UNIVERSIDADE
LUSÓFONA

Aplicação móvel para rastreio da hepatite C

Trabalho Final de Curso

Relatório final

Nome do Aluno: Ana Salvador a22301016
Nome do Aluno: Miguel Santos a22304909
Nome do Orientador: Pedro Alves

Trabalho Final de Curso | LEI | 20/06/2025

Direitos de cópia

Agradecimentos

Gostaríamos de expressar a nossa imensa gratidão aos profissionais da Ares do Pinhal, que confiaram em nós para levar adiante o desenvolvimento desta aplicação.
Um agradecimento especial ao nosso orientador, cujo acompanhamento constante, orientações precisas e apoio inestimável nos guiaram em cada etapa desta jornada.
Reconhecemos igualmente o contributo dos docentes do curso, que, ao transmitirem os seus conhecimentos e experiências, nos proporcionaram uma base teórica e prática robusta para enfrentar os desafios que surgiram ao longo do desenvolvimento deste Trabalho Final de Curso.
Não podemos deixar de mencionar o apoio da instituição de ensino, que nos forneceu os recursos e as ferramentas necessárias para alcançar os objetivos traçados.
Por fim, estendemos nossa gratidão a todos os que, de forma direta ou indireta, contribuíram para a realização deste trabalho, seja através de feedbacks, sugestões ou simples palavras de encorajamento.

Resumo

Este documento apresenta a conceção e desenvolvimento de uma aplicação móvel em colaboração com a Associação Ares do Pinhal, uma instituição com vasta experiência no apoio a pessoas com dependências. A aplicação tem como objetivo melhorar a monitorização e a adesão ao tratamento da hepatite C, especialmente para utentes com um histórico de toxicodependência. Focada na recolha e gestão eficiente de dados, a solução proposta visa apoiar os profissionais de saúde na prestação de cuidados mais precisos e personalizados, ao mesmo tempo que se prepara para ser adaptável a outras patologias.

Abstract

This document outlines the design and development of a mobile application in collaboration with Associação Ares do Pinhal, an institution with extensive experience in supporting individuals with substance dependencies. The application aims to enhance the monitoring and adherence to hepatitis C treatment, particularly for patients with a history of drug addiction. Focused on efficient data collection and management, the proposed solution seeks to assist healthcare professionals in providing more accurate and personalized care, while also being adaptable to other pathologies.

Conteúdo

Lista de Figuras

Lista de Tabelas

1 - Introdução

O problema em estudo resulta de circunstâncias reais, uma vez que a Associação Ares do Pinhal tem enfrentado desafios na monitorização da adesão ao tratamento de pacientes com hepatite C, agravados pelo registo manual de dados. A solução a desenvolver representa um passo significativo no sentido de superar estas dificuldades. Para fundamentar este trabalho, foram consideradas evidências científicas e contribuições de terceiros, incluindo os profissionais de saúde e utentes da Ares do Pinhal, que são diretamente impactados pelo problema.

1.1 Enquadramento

Este trabalho é uma continuação de um projeto anterior, que ficou incompleto. A Associação Ares do Pinhal, desde a sua fundação em 1986, tem desempenhado um papel crucial no tratamento de pessoas com dependências em Portugal. A instituição implementa métodos inovadores para mitigar os efeitos do abuso de substâncias, mas enfrenta desafios significativos na gestão e monitorização da adesão ao tratamento de doenças como a hepatite C.
Atualmente, a hepatite C é uma doença tratável, e a adesão rigorosa aos tratamentos prescritos é fundamental para garantir o sucesso terapêutico e salvar vidas. No entanto, a falta de um sistema de registo eficiente impede um acompanhamento eficaz, dificultando a avaliação da adesão e dos resultados dos pacientes.
O trabalho baseia-se em conceitos de sistemas de monitorização digital, registo de dados clínicos e intervenções personalizadas, todos suportados por referências científicas e pela experiência prática da Ares do Pinhal. Este projeto dá continuidade ao trabalho iniciado anteriormente, focando-se na criação de uma solução digital robusta que supere as limitações previamente identificadas e responda de forma eficaz às necessidades da instituição.
AdP utiliza atualmente, existe já uma aplicação web chamada *Zeus*, usada para registar uma série de informações relevantes no contexto clínico e administrativo — embora, até à data, não contemple o registo específico relacionado com a hepatite C.

1.2 Motivação e Identificação do Problema

1.2.1 Motivação

A motivação para este trabalho surge das dificuldades identificadas na monitorização do tratamento da hepatite C por parte da Associação Ares do Pinhal. O registo manual em folhas de papel apresenta várias limitações, que comprometem a qualidade do acompanhamento clínico. Além disso, existe uma oportunidade de melhorar os processos existentes, utilizando tecnologias digitais para facilitar o registo e a análise de dados. O impacto social e clínico de uma aplicação para este problema é significativo, podendo contribuir para melhores resultados no tratamento dos utentes.

1.2.2 Identificação do Problema

Perda de dados: Registos em papel são suscetíveis a perdas, danos ou extravios.
Erros e inconsistências: O registo manual aumenta o risco de enganos, como erros de dosagem ou omissões.
Dificuldades de rastreamento: Sem um sistema centralizado, monitorizar padrões de adesão é um desafio, dificultando intervenções rápidas e eficazes.

Estas lacunas prejudicam o acompanhamento dos tratamentos e comprometem a eficácia das intervenções da instituição.

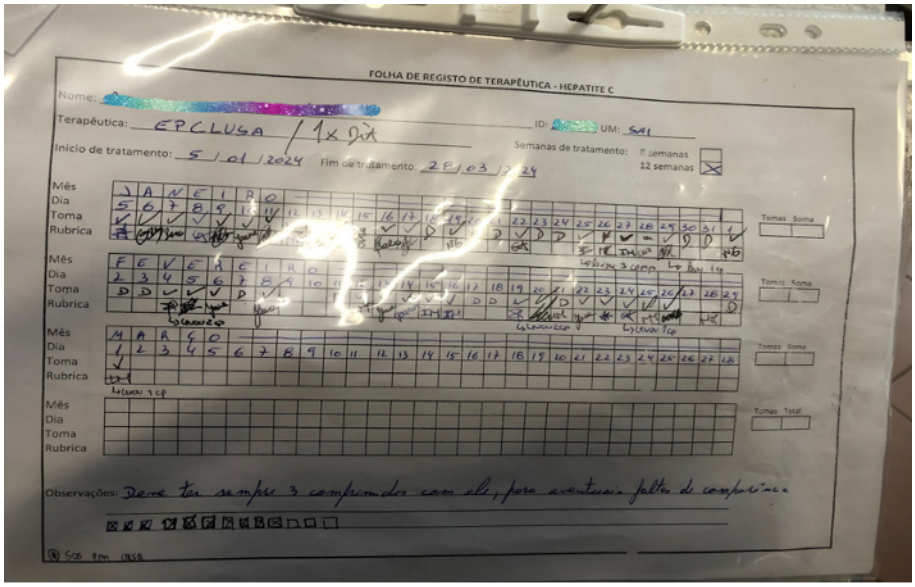


Figura 1.1: Registo dos dados

1.2.3 Problemas da aplicação desenvolvida anteriormente

A aplicação desenvolvida anteriormente visava solucionar os problemas acima, mas apresentou limitações que impediram sua utilização eficaz:

- **Instabilidade da aplicação:** Ocorrência frequente de erros críticos que resultavam em crashes, comprometendo a utilização da aplicação.
- **Interface gráfica deficiente:** Design pouco polido, com problemas de usabilidade, dificultando a navegação e a interação dos utilizadores.
- **Funcionalidades incompletas:** Algumas funcionalidades previstas, como o registo e acompanhamento das tomas de medicamentos através de um calendário interativo, não foram implementadas.
- **Falta de integração com sistemas externos:** A aplicação não incluía integração com a base de dados do Zeus, dificultando a sincronização de informações e a consistência dos registos clínicos.

Com base na identificação dos problemas gerais e nas limitações da solução anterior, este trabalho tem como objetivo aprimorar a aplicação, superando essas deficiências e alinhando-a às necessidades dos profissionais da Associação Ares do Pinhal (AdP).

1.3 Objetivos

1.3.1 Objetivo Geral

Desenvolver uma aplicação para monitorizar a adesão ao tratamento da hepatite C, integrando os processos existentes da Associação Ares do Pinhal.

1.3.2 Objetivos Específicos

- Criar uma aplicação para registo centralizado e eficiente dos dados dos utentes, minimizando perdas e erros.
- Facilitar o acompanhamento em tempo real da adesão ao tratamento, permitindo intervenções mais ágeis.
- Garantir que a aplicação se integre harmoniosamente nas práticas e iniciativas da Ares do Pinhal, assegurando a sua viabilidade e aceitação pelos profissionais de saúde e utentes.

1.4 Estrutura do Documento

Este relatório está organizado da seguinte forma:

- **Secção 1 - Introdução:** Apresenta a análise da viabilidade e pertinência do trabalho desenvolvido, incluindo o enquadramento, motivação, identificação do problema e os objetivos do projeto.
- **Secção 2 - Pertinência e Viabilidade:** Avalia a relevância e a viabilidade da aplicação desenvolvida, analisando como o projeto atende às necessidades da Associação Ares do Pinhal (AdP). Esta secção inclui uma análise comparativa com soluções existentes, identifica elementos inovadores e destaca as vantagens e benefícios da solução. Além disso, discute o potencial de adaptação da aplicação para outros contextos de saúde pública, mesmo sem fins comerciais.
- **Secção 3 - Especificação e Modelação:** Detalha as características da solução desenvolvida, com base na identificação de requisitos, descrição de elementos principais e exploração de cenários reais de utilização. Inclui a análise de requisitos, a enumeração e descrição detalhada dos requisitos principais e os modelos de interação da solução.
- **Secção 4 - Solução Proposta:** Apresenta a descrição funcional da solução desenvolvida, a arquitetura técnica, as tecnologias e ferramentas utilizadas, os ambientes de teste e produção, a abrangência académica e os detalhes técnicos dos componentes e webservices.
- **Secção 5 - Testes e Validação:** Descreve o processo de validação da aplicação, incluindo os testes realizados com base nos requisitos definidos e nas issues registadas durante o desenvolvimento. Aborda também os ambientes de teste utilizados, os dados simulados, e os contributos dos profissionais da AdP na validação da solução.
- **Secção 6 - Método e Planeamento:** Descreve a abordagem metodológica adotada no projeto, incluindo cronograma, distribuição de tarefas e métodos de gestão de projeto.
- **Secção 7 - Resultados:** Apresenta as funcionalidades implementadas durante o projeto, incluindo melhorias sugeridas após os testes com os profissionais da AdP.
- **Secção 8 - Conclusão:** Resume os principais contributos do projeto e o impacto positivo da aplicação na monitorização do tratamento da hepatite C.

2 - Pertinência e Viabilidade

A presente secção aborda a relevância e viabilidade da aplicação desenvolvida, demonstrando o impacto positivo esperado e avaliando a sua implementação em termos técnicos, económicos, sociais e ambientais. Inclui também uma análise comparativa com soluções existentes e discute a inovação, as mais-valias e potenciais oportunidades de negócio associadas ao projeto.

2.1 Pertinência

O projeto é altamente pertinente, pois responde diretamente às dificuldades identificadas na monitorização de tratamentos, especialmente no contexto da hepatite C, na Associação Ares do Pinhal (AdP).

Através da aplicação desenvolvida, espera-se:

- **Melhoria da adesão ao tratamento:** fornecendo ferramentas de rastreio e monitorização em tempo real.
- **Redução de erros e perdas de dados:** substituindo registos manuais por uma solução automatizada.
- **Impacto positivo nos cuidados de saúde:** permitindo intervenções mais ágeis e eficazes, aumentando a qualidade dos serviços prestados pela AdP.

O impacto esperado foi validado por feedback direto de profissionais de saúde e beneficiários da AdP, que destacaram a necessidade de uma solução prática e acessível. Os resultados preliminares indicam que a aplicação pode contribuir significativamente para a resolução dos problemas identificados.

2.2 Viabilidade

A viabilidade da aplicação proposta é avaliada com base em critérios técnicos, económicos, sociais e ambientais, garantindo que ela possa ser implementada e sustentada de forma eficaz.

2.2.1 Viabilidade Técnica

A aplicação é projetada para ser utilizada em dispositivos móveis, garantindo acessibilidade e facilidade de uso. Os testes com protótipos interativos demonstraram que:

- As funcionalidades atendem às expectativas dos utilizadores da AdP.
- A integração com bases de dados existentes é tecnicamente viável, facilitando a adoção da aplicação no ambiente real.

2.2.2 Viabilidade Económica

A aplicação é gratuita, eliminando barreiras de custo para os profissionais de saúde e pacientes. A análise de custo-benefício indica:

- Economia de recursos ao eliminar registos em papel.
- Redução de custos associados a erros clinicos.

2.2.3 Viabilidade Social

A viabilidade social da aplicação foi avaliada com base em diálogos e interações diretas com os profissionais da AdP. Durante o processo de levantamento de requisitos, os membros da associação manifestaram de forma clara a necessidade de uma aplicação que atendesse às especificidades de suas operações, especialmente no acompanhamento de pacientes e gestão de dados clínicos.

