADO À GESTÃO DA SAÚDE RURAL

RESTRIÇÕES DE ACESSO AO DOCUMENTO

Documento confidencial: ☑ Não ☐ Sim, justifique:

Informe a data que poderá ser disponibilizado no RIIF Goiano: ☐/☐/☐

O documento está sujeito a registro de patente? ☐ Sim ☑ Não

O documento pode vir a ser publicado como livro? ☑ Sim ☐ Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O(a) referido(a) autor(a) declara:

• Que o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

• Que obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autoria, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

• Que cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Morrinhos — Local
06 / 10 / 2024 — Data

gov.br Documento assinado digitalmente
RAFAEL DE SOUZA COSTA
Data: 06/10/2025 17:14:19-0300
Verifique em https://validar.iti.gov.br

Assinatura do autor e/ou detentor dos direitos autorais

Ciente e de acordo:

gov.br Documento assinado digitalmente
MARCEL DA SILVA MELO
Data: 06/10/2025 17:27:35-0300
Verifique em https://validar.iti.gov.br

Assinatura do(a) orientador(a)

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

Ata nº 22/2025 - CCEPTNM-MO/CEPTNM-MO/DE-MO/CMPMHOS/IFGOIANO

ATA DE DEFESA DE TRABALHO DE CURSO

Ao(s) **08** dia(s) do mês de setembro de 2025, às 19 horas e 30 minutos, reuniu-se a banca examinadora composta pelos docentes: Me. Marcel da Silva Melo, presidente da banca e orientador do discente, prof. Dr. Fernando Barbosa Matos e prof. Dr. Rodrigo Elias Francisco para examinar o Trabalho de Curso intitulado "**Projeto Saúde do Interior – Desenvolvimento e Aplicação de um Sistema Voltado à Gestão da Saúde Rural**" do estudante **Rafael de Souza Costa**, Matrícula nº 2020104211710109 do Curso Superior de Tecnologia em Sistemas para Internet do IF Goiano – Campus Morrinhos. A palavra foi concedida ao estudante para a apresentação oral do TC, houve arguição do candidato pelos membros da banca examinadora. Após tal etapa, a banca examinadora decidiu pela APROVAÇÃO do discente, mediante as correções solicitadas pelo membros da banca. Ao final da sessão pública de defesa foi lavrada a presente ata que segue assinada pelos membros da Banca Examinadora.

(Assinado Eletronicamente)
Marcel da Silva Melo
Presidente da Banca(a)

(Assinado Eletronicamente)
Fernando Barbosa Matos
Membro

(Assinado Eletronicamente)
Rodrigo Elias Francisco
Membro

Documento assinado eletronicamente por:

- **Marcel da Silva Melo**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 08/09/2025 20:58:53.
- **Rodrigo Elias Francisco**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 08/09/2025 21:00:53.
- **Fernando Barbosa Matos**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 09/09/2025 11:24:59.

Este documento foi emitido pelo SUAP em 08/09/2025. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 742058
Código de Autenticação: f6374bfbe4

INSTITUTO FEDERAL GOIANO
Campus Morrinhos
Rodovia BR-153, Km 633, Zona Rural, SN, Zona Rural, MORRINHOS / GO, CEP 75650-000
(64) 3413-7900

AGRADECIMENTOS

Este trabalho representa o resultado de uma trajetória marcada pelo aprendizado, dedicação e pelo apoio de inúmeras pessoas que, de forma direta ou indireta, contribuíram para sua concretização. Agradeço, primeiramente, a Deus, pela vida, pelas oportunidades e pela força concedida em cada etapa desta jornada. Aos meus pais e familiares, pelo amor, incentivo e paciência constantes, fundamentais para que eu pudesse seguir em frente mesmo diante das dificuldades.

Expresso também o meu agradecimento ao meu orientador Marcel, pela orientação precisa, paciência e pelas valiosas contribuições que possibilitaram o amadurecimento deste projeto. Aos professores e colegas de curso, pelo compartilhamento de conhecimentos, experiências e incentivo ao longo de todo o percurso acadêmico.

Por fim, agradeço aos representantes da Secretaria Municipal de Saúde de Professor Jamil – GO, pela colaboração e pelas informações que serviram de base e inspiração para o desenvolvimento do Projeto Saúde do Interior, e a todos que, de alguma forma, contribuíram para a realização deste trabalho.

RESUMO

Esta pesquisa apresenta o desenvolvimento do sistema Saúde do Interior, criado para apoiar agentes de saúde em visitas a áreas rurais com pouca ou nenhuma conectividade. O objetivo é descrever o funcionamento da solução e analisar sua aplicação na gestão da saúde pública em regiões remotas. O estudo considera aspectos técnicos e o contexto da saúde rural, com base em práticas observadas na Secretaria de Saúde de Professor Jamil – GO. A abordagem metodológica combina os métodos descritivo e exploratório, permitindo detalhar a estrutura do sistema e investigar tecnologias adequadas à sua implementação.

Os resultados mostram que o sistema permite o registro e gerenciamento de dados de pacientes em áreas com conectividade limitada, organizando informações por regiões, famílias, pacientes e serviços. Funcionalidades como sincronização de dados, armazenamento local com SQFLite e interface em Flutter viabilizam o uso offline, atendendo aos desafios iniciais. A análise dos casos de uso demonstra como o sistema auxilia os profissionais na coleta e acompanhamento das informações dos pacientes, promovendo uma abordagem estruturada e eficiente.

Além disso, a pesquisa identifica desafios e oportunidades no uso de tecnologias móveis para aprimorar a gestão da saúde rural, considerando conectividade, armazenamento e segurança da informação. Para trabalhos futuros, propõe-se a integração com uma plataforma web centralizada, expansão de funcionalidades e inclusão de novos módulos e relatórios, visando melhorar a gestão da saúde pública em áreas de difícil acesso.

Palavras-chave: Saúde pública na zona rural; Sincronização de dados offline; Armazenamento *Offline*; Tecnologia da informação em saúde; Sistema de informação em saúde.

LISTA DE ILUSTRAÇÕES

FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACID (Atomicidade, Consistência, Isolamento e Durabilidade)
API (*Application Programming Interface*)
ARM (*Advanced RISC Machine*)
CPF (Cadastro de Pessoa Físicas)
ECDSA (*Elliptic Curve Digital Signature Algorithm*)
GPS (*Global Positioning System*)
HMAC (*Keyed-hash Message Authentication Code*)
HTML (*HyperText Markup Language*)
HTTP (*Hypertext Transfer Protocol*)
IBGE (Instituto Brasileiro de Geografia e Estatística)
ID (*Identification*)
IoT (*Internet of Things*)
JIT (*Just-In-Time*)
JSON (JavaScript Object Notation)
JWT (JSON *Web Token*)
MAC (*Macintosh*)
PDF (*Portable Document Format*)
POO (Programação Orientada a Objetos)
REST (*Representational State Transfer*)
RFC 7519 (*Request for Comments 7519*)
RSA (*Rivest Shamir Adleman*)
SGBD (*Database Management System*)
SGBDR (*Database Management System Relational*)
SHA256 (*Secure Hash Algorithm 256-bit*)
SQL (*Structured Query Language*)
UC (Caso de Uso)
UI (Interface de Usuário)
UML (*Unified Modeling Language*)
URL (*Uniform Resource Locator*)
WEB (*World Wide Web*)

SUMÁRIO

1. INTRODUÇÃO

O acesso à saúde em áreas rurais apresenta desafios significativos devido à distância entre comunidades e unidades de atendimento, dificuldades logísticas e limitações na conectividade com a internet. Diante desse cenário, soluções tecnológicas voltadas para a gestão da saúde pública têm sido exploradas, como alternativas para otimizar o acompanhamento de pacientes e a organização dos atendimentos realizados por agentes de saúde.

Diferente de abordagens tradicionais baseadas em registros físicos, o “Saúde do Interior” adota uma estrutura digital que permite a organização e sincronização de dados, facilitando o acesso e a tomada de decisão pelos agentes de saúde, proporcionando um sistema estruturado para o registro de informações sobre pacientes, grupos familiares, atendimentos e serviços prestados. A aplicação foi projetada para operar de forma *offline* (sem internet), permitindo a coleta e armazenamento de dados mesmo em regiões onde a conectividade é limitada, com posterior sincronização quando a internet estiver disponível.

Para viabilizar essa solução, foram adotadas tecnologias como *Flutter* para o desenvolvimento do aplicativo, *SQFLite* para o armazenamento local dos dados e um sistema de sincronização via *Application Programming Interface* (API) para integrar as informações a uma base centralizada. A estruturação do sistema foi baseada em um modelo de dados que organiza os registros em categorias como regiões, grupos, pacientes e serviços, facilitando a consulta e a gestão das informações coletadas.

A pesquisa adota uma abordagem descritiva e exploratória, sendo descritiva ao apresentar a estrutura, funcionalidades e implementação do sistema, e exploratória ao investigar tecnologias e metodologias antes da implementação, analisando sua aplicabilidade no contexto da saúde rural.

Este trabalho visa documentar o desenvolvimento do “Saúde do Interior”, desde sua concepção até a implementação, destacando sua contribuição para a informatização da saúde rural e os desafios técnicos enfrentados ao criar um sistema offline e integrado. Além de documentar a implementação do projeto, este estudo discute os desafios enfrentados na adoção de tecnologias móveis em regiões rurais, bem como as soluções aplicadas para garantir a funcionalidade do sistema em contextos de baixa conectividade.

2. REFERENCIAL TEÓRICO

2.1. FLUTTER

Flutter é um *framework* de desenvolvimento de Interface de Usuário (UI) de código aberto, lançado pelo Google em 2015. Essa ferramenta oferece suporte à criação de aplicativos multiplataforma para uma variedade de sistemas operacionais, incluindo *World Wide Web* (Web*)*, *Android*, *iOS*, *Windows*, *Macintosh (Mac)* e *Linux*. O grande diferencial do Flutter está na capacidade de compilar aplicativos nativos a partir de uma única base de código (Flutter, 2024).

Baseado na linguagem de programação *Dart*, o Flutter proporciona um ambiente de desenvolvimento moderno e seguro. O Dart é compilado em código de máquinas *Advanced RISC Machine* (ARM) ou Intel, garantindo eficiência e desempenho nas aplicações desenvolvidas com Flutter (Flutter, 2024).

2.1.1. Funcionamento Flutter

O funcionamento do Flutter é baseado em alguns conceitos:

1. *Widgets* (elementos de interface) como blocos de construção:

   No Flutter, tudo é um widget. Desde os elementos básicos da interface (como botões e caixas de texto) até layouts complexos, todos são widgets. Essa abordagem facilita a criação e personalização da interface do aplicativo;

2. Renderização direta:

   O Flutter não utiliza componentes nativos (como *Views* (visualizações) no Android ou *UIViews* no iOS). Em vez disso, ele renderiza diretamente na tela usando a biblioteca Skia (SKIA, 2025). Isso permite maior controle sobre a aparência e o desempenho dos elementos da interface;

3. *Hot Reload* (recarga instantânea):

    O recurso de Hot Reload é uma das joias do Flutter. Ele permite que você veja as alterações no código imediatamente, sem precisar reiniciar o aplicativo. Isso agiliza muito o desenvolvimento e facilita a correção de erros;

4. Linguagem Dart:

    O Flutter utiliza a linguagem de programação Dart. Embora o Dart não seja tão amplamente conhecido quanto outras linguagens, ele foi projetado especificamente para o desenvolvimento de aplicativos e tem algumas características interessantes;

    - Compilação *Just-In-Time* (JIT):

        O Dart é compilado em tempo de execução, o que significa que você pode fazer alterações no código e ver os resultados imediatamente durante o desenvolvimento;

    - Eficiência e Desempenho:

        O Dart é compilado em código de máquinas ARM ou Intel, o que garante um bom desempenho nas aplicações desenvolvidas com Flutter.

2.2. JWT

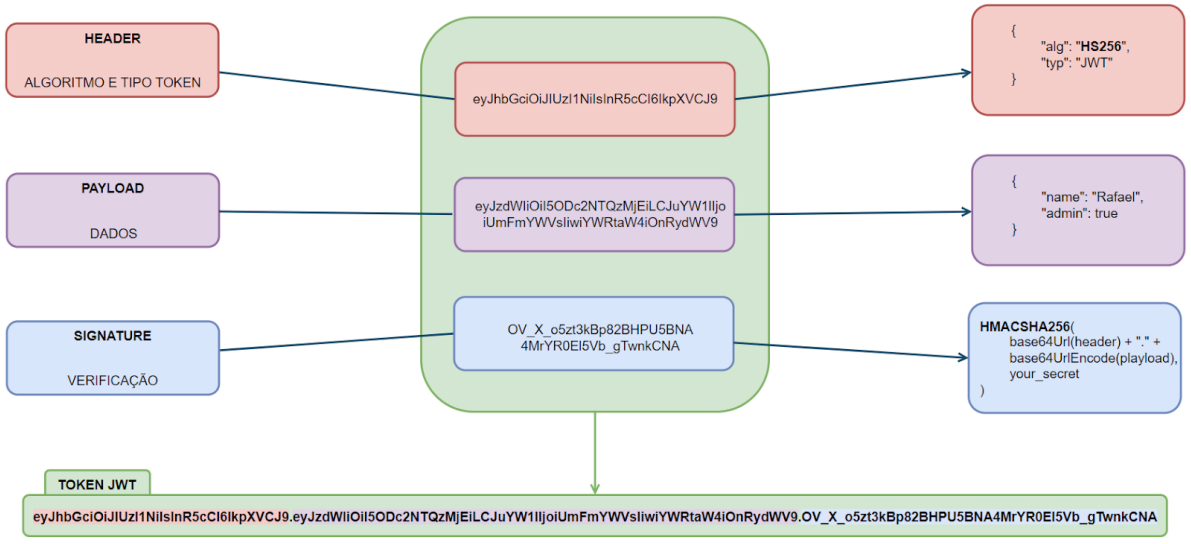
O *JSON Web Token* (JWT) é um padrão aberto definido pela *Request for Comments* (RFC) 7519, publicado pelo *Internet Engineering Task Force* (IETF) (JONES, 2015). Esse formato de *token* (credencial) é utilizado para a transmissão de informações de forma segura entre diferentes partes, como aplicativos, APIs e servidores, sendo aplicado principalmente em processos de autenticação e autorização (JONES, 2015).

Os JSON Web Tokens (JWTs) podem conter informações como nome de usuário, email e outros dados pertinentes, transmitidos entre diferentes partes envolvidas em um processo de comunicação (JONES, 2015). Esses tokens são

assinados digitalmente, permitindo a verificação de autenticidade e a integridade das informações trocadas.

A assinatura digital dos JWTs pode ser realizadas por meio de diferentes algoritmos, como *Hash-based Message Authentication Code* (HMAC), *Rivest-Shamir-Adleman* (RSA) ou *Elliptic Curve Digital Signature Algorithm* (ECDSA). Além disso, os JTWs apresentam características de leveza e compacidade, favorecendo sua utilização em ambientes distribuídos que envolvem autenticação e controle de acesso (JONES, 2015).

Figura 1 – Representação Token JWT



Fonte: Figura do Autor

Os JWTs são compostos por três partes: *Header* (cabeçalho), *Payload* (carga útil), *Signature* (assinatura), sendo que cada uma desempenha uma função específica:

1. Cabeçalho (*Header*):
    - Este segmento contém informações sobre o processamento do token. Trata-se de um objeto JavaScript Object Notation (JSON), geralmente com dois campos. O campo typ indica que se trata de um JWT, enquanto o alg especifica o algoritmo criptográfico usado para a assinatura, como HMAC SHA256 ou RSA.

2. Carga útil (*Payload*):
    - Esta seção corresponde aos dados transmitidos. Também em formato JSON, ela pode incluir informações como um identificador de usuário (id), um nome (name) ou privilégios de *admin* (administrador), entre outros dados relevantes para a comunicação.

3. Assinatura (*Signature*):
    - A assinatura é a parte que garante a integridade do token. Ela é gerada a partir da combinação do cabeçalho, da carga útil e de uma chave secreta (ou um par de chaves pública/privada). O resultado é uma sequência criptografada que garante que o conteúdo do token não foi alterado.

A representação final do JWT é a concatenação desses três segmentos, separados por pontos: *header.payload.signature*. Essa estrutura é amplamente utilizada em processos de autenticação e autorização, garantindo que a comunicação entre o cliente e o servidor seja segura e confiável.

2.3. API REST

Uma API (*Application Programming Interface*) é um conjunto de regras que definem como diferentes aplicativos ou dispositivos podem se comunicar entre si. Uma API REST é uma API baseada nos princípios de arquitetura *Representational State Transfer* (REST), também conhecidas como API RESTful (API REST, 2024).

O conceito de REST foi introduzido pela primeira vez em 2000 pelo cientista da computação Dr. Roy Fielding em sua dissertação de doutorado intitulada "*Architectural Styles and the Design of Network-based Software Architectures*" (Fielding, R. T., 2000). Desde então, as APIs REST têm se destacado por sua flexibilidade e liberdade oferecidas aos desenvolvedores. Essa adaptabilidade é uma das razões pelas quais as APIs REST se tornaram um método amplamente utilizado para comunicação em arquiteturas baseadas em *microservices* (microsserviços) (API REST, 2024).

As APIs REST utilizam um conjunto de métodos *Hypertext Transfer Protocol* (HTTP) para acessar os recursos disponíveis. Os métodos HTTP mais comuns são GET, POST, PUT e DELETE. O método GET é usado para obter um recurso, o método POST é usado para criar um recurso, o método PUT é usado para atualizar um recurso e o método DELETE é usado para excluir um recurso (API REST, 2024).

As APIs REST representam uma abordagem eficiente e flexível para conectar aplicativos e serviços. Elas desfrutam de ampla adoção em arquiteturas de microservices, permitindo que os diversos componentes de um sistema se comuniquem de maneira independente. Além disso, sua simplicidade e facilidade de uso as tornam uma escolha popular, especialmente no contexto de aplicativos móveis (API REST, 2024).

2.4. NODE.JS

O Node.js é um ambiente de execução de código aberto que possibilita a execução de JavaScript fora do navegador. Por ser construído por motor de JavaScript V8 do Google, essa plataforma oferece um desempenho rápido e eficiente (NODE.JS, 2024).

Frequentemente, desenvolvedores optam pelo Node.js para criar aplicações que exigem alta performance e escalabilidade. Isso se deve ao fato de que ele permite a execução do código JavaScript no lado do servidor, o que torna as aplicações mais rápidas e responsivas em comparação com outras tecnologias (NODE.JS, 2024).

Além disso, a versatilidade do Node.js como plataforma permite o desenvolvimento de uma ampla gama de aplicações, incluindo aquelas para web, dispositivos móveis desktop e até mesmo aplicações *Internet of Things* (IoT) (NODE.JS, 2024).

2.5. EXPRESS

Express ou Express.js é um framework Node.js que fornece recursos para a criação de servidores de API REST. Reconhecido por sua leveza, o Express oferece

facilidade e rapidez na criação de utilitários HTTP e *middleware*, sem comprometer os recursos do Node.js (Express, 2024). Ele se caracteriza por sua estrutura modular e oferece uma interface minimalista para a criação e gestão de servidores HTTP. Entre as suas funcionalidades, destacam-se a definição de rotas, o uso de *middleware* e a manipulação de requisições e respostas HTTP.

1. Definição de Rotas

   Uma das funcionalidades do Express.js é a definição de rotas, que permite mapear as *Uniform Resource Locator* (URL) para funções específicas. Com isso, pode-se organizar o fluxo das requisições, direcionando cada caminho a um controlador responsável por determinada ação. Por exemplo, uma requisição para uma rota */pacientes* pode ativar uma função que retorna uma lista de pacientes.

