INSTITUTO FEDERAL SUL-RIO-GRANDENSE - IFSUL
***CÂMPUS* CHARQUEADAS**
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

GABRIEL DA COSTA BAPTISTA

AMPLIAÇÃO DE UM ECOSSISTEMA FHIR COM SUPORTE A SMART ON FHIR E VISUALIZAÇÃO WEB DE EXAMES DE ECG

Orientador: Prof. Dr. André Luís Del Mestre Martins

CHARQUEADAS, 2025

RESUMO

A interoperabilidade entre sistemas de saúde é fundamental para garantir a troca segura e padronizada de informações clínicas. Este trabalho apresenta o desenvolvimento de uma aplicação web interoperável para acompanhamento médico de eletrocardiogramas (ECGs), baseada no padrão FHIR (Fast Healthcare Interoperability Resources) e no protocolo de segurança SMART on FHIR. O projeto teve como ponto de partida três sistemas independentes do grupo IF4Health: FASS-ECG (API FHIR para ECGs), H2Cloud (autenticação SMART on FHIR) e ECG Monitor (visualizador de sinais). Inicialmente, os sistemas foram unificados em uma única API, com melhorias estruturais e suporte a novos recursos FHIR, como *Practitioner* e *Bundle*. Em seguida, foram implementadas rotinas de autenticação e criação de usuários com escopos de autorização, viabilizando o controle de acesso baseado em papéis. Por fim, foi desenvolvido o *Biomonitor*, uma nova aplicação front-end moderna, integrada à API autenticada. A plataforma está documentada, com código-fonte aberto e disponível para testes.

Palavras-chave: ECG. Eletrocardiogramas. telemedicina. FHIR. Front-end.

ABSTRACT

Interoperability between healthcare systems is essential for the secure and standardized exchange of clinical information. This work presents the development of an interoperable web application for medical monitoring of electrocardiograms (ECGs), based on the FHIR (Fast Healthcare Interoperability Resources) standard and the SMART on FHIR security protocol. The project originated from three independent systems developed by the IF4Health group: FASS-ECG (FHIR API for ECGs), H2Cloud (SMART on FHIR authentication), and ECG Monitor (signal visualizer). These systems were unified into a single API, with structural improvements and support for additional FHIR resources such as *Practitioner* and *Bundle*. Authentication and user creation routines with scope-based access control were implemented, enabling role-based authorization. Finally, a new modern front-end called *Biomonitor* was developed and integrated with the authenticated API. The platform is documented, open-source, and publicly available for testing.

Keywords: ECG. EKG. Electrocardiogram. telemedicine. FHIR. Front-end.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS-FONTE

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
AWS	Amazon Web Services
AVL	Augmented Vector Left
AVR	Augmented Vector Right
AVF	Augmented Vector Foot
BPM	Batimento por Minuto
ECG	Eletrocardiograma
EHR	Eletronic Health Records
FHIR	Fast Health Interoperability Standards
HTTP	Hypertext Transfer Protocol
IFSul	Instituto Federal Sul-rio-grandense
JSON	JavaScript Object Notation
PWA	Progressive Web App
REST	Representational State Transfer
TSI	Tecnólogo em Sistemas para Internet
XML	Extensible Markup Language
UTI	Unidade de Tratamento Intensivo

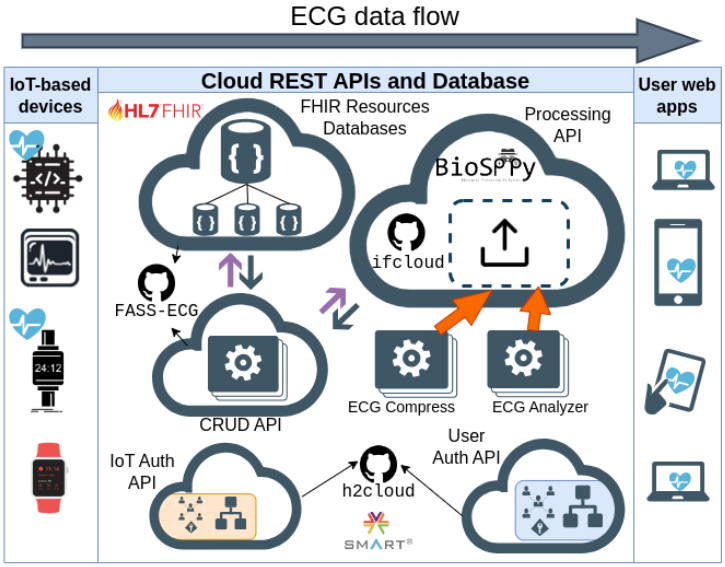
SUMÁRIO

1 INTRODUÇÃO

Informações de saúde são um dos bens mais sensíveis e valiosos da era digital. Ao contrário de dados bancários, que podem ser revertidos em casos de fraude, a exposição indevida de dados clínicos pode comprometer diagnósticos, tratamentos e a privacidade dos indivíduos. Apesar da crescente adoção do padrão *FHIR* (Seção 2.4) para interoperabilidade entre sistemas de saúde, o fato de ele não definir mecanismos de controle de acesso representa uma grave fragilidade diante da natureza crítica dos Prontuários Eletrônicos do Paciente (PEPs) e das exigências legais de proteção de dados (SANTOS et al., 2023).

Apesar da interoperabilidade de dados em saúde ser uma demanda global, o ecossistema brasileiro de Saúde Digital enfrenta desafios cruciais para a integração e a disponibilização dos PEPs (MÉLO et al., 2024). Um PEP é controlado individualmente e potencialmente combina informações de vários sistemas com dados coletados de diversas instituições de saúde e, hoje em dia, até por dispositivos vestíveis. Além das dificuldades técnicas para integração dessas informações (ROEHRS; DA COSTA; ROSA RIGHI, 2017), também é um ponto crítico a gestão de consentimento (KAKARLAPUDI; MAHMOUD, 2021) - ou seja, conjunto de ferramentas que garantem que os pacientes possam autorizar, negar ou revogar o uso de seus dados de saúde, de forma segura, transparente e controlada - verdadeiramente centrada no paciente. Este trabalho foca na autorização de dados, incluindo a complexidade técnica de integrar diferentes Registros Eletrônicos de Saúde em um PEP.

Figura 1.1: Visão geral do ecossistema de serviços e fluxo IF4Health



Desde 2020, o grupo de pesquisa IF4Health tem se dedicado ao desenvolvimento de soluções em nuvem para aplicações de saúde digital, dando ênfase a conectividade dos sistemas. A Figura 1.1 apresenta uma visão geral da arquitetura em desenvolvimento, que integra dispositivos IoT para aquisição de biossinais (exemplo: eletrocardiograma – ECG), APIs para armazenamento e processamento (VIEIRA et al., 2024) desses sinais na nuvem, e aplicações

web para sua visualização remota. Entre os projetos já implementados, destacam-se o *FASS-ECG*, responsável pela interoperabilidade dos dados via padrão FHIR, e o *ECG Monitor*, um *front-end* voltado à exibição interativa dos exames.

FASS-ECG (PEREIRA et al., 2023) atualmente suporta apenas dois recursos FHIR, *Patient* e *Observation*, o que é insuficiente para implementar um sistema clínico simples ou completo, considerando a complexidade envolvida em aplicações de saúde. O padrão FHIR define mais de 200 tipos de recursos distintos, incluindo *Practitioner*, *Bundle*, *Encounter*, entre outros que são necessários para compor uma aplicação clínica realista.

No quesito de segurança, o servidor H2Cloud, desenvolvido pelo grupo IF4Health, implementa com sucesso o protocolo de autenticação SMART on FHIR (Seção 2.5). No entanto, embora permita autenticar usuários previamente cadastrados, ele não conta com um fluxo de criação de novos usuários, nem conta com mecanismos mais robustos de autorização baseados em escopos. Isso limita a capacidade da aplicação de gerenciar perfis distintos, como pacientes e profissionais de saúde, e restringe o acesso a funcionalidades mais avançadas conforme os papéis de cada usuário.

Avançar no desenvolvimento de uma aplicação web compatível com FHIR demanda um back-end capaz de lidar com recursos diversos. Além disso, é essencial garantir que cada usuário tenha acesso apenas aos dados necessários, o que requer controle de escopo e autorização granular. Este trabalho tem como principal desafio dar suporte a esse desenvolvimento por meio da ampliação do servidor FASS-ECG e da integração completa com o H2Cloud, criando uma plataforma segura, interoperável e funcional para o acompanhamento médico de dados de ECG, e possibilitando o objetivo final de expandir as capacidades do ECG Monitor.

1.1 Objetivo

Evoluir funcionalidades centrais do ecossistema IF4Health, incorporando autenticação e autorização SMART on FHIR e ampliando o front-end ECGMonitor com novos fluxos e capacidades.

