

**Instituto Politécnico do Cávado e do Ave**

**Escola Superior de Tecnologia**

**Engenharia em Desenvolvimento de Jogos Digitais**

Italo Teófilo Filho 25961

**Fibro Apoio**

**Junho de 2025**



**Instituto Politécnico do Cávado e do Ave**

**Escola Superior de Tecnologia**

**Engenharia em Desenvolvimento de Jogos Digitais**

**Relatório de Estágio/Projeto**

Trabalho efectuado sob a orientação de

Duarte Duque

**Julho de 2025**

**Resumo**

O FibroApoio é um aplicativo mobile desenvolvido com o objetivo de facilitar o acompanhamento de condições de saúde por meio da coleta de dados médicos com gamificação. Voltado principalmente para pacientes com dores crônicas, o app permite o registro diário de informações como níveis de dor, uso de medicações e prática de exercícios físicos.

Utilizando elementos como sistema de pontos, *streaks* e *rankings* semanais, o FibroApoio transforma o monitoramento da saúde em uma experiência interativa e motivadora. A arquitetura MVVM com injeção de dependências via Swinject, a navegação desacoplada com AppCoordinatorService e o gerenciamento de estado com Combine garantem modularidade e responsividade.

O projeto foi desenvolvido em Swift com armazenamento em Firebase Firestore, oferecendo escalabilidade e integração em tempo real. O sistema de gamificação foi cuidadosamente estruturado para promover hábitos saudáveis de forma consistente, com recompensas baseadas em *check-ins*, registros e progressão de níveis. A estrutura de banco de dados relacional no Firestore permite o rastreio eficiente de dados por usuário.

Foram enfrentados desafios técnicos como sincronização de dados entre memória, Firestore e *cache* local, além de falhas silenciosas de persistência que foram resolvidas com checagens automáticas. O resultado final é um aplicativo robusto, responsivo e centrado no usuário, que potencializa a adesão ao tratamento de saúde através de motivação lúdica e interatividade digital.

Índice

[Índice de Figuras 5](#_Toc199851766)

[Índice de Tabelas 6](#_Toc199851767)

[Lista de acrónimos 7](#_Toc199851768)

[1. Introdução 8](#_Toc199851769)

[**Cronograma** 9](#_Toc199851770)

[2. Tecnologias de Suporte ao projeto 10](#_Toc199851771)

[**Sistema de Armazenamento de Dados** 11](#_Toc199851772)

[3. Desenvolvimento 12](#_Toc199851773)

[**Requisitos Funcionais** 12](#_Toc199851774)

[Registro e Monitoramento de Dores 12](#_Toc199851775)

[Registro de Medicações 12](#_Toc199851776)

[Registro de Exercícios Físicos 12](#_Toc199851777)

[Check-in e Integração com Gamificação 12](#_Toc199851778)

[Interface e Design 12](#_Toc199851779)

[**Requisitos Não Funcionais** 13](#_Toc199851780)

[Segurança e Privacidade dos Dados 13](#_Toc199851781)

[Desempenho e Responsividade 13](#_Toc199851782)

[Disponibilidade e Escalabilidade 13](#_Toc199851783)

[Usabilidade e Acessibilidade 13](#_Toc199851784)

[Robustez e Manutenção 13](#_Toc199851785)

[**Tabelas de Sistema de Pontos** 15](#_Toc199851786)

[4. Estrutura da Base de Dados 17](#_Toc199851787)

[**5.** **Fluxo do Utilizador** 19](#_Toc199851788)

[**6.** **Gerenciamento de Estado** 20](#_Toc199851789)

[**7.** **Design System** 21](#_Toc199851790)

[8. Resultados 22](#_Toc199851791)

[9. Conclusão 23](#_Toc199851792)

[**Desafios Técnicos e Soluções** 23](#_Toc199851793)

[Bibliografia 24](#_Toc199851794)

# Índice de Figuras

[Figura 1 - Exemplo de icone raio 13](#_Toc200119597)

[Figura 2 - Estrutura de base de dados 16](#_Toc200119598)

[Figura 3 - Fluxo de utilizador no dashboard 18](#_Toc200119599)

[Figura 4 - exemplo de função que usa gerenciamento de estado 19](#_Toc200119600)