2. Middleware

   O conceito de middleware no Express.js permite a inserção de funções intermediárias que são executadas antes ou depois do processamento das requisições. Esses middlewares podem ser utilizados para uma variedade de tarefas, como validação de dados, autenticação de usuários ou registro de *logs* (registros). Isso possibilita a criação de camadas intermediárias que gerenciam a lógica de negócio e a segurança da aplicação de forma organizada.

3. Manipulação de Requisições e Respostas

   No Express.js, as requisições HTTP (GET, POST, PUT, DELETE) podem ser capturadas e processadas facilmente. A partir dessas requisições, o desenvolvedor tem acesso a parâmetros de URL, corpo de requisição e cabeçalhos, podendo tomar decisões com base nesses dados. Após o processamento, o servidor responde ao cliente com o conteúdo necessário, seja em formato JSON, *HyperText Markup Language* (HTML) ou outro, conforme a necessidade da aplicação.

4. Modularidade

A modularidade é um dos aspectos observados no Express.js, o que permite a organização do código em diferentes arquivos ou módulos. Essa separação facilita a manutenção e o entendimento do código, permitindo que cada parte do sistema seja responsável por uma função específica. A modularidade contribui para a construção de aplicações mais organizadas e com maior facilidade de expansão.

2.6. MYSQL

MySQL é um Sistema de Gerenciamento de Banco de Dados (SGBD) de código aberto amplamente utilizado por seu desempenho, confiabilidade e robustez. Este servidor de banco de dados *Structured Query Language* (SQL) é aplicado para armazenar dados de aplicações *web*, APIs e outros serviços que requerem persistência de dados (MYSQL, 2024).

Como um Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR), o MySQL organiza e armazena dados de maneira estruturada. Nesse modelo, os dados são distribuídos em tabelas que seguem um formato de linhas e colunas, onde cada coluna representa um tipo de dado e cada linha um registro único. A abordagem relacional facilita o estabelecimento de conexões lógicas entre diferentes conjuntos de dados, ajudando a organizar e acessar as informações de forma eficiente (MYSQL, 2024).

Além disso, o MySQL oferece suporte a diversos mecanismos de armazenamento, como InnoDB e MyISAM, que permitem diferentes níveis de desempenho e integridade transacional, dependendo das necessidades da aplicação. A flexibilidade em termos de mecanismos de armazenamento, combinada com a capacidade de realizar consultas complexas e transações seguras, faz com que o MySQL seja uma escolha frequente para desenvolvedores que buscam soluções robustas e escaláveis para o gerenciamento de grandes volumes de dados (MYSQL, 2024).

2.7. SQFLITE

A biblioteca SQFLite (Tekartik, 2025) permite a implementação do banco de dados *SQLite* (SQLite, 2025) em aplicativos Flutter, oferecendo uma estrutura robusta para o armazenamento local de informações diretamente no dispositivo móvel. Essa solução é ideal para situações que exigem o funcionamento offline, o que é comum em áreas com conectividade limitada (Tekartik, 2025).

O SQLite, um sistema de gerenciamento de banco de dados relacional de código aberto, se diferencia por não necessitar de um servidor de banco de dados a parte para operar (SQLite, 2025). Por armazenar o banco de dados em um único arquivo no dispositivo, ele simplifica a configuração e o gerenciamento dos dados em aplicações móveis. A integração do SQLite ao ambiente Flutter é possível por meio da biblioteca SQFLite, que disponibiliza as funcionalidades do banco de dados para o desenvolvimento de aplicativos (Tekartik, 2025).

No contexto específico de aplicações móveis, o SQFLite inclui funcionalidades suportam diversas demandas de armazenamento e manipulação de dados, como:

- **Transações e Operações em Lote:** O SQFLite possibilita a execução de transações e operações em lote, tramitando múltiplas instruções SQL como uma única operação atômica. Esse recurso é fundamental para garantir a consistência dos dados, especialmente em cenários de operação offline, onde falhas de conexões podem interromper processos;
- **Gerenciamento de Versões:** O suporte ao controle de versões no SQFLite facilita a atualização do esquema do banco de dados, caso sejam necessárias mudanças, como a adição de novas tabelas ou colunas. Esse processo de migração é essencial para a evolução contínua das aplicações;
- **Armazenamento Local e Portabilidade:** Semelhante ao SQLite, o SQFLite armazena todos os dados em um único arquivo, o que atende às necessidades de armazenamento local. Essa estrutura leve e compacta é compatível com aplicações que operam offline e exigem uma arquitetura de dados eficiente;
- **Princípios ACID para Persistência de Dados:** O SQFLite segue os princípios ACID (Atomicidade, Consistência, Isolamento e Durabilidade),

assegurando que as transações mantenham a integridade dos dados, mesmo em cenários de uso offline.

2.8. *SWAGGER*

O Swagger é uma ferramenta de documentação de APIs que oferece a geração de documentação interativa. Com uma interface intuitiva, o Swagger permite que desenvolvedores compreendam, testem e interajam com a API de forma eficiente e amigável (Swagger, 2024).

O Swagger possui uma variedade de recursos que facilitam a criação, a visualização e a manutenção da documentação de APIs. Alguns deles incluem (Swagger, 2024):

Swagger Editor:

- Permite criar e editar definições de API diretamente no navegador;
- Oferece *feedback* (retorno) em tempo real e autocompletar de sintaxe.

Swagger UI:

- Visualiza e interage com os recursos da API sem a necessidade de implementação lógica;
- É uma interface interativa para explorar *endpoints* (pontos de extremidade), parâmetros e respostas;
- Ótimo para compartilhar a documentação com toda a equipe de desenvolvedores e consumidores externos.

Swagger Autogen:

- Gera bibliotecas de clientes, *stubs* e documentação de API a partir de definições diretas no código;
- Realiza a geração automática, reconhecendo endpoints e parâmetros.

Em resumo, o Swagger é uma ferramenta para quem trabalha com APIs, tornando o processo de documentação mais eficiente e colaborativo.

2.9. Paradigma de Programação Orientada a Objetos (POO)

O desenvolvimento de *software* é extremamente amplo. Nesse mercado, existem diversas linguagens de programação, que seguem diferentes tipos de paradigmas. Um desses paradigmas é a Orientação a Objetos (POO), que atualmente é o mais difundido entre todos. Isso acontece porque se trata de um padrão que tem evoluído muito, principalmente em questões voltadas à segurança e reaproveitamento de código, o que é muito importante no desenvolvimento de qualquer aplicação moderna (DevMedia, 2014).

Esse padrão se baseia em quatro pilares que serão brevemente abordados a seguir. Além disso, a POO traz diversas vantagens em sua utilização, as quais serão apresentadas a seguir.

2.9.1. Abstração

No desenvolvimento de software, a abstração pode ser comparada à forma como simplificam informações complexas no cotidiano. O foco é se concentrar nos aspectos essenciais de um objeto, ignorando características secundárias para alcançar um objetivo específico. Conforme a definição apresentada no Dicionário Michaelis (Carvalho, 2020), a abstração consiste em "isolar características de um objeto considerando aquelas que tenham em comum certos grupos de objetos". Esse conceito permite maior flexibilidade na modelagem, pois o desenvolvimento se concentra em uma versão mais genérica e adaptável do objeto. A partir desse modelo abstrato inicial, podem ser criadas diversas variações mais específicas, um processo que ilustra os princípios de generalização e especialização (Carvalho, 2020).

Além da abstração, a reutilização de código é um princípio fundamental da Programação Orientada a Objetos (POO). A repetição de código é considerada uma prática ineficiente, pois pode levar a inconsistências, fragilidade e dificultar a manutenção do software. A POO oferece conceitos como herança e associação para facilitar a reutilização. Na herança, uma nova classe pode ser criada a partir de uma já existente, aproveitando seus atributos e comportamentos. Já na associação, uma classe pode depender de outra para executar funções, promovendo

modularização e organização (Carvalho, 2020). O uso adequado desses conceitos resulta em um código mais intuitivo, robusto e de fácil manutenção, contribuindo para o desenvolvimento de aplicações escaláveis e sustentáveis (Carvalho, 2020).

2.9.2. Encapsulamento

O encapsulamento na Programação Orientada a Objetos (POO) permite esconder a implementação interna de um objeto, expondo apenas o que é necessário para quem o utiliza. Esse princípio garante que um método ou atributo possa ser protegido, sem que sua forma de desenvolvimento afete o uso externo. O foco está no resultado final, e não na complexidade do processo interno. Assim, um usuário pode continuar utilizando uma funcionalidade sem ser afetado caso seu comportamento interno seja modificado. O encapsulamento, portanto, protege a integridade dos dados e evita resultados inesperados (CARVALHO, 2020).

2.9.3. Herança

A herança, um conceito da Programação Orientada a Objetos (POO), permite a criação de novas classes (subclasses) com base em classes já existentes (superclasses), herdando atributos e métodos previamente definidos. Esse princípio, que facilita a reutilização de código, deve ser utilizado com moderação para evitar complexidade excessiva. O conceito envolve generalização e especialização: as classes mais genéricas servem de base para classes mais específicas (CARVALHO, 2020). A herança só deve ser utilizada quando há uma relação de "é um tipo de", como um funcionário "é um tipo de" pessoa. O uso inadequado pode comprometer a lógica do sistema (CARVALHO, 2020).

2.9.4. Polimorfismo

O polimorfismo, na Programação Orientada a Objetos (POO), é o princípio que permite que um mesmo método se comporte de maneiras diferentes, dependendo do objeto que o executa. Esse conceito garante flexibilidade, permitindo que subclasses modifiquem o comportamento herdado de uma superclasse (CARVALHO, 2020). A grande vantagem é permitir que diferentes objetos utilizem

um mesmo método, tornando o código mais dinâmico, reutilizável e de fácil manutenção (CARVALHO, 2020).

2.10. Diagrama de Classe

O diagrama de classe é uma das ferramentas mais importantes da *Unified Modeling Language* (UML), desempenhando um papel essencial na modelagem de sistemas orientados a objetos. Ele oferece uma representação visual da estrutura do sistema, detalhando as classes e os relacionamentos entre elas, como associações, generalizações e dependências (Fowler, 2014, p. 50).

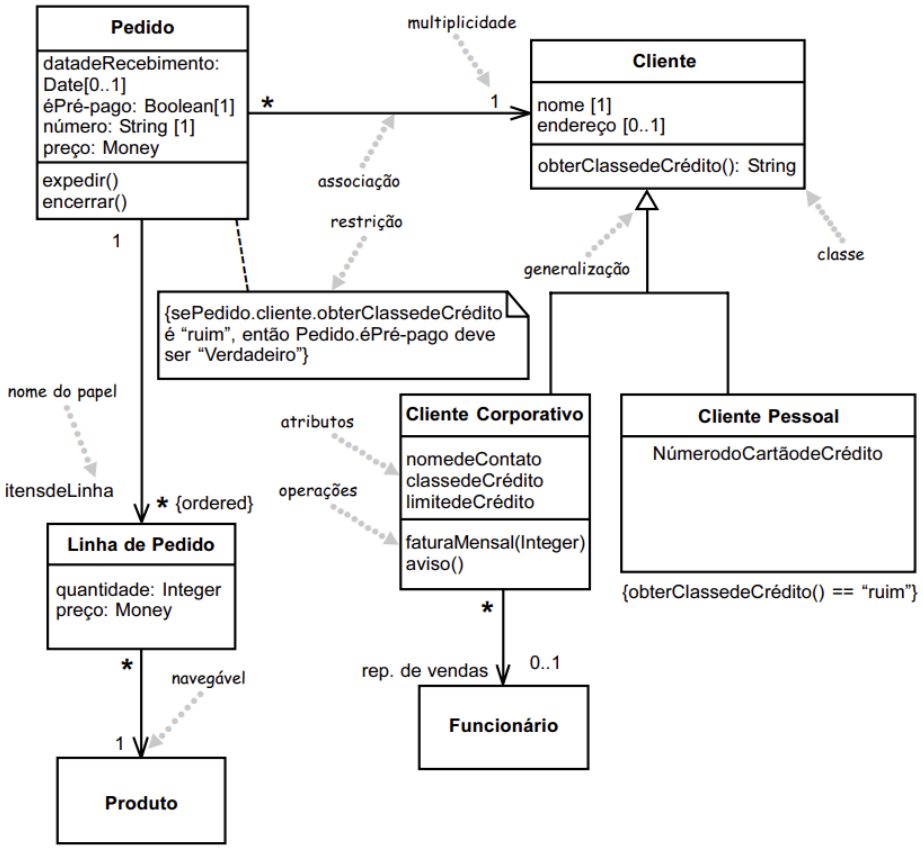
Essa abordagem gráfica permite compreender com clareza a arquitetura do software, destacando os atributos, métodos e interações entre as entidades. Facilita, ainda, a comunicação entre a equipe de desenvolvimento, atuando como uma documentação detalhada e precisa durante o processo (Fowler, 2014, p. 50).

Na Figura 2 apresenta um exemplo de diagrama de classe, criado por Fowler, (2014, p. 50). O modelo ilustra um sistema completo, representando elementos, como:

- Classes principais (Pedido, Cliente, Produto, entre outras);
- Relacionamentos entre elas (associação, generalização e restrição);
- Multiplicidade (exemplo: um Pedido pode conter várias Linhas de Pedido);
- Operações e atributos relevantes.

Essa estrutura evidencia como os sistemas podem ser organizados, integrando suas partes de forma lógica e funcional. O diagrama de classe é, portanto, uma ferramenta indispensável na construção de sistemas orientados a objetos.

Figura 2 – Exemplo de Diagrama de Classe



Fonte: Fowler, (2014, p. 53).

2.11. Casos de Uso (UC)

Na UML, os diagramas de caso de uso (UC) são utilizados para modelar o comportamento de um sistema e capturar os requisitos. Esses diagramas descrevem funções de alto nível, além de identificar as interações entre o sistema e seus agentes. Os casos de uso e os agentes demonstram o que o sistema faz e como os agentes utilizam, mas não detalham o funcionamento do sistema (IBM, 2025).

Os diagramas de casos de uso são ferramentas para ilustrar e definir o contexto e os requisitos de um sistema, seja ele completo ou das partes importantes. Um sistema complexo pode ser representado por um único diagrama ou por múltiplos diagramas que modelam os comportamentos individuais. A criação desses diagramas geralmente ocorre nas fases iniciais de um projeto e serve como referência durante todo o processo de desenvolvimento (IBM, 2025).

Os diagramas de caso de uso são úteis em diversas etapas do desenvolvimento de um sistema:

- **Definição do negócio:** Antes mesmo do início do projeto, eles ajudam a representar o funcionamento do negócio, permitindo uma visão clara e compartilhada sobre as funções, usuários e processos envolvidos para todos os interessados;

- **Levantamento de requisitos:** Na etapa de levantamento de requisitos, os diagramas facilitam o registro e a ilustração das funcionalidades que o sistema deve oferecer, auxiliando a comunicação com as partes interessadas;

- **Análise e *designer* (projeto):** Durante a de análise e design, os diagramas contribuem para a identificação das classes e estruturas necessárias no sistema, com base nos casos de uso e agentes;

- **Testes:** Na fase de testes, os diagramas de caso de uso servem como base para o planejamento dos testes, garantindo que todas as funções previstas sejam verificadas.

Os principais elementos presentes nesses diagramas incluem:

Casos de Uso

Representam as funções ou ações que o sistema executa para atingir os objetivos do usuário. Cada caso de uso deve proporcionar um benefício ou uma resposta concreta e útil para quem utiliza o sistema.

Agentes (Atores)

São as entidades externas que interagem com o sistema. Os agentes podem ser pessoas, outras organizações, equipamentos ou sistemas externos que participam do processo.

Subsistemas

Na UML, subsistemas representam componentes de maior escala que funcionam como blocos independentes dentro do sistema. Eles são úteis para organizar o projeto em módulos, sendo empregados em diversos tipos de diagramas, incluindo os de caso de uso, classes e componentes.

Relacionamentos

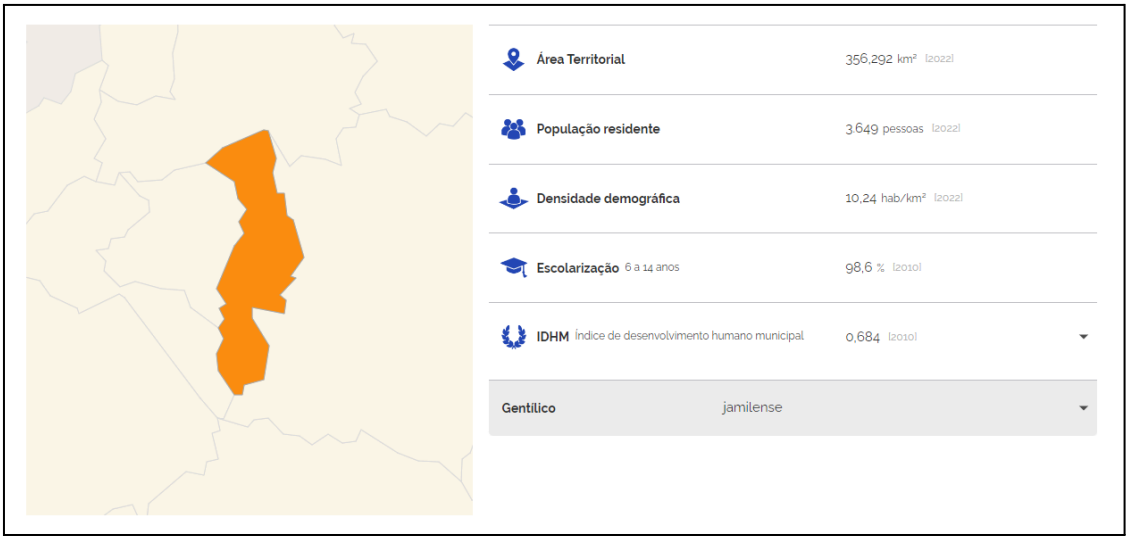
São as conexões entre elementos do diagrama, que definem como casos de uso, agentes e subsistemas se ligam e interagem, estruturando o comportamento geral do sistema modelado.

3. ANÁLISE ESTRUTURAL E FUNCIONAL DO SISTEMA

O aplicativo “Saúde do Interior” foi desenvolvido com o intuito de atender às necessidades das unidades de saúde localizadas em regiões rurais do país. Sua criação teve como objetivo principal auxiliar na gestão dos pacientes residentes na zona rural, onde o acesso aos serviços de saúde pública é limitado.

A concepção do aplicativo surgiu após uma visita a uma unidade de saúde de Professor Jamil, interior de Goiás. Na ocasião, foram identificadas as dificuldades enfrentadas para realizar visitas regulares às famílias da zona rural. Conforme dados do Instituto Brasileiro de Geografia e Estatística (IBGE), a unidade de saúde atende uma população de aproximadamente 3.649 habitantes, distribuídos em uma área de cerca de 356,292 km² (IBGE, 2022).

Figura 3 – Dados IBGE cidade de Professor Jamil - GO



Fonte: IBGE (2022)

As dificuldades para as visitas regulares às famílias incluíam:

- A distância entre as comunidades rurais e a unidade de saúde;
- A dificuldade do profissional em localizar os grupos familiares;
- A dependência de registros físicos, como fichas de papel, para manter os dados.