1.1.1 Objetivos específicos

O presente trabalho tem os seguintes objetivos específicos:

1. Compreender as APIs de saúde digital do Grupo IF4Health de onde os dados serão consumidos;
2. Atualizar o back-end do FASS-ECG para suportar novos recursos FHIR;
3. Suportar a funcionalidade autenticação de usuário SMART on FHIR;
4. Permitir a criação de novos usuários SMART on FHIR;
5. Atribuir as Autorizações adequadas aos usuários SMART on FHIR;

6. Publicar o codigo-fonte no Git-Hub (https://github.com/if4health) com documentacao adequada (readme.md);
7. Permitir a visualização do monitor ECG do paciente na plataforma;
8. Adicionar funcionalidades mínimas para tornar o ECG Monitor em uma aplicação web, como seleção de diferentes ECGs e pacientes;

1.2 Organização do Trabalho

O Capítulo 2 apresenta os conceitos específicos necessários para a compreensão deste trabalho. O Capítulo 3 detalha o desenvolvimento parcial do trabalho, deixando claro qual o ponto de partida deste trabalho considerando o ecossistema IF4Health. Finalmente, o Capítulo 4 apresenta os resultados deste trabalho.

2 REFERENCIAL TEÓRICO

Esse capítulo apresenta os principais conceitos e tecnologias necessários para compreensão deste trabalho. O ecossistema de aplicações da Figura 1.1 é o estudo de caso deste trabalho. Portanto, são descritas as principais aplicações do grupo IF4Health, como o FASS-ECG (Seção 2.1), o H2Cloud (Seção 2.2) e o ECGMonitor (Seção 2.3). Por fim, são introduzidos os padrões e protocolos utilizados nestes projetos para garantir a interoperabilidade, segurança e integridade dos dados clínicos (Secoes 2.4 e 2.5).

2.1 API FASS-ECG

FASS-ECG é uma solução interoperável voltada à transmissão e ao armazenamento de longo prazo de sinais eletrocardiográficos (ECG), utilizando como base o protocolo FHIR de forma a garantir a compatibilidade com uma ampla gama de aplicações e plataformas com a mesma padronização (PEREIRA et al., 2023).

A API disponibiliza endpoints que permitem tanto a escrita quanto a leitura de dados ECG. A escrita dos dados é estruturada em três etapas:

- Um request POST para iniciar a sessão de transmissão;
- Uma sequência de requests PATCH contendo os blocos de dados coletados progressivamente;
- Um request PUT finalizando a sessão e consolidando o registro.

A API também disponibiliza três rotas principais para leitura de dados, sendo elas:

- `GET /Observation/id`: retorna o conteúdo completo de um recurso Observation, incluindo todas as derivações e amostras registradas.
- `GET /Observation/{id}/data/{minute}`: retorna apenas as amostras de um determinado minuto do ECG.
- `GET /Observation/{id}/data/{start}/{end}`: retorna todas as amostras dentro de um intervalo de minutos especificado.

Apenas a primeira rota segue o padrão oficial de acesso a recursos no FHIR. As demais foram desenvolvidas como extensões personalizadas, com o objetivo de lidar eficientemente com o grande volume de dados característico de sinais eletrocardiográficos. Além disso, essas extensões são especialmente úteis na implementação de visualizadores interativos, permitindo a exibição segmentada ou contínua dos registros.

Como parte do processo de validação e teste da conformidade com o padrão FHIR (Seção 2.4), as requisições realizadas com a FASS-ECG são comparadas com as respostas obtidas por meio do HAPI FHIR (CDR, 2019), um servidor público de código aberto amplamente utilizado para testes de interoperabilidade.

2.2 API H2Cloud

O *H2Cloud* (*Heterogeneous Health Cloud*) é uma solução em nuvem para autenticação de aplicações web de saúde, utilizando os protocolos estabelecidos pelo SMART on FHIR (SANTOS et al., 2023).

O servidor atua como uma camada intermediária heterogênea, responsável por prover a autenticação entre o cliente web e o servidor FHIR que hospeda os dados clínicos. Para isso, ele oferece suporte aos dois principais fluxos do SMART on FHIR: SMART App Launch, para aplicações com interfaces de usuário, e o SMART Backend Services, para dispositivos e serviços sem interações diretas com o usuário (Seção 2.5).

Através do H2Cloud, viabiliza-se a integração de dispositivos IoT, aplicativos, sistemas internos, todos operando de forma autenticada, e com controle de acesso, favorecendo a interoperabilidade e segurança no ecossistema da saúde digital.

2.3 ECG Monitor

O *ECG Monitor* é um aplicativo *front-end* desenvolvido para a visualização dinâmica e estática de eletrocardiogramas, utilizando como base o protocolo *FHIR* para consumo dos dados clínicos (DUARTE, 2024). O projeto apresenta duas interfaces principais para visualizar eletrocardiogramas: uma voltada a exibição de forma dinâmica dos sinais (Figura 2.1), apresentando a onda animada em tempo real e conta também com monitor de batimentos por minuto; e outra com uma visualização estática, simulando o tradicional papel quadriculado usado em exames impressos.

Figura 2.1: Tela com o visualizador dinâmico do ECG Monitor

O projeto foi desenvolvido com a biblioteca *React*, integrando-se ao uso do stack de desenvolvimento MERN (MongoDB, Express, React, NodeJS), que outras aplicações do grupo

IF4Health utilizam. Um dos méritos do trabalho foi a criteriosa exploração e avaliação de diferentes bibliotecas gráficas voltadas à visualização de dados, o que permitiu identificar as soluções mais adequadas para cada tipo de exibição dos sínais eletrocardiográficos. Como resultado, foram adotadas duas bibliotecas especializadas: a *SciChart*, para visualização dinâmica com alto desempenho gráfico, e a *CanvasJS*, utilizada para a visualização estática dos sinais.

O aplicativo busca os dados necessários por meio da *API FASS-ECG*, que conta com funcionalidades de *streaming* para os sinais eletrocardiográficos. Essa funcionalidade permite a transmissão contínua dos dados, reduzindo significativamente a latência nas requisições e otimizando o desempenho do visualizador no processamento e exibição dos dados.

Apesar de sua utilidade na visualização dos sinais de eletrocardiogramas, o ECG Monitor apresenta limitações significativas em termos de funcionalidades. A aplicação atua exclusivamente como visualizador de sinais, permitindo apenas a troca de diferentes derivações, sem exibição simultânea de múltiplas derivações em um mesmo painel. Além disso, não existem mecanismos para seleção de pacientes ou médicos, nem funcionalidade para inserção de laudos, preenchimento de formulários ou qualquer tipo de entrada de dados. Devido a essa simplicidade e restrição do sistema, sequer existia a necessidade para autenticação de usuário. Essas restrições evidenciam a necessidade de uma evolução da plataforma para atender a fluxos clínicos mais complexos.

2.4 Padrão FHIR

O padrão *FHIR* (*Fast Healthcare Interoperability Resources*), é um protocolo de interoperabilidade estabelecido pela organização internacional *HL7 International*, buscando padronizar os meios de troca de dados clínicos num contexto global moderno. Desenvolvido a partir de 2011, o FHIR propõe um modelo mais acessível e eficiente de integração entre sistemas de informação de saúde (HL7, 2019).

O padrão define regras e estruturas pensadas para o armazenamento e a transmissão de dados clínicos, tais como dados de pacientes, resultados de exames, laudos médicos, prescrições, procedimentos, estruturas organizacionais (como unidades e hierarquias hospitalares), entre outros. Sua proposta central é promover uma linguagem central para a comunicação entre esses sistemas, independente de tecnologias ou fornecedores utilizados.

Uma de suas principais características é sua implementação baseada em formato *RESTful* (*Representational Data Transfer*), muito utilizado em APIs modernas, modelando dados como recursos que podem ser transacionados por requisições HTTP simples. Os recursos são unidades fundamentais do protocolo, existindo vários tipos que encapsulam dados diferentes, como recursos *Patient*, que representam dados de pacientes, ou recursos *Observation*, que podem representar sinais vitais, dados laboratoriais, e principalmente no contexto destes trabalhos, sinais eletrocardiográficos.

O Código-Fonte 2.1 apresenta um exemplo de recurso Practitioner em formato JSON. No exemplo, observa-se como são organizadas informações como nome, prefixo e qualificações do profissional, estruturadas de forma consistente e extensível.