2.2.4 Alinhamento com os Objetivos de Desenvolvimento Sustentável (ODS)

A solução proposta alinha-se com diversos Objetivos de ODS, nomeadamente:

- **ODS 3 - Saúde de Qualidade:** ao melhorar o acompanhamento clínico e os resultados dos tratamentos.
- **ODS 10 - Redução das Desigualdades:** ao disponibilizar uma solução acessível e inclusiva para todos os pacientes.

2.3 Análise Comparativa com Soluções Existentes

2.3.1 Soluções Existentes

Uma análise das soluções disponíveis no mercado, como o Addiction-Comprehensive Health Enhancement Support System (A-CHESS), destaca funcionalidades como monitorização de saúde, ferramentas de comunicação, e conteúdos educacionais extensivos. Contudo, estas soluções nem sempre são acessíveis ou adaptadas às necessidades específicas da AdP, especialmente em termos de rastreio detalhado e custos.

2.3.2 Análise de Benchmarking

A tabela abaixo resume as principais características do A-CHESS e da aplicação desenvolvida:

Funcionalidades	A-CHESS	Aplicação proposta
Rastreio Detalhado	Não	Sim
Monitorização em Tempo Real	Sim	Sim
Gratuito	Não	Sim

Tabela 2.1: Tabela de comparações das aplicações

2.4 Proposta de Inovação e Mais-Valias

A aplicação destaca-se pela sua capacidade de:

- **Inovar na gestão de tratamentos:** introduzindo funcionalidades de rastreio e diagnóstico detalhados.
- **Melhorar a eficiência operacional:** reduzindo erros e otimizando os fluxos de trabalho na AdP.
- **Aumentar a acessibilidade:** ao ser disponibilizada gratuitamente.
- **Impacto social positivo:** ao responder diretamente às necessidades específicas dos profissionais da AdP e dos seus utentes.

Estas características diferenciam a solução das alternativas existentes, oferecendo uma abordagem personalizada.

2.5 Identificação de Oportunidade de Negócio

Embora esta aplicação tenha sido desenvolvida exclusivamente para atender às necessidades da AdP e não tenha finalidade comercial, é relevante destacar o seu potencial de impacto e escalabilidade no setor da saúde.

- **Mercado-Alvo Potencial:** Instituições de saúde pública e associações que atuam no acompanhamento de doenças crónicas ou infecciosas, como hepatite C, tuberculose ou HIV.
- **Contribuição para o Setor da Saúde:** Apesar da ausência de uma orientação comercial, o projeto reforça o compromisso com o desenvolvimento de soluções tecnológicas sustentáveis e adaptáveis, alinhadas com os ODS, promovendo impacto social significativo.

3 - Especificação e Modelação

Esta secção apresenta uma visão detalhada das características da solução desenvolvida, com base na identificação de requisitos, descrição de elementos principais e exploração de cenários reais de utilização.

3.1 Análise de Requisitos

A análise de requisitos identifica as características essenciais da solução para resolver o problema apresentado, considerando tanto os requisitos implementados quanto aqueles propostos para a continuidade do projeto em contextos académicos ou empresariais.

3.1.1 Enumeração de Requisitos

Os requisitos foram classificados com base na sua funcionalidade, prioridade e dificuldade de implementação. Cada requisito foi avaliado com critérios de aceitação para validação durante os testes. As tabelas correspondentes aps requisitos funcionais e não-funcionais podem ser consultadas em Tabela 3.1 e Tabela 3.2.

3.1.2 Descrição detalhada dos requisitos principais

Permitir introdução de novos testes de rastreio ou diagnostíco:

Introdução de novos testes de rastreio ou diagnostíco, incluindo informações como data, tipo de teste, local, resultado e data do resultado.

- **Dependências:** Interface de entrada para detalhes do teste.
  Validação dos dados fornecidos pelo utilizador.
  Base de dados para armazenar os dados dos testes.
- **Objetivos:** Fornecer aos técnicos uma maneira eficiente e precisa de registrar novos testes.
  Reduzir erros e simplificar a gestão de dados dos utentes.
- **Critérios de Aceitação:** Todos os campos obrigatórios devem ser preenchidos antes de submeter o teste.
  Não é permitido registrar datas futuras para os testes ou resultados.
  Dados válidos são armazenados com sucesso no sistema.
- **Casos de Uso Relacionado:** Adicionar Teste de Rastreio - o técnico adiciona detalhes do teste de um utente.

Suportar o registo diário da toma do medicamento:

Registrar diariamente se os utentes tomaram os medicamentos prescritos, onde foi tomado e quaisquer notas adicionais.

- **Dependências:** Interface para entrada dos dados diários das tomas.
  Estruturação da base de dados para armazenar o histórico das tomas.
  Integração com funcionalidades de gestão de tratamentos.
- **Objetivos:** Garantir um registro completo e preciso do cumprimento dos tratamentos pelos utentes.
  Facilitar o acompanhamento da adesão ao tratamento pelos técnicos.

- **Critérios de Aceitação:** Permitir registar a toma apenas para datas correntes ou anteriores.
  O registo deve incluir informações completas, como local e possíveis notas.
  O sistema deve sinalizar dias que as tomas não foram realizadas.
- **Casos de Uso Relacionado:** Registrar Toma do Medicamento - o técnico acessa e registra a toma do dia.

3.1.3 Casos de Uso/User Stories

Nesta seção, apresenta-se o Diagrama de Casos de Uso, que ilustra as principais interações entre o ator (utilizador) e as funcionalidades disponíveis no sistema. Este diagrama foi elaborado para descrever os processos de navegação e operação da aplicação, destacando as ações possíveis de acordo com as permissões e responsabilidades do utilizador.

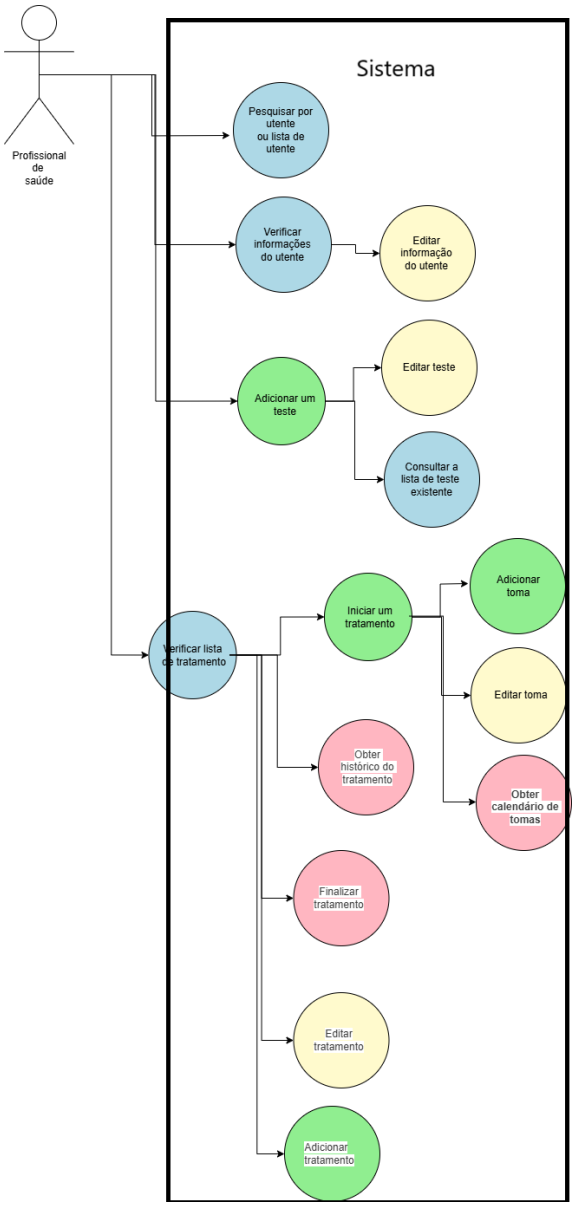


Figura 3.1: Casos de uso

ID Requisito	Funcionalidade	Prioridade	Dificuldade
FR1	Implementar um mecanismo de login seguro para utilizadores (técnico) aceder à aplicação	Alta	Média
FR2	Fornecer uma funcionalidade de pesquisa pelo nome ou id para os utilizadores, mostrando uma lista de utentes para a seleção do utilizador	Média	Baixa
FR3	Mostrar detalhes abrangentes dos utentes, incluindo nome completo, data de nascimento, data de entrada no último programa, ID de paciente (CC ou passaporte) e idade	Alta	Média
FR4	Exibir um histórico de testes de rastreio que inclua a data do teste, tipo de teste, localização onde foi realizado o teste, resultado e data do resultado	Alta	Média
FR5	Permitir aos utilizadores (técnicos) introduzirem um novo teste de rastreio com detalhes que incluam data do teste, tipo de teste, localização onde foi realizado o teste, resultado e data do resultado	Média	Média
FR6	Permitir aos utilizadores (técnicos) editar detalhes como data do teste, tipo de teste, localização onde foi realizado o teste, resultado, e data do resultado de um teste de rastreio já existente	Média	Média
FR7	Exibir um histórico de diagnóstico que inclua a data do teste, tipo de teste, localização onde foi realizado o teste, resultado e data de resultado	Alta	Média
FR8	Permitir aos utilizadores (técnicos) introduzirem um novo teste diagnóstico com detalhes que incluam data do teste, tipo de teste, localização onde foi realizado o teste, resultado e data do resultado	Média	Média
FR9	Permitir aos utilizadores (técnicos) editar detalhes como data do teste, tipo de teste, localização onde foi realizado o teste, resultado, e data do resultado de um teste de diagnóstico já existente	Média	Média
FR10	Suporte a campos de entrada para data de início do tratamento, elegibilidade para tratamento, seleção do nome do medicamento, próxima data de tratamento, data esperada de término do tratamento, data real de término do tratamento, notas/observações e análise do término do tratamento	Alta	Média
FR11	Suportar campos de entrada para um novo tratamento com os detalhes especificados em FR10	Alta	Média
FR12	Suportar edição dos campos especificados em FR10 de um tratamento atual	Alta	Média
FR13	Implementar um recurso para exportar dados para um ficheiro, de preferência no formato Excel, para facilitar a análise de dados e a geração de relatórios	Média	Média
FR14	Opcionalmente fornecer acesso a todas as páginas da aplicação através de um menu disponível em cada página para a conveniência do utilizador	Baixa	Média
FR15	Suportar o registo diário da toma do medicamento para pacientes, incluindo se o medicamento foi ou não tomado, se foi tomado em casa ou no SAI e qualquer tipo de notas relevantes	Alta	Média
FR16	Implementar um calendário que permite aos profissionais de saúde aceder e editar os registos diários de medicamentos clicando no dia em específico. Isto deve facilitar a navegação e modificação dos registos de tomas de medicamentos	Média	Alta
FR17	Implementar ecrã novo ”Utentes em tratamento”s,	Média	Alta

Tabela 3.1: Requisitos funcionais

ID Requisito	Funcionalidade	Prioridade	Dificuldade
NFR1	Ser intuitiva, ter interface amigável e garantir facilidade de utilização para o profissional de saúde	Alta	Baixa
NFR2	Implementar medidas de segurança robustas, especialmente durante o login do utilizador e a transmissão de dados, para proteger as informações dos pacientes e cumprir com as diretrizes de privacidade	Alta	Alta
NFR3	Garantir que a aplicação seja escalável para integrar-se com os sistemas da associação e suportar funcionalidades adicionais para o tratamento de pacientes	Alta	Média
NFR4	Operar de forma confiável, minimizando períodos de indisponibilidade e garantindo acesso consistente a informações críticas	Alta	Média
NFR5	Otimizar o desempenho da aplicação, particularmente no tratamento da recuperação de dados, para proporcionar uma experiência de utilização fluida	Média	Média
NFR6	Garantir a interoperabilidade com diferentes sistemas e tecnologias, suportando os requisitos de integração com os sistemas existentes da associação	Alta	Média

Tabela 3.2: Requisitos não funcionais

3.2 Modelação

Nesta secção, apresenta-se o Diagrama de Entidade-Relacionamento (ER) da aplicação, representando a estrutura do modelo de dados utilizado para gerir o rastreio da hepatite C. A figura ilustra 3.2 as entidades principais da aplicação.

3.2.1 Relações entre as entidades:

Segue-se uma descrição sumária das entidades e das suas relações:

- **Tabela user (staff) ↔ Tabela userRole:** Cada técnico ou membro da equipa (user) pode possuir diferentes permissões definidos na tabela userRole.

- **Tabela user (staff) ↔ Tabela test:** Técnicos são responsáveis por realizar ou registar testes de rastreio ou diagnóstico na tabela test.

- **Tabela patient ↔ Tabela test:** A tabela test associa informações sobre os testes realizados por cada paciente (zeusId), incluindo tipo, data e resultado.

- **Tabela patient ↔ Tabela treatment:** A tabela treatment guarda informações sobre os tratamentos associados a cada paciente, como datas de início e fim, medicação prescrita e motivos de abandono.

- **Tabela treatment ↔ Tabela dailyMedicine:** A tabela dailyMedicine acompanha o histórico diário da adesão ao tratamento (tomas de medicação) dentro de um período específico para o paciente.