O aplicativo “Saúde do Interior” foi projetado para superar essas dificuldades, permitindo que os profissionais de saúde realizem visitas regulares às famílias da zona rural, mesmo em locais sem acesso a internet. Entre as principais funcionalidades do aplicativo, pode-se destacar as seguintes:

- **Cadastramento de Regiões e Grupos Familiares:** Os profissionais podem cadastrar as regiões de atendimento conforme as divisões geográficas do município e registrar os grupos familiares e seus integrantes, organizando as informações de forma clara e acessível;
- **Monitoramento de Atendimentos:** O aplicativo facilita a visualização dos grupos ou pacientes que ainda não foram atendidos, permitindo que os profissionais organizem as visitas de maneira eficiêncintes e garanta o acompanhamento de todas as famílias;
- **Registro de Informações dos Pacientes:** Durante as visitas, o profissional pode registrar informações relevantes, como histórico de saúde e observações específicas, diretamente no aplicativo.

O aplicativo proporciona uma maneira mais prática e eficiente de gerenciar as informações relacionadas ao atendimento nas áreas rurais. Ele possibilita o acompanhamento do progresso das visitas de forma organizada e detalhada, assegurando que nenhum paciente ou grupo familiar seja negligenciado. A gestão centralizada e a acessibilidade dos dados em locais remotos oferecem uma melhora significativa no processo de atendimento, pois os profissionais podem consultar as informações a qualquer momento, o que contribui para um cuidado de saúde mais rápido e eficaz.

Espera-se que, ao utilizar o aplicativo, as equipes de saúde otimizem o tempo em campo e aumentem a cobertura de atendimento nas regiões rurais. Isso pode ampliar o acesso aos serviços de saúde para populações que, de outra forma, enfrentam barreiras para receber cuidados regulares. Além disso, a ferramenta ajuda a reduzir a dependência de registros manuais, promovendo uma gestão mais eficiente e sustentável a longo prazo.

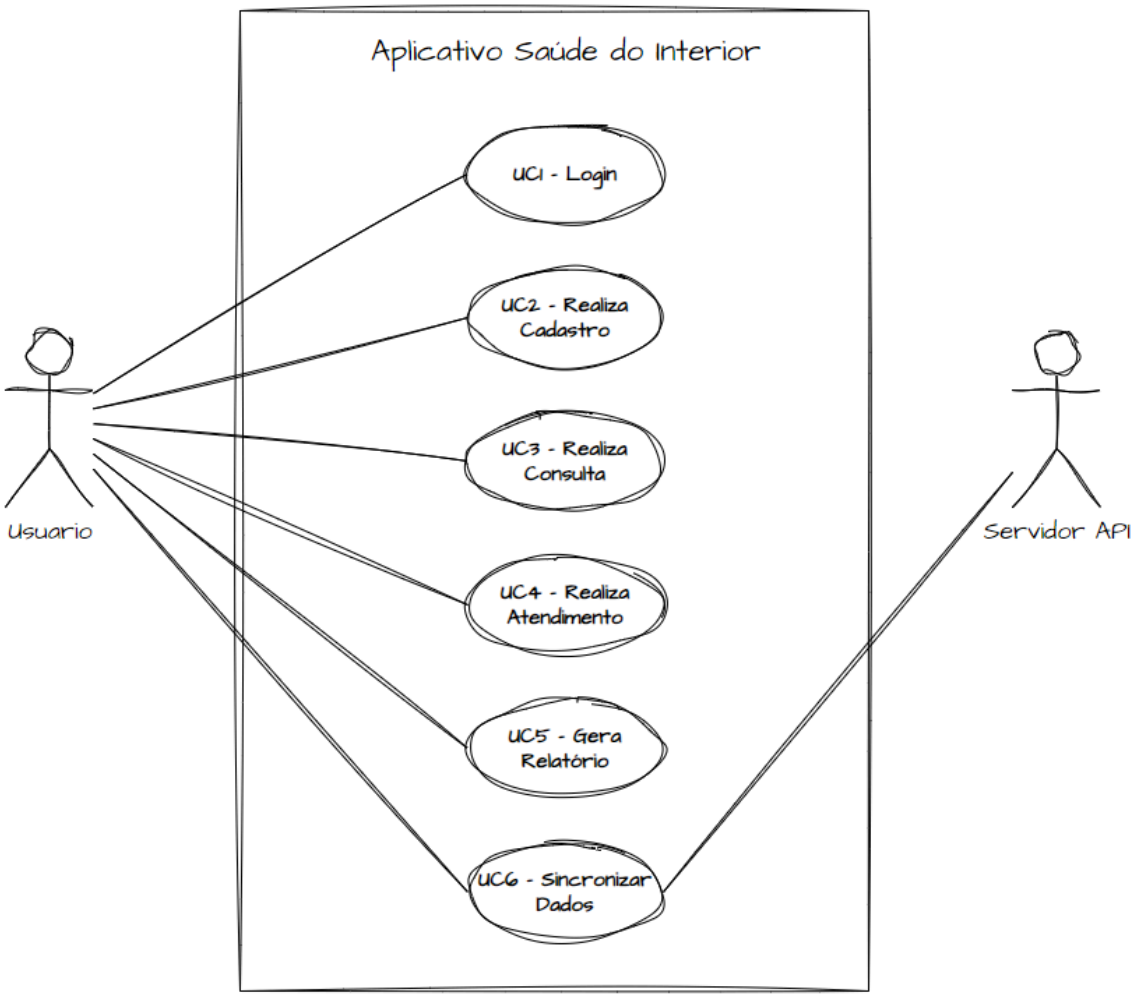
3.1. APLICATIVO SAÚDE DO INTERIOR

Nas últimas décadas, os sistemas de software passaram a ocupar um espaço cada vez mais relevante em diversas áreas, sendo aplicados em contextos como processos comerciais, sistemas industriais, serviços públicos, entre outros. À medida que esses sistemas se tornam mais complexos, uma compreensão mais clara de sua estrutura, funcionalidades e formas de interação pode ser útil. Essa visão ampla contribui para o planejamento, o desenvolvimento e uso alinhado às necessidades de cada projeto. Dentro desse cenário, a representação geral de um sistema serve como um recurso importante para visualizar suas partes principais e como elas se relacionam.

Com base nisso, será apresentada a seguir uma visão geral do sistema “Saúde do Interior”, com o intuito de oferecer uma perspectiva simplificada de seu funcionamento. Para isso, serão utilizados casos de uso que ajudam a ilustrar as principais interações dentro do sistema.

O sistema “Saúde do Interior” foi idealizado com a proposta de auxiliar no gerenciamento de atividades de saúde em áreas rurais, disponibilizando uma plataforma de apoio aos profissionais que atuam diretamente nessas regiões. A Figura 4 traz um diagrama de caso de uso que mostra, de forma esquemática, como ocorrem as interações entre os principais atores envolvidos: o usuário, a API e o aplicativo.

Figura 4 – Diagrama de caso de uso aplicativo “Saúde do Interior”



Fonte: Figura do Autor

3.2. ESTRUTURA DE DADOS

A estrutura de dados no projeto "Saúde do Interior" foi organizada para facilitar o registro e a gestão de informações referentes à saúde nas zonas rurais. Os tipos de dados estão divididos em quatro categorias principais: Regiões, Grupos, Pacientes e Serviços, cada uma com funções específicas no sistema. A seguir, será apresentada uma descrição dessas categorias e a justificativa para sua inclusão, baseando-se nas necessidades observadas em contextos de saúde pública em áreas rurais.

3.2.1. Justificativa da Estrutura de Dados

Cada uma das categorias de dados foi planejada com base nas necessidades identificadas em contextos de saúde pública em áreas rurais do município de Professor Jamil Goiás. O cadastro de Regiões fornece uma base geográfica para planejar e organizar as atividades, enquanto o registro dos Grupos e Pacientes permite uma organização das famílias e indivíduos atendidos. A categoria Serviços, por sua vez, facilita o monitoramento das atividades dos agentes de saúde, fornecendo informações sobre os atendimentos realizados e permitindo ajustes nas operações conforme a demanda local.

Essa organização de dados tem como objetivo fornecer um suporte eficiente ao sistema de saúde, permitindo que as ações sejam melhor planejadas e executadas em áreas rurais, levando em conta as características locais e as necessidades específicas da população.

3.2.2. Regiões

A categoria Regiões registra as divisões geográficas da zona rural dos municípios atendidos. Essas divisões são realizadas pelos agentes de saúde com base em diferentes critérios, como:

- **Aspectos físicos**: Características do relevo ou da hidrografia;
- **Fatores sociais e econômico:** Relacionados à distribuição populacional e às condições de vida das pessoas em determinada área;
- **Critérios políticos ou administrativos:** Orientam a divisão territorial para fins de planejamento.

A organização por Regiões auxilia a elaboração a alocação de recursos e o planejamento das visitas. O mapeamento regional facilita a gestão das intervenções de saúde, permitindo identificar, por exemplo, áreas mais distantes ou de difícil acesso, onde o planejamento de deslocamento pode ser necessário.

3.2.3. Grupos

A categoria Grupos representa um conjunto de pessoas, geralmente familiares ou moradores de uma mesma residência. Os dados cadastrados incluem:

- Localização geográfica da residência do grupo;
- Informações sobre os integrantes, como idade, gênero e outros dados relevantes.

A organização por Grupos oferece uma visão mais prática das famílias atendidas, facilitando o registro das informações e o acompanhamento das condições de saúde de forma coletiva. Isso é especialmente relevante em comunidades rurais, onde o atendimento pode ser feito em visitas domiciliares abrangendo toda a família.

3.2.4. Pacientes

A categoria Pacientes contém os dados individuais dos integrantes dos grupos. Cada paciente é cadastrado de forma separada, incluindo informações como:

- Dados pessoais (nome, idade e sexo);
- Histórico de saúde, onde são registradas observações feitas pelos agentes durante as visitas.

O registro individualizado dos pacientes possibilita o acompanhamento contínuo das condições de saúde de cada pessoa. Esse acompanhamento pode ser útil para adaptar o atendimento às necessidades específicas de cada indivíduo, garantindo que, em visitas subsequentes, o histórico de saúde esteja facilmente acessível para os profissionais.

3.2.5. Serviços

A categoria serviços abrange os dados relativos aos atendimentos realizados pelos agentes de saúde. Cada atendimento é registrado em um formulário pré-cadastrado, com informaçoes sobre:

- Tipo de atendimento (consultas, vacinas ou orientações);
- Data e local do atendimento;
- Observações adicionais.

A inclusão da categoria serviços permite que os gestores e profissionais de saúde monitorem as atividades realizadas e avaliem quais serviços estão sendo prestados em cada região. Dessa forma, pode-se analisar a cobertura dos serviços de saúde e planejar melhorias conforme necessário.

3.3. FLUXO DE DADOS

O fluxo de dados no aplicativo "Saúde do Interior" é projetado para garantir a coleta, o processamento, o armazenamento e utilização eficientes das informações dos pacientes. Com um fluxo bem definido e seguro, o sistema proporciona uma gestão eficaz dos dados, contribuindo para a melhoria dos serviços de saúde em áreas rurais. Esse tópico detalha como os dados são manipulados e gerenciados dentro do sistema, proporcionando uma visão clara de como o aplicativo opera e como as informações são tratadas ao longo de seu ciclo de vida.

3.3.1. Captura de Dados

O processo inicia com a captura de dados, que ocorre quando os profissionais de saúde inserem informações no aplicativo durante as visitas domiciliares. Isso inclui dados sobre pacientes, grupos familiares, registros de atendimentos e detalhes de localização. A captura de dados é feita por meio de formulários eletrônicos no aplicativo, que permitem a coleta eficiente de informações essenciais.

3.3.2. Processamento de Dados

Após a captura, os dados são processados pelo sistema. O processamento envolve a validação e a organização das informações para garantir a precisão e a integridade. No "Saúde do Interior", isso inclui a verificação de dados inseridos, a associação de registros com os pacientes e grupos correspondentes, e a preparação dos dados para análise posterior.

3.3.3. Armazenamento de Dados

Os dados processados são armazenados no banco de dados MySQL, tanto no banco de dados offline do aplicativo quanto na API online. O armazenamento inclui dados de pacientes, grupos familiares, registros de atendimentos e informações de localização. A estrutura relacional do banco de dados organiza esses dados em tabelas inter-relacionadas, facilitando a recuperação e o gerenciamento eficiente das informações.

3.3.4. Acesso e Consulta de Dados

Os profissionais de saúde acessam e consultam os dados por meio da interface do aplicativo. O sistema permite buscas e consultas para visualizar informações sobre pacientes, grupos familiares e registros de atendimentos. As funcionalidades de busca são projetadas para fornecer resultados rápidos e precisos, facilitando o trabalho dos profissionais e a gestão das visitas.

3.3.5. Atualização de Dados

Durante as visitas e interações subsequentes, os dados podem ser atualizados. O aplicativo permite que os profissionais de saúde modifiquem informações existentes, como dados pessoais dos pacientes e os detalhes dos atendimentos realizados. Essas atualizações são sincronizadas com o banco de dados, garantindo que todas as informações estejam atualizadas e refletindo o estado mais recente.

3.3.6. Sincronização de Dados

Para garantir a consistência dos dados entre o banco de dados offline do aplicativo e o banco de dados online na API, é necessário uma sincronização periódica. Esse processo atualiza o banco de dados online com as informações coletadas offline e vice-versa, mantendo a integridade e a atualidade dos dados em ambos os ambientes.

3.3.7. Análise e Relatórios

Com os dados armazenados e atualizados, o sistema gera relatórios e análises sobre a saúde da população atendida. Esses relatórios podem incluir dados agregados sobre atendimentos, condições de saúde e outros indicadores importantes. A análise desses dados ajuda na tomada de decisões e no planejamento de estratégias de saúde.

3.3.8. Segurança e Proteção de Dados

Durante todo o fluxo de dados, a segurança e a proteção das informações são prioritárias. Medidas de segurança são implementadas para proteger os dados contra acesso não autorizado e garantir a privacidade das informações dos pacientes. Isso inclui autenticação de usuários, criptografia de dados e controles de acesso apropriados.

3.4. SEGURANÇA E PRIVACIDADE

3.4.1. Proteção de Dados Pessoais

A proteção de dados pessoais é uma prioridade no aplicativo "Saúde do Interior". Medidas são implementadas para garantir que as informações dos pacientes sejam mantidas em sigilo e sejam acessíveis apenas por usuários autorizados. O sistema adota boas práticas de segurança para proteger dados sensíveis contra acessos não autorizados e vazamentos.

3.4.2. Autenticação e Controle de Acesso

O sistema implementa mecanismos de autenticação robustos para garantir que apenas usuários autorizados possam acessar o aplicativo e as informações. Isso inclui autenticação por senha e gerenciamento de sessões para proteger as contas dos usuários. Os controles de acesso garantem que os profissionais de saúde só possam acessar os dados relevantes para suas funções e responsabilidades.

3.5. IMPLEMENTAÇÃO

Após uma análise minuciosa dos casos de uso do sistema "Saúde do Interior", que descrevem as interações entre os atores (usuário, servidor API e o aplicativo), a fase de implementação foi iniciada. Nesta etapa crucial, os conceitos e requisitos definidos nos casos de uso foram transformados em funcionalidades reais do sistema. A implementação requereu a aplicação prática de tecnologias e ferramentas específicas para garantir a eficiência, segurança e usabilidade do sistema.

Durante o desenvolvimento, a opção por seguir o paradigma de programação amplamente conhecido: a POO. Esse paradigma oferece uma abordagem estruturada e modular para o desenvolvimento de software, permitindo a criação de sistemas mais organizados, flexíveis e fáceis de manter.

3.5.1. Desafios e Soluções

Os desafios enfrentados durante o desenvolvimento e a implementação do "Saúde do Interior" foram abordados com uma combinação de soluções técnicas e operacionais. A capacidade de adaptar o sistema às necessidades específicas dos usuários e garantir sua funcionalidade e segurança foi de grande importância para o projeto. A abordagem para superar esses obstáculos contribuiu para a criação de um aplicativo voltado para atender às necessidades dos profissionais de saúde na zona rural.

Esse tópico oferece uma visão detalhada dos principais obstáculos encontrados e das estratégias adotadas para superá-los, destacando a abordagem prática e as soluções implementadas para garantir o desenvolvimento do sistema.

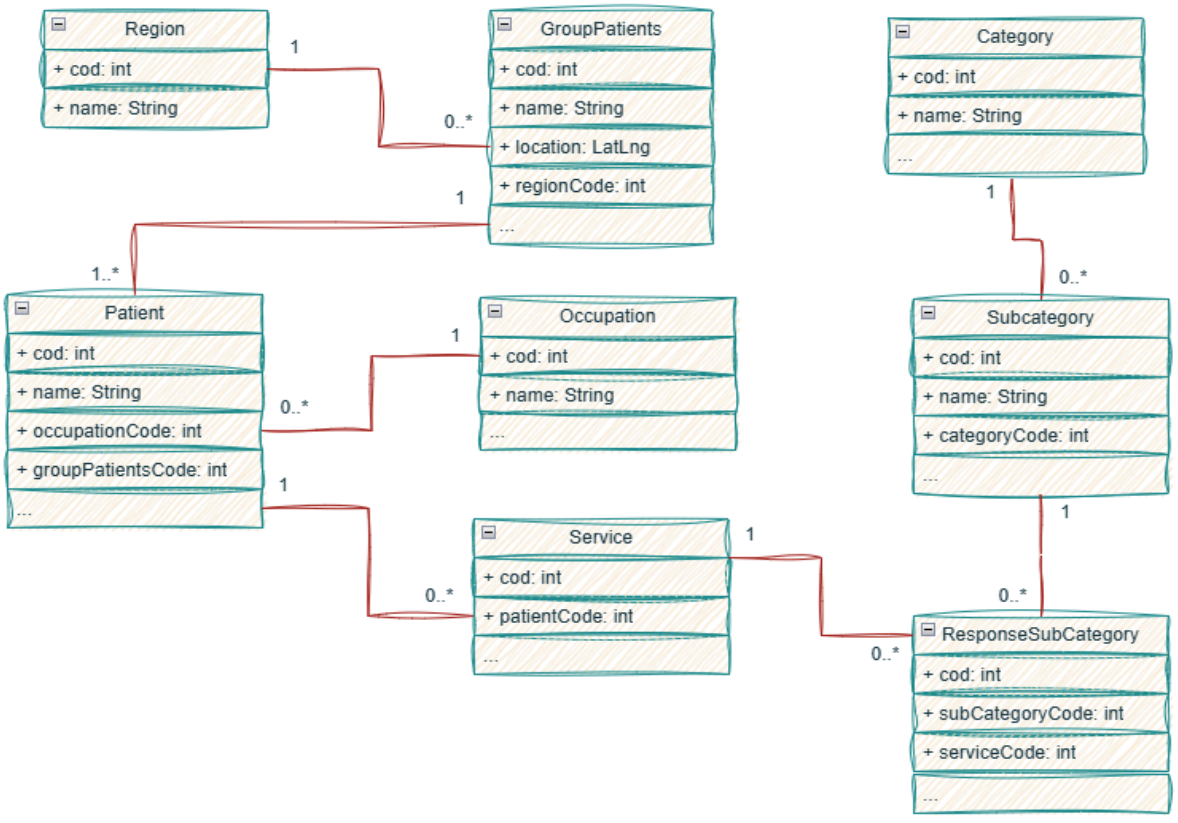
- **Integração com Sistemas Offline:** Garantir a funcionalidade completa do aplicativo em áreas sem conectividade à internet (offline) foi um desafio significativo. A solução envolveu o desenvolvimento de um banco de dados local e a implementação de mecanismos de sincronização de dados para garantir que as informações fossem atualizadas e consistentes quando a conectividade fosse restabelecida;
- **Desempenho e Escalabilidade:** O aplicativo precisava lidar com grandes volumes de dados provenientes das visitas domiciliares e do registro de informações. Para enfrentar isso, técnicas de otimização de banco de dados e estratégias de gerenciamento de dados foram implementadas para assegurar que o sistema mantivesse um desempenho eficiente;
- **Segurança dos Dados:** Proteger os dados sensíveis dos pacientes contra acessos não autorizados e vazamentos foi um desafio. A solução incluiu a implementação de criptografia, autenticação e controles de acesso para garantir a segurança das informações;
- **Manutenção dos *Tiles* do Mapa Offline:** Manter os *tiles* (mosaicos) do mapa disponíveis no aplicativo para compor o mapa enquanto está offline apresentou uma dificuldade significativa. Para enfrentar esse desafio, foi implementado um sistema de *cache* (armazenamento temporário) eficiente que armazena os tiles necessários para a navegação offline, garantindo que os dados geoespaciais estejam acessíveis mesmo sem conexão.