[Figura 5 - Tema de cores em assets 20](#_Toc200119601)

[Figura 6 - exemplo de tela 22](#_Toc200119602)

# Índice de Tabelas

[Tabela 1 - Sistema de pontos por ação 14](#_Toc200017780)

# Lista de acrónimos

| **Acrónimo** | **Significado** |
| --- | --- |
| @Published | Atributo reativo do Swift/Combine |
| API | Application Programming Interface |
| CRUD | Create, Read, Update, Delete |
| DB | Database (Base de Dados) |
| iOS | Sistema Operacional da Apple para mobile |
| MVVM | Model-View-ViewModel |
| UI | User Interface (Interface do Usuário) |
| UX | User Experience (Experiência do Usuário) |
| UX/UI | User Experience / User Interface |

# Introdução

O **FibroApoio** é um aplicativo inovador que utiliza métodos de gamificação para coletar dados médicos de pacientes. A partir de um sistema de pontuação dinâmico e elementos visuais interativos, o app incentiva os usuários a registrarem informações sobre sua saúde, permitindo um acompanhamento mais eficiente do seu estado de bem-estar.

O objetivo principal do FibroApoio é facilitar a adesão dos pacientes ao monitoramento de suas condições, transformando o processo de coleta de dados em uma experiência mais motivadora e menos burocrática. O uso de elementos como *check-ins* diários, *ranking* de saúde e recompensas contribui para manter os usuários engajados e, ao mesmo tempo, fornecer informações valiosas para profissionais da saúde.

1. **Identificação do problema**  
   Muitos pacientes com condições crônicas, como dores musculoesqueléticas, têm dificuldade em manter o acompanhamento contínuo do seu estado de saúde. A coleta de dados costuma ser negligenciada por falta de motivação, esquecimento ou complexidade dos métodos tradicionais. Isso compromete tanto o tratamento quanto a análise por parte dos profissionais de saúde.
2. **Objetivos**  
   O projeto FibroApoio visa criar uma solução digital que torne o processo de coleta de dados médicos mais atrativo, eficiente e contínuo. Os objetivos principais incluem:

* Implementar mecânicas de gamificação para promover o engajamento;
* Garantir o registro estruturado e seguro de informações como dores, medicações e exercícios físicos;
* Disponibilizar dados úteis aos profissionais de saúde de forma organizada.

**Apresentação dos capítulos seguintes**  
No decorrer deste relatório, o leitor encontrará primeiramente a definição dos requisitos do sistema, tanto funcionais quanto não funcionais, seguida das decisões tecnológicas adotadas, como a escolha da linguagem Swift, Firebase para o *backend* e arquitetura MVVM. Em seguida, são descritos os sistemas de gamificação, incluindo pontuações, *streaks* e *rankings*. Também é detalhada a estrutura da base de dados, o fluxo de navegação dentro do app, o *design system* adotado e os principais desafios enfrentados durante o desenvolvimento. O relatório é finalizado com as conclusões e perspectivas futuras para o FibroApoio.

## **1. 3 Cronograma**

**[17 a 28 de fevereiro]**

Definição de requisitos funcionais e não-funcionais.

Criação de *wireframes* da aplicação.

Escolha de linguagem/plataforma de desenvolvimento.

Escolha de base de dados.

**[3 a 14 de março]**

Criação de base de dados para interação com a app.

Desenvolvimento de funções básicas da app (e.g., registo, login, perfil de utilizador).

Estudo de técnicas de gamificação a aplicar.

**[17 de março a 11 de abril]**

Implementação de funcionalidades de recolha de dados dos pacientes.

Armazenamento na base de dados dos dados recolhidos.

Testes.

**[14 de abril a 2 de maio]**

Implementação, na App, de técnicas de gamificação (e.g., sistema de pontos, recompensas, *badges*, etc.)

**[5 de maio a 23 de maio]**

Testes funcionais.

Correção de *bugs*.

Testes de usabilidade.

**[26 de maio]** Envio, ao orientador, da primeira versão do relatório para revisão.

**[2 de junho]** Envio, ao orientador, da versão final do relatório para revisão.

**[6 de junho]** Entrega do Relatório de Projeto.

# Tecnologias de Suporte ao projeto

Neste capítulo, são apresentadas as principais tecnologias utilizadas no desenvolvimento do FibroApoio. A escolha da linguagem de programação, o sistema de armazenamento de dados e a arquitetura de software são discutidos com base em critérios como desempenho, escalabilidade, compatibilidade com iOS e facilidade de manutenção. Estas decisões fundamentam a estrutura técnica do projeto.