- **Tabela user (staff) ↔ Tabela comment:** Técnicos podem deixar comentários nos tratamentos para documentar observações ou registrar informações adicionais.

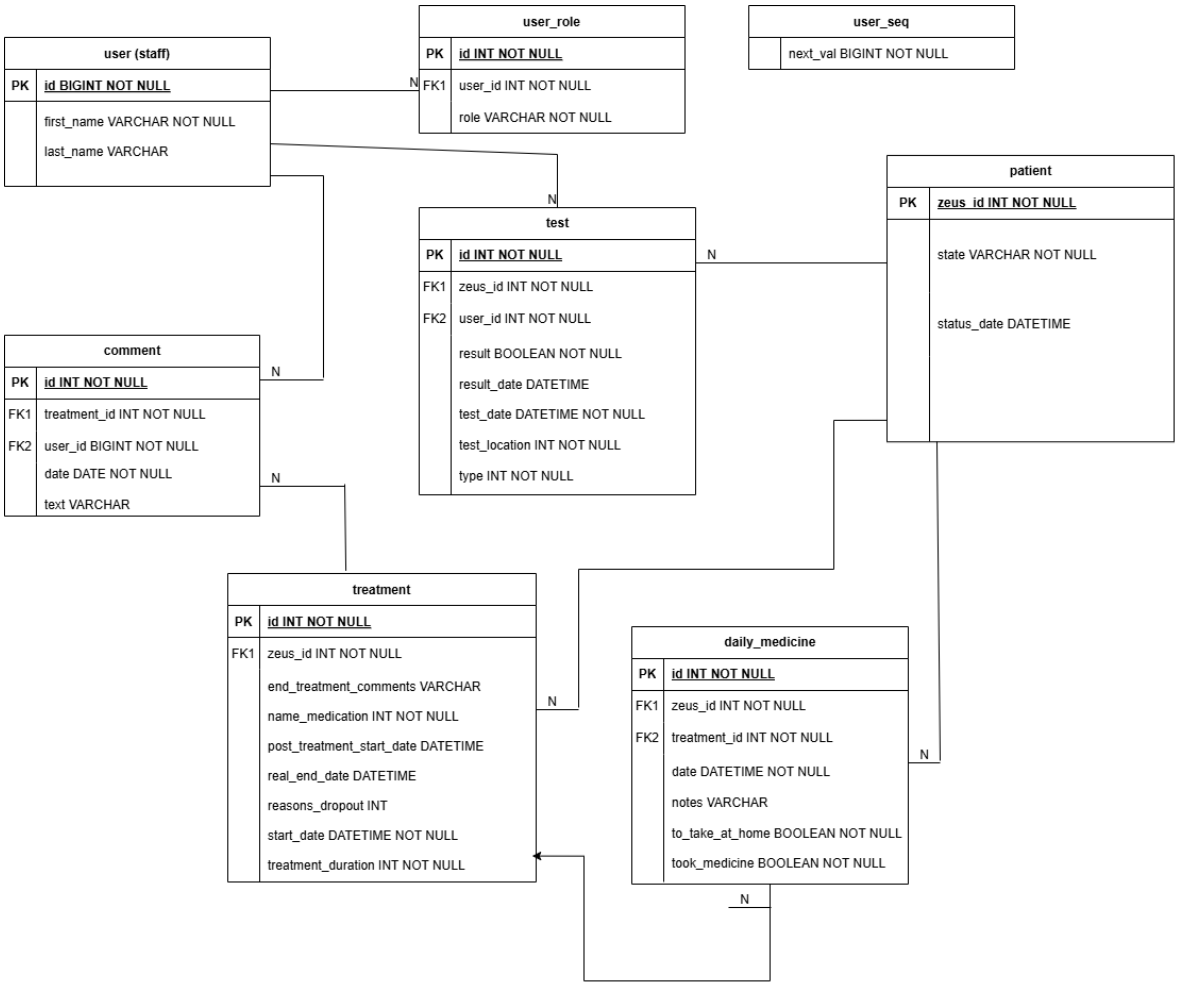


Figura 3.2: Diagrama entidade-relação

- **Tabela treatment ↔ Tabela comment:** Comentários específicos são associados a um tratamento para acompanhar detalhes, como motivos de abandono ou progresso do paciente.
- **Tabela userSeq:** Utilizada para gerir sequências únicas para identificação de utilizadores no sistema, garantindo unicidade no identificador de técnicos.

3.3 Protótipos de Interface

Nesta secção, apresenta-se o Mapa Aplicacional da aplicação, que ilustra a estrutura lógica e hierárquica dos principais ecrãs de navegação disponíveis ao utilizador. O mapa tem como objetivo fornecer uma visão técnica do fluxo de interação do utilizador, detalhando as relações entre os módulos funcionais e as ações possíveis em cada etapa.

3.3.1 Nível 0 - Login e Acesso ao Menu Principal

O utilizador inicia a interação ao realizar a autenticação via credenciais válidas. Após a validação, é direcionado ao menu principal, que atua como ponto de acesso às funcionalidades principais.

3.3.2 Nível 1 - Navegação Geral

O menu principal apresenta duas funcionalidades centrais:

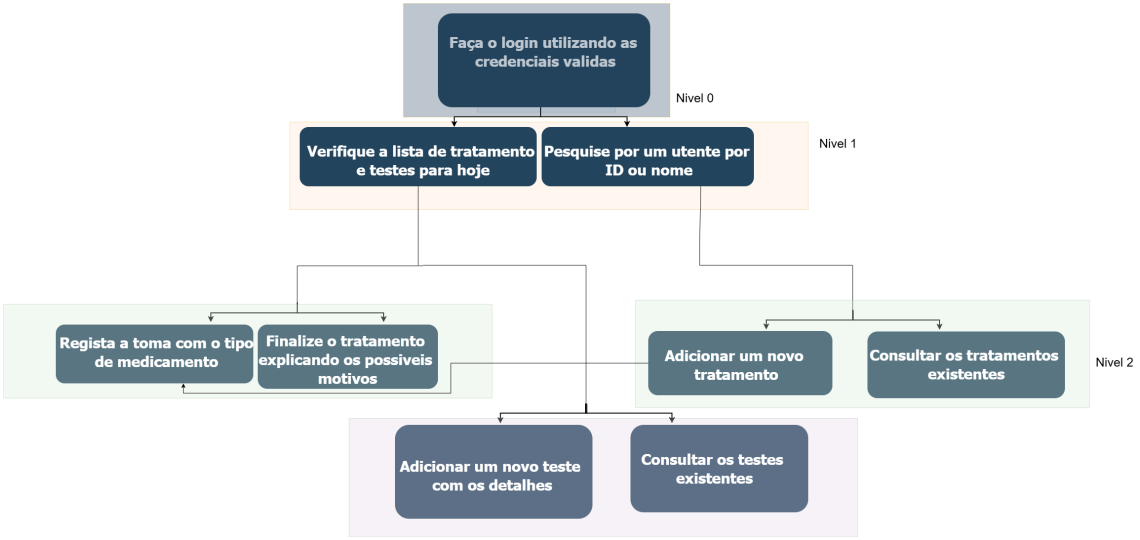


Figura 3.3: Mapa aplicacional

- **Listagem de tratamentos para o dia atual**: permite o registo de tomas e a finalização de tratamentos.
- **Pesquisa de utentes**: Permite pesquisar utentes por ID ou nome, existente no Zeus para ações relacionadas a testes e tratamentos.

3.3.3 Nível 2 - Funcionalidades Detalhadas

A partir das opções intermediárias, o utilizador pode:

- **Testes:**
  - Adicionar novos registos de testes ou consultar o histórico existente.
  - Alterar o estado do utente com base nos resultados dos testes.
- **Tratamentos:**
  - Criar novos tratamentos ou consultar os tratamentos existentes.
  - Registar tomas de medicação associadas a tratamentos ativos.
  - Finalizar tratamentos, incluindo a possibilidade de justificar desistências.

No âmbito do presente trabalho, não foi necessário o desenvolvimento de mockups para a interface gráfica da aplicação. Este facto justifica-se pelo motivo de que o design da interface já foi previamente definido e documentado no contexto de um Trabalho Final de Curso (TFC) anterior, elaborado por Sofia Caldas.

4 - Solução Proposta

4.1 Apresentação

A presente entrega corresponde à versão funcional da aplicação, com várias funcionalidades já implementadas e validadas em ambiente de testes. Esta versão representa um MVP (Minimum Viable Product), desenvolvido com base nas necessidades reais da Associação Ares do Pinhal, validando o conceito proposto inicialmente.
Ao longo do desenvolvimento, foram feitas adaptações e decisões técnicas que resultaram numa solução mais alinhada com o contexto operacional da associação, como é possível observar na comparação com as propostas iniciais discutidas nos primeiros meses do projeto.
Abaixo apresentam-se os recursos complementares que documentam e demonstram o funcionamento da solução:

- **Vídeo demonstrativo dettalhado da aplicação (video anterior):** ver vídeo aqui
- **Vídeo demonstrativo da aplicação em funcionamento (melhorado):** ver vídeo aqui
- **Repositório Git com o código-fonte:** Frontend e Backend para teste
- **Credenciais de acesso:**
  - **Utilizador:** user
  - **Palavra-passe:** password

4.2 Arquitetura

A arquitetura da solução foi projetada com foco na simplicidade, eficiência e escalabilidade, garantindo uma integração eficaz entre os seus componentes principais, como se pode observar na figura 4.1. Esta abordagem visa assegurar um desempenho otimizado e uma manutenção facilitada, alinhada às necessidades funcionais e técnicas do sistema. Os principais componentes da arquitetura incluem:

- **Frontend:** Para o desenvolvimento do frontend, será utilizado o Flutter, uma framework que permite a criação de interfaces de utilizador modernas e responsivas. Esta tecnologia assegura uma experiência consistente e agradável em dispositivos com sistemas operativos iOS e Android.
- **Backend:** O backend será fornecido pelo *Zeus*, uma aplicação desenvolvida pela AdP, sobre a qual não temos controlo direto. A comunicação com o *Zeus* será realizada através de um conjunto de webservices propostos por nós, cuja especificação detalhada será apresentada na Secção 4.7.

  No entanto, como a integração com o *Zeus* ainda não se encontra concluída, foi desenvolvido um backend local de apoio durante a implementação. Este backend de testes permitiu validar funcionalidades da aplicação e garantir o seu correto funcionamento enquanto a ligação ao sistema final não está operacional.

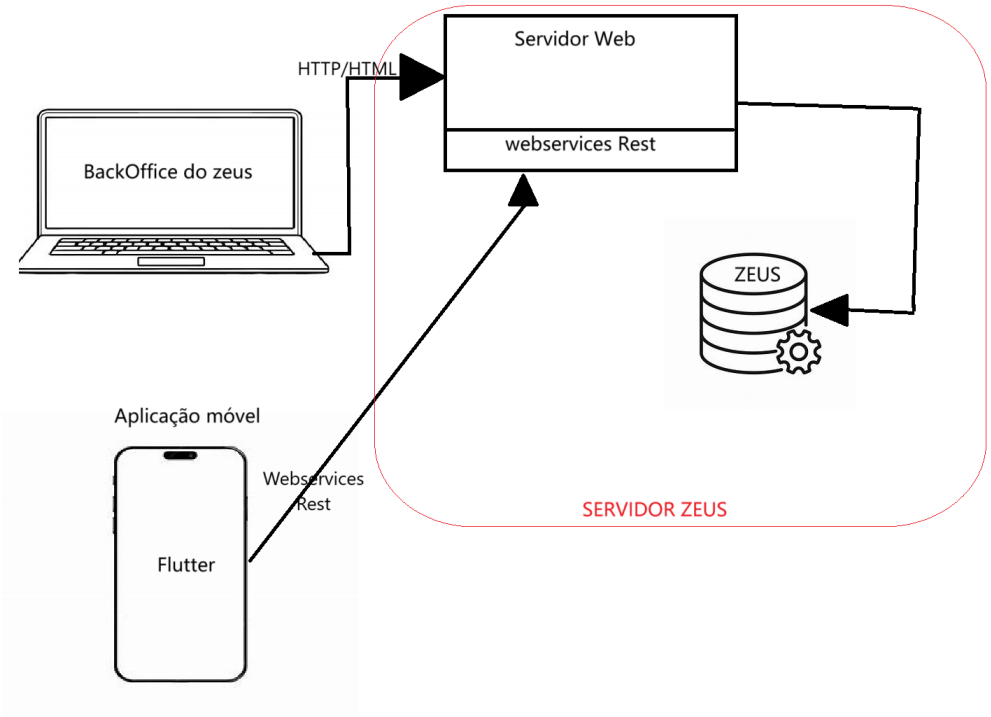


Figura 4.1: Arquitetura da aplicação

4.3 Tecnologias e Ferramentas Utilizadas

Atendendo às necessidades dos utilizadores e aos requisitos definidos, foram selecionadas tecnologias e ferramentas específicas para o desenvolvimento da aplicação.

4.3.1 Tecnologias

- **Flutter e Dart:** O Flutter, um framework open-source que foi utilizado para o desenvolvimento do frontend da aplicação. Ele oferece uma experiência de utilizador consistente e de alto desempenho em dispositivos iOS e Android. A linguagem de programação Dart, que serve como base para o Flutter, foi escolhida pela sua eficiência na criação de interfaces gráficas interativas e dinâmicas.