Adicionalmente, uma funcionalidade foi incorporada para limitar a área de alcance do mapa, permitindo aos usuários informar uma área específica para *download* (transferência) em cache. Essa abordagem otimiza o uso do espaço de armazenamento no dispositivo e melhora a eficiência do aplicativo ao reduzir a quantidade de dados armazenados localmente.

3.5.2. Representação de Associações

Na Figura 5, essa representação oferece uma visão detalhada da estrutura do desenvolvimento “Saúde do Interior”, permitindo uma compreensão abrangente das classes e seus relacionamentos dentro do sistema.

Figura 5 – Diagrama de Classe Estrutura do Desenvolvimento Saúde do Interior.



Fonte: Figura do Autor

A Figura 5, apresentada anteriormente, ilustra a estrutura de dados do sistema "Saúde do Interior". Esse diagrama de classes organiza os dados em entidades distintas e modela as relações entre elas, destacando atributos

importantes para cada classe e a cardinalidade dos relacionamentos. A seguir, uma descrição de cada classe e suas relações.

1. Classe Region

    - **Descrição:** Representa uma região geográfica, que pode corresponder a uma divisão territorial na zona rural do município;
    - **Atributos:**
        - cod (Identificador único da região);
        - name (Nome da região);
    - **Relacionamento:** Relaciona-se com a classe GroupPatients, indicando que uma região pode conter múltiplos grupos de pacientes, mas cada grupo está associado a uma única região (relação de 1 para 0..*).

2. Classe GroupPatients

    - **Descrição:** Representa um grupo de pacientes, organizado de acordo com sua localização geográfica dentro de uma região;
    - **Atributos:**
        - cod (Identificador único do grupo de pacientes);
        - name (Nome do grupo);
        - location (Localização do grupo, representada por coordenadas (LatLng));
        - regionCode (Código que referencia a região à qual o grupo pertence);
    - **Relacionamento:** Cada grupo de pacientes pertence a uma única região, estabelecendo uma ligação com a classe Region. Relaciona-se com a classe Patient, indicando que um grupo pode ter vários pacientes, mas cada paciente pertence a um único grupo (relação de 1 para 0..*).

3. Classe Patient

    - **Descrição:** Armazena dados pessoais e informações específicas de cada paciente;
    - **Atributos:**
        - cod (Identificador único do paciente);
        - name (Nome do paciente);
        - occupationCode (Código da ocupação do paciente, relacionando-o com uma ocupação específica);
        - groupPatientsCode (Código do grupo ao qual o paciente está associado);
    - **Relacionamento:** Cada paciente pertence a um grupo (GroupPatients) e possui uma única ocupação (Occupation). Relaciona-se com a classe (Service), indicando que um paciente pode ter múltiplos serviços de atendimento associados a ele, mas cada serviço se refere a um único paciente (relação de 1 para 0..*).

4. Classe Occupation

    - **Descrição:** Representa a ocupação profissional ou ocupação relevante do paciente;
    - **Atributos:**
        - cod (Identificador único da ocupação);
        - name (Nome da ocupação);
    - **Relacionamento:** Cada paciente possui uma única ocupação, e a ocupação pode estar associada a múltiplos pacientes (relação de 1 para 1..*).

5. Classe Service

    - **Descrição:** Armazena informações dos serviços de atendimento realizados para cada paciente, registrando informações coletadas durante visitas domiciliares;
    - **Atributos:**
        - cod (Identificador único do serviço);

    - patientCode (Código do paciente ao qual o serviço está associado);
- **Relacionamento:** Cada serviço refere-se a um único paciente, mas um paciente pode ter múltiplos serviços associados (relação de 1 para 0..*). Relaciona-se com a classe ResponseSubCategory, que armazena as respostas detalhadas por subcategoria de cada serviço.

6. Classe Category

- **Descrição:** Representa uma categoria de dados ou observações que podem ser associadas a uma subcategoria;
- **Atributos:**
    - cod (Identificador único da categoria);
    - name (Nome da categoria);
- **Relacionamento:** Cada categoria pode conter várias subcategorias, mas cada subcategoria pertence a uma única categoria (relação de 1 para 0..*).

7. Classe Subcategory

- **Descrição:** Representa subcategorias de dados ou observações, que podem detalhar mais uma categoria associada a um serviço;
- **Atributos:**
    - cod (Identificador único da subcategoria);
    - name (Nome da subcategoria);
    - categoryCode (Código da categoria à qual a subcategoria pertence);
- **Relacionamento:** Cada subcategoria pertence a uma única categoria, mas uma categoria pode conter várias subcategorias (relação de 1 para 0..*). Relaciona-se com a classe ResponseSubCategory, indicando que cada subcategoria pode ter várias respostas associadas a diferentes serviços.

8. Classe ResponseSubCategory

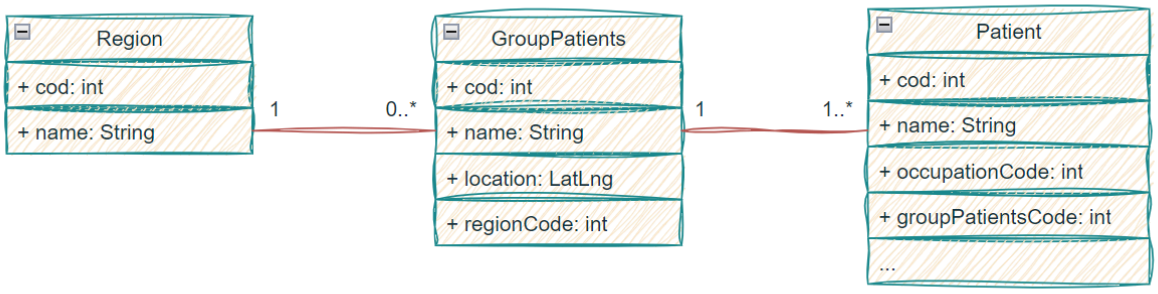
- **Descrição:** Armazena respostas ou observações específicas para uma subcategoria dentro de um serviço de atendimento;
- **Atributos:**
  - cod (Identificador único da resposta);
  - subCategoryCode (Código da subcategoria à qual a resposta está associada);
  - serviceCode (Código do serviço ao qual a resposta pertence);
- **Relacionamento:** Cada resposta está ligada a uma única subcategoria e a um único serviço. No entanto, uma subcategoria pode ter múltiplas respostas, cada uma associada a serviços diferentes (relação de 1 para 0..*).

Resumo das Relações

Este modelo de classes define um sistema estruturado, no qual:

1. A Region agrupa GroupPatients, que por sua vez reúne Patient;

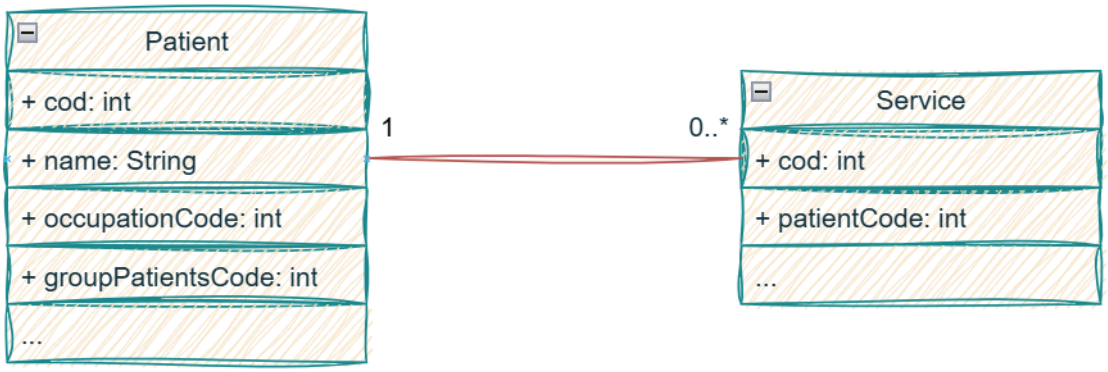
Figura 6 – Diagrama de Classe Region, GroupPatients e Patient



Fonte: Figura do Autor

2. Cada Patient pode ter múltiplos serviços de atendimento;

Figura 7 – Diagrama de Classe Patient e Service



Fonte: Figura do Autor

3. Os serviços são detalhados em ResponseSubCategory, com as observações organizadas por Category e Subcategory;
4. A Occupation fornece informações adicionais sobre cada paciente.

Esse diagrama de classes foi elaborado para atender às necessidades do projeto "Saúde do Interior", promovendo uma organização eficiente de dados de pacientes em um sistema de informações adequado para regiões rurais e de difícil acesso, onde a conectividade pode ser limitada.

3.5.3. Desenvolvendo classe

Neste tópico, serão apresentados trechos de códigos da aplicação que seguem as classes previamente discutidas. A implementação foi realizada utilizando Flutter, uma estrutura de desenvolvimento de aplicativos móveis amplamente adotada pela sua eficiência e versatilidade. O Dart foi a linguagem escolhida para codificar o sistema, proporcionando uma sintaxe limpa e expressiva para o desenvolvimento ágil de aplicações. Todos os trechos de código apresentados neste tópico estão relacionados à linguagem Dart.

Exemplos Práticos de Implementação

Após a análise dos casos de uso e a introdução aos conceitos de POO, o próximo passo é a implementação prática. Nessa etapa, selecionamos um caso de uso para demonstrar sua tradução em código, detalhando as interações entre os diversos componentes do sistema para garantir uma experiência satisfatória aos usuários.

A seguir, será demonstrada a implementação da classe GroupPatients como exemplo prático da aplicação dos conceitos abordados. Esta representação codificada ilustra como os elementos da classe são definidos e utilizados no contexto do sistema “Saúde do Interior”.

Figura 8 – Classe GroupPatients implementada em Dart

```dart
class GroupPatients {
  late int cod;
  late String name;
  late int? patientCount;
  late LatLng location;
  late int regionCode; //Associação de Grupo com Região
  late bool served;
  late int creationDate;
  late int lastDataChange;
  late bool deleted;

  GroupPatients({
    required this.cod,
    required this.name,
    this.patientCount,
    required this.location,
    required this.regionCode,
    required this.served,
    required this.creationDate,
    required this.lastDataChange,
    this.deleted = false,
  })

  // Outros métodos e funcionalidades da classe aqui...

}
```

Fonte: Figura do Autor

A classe GroupPatients, demonstrada anteriormente no Figura 8, representa um grupo de pacientes e possui atributos como códigos, nome, quantidade de pacientes, localização, código de região, dentre outros. Essa implementação em Dart proporciona uma estrutura organizada para a manipulação e gerenciamento de grupos de pacientes dentro do sistema.

Diante dessa relação, o atributo regionCode foi adicionado a classe GroupPatients, atuando como uma chave estrangeira que estabelece uma conexão com a classe Region. Essa estrutura simplifica a associação de grupos a regiões específicas cadastradas no sistema.

Para ilustrar essa relação, a Figura 9 apresentará um exemplo prático da associação entre as classes Patient e GroupPatients, demonstrando como os pacientes são agrupados e gerenciados dentro do sistema.

Figura 9 – Associação Classe GroupPatients e Patient implementado em Dart

```dart
class Patient {
  late int cod;
  late String name;
  late String document;
  late String maritalStatus;
  late String contact;
  late String observation;
  late int occupationCode; //Associação de Paciente com Ocupação
  late String religion;
  late int groupPatientsCode; //Associação de Paciente com Grupo
  late bool responsible;
  late bool served;
  late int creationDate;
  late int dateOfBirth;
  late int lastDataChange;
  late bool deleted;

  Patient({
    required this.cod,
    required this.name,
    required this.document,
    required this.maritalStatus,
    required this.contact,
    required this.observation,
    required this.occupationCode,
    required this.religion,
    required this.groupPatientsCode,
    required this.responsible,
    required this.served,
    required this.creationDate,
    required this.dateOfBirth,
    required this.lastDataChange,
    this.deleted = false,
  });

// Outros métodos e funcionalidades da classe aqui...

}
```

Fonte: Figura do Autor

A classe apresentada na Figura 9 está completa, fornecendo uma visão detalhada dos parâmetros utilizados pelo aplicativo. Este conjunto de atributos reflete a versão atual do sistema, representando os dados necessários para a manipulação e gerenciamento eficaz dos pacientes.

Os códigos apresentados nas Figuras 8 e 9 apresentam exemplos de associação utilizados no desenvolvimento do aplicativo. No entanto, existem outras

formas de implementar uma associação. A abordagem adotada no aplicativo Saúde do Interior é utilizar a chave do objeto associado, uma prática otimizada pensada especificamente para este contexto. Nessa abordagem, não é incluído um objeto dentro do outro, como ocorre na maioria das implementações de associação, visando melhorar o desempenho de memória do dispositivo e otimizar a manipulação de dados, especialmente considerando o grande volume de informações processadas localmente pela aplicação.

3.5.4. Desenvolvendo mapa

O aplicativo “Saúde do Interior” apresenta uma funcionalidade crucial baseada em geolocalização, permitindo aos usuários visualizar as localizações dos grupos familiares em um mapa. Durante o processo de cadastro de pacientes, o sistema registra a localização em tempo real por meio do *Global Positioning System* (GPS) dos dispositivos. Para garantir a precisão, as regras de negócio estabelecem que não é possível inserir coordenadas manualmente; é obrigatório que o agente de saúde esteja presente no local onde o paciente ou grupo familiar reside para obter a localização exata. Essa funcionalidade reforça a necessidade de presença física para o cadastro e garante a integridade e precisão dos dados.

Ao longo do processo de desenvolvimento, diversos *plugins* (módulo de extensão) Flutter foram empregados para aprimorar as funcionalidades do sistema, mas a exploração detalhada desses plugins não será abordada neste tópico.

Tiles Layers (camadas de mosaicos)

Considerando a necessidade de operar em locais sem acesso à internet limitado ou ausente, foi implementado um mapa que funciona offline, armazenando os tiles do mapa em cache. Os tiles são pequenas imagens que compõem o mapa e são carregadas dinamicamente. Ao armazená-los em cache, o mapa pode ser visualizado de forma eficaz, uma experiência contínua aos usuários em áreas com conexão intermitente. Esse recurso é fundamental para garantir a funcionalidade do aplicativo em regiões remotas ou com acesso limitado à internet.

Tiles Server (servidor de mosaicos)

Os tiles utilizados pelo aplicativo Saúde do Interior são fornecidos por um servidor de tiles. Esse servidor é responsável por fornecer os tiles de mapa para os aplicativos de mapeamento que solicitam esses dados. Cada tile contém uma imagem de uma parte específica do mapa em um determinado nível de *zoom* (nível de ampliação). Em resumo, o servidor de tiles serve essas imagens para os clientes, como navegadores e aplicações móveis, permitindo a visualização eficiente do mapa. O termo "*tile server*" é comumente utilizado para se referir a esse tipo de servidor que fornece os *tiles* de mapas para aplicativos de mapeamento.

Consumindo Tiles

A obtenção dos tiles segue um padrão específico, dependendo do servidor utilizado. No caso do servidor empregado nesta aplicação, é utilizado um link que requer três parâmetros. Esse link segue um formato comum para solicitação de tiles de mapas, como no OpenStreetMap (OpenStreetMap, 2024). Os parâmetros "z", "x" e "y" representam diferentes níveis de zoom e coordenadas na grade de *tiles* do mapa.

- “z”: Representa o nível de zoom, onde 0 é o nível mais amplo e o zoom aumenta à medida que o valor de “z” aumenta (OpenStreetMap, 2024);
- “x” e “y”: Representam as coordenadas de colunas e linhas do tiles, respectivamente, na grade do mapa. O valor de “x” varia de 0 a 2^z-1, e o valor de “y” varia de 0 a 2^z-1, onde z é o nível de zoom atual (OpenStreetMap, 2024).

Esses parâmetros são substituídos por valores específicos quando a solicitação é feita para o servidor de tiles. Por exemplo, o link “https://tile.openstreetmap.org/10/100/200.png” solicita o tile na coluna 100 e linha 200 no nível de zoom 10 do mapa OpenStreetMap.

Esse sistema de tiles permite que os mapas sejam divididos em pequenos blocos para carregamento eficiente e exibição em aplicativos de mapas, como o Saúde do Interior. Cada tile contém uma parte do mapa em uma determinada resolução e zoom, e são carregados dinamicamente conforme o usuário navega pelo mapa. Isso ajuda a economizar largura de banda e acelerar o carregamento e a renderização do mapa.

Dentro da aplicação, há um controlador responsável por identificar a grade e o nível de zoom exibidos na tela do mapa conforme o usuário movimenta-a. Assim que esses parâmetros são reconhecidos, o controlador envia os dados necessários ao servidor para solicitar os tiles daquela região em exibição. Esse processo garante que os tiles correspondentes à área visualizada sejam baixados de forma eficiente, proporcionando uma experiência contínua ao usuário durante a navegação pelo mapa.

Tiles em cache

Para garantir uma visualização contínua e eficiente do mapa, especialmente em áreas com conectividade limitada, o aplicativo adota uma abordagem inteligente de armazenamento de tiles localmente. Essa estratégia permite que os usuários acessem rapidamente os tiles necessários para a navegação, sem depender exclusivamente da disponibilidade de uma conexão com a internet. Assim, mesmo em locais remotos ou com sinal fraco, os usuários podem continuar utilizando o aplicativo com confiança, aproveitando todas as suas funcionalidades essenciais.

Além disso, a implementação do cache de tiles contribui significativamente para a economia de dados, reduzindo a quantidade de informações que precisam ser baixadas repetidamente através da internet. Essa eficiência no consumo de dados não apenas otimiza o desempenho do aplicativo, mas também oferece uma experiência mais agradável ao usuário, minimizando os tempos de espera e garantindo uma navegação mais fluida e responsiva. Dessa forma, os usuários

podem desfrutar de uma experiência de uso mais eficiente e econômica, sem comprometer a qualidade ou a integridade dos dados do mapa.

Ademais, a estratégia de cacheamento dinâmico dos tiles permite que o aplicativo se adapte de forma inteligente às condições de conectividade variáveis dos usuários. Mesmo em ambientes com conexões instáveis ou intermitentes, o aplicativo continua funcionando de maneira confiável, utilizando os tiles previamente armazenados em cache para garantir uma experiência de navegação contínua. Essa capacidade de operar de forma consistente em diferentes cenários de conectividade torna o aplicativo mais versátil e acessível, atendendo às necessidades dos usuários em diversas situações do dia a dia.