## **2.1 Linguagem de programação**

Para garantir que o **FibroApoio** seja acessível a um maior número de usuários, foi necessário escolher uma tecnologia que possibilitasse o desenvolvimento com maior praticidade, reduzindo o tempo de desenvolvimento sem comprometer a performance. Após uma análise comparativa entre diferentes linguagens, o **Swift** foi escolhido como a melhor opção para o projeto.

Tabela 1 - Comparativo de linguagens

|  |  |  |
| --- | --- | --- |
| **Linguagem** | **Vantagens** | **Desvantagens** |
| Java | Amplamente utilizado, robusto, grande suporte da comunidade | Código mais verboso, maior complexidade para desenvolvimento mobile |
| Kotlin | Excelente para Android nativo, alta interoperabilidade com Java, suporte a Kotlin Multiplatform, linguagem a qual já possuo experiência | Restrito ao ecossistema Android quando usado sem multiplatform |
| Swift | Alto desempenho, otimizado para iOS, gerenciamento eficiente de memória | Limitado ao ecossistema Apple |
| Dart (Flutter) | Multiplataforma, interfaces ricas, boa performance, curva de aprendizado menor | Pode apresentar um leve overhead comparado a soluções nativas |

## **2.2 Arquitetura MVVM**

O aplicativo segue a arquitetura **MVVM (Model-View-ViewModel)**, com separação clara entre dados, lógica e apresentação. Essa abordagem melhora a testabilidade e manutenção do código.

Para gerenciamento de dependências, utilizei o **Swinject**. Um DependencyContainer central que registra e disponibiliza todas as instâncias necessárias: serviços, *view models* e *views*. Isso garante que cada tela use instâncias únicas e facilita a manutenção e testes.

A navegação é controlada por um AppCoordinatorService, responsável por trocar a tela ativa com base em uma variável observada. Isso evita acoplamento entre *views* e lógica de fluxo.

Todos os serviços ficam acessíveis nas *views* através do property *wrapper* @Service, implementado com base no Swinject. Isso permite injeção direta e segura de dependências sem uso de EnvironmentObject

**2.3 Sistema de Armazenamento de Dados**

Para a aplicação FibroApoio, várias soluções de bases de dados foram consideradas para garantir a melhor integração com Swift e otimizar o desempenho. A tabela abaixo compara algumas das opções avaliadas:

Tabela 2 - Compartivo sistemas de armazenamento de dados

|  |  |  |
| --- | --- | --- |
| **Base de Dados** | **Descrição** | **Vantagens** |
| Firebase | Base de dados NoSQL, escalável e com integração facilitada com Swift. | Escalabilidade automática, tempo real, autenticação simplificada, fácil integração com plataformas móveis. |
| Realm | Base de dados móvel, leve e rápida. | Ideal para dados locais, alta performance, fácil de usar em Swift. |
| SQLite | Base de dados relacional leve, utilizada para armazenamento local. | Simples, sem necessidade de servidor, ideal para dados estruturados. |
| PostgreSQL | Base de dados relacional robusta e escalável. | Suporte a ACID, extensibilidade, ideal para sistemas complexos com alta consistência. |

Optou-se pelo Firebase devido à sua escalabilidade inerente e capacidade de lidar com um crescimento significativo de utilizadores e dados. A infraestrutura do Firebase permite uma expansão automática dos recursos, garantindo que a aplicação permaneça responsiva e eficiente mesmo com um grande volume de acessos e informações.

# Desenvolvimento

Neste capítulo, dedicado ao desenvolvimento, abordam-se os requisitos levantados durante a concepção do projeto, bem como as implementações realizadas. Ele está dividido em requisitos funcionais e não funcionais, sistema de gamificação, estrutura de base de dados, fluxo de utilizador e gerenciamento de estado. Essa organização permite entender como o aplicativo evoluiu de uma ideia para um produto funcional.