- **Kotlin:** O Kotlin foi utilizado para o desenvolvimento do backend da aplicação. Trata-se de uma linguagem moderna e concisa que, em conjunto com o Spring Framework, facilita a implementação de APIs e de integração da aplicação.

4.3.2 Ferramentas

- **Android Studio:** O Android Studio foi utilizado como ferramenta principal para o desenvolvimento e teste do frontend, oferecendo recursos como um emulador integrado e suporte ao Flutter para criar uma interface de utilizador intuitiva e eficiente.

- **Intellij IDA:** O IntelliJ IDEA foi a ferramenta escolhida para o desenvolvimento do backend e da integração com os serviços web. Além disso, foi utilizado para configurar e gerenciar a base de dados, garantindo um ambiente de desenvolvimento integrado e produtivo.

4.4 Ambientes de Teste e de Produção

A aplicação está configurada para comunicar diretamente com o servidor Zeus, utilizado como infraestrutura central para processamento de dados e armazenamento. Este servidor, que é gerido pela AdP, será utilizado tanto em ambiente de teste quanto em ambiente de produção.
A aplicação já foi disponibilizada (total ou parcialmente) em ambiente de testes internos nas lojas Android e iOS, permitindo à equipa validar funcionalidades.

4.5 Abrangência

A solução proposta integra conhecimentos adquiridos em diversas unidades curriculares ao longo do curso, aplicados diretamente no desenvolvimento da aplicação. Abaixo são descritas as contribuições de cada unidade:

- **Fundamentos de Programação:** Introduziu os princípios básicos de programação, que serviram como base para o desenvolvimento do *backend* da aplicação utilizando Kotlin, permitindo a implementação de lógica robusta e eficiente.
- **Base de Dados:** Forneceu o conhecimento necessário para o design e a implementação do sistema de base de dados. Isso incluiu a modelagem, normalização e persistência de dados, essenciais para garantir a integridade da aplicação.
- **Engenharia de Requisitos e Testes:** Proporcionou habilidades para levantamento de necessidades, definição de requisitos e critérios de aceitação.
- **Engenharia de Software:** Contribuiu com técnicas e metodologias de planeamento, arquitetura e integração de componentes do sistema.
- **Computação Móvel:** Forneceu conhecimentos específicos para o desenvolvimento de *frontends* utilizando Flutter, permitindo a criação de uma aplicação móvel compatível com sistemas iOS e Android, com interfaces modernas e responsivas.

4.6 Componentes

A aplicação móvel comunica com o servidor através de webservices, que estão estruturados em componentes responsáveis por diferentes áreas de funcionalidade no sistema. Esses componentes são descritos abaixo, destacando as suas responsabilidades específicas e interações:

4.6.1 DailyMedicineController

Responsável por gerir os registros de ”daily medicine”dos pacientes. Isso inclui a criação e atualização de registros de medicação diária, além de fornecer acesso às informações sobre medicação diária de um paciente específico em uma data determinada.

4.6.2 PatientController

Responsável por gerir as informações dos pacientes, incluindo funcionalidades de pesquisa por ID ou nome, verificação do status de tratamento e pós-tratamento, e obtenção de dados sobre o último teste realizado.

4.6.3 TestController

Responsável por gerir os registros de testes dos pacientes, incluindo a criação de novos testes, atualização de testes existentes, e a recuperação de testes realizados por um paciente específico ou por todos os pacientes.

4.6.4 TreatmentController

Responsável por gerenciar os tratamentos dos pacientes, incluindo a criação de novos tratamentos, término de tratamentos (com possível registro de desistência), e recuperação do histórico e do tratamento atual de um paciente.

4.7 Lista dos web services

Nesta secção, apresenta-se a lista dos webservices propostos para integrar a aplicação com o sistema Zeus da AdP.

- **(GET) validateCredentials (/api/validateCredentials)**: responsável pela validação das credenciais de autenticação do utilizador. Ao ser invocado, ele verifica as credenciais enviadas no cabeçalho HTTP utilizando o método Basic Auth. Caso as credenciais sejam válidas, o serviço retorna uma resposta com o código HTTP 200. Se as credenciais forem inválidas, a resposta será o código HTTP 401, indicando que a autenticação falhou.

- **(GET) getPatientsInTreatment (/api/patient/activeTreatment)**: permite obter uma lista de todos os utentes que estão atualmente em tratamento para hepatite C. O serviço realiza uma consulta à base de dados e retorna as informações dos pacientes ativos, ou seja, aqueles que têm um tratamento em curso.

- **(POST) insertDailyMedicine (/api/dailyMedicine/new)**: recebe um pedido para criar ou atualizar um registo de ”toma de medicamento”no tratamento da hepatite C para um determinado paciente. Ao receber a solicitação, o serviço verifica se já existe um registo de dailyMedicine para a data atual. Se não houver nenhum registo existente, o serviço cria um novo com os detalhes fornecidos. Caso já exista um registo para o dia, o serviço procede à atualização do registo atual com as novas informações.

- **(GET) getPatients (/api/patient/search/input)**: permite a pesquisa de pacientes no sistema com base em um identificador fornecido.

- **(POST) insertNewTest (/api/tests/new)**: recebe um pedido para criar um novo registo de teste de rastreio para um utente, identificado pelo zeusID, que é pesquisado através do repositório de pacientes. Caso o utente não exista no sistema, um novo registo de utente é criado automaticamente, com as informações fornecidas na solicitação. O teste inclui dados como tipo e resultado, e o estado do utente será alterado com base nessas informações.

- **(GET) getLastScreening (/api/patient/lastScreening/zeusId)**: recupera o último teste realizado por um paciente, identificado pelo zeusID fornecido como parâmetro.

- **(GET) getTestsByPatient (/api/tests/patient/zeusId)**: retorna todos os testes realizados por um paciente.

- **(POST) updateTest (/api/tests/update)**: permite a atualização das informações de um teste existente.

- **(POST) createTreatment (/api/treatment/new)** : cria um novo registo de tratamento para um utente, associado ao paciente. Ao registrar o tratamento, o sistema altera automaticamente o estado do utente para ”em tratamento”

- **(GET) getCurrentTreatment (/api/treatment/current/zeusId)**: retorna o tratamento atual de um paciente, incluindo todas as tomas associadas a esse tratamento.

- **(POST) endTreatment (/api/treatment/endTreatment)**: responsável por encerrar o tratamento atual de um utente. Ao ser acionado, ele registra notas finais relacionadas ao tratamento, incluindo informações sobre o motivo de uma possível. Além disso, o estado do utente é alterado para ”pós-tratamento”.

- **(GET) getTreatmentHistory (/api/treatment/history/zeusId)**: retorna o histórico completo de tratamentos de um paciente.

- **(GET) getDailyMedicineHistory (/api/dailyMedicine/historyzeusId) (/api/treatment/history/zeusId)**: Retorna o histórico completo de medicação diária associada ao utente com o ID Zeus fornecido como parâmetro.

- **(GET) getPatientsWithoutDailyMedicineToday (/api/dailyMedicine/patientsWithoutDailyMedicineToday)**: Retorna uma lista de utentes que se encontram em tratamento mas que ainda não receberam medicação diária na data atual.

4.7.1 Ecrãs Desenvolvidos e Justificação Funcional

A presente subsecção apresenta os principais ecrãs da aplicação, organizados de acordo com o mapa aplicacional descrito anteriormente. Para cada ecrã, é feita uma breve explicação do seu propósito funcional, da lógica de funcionamento.
O objetivo é demonstrar como as funcionalidades foram implementadas na prática, destacando as dificuldades técnicas ultrapassadas. Para reforçar a compreensão da solução desenvolvida, além da explicação textual e das imagens dos ecrãs, foi também adicionado um vídeo demonstrativo que evidencia o fluxo de utilização da aplicação.

- **Ecrã do login:**
  Este ecrã é apresentado logo no arranque da aplicação. O utilizador insere as credenciais válidas e, após a autenticação com sucesso, é redirecionado para a lista de tratamentos agendados para o dia atual (caso existam). Na ausência de tratamentos, o utilizador tem a possibilidade de pesquisar por utentes manualmente. Como se pode observar nas figuras 4.2, 4.3 e 4.4.

- **Ecrã Tratamentos para hoje:**
  Neste ecrã, são apresentados os tratamentos agendados para o dia atual. Para cada utente listado, o utilizador pode registar a toma da medicação, indicando o local onde esta foi realizada — ”unidade”ou ”casa”. Esta seleção é exclusiva, ou seja, apenas um dos locais pode ser escolhido por toma. Como se pode observar nas figuras 4.5, 4.6 e 4.7.

- **Ecrã Pesquisar Utentes:**
  Neste ecrã, é disponibilizada uma barra de pesquisa que permite ao utilizador procurar utentes pelo nome. A pesquisa por ID encontra-se sob responsabilidade dos técnicos do Zeus. Existe ainda a opção ”Search”, que permite listar todos os utentes existentes na base de dados. Como se pode observar nas figuras 4.8, 4.9 e 4.10.

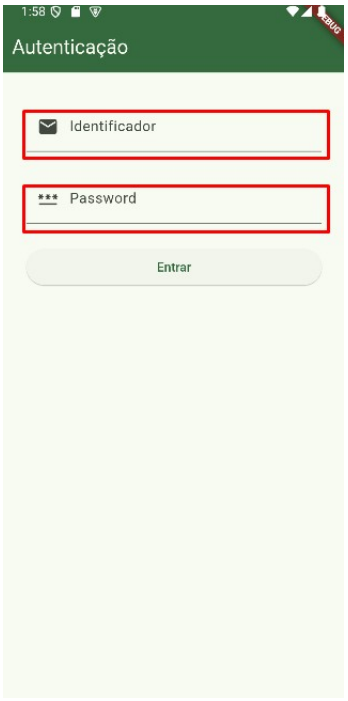


Figura 4.2: Ecrã do login

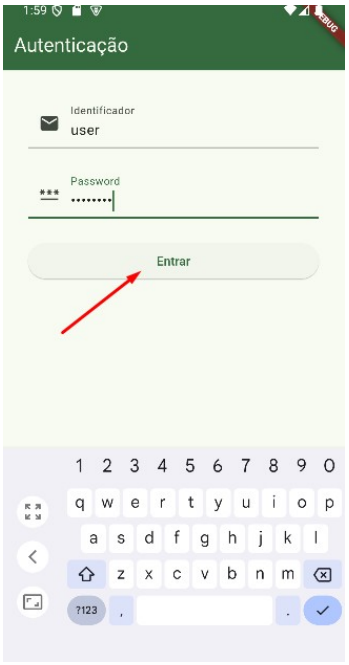


Figura 4.3: Login preenchido

Figura 4.4: Login inválido

- **Ecrã Adicionar Novo Teste:**
  Este ecrã permite adicionar um novo teste ao utente selecionado. Para concluir a operação, é obrigatório o preenchimento de todos os campos.
  Caso o resultado do teste seja reativo, torna-se posteriormente possível iniciar um tratamento para o utente.
  Se o resultado for não reativo, o utilizador é redirecionado novamente para o ecrã de adição de novo teste. Como se pode observar nas figuras 4.11, 4.12, 4.13, 4.14 e 4.15.

- **Ecrã Tratamento:** Neste ecrã, o utilizador pode optar por iniciar um novo tratamento ou adicionar um teste ao utente selecionado. Como se pode observar nas figuras 4.16.

  Caso o utilizador escolha iniciar o tratamento, todos os campos obrigatórios devem ser devidamente preenchidos. Após o registo bem-sucedido, o utilizador será redirecionado para:

  - **Calendário das Tomas:**
    Neste ecrã, o utilizador pode registar a toma diária de medicação ou finalizar o tratamento de um determinado utente, conforme a evolução do plano terapêutico, pode tornar explicito o motivo da desistência. Como se pode observar nas figuras 4.17.

  - **Detalhes do Tratamento:**
    Neste ecrã são apresentados os detalhes previamente preenchidos aquando do registo do tratamento, permitindo ao utilizador consultar a informação associada ao processo do utente. Como se pode observar nas figuras 4.18.

- **Ecrã Pós-Tratamento:**
  Este ecrã permite o registo do teste pós-tratamento, que é realizado 90 dias após o fim do tratamento, exclusivamente no hospital. Os resultados possíveis são: Cura ou Não Cura, conforme ilustrado nas Figuras 4.18 e 4.19.