Tiles Bounds (limites de mosaicos)

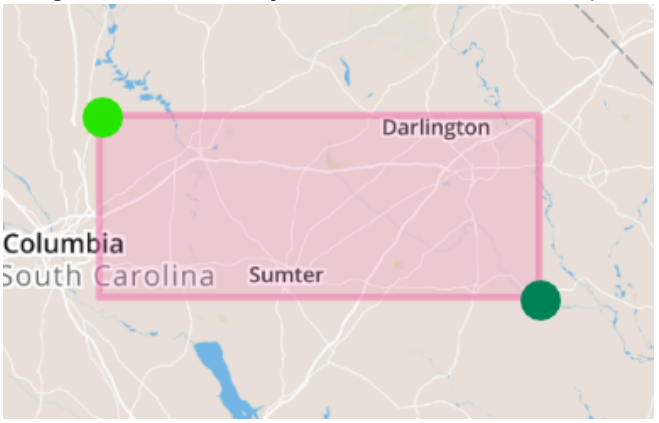
O conceito de *Tiles Bounds* (limite de mosaicos) foi introduzido para otimizar o uso de recursos dentro do aplicativo Saúde do Interior. Como mencionado anteriormente, os tiles são armazenados dinamicamente em cache, mas isso pode resultar em um consumo desnecessário de recursos de armazenamento. Para resolver essa questão, foi implementada a funcionalidade de definir limites geográficos dentro do aplicativo. Isso permite que os usuários delimitem a área de interesse no território que será coberto pelo aplicativo, restringindo assim a navegação apenas a essa região específica. Dessa forma, é garantido uma utilização mais eficiente dos recursos de armazenamento, ao mesmo tempo que é proporcionado uma experiência mais direcionada e relevante aos usuários.

Além de delimitar os limites geográficos no aplicativo, também é implementado um mecanismo de atualização desses limites. Isso permite que os usuários ajustem facilmente as áreas de interesse conforme necessário, garantindo que apenas os tiles relevantes sejam armazenados em cache. Essa abordagem flexível e dinâmica oferece aos usuários maior controle sobre os recursos do aplicativo, tornando a experiência de navegação mais personalizada e eficiente.

A Figura 10 ilustra um exemplo de delimitação dos limites geográficos no mapa do aplicativo Saúde do Interior. Por meio da definição das coordenadas de latitude e longitude de dois pontos extremos na diagonal, é possível criar uma área de limite quadrilátero. Essa delimitação restringe todas as operações do aplicativo a

esse perímetro geográfico específico, garantindo que as funcionalidades sejam aplicadas apenas dentro dessas regiões delimitadas.

Figura 10 – Delimitação de Tiles Bounds no mapa



Fonte: Figura do Autor

3.5.5. Banco de Dados

No desenvolvimento do projeto "Saúde do Interior", o banco de dados MySQL foi utilizado como Sistema de Gerenciamento de Banco de Dados (SGBD). A escolha dessa tecnologia se baseou em fatores como confiabilidade, escalabilidade e a capacidade de manipulação de grandes volumes de dados, características essenciais para o gerenciamento das informações no contexto do projeto.

Estrutura Relacional

O MySQL, por ser um banco de dados relacional, se adequa perfeitamente ao modelo de dados estabelecido no projeto, que envolve categorias como Regiões, Grupos, Pacientes e Serviços. A estrutura relacional facilita a criação de tabelas interligadas por meio de chaves primárias e estrangeiras, o que garante a consistência e a integridade dos dados. No contexto do projeto, cada categoria foi mapeada para tabelas específicas:

- A tabela Regiões armazena dados geográficos das áreas atendidas;
- A tabela Grupos contém informações sobre as unidades familiares;
- A tabela Pacientes organiza os dados pessoais e da saúde dos indivíduos;

- A tabela Serviços registra todos os atendimentos realizados pelos agentes de saúde.

Essas tabelas interligadas permitem consultas complexas e cruzamento de informações, úteis para gerar relatórios e acompanhar atividades de saúde.

Gerenciamento de Grandes Volumes de Dados

A escolha do MySQL se deu pela sua eficiência no gerenciamento de grandes volumes de dados, como coletados em diversas regiões, de pacientes e de atendimentos. A capacidade de escalabilidade do MySQL garante que o sistema possa crescer e suportar consultas rápidas mesmo com grandes bases de dados. Com a otimização de consultas, por meio de índices e práticas de normalização, o sistema permite que os agentes de saúde e gestores acessem rapidamente as informações necessárias, otimizando o atendimento e a gestão.

Segurança e Integridade dos Dados

O MySQL também foi escolhido pela implementação de mecanismos de segurança e integridade dos dados. Ele oferece suporte a controles de acesso por autenticação de usuários, garantindo que apenas agentes autorizados tenham acesso às informações. A integridade dos dados é mantida pela definição de restrições e regras de consistência dentro do banco de dados, evitando a inserção de dados duplicados ou incorretos.

3.5.6. Banco de Dados

No desenvolvimento do projeto "Saúde do Interior", o banco de dados MySQL foi utilizado como SGBD. A escolha por essa tecnologia se baseou em vários fatores, como a confiabilidade, escalabilidade e a capacidade de manipulação de grandes volumes de dados, características essenciais para o gerenciamento das informações no contexto do projeto.

Estrutura Relacional

O MySQL, por ser um banco de dados relacional, se adequa perfeitamente ao modelo de dados estabelecido no projeto, que envolve categorias como Regiões, Grupos, Pacientes e Serviços. A estrutura relacional facilita a criação de tabelas interligadas por meio de chaves primárias e estrangeiras, o que garante a consistência e a integridade dos dados. No contexto do projeto, cada categoria foi mapeada para tabelas específicas:

- A tabela Regiões armazena dados geográficos das áreas atendidas;
- A tabela Grupos contém informações sobre as unidades familiares;
- A tabela Pacientes organiza os dados pessoais e da saúde dos indivíduos;
- A tabela Serviços registra todos os atendimentos realizados pelos agentes de saúde.

Essas tabelas interligadas permitem consultas complexas e cruzamento de informações, úteis para gerar relatórios e acompanhar atividades de saúde.

Gerenciamento de Grandes Volumes de Dados

O MySQL foi escolhido pela sua eficiência no gerenciamento de grandes volumes de dados. O projeto "Saúde do Interior" armazena dados de diversas regiões, pacientes e atendimentos. A capacidade de escalabilidade do MySQL garante que o sistema possa crescer e suportar consultas rápidas mesmo com grandes bases de dados.

Com a otimização de consultas, por meio de índices e práticas de normalização, o sistema permite que os agentes de saúde e gestores acessem rapidamente as informações necessárias, otimizando o atendimento e a gestão.

Segurança e Integridade dos Dados

Outra característica relevante do MySQL no contexto do projeto foi a implementação de mecanismos de segurança e integridade dos dados. O banco de dados oferece suporte a controles de acesso por meio de autenticação de usuários, garantindo que apenas agentes autorizados tenham acesso às informações. Além disso, a integridade dos dados é mantida por meio da definição de restrições e regras de consistência dentro do banco de dados, evitando a inserção de dados duplicados ou incorretos.

Backup (cópia de segurança) e Recuperação de Dados

A continuidade dos serviços e a segurança das informações armazenadas foram prioridade no projeto. O MySQL oferece funcionalidades robustas para backup e recuperação de dados, possibilitando que o sistema mantenha cópias de segurança regulares das informações críticas em casos de falhas ou incidentes, garantindo que os dados estejam sempre disponíveis e seguros.

Compatibilidade e Integração

Para o projeto "Saúde do Interior", a integração entre a API e o aplicativo foi projetada com atenção especial à compatibilidade e à sincronização de dados. Na API, foi utilizado o MySQL. No aplicativo móvel, por sua vez, foi utilizado o SQFLite, uma implementação local do SQLite,, adequada para o armazenamento offline de dados. O SQFLite oferece um solução leve e eficiente para o armazenamento no próprio dispositivo, permitindo o registro e a consulta de informações mesmo em áreas com conectividade limitada.

Quando a conectividade está disponível, ocorre a sincronização entre o banco de dados local (SQFLite) do aplicativo e o banco de dados MySQL da API. Embora sejam tecnologias diferentes, a estrutura das tabelas e dados foi cuidadosamente planejada para assegurar uma integração eficiente e confiável. Essa abordagem

evita problemas de compatibilidade e perda de informações durante o processo de sincronização, proporcionando uma troca de dados rápida e segura entre o dispositivo móvel e o servidor central.

A decisão de usar MySQL na API e SQFLite no aplicativo foi estratégica, visando atender às especificidades do projeto "Saúde do Interior". Essa escolha trouxe benefícios significativos em termos de organização, escalabilidade e segurança dos dados, além de garantir que o sistema pudesse operar de maneira robusta em ambientes com conectividade limitada, atendendo com eficácia às necessidades de monitoramento de saúde em áreas rurais.

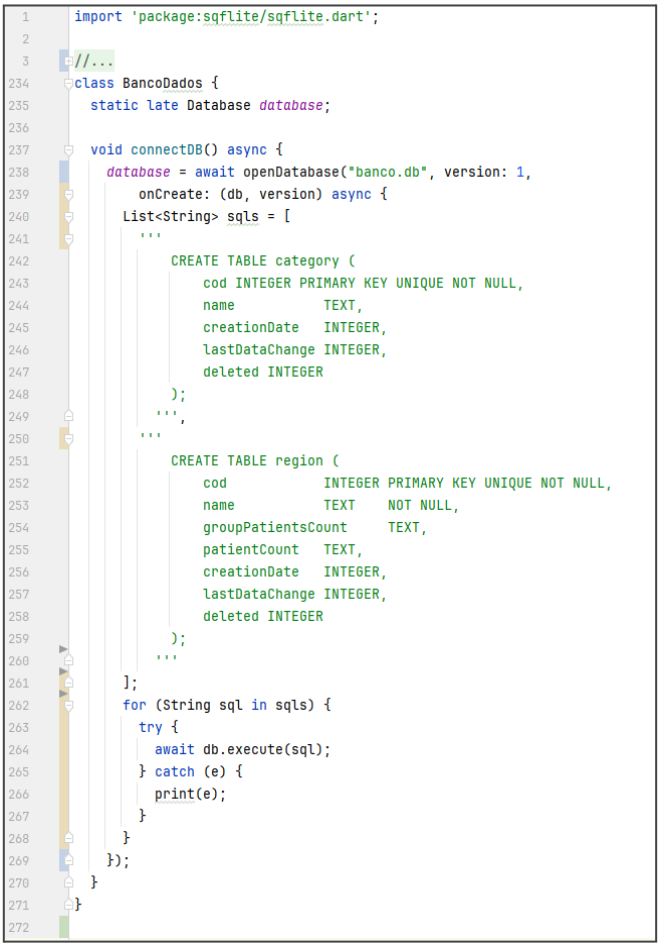
A questão da continuidade dos serviços e da segurança das informações armazenadas foi uma prioridade no projeto. O MySQL oferece funcionalidades robustas para backup e recuperação de dados, possibilitando que o sistema mantenha cópias de segurança regulares dos dados, protegendo informações críticas em casos de falhas ou incidentes. Dessa forma, o banco de dados garante que os dados estejam sempre disponíveis e seguros.

Exemplos Práticos de Implementação

Nesta seção, será apresentada a implementação prática do banco de dados, utilizando o plugin SQFLite no Flutter. O exemplo ilustra como configurar e manipular um banco de dados SQLite para armazenar dados coletados no aplicativo, garantindo que as operações possam ser realizadas offline e sincronizadas posteriormente.

A Figura 11 a seguir demonstra o processo de criação de um banco de dados SQLite com o plugin SQFLite, destacando as etapas principais, como a inicialização do banco, a definição de tabelas e o uso de transações para inserir, atualizar e consultar registros.

Figura 11 – Criando Banco de Dados SQLite com SQFLite



```
1   import 'package:sqflite/sqflite.dart';
2
3   //...
234 class BancoDados {
235   static late Database database;
236
237   void connectDB() async {
238     database = await openDatabase("banco.db", version: 1,
239         onCreate: (db, version) async {
240       List<String> sqls = [
241         '''
242             CREATE TABLE category (
243                 cod INTEGER PRIMARY KEY UNIQUE NOT NULL,
244                 name           TEXT,
245                 creationDate   INTEGER,
246                 lastDataChange INTEGER,
247                 deleted INTEGER
248             );
249         ''',
250         '''
251             CREATE TABLE region (
252                 cod            INTEGER PRIMARY KEY UNIQUE NOT NULL,
253                 name           TEXT    NOT NULL,
254                 groupPatientsCount     TEXT,
255                 patientCount   TEXT,
256                 creationDate   INTEGER,
257                 lastDataChange INTEGER,
258                 deleted INTEGER
259             );
260         '''
261       ];
262       for (String sql in sqls) {
263         try {
264           await db.execute(sql);
265         } catch (e) {
266           print(e);
267         }
268       }
269     });
270   }
271 }
272
```

Fonte: Figura do Autor

Na Figura 11, é apresentada a classe BancoDados, que desempenha um papel fundamental na criação e gestão do banco de dados local. Um dos principais elementos dessa classe é a propriedade *database* (linha 235), declarada como uma instância estática para garantir a consistência e economia de recursos, evitando a criação de múltiplas conexões.

O método *connectDB* (linha 237) é responsável pela inicialização do banco de dados e é executado uma única vez durante o ciclo ativo do aplicativo. Na linha 238, ocorre a instância da propriedade database usando o método openDatabase. Esse método abre o banco de dados especificado, neste caso, banco.db, passado como o primeiro parâmetro. Caso o banco de dados já exista, ele é simplesmente aberto e a instância é retornada, caso contrário, um novo banco de dados é criado e sua instância é fornecida.

Na linha 240, observamos uma lista de comandos SQL que será utilizada para gerar as tabelas necessárias para o funcionamento do sistema. Em seguida, na linha 262, é implementado um laço de repetição FOR, responsável por iterar sobre cada script SQL na lista. A linha 264 executa cada comando SQL individualmente com await db.execute(sql), criando as tabelas de acordo com as definições especificadas.

Figura 12 – Buscando paciente no banco de dados por ID

```
59  static Future<PatientController?> getPatientId(int cod) async {
60    List<Map<String, Object?>> result = await BancoDados.database.query(
61        'patient',
62        where: 'cod = ? AND deleted = 0',
63        whereArgs: [cod.toString()],
64        limit: 1);
65    if (result.isEmpty) {
66      return null;
67    }
68    return PatientController.initialize(Patient.fromMap(result.first));
69  }
```

Fonte: Figura do Autor

Na Figura 12, o código Flutter utiliza o plugin SQFLite para realizar uma consulta no banco de dados local SQLite, buscando um registro específico na tabela patient, com base no *Identification* (ID) do paciente (cod). A função getPatientId é uma função assíncrona, definida como um método Future que retorna um objeto PatientController, ou *null* (nulo) caso nenhum paciente com o ID especificado seja encontrado.

No início do código, a função faz uma *query* (consulta) na tabela patient por meio da instância BancoDados.database. A cláusula where define a condição de busca, onde o campo cod deve corresponder ao valor do ID passado como parâmetro para a função. Além disso, há um filtro adicional que verifica se o campo *deleted* (excluído) é igual a 0, o que significa que somente pacientes não excluídos (não marcados como deletados) serão retornados na busca. Esse tipo de verificação é importante em sistemas onde registros são logicamente excluídos, mas permanecem no banco de dados.

O parâmetro limit: 1 garante que apenas o primeiro resultado que atender às condições especificadas seja retornado. Isso otimiza a consulta, pois impede que o banco de dados retorne mais registros do que o necessário, reduzindo o uso de memória e aumentando a eficiência do aplicativo.

Depois da execução da consulta, o código verifica se o result está vazio usando if(result.isEmpty). Caso esteja vazio, a função retorna null, indicando que nenhum paciente com o ID especificado foi encontrado. Se o resultado não estiver vazio, ou seja, se um registro correspondente foi encontrado, o código utiliza Patient.fromMap(result.first) para converter o resultado em um objeto Patient. Esse objeto é então passado para o método *initialize* de PatientController, que retorna um controlador de paciente inicializado com os dados recuperados. Essa estrutura permite uma forma simples e eficiente de buscar um registro específico no banco de dados e retornar a instância de PatienteController pronta para uso.

3.5.7. Interface de Usuário (UI)

Tela Login

A Figura 13, apresenta a tela de *login* (conexão) da aplicação. Na parte superior, está exibida a logo do aplicativo, que proporciona uma identidade visual imediata. Abaixo, encontram-se os campos para inserção do login e da senha, onde o usuário deve inserir suas credenciais, previamente fornecidas pela equipe responsável. Essa interface foi projetada para garantir um acesso seguro e eficiente ao sistema.

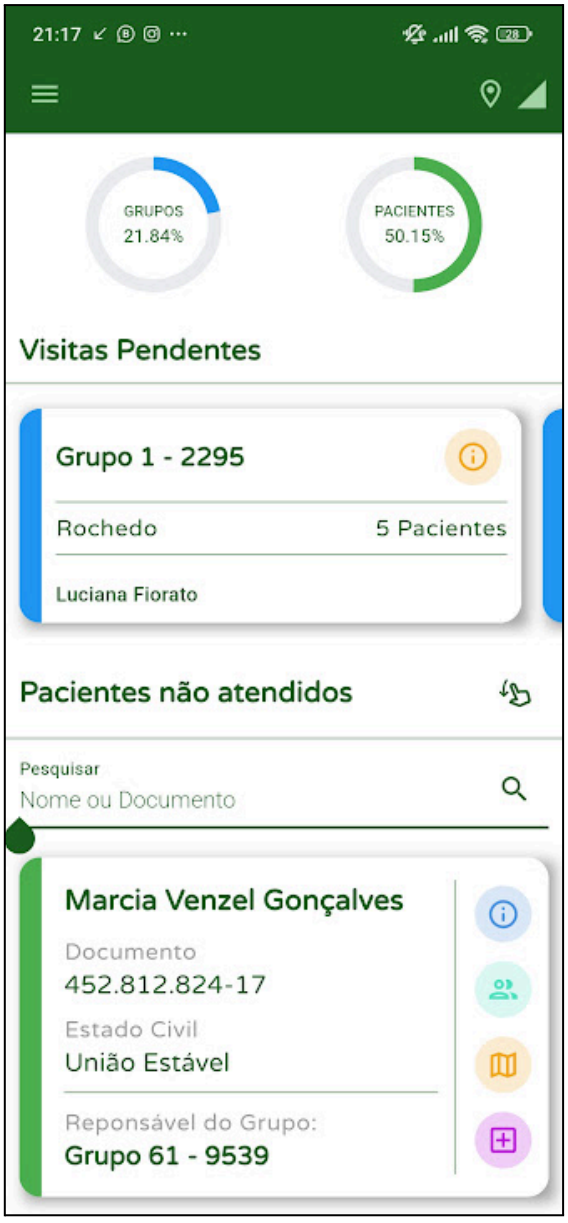
Figura 13 – Tela Login Aplicação

Fonte: Figura do Autor

Tela Inicial

A Interface de Usuário (UI), apresentada na Figura 14, exibe uma tela Inicial clara e bem organizada, facilitando o acesso às principais funcionalidades do sistema. Ela utiliza uma palheta de cores predominante verde, que transmite tranquilidade e está associada ao campo da saúde, além de manter a consistência visual ao longo do aplicativo.

Figura 14 – Tela inicial, exibindo painéis e indicadores de atendimento e conexão.