## **3.1 Requisitos**

### 3.1.1 Requisitos Funcionais

### Registro e Monitoramento de Dores

* O aplicativo deve permitir que o paciente registre seu nível de dor em uma escala de 0 a 10.
* Os registros podem ser realizados em momentos diferentes do dia: ao acordar, em meio‐dia e antes de dormir, com a possibilidade de inserção adicional a qualquer momento.
* O paciente poderá indicar a localização e a região do corpo onde sente dor, possibilitando um registro detalhado dos sintomas.

### Registro de Medicações

* O sistema deve permitir que o paciente registre o momento em que tomou um medicamento.
* Cada registro deverá conter informações como:
  + Hora do consumo.
  + Nome do medicamento.
  + Observações extras (dosagem, observações de efeitos, etc.).

### Registro de Exercícios Físicos

* O aplicativo deve possibilitar o registro dos exercícios realizados pelo paciente. Cada registro incluirá:
  + Hora da atividade.
  + Tipo/nome do exercício.
  + Duração.
  + Observações extras (dores, etc.).

### Check-in e Integração com Gamificação

* Permitir um sistema de *check-in* diário, facilitando a entrada de dados e incentivando a consistência.
* Integrar as mecânicas de gamificação para converter as ações diárias em pontos e recompensas, reforçando o engajamento do usuário.

### Interface e Design

* A interface do aplicativo deverá seguir o design base definido no Figma, garantindo usabilidade e clareza na apresentação dos dados e interações.

### 3.1.2 Requisitos Não Funcionais

### Segurança e Privacidade dos Dados

* Os dados dos pacientes devem ser armazenados de forma criptografada. É imprescindível que o armazenamento e o acesso às informações sigam as normas de proteção de dados vigentes, garantindo a privacidade do usuário.

### Desempenho e Responsividade

* O aplicativo deve oferecer um desempenho ágil, com baixa latência mesmo sob alta demanda de acessos e registros. A resposta rápida às ações dos usuários é essencial para manter a usabilidade e a satisfação.

### Disponibilidade e Escalabilidade

* O sistema precisa ser capaz de atender a um grande número de usuários simultaneamente, permitindo expansões futuras sem comprometer a performance.

### Usabilidade e Acessibilidade

* Seguir as melhores práticas de design centrado no usuário para oferecer uma interface intuitiva. Garantir a acessibilidade, com elementos visuais claros e orientações que auxiliem todos os perfis de usuários.

### Robustez e Manutenção

* Implementar métodos para monitoramento de falhas e erros, possibilitando ações de manutenção preventiva e corretiva. Priorizar um sistema resiliente, que mantenha a integridade dos dados mesmo diante de problemas técnicos.

## **3.2 Wireframes**

Os wireframes foram utilizados para planejar a interface do aplicativo antes do desenvolvimento. Eles serviram como guia visual para a construção das telas, alinhando os requisitos funcionais com uma estrutura de navegação fluida e centrada no usuário. Abaixo, são apresentados os principais modelos de telas que serviram como base para o desenvolvimento.

**Uma imagem com software, captura de ecrã, Ícone de computador, Software de multimédia

Os conteúdos gerados por IA podem estar incorretos.**

Figura 1 - Wireframes

Para garantir uma experiência de uso clara e fluida, o aplicativo foi desenhado com um fluxo de navegação simples, direto e adaptado à jornada do paciente. As telas apresentadas nos *wireframes* representam as etapas principais dessa navegação e refletem os requisitos funcionais definidos para o projeto.

O fluxo inicia-se na ***Splash Screen***, responsável por carregar os dados essenciais e verificar se o usuário já está autenticado. A seguir, apresenta-se o ***Onboarding,*** que introduz os principais recursos do app de forma didática para novos usuários.

Caso o usuário ainda não tenha conta, ele é direcionado para a tela de **Registro**; se já for um usuário, acessa diretamente a tela de ***Login***. Após a autenticação, o app redireciona para o ***Dashboard***, o centro principal de navegação, onde o utilizador pode acompanhar seu progresso, pontuação, *ranking* e acessar outras funcionalidades.

As funcionalidades principais estão organizadas em telas específicas:

***Pain* (Dores)**: permite registrar local, intensidade e frequência da dor sentida, com interface visual interativa baseada em um modelo corporal.