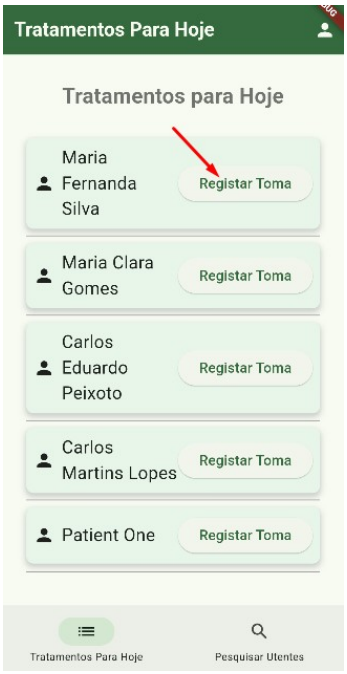


Figura 4.5: Ecrã tratamentos para hoje

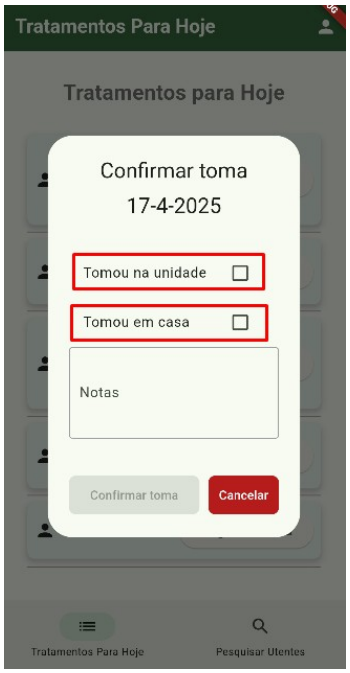


Figura 4.6: Registo da toma

Figura 4.7: Local da realização da toma

- **Ecrã do Histórico:**
  Este ecrã apresenta o histórico completo de testes e tratamentos realizados pelo utente. Permite ao utilizador consultar, de forma organizada e cronológica, todos os registos associados, facilitando o acompanhamento da evolução clínica. Como se pode observar nas figuras 4.21.

- **Ecrã Dados do Utente:**
  Este ecrã apresenta as informações essenciais do utente, nomeadamente o nome completo, número de identificação (ID) e género. Estes dados permitem ao utilizador confirmar rapidamente a identidade do utente selecionado antes de proceder a qualquer ação, como a adição de testes ou o início de um tratamento. Como se pode observar nas figuras 4.22 e 4.23.

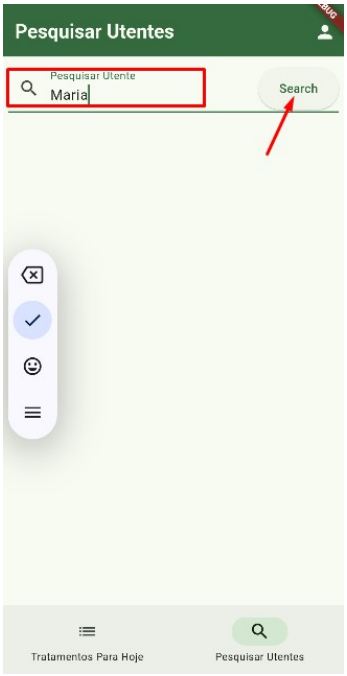


Figura 4.8: Ecrã de pesquisa por nome

Figura 4.9: Pesquisa por nome

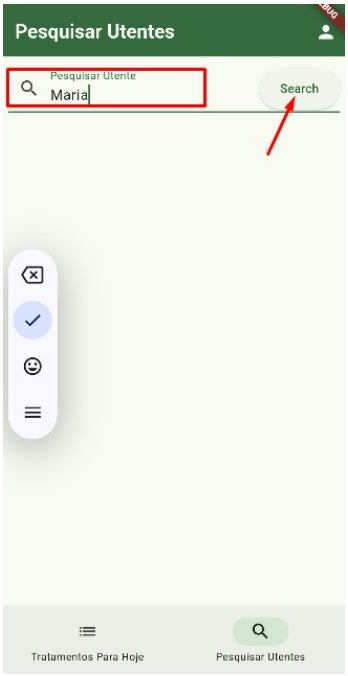


Figura 4.10: Lista dos utentes

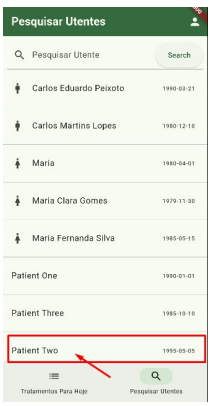


Figura 4.11: Adicionar novo teste

Figura 4.12: Adicionar novo teste

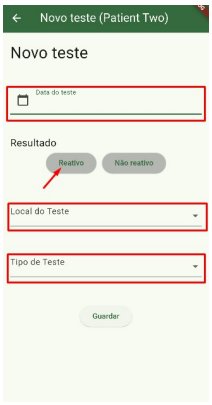


Figura 4.13: Adicionar novo teste

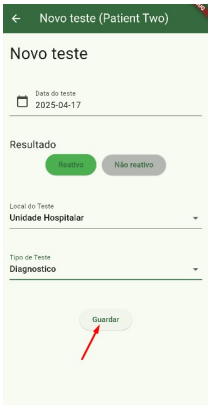


Figura 4.14: Adicionar novo teste

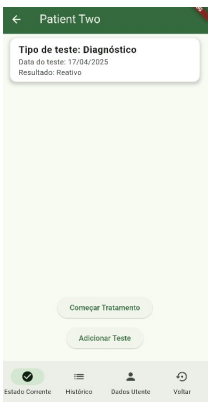
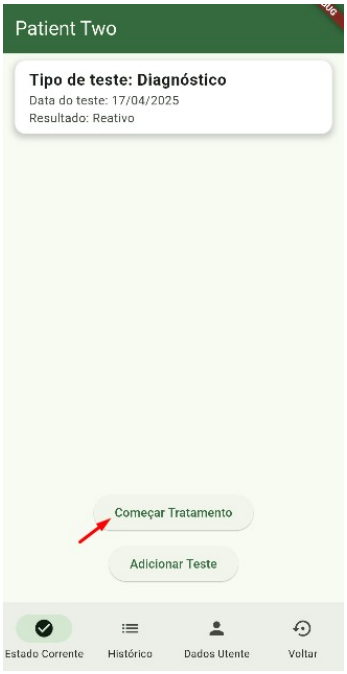


Figura 4.15: Adicionar novo teste



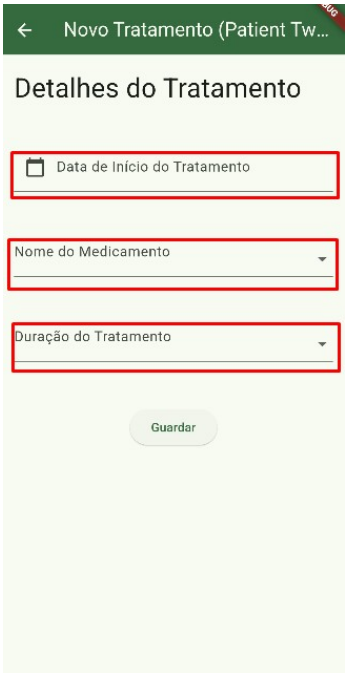
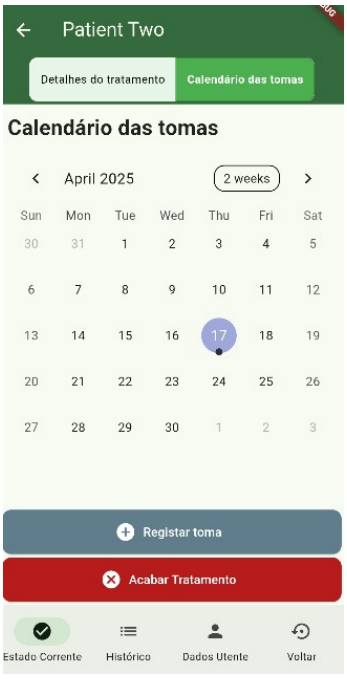
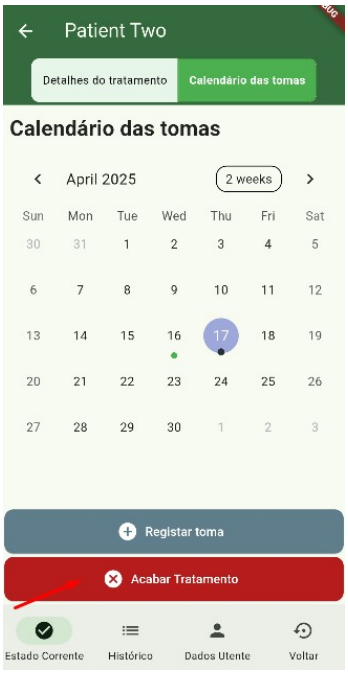