Fonte: Figura do Autor

1. **Indicadores de Conexão:** Localizados no canto superior direito da interface, esses indicadores fornecem informações em tempo real sobre a conexão do aplicativo com a internet e o GPS. Eles são fundamentais para auxiliar o usuário, especialmente em áreas de cobertura limitada, informando a disponibilidade de dados e a localização. Dessa forma, o usuário pode ajustar as operações conforme a conectividade disponível, garantindo a continuidade e a precisão no registro dos atendimentos;

2. **Indicadores de Atendimento:** Localizados no topo da tela, dois gráficos de progresso em formato circular exibem percentuais relativos aos grupos e pacientes, representando de forma visual o *status* (estado) das visitas e atendimentos. Esses gráficos permitem uma rápida percepção do progresso das atividades, indicando claramente a quantidade de visitas e atendimentos pendentes. Dessa forma, o usuário pode monitorar de forma prática e imediata o andamento das tarefas, facilitando o planejamento e o cumprimento das metas de atendimento;

3. **Listagem de Grupos:** A seção “Visitas pendentes” exibe informações sobre grupos familiares que ainda precisam ser atendidos. Cada cartão mostra o nome do grupo, a região, o número de pacientes e um ícone de visualização, o que permite uma rápida identificação e fácil navegação para mais detalhes. A disposição em cartões organiza as informações de forma clara e separada, facilitando a compreensão;

4. **Busca por Pacientes não Atendidos:** Logo abaixo, há uma área dedicada à pesquisa de pacientes não atendidos, onde o usuário pode digitar o nome do paciente, e o sistema sugere automaticamente resultados relevantes, como demonstrado no exemplo da paciente “Márcia”. A pesquisa é funcional e direta, com o nome do paciente, CPF e status civil exibidos, proporcionando uma visão rápida e objetiva;

   Ações Rápidas: A interface inclui quatro ícones para cada paciente, otimizando a navegação:

- O primeiro ícone, no canto superior direito do cartão, permite acessar mais informações do paciente em questão;
- O segundo ícone leva aos detalhes do grupo familiar, expandindo as informações pertinentes ao grupo;
- O terceiro ícone, de mapa, permite a visualização da localização geográfica do grupo no mapa, o que é essencial para visitas domiciliares;
- O quarto ícone, de atendimento, dá acesso direto ao início do atendimento do paciente selecionado, agilizando o processo de acompanhamento;

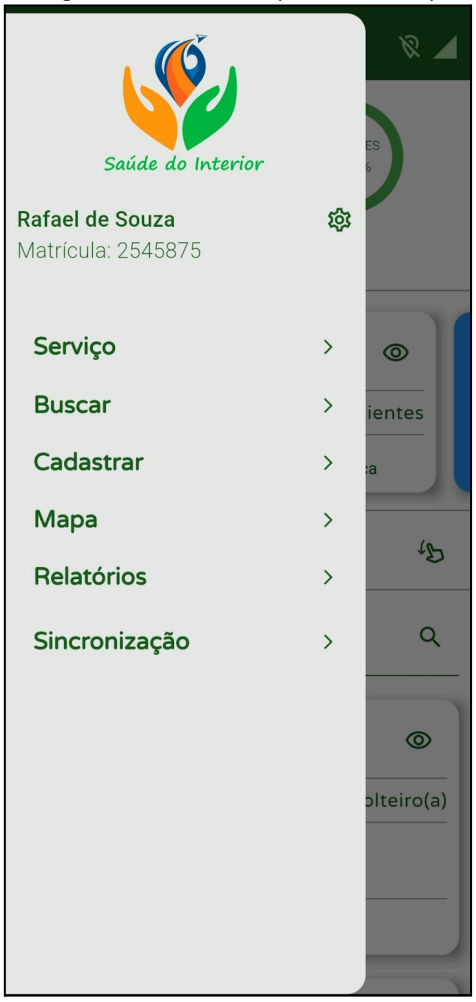
5. Designer Simples e Funcional: O layout é minimalista, com uso eficaz do espaço, proporcionando uma experiência confortável ao usuário. A interface utiliza textos claros e ícones visuais, facilitando a navegação para diferentes ações com poucos cliques.

Essa abordagem centrada no usuário facilita a interação do profissional de saúde com o aplicativo, promovendo agilidade e eficiência nas operações, elementos fundamentais para o contexto de saúde em áreas rurais.

Drawer (Menu Lateral)

A Interface de Usuário (UI) do *Drawer* (menu lateral) apresentada na Figura 15, é simples, organizada e fácil de usar, proporcionando ao usuário um acesso rápido às principais funcionalidades da aplicação "Saúde do Interior".

Figura 15 – Drawer (Menu lateral)



Fonte: Figura do Autor

1. **Identificação do Usuário:** Na parte superior do menu, o usuário é identificado pelo nome e matrícula, o que personaliza a experiência e oferece segurança ao garantir que o profissional de saúde esteja ciente de que está utilizando suas próprias credenciais. Isso também facilita a responsabilidade individual sobre as operações realizadas no sistema;
2. **Logo e Identidade Visual:** O *logo* (logotipo) da aplicação aparece no topo, reforçando a identidade visual do aplicativo e o propósito voltado para a saúde rural. A combinação de cores vibrantes e o símbolo das mãos denotam cuidado, acolhimento e proteção, valores essenciais no contexto da saúde;
3. **Menu de Navegação:** O menu exibe seis opções principais, organizadas verticalmente.

- Serviço: Acessa a área de operações e atendimentos realizados;
- Buscas: Facilita a localização de grupos, pacientes e informações;
- Cadastrar: Direciona o usuário para o cadastro de novos grupos ou pacientes;
- Mapa: Acesso rápido ao mapa, essencial para a visualização da localização das famílias e pacientes;
- Relatórios: Para gerar e consultar relatórios de atendimentos e visitas realizadas;
- Sincronização: Voltado para sincronizar os dados offline com a base de dados central, importante no contexto de áreas rurais com pouca ou nenhuma conectividade.

3.5.8. Sincronização de Dados Offline

A sincronização de dados offline é um componente essencial para o bom funcionamento do projeto "Saúde do Interior", considerando que o público-alvo são pacientes residentes em áreas rurais, onde a conectividade à internet é limitada ou inexistente. Dessa forma, a capacidade de operar offline e sincronizar dados posteriormente, quando uma conexão se torna disponível, garante que os profissionais de saúde possam realizar suas atividades sem interrupções ou perda de dados.

Os dados coletados durante as visitas domiciliares, como informações de pacientes, relatórios de atendimento, geolocalização dos grupos e outras observações clínicas, são temporariamente armazenados no dispositivo móvel do profissional. Esses dados precisam ser mantidos de forma segura e organizada até que uma conexão à internet seja restabelecida.

Durante o processo de sincronização, é possível que ocorram conflitos de dados, especialmente se as informações forem editadas em diferentes dispositivos. Para lidar com esse desafio, é essencial que o sistema implementa políticas de resolução de conflitos, como a prevalência de dados mais recentes ou a apresentação de um resumo para que o usuário escolha qual versão deve ser mantida;

É importante considerar que a escolha da estratégia de resolução de conflitos depende de uma avaliação detalhada do fluxo de trabalho dos usuários e da criticidade das informações envolvidas. No contexto do "Saúde do Interior", onde os agentes de saúde podem acessar e editar os dados em diferentes dispositivos e momentos, a solução de prevalência dos dados mais recentes foi definida como a abordagem mais adequada, garantindo que as informações mais atualizadas estejam sempre disponíveis.

Para garantir que essa política atenda a todos os cenários e não ocasione a perda de dados importantes, foi realizada uma análise de requisitos e comportamentos, considerando situações como;

- Acessos simultâneos: Quando dois ou mais agentes de saúde atualizam os dados de um mesmo paciente em dispositivos diferentes. Nesse caso, manter os dados mais recentes garante que o histórico mais atualizado seja preservado no sistema;
- Alterações em múltiplos campos: Para evitar a sobreposição de edições em diferentes campos de um mesmo registro, o sistema poderia também aplicar a prevalência de dados por campo, mantendo as modificações mais recentes em cada campo específico, minimizando a perda de informação.

Em síntese, a opção de prevalência dos dados mais recentes foi adotada como a solução mais simples e eficaz para a maioria dos cenários, especialmente por estar alinhada ao fluxo de trabalho em áreas rurais com baixa conectividade. Entretanto, a implementação de mecanismos complementares, como a revisão de histórico de alterações e notificações de conflitos, também foi cuidadosamente considerada e está prevista para ser adotada em futuras atualizações do sistema, garantindo maior controle e segurança na gestão de dados.

3.6. Funcionalidades

Nesta seção, serão exploradas as funcionalidades essenciais disponíveis na aplicação desenvolvida para auxiliar a gestão da saúde na zona rural. Essas funcionalidades foram projetadas para atender às necessidades dos profissionais de saúde que atuam em áreas remotas e com limitada conectividade à internet.

3.6.1. Cadastros

Regiões

A funcionalidade de cadastro de regiões requer apenas o nome da região para ser registrado. Isso permite que os profissionais identifiquem e organizam as diferentes áreas geográficas abrangidas pela aplicação.

Grupos Familiares

Ao cadastrar grupos familiares, os profissionais devem fornecer um nome que represente de forma clara e concisa o grupo em questão. Além disso, informações sobre a região à qual o grupo pertence e sua geolocalização são essenciais para contextualizar o atendimento.

Pacientes

O cadastro de pacientes exige informações detalhadas sobre cada indivíduo atendido. Além dos dados pessoais, é importante registrar a qual grupo familiar o paciente pertence. Essa associação facilita o acompanhamento e a prestação de cuidados personalizados.

Ocupações

A funcionalidade de ocupações permite que os agentes de saúde registrem informações sobre a ocupação de cada paciente. Por exemplo, um paciente pode ser identificado como “pecuarista”, “agricultor” ou em outra ocupação relevante. Esses dados pré-cadastrados enriquecem o perfil de cada paciente.

Formulários de Atendimento

Os agentes podem criar formulários de atendimento de forma dinâmica. Esses formulários são preenchidos durante as consultas com os pacientes. As questões disponibilizadas no formulário podem variar, abrangendo desde perguntas dicotômicas até questões discursivas. Essa flexibilidade permite adaptar o registro de informações às necessidades específicas de cada atendimento.

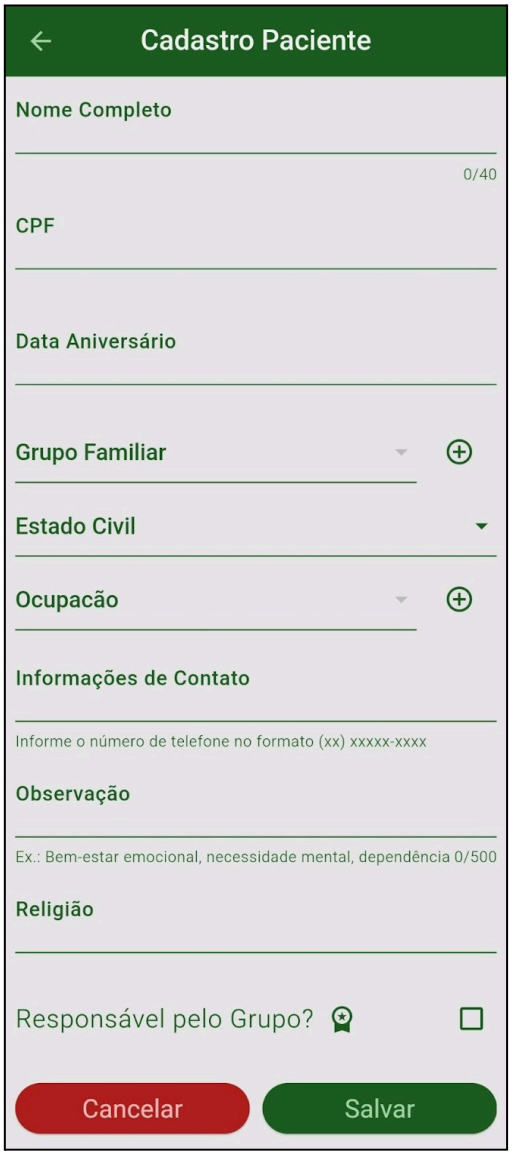
Latitude e Longitude

Os agentes cadastram as coordenadas de latitude e longitude para definir a área de operações no mapa. Essa delimitação é fundamental para otimizar o uso do aplicativo em modo offline, reduzindo o download de dados desnecessários.

Realizando Cadastro de Paciente

A Figura 16, apresentada a seguir, exibe a tela de cadastro de paciente, na qual o agente efetua o registro dos pacientes. Destaca-se a presença do campo “Grupo Familiar”, no qual o agente seleciona o grupo ao qual o paciente pertence. Caso o grupo não tenha sido cadastrado anteriormente, é possível realizar o cadastro clicando no ícone (+) para adicionar um novo grupo.

Figura 16 – Tela de Cadastro de Paciente



Fonte: Figura do autor

3.6.2. Buscas

Os dados cadastrados podem ser acessados de forma simples, permitindo que o profissional visualize informações sobre regiões, grupos familiares, pacientes e localizações. A interface foi projetada para facilitar essa visualização, evitando obstáculos desnecessários. O aplicativo oferece uma variedade de recursos de busca para facilitar o trabalho dos profissionais de saúde na zona rural. Vamos detalhar dois deles.

Regiões, Grupos Familiares e Pacientes

Os agentes podem explorar o mapa interativo para buscar informações. Ao selecionar um grupo familiar, o aplicativo exibirá os pacientes associados a essa área. Isso permite uma visualização contextualizada dos dados.

Além do mapa, os profissionais também podem realizar buscas digitando o nome específico de uma região, grupo familiar ou paciente. Essa flexibilidade torna a busca mais ágil e direcionada.

Localização

A funcionalidade de localização permite que os agentes encontrem rapidamente pontos geográficos relevantes. Seja para identificar um posto de saúde, uma residência ou um ponto de referência, o mapa é uma ferramenta poderosa.

Além disso, os próprios grupos familiares podem ser usados como referência para localização. Por exemplo, ao selecionar um grupo, o aplicativo pode exibir sua posição no mapa, facilitando a navegação até o local.

Essa combinação de busca utilizando mapa e campo de texto, juntamente com a flexibilidade de localização, torna o aplicativo uma ferramenta versátil para os profissionais de saúde em suas atividades diárias.

Realizando Busca por Grupo

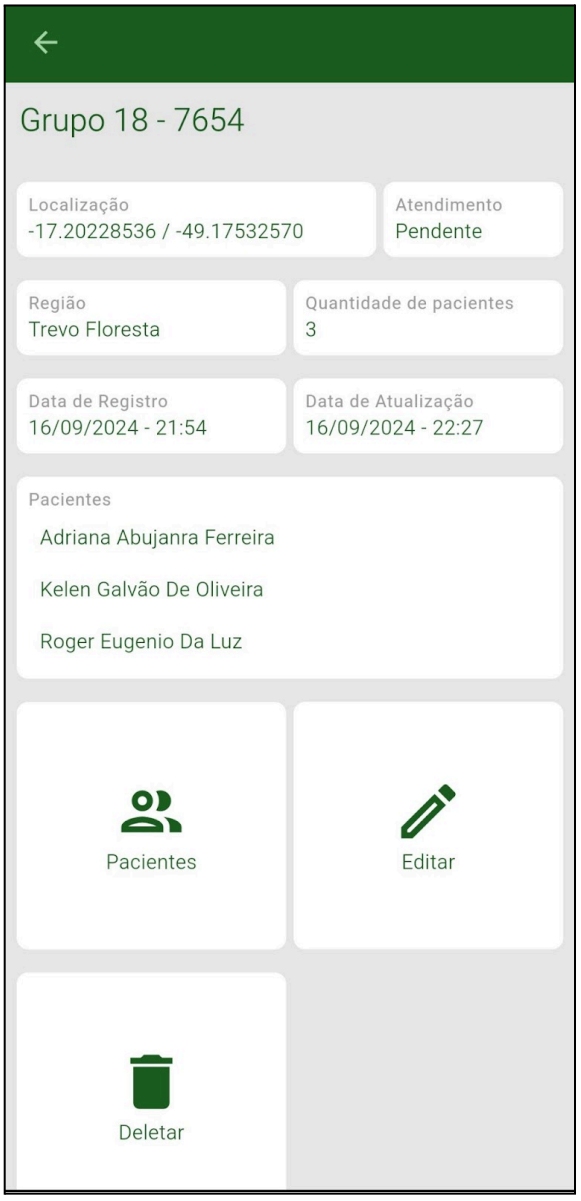
A Figura 17, apresentada a seguir, ilustra a tela de busca por grupo. Nessa tela, há um campo de texto onde o usuário pode realizar a pesquisa pelo nome do grupo desejado. Como exemplificado na imagem com dados de teste, foi pesquisado o ‘grupo 18', e o aplicativo sugeriu automaticamente 'Grupo 18 - 7654', utilizando a funcionalidade de autocompletar. A visualização exibe de forma simples o nome do grupo, acompanhado do nome da região correspondente logo abaixo.

Figura 17 – Tela de Busca por Grupo

Fonte: Figura do autor

Após realizar a busca, conforme demonstrado anteriormente na Figura 17, o usuário tem a possibilidade de expandir as informações e visualizar os detalhes do grupo. A Figura 18, apresenta alguns dados referentes ao grupo selecionado. Nessa tela, são exibidas as coordenadas da localização do grupo familiar, e, ao clicar sobre a localização, o aplicativo abrirá um mapa, exibindo a posição geográfica do grupo. Logo após a localização, é indicado o status de atendimento, que, no caso ilustrado, está com 'atendimento pendente', indicando que nem todos os integrantes do grupo foram atendidos. Além disso, é possível visualizar a região, a quantidade de pessoas no grupo, a data de registro e a data da última atualização. Na parte inferior da tela, há uma lista com os nomes dos pacientes cadastrados no grupo. Ao clicar no nome de um paciente, abre-se uma nova tela com os dados específicos desse indivíduo.

Figura 18 – Tela de Detalhes do Grupo



Fonte: Figura do autor

3.6.3. Localizações

A funcionalidade de localização no projeto “Saúde do Interior” foi desenvolvida com o propósito de aprimorar o acesso aos serviços de saúde em regiões remotas e menos assistidas. Essa ferramenta tem como objetivo principal otimizar os processos de atendimento, conectando usuários e profissionais de saúde de maneira mais eficaz, especialmente em áreas onde a infraestrutura é limitada.

Por meio da tecnologia de geolocalização integrada, o sistema consegue identificar a localização geográfica dos pacientes. Isso possibilita o

acompanhamento regular dos pacientes e, em casos de emergência, permite uma resposta rápida, otimizando o tempo necessário para deslocamentos, por exemplo. Outro aspecto relevante dessa funcionalidade é sua capacidade de fornecer uma visão abrangente da distribuição dos serviços de saúde no interior. Isso permite que gestores públicos e profissionais da área realizem uma análise detalhada das necessidades locais. Com base nessa informação, o sistema contribui para uma alocação mais eficiente de recursos, direcionando esforços para áreas com maior demanda e carência de serviços.