***Medicines* (Medicações)**: possibilita o registo de medicamentos tomados, horários e observações relacionadas.

***Exercises* (Exercícios)**: permite registrar atividades físicas realizadas, tempo de duração e observações sobre o impacto no bem-estar.

Essas três áreas representam os principais tipos de entrada de dados que alimentam o sistema de gamificação e a progressão do paciente dentro do app. A navegação foi projetada para que o utilizador possa realizar *check-ins*, registrar atividades e acompanhar seus dados de forma rápida e intuitiva.

## **3.3 Gamificação**

### 3.3.1 Sistema de Pontuações

O sistema de pontuação do FibroApoio foi desenvolvido com o objetivo de incentivar o engajamento e a consistência dos usuários através de mecânicas de gamificação. Através de recompensas e desafios diários, o aplicativo busca criar uma experiência motivadora e interativa, transformando o cuidado com a saúde em uma jornada mais divertida e recompensadora.

**Temática e Representação**

O sistema utiliza dois elementos simbólicos para representar diferentes dimensões da experiência do usuário:

**Raio (⚡ Energia Vital):** Representa a energia acumulada pelo usuário, obtida através de interações diárias e atividades no aplicativo. A energia expira se não for utilizada dentro de um período específico, incentivando o uso contínuo.

**Chama (🔥 Streak):** Simboliza a sequência de dias consecutivos em que o usuário realiza o *check-in*, promovendo a consistência e o hábito saudável.

Uma imagem com Tipo de letra, logótipo, símbolo, Gráficos

Os conteúdos gerados por IA podem estar incorretos.

Figura 2 - Exemplo de icone raio

## **3.4 Sistema de Pontos**

Ganho de Pontos

O processo de ganho e perda de pontos foram baseados em um conceito de numerologia e seus efeitos psicológicos que é apenas pseudociência.

As ações que conferem pontos aos usuários encontram-se descritas na tabela abaixo.

Tabela 3 - Sistema de pontos por ação

|  |  |  |  |
| --- | --- | --- | --- |
| **Ação** | **Pontos Ganhos** | **Frequência** | **Impacto Psicológico** |
| 🏆 **Check-in diário** | 900-1100 | 1x/dia | Consistência, reforço positivo. Ganhos aleatórios mantém o engajamento. |
| 📊 **Registro de dores/dados** | 500-700 | 3-5x/dia | Reflexão pessoal, recompensa rápida e incentivo a registrar durante o dia. |
| **🎁 Bônus inicial (registro)** | 5000 | 1x (único) | Motivação inicial, cria uma sensação de conquista logo no começo. |
| 💊 Registro de medicações | 200-400 | Baseado na rotina | Estimula o acompanhamento correto da rotina. Recompensa aumenta com a regularidade. |
| 🏃‍♀️ Registro de exercícios | 300-500 | Baseado no tempo/frequência | Promove a saúde física e incentiva o comportamento ativo. |
| 🔥 Sequência completa (7 dias) | 1000-4000 | 1x/7 dias acumulativo no mês | Reforça hábitos diários consistentes. Compensa mais após longas sequências de atividade. |

### 3.4.1 Estratégias de Pontuação

Além dos pontos básicos, são aplicadas estratégias complementares, como:

**Sistema de *Rankings***

Inspirado no modelo de progressão visto em aplicativos como o Duolingo, o FibroApoio organiza os usuários em *rankings* que evoluem conforme a pontuação acumulada. Essa dinâmica oferece um senso de competição amigável e recompensa a consistência e a qualidade dos registros. Os *rankings* são estruturados da seguinte forma:

**Hierarquia dos *Rankings***

Ordem de evolução: Madeira, Pedra, Bronze, Prata, Ouro e Diamante.

**Critérios de Evolução**

Ao final de cada semana, os usuários com as pontuações mínimas e, dependendo do *ranking* atual, avançam para o próximo *ranking*.

A progressão é pensada de forma equilibrada, incentivando que o usuário registre suas atividades reais (idealmente, 1 exercício diário e 3 registros de dores diários) sem a tentação de inserções artificiais para ganho de pontos.