Figura 4.16: Ecrã do tratamento





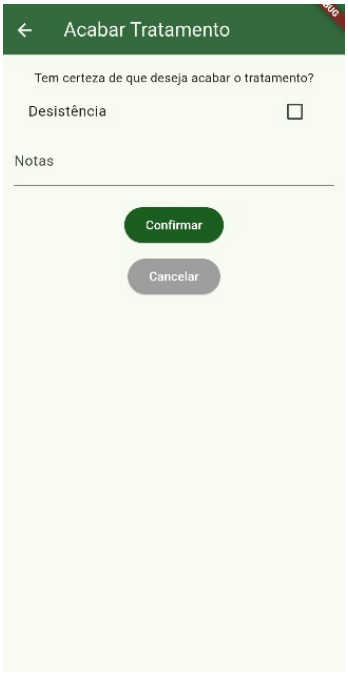


Figura 4.17: Calendário das tomas

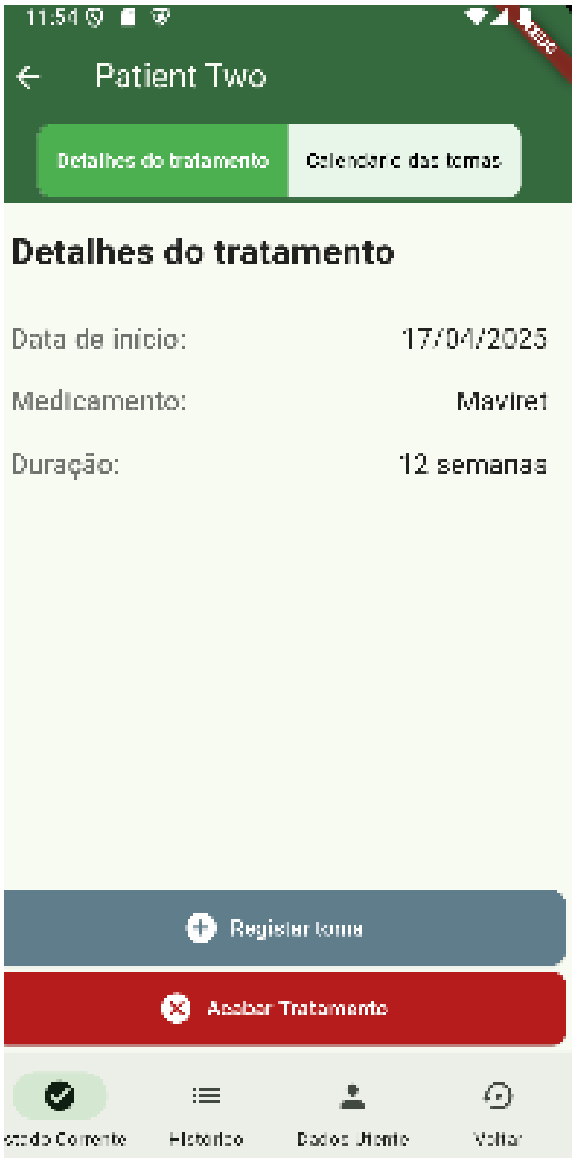


Figura 4.18: Detalhes do tratamento

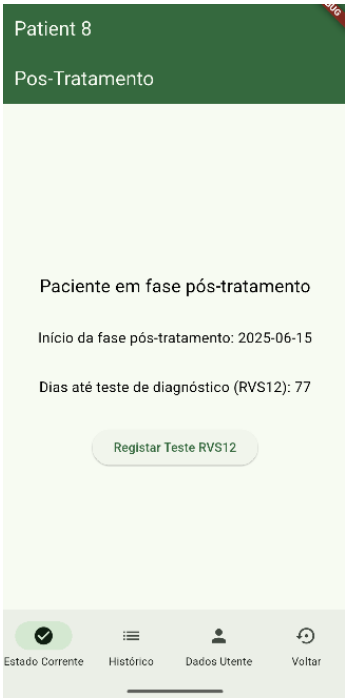


Figura 4.19: Ecrã dos pós tratamento

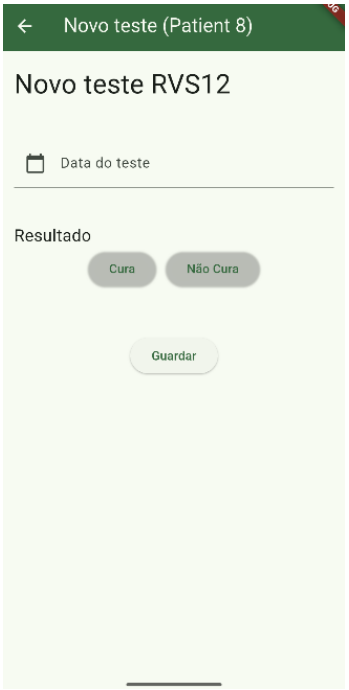


Figura 4.20: Novo teste RVS12

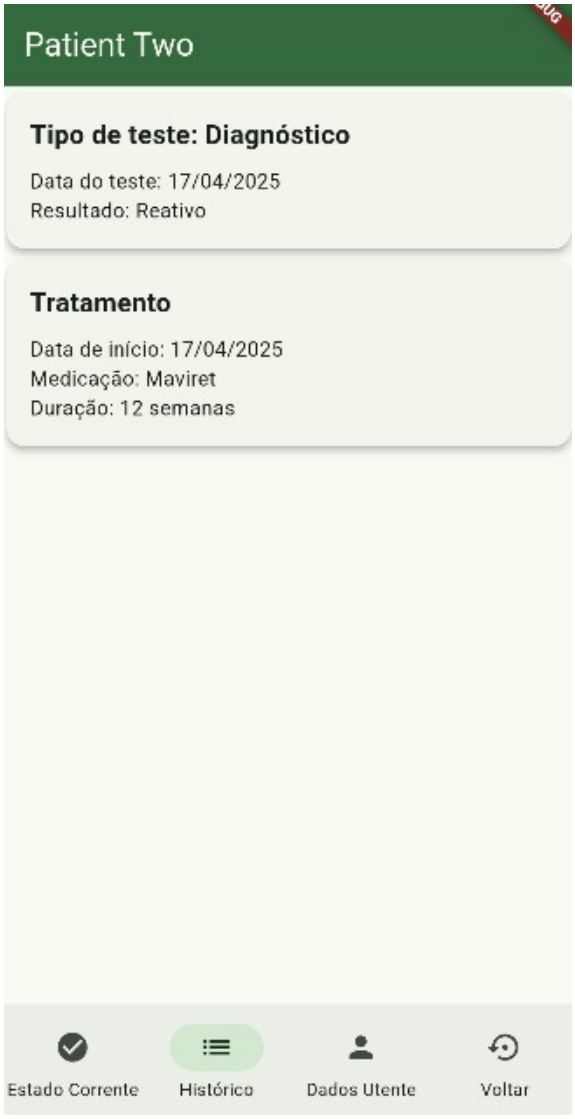


Figura 4.21: Histórico dos testes e dos tratamentos

Figura 4.22: Ecrã com dados do utente

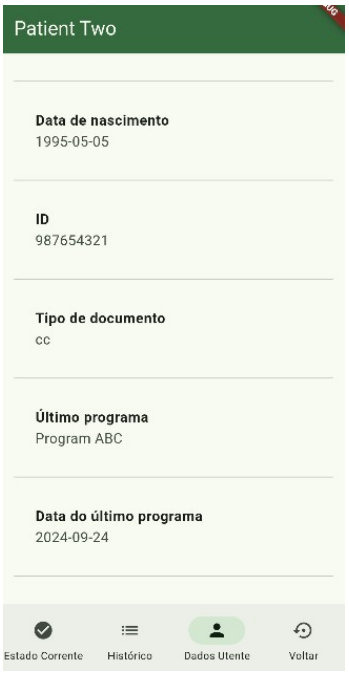


Figura 4.23: Ecrã com dados do utente

5 - Testes e Validação

A validação da solução desenvolvida foi conduzida com base nos requisitos funcionais previamente definidos em colaboração com os profissionais da Associação Ares do Pinhal. O objetivo principal dos testes foi garantir que a aplicação não só funcionasse corretamente, como também respondesse de forma eficaz às necessidades reais da associação.

5.1 Metodologia de Testes

Para garantir uma abordagem prática e orientada aos objetivos, adotámos um plano de testes baseado na criação de *issues* correspondentes a cada funcionalidade implementada. Cada *issue* foi testada em ambiente real de utilização.

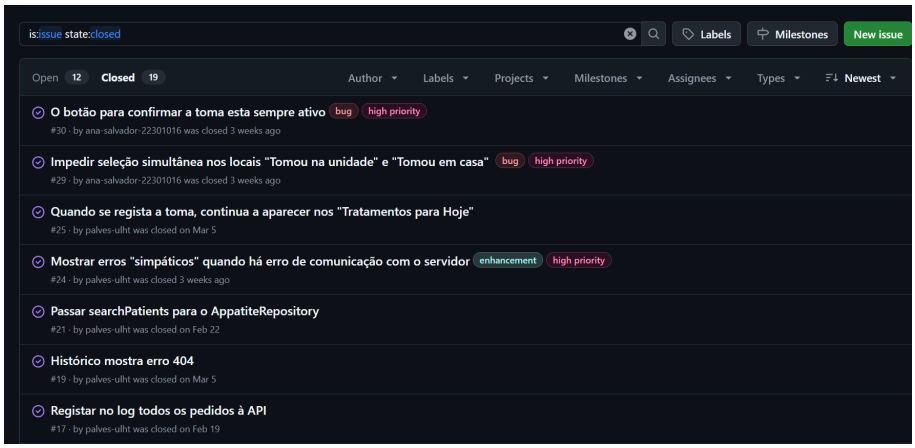


Figura 5.1: Algumas issues

A estratégia de validação centrou-se em três eixos fundamentais:

- **Validação funcional**: Testes realizados com foco no correto funcionamento das funcionalidades, com base nos critérios de aceitação definidos.
- **Usabilidade e aplicabilidade**: Recolha de feedback junto dos técnicos da Ares do Pinhal, avaliando a facilidade de uso e adequação às práticas de trabalho da equipa.
- **Validação operacional**: Testes efetuados em ambiente de testes internos (com deploy parcial nas lojas Android e iOS), utilizando a infraestrutura do servidor Zeus.

5.2 Resultados dos Testes e Melhorias Propostas

Através da demonstração da aplicação aos profissionais da Ares do Pinhal, foi possível identificar oportunidades de melhoria que originaram os seguintes desafios propostos pela equipa:

- **Persistência de Login:** Na versão atual da aplicação, o utilizador precisa de realizar autenticação sempre que reabre a aplicação. Foi-nos lançado o desafio de implementar um sistema de *login persistente*, para melhorar a experiência do utilizador.
- **Avisos Automáticos Relacionados com a Adesão ao Tratamento:** Foi solicitado um sistema de alertas automáticos nos seguintes casos:

  - Quando um utente acumula **quatro dias consecutivos sem registo de toma**, deverá ser apresentado um **aviso na página inicial** da aplicação. Este aviso só desaparecerá após a realização de um novo teste.
  - Quando a **data de fim do tratamento ultrapassa os 90 dias**, deverá ser apresentado um alerta indicando a necessidade de reavaliação do caso clínico.

- **Validação do Teste Pós-Tratamento:** Atualmente, a aplicação permite o registo do teste pós-tratamento (teste de seguimento) em qualquer momento. Foi-nos proposto garantir que este teste só possa ser realizado **90 dias após a conclusão do tratamento**, através de uma validação temporal que respeite o protocolo clínico e o teste passa a ter o nome de Teste RVS12 .

5.3 Ambiente de Testes e Recursos

Todos os testes foram realizados em ambiente de testes, utilizando a infraestrutura de servidores disponibilizada pela universidade, uma vez que, até ao momento, a aplicação ainda não foi integrada com o servidor Zeus, previsto para o ambiente de produção. A aplicação foi instalada em dispositivos móveis Android e iOS, com registos reais simulados (sem dados reais de utentes), garantindo que a performance, armazenamento e comunicação com os web services foram adequadamente validados.

5.4 Validação e Resultados

Embora não tenha sido elaborado um guião de testes formal, a validação da aplicação foi realizada com base nos requisitos previamente definidos em conjunto com a AdP e nas *issues* levantadas durante o desenvolvimento. Esta abordagem permitiu testar os fluxos principais da aplicação, incluindo:

- A correção de erros funcionais, como a persistência do botão de confirmação de toma e o erro 404 ao aceder ao histórico de utentes ;
- Melhorias na usabilidade, como impedir a seleção simultânea dos locais ”Tomou na unidade”e ”Tomou em casa”;
- Verificação do comportamento correto do registo de tomas no calendário;
- Implementação de mensagens de erro mais compreensíveis para falhas na comunicação com o servidor;
- Registo completo das chamadas feitas à API para facilitar a rastreabilidade;
- Fluxo de login e persistência de sessão;
- Criação e visualização de testes e tratamentos;
- Registo de tomas no calendário, incluindo a marcação de desistências;
- Consulta do histórico de utentes;
- Emissão e validação de mensagens de aviso em casos definidos.

A validação foi feita com dados simulados, garantindo o correto funcionamento das funcionalidades principais, a clareza das interfaces e a coerência com os objetivos definidos. Esta validação prática, realizada com os profissionais da Associação, demonstrou que a solução desenvolvida está num bom caminho para supear as limitações actuais dos rastreio da hepatite C.

6 - Metódo e Planeamento

6.1 Planeamento inicial

O cronograma foi baseado nos requisitos não implementados ou que necessitam de melhorias do tfc anterior, divididos em tarefas para cada fase do desenvolvimento. Cada Sprint será dedicada a implementar requisitos não implementados e realizar melhorias, seguindo a metodologia agile de ciclos curtos com entregas contínuas e testes.
Segue-se apresentação do planeamento:

	Nome	Duração	Início	Fim	Antecessores	Nomes dos Recursos
1	Escolha do tema para o tfc	1 dia	09-09-2024 18:00	10-09-2024 17:00		
2	Atribuição do tema	17 dias	25-09-2024 8:00	17-10-2024 17:00		
3	Reunião com orientador para instalação de IDE	0,188 dias	18-10-2024 8:00	18-10-2024 9:30	2	
4	Reunião com o orientador para apresentação do tfc	1,125 dias	18-10-2024 9:30	21-10-2024 10:30	3	
5	Analise de requisitos	0,25 dias	21-10-2024 10:30	21-10-2024 13:30	4	
6	Analise da base de dados	0,375 dias	22-10-2024 18:00	23-10-2024 11:00		
7	Analise dos web services	0,125 dias	29-10-2024 16:00	29-10-2024 17:00		
8	Reunião com os profissionais da Ares do Pinhal	0,25 dias	05-11-2024 16:00	06-11-2024 9:00	7	
9	Pertinencia(relatorio)	1,25 dias	14-11-2024 15:00	15-11-2024 17:00	5	
10	Viabilidade(relatorio)	2 dias	18-11-2024 8:00	19-11-2024 17:00	9	
11	Especificação(relatorio)	4 dias	20-11-2024 18:00	26-11-2024 17:00	9;10	
12	Modelação(relatorio)	1 dia	27-11-2024 8:00	27-11-2024 17:00		
13	Solução proposta e proposta inicial(relatorio)	2 dias	28-11-2024 8:00	29-11-2024 17:00	9;10;11;12	
14	Overflow no separador do calendário de tomas	4 dias	02-02-2025 12:00	06-02-2025 17:00		
15	Problemas de UI nos detalhes do tratamento	8 dias	02-02-2025 8:00	12-02-2025 17:00		
16	Início do tratamento não aparece no histórico	6 dias	02-02-2025 15:00	10-02-2025 17:00		
17	Back no ecrã "Detalhes do tratamento"	10 dias	02-02-2025 15:00	14-02-2025 17:00		
18	Calendário de tomas - não se percebe se já fez a toma	7,5 dias	12-02-2025 12:00	21-02-2025 17:00		
19	Calendário de tomas - não mostra as tomas registadas na BD	5 dias	12-02-2025 11:00	19-02-2025 17:00		
20	Calendário de tomas - selecionar e cancelar, deixa o dia "marcado"	1 dia	13-02-2025 8:00	13-02-2025 17:00		
21	Calendário de tomas - janela devia mostrar o dia	3,5 dias	14-02-2025 13:00	19-02-2025 17:00		
22	Janela que mostra informação da toma devia mostrar o dia	5 dias	16-02-2025 13:00	21-02-2025 17:00		
23	Janela para editar toma deveria ser coerente com a janela para reg	5 dias	16-02-2025 13:00	21-02-2025 17:00		
24	Calendário de tomas deveria ser mostrado por omissão	5 dias	16-02-2025 13:00	21-02-2025 17:00		

	Nome	Duração	Início	Fim	Antecessores	Nomes dos Recursos
23	Janela para editar toma deveria ser coerente com a janela para reg	5 dias	16-02-2025 13:00	21-02-2025 17:00		
24	Calendário de tomas deveria ser mostrado por omissão	5 dias	16-02-2025 13:00	21-02-2025 17:00		
25	Avaliar a necessidade do "Testes a serem realizados"	5,5 dias	14-02-2025 13:00	21-02-2025 17:00		
26	Registar no log todos os pedidos à API	7,875 dias	18-02-2025 9:00	27-02-2025 17:00		
27	Histórico mostra erro 404	3 dias	10-03-2025 9:00	13-03-2025 9:00		
28	Mostrar erros "simpáticos" quando há erro de comunicação com o se	1 dia	17-03-2025 9:00	18-03-2025 9:00		
29	Quando se regista a toma, continua a aparecer nos "Tratamentos p	2,875 dias	19-03-2025 9:00	21-03-2025 17:00		
30	Impedir seleção simultânea nos locais "Tomou na unidade" e "Tomou	2,875 dias	19-03-2025 9:00	21-03-2025 17:00		
31	O botão para confirmar a toma esta sempre ativo	5 dias	12-03-2025 9:00	19-03-2025 9:00		
32	Reunião com os profissionais da Ares do Pinhal	0,75 dias	20-03-2025 10:00	20-03-2025 17:00		
33	Criar um écran "Sobre" (about)	2 dias	02-05-2025 8:00	05-05-2025 17:00		
34	Criar um ecrã novo "Utentes em tratamento	12 dias	15-05-2025 8:00	30-05-2025 17:00		
35	Log-in persistente	3 dias	03-05-2025 14:00	07-05-2025 17:00		
36	Swipe para cima no calendario muda o display	1 dia	05-04-2025 12:00	07-04-2025 17:00		
37	Warning para registos de toma que sejam antes do inicio do tratam	4 dias	07-05-2025 8:00	12-05-2025 17:00		
38	Historico de testes não aparece quando paciente não esteve em tr	4 dias	13-05-2025 8:00	16-05-2025 17:00		
39	Registo de tomas que um paciente possui em casa	1 dia	16-04-2025 8:00	16-04-2025 17:00		
40	Filtragem por utentes em tratamento, teste de rastreio, teste diagr	3 dias	18-05-2025 12:00	21-05-2025 17:00		
41	Warning quando a data desde fim de tratamento de um paciente ul	3 dias	20-05-2025 10:00	23-05-2025 10:00		
42	Warning após 4 dias seguidos sem toma de um utente	1 dia	06-05-2025 10:00	07-05-2025 10:00		
43	Teste RVS12 (pós tratamento)	2 dias	17-05-2025 12:00	20-05-2025 17:00		
44	Exportar tratamentos para formato CSV	7 dias	17-05-2025 14:00	27-05-2025 17:00		