Código-fonte 2.1: Exemplo de um recurso FHIR *Practitioner* em formato JSON.

```
{
    "resourceType": "Practitioner",
    "name": [{
      "family": "de Tal",
      "given": ["Fulano"],
      "prefix": ["Dr"]
    }],
    "qualification": [{
      "code": {
        "coding": [{
          "system": "http://www.saude.gov.br/fhir/r4/CodeSyst⌋
           ↪ em/BRCBO-1.0",
          "code": "225125",
          "display": "Médico clínico"
        }]
      }
    }]
}
```

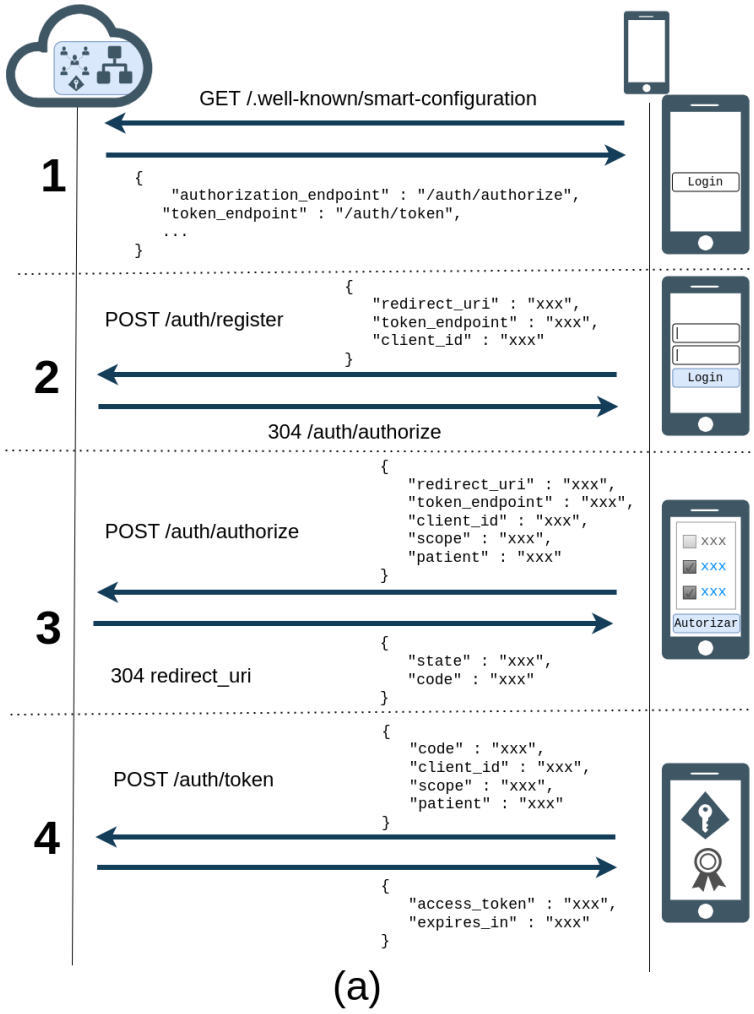
2.5 SMART on FHIR

Em questão de segurança de dados, a organização HL7 recomenda a implementação do protocolo *SMART on FHIR*, uma especificação complementar ao FHIR, desenvolvida com o objetivo de fornecer uma camada de segurança para aplicações que consomem APIs FHIR, ao incorporar padrões modernos de autenticação e autorização, como o *OAuth 2.0* e o *OpenID* (MANDEL et al., 2016).

A especificação define dois fluxos distintos de autenticação: o *SMART App Launch*, voltado para aplicações com interface de usuário (como portais e aplicativos móveis), e o *SMART Backend Services*, direcionado a aplicações sem interação humana direta, como dispositivos IoT ou serviços de backend.

O processo típico de autenticação segue os princípios do OAuth 2.0, conforme ilustrado na Figura 2.2. Inicialmente (Etapa 1), é feita a descoberta automática das capacidades do servidor, por meio de uma rota `/.well-known/smart-configuration`, que expõe endpoints de autorização, geração de tokens e as capacidades de segurança do servidor. Em seguida, é realizada a autenticação de usuário por meio de uma interface de login fornecida pelo servidor de autenticação (Etapa 2). Após a autenticação, o usuário é solicitado a autorizar e consentir

Figura 2.2: Fluxo de autenticação e autorização SMART on FHIR



o acesso da aplicação a determinados escopos, como a leitura e escrita de dados (Etapa 3). Quando concedida a autorização, o servidor emite um código de autorização, que será devolvido pela aplicação para finalmente obter o token de acesso definitivo (Etapa 4).

Para aplicações sem interface de usuário, como dispositivos previamente registrados, o fluxo é simplificado, utilizando o mecanismo *client credentials*, em que o dispositivo apresenta seu identificador e segredo previamente cadastrados, e o servidor emite diretamente o token de acesso correspondente, eliminando a etapa de login e consentimento (Etapas 2 e 3 da Figura 2.2).

Esse fluxo garante que apenas aplicativos autenticados e autorizados possam acessar dados sensíveis em servidores FHIR. A autorização é controlada por meio de escopos granulares, definidos no momento de acesso. Esses escopos representam permissões específicas que delimitam as ações permitidas pela aplicação e os tipos de recursos FHIR que podem ser acessados.

Tais mecanismos são fundamentais para assegurar a segurança e a privacidade dos dados clínicos, promovendo um acesso controlado, transparente e baseado em consentimento. Ao permitir que apenas aplicações confiáveis e com permissões explicitamente concedidas acessem informações sensíveis, o modelo reforça a proteção contra acessos indevidos, vazamentos

ou usos não autorizados.

2.5.1 Autenticação

A autenticação, no contexto SMART on FHIR, se refere ao processo pela qual a aplicação comprova a identidade do usuário ou sistema que está tentando acessar dados clínicos protegidos. Nesse protocolo, o processo é baseado no fluxo OAuth 2.0, onde o usuário é redirecionado ao servidor de autenticação, realiza o login e recebe um token de acesso que servirá como prova de identidade nas requisições seguintes. Esse mecanismo e todas suas etapas já foram completamente desenvolvidas e implementadas no ecossistema IF4Health, encontrando-se detalhado no trabalho de PEREIRA (2022). Por essa razão, esse fluxo não será aprofundado neste trabalho, focando somente no próximo aspecto crucial de segurança, a autorização por escopos.

2.5.2 Autorização

A autorização, no contexto SMART on FHIR, determina o que o usuário pode realmente fazer após estar devidamente identificado no sistema, definindo com quais recursos FHIR o mesmo pode interagir e quais operações em específico lhe são permitidas em cada um.

No SMART on FHIR, os escopos seguem uma estrutura padronizada em três partes:

- Um prefixo, que define o contexto de acesso;
- o tipo de recurso FHIR;
- as operações permitidas no recurso.

Essa estrutura aparece em escopos tais como:

- `patient/Observation.rs`
- `user/Patient.cruds`
- `system/*.cruds`

Cada elemento desempenha um papel específico no controle de acesso. No caso dos prefixos, *patient/* restringe o usuário aos seus próprios recursos, *user/* identifica um profissional ou agente que pode acessar dados relacionados ao seu exercício funcional, e *system/* é reservado para serviços automatizados sem interação humana. Já o sufixo de operações corresponde aos tipos de requisição permitidos: *c* para criação, *r* para leitura, *u* para atualização, *d* para remoção, e *s* para busca, formando combinações como *cru*, *cruds* ou *rs*, conforme o nível de permissão concedido pelo servidor SMART on FHIR.

Por exemplo, um escopo como `patient/Observation.read` concede à aplicação permissão apenas para ler recursos do tipo Observation relacionados ao paciente autenticado, enquanto algo como `user/*.*`, um escopo mais amplo, permitiria acesso a todos os tipos de recursos vinculados a um profissional de saúde autenticado.

Esse mecanismo garante que aplicações diferentes, e até usuários diferentes dentro da mesma aplicação, recebam permissões adequadas ao seu papel. Assim, um paciente pode receber escopos restritivos que permitem apenas consultar seus próprios dados, enquanto um profissional de saúde recebe escopos que autorizam leituras mais amplas e operações de escrita, desde que associadas a pacientes sob sua responsabilidade.

No fluxo SMART on FHIR, esses escopos são incluídos no token de acesso emitido após o processo de autenticação, como pode ser visto na etapa 4 da Figura 2.2. A partir desse momento, qualquer requisição feita pelo cliente à API FHIR deve incluir esse token, e cabe ao servidor validar se o escopo ali declarado realmente autoriza aquela operação. Caso contrário, a requisição é negada, preservando a segurança do ecossistema.

Esse modelo reforça o princípio do menor privilégio, fundamental em sistemas clínicos: cada usuário só deve ter acesso exatamente ao que é necessário para sua atividade. Ele também garante isolamento natural entre perfis. Por exemplo, dois médicos autenticados podem acessar apenas os pacientes vinculados a cada um, enquanto os pacientes, ao autenticar-se, veem exclusivamente seus próprios dados.

3 METODOLOGIA E SOLUÇÃO PROPOSTA

Esse capítulo inicia contextualizando o desenvolvimento do trabalho. Seguindo com a implementação de diversas funcionalidades as aplicações que formam sua base, até finalmente apresentar a implementação do sistema.

3.1 Ponto de partida / Preliminares

O desenvolvimento do projeto teve como ponto de partida os sistemas existentes do grupo IF4Health, em especial o *ECGMonitor*, *FASS-ECG*, e *H2Cloud*, (Capítulo 2) todos construídos sobre a arquitetura MERN stack (MongoDB, Express.js, React e Node.js).

Embora o objetivo fosse o aprimoramento do *ECGMonitor*, constatou-se que seria necessário solidificar as APIs que a aplicação utilizaria, em específico, implementar e verificar o funcionamento de capacidades de autenticação.

Até esse ponto, os sistemas apresentavam funcionalidades que cumpriam um papel de base, mas estavam necessitando de uma expansão de suas capacidades para que pudessem atender melhor os novos requisitos propostos, além de não contarem com nenhum tipo de deploy em ambiente online.