Tabela 4 - Sistema de subida de rank

|  |  |  |
| --- | --- | --- |
| **Rank Atual** | **Pontuação Mínima para Subir** | **Descrição do Nível de Atividade** |
| Madeira | 12.000 | Pouca interação no aplicativo, check-ins diários básicos e poucos registros adicionais. |
| Pedra | 20.000 | Check-ins frequentes, registros de dores e medicações regulares (mínimo entre 3-4 dias na semana). |
| Bronze | 35.000 | Check-in diário, alguns dias de exercícios físicos, registros de medicação em horários regulares e participação ativa, mas não intensa. |
| Prata | 55.000 | Atividade moderada: check-in todos os dias, cerca de 5 dias/semana de exercícios ou registros consistentes de dores, medicações e rotina. |
| Ouro | 85.000 | Consistência diária: check-in todos os dias, registro de exercícios pelo menos 5x por semana, registros completos de dores e medicações. |
| Diamante | +100.000 | Nível máximo: Usuário se mantém ativo diariamente, realiza check-ins constantes e mantém uma rotina regular de exercícios e registros no aplicativo. |

# Estrutura da Base de Dados

A estrutura de dados do FibroApoio foi pensada para permitir rápida consulta e atualização, garantindo que todas as interações do usuário sejam armazenadas de forma organizada e segura. Nesta seção, é apresentada a modelagem das coleções no Firebase Firestore, bem como a relação entre elas. O uso complementar de armazenamento local (UserDefaults) também é explicado para fins de desempenho e acesso *offline*.

Uma imagem com texto, captura de ecrã, design

Os conteúdos gerados por IA podem estar incorretos. baixo, detalhamos as tabelas principais e as suas relações , mostrando o modelo ER das tabelas implementadas em noSQL, no caso, firebase.

Figura 3 - Estrutura de base de dados

**Usuários**

* **Descrição**: Armazena informações essenciais dos utilizadores, como nome, email, data de nascimento, telefone, altura, peso, género, pontuação, *streak* atual, *ranking* e *nickname*.
* **Relações**:
  + **Ranking**: Cada utilizador está associado a um nível de *ranking* por meio da chave estrangeira id\_rank.
  + **Registros**: Liga-se às tabelas de **Histórico de Pesos**, **Dores**, **Medicações**, **Exercícios** e **Check-ins** através da chave estrangeira id\_usuario.

**Histórico de Pesos**

* **Descrição**: Registra o histórico de pesos dos utilizadores, incluindo o peso em quilogramas.
* **Relações**:
  + **Usuários**: Cada registro está associado a um utilizador específico (id\_usuario).

**Dores**

* **Descrição**: Armazena informações sobre os episódios de dor dos utilizadores, como nível da dor e local da dor.
* **Relações**:
  + **Usuários**: Cada registro está vinculado a um utilizador (id\_usuario).

**Medicações**

* **Descrição**: Armazena informações sobre as medicações utilizadas pelos utilizadores, como nome do medicamento, data de consumo, período e observações.
* **Relações**:
  + **Usuários**: Cada medicação está associada a um utilizador específico (id\_usuario).

**Exercícios**

* **Descrição**: Armazena os registros de exercícios físicos realizados pelos utilizadores, incluindo duração e observações.
* **Relações**:
  + **Usuários**: Cada exercício está associado a um utilizador (id\_usuario).

**Check-ins Diários**

* **Descrição**: Registra os *check-ins* diários realizados pelos utilizadores, incluindo *status* do *streak*.
* **Relações**:
  + **Usuários**: Cada *check-in* está associado a um utilizador específico (id\_usuario).

**Ranking**

* **Descrição**: Define os níveis de *ranking*, com um nome e a pontuação mínima para cada nível.
* **Relações**:
  + **Usuários**: Um utilizador pertence a um *ranking* específico (id\_rank).

## **Fluxo do Utilizador**

Entender o fluxo de utilização do aplicativo é essencial para avaliar sua usabilidade. Esta seção descreve as etapas de navegação desde a abertura do app até a realização de registros e interações. O fluxo foi desenhado para ser intuitivo, rápido e adaptado à rotina do usuário, promovendo a constância no uso.

Ao iniciar o app:

1. O Firebase é inicializado.
2. Os serviços e *view models* são registrados no DependencyContainer.
3. A primeira tela exibida é a SplashScreen, que executa:
   * Carregamento do usuário atual (UserService.loadUser()).
   * Carregamento de dados semanais (loadWeeklyData()).
   * Carregamento da lista de usuários com mesmo *rank*.
4. Após o carregamento:
   * Se o usuário não estiver logado, verifica-se se ele já viu a WelcomeScreen.
   * Caso já tenha cadastro e esteja logado, navega-se para o DashboardScreenView.