Figura 6.1: Cronograma

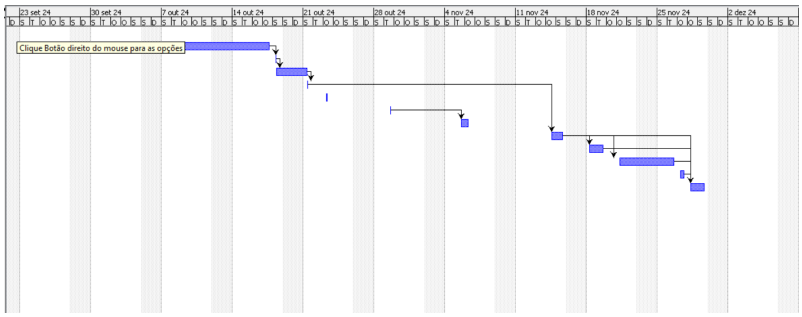


Figura 6.2: Gantt chart

6.2 Análise Crítica ao Planeamento

O progresso do projeto foi, de forma geral, satisfatório, tendo sido possível concretizar a maioria das tarefas previstas. Apesar de nem todas as funcionalidades inicialmente propostas terem sido implementadas, o desenvolvimento seguiu uma linha bem estruturada e coerente com os objetivos principais definidos.

Contudo, surgiram algumas limitações ao longo do processo, sendo a principal dificuldade relacionada com a comunicação com o responsável pela integração da aplicação com o sistema Zeus. Esta limitação condicionou o avanço de determinadas funcionalidades dependentes dessa integração, obrigando a ajustes no planeamento e nas prioridades estabelecidas.

Além disso, o tempo disponível e a complexidade técnica de certas funcionalidades também contribuíram para que algumas tarefas permanecessem por concluir, sendo agora apontadas como possíveis trabalhos futuros.

7 - Resultados

7.1 Resultados dos Testes

A validação da aplicação foi conduzida com foco na verificação da conformidade funcional, usabilidade e viabilidade operacional em contexto real. Para isso, recorreu-se à abordagem prática já detalhada no capítulo 5, nomeadamente através da criação e execução de issues correspondentes a cada requisito funcional implementado.

Os testes foram realizados em ambiente de simulação, utilizando a infraestrutura da universidade, e a aplicação foi instalada em dispositivos móveis Android e iOS. Todos os testes envolveram dados fictícios. No entanto, os profissionais da Associação Ares do Pinhal tiveram a oportunidade de testar a aplicação diretamente nos seus próprios telemóveis. Embora houvesse limitações devido aos dados reais não estarem disponíveis para todos os testes, os profissionais puderam sugerir melhorias importantes, que foram devidamente analisadas. No entanto, algumas dessas melhorias não foram implementadas a tempo, devido a limitações de tempo e de recursos.

Os resultados demonstraram que os fluxos principais da aplicação funcionam corretamente, nomeadamente:

- Autenticação de utilizadores e fluxo de login;
- Criação, visualização e edição de testes de rastreio e diagnóstico;
- Criação, visualização e edição de tratamentos;
- Registo de tomas diárias no calendário e gestão de desistências;
- Consulta do histórico clínico dos utentes;
- Consulta do histórico clínico dos utentes;
- Usabilidade adequada segundo feedback direto dos profissionais da AdP.

Apesar de alguns desafios propostos pela AdP não terem sido ainda implementados, a aplicação já representa uma melhoria significativa face aos processos anteriormente usados, e pode ser usada. Dos 31 testes representados por *issues*, 24 foram concluídos com sucesso (estado *Closed- implementada*), o que corresponde a uma taxa de realização de cerca de 77%. As restantes 7 *issues* (estado *Open- não implementada*) não foram concluídas devido a limitações técnicas e de tempo.

7.2 Cumprimento de requisitos

A Tabela 7.1 apresenta o estado das issues desenvolvidas durante o período de trabalho. Todas as funcionalidades inicialmente previstas no planeamento foram implementadas com sucesso, estando concluídas antes da fase de testes com os profissionais da AdP. As restantes issues listadas correspondem a melhorias sugeridas posteriormente, durante os testes realizados com a aplicação instalada nos dispositivos móveis dos profissionais. Estas sugestões refletiram necessidades reais identificadas em contexto de utilização.

Nº issue	Descrição	Estado	Origem da Issue
4	Overflow no separador do calendário de tomas	Implementada	Já existia
5	Problemas de UI nos detalhes do tratamento	Implementada	Já existia
6	Início do tratamento não aparece no histórico	Implementada	Já existia
7	Back no ecrã ”Detalhes do tratamento”	Implementada	Já existia
8	Calendário de tomas - não se percebe se já fez a toma	Implementada	Já existia
9	Calendário de tomas - não mostra as tomas registadas na BD	Implementada	Já existia
10	Calendário de tomas - selecionar e cancelar, deixa o dia ”marcado”	Implementada	Já existia
11	Calendário de tomas - janela devia mostrar o dia	Implementada	Já existia
12	Janela que mostra informação da toma devia mostrar o dia	Implementada	Já existia
13	Janela para editar toma deveria ser coerente com a janela para registar toma	Implementada	Já existia
14	Calendário de tomas deveria ser mostrado por omissão	Implementada	Já existia
15	Avaliar a necessidade do ”Testes a serem realizados”	Implementada	Já existia
17	Registar no log todos os pedidos à API	Implementada	Já existia
19	Histórico mostra erro 404	Implementada	Já existia
21	Passar searchPatients para o AppatiteRepository	Implementada	Já existia
24	Mostrar erros ”simpáticos”quando há erro de comunicação com o servidor	Implementada	Já existia
25	Quando se regista a toma, continua a aparecer nos ”Tratamentos para Hoje”	Implementada	Já existia
27	Criar um écran ”Sobre”(about)	Implementada	Surgiu posteriormente
29	Impedir seleção simultânea nos locais ”Tomou na unidade”e ”Tomou em casa”	Implementada	Surgiu posteriormente
30	O botão para confirmar a toma está sempre ativo	Implementada	Surgiu posteriormente
32	Log-in persistente	Implementada	Surgiu posteriormente
33	Swipe para cima no calendário muda o display	Implementada	Surgiu posteriormente
35	Histórico de testes não aparece quando paciente não esteve em tratamento	Implementada	Surgiu posteriormente
40	Teste RVS12 (pós tratamento)	Implementada	Surgiu posteriormente
34	Warning para registos de toma que sejam antes do início do tratamento	Não implementada	Surgiu posteriormente
36	Registo de tomas que um paciente possuí em casa	Não implementada	Surgiu posteriormente
37	Filtragem por utentes em tratamento, teste de rastreio, teste diagnóstico	Não implementada	Surgiu posteriormente
38	Warning quando a data desde fim de tratamento de um paciente ultrapassar os 90 dias	Não implementada	Surgiu posteriormente
39	Warning após 4 dias seguidos sem toma de um utente	Não implementada	Surgiu posteriormente
41	Exportar tratamentos para formato CSV	Não implementada	Surgiu posteriormente

Tabela 7.1: Estado de implementação das issues

8 - Conclusão

8.1 Conclusão

A realização deste TFC representou uma oportunidade valiosa para aplicar conhecimentos adquiridos ao longo da licenciatur e refletir sobre a aplicação real de software em contextos de saúde pública.

O grau de concretização do plano foi elevado. Grande parte das funcionalidades inicialmente previstas foi implementada com sucesso. Das 31 *issues* levantadas durante o processo, 24 foram resolvidas com sucesso, as restantes 7 não foram implementadas por limitações técnicas ou de tempo, mas estão documentadas para possível desenvolvimento futuro.

Houve diferenças entre a solução proposta inicialmente e a solução final. A proposta inicial previa um sistema mais limitado ao registo de tomas e acompanhamento de tratamento para hepatite C. Contudo, ao longo do projeto, a solução evoluiu e melhorias significativas na interface, com base no feedback dos testes realizados.

A evolução do trabalho e dos conhecimentos ao longo do TFC foi notória. Foi possível aprofundar competências em Flutter, testes automatizados com *widget tests*, práticas de design centrado no utilizador, gestão de dependências com `Provider`, e boas práticas de repositórios. O contacto com casos reais e os desafios de integração com sistemas existentes contribuíram também para uma visão mais realista do desenvolvimento de software.

Se o TFC voltasse ao início, dar-se-ia maior ênfase ao planeamento da arquitetura e aos testes desde as fases iniciais, de modo a facilitar a escalabilidade e a identificação precoce de bugs. Além disso, algumas funcionalidades consideradas ”menos prioritárias”no início revelaram-se essenciais para a experiência do utilizador e mereciam ter sido priorizadas.

As maiores dificuldades surgiram na integração com a API externa do sistema Zeus, com o qual a aplicação precisava de comunicar. Até à entrega do TFC, foi apenas possível simular essa integração em ambiente local. Curiosamente, apenas dois dias antes do prazo de entrega, recebemos o primeiro contacto direto do responsável por essa API, o que limitou o avanço neste aspeto crucial. A gestão de tempo, sobretudo nas fases finais, foi também um desafio, tendo em conta a sobreposição com avaliações, outros projetos académicos e o esforço de testes e documentação.

Importa referir que, ao longo do desenvolvimento, a aplicação foi instalada nos telemóveis dos profissionais da Ares do Pinhal, o que representa um marco importante de validação prática da solução construída.

8.1.1 Trabalhos Futuros

Num cenário de continuidade deste projeto, o primeiro passo será tentar integrar a comunicação com a API externa e de seguida o desenvolvimento das *issues* atualmente em aberto. Estas funcionalidades não implementadas abrangem avisos importantes para o acompanhamento terapêutico, exportação de dados em formato CSV e funcionalidades que reforçam a utilidade da aplicação em ambiente clínico.

Posteriormente, pretende-se expandir o âmbito da aplicação para além do rastreio e acompanhamento da hepatite C, incluindo também outras quatro doenças relevantes em contexto de saúde pública, tais como VIH. Esta extensão exigirá uma reformulação na estrutura dos dados, generalização de fluxos de tratamento e novas interfaces de visualização, mas aumentará significativamente o impacto e utilidade da solução.

Em resumo, este TFC resultou numa aplicação funcional, com verdadeiro potencial de impacto na saúde pública. Embora já apresente um contributo relevante, ainda há espaço para

crescimento e inovação. A base está lançada — agora, é tempo de continuar a construir sobre ela.

Bibliografia

[Mat24] João P. Matos-Carvalho. *The Lusófona LaTeX Template User's Manual*. Lusófona University. 2024. URL: https://github.com/jpmcarvalho/UL-Thesis.

[Flu23] Flutter.dev. *Documentação do Flutter*. Acessado em novembro de 2023. 2023. URL: https://docs.flutter.dev/.

[24] *JSON: JavaScript Object Notation*. Acessado em janeiro de 2024. 2024. URL: https://www.json.org/json-en.html.

[Dar24] Dart.dev. *Visão geral da linguagem de programação Dart*. Acessado em janeiro de 2024. 2024. URL: https://dart.dev/overview.

[Ven14] C. L. Ventola. *Dispositivos móveis e aplicações para profissionais de saúde: usos e benefícios*. 5. 2014, pp. 356–364. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4029126/.

[Hoc+19] K. Hochstatter et al. *Uma intervenção móvel de saúde para melhorar os resultados da hepatite C entre pessoas com transtorno por uso de opioides: protocolo para um ensaio clínico randomizado*. 8. 2019, e12620. DOI: 10.2196/12620. URL: https://www.researchprotocols.org/2019/8/e12620.

[Flu23; 24; Dar24; Ven14; Hoc+19].