3.2 Merge de FASS-ECG e H2Cloud

Na Figura 1.1 é possível observar os sistemas H2Cloud e FASS-ECG como serviços operando separadamente. Com o principal objetivo de reduzir os custos com nuvem e simplificar o *deploy*, optou-se por mesclar estes dois serviços em uma única aplicação. Uma tentativa anterior de junção entre os sistemas FASS-ECG e H2Cloud resultou na criação do sistema denominado *neoFASS-ECG*[1].

Após estudo deste protótipo foi feito um novo branch no projeto, denominado `refactoring`, e mais tarde, `user`, visando versões do projeto em uso para outros projetos. Foram feitas melhorias estruturais e revisões na arquitetura do sistema, com o objetivo de melhor se adequar aos protocolos e fluxos de autorização e autenticação estabelecidos pelo padrão SMART on FHIR.

Entre as melhorias inicialmente implementadas, destacam-se a correção de inconsistências na documentação (como guias de instalação e implantação), a atualização da interface Swagger para uma documentação de API mais clara, a remoção de variáveis de ambiente relacionadas a serviços da AWS, que ancoravam a aplicação em seu uso e disponibilidade, e também uma integração mais livre com banco de dados MongoDB ao permitir o uso direto da URL de conexão, ao invés de exigir um banco com usuário e senha.

[1] `https://github.com/if4health/neoFASS-Ecg`

3.3 Adição de CRUDs de *Bundle* e *Practitioner*

Após as melhorias na base da aplicação, foram implementados funções de criação, leitura, e remoção de recursos do tipo *Practitioner* (profissionais de saúde), bem como suporte à manipulação de *Bundles*, ambos em conformidade as especificações do padrão FHIR.

A inclusão do recurso *Practitioner* é essencial para adicionar funcionalidades na aplicação *front-end* em que os profissionais de saúde são parte.

O suporte a *Bundles*, por sua vez, foi incorporado com o objetivo de facilitar a troca e o processamento de conjuntos de dados clínicos, permitindo que múltiplos recursos FHIR sejam agrupados em uma única estrutura lógica (HL7, 2023). Ferramentas que geram e exportam dados no padrão FHIR, frequentemente usam o formato Bundle para encapsular seus recursos em uma única estrutura. Com essa funcionalidade, a aplicação passa a ser capaz de consumir e interpretar esses pacotes, facilitando processos de importação, testes e integração entre sistemas.

3.3.1 Recurso *Practitioner*

O recurso *Practitioner*, definido pelo padrão FHIR, é utilizado para representar profissionais de saúde, como médicos, enfermeiros, terapeutas e outros indivíduos que prestam serviços clínicos (HL7, 2019). Ele descreve informações essenciais sobre o profissional de saúde, incluindo nome completo, identificadores institucionais, gênero e dados de contato.

Código-Fonte 2.1 mostra um exemplo de recurso FHIR *Practioner* com algumas informações importantes para a definição de um profissional de saúde conforme segue:

- `"resourceType"` - tipo de Recurso FHIR;
- `"name"` - nome completo do profissional;
- `"qualification"` - qualificações do profissional;

3.3.2 Recurso *Bundle*

O recurso *Bundle*, no padrão FHIR, representa uma estrutura de agrupamento que permite encapsular múltiplos recursos em um único pacote de dados. Ele é comumente utilizado para transportar, importar ou exportar coleções de informações clínicas de forma padronizada e coesa. Um *Bundle* pode incluir, por exemplo, os dados de um paciente, seus profissionais vinculados, observações clínicas, dispositivos utilizados e resultados de exames, reunidos em uma única estrutura lógica.

O objetivo inicial da implementação desse recurso era estabelecer um meio simples e padronizado de importar e exportar dados entre diferentes ambientes rodando o projeto, facilitando tanto a migração quanto a reprodução de cenários de teste. O protocolo FHIR define

diversos tipos de Bundle para finalidades específicas, como transações atômicas, resultados de buscas, históricos ou lotes de requisições, porém para este projeto, foi adotado inicialmente o tipo collection, a forma mais simples, utilizada apenas para agrupar recursos sem impor lógica adicional de processamento.

Além disso, planejou-se integrar o sistema a geradores de dados sintéticos, permitindo criar rapidamente conjuntos de pacientes e profissionais para testes, demonstrações ou validações de fluxo, sem depender de inserções manuais no banco de dados. Isso tornaria o ciclo de desenvolvimento mais ágil e permitiria reproduzir cenários completos em diferentes máquinas.

Código-fonte 3.1: Exemplo de um recurso FHIR *Bundle* em formato JSON agrupando três informações em um único registro: 2 pacientes e 1 profissional de saúde.

```
{
  "resourceType": "Bundle",
  "type": "collection",
  "entry": [
    {
      "resource": {
        "resourceType": "Patient",
        ...
      }
    },
    {
      "resource": {
        "resourceType": "Patient",
        ...
      }
    },
    {
      "resource": {
        "resourceType": "Practitioner",
        ...
      }
    }
  ]
}
```

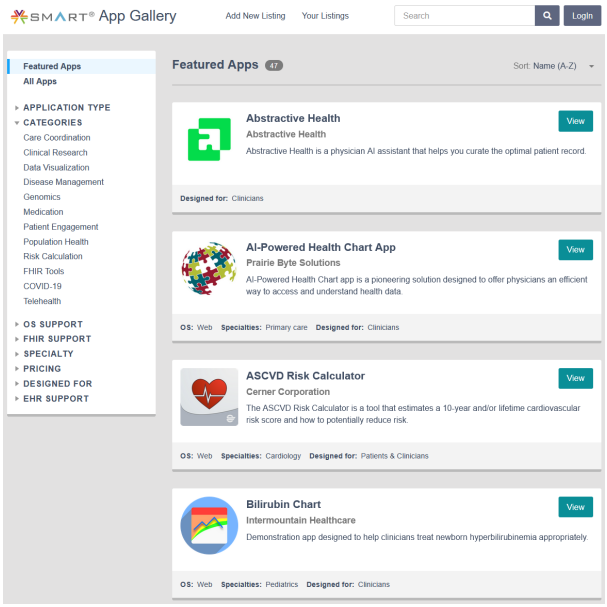
O exemplo do Código-fonte 3.1 foi salvo no servidor público de testes HAPI FHIR (CDR, 2019). Por ser um servidor gratuito e de terceiros, eventualmente os recursos salvos no HAPI FHIR podem estar indisponíveis ou ser deletados. Por enquanto, é possível acessar o *Bundle* do Código-fonte 3.1, na íntegra, por meio da requisição GET a seguir:

```
curl -X 'GET' \
 'https://hapi.fhir.org/baseR4/Bundle/48281004' \
 -H 'accept: application/fhir+json'
```

3.4 Criação de Novo Usuário SMART on FHIR

Observou-se que uma das principais barreiras para a adoção do protocolo SMART on FHIR estava na ausência de um mecanismo de criação de usuários diretamente na aplicação. Essa dificuldade é agravada pela ausência de materiais de referência e guias de implementação de tais sistemas, visto que essa funcionalidade não é encontrada em aplicativos certificados na *SMART App Gallery* (HOSPITAL, 2022), um repositório oficial de aplicativos que utilizam o protocolo SMART on FHIR, servindo como referência para soluções interoperáveis de saúde digital (Figura 3.1). Essa deficiência dificultava a integração de fluxos SMART on FHIR no sistema, uma vez que o cadastro prévio de usuários e dispositivos dependia de registro.

Figura 3.1: Lista de aplicações FHIR na SMART App Gallery



Com base em estudos feitos em cima de implementações mais genéricas do protocolo OAuth 2.0 (Capítulo 2), tecnologia fundamental sobre o qual o SMART on FHIR é construído, foi possível montar um plano de implementação para ser utilizado na aplicação.

Foi necessária a reestruturação do modelo de autenticação anteriormente adotado no *neoFASS-ECG*, separava rigidamente os fluxos de login para pacientes e médicos, utilizando nomenclaturas confusas para o desenvolvimento e dificultando a manutenção e expansão futura. Como solução, implementou-se um modelo **Usuário** unificado, contendo os campos de nome, senha, papel funcional (como paciente ou médico) do usuário e recurso FHIR correspondente.

De forma complementar a implementação do cadastro de novos usuários, foi realizada uma reorganização no fluxo de autenticação de sistemas sem interface de usuários como dispositivos IoT e servicos de back-end. Na versão anterior do sistema, esses sistemas utilizavam *tokens* persistentes, que, por razões desconhecidas, podiam expirar de forma inesperada. Isso resultava em falhas na autenticação, atrasos no uso do sistema e, em muitos casos, exigia a recriação manual dos dispositivos no banco de dados.

Na nova abordagem, os dispositivos passaram a ser autenticados utilizando um fluxo mais comum do OAuth 2.0, no qual o dispositivo envia seu Client ID e Client Secret para obtenção de um *token* de acesso temporário. Esse modelo é mais aderente às práticas recomendadas, além de oferecer maior previsibilidade e controle sobre o ciclo de vida dos tokens.

A partir destas novas estruturas, foram desenvolvidas funcionalidades que permitem o cadastro direto de pacientes, médicos e dispositivos, criando simultaneamente tanto o usuário de autenticação quanto o recurso no padrão FHIR correspondente (*Patient* ou *Practitioner*).