Dentro do *dashboard*, o usuário pode navegar entre:

* **Home**: exibe dados semanais e *check-in*.
* **Rank**: mostra pontuação e posição.
* **Add Entry**: permite registrar dor, exercício ou medicação.
* **Activity**: histórico da semana.
* **Profile**: dados pessoais e botão de *logout*.

Uma imagem com texto, captura de ecrã, multimédia

Os conteúdos gerados por IA podem estar incorretos.

Figura 4 - Fluxo de utilizador no dashboard

## **Gerenciamento de Estado**

O gerenciamento de estado no FibroApoio é responsável por manter os dados do usuário atualizados e sincronizados entre a interface, a memória do dispositivo e o banco de dados remoto. Esta seção apresenta a estrutura reativa baseada em Combine e a lógica implementada para garantir que qualquer alteração feita pelo usuário reflita corretamente em todas as camadas do sistema.

O estado é gerenciado com Combine e @Published. O UserService mantém:

* currentUser: o usuário atual.
* weeklyData: dados da semana atual.
* rankUsers: lista dos 50 usuários do mesmo *ranking*.

Esses dados são carregados uma vez na *splash screen* e propagados automaticamente para todas as *views* que os utilizam.

A sincronização entre Firestore, memória e *cache* local (UserDefaults) é centralizada. Após salvar um dado novo, o UserService atualiza todos os pontos de persistência.

**Uma imagem com texto, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.**

Figura 5 - exemplo de função que usa gerenciamento de estado

## **Design System**

Para garantir consistência visual e facilitar a manutenção, o projeto adota um *design system* centralizado. Esta seção apresenta a estrutura de temas, paletas de cores, tamanhos de fonte e componentes reutilizáveis (AtomViews) que compõem a identidade visual do aplicativo. A padronização garante uma experiência de uso clara e coesa em todas as telas.

Foi criado um serviço de tema (Theme.swift) com:

* Paleta de cores customizada (Colors).
* Tamanhos de fonte (FontSizes).
* Estilos tipográficos (TextTypes), como .title(), .body(), .caption() etc.

Além disso, componentes reutilizáveis (AtomViews) foram desenvolvidos para garantir consistência visual e de comportamento:

* AtomTextInput, AtomNumberInput, AtomButton, AtomDateInput, entre outros.

As views seguem um padrão visual definido no Figma e utilizam o tema via .title(theme), .body(theme) e similares.

Uma imagem com texto, captura de ecrã, software, Ícone de computador

Os conteúdos gerados por IA podem estar incorretos.

Figura 6 - Tema de cores em assets

# Resultados

O desenvolvimento do FibroApoio resultou em um aplicativo funcional, estável e com uma interface amigável, pronto para ser testado por usuários reais. Todas as funcionalidades principais planejadas foram implementadas com sucesso, incluindo:

**Registro de dores** com escala de intensidade e localização no corpo;

**Registro de medicações**, com informações sobre nome, hora e observações;

**Registro de exercícios físicos**, incluindo tipo, duração e notas;

**Sistema de check-in diário**, para promover consistência no uso;

**Gamificação** com pontos, *streaks* e *rankings* semanais;

**Interface modular**, com componentes reutilizáveis e tema visual unificado;

**Fluxo de navegação desacoplado**, garantindo flexibilidade para escalabilidade futura;

**Base de dados em Firebase Firestore**, com sincronização eficiente e em tempo real.

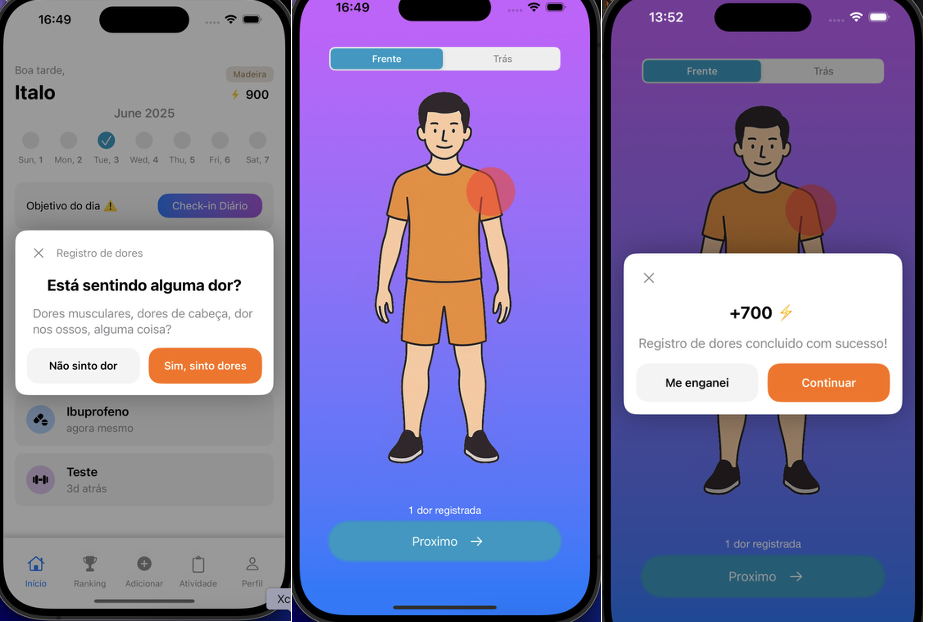
**Fluxo adicionar dor/fazer checkin**

Figura 7 - Exemplo fluxo adicionar dor

**Fluxo adicionar medicamento**

**Uma imagem com Telemóvel, texto, Dispositivo móvel, captura de ecrã

Os conteúdos gerados por IA podem estar incorretos.**

Figura 8 - Exemplo fluxo adicionar medicamento

# Conclusão

O desenvolvimento do FibroApoio proporcionou uma experiência rica tanto em termos técnicos quanto em aprendizado prático. O projeto foi além da simples criação de um aplicativo funcional — ele demonstrou que a gamificação pode ser uma poderosa aliada na saúde digital, promovendo a adesão do usuário de forma voluntária e prazerosa.

Foram vencidos desafios técnicos relevantes, como a consistência entre múltiplas camadas de persistência de dados, a construção de um sistema de navegação desacoplado e a organização modular de serviços e interfaces. Tais soluções garantiram não apenas o funcionamento atual da aplicação, mas também a sua escalabilidade futura.

Com o projeto concluído, ficou evidente que a proposta inicial — transformar a coleta de dados médicos em uma atividade simples e motivadora — foi alcançada com sucesso. O app FibroApoio já se apresenta como um produto com potencial de uso real por pacientes e profissionais de saúde, e poderá evoluir ainda mais com testes de campo e *feedback* contínuo.

Como próximos passos, recomenda-se:

* Realizar testes com usuários reais em ambiente clínico supervisionado;
* Avaliar métricas de engajamento e retenção;
* Explorar a possibilidade de expandir para Android utilizando soluções multiplataforma.

## **Desafios Técnicos e Soluções**

**Sincronização de dados**: resolver a consistência entre Firestore, LocalStorage e variáveis do app exigiu padronizar todas as gravações no UserService, com persistência tripla e revalidação em cada carregamento.

**Reatividade nas *views***: precisei garantir que o uso de @Published em UserService fosse observado corretamente em todas as *views*, evitando chamadas desnecessárias.

**Navegação desacoplada**: para evitar NavigationLinks presos às *views*, criei um AppCoordinatorService reativo, que resolve a navegação via binding.

**Problemas silenciosos de persistência**: em alguns momentos, documentos não estavam sendo salvos no Firestore por ausência de createdAt. Resolvi isso adicionando verificações automáticas de campos obrigatórios ao salvar (persistWeeklyData()).

**Organização do código**: para evitar dependência circular, movi a lógica de pontuação para um GamificationService, centralizando a manipulação de *streaks*, pontuação e *rank.*

# Bibliografia

1. Clin Anat, Benefits of gamification in medical education, <https://pubmed.ncbi.nlm.nih.gov/35637557/>
2. Apple Developer Documentation, <https://developer.apple.com/documentation>
3. Firebase Official Documentation, <https://firebase.google.com/docs?hl=pt-br>
4. Figma Design System, <https://www.figma.com/>