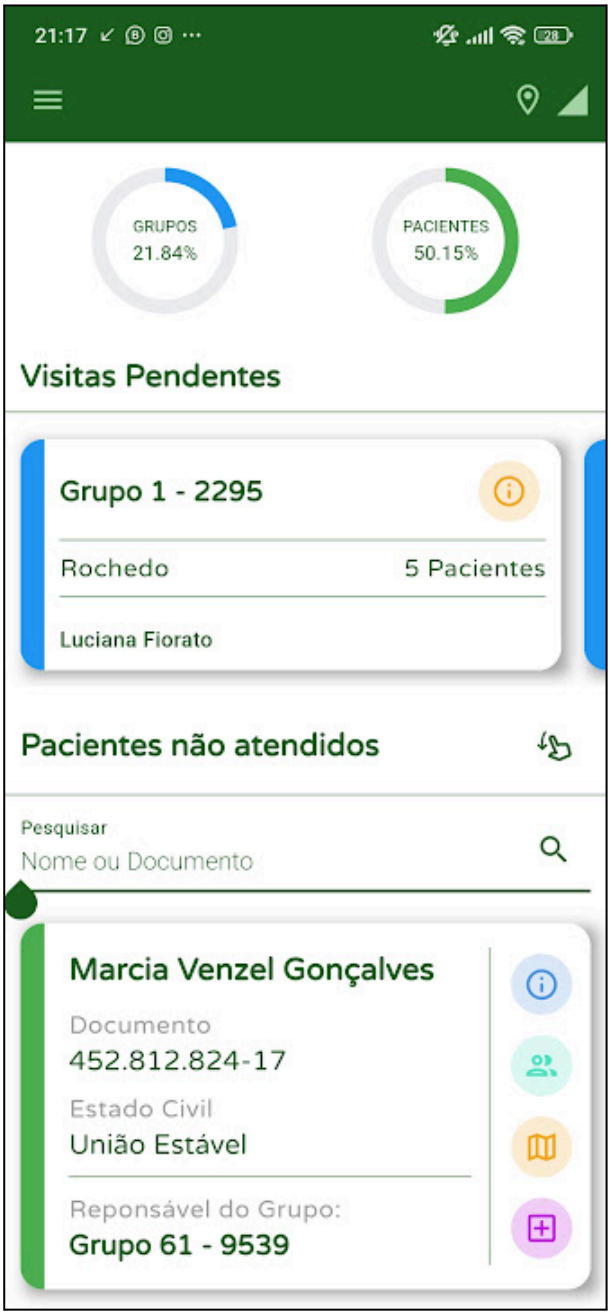
Além das visitas de rotina, a funcionalidade de localização também facilita o planejamento de rotas para atendimentos domiciliares e transporte de pacientes. Isso impacta diretamente na gestão inteligente dos recursos de saúde e na redução do tempo de espera no acesso aos serviços públicos de saúde.

Em resumo, a funcionalidade de localização no projeto “Saúde do Interior” destaca-se como uma solução que utiliza a tecnologia para ampliar o acesso à saúde, melhorando a interação entre profissionais e pacientes e garantindo que os serviços sejam prestados com maior agilidade e precisão.

Visualizando uma Localização

A partir da tela inicial da aplicação, é possível visualizar a localização do grupo familiar de Márcia. Na Figura 19, observa-se um campo de pesquisa para pacientes não atendidos. No exemplo, ao pesquisar pela paciente, seus dados são exibidos imediatamente em um card abaixo do campo de pesquisa. Nesse *card* (cartão), são apresentadas algumas informações sobre a paciente. Há também três ícones disponíveis: o primeiro ícone permite acessar os detalhes do grupo, conforme demonstrado na Figura 19, o segundo ícone é utilizado para visualizar a localização do grupo familiar, e, por fim, o terceiro ícone serve para iniciar o atendimento ao paciente.

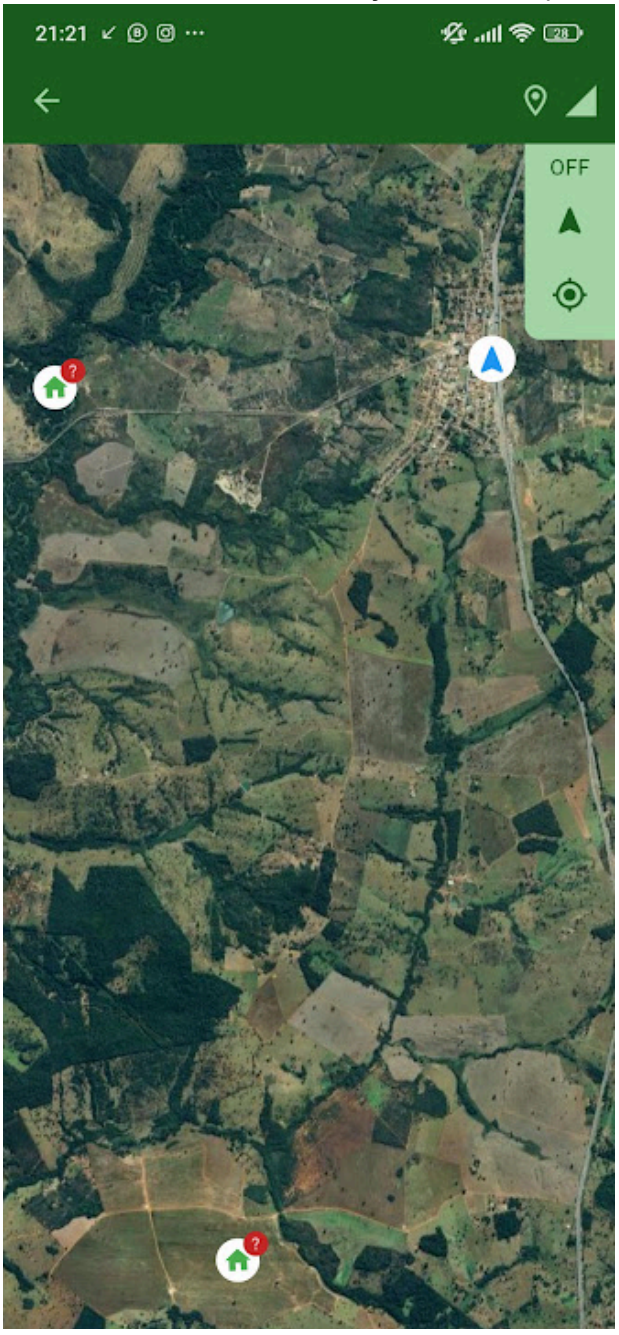
Figura 19 – Tela Inicial (Pacientes não atendidos)



Fonte: Figura do autor

Após localizar o cadastro desejado, conforme apresentado na Figura 19, deve-se clicar no ícone de mapa para ser redirecionado à localização correspondente ao referido cadastro. A Figura 20, exibe a localização do grupo no mapa.

Figura 20 – Tela Exibindo a Localização de um Grupo Familiar



Fonte: Figura do autor

A Figura 20, exibe a localização exata do grupo familiar. O usuário pode clicar no ícone de casa visualizado no mapa para acessar os dados do grupo, incluindo informações sobre os pacientes, conforme demonstrado anteriormente.

3.6.4. Relatórios

No contexto do projeto “Saúde do Interior”, a funcionalidade de geração de relatórios desempenha um papel relevante na análise e gestão das atividades e serviços de saúde oferecidos em áreas remotas. Essa ferramenta foi concebida com o objetivo de fornecer dados consolidados que auxiliam tanto gestores públicos quanto profissionais de saúde na tomada de decisões estratégicas, com base em informações precisas e reais.

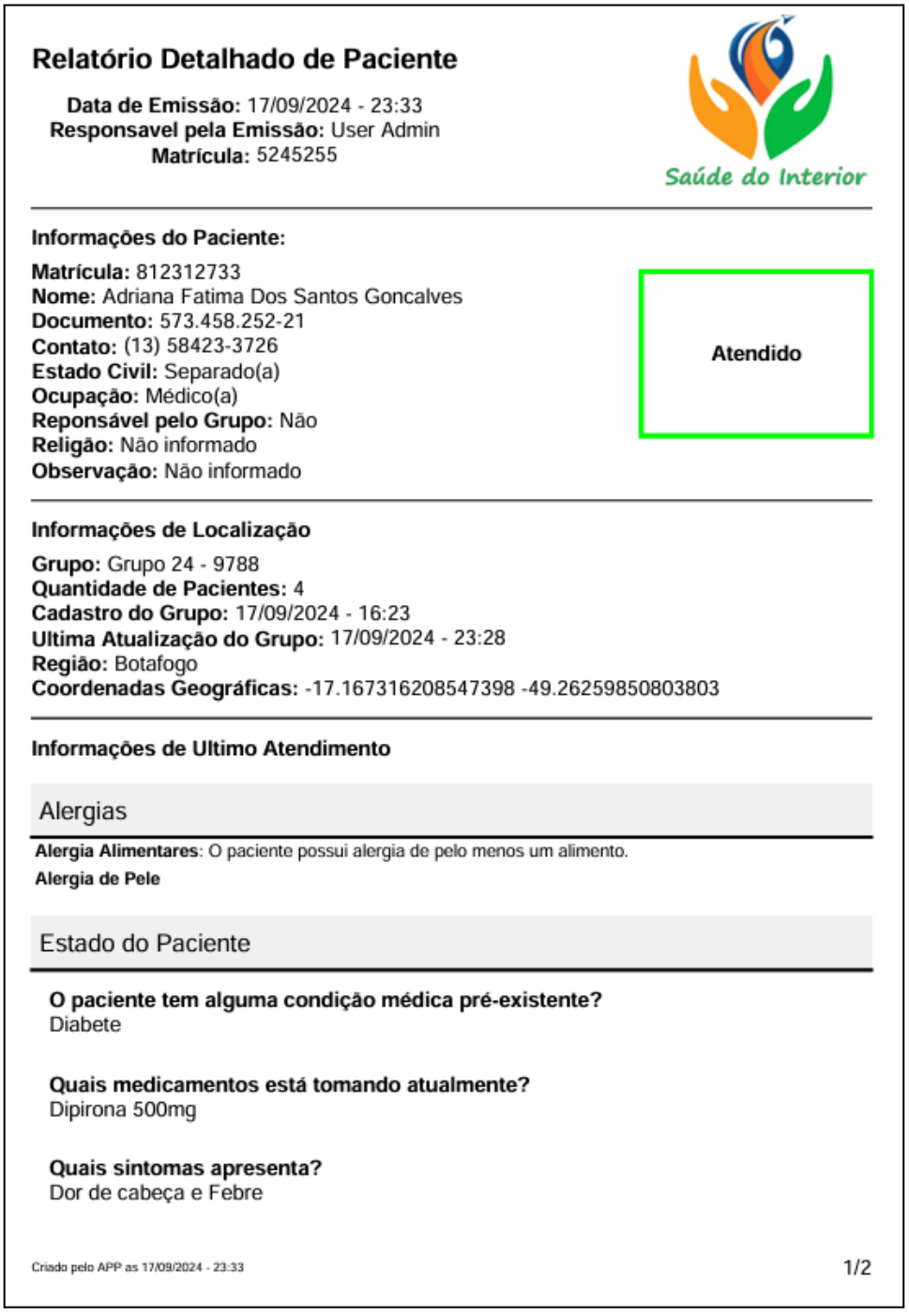
Os relatórios gerados pelo sistema abrangem uma variedade de aspectos relacionados ao atendimento à saúde. Além disso, a personalização desses relatórios é um ponto importante do sistema, permitindo que os usuários selecionem parâmetros e variáveis de interesse.

É possível gerar relatórios em formato *Portable Document Format* (PDF) e compartilhá-los posteriormente, a qualquer momento. Esses relatórios podem ser disponibilizados de forma independente por pacientes, sejam eles atendidos ou não. Além disso, é viável visualizar informações de todos os pacientes ou apenas daqueles que são responsáveis por um grupo específico.

Visualizando Relatórios e Gráficos

Na Figura 21, apresenta-se uma parte do PDF de detalhes do paciente, gerado pelo aplicativo. Nele, é possível visualizar informações como região, grupo e localização em coordenadas. Ademais, é viável examinar detalhes do atendimento realizado pelo agente. No exemplo da Figura 21, destaca-se que o paciente foi atendido.

Figura 21 – Detalhes do Paciente (PDF)



Relatório Detalhado de Paciente

Data de Emissão: 17/09/2024 - 23:33
Responsavel pela Emissão: User Admin
Matrícula: 5245255

Saúde do Interior

Informações do Paciente:

Matrícula: 812312733
Nome: Adriana Fatima Dos Santos Goncalves
Documento: 573.458.252-21
Contato: (13) 58423-3726
Estado Civil: Separado(a)
Ocupação: Médico(a)
Reponsável pelo Grupo: Não
Religão: Não informado
Observação: Não informado

Atendido

Informações de Localização

Grupo: Grupo 24 - 9788
Quantidade de Pacientes: 4
Cadastro do Grupo: 17/09/2024 - 16:23
Ultima Atualização do Grupo: 17/09/2024 - 23:28
Região: Botafogo
Coordenadas Geográficas: -17.167316208547398 -49.26259850803803

Informações de Ultimo Atendimento

Alergias

Alergia Alimentares: O paciente possui alergia de pelo menos um alimento.
Alergia de Pele

Estado do Paciente

O paciente tem alguma condição médica pré-existente?
Diabete

Quais medicamentos está tomando atualmente?
Dipirona 500mg

Quais sintomas apresenta?
Dor de cabeça e Febre

Criado pelo APP as 17/09/2024 - 23:33

1/2

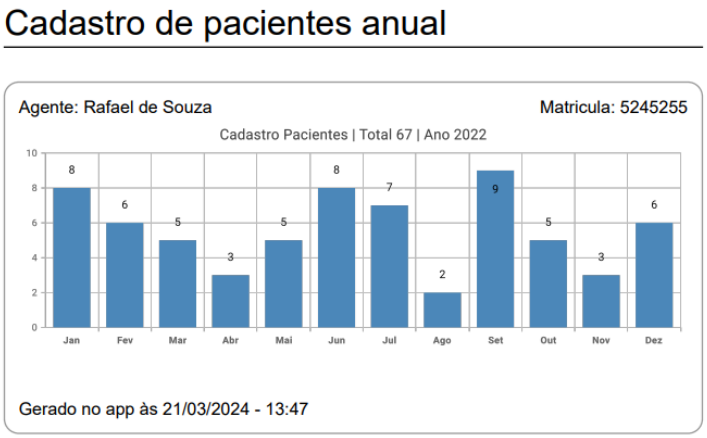
Fonte: Figura do Autor.

Analisando os dados cadastrados e os relatórios gerados pelo aplicativo, percebemos a importância de visualizar essas informações de forma mais dinâmica e interpretativa. Nesse sentido, os gráficos desempenham um papel fundamental, proporcionando uma apresentação visual clara e objetiva dos dados coletados.

Com base em relatórios disponíveis, podemos observar padrões, tendências e informações relevantes que contribuem para uma melhor compreensão do contexto da saúde na zona rural. Dessa forma, os gráficos gerados pelo aplicativo constituem uma ferramenta essencial para a tomada de decisão e o planejamento de ações na área da saúde.

A seguir, na Figura 22, apresenta-se um gráfico que ilustra a quantidade de novos cadastros de pacientes a cada mês do ano.

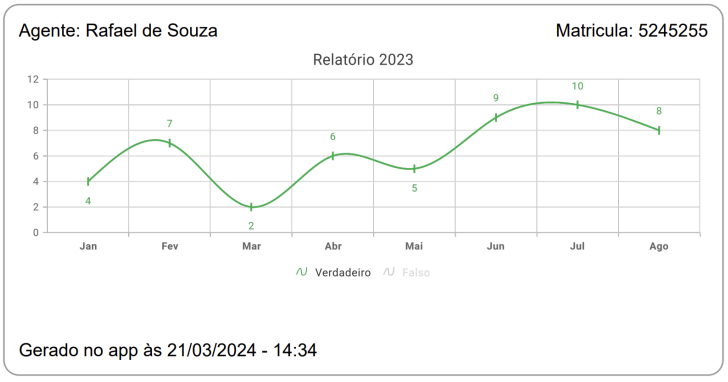
Figura 22 – Gráfico em PDF de novos cadastros de pacientes por mês



Fonte: Figura do Autor

A Figura 23 apresenta um gráfico que ilustra a quantidade de pacientes que apresentaram algum tipo de alergia à pele ao longo do tempo.

Figura 23 – Gráfico que ilustra a incidência de doenças ao longo dos meses.



Fonte: Figura do Autor.

Além dos relatórios apresentados anteriormente, o aplicativo é capaz de gerar vários outros relatórios, sempre visando facilitar a gestão do agente. Esses relatórios adicionais abrangem uma variedade de aspectos relacionados à saúde na zona rural, fornecendo informações detalhadas e *insights* (percepções) valiosos para o profissional de saúde.

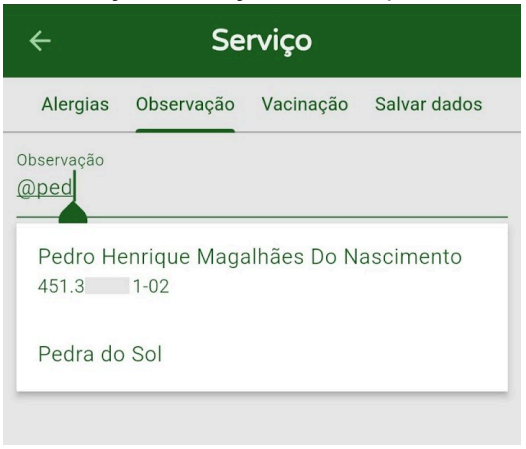
Essa funcionalidade ampla e diversificada de geração de relatórios visa otimizar a tomada de decisões, melhorar o planejamento estratégico e promover uma abordagem baseada em dados.

3.6.5. Funcionalidade Extra

Dentro da aplicação, disponibilizamos diversas funcionalidades com o objetivo de tornar o processo de atendimento ao paciente mais prático e eficiente. A seguir, destacamos uma dessas funcionalidades, cuidadosamente projetada para atender às necessidades dos profissionais de saúde:

O recurso de autocompletar foi desenvolvido para simplificar a inserção de informações durante o atendimento. Para um profissional de saúde mencionar um paciente ou grupo, basta inserir o símbolo “@” seguido do nome desejado. Como ilustrado na Figura 24, a seguir, o sistema apresentará uma lista de opções de autocompletar, tanto para pacientes quanto para grupos, tornando o processo de seleção mais ágil e preciso.

Figura 24 – Demonstração da função autocomplete em um atendimento



Fonte: Figura do Autor.

A saída da seleção, conforme ilustrada pela Figura 24, resultará em um autocomplete do nome seguido do Cadastro de Pessoa Física (CPF) no caso do paciente. Conforme o texto digitado, "@ped" seria substituído por “Pedro Henrique Magalhães do Nascimento - 451.3**.**1-02”. No caso da seleção do grupo, a saída seria “Pedra do Sol”.

3.7. IMPACTO DE TECNOLOGIAS DE DADOS EM SAÚDE RURAL

3.7.1. Melhoria no Acesso à Saúde

Tecnologias de dados, como o aplicativo "Saúde do Interior",impactam significativamente no acesso aos serviços de saúde em regiões rurais, onde o acesso a recursos de saúde é frequentemente limitado. O uso de aplicativos móveis para gerenciar dados de pacientes e realizar visitas domiciliares permite que os profissionais de saúde alcancem e atendam comunidades remotas de maneira mais eficiente. Isso reduz a necessidade de deslocamentos frequentes e facilita o acompanhamento regular dos pacientes, mesmo em áreas sem conectividade constante à internet.

3.7.2. Eficiência na Gestão de Dados

A digitalização de dados através de aplicativos como o "Saúde do Interior" pode contribuir para uma gestão mais eficiente das informações dos pacientes. Ao substituir registros físicos por dados eletrônicos, os profissionais de saúde podem acessar e atualizar informações rapidamente, melhorar a precisão dos registros e reduzir a probabilidade de erros. A capacidade de visualizar dados em tempo real também ajuda na tomada de decisões clínicas mais informadas.

3.7.3. Coordenação e Planejamento

O aplicativo permite uma melhor coordenação entre diferentes profissionais de saúde e unidades de atendimento. A capacidade de cadastrar e visualizar a localização de grupos familiares e pacientes facilita o planejamento das visitas e o gerenciamento das tarefas dos profissionais de saúde. Isso também melhora a

eficiência do uso dos recursos disponíveis e permite uma abordagem mais estruturada para a prestação de cuidados.

3.7.4. Monitoramento e Avaliação da Saúde

Com o registro detalhado de informações sobre pacientes e serviços prestados, o aplicativo oferece uma base sólida para o monitoramento e avaliação da saúde na zona rural. Os dados coletados podem ser analisados para identificar padrões, monitorar tendências e avaliar a eficácia das intervenções de saúde. Isso contribui para uma abordagem mais proativa e baseada em evidências para o gerenciamento da saúde.