A implementação do cadastro de novos usuários foi integrada diretamente à criação dos recursos FHIR correspondentes. Embora um usuário SMART on FHIR não seja conceitualmente o mesmo que um recurso Patient ou Practitioner, o sistema estabelece uma relação entre ambos no momento do registro. Os dados informado no formulário são convertidos para um JSON FHIR válido, que é enviado e armazenado como recurso FHIR. Em paralelo, o backend cria o usuário de autenticação, definindo seu papel funcional. Essa relação é registrada internamente: o usuário passa a possuir um campo que referencia o identificador do recurso FHIR criado, enquanto o próprio recurso também mantém o identificador do usuário que o originou. Isso garante que ambos permaneçam sincronizados e que o fluxo SMART on FHIR possa aplicar corretamente escopos, filtros e políticas de acesso ao longo do uso do sistema.

No processo de autorização, o controle de acesso baseado em escopos é aplicado diretamente nas consultas realizadas pelo servidor FHIR. Quando o usuário obtém um token contendo escopos do tipo user/ ou patient/, o backend utiliza essas informações para filtrar automaticamente os recursos retornados. No caso de tokens user/, o servidor limita as respostas aos recursos associados ao identificador FHIR vinculado ao profissional autenticado, por exemplo, retornando apenas pacientes e observações cadastrados por aquele médico. Já para tokens patient/, o escopo restringe todas as consultas ao próprio recurso do paciente logado, impedindo que dados de terceiros sejam acessados. Esse mecanismo garante que a autorização seja aplicada de forma consistente e automática, reforçando o isolamento entre usuários e assegurando que cada perfil visualize apenas os dados pertinentes ao seu contexto.

A adoção desse novo modelo trouxe diversos benefícios, como a melhora na escalabilidade, ao facilitar a criação de novos papéis (como enfermeiros, recepcionistas, administradores, etc.) sem a necessidade de mudanças estruturais complexas. Também proporcionou maior aderência ao padrão FHIR ao vincular diretamente o usuário ao seu recurso correspondente. Além disso, possibilitou maior controle sobre os mecanismos de autorização, permitindo gerenciar escopos e permissões de acesso aos dados.

3.5 Desenvolvimento do Biomonitor

Paralelamente à implementação das funções necessárias para aprimorar a API, foi conduzido um estudo aprofundado sobre a estrutura e funcionamento do sistema *ECGMonitor* (DUARTE, 2024), com o objetivo de avaliar suas funcionalidades e potencial de expansão.

Durante essa análise, constatou-se que as bibliotecas responsáveis pela visualização dinâmica dos eletrocardiogramas utilizavam uma versão de demonstração, cuja continuidade implicaria em custos financeiros inviáveis para o projeto.

Adicionalmente, verificou-se que a aplicação[2] fazia uso de versões desatualizadas de componentes essenciais da stack, como o Node.js, além de empregar ferramentas de *build* e gerenciamento de dependências tecnicamente ultrapassadas, que tornavam a manutenção e o desenvolvimento do sistema mais difíceis. Esses fatores representariam uma sobrecarga significativa no processo de atualização da aplicação, dificultando sua integração com o restante da infraestrutura.

Diante desse cenário, optou-se pela criação de um novo projeto, denominado *Biomonitor*[3], com o objetivo de reconstruir a aplicação utilizando tecnologias atualizadas e compatíveis, ao mesmo tempo em que se buscou reaproveitar partes relevantes da lógica e funcionalidades originalmente presentes no *ECGMonitor*.

Ao longo do desenvolvimento, o Biomonitor passou também a assumir um papel demonstrativo dentro da solução. Embora tenha surgido inicialmente como uma reimplementação moderna do ECGMonitor, ele evoluiu para servir como um front-end voltado a evidenciar, de forma prática, as capacidades habilitadas pela autenticação e autorização SMART on FHIR. Dessa forma, além de prover as funcionalidades básicas de acompanhamento clínico, o Biomonitor tornou-se o ambiente utilizado para validar e visualizar como diferentes papéis de usuário, escopos de acesso e regras de controle influenciam o consumo dos recursos FHIR disponibilizados pela API.

O fluxo de funcionamento da aplicação inicia-se com a verificação da autenticação do usuário. Caso não exista um token válido armazenado no navegador, o usuário é redirecionado para uma tela inicial contendo duas opções de acesso: entrar como paciente ou como médico. Cada opção aciona um fluxo SMART on FHIR pedindo diferentes tipos de escopo para acesso, que serão enviados ao servidor de autenticação no fluxo SMART on FHIR e validados. Após a conclusão do fluxo, incluindo descoberta das capacidades do servidor, autenticação e emissão do token, o Biomonitor armazena o token de acesso e o utiliza nas requisições subsequentes, permitindo que o backend identifique o perfil do usuário e aplique corretamente as regras de segurança e autorização.

Com base no token recebido, a aplicação adapta dinamicamente sua interface e suas funcionalidades. Usuários do tipo médico recebem escopos que permitem leitura e escrita de recursos clínicos associados a seus pacientes (`Practitioner/*.cruds`), incluindo operações de criação e atualização de registros. Já usuários autenticados como pacientes recebem escopos restritos a leitura (`Patient/Observation.rs`), permitindo apenas acessar seus próprios dados e exames.

Essa configuração de escopos foi adotada no projeto para fins de demonstração, e representa

[2] `https://github.com/if4health/ecgmonitor`
[3] `https://github.com/bapgabriel/biomonitor`

apenas uma entre várias possibilidades do modelo SMART on FHIR. EM contextos reais, esses escopos podem, e muitas vezes precisam, ser ajustados para refletir diferentes legislações, políticas institucionais e interpretações de consentimento, garantindo que o controle de acesso esteja alinhado às necessidades de cada ambiente de saúde.

Essa diferenciação de comportamento baseada no escopo do token é central para o modelo SMART on FHIR e fundamenta a estrutura do Biomonitor, garantindo que cada usuário visualize apenas informações compatíveis com seu papel funcional dentro do sistema.

4 RESULTADOS

Os resultados estão agrupados de forma tradicional, onde a Seção apresenta os resultados relacionados ao desenvolvimento de melhorias no *back-end* da infraestrutura das APIs do Grupo IF4Health (Figura 1.1) e a Seção 4.1 apresenta o *front-end* Biomonitor dividido em 3 experimentos diferentes:

1. Implementação do protocolo SMART on FHIR;
2. Teste de controle de escopo considerando 2 profissionais de saúde;
3. Comparação qualitativa com o trabalho relacionado ECG Monitor (DUARTE, 2024);

4.1 Integração e disponibilização do sistema

Para estabelecer a fundação para o projeto, foi realizado a junção das APIs FASS-ECG e H2Cloud em uma única aplicação unificada, denominada *neoFASS-ECG*, além de ter sido realizado o deploy dos sistemas necessários.

4.1.1 Merge e Melhorias ao Backend

Para demonstração da junção entre as APIs, bem como a implementação das novas funcionalidades que dão base ao projeto principal, podemos comparar as telas do Swagger entre as versões das aplicações. A Figura 4.1 ilustra a evolução entre a interface da versão anterior (à esquerda) e a versão em desenvolvimento (à direita).