Anexos

Webservices da aplicação

Webservices Rastreio Hepatite C

Documento técnico

Lista de webservices que serão utilizados pela aplicação móvel de rastreio da hepatite C, desenvolvida pela Universidade Lusófona para a associação Ares do Pinhal

Índice

Definição de entidades e enumerados utilizados

Patient

- zeusID (int) - ID do utente (zeus)
- name (String) - Nome do utente
- birthDate (LocalDate) - Data de nascimento do utente
- realID (String) - Parece-nos ser o ‘ID’ do documento de identificação
- documentType (String) - Tipo de documento (ex. CC)
- lastProgramName (String) - Nome do último programa
- lastProgramDate (LocalDate) - Data do último programa
- currentTreatmentID (int) - ID do tratamento atual do utente
- lastScreeningID (int) - Referenciação ao ID do último teste realizado pelo utente
- state (Enum - PatientStatus) - Estado do utente
- gender (String) - Genero do utente (‘male’ ou ‘female’)
- statusDate (LocalDate) - Data em que o estado do utente foi atualizado pela última vez

Enum PatientStatus

- NED (No evidence of disease)
- POSITIVE_SCREENING
- POSITIVE_DIAGNOSIS
- TREATMENT
- POST_TREATMENT_ANALYSIS
- FINISHED
- NOT_IN_DATABASE

Lógica do PatientStatus

Todos os utentes começam no estado NED (estado = NED).

- Se (estado == NED && resultadoTesteRastreio == negativo) então estado = NED

- Se (estado == NED && resultadoTesteRastreio == positivo) então estado = POSITIVE_SCREENING

- Se (estado == POSITIVE_SCREENING && resultadoTesteDiagnostico == negativo) então estado = POSITIVE_SCREENING

- Se (estado == POSITIVE_SCREENING && resultadoTesteDiagnostico == positivo) então estado = POSITIVE_DIAGNOSIS

- Quando é iniciado um tratamento, estado passa para TREATMENT (estado = TREATMENT)

- Quando é terminado um tratamento, estado passa para POST_TREATMENT_ANALYSIS (estado = POST_TREATMENT_ANALYSIS)

- Se (estado == POST_TREATMENT_ANALYSIS && resultadoTeste == true) então estado = TREATMENT

- Se (estado == POST_TREATMENT_ANALYSIS && resultadoTeste == false) então estado = POSITIVE_SCREENING

DailyMedicine

- id (int)
- date (LocalDate) - Data onde foi feito o registo desta toma
- tookMedicine (Boolean) - Marcar se o utente tomou ou não a medicação
- takeAtHome (Boolean) - Marcar se o utente fez ou irá fazer a toma em casa
- notes (String) - Notas adicionais
- treatmentID (int) - ID do tratamento ao qual está associada esta toma
- patientID (int) - ID do utente ao qual pertence esta toma

Test

- id (int)
- type (int) - 0 se for de rastreio, 1 se for de diagnóstico
- test_date (LocalDate) - data no qual foi realizado o teste
- result_date (LocalDate) - data no qual foi disponibilizado o resultado do teste
- result (Boolean) - True se foi Reativo, False se for Não reativo

- testLocation (int) - Local onde foi feito o teste, 1 se foi feito no SAI, 2 se foi feito no Hospital, 3 se foi feito na Unidade Móvel, 4 se foi feito no CAEM
- patientID (int) - ID do utente ao qual foi feito o teste
- staffID (int) - ID do Staff que realizou/registou o teste

Treatment

- id (int)
- startDate (LocalDate) - Data de início do tratamento
- realEndDate (LocalDate) - Data de fim do tratamento
- postTreatmentStartDate (LocalDate) - Data de início do pós-tratamento
- nameMedication (int) - Nome da medicação, 1 se for Maviret, 2 se for Epclusa
- reasonsDropout (int) - Motivos de desistência (1 – Morte, 2 – Desistiu, 3 – Mudou de residência)
- endTreatmentComment (String) - Notas/Comentários ao fim do tratamento
- treatmentDuration (int) - Duração do tratamento em número de semanas
- patientID (int) - ID do utente

Descrição dos webservices

Regras gerais

- Todos os webservices levam um cabeçalho HTTP "Basic Auth" com as credenciais do utilizador:

  Authorization: Basic basic

  password

- Nos webservices por GET, o input vai diretamente no URL, sob a forma de parâmetros
- Nos webservices por POST, o input vai sob a forma de JSON
- A resposta vem sempre em JSON, seja por GET ou POST

- O código HTTP de resposta deve ser 200 se tiver sucesso. Se as credenciais forem inválidas, deve responder 401. Se ocorrer um erro deve responder 500, e o output deve ser um JSON com um único atributo "errorMessage".

1. (GET) validateCredentials (/api/validateCredentials)

- **Input**: Nenhum (as credenciais são enviadas no cabeçalho HTTP Basic Auth).
- **Output**:
  - HTTP 200: Credenciais válidas.
  - HTTP 401: Credenciais inválidas.

2. (GET) getPatientsInTreatment (/api/patient/activeTreatment)

- **Descrição:** Retorna uma lista de todos os utentes que estão em tratamento de hepatite C atualmente.
- **Input**: Nenhum.
- **Output**: Lista de pacientes em tratamento.
- **Exemplo de resposta**:


```
[
  {
  "idZeus": 123,
  "name": "Patient One",
  "birthDate": "1990-01-01",
  "realId": "123456789",
  "documentType": "cc",
  "lastProgramName": "Program XYZ",
  "lastProgramDate": "2024-09-24",
  "currentTreatmentId": 0,
  "lastScreeningID": 1,
  "state": "TREATMENT",
 "gender": "male",
```

```
    "statusDate": "2024-09-27"
  }
]
```

3. (POST) insertDailyMedicine (/api/dailyMedicine/new)

- **Descrição:** Recebe um pedido para criar ou atualizar uma toma do medicamento usado no tratamento da hepatite C de um determinado utente, verifica se já existe registo de dailyMedicine no dia atual. Caso não exista, cria um novo, caso exista, atualiza o atual.
- **Input (exemplo)**:

```
{
  "patientId": 3,
  "date": "2024-10-07",
  "notes": "Exemplo de notas adicionais",
  "tookMedicine": true,
  "takeAtHome": false
}
```

- **Output**:
  - Confirmação da criação/atualização (Código HTTP 200).
  - Mensagem no body se não existia: “Daily medicine created”
  - Mensagem no body se já existia: “Daily medicine updated”

4. (GET) getPatients (/api/patient/search/{input})

- **Input**:
  - **input (String)** – ID (zeus) ou nome do utente
- **Output**: Lista de pacientes correspondentes.
  - Exemplo de resposta:

```
[
    {
       "idZeus": 123,
       "name": "Patient One",
       "birthDate": "1990-01-01",
       "realId": "123456789",
       "documentType": "cc",
       "lastProgramName": "Program XYZ",
```

```
    "lastProgramDate": "2024-09-24",
    "currentTreatmentId": 0,
    "lastScreeningID": 1,
    "state": "NED",
    "gender": “male”,
    "statusDate": "2024-09-27"
  }
]
```

5. (POST) insertNewTest (/api/tests/new)

- **Descrição**: Cria um novo registo de um teste para um utente (pesquisado por zeusID através do repositório de pacientes). Caso o utente não exista, é criado um novo registo de utente também (com informação que vem no request). O estado do utente é alterado com base no tipo e resultado do teste.
- **Input**:

```
        {
  "type": 1,
  "result": false,
  "resultDate": "2024-04-01",
  "testLocation": 1,
  "testDate": "2024-04-01",
  "patient": {
    "idZeus": 234,
    "name": "Maria",
    "birthDate": "1980-04-01",
    "gender": "female"
  }
}
```

- **Output**:
  - Confirmação da criação (Código HTTP 200)
  - Mensagem no body: “ok”

6. (GET) getLastScreening (/api/patient/lastScreening/{zeusId})

- **Descrição**: Retorna o último teste realizado pelo utente com o zeusID passado por parametro.
- **Input:**
  - **zeusId (int)** - ID do utente (zeus).
- **Output**: Último teste realizado pelo utente.
  - Exemplo de resposta:


```
{
  "id": 10,
  "type": 0,
  "testDate": "2024-11-05",
  "resultDate": null,
  "result": true,
  "testLocation": 1,
  "patientId": 3,
  "staffId": 1
}
```


7. (GET) getTestsByPatient (/api/tests/patient/{zeusId})

- **Descrição**: Retorna todos os testes de um utente.
- **Input**:
  - **zeusId (int)** - ID do utente (zeus).
- **Output**: Lista de testes realizados.
  - Exemplo de resposta:


```
[
  {
    "id": 11,
    "type": 1,
    "testDate": "2024-11-06",
    "resultDate": null,
```

```
    "result": true,
    "testLocation": 3,
    "patientId": 4,
    "staffId": 1
  }
]
```

8. (POST) updateTest (/api/tests/update)

- **Descrição**: Atualiza informações de um teste existente.
- **Input**:

```
{
  "testId": 1,
  "type": 1,
  "testDate": "2024-11-22",
  "resultDate": "2024-11-23",
  "result": true,
  "testLocation": "Hospital ABC",
  "patientId": 123
}
```

- **Output**:
  - Confirmação da atualização (Código HTTP 200)
  - Mensagem no body: "Test updated successfully"

9. (POST) createTreatment (/api/treatment/new)

- **Descrição**: Cria um novo registo de tratamento para um paciente, alterando também o estado do paciente para "em tratamento".
- **Input**:

```
{
  "patientId": 3,
```

```
  "startDate": "2024-11-22",
  "nameMedication": "Epclusa",
  "treatmentDuration": 12
}
```


- **Output**:
  - Confirmação da criação (Código HTTP 200)
  - Mensagem no body: "Treatment created successfully"

10. (GET) getCurrentTreatment (/api/treatment/current/{zeusId})

- **Descrição**: Retorna o tratamento atual de um utente (incluindo todas as tomas referentes a esse tratamento), caso o utente não se encontre em tratamento não retorna informação.
- **Input:**
  - zeusId (int) - ID do utente (zeus).
- **Output**: Detalhes do tratamento atual e a lista de tomas até agora
  - Exemplo de resposta:


```
  {
  "id": 6,
  "startDate": "2024-11-05",
  "realEndDate": null,
  "postTreatmentStartDate": null,
  "nameMedication": 1,
  "reasonsDropout": 0,
  "endTreatmentComment": null,
  "treatmentDuration": 12,
  "patientId": 3
 "dailyMedicines": [
  {
     "date": "2024-10-07",
     "tookMedicine": true,
     "takeAtHome": false,
```

```
      "notes": ""
    },
   {
      "date": "2024-10-08",
      "tookMedicine": true,
      "takeAtHome": false,
      "notes": ""
    }

 ]
}
```

11. (POST) endTreatment (/api/treatment/endTreatment)

- **Descrição**: Termina o tratamento atual de um utente, guarda notas de fim de tratamento, e caso seja uma desistência, guarda também o motivo de desistência, e altera o estado do utente para "pós-tratamento".
- **Input**:

```
{
  "idZeus": 3,
  "desistencia": true,
  "dropoutReason": 2,
  "notes": "Paciente optou por interromper o tratamento."
}
```

- **Output**:
  - Confirmação do encerramento (Código HTTP 200).
  - Mensagem no body: “ok”

12. (GET) getTreatmentHistory (/api/treatment/history/{zeusId})

- **Descrição**: Retorna uma lista de tratamentos de um utente.
- **Input**:
  - zeusId (int) - ID do utente (zeus).
- **Output**: Histórico de tratamentos do paciente.
  - Exemplo de resposta:


```
[
  {
    "id": 1,
    "startDate": "2024-10-01",
    "realEndDate": null,
    "postTreatmentStartDate": null,
    "nameMedication": 1,
    "reasonsDropout": 0,
    "endTreatmentComment": null,
    "treatmentDuration": 12,
    "patientId": 3
  },
  {
    "id": 2,
    "startDate": "2024-10-09",
    "realEndDate": null,
    "postTreatmentStartDate": "2024-10-05",
    "nameMedication": 1,
    "reasonsDropout": null,
    "endTreatmentComment": "",
    "treatmentDuration": 8,
    "patientId": 3
  }
]
```

13. (GET) getDailyMedicineHistory (/api/dailyMedicine/history{zeusId})

- **Descrição**: Retorna o histórico completo de medicação diária associada ao utente com o ID Zeus fornecido como parâmetro.
- **Input**:
  - zeusId (int) - ID do utente (zeus).
- **Output**:Lista de objetos DailyMedicine com todos os registos de administração de medicação diária do utente.
  - Exemplo de resposta:


```
 [
{
   "id": 12,
  "date": "2024-11-05",
  "given": true,
  "staffId": 5,
  "patientId": 3

},
{
   "id": 14,
  "date": "2024-11-06",
  "given": false,
  "staffId": 7,
  "patientId": 3

}
]
```

14. (GET) getPatientsWithoutDailyMedicineToday (/api/dailyMedicine/patientsWithoutDailyMedicineToday)

- **Descrição**:Retorna uma lista de utentes que se encontram em tratamento mas que ainda não receberam medicação diária na data atual.
- **Input**: Nenhum.
- **Output**:Lista de objetos Patient representando os utentes que ainda não tomaram a medicação no dia atual.
    - Exemplo de resposta:

```
 [
  {
    "idZeus": 3,
    "name": "Maria João",
    "age": 28,
    "state": "TREATMENT",
    "admissionDate": "2024-10-12"
  },
 {
    "idZeus": 7,
    "name": "Carlos Manuel",
    "age": 34,
    "state": "TREATMENT",
    "admissionDate": "2024-10-18"
  }

]
```