3.7.5. Impacto na Comunidade

O impacto das tecnologias de dados vai além da eficiência operacional, ele também afeta diretamente a qualidade de vida das comunidades rurais. O acesso facilitado a cuidados de saúde e a capacidade de monitorar e gerenciar a saúde de forma mais eficaz contribuem para a melhoria geral da saúde e bem-estar dos residentes em áreas remotas.

4. CONCLUSÃO

O desenvolvimento do projeto “Saúde do Interior” viabilizou a elaboração de uma proposta de apoio das atividades realizadas por profissionais de saúde em regiões rurais. O sistema apresentado foi estruturado com foco na operação offline, levando em consideração contextos nos quais a conectividade é limitada. Durante sua concepção, buscou-se implementar recursos para registro de atendimentos, organização de informações por categorias e posterior sincronização com uma base central, sempre considerando características do ambiente de atuação e os requisitos identificados ao longo do processo.

A continuidade do projeto envolve a criação de uma plataforma web centralizada, já em desenvolvimento, que terá como objetivo ampliar as possibilidades de gestão e visualização dos dados. A utilização futura desta interface poderá contribuir para o acompanhamento mais detalhado das informações coletadas, além de permitir a incorporação de novas funcionalidades e formas de análise. A proposta de expansão busca atender demandas relacionadas à organização e ao uso eficiente dos dados, considerando o contexto de atuação dos profissionais e os desafios observados no cotidiano das ações de saúde em áreas de difícil acesso.

4.1. TRABALHOS FUTUROS

4.1.1. Plataforma Web Centralizada de Gestão

Apesar dos avanços alcançados com o desenvolvimento do aplicativo Saúde do Interior, o projeto apresenta potencial para expansão e aprimoramento. Uma das principais iniciativas planejadas para os próximos passos é o desenvolvimento completo da Plataforma Web Centralizada de Gestão, que complementa e expandirá as funcionalidades já implementadas no sistema móvel.

Essa plataforma permitirá que os gestores tenham acesso a uma visão consolidada dos dados coletados em campo. Ela terá como objetivo principal centralizar as informações processadas pelo aplicativo, permitindo análises mais aprofundadas, a geração de relatórios customizados e a gestão avançada de dados, como ajuste de registros e acompanhamentos de indicadores de saúde.

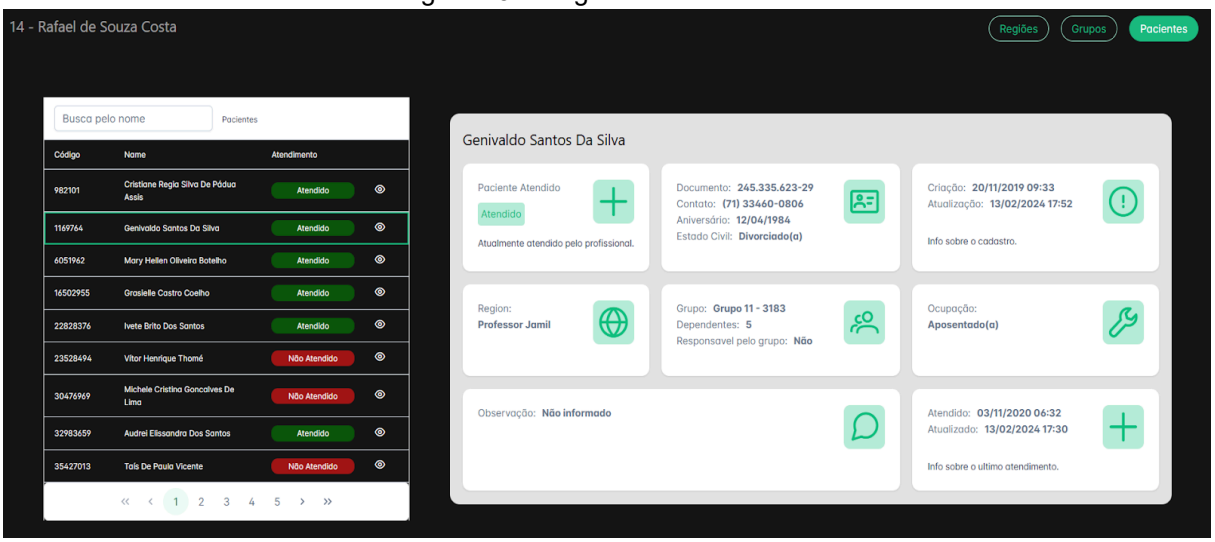
Embora o planejamento e a estrutura inicial para a plataforma já estejam em desenvolvimento, este tópico foi reservado para trabalhos futuros devido a sua complexidade e ao mesmo tempo necessário para implementar todas as funcionalidades propostas. Aspectos como segurança de dados, escalabilidade e responsividade serão tratados como prioridade para garantir que a plataforma atenda às exigências do contexto de saúde pública.

O desenvolvimento de uma Plataforma Centralizada de Gestão representa um passo significativo em direção à otimização e eficiência na gestão dos dados coletados pelo aplicativo. Trata-se de uma plataforma web que concentra todas as informações processadas e armazenadas pelo sistema, proporcionando aos gestores autorizados acesso às informações e a capacidade de realizar ajustes conforme necessário. A implementação dessa plataforma oferecerá uma análise mais precisa e abrangente dos dados coletados, contribuindo para uma tomada de decisão mais informada e eficaz.

Na Figura 25, é possível visualizar uma prévia da página *web* atualmente em desenvolvimento. Apesar de ainda estar em fase inicial, essa interface já apresenta a capacidade de tratar dados sincronizados diretamente da aplicação Saúde do Interior. Essa funcionalidade demonstra o potencial de integração entre a plataforma web e o aplicativo, evidenciando como as informações coletadas em campo podem

ser processadas e visualizadas em um ambiente centralizado, contribuindo para análises mais detalhadas e uma gestão eficiente.

Figura 25 – Página Web Pacientes



Fonte: Figura do Autor.

Tecnologias no Desenvolvimento da Plataforma Web

Para o desenvolvimento futuro da plataforma *web* do projeto Saúde do Interior, planeja-se a utilização de tecnologias que favoreçam a criação de uma interface moderna, funcional e eficiente, além de permitir uma integração robusta com o aplicativo móvel e o banco de dados centralizado.

O *framework* Vue.js foi adotado para o desenvolvimento da interface de usuário, devido à sua capacidade de criar componentes reativos e reutilizáveis, proporcionando uma experiência intuitiva para os gestores. Além disso, sua estrutura modular permitirá uma evolução gradual da plataforma, alinhando-se ao objetivo de expansão e melhorias contínuas.

A comunicação entre o *frontend* (interface do usuário) da plataforma e a API do projeto será realizada utilizando *Axios*, uma biblioteca amplamente reconhecida por sua simplicidade e eficiência na realização de requisições HTTP. Essa integração será essencial para sincronizar os dados coletados pelo aplicativo e possibilitar o gerenciamento centralizado de informações.

Outras tecnologias e ferramentas complementares serão avaliadas e aplicadas conforme necessário, visando atender às demandas específicas do projeto. A escolha dessas tecnologias não apenas facilitará o desenvolvimento, mas também garantirá a escalabilidade e a manutenção da plataforma a longo prazo.

Este planejamento reforça o compromisso do projeto em oferecer uma solução completa e integrada, unindo os dados coletados em campo a uma interface centralizada que permita análises detalhadas e tomadas de decisão mais informadas.

4.1.2. Adicionando Grupos

Conforme detalhado na seção 3.5.4 - Desenvolvimento do Mapa, uma das regras de negócio mais importantes do aplicativo “Saúde do Interior” é a exigência de que o cadastro de grupos seja realizado exclusivamente em campo, ou seja, na residência do paciente. Esse requisito foi implementado para assegurar a precisão das informações geográficas e evitar potenciais fraudes, como a realização de cadastros fictícios por agentes sem que tenham efetivamente visitado o local.

Ao exigir a coleta em tempo real da coordenada geográfica durante o registro, o sistema reforça a integridade dos dados e garante que cada grupo cadastrado esteja associado a uma localização real e verificável. Essa abordagem é especialmente importante no contexto de gestão de saúde em áreas rurais, onde o planejamento e a alocação de recursos dependem diretamente da confiabilidade das informações registradas.

Paralelamente, para atender a situações excepcionais, a Plataforma Centralizada de Gestão permitirá que gestores autorizados realizem cadastros de novos grupos, desde que sejam atendidos critérios rigorosos de segurança e homologação. Esse cadastro poderá incluir a inserção da geolocalização no momento do registro ou em uma etapa posterior. No entanto, caso a localização exata não seja registrada no momento da criação, o sistema marcará o cadastro como "pendente". Nessa situação, o grupo ficará visível na plataforma, mas a inclusão de seus dependentes e o uso pleno das funcionalidades relacionadas a ele serão restringidos até que a geolocalização seja confirmada. Essa estrutura visa equilibrar a flexibilidade no uso do sistema com a necessidade de garantir dados

confiáveis e alinhados às políticas de saúde pública, preservando a integridade do processo de cadastramento e mitigando riscos de fraudes ou inconsistências.

4.1.3. Histórico de Serviços na Aplicação

A manutenção de um histórico de serviços na aplicação desempenha um papel fundamental na continuidade do acompanhamento de pacientes. Inicialmente, o sistema foi projetado para exibir apenas o serviço mais recente de cada paciente, visando reduzir redundâncias e otimizar o armazenamento de informações. No entanto, essa abordagem limitava a análise evolutiva do atendimento prestado.

Com a implementação da funcionalidade de histórico dos três últimos serviços registrados, a aplicação passou a oferecer um registro contínuo e estruturado dos atendimentos, permitindo que os agentes de saúde consultem informações anteriores e comparem a evolução do quadro clínico de cada paciente. Isso proporciona maior segurança na tomada de decisões e facilita a identificação de padrões que podem ser essenciais no diagnóstico e no planejamento terapêutico.

Além disso, essa funcionalidade reforça a importância da rastreabilidade das informações médicas, garantindo que os dados não sejam simplesmente substituídos, mas sim organizados de forma a oferecer uma visão mais ampla do histórico do paciente. Isso contribui para um atendimento mais qualificado, pois os profissionais podem basear suas ações em um contexto mais completo e detalhado.

Com essa evolução, o "Saúde do Interior" se consolida como uma ferramenta que não apenas registra atendimentos, mas também auxilia na gestão eficiente da saúde rural, garantindo que cada paciente receba um acompanhamento adequado e contínuo. Dessa forma, o sistema se alinha às necessidades dos profissionais de saúde, promovendo um fluxo de trabalho mais organizado e eficiente, podendo impactar positivamente na qualidade do atendimento prestado.

5. REFERÊNCIAS

Carvalho, Thiago Leite e. Orientação a Objetos. Edição de Vivian Matsui e Carlos Felício. São Paulo: Casa do Código, 2020. ISBN 978-85-5519-213-5.

Express, (OPENJS FOUNDATION). *Express.js Documentation*. Disponível em: https://expressjs.com/. Acesso em: 3 mar. 2024.

Fielding, R. T. *Architectural Styles and the Design of Network-based Software Architectures.* Dissertação (Doutorado em Ciências da Computação) – University of California, Irvine, 2000. Disponível em: https://ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf. Acesso em: 14 set. 2024.

Flutter, (GOOGLE). *Flutter Documentation*. Disponível em: https://flutter.dev. Acesso em: 3 mar. 2024.

Fowler, Martin. *UML Essencial: Um Breve Guia para Linguagem Padrão.* Tradução de João Tortello. [S.I.]: Bookman Editora, 2014. Disponível em: https://www.kufunda.net/publicdocs/UML%20Essencial%20(Martin%20Fowler).pdf. Acesso em: 09 fev 2025.

GASPAROTTO, Henrique Machado. POO: *Os 4 pilares da Programação Orientada a Objetos*. **DevMedia**, 2014. Disponível em: https://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264 . Acesso em: 3 mar. 2024.

API REST. IBM. Disponível em: https://www.ibm.com/br-pt/topics/rest-apis. Acesso em: 3 mar. 2024.

IBM. *Rational Software Modeler*. Disponível em: https://www.ibm.com/docs/pt-br/rsm/7.5.0?topic=diagrams-use-case. Acesso em: 23 mar. 2025.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA (IBGE). Professor Jamil – GO: panorama das cidades. Disponível em: https://cidades.ibge.gov.br/brasil/go/professor-jamil/panorama. Acesso em: 16 set 2024.

JONES, Michael B. **RFC 7519: JSON Web Token (JWT**). Internet Engineering Task Force, maio 2015. Disponível em: https://datatracker.ietf.org/doc/html/rfc7519. Acesso em: 15 set. 2025.

MYSQL, (ORACLE). *MySQL Documentation*. Disponível em: https://dev.mysql.com/doc/. Acesso em: 3 mar. 2024.

NODE.JS, (OPENJS FOUNDATION). *Node.js v22.0.0 Documentation*. Disponível em: https://nodejs.org/docs/latest/api/. Acesso em: 3 mar. 2024.

OpenStreetMap. *OpenStreetMap Project.* Disponível em: https://www.openstreetmap.org/. Acesso em: 30 abr. 2024.

Flutter, (GOOGLE). Skia Documentation. Disponível em: https://skia.org/docs/. Acesso em: 17 mar. 2025.

SQLite. *SQLite Documentation*. Disponível em: https://www.sqlite.org/docs.html. Acesso em: 10 fev. 2025.

Swagger, (SmartBear). *Swagger Documentation.* Disponível em: https://swagger.io/docs/. Acesso em: 16 set. 2024.

Tekartik. SQFLite Documentation, GitHub. Disponível em: <https://github.com/tekartik/sqflite/tree/master/sqflite/doc>. Acesso em: 10 Fev 2025.

ANEXO I - TABELAS EXPLICATIVAS

ATORES

Tabela 1 – Especificação de Casos de Uso Atores

Atores	Descrição
Usuário	O ator Usuário engloba os profissionais de saúde e outros usuários autorizados que interagem diretamente com o sistema “Saúde do Interior”.
Servidor API	Por meio da API, o sistema pode sincronizar dados com outras fontes, como sistemas de saúde regionais ou banco de dados centralizados. Essa sincronização permite que o aplicativo obtenha informações atualizadas e mantenha a consistência dos dados em tempo real, garantindo uma visão abrangente e precisa da situação da saúde na região rural.

Fonte: Tabela do Autor

UC1 - *LOGIN*

Tabela 2 – Especificação de Casos de Uso UC1

Diagrama de Caso de Uso	Aplicativo Saúde do Interior
Caso de Uso Geral	*Login*
Objetivo	Permitir que usuários autorizados efetue *login*.
Ator Principal	Usuário
Prioridade	Alta
Frequência de uso	Regular

Condições de Entrada	O usuário informa login e senha para acessar.
Fluxo Principal	Acessa a tela principal da aplicação, permitindo realizar as operações de cadastros, consultas, atendimentos, emissão de relatórios e sincronização de dados com o servidor.
Pós-condições	Realizar Cadastro Realizar Consulta Realizar Atendimento Gerar Relatório Sincronizar Dados

Fonte: Tabela do Autor

UC2 - REALIZAR CADASTRO

Tabela 3 – Especificação de Casos de Uso UC2

Diagrama de Caso de Uso	Aplicativo Saúde do Interior
Caso de Uso Geral	Realizar Cadastro
Objetivo	Permitir que usuários realizem cadastro
Ator Principal	Usuário
Frequência de uso	Regular
Condições de Entrada	O usuário tem autonomia de selecionar o tipo de dados que deseja registrar no sistema, conforme as necessidades. O usuário tem as seguintes opções de cadastro: regiões, grupos, pacientes, ocupação.
Fluxo Principal	Escolhido o tipo de dado o sistema apresenta um formulário para ser preenchido de acordo com a escolha:

	1. Região – Nome 2. Grupo – Nome – Região – Localização 3. Paciente – Nome – CPF – Data de Aniversário – Grupo – Estado Civil – Ocupação – Contato – Observação – Religião – Responsável pelo grupo (checkbox) 4. Ocupação – Nome
Pós-condições	Realizar Consulta Realizar Atendimento Gerar Relatório Sincronizar Dados

Fonte: Tabela do Autor

UC3 - REALIZAR CONSULTA

Tabela 4 – Especificação de Casos de Uso UC3

Diagrama de Caso de Uso	Aplicativo Saúde do Interior
Caso de Uso Geral	Realizar Consulta
Objetivo	Faculta ao usuário consultar as informações armazenadas no sistema, proporcionando acesso rápido e eficiente aos dados cadastrados.
Ator Principal	Usuário
Frequência de uso	Frequente
Condições de Entrada	O usuário tem autonomia de selecionar o tipo de dados que deseja consultar no sistema, conforme as necessidades. O usuário tem as seguintes opções de consultas: regiões, grupos, pacientes.
Fluxo Principal	Após selecionar o tipo de dado desejado, o sistema exibe todas as informações correspondentes cadastradas. Isso possibilita ao usuário realizar edições e exclusões conforme necessário, garantindo flexibilidade e controle sobre os dados armazenados.
Pós-condições	Realizar Atendimento Gerar Relatório Sincronizar Dados

Fonte: Tabela do Autor

UC4 - REALIZAR ATENDIMENTO

Tabela 5 – Especificação de Casos de Uso UC4

Diagrama de Caso de Uso	Aplicativo Saúde do Interior
Caso de Uso Geral	Realizar Atendimento
Objetivo:	Permite ao usuário realizar um atendimento, coletando e armazenando de forma precisa e organizada todas as informações relevantes pré-cadastradas no formulário disponível.
Ator Principal	Usuário
Frequência de uso	Essencial
Condições de Entrada	Usuário seleciona o paciente a ser atendido e inicia o registro do atendimento
Fluxo Principal	O sistema apresentará um formulário com questões dinâmicas previamente cadastradas, permitindo ao usuário preencher e salvar o atendimento com as informações relevantes coletadas durante a consulta.
Pós-condições	Gerar Relatório Sincronizar Dados

Fonte: Tabela do Autor

UC5 - GERAR RELATÓRIO

Tabela 6 – Especificação de Casos de Uso UC5

Diagrama de Caso de Uso	Aplicativo Saúde do Interior
Caso de Uso Geral	Gerar Relatório
Objetivo	Permite ao usuário gerar e compartilhar, em formato PDF, relatórios detalhados dos dados armazenados previamente no sistema, oferecendo uma visão abrangente e organizada das informações relevantes.
Ator Principal	Usuário
Frequência de uso	Regular
Condições de Entrada	O usuário aciona o botão 'Export' para gerar o relatório.
Fluxo Principal	O sistema gera um arquivo PDF, permitindo a visualização dos dados ou o compartilhamento do relatório.
Pós-condições	Retorna ao estado anterior. O sistema permite a emissão de outros relatórios.

Fonte: Tabela do Autor

UC6 - SINCRONIZAR DADOS

Tabela 7 – Especificação de Casos de Uso UC6

Diagrama de Caso de Uso	Aplicativo Saúde do Interior
Caso de Uso Geral	Sincronizar Dados
Objetivo	Permite ao usuário realizar a sincronização dos dados com o banco de dados online, garantindo o envio e recebimento de informações atualizadas.
Ator Principal	Usuário
Frequência de uso	Essencial
Condições de Entrada	O usuário aciona o botão sincronizar.
Fluxo Principal	O sistema identifica as alterações nos dados,enviando essas modificações para o servidor API. Após a conclusão bem-sucedida desse processo, solicita ao servidor os dados que estejam desatualizados no aplicativo.
Pós-condições	O sistema está totalmente atualizado com as últimas informações disponíveis.

Fonte: Tabela do Autor