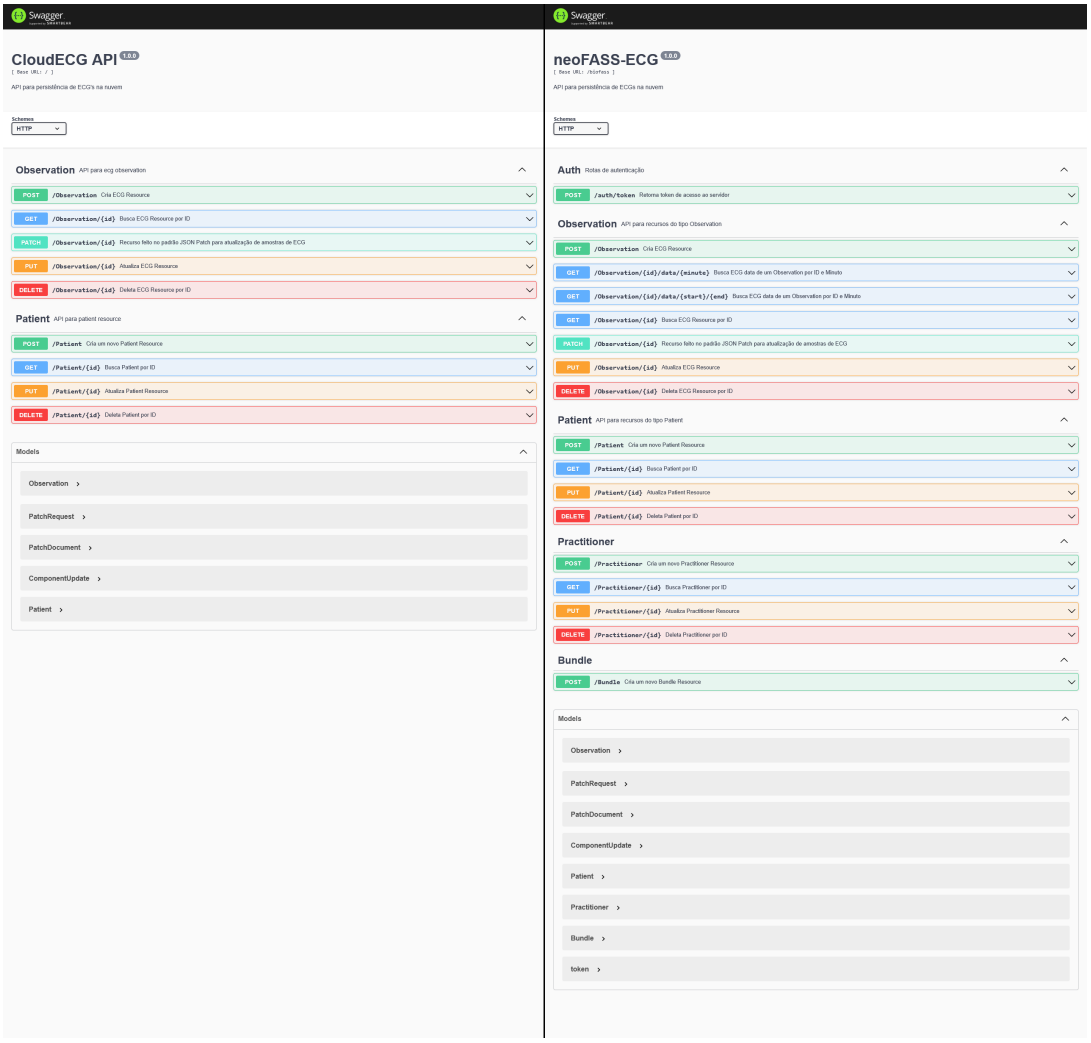
Observa-se que na versão anterior, as funções eram limitadas aos recursos *Patient* e *Observation*. Em contraste, a junção apresenta avanços significativos, dentre os quais se destacam a inclusão da rota de emissão de token (`auth/token`), e o suporte aos novos recursos *Practitioner* e *Bundle*.

Embora o suporte ao recurso *Bundle* tenha sido incluído na nova versão do backend, seu papel dentro do sistema acabou ficando mais limitado do que inicialmente planejado. A funcionalidade foi concebida para permitir importação e exportação de conjuntos completos de dados entre diferentes ambientes, mas, conforme o projeto evoluiu e as prioridades se voltaram para autenticação, autorização e integração SMART on FHIR, esse fluxo acabou não avançando.

Ainda assim, a implementação realizada oferece uma importante contribuição ao ecossistema: o *neoFASS* é capaz de processar qualquer *Bundle* FHIR do tipo *collection* válido, que contenha os recursos suportados pelo projeto, independentemente de ordem ou combinação de recursos contidos nele.

No estado atual do projeto, o único uso ativo de *Bundle* é o retorno de resultados de pesquisa, seguindo a forma recomendada pelo protocolo FHIR para respostas que contenham múltiplos recursos. Ainda que o recurso não tenha evoluído para suportar os fluxos de importação e exportação planejados inicialmente, sua infraestrutura permanece funcional e serve de base

Figura 4.1: Comparação entre o Swagger antigo na esquerda e o novo na direita.



para expansões futuras caso essas demandas retornem ao sistema.

4.1.2 Interface Gráfica do Backend

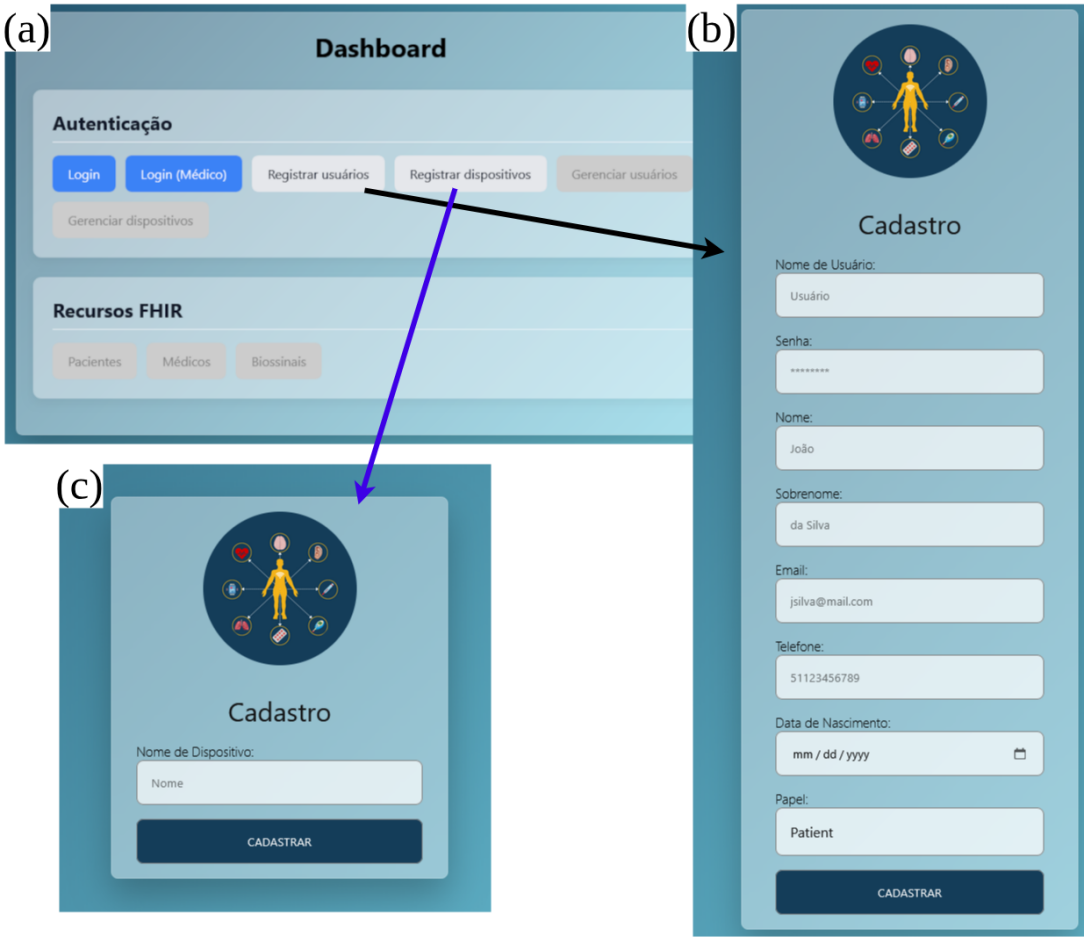
Como resultado da implementação do novo modelo unificado de usuários e da reorganização dos fluxos de autenticação, foi desenvolvida uma interface administrativa para gerenciamento do sistema. Essa interface permite o cadastro de pacientes, profissionais e dispositivos, além do acesso aos diferentes fluxos SMART on FHIR do ecossistema. Além disso, a interface gráfica abre espaço para futuras implementações de gerenciamento de recursos. A Figura 4.2(a) apresenta o painel principal dessa interface, e a Figura 4.2(b-c) apresenta os formulários para cadastro de usuários e dispositivos. Esta adição tornou o fluxo mais completo e automatizado, eliminando a dependência de pré-registros manuais no banco de dados da aplicação.

Embora as interfaces gráficas desenvolvidas façam parte essencial do fluxo da aplicação, é importante distinguir claramente seus propósitos dentro do sistema. A interface administrativa, utilizada para o cadastro de novos usuários, dispositivos e recursos FHIR correspondentes,

integra o backend do ecossistema e constitui o resultado direto da implementação do novo modelo de autenticação e gerenciamento de entidades.

Por outro lado, o Biomonitor corresponde a uma aplicação cliente independente, desenvolvida especificamente como front-end de demonstração para validar na prática os fluxos de autenticação e autorização SMART on FHIR, não devendo ser confundido com o painel administrativo do servidor.

Figura 4.2: (a) Nova tela de gerenciamento do *NeoFASS*. Os botões em cinza escuro estão desabilitados. Os botões em azul levam o usuário para telas de login com escopos de acesso diferentes, e os botões em cinza claro levam para (b) telas de cadastro de usuário e (c) dispositivo. A tela de cadastro de usuário (b) permite a escolha entre paciente e profissional de saúde.



4.1.3 Deploy

Inicialmente, foi feito o *deploy* dos sistemas em uma instância EC2 da AWS, por meio de uma conta no plano *free tier*. Embora tenha sido útil para testes iniciais e aprendizado da montagem dos sistemas envolvidos, esse *deploy* foi descontinuado e não se encontra mais ativo.

Visando maior controle sobre a infraestrutura, sustentabilidade financeira e maior confiabilidade a longo prazo, optou-se por migrar o ambiente de produção para um servidor institucional do IFSUL. Esse processo exigiu a reconfiguração de diversos aspectos do sistema,

incluindo ajustes de DNS, revisão das dependências e adaptação das variáveis de ambiente para o novo contexto de implantação, além de estudos contínuos sobre boas práticas em processos de *deploy*.

Atualmente, o sistema está disponível para acesso por meio do seguinte endereço: `https://if4health.charqueadas.ifsul.edu.br/biofass/dashboard`

4.2 Biomonitor

O Biomonitor é a aplicação web desenvolvida para demonstrar, na prática, o funcionamento do ecossistema neoFASS, servindo tanto como cliente SMART on FHIR quanto como interface de visualização de dados clínicos. Nesta seção, apresentam-se os testes realizados na aplicação a fim de validar o fluxo de autenticação e autorização, o controle de escopos e o módulo de visualização de sinais ECG. Os resultados obtidos permitem verificar como as funcionalidades implementadas se comportam em cenários reais de uso, ilustrando o grau de integração alcançado entre o front-end e o servidor FHIR.

4.2.1 Teste funcional do SMART on FHIR

Para validar na prática a implementação do protocolo SMART on FHIR, o Biomonitor foi utilizado como aplicação cliente durante os testes. A seguir, apresentam-se as quatro etapas principais do fluxo de autenticação e autorização, conforme ilustrado na Figura 2.2. Cada etapa corresponde diretamente às transições previstas no padrão SMART on FHIR, demonstrando o funcionamento integrado entre o front-end e o servidor FHIR.

Conforme ilustrado na Figura 4.3, a etapa 1 inicia assim que o usuário acessa o Biomonitor. A aplicação realiza uma requisição ao endpoint `.well-known/smart-configuration`, obtendo dinamicamente as URLs de autorização, de emissão de token e demais capacidades oferecidas pelo servidor. Com essas informações, o Biomonitor exibe uma tela na qual o usuário seleciona seu tipo funcional (paciente, profissional ou dispositivo), permitindo que a aplicação determine os escopos que serão requisitados no fluxo.

Em seguida, nas etapas 2 e 3, ocorre o processo de autenticação e autorização no servidor neoFASS. O usuário informa suas credenciais, que são verificadas no banco de dados, e, ao serem validadas, o sistema apresenta a confirmação de consentimento para os escopos solicitados pela aplicação cliente. Uma vez autorizado, o servidor envia de volta ao Biomonitor um authorization code, que é então trocado por um access token por meio da rota `/auth/token`.

Finalmente, na etapa 4, o Biomonitor recebe o token de acesso emitido pelo servidor SMART on FHIR, e esse token é armazenado de forma segura na sessão do navegador. A partir desse momento, todas as requisições feitas à API FHIR incluem esse token no cabeçalho de autorização, garantindo acesso apenas aos recursos compatíveis com os escopos concedidos.

Em síntese, o fluxo reproduzido pelo Biomonitor demonstra que o ecossistema neoFASS

implementa corretamente todas as fases do protocolo SMART on FHIR, desde a descoberta automática do servidor até a autorização e o uso efetivo do token para acesso a recursos FHIR. A validação prática confirma que as estruturas de autenticação, escopos e gerenciamento de sessão estão funcionando de modo integrado e em conformidade com as especificações do padrão.

Figura 4.3: Fluxo SMART on FHIR aplicado pelo Biomonitor.

4.2.2 Teste de Controle de Escopos

Como parte da validação prática do sistema desenvolvido, foram realizados testes no Biomonitor utilizando usuários com diferentes papéis de acesso. O objetivo foi demonstrar, de

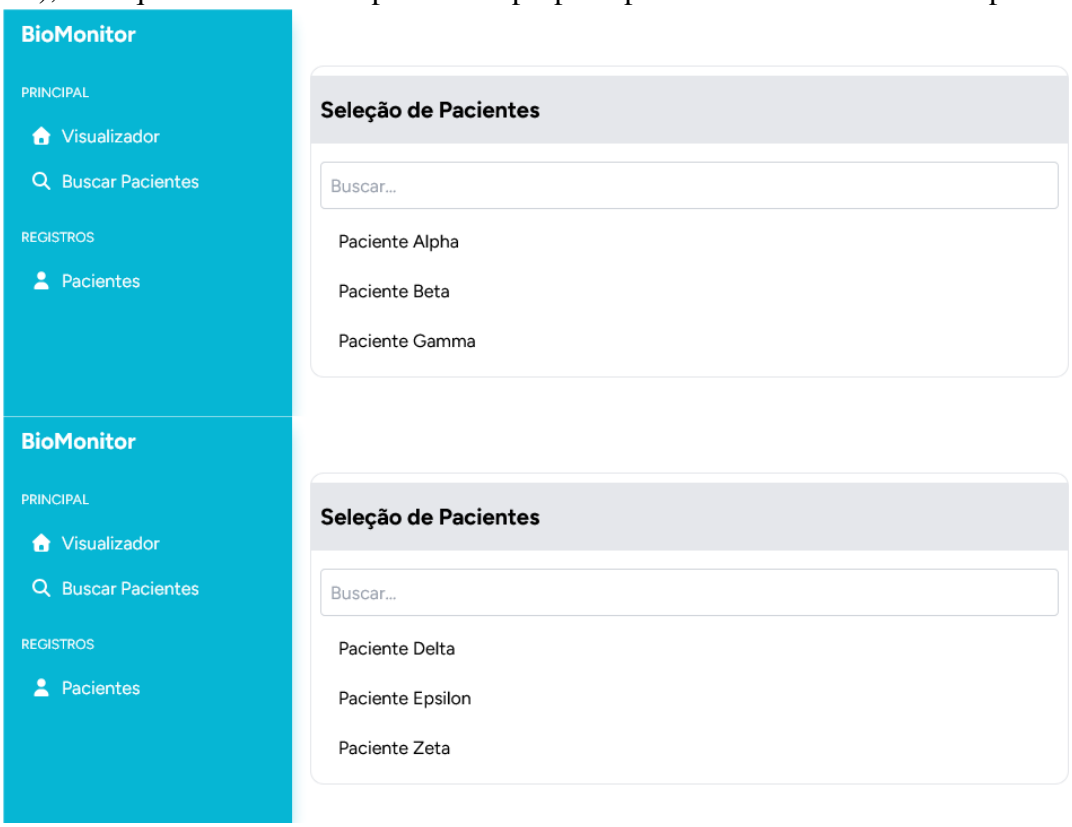
forma concreta, o funcionamento do controle de autorização baseado em escopos do SMART on FHIR e sua aplicação no ecossistema construído.

Foram cadastrados dois profissionais fictícios no sistema, identificados como Doutor A e Doutor B, ambos representados por recursos FHIR do tipo *Practitioner*. Em seguida, cada profissional autenticou-se pelo fluxo SMART on FHIR, recebendo um token contendo escopos do tipo `user/Patient.cruds` e `user/Observation.cruds`, que permitem consultar, criar e modificar recursos associados aos seus pacientes.

Após a autenticação do Doutor A, foram criados três pacientes fictícios: Paciente Alpha, Paciente Beta e Paciente Gamma. O mesmo procedimento foi repetido com o Doutor B, que registrou seus próprios três pacientes: Paciente Delta, Paciente Épsilon e Paciente Zeta.

A partir desse ponto, foi possível observar o comportamento esperado do mecanismo de autorização SMART on FHIR. Quando o Doutor A acessou o painel de seleção de pacientes, apenas os três pacientes criados por ele foram retornados pelo servidor FHIR (Figura 4.4). O mesmo ocorreu para o Doutor B, que visualizou somente Paciente Delta, Épsilon e Zeta. Isso confirma que o servidor aplica corretamente o escopo user/, retornando apenas recursos vinculados ao profissional autenticado, mesmo que outros pacientes existam na base.

Figura 4.4: Tela de seleção de pacientes no Biomonitor para o Doutor A (acima) e o Doutor B (abaixo), cada qual visualizando apenas seus próprios pacientes conforme definido pelo escopo.



A segunda etapa da validação envolveu o acesso com usuários do tipo paciente. Para isso, os pacientes criados foram autenticados individualmente utilizando o fluxo SMART on FHIR, que retorna tokens com escopos restritos, como `patient/Patient.rs` e `patient/`

`Observation.rs`. Após o login, cada paciente acessou sua versão simplificada da interface, que apresenta apenas seus dados pessoais e seus exames clínicos (Figura 4.5).

Figura 4.5: Tela simplificada exibida a um usuário do tipo Patient, sem maiores permissões.

Durante os testes, verificou-se que cada paciente tinha acesso exclusivamente às suas informações. Nenhum paciente pôde visualizar registros de outros usuários ou acessar recursos do tipo *Patient* ou *Observation* que não estivessem vinculados ao seu próprio identificador. Além disso, todas as operações de escrita utilizando o token em ferramentas externas foram bloqueadas, já que os escopos limitavam o acesso exclusivamente à leitura.

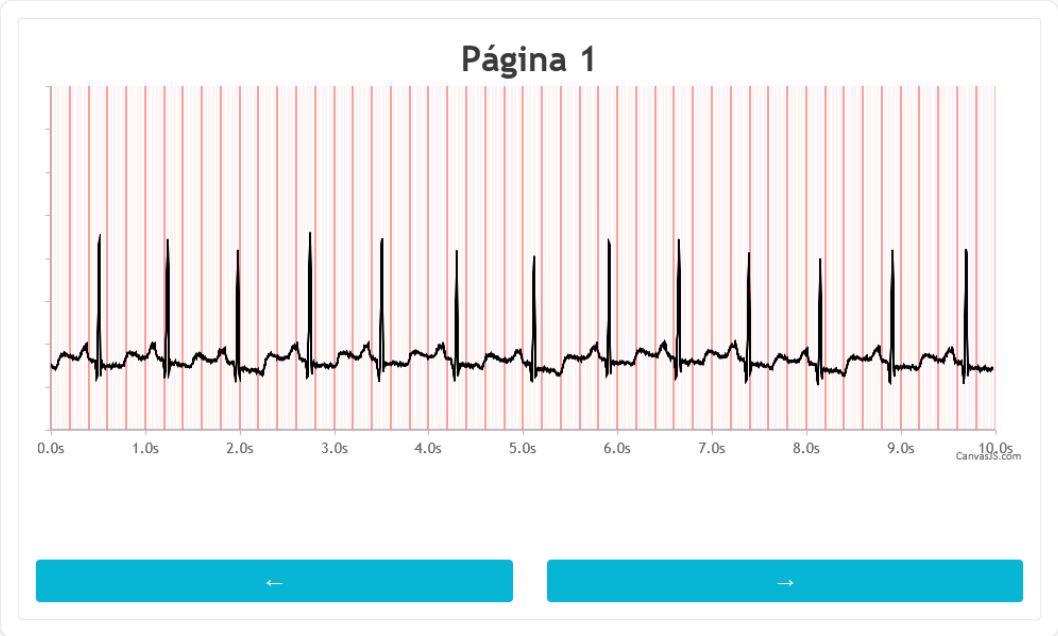
Esses resultados demonstram, de forma prática, a integração entre o servidor SMART on FHIR e a aplicação Biomonitor; seus fluxos e mecanismos estão funcionando corretamente. Os escopos configurados nos tokens controlam com precisão o que cada tipo de usuário pode visualizar ou modificar, garantindo a aplicação do princípio de menor privilégio, a proteção dos dados clínicos e o comportamento esperado em um ambiente de saúde conforme as diretrizes do padrão FHIR.

4.2.3 Comparação com o ECG Monitor

A Figura 4.6 apresenta a interface de visualização do sinal ECG implementada no Biomonitor, construída utilizando a biblioteca CanvasJS, uma solução gratuita e amplamente adotada para visualizações interativas. Apesar de ser menos especializada que bibliotecas pagas como o SciChart, utilizado no ECGMonitor original, o CanvasJS mostrou-se plenamente capaz de atender às necessidades do projeto, oferecendo desempenho adequado, boa responsividade e facilidade de integração com o restante do front-end.

A Tabela 4.1 sintetiza as principais diferenças entre o Biomonitor e o ECGMonitor. Observa-se que, embora o trabalho anterior forneça um ambiente de visualização mais robusto em ter-

Figura 4.6: Visualizador de eletrocardiograma implementado no Biomonitor.



mos de modos de exibição, o Biomonitor se destaca por adotar um modelo de acesso seguro e padronizado aos dados, baseado no protocolo SMART on FHIR. Além disso, o suporte a múltiplos tipos de usuários, agora incluindo profissionais de saúde, amplia o escopo de aplicação e aproxima o sistema de cenários reais de uso. Assim, mesmo com um modo de visualização mais simples, o Biomonitor apresenta avanços significativos em interoperabilidade, segurança e alinhamento com padrões modernos em saúde digital, aspectos essenciais para sua evolução futura.

Tabela 4.1: Tabela de comparações entre trabalhos Biomonitor e ECG monitor

Autores	**Acesso aos dados**	**Tipos de Usuário**	**Biblioteca Grafica**	**Modos de visualização**
DUARTE (2024)	Público	Paciente	SciChart (paga)	Dinâmico e Estático
Este trabalho	SMART on FHIR	Paciente e Prof. de Saúde	CanvasJS (gratuito)	Estático

Fonte: feita pelo autor

5 CONCLUSÕES E TRABALHOS FUTUROS

O desenvolvimento deste trabalho permitiu consolidar uma etapa importante da evolução do ecossistema IF4Health, unificando os serviços FASS-ECG e H2Cloud, atualizando sua arquitetura e introduzindo autenticação e autorização baseadas em SMART on FHIR. Essa integração resultou em um servidor mais coeso, padronizado e aderente às práticas recomendadas para sistemas clínicos modernos. A criação do Biomonitor demonstrou, na prática, o funcionamento do fluxo SMART on FHIR, validando a aplicação de escopos, a distinção entre tipos de usuário e o consumo seguro dos recursos FHIR disponibilizados pelo neoFASS. Os testes realizados evidenciam que o sistema cumpre os objetivos propostos e estabelece uma base sólida para aplicações interoperáveis de acompanhamento de eletrocardiogramas.

Ao longo do desenvolvimento, porém, diversas oportunidades de avanço foram identificadas. A primeira envolve a ampliação do painel administrativo do neoFASS, de modo a permitir o gerenciamento completo dos recursos FHIR suportados, incluindo visualização, edição, exclusão e auditoria. Outra frente relevante diz respeito às questões legais de consentimento e proteção de dados, que exigem estudos aprofundados sobre como traduzir diretrizes institucionais e requisitos regulatórios em políticas de escopo SMART on FHIR automatizáveis. Além disso, tanto o Biomonitor quanto potenciais novas aplicações podem expandir suas capacidades, incorporando formulários clínicos, visualizações adicionais de sinais, suporte a recursos mais complexos e integração com dispositivos e workflows reais.

Outras linhas promissoras de continuidade incluem o aprimoramento do suporte a Bundles, possibilitando importação, exportação e validação formal de pacotes de dados clínicos; a inclusão de novos recursos FHIR essenciais para cenários clínicos reais, como Encounter, Condition, Procedure, Device e DocumentReference; e a adoção de mecanismos de auditoria que registrem acessos e operações de forma estruturada e rastreável.

Finalmente, há perspectivas significativas relacionadas à infraestrutura e desempenho do sistema, incluindo containerização, pipelines de integração contínua, monitoramento, e estudos de escalabilidade. Essas direções indicam que o ecossistema IF4Health possui amplo potencial de expansão, tanto no âmbito acadêmico quanto na perspectiva de adoção prática, e que o presente trabalho estabelece a fundação necessária para essas evoluções.

REFERÊNCIAS

CDR, S. **HAPI FHIR - The Open Source FHIR API for Java.**, 2019. Acessado em 11/12/2022, Disponível em: `https://hapifhir.io/`.

DUARTE, E. W. **Desenvolvimento front-end para Visualização de Exames de Eletrocardiograma**, 2024. [S.l.]: IFSul câmpus Charqueadas, 2024.

HL7. **FHIR Release 4**, 2019. Acessado em 11/12/2018, Disponível em: `https://www.hl7.org/fhir/`.

HL7. **FHIR Resource**: bundle, 2023. Acessado em 05/07/2025, Disponível em: `https://hl7.org/fhir/bundle.html`.

HOSPITAL, B. C. **SMART App Gallery**, 2022. Acessado em 07/07/2025, Disponível em: `https://gallery.smarthealthit.org/`.

KAKARLAPUDI, P. V.; MAHMOUD, Q. H. A systematic review of blockchain for consent management. In: HEALTHCARE, 2021. **Anais...** [S.l.: s.n.], 2021. v.9, n.2, p.137.

MANDEL, J. C. et al. SMART on FHIR: a standards-based, interoperable apps platform for electronic health records. **Journal of the American Medical Informatics Association**, [S.l.], v.23, n.5, p.899–908, 2016.

MÉLO, C. B. et al. e-SUS na Atenção Primária à Saúde: uma revisão integrativa. **Journal of Health Informatics**, [S.l.], v.16, n.Especial, 2024.

PEREIRA, C. R. **CloudECG - Interoperabilidade e Streaming de Registros Eletrônicos de Saúde de Eletrocardiogramas (ECG) na Nuvem**, 2022. Trabalho de Conclusão de Curso, IFSul câmpus Charqueadas.

PEREIRA, C. R. et al. FASS-ECG: a fhir cloud api to enable streaming and storage of continuous 12-leads ecgs. In: IEEE 36TH INTERNATIONAL SYMPOSIUM ON COMPUTER-BASED MEDICAL SYSTEMS (CBMS), 2023., 2023. **Anais...** [S.l.: s.n.], 2023. p.898–903.

ROEHRS, A.; DA COSTA, C. A.; ROSA RIGHI, R. da. OmniPHR: a distributed architecture model to integrate personal health records. **Journal of biomedical informatics**, [S.l.], v.71, p.70–81, 2017.

SANTOS, L. S. dos et al. Interoperabilidade e Segurança na Implementação de Aplicações Web de Saúde com SMART on FHIR. **Journal of Health Informatics**, [S.l.], v.15, n.Especial, 2023.

VIEIRA, J. M. et al. IF-Cloud: api fhir para integração de projetos de saúde digital. **Journal of Health Informatics**, [S.l.], v.16, n.Especial, 2024.