University of Minho
School of Engineering

Desenvolvimento de uma Plataforma Digital para Monitorização Clínica de Diabetes

Processo Clínico Eletrónico

Daniel Sá (PG56120)

João Cardoso (PG56135)

Jorge Costa (PG56136)

june 2025

Índice

1 Contextualização

Nas últimas décadas, a diabetes mellitus tem registado um crescimento alarmante a nível mundial. De acordo com estudos recentes, o número de casos quadruplicou desde 1990, atingindo atualmente cerca de 800 milhões de pessoas afetadas. Estima-se que, até 2050, este número ultrapasse os 1310 milhões. Apesar dos esforços de sensibilização e das campanhas de prevenção, fatores como a má alimentação, o sedentarismo e o excesso de peso continuam a ser determinantes no aumento da prevalência de diabetes tipo 2.

Neste cenário, a utilização de tecnologias digitais na saúde assume um papel fundamental na monitorização contínua e na gestão personalizada da condição clínica dos doentes. Aplicações móveis que permitam o registo, análise e partilha de dados em tempo real contribuem não só para a autonomia do utente como também para a obtenção de dados estruturados que apoiam decisões clínicas mais informadas.

Paralelamente, a interoperabilidade entre sistemas de informação de saúde revela-se essencial para garantir a comunicação eficaz entre diferentes plataformas, profissionais e dispositivos. Deste modo, é indispensável que os dados recolhidos estejam estruturados segundo normas reconhecidas como o FHIR (*Fast Healthcare Interoperability Resources*), facilitando assim a integração e reutilização em diferentes contextos clínicos.

Neste âmbito, no contexto da unidade curricular Processo Clínico Eletrónico, foi desenvolvida uma aplicação destinada ao controlo de diabetes, de utilização diária por parte de utentes. Esta aplicação tem como principais funcionalidades o registo de medições de glucose e administrações de insulina, a apresentação de estatísticas, o acesso ao histórico dos registos, um sistema de notificações, uma agenda para marcações de medições e administrações e um chatbot interativo para apoio informativo ao utilizador. Todos os registos são compatíveis com padrões de interoperabilidade, permitindo a sua partilha com profissionais de saúde de forma segura e estruturada.

2 Modelação da Informação Clínica com openEHR

Um princípio fundamental do projeto é a garantia de que os dados clínicos sejam capturados, armazenados e partilhados de uma forma semanticamente interoperável. Para tal, adotou-se a norma openEHR, uma especificação formal para a criação de sistemas de registos de saúde eletrónicos (EHR) robustos e preparados para o futuro.

A escolha do openEHR foi estratégica por várias razões:

- **Interoperabilidade semântica:** Ao contrário das normas que apenas definem a estrutura da troca de dados, o openEHR centra-se na definição de conceitos clínicos de uma forma computável e inequívoca.
- **Dados à prova de futuro:** Ao separar os modelos clínicos (Arquétipos e Templates) do esquema físico da base de dados e do código da aplicação, o sistema torna-se altamente adaptável.
- **Conceção orientada pelo médico:** A metodologia openEHR promove a criação de modelos de dados por especialistas no domínio (clínicos, engenheiros biomédicos) e não apenas por programadores de software. Isto assegura que os modelos são clinicamente sólidos e relevantes.

Para implementar a metodologia openEHR, recorreu-se à ferramenta Archetype Designer, um ambiente visual que facilitou a criação e reutilização de arquétipos – blocos de construção essenciais que representam conceitos clínicos distintos adaptados às necessidades do nosso projeto. Além disso, utilizou-se a ferramenta para criar modelos, que são composições estruturadas de um ou mais arquétipos. Estes modelos são concebidos para definir formulários ou documentos de captura de dados específicos, traduzindo efetivamente os arquétipos abstratos em aplicações práticas para casos de utilização específicos.

No Archetype Designer, foram concebidos três modelos distintos para servir de base aos formulários de introdução de dados da aplicação desenvolvida. Cada modelo é construído sobre um arquétipo de base `openEHR-EHR-COMPOSITION.encounter.v1`, assegurando uma estrutura de composição consistente.

- **Informação do utilizador (Template de Registo do indivíduo):** Este modelo capta o perfil demográfico e administrativo principal do utilizador. É construído em torno do arquétipo `openEHR-EHR-ADMIN_ENTRY.demographics.v0`. Esta entrada administrativa contém um cluster principal `openEHR-EHR-CLUSTER.person.v1`, que agrega vários outros clusters para definir um perfil abrangente do utente:
  - Identificação: Nome, Identificador, e Data de Nascimento.

- Contacto: openEHR-EHR-CLUSTER.address.v1 para endereço físico e openEHR-EHR-CLUSTER.electronic_communication.v1 para telefone e correio eletrónico.
- Atributos pessoais: openEHR-EHR-CLUSTER.genero.v0 para o género.
- Métricas corporais: Um openEHR-EHR-CLUSTER.metricas.v0 personalizado para registar a altura e o peso de base do utilizador.

- **Medição da glucose (Template do Formulário de Glucose):** Este modelo foi concebido para registar as leituras de glicemia e o seu contexto. Utiliza como núcleo o arquétipo openEHR-EHR-OBSERVATION.laboratory_test_result.v1. Os principais dados registados são:
  - Resultado Laboratorial: Incluido num openEHR-EHR-CLUSTER.laboratory_test_analyte.v1, este é o campo DV_QUANTITY primário para o valor da glucose, com unidades limitadas a mg/dL.
  - Contexto da refeição: Um openEHR-EHR-CLUSTER.diet.v0 permite ao utilizador especificar o momento do teste relativamente às refeições, com opções de texto codificadas como Jejum, Pós-prandial e Pré-prandial.
  - Informação contextual: Os clusters personalizados openEHR-EHR-CLUSTER.atividade_fisica.v0 e openEHR-EHR-CLUSTER.peso.v0 estão incluídos para permitir que o utilizador registe opcionalmente a sua atividade física e o seu peso atual no momento da medição. Este valor de peso é utilizado pelo trigger da base de dados para manter atualizado o perfil principal do utilizador.

- **Administração de insulina (Template do Formulário de Insulina):** Este modelo regista a administração de uma dose de insulina. Tal como o formulário de glucose, também se baseia no arquétipo openEHR-EHR-OBSERVATION.laboratory_test_result.v1. Capta duas informações críticas:
  - Dosagem: O campo Resultado da análise DV_QUANTITY é utilizado para registar o número de unidades de insulina administradas, com a unidade definida para [arb'U] (unidades arbitrárias).
  - Via de administração: Um openEHR-EHR-CLUSTER.rota.v0 personalizado fornece opções de texto codificado para a via de administração, como Subcutânea ou Intravenosa.

Os formulários de entrada de dados no frontend React são gerados dinamicamente com base nesses modelos openEHR. Quando um utilizador submete dados, estes são estruturados como um objeto JSON que está em conformidade com o modelo do template. Este JSON é então enviado para o backend Node.js e armazenado na coluna dados (JSONB) da tabela de registos, preservando a informação clínica rica e estruturada. Esta estratégia não só garante a qualidade dos dados, como também simplifica o mapeamento subsequente para recursos FHIR para comunicação externa.

3 Arquitetura

O projeto foi desenvolvido segundo o paradigma cliente-servidor, adotando uma arquitetura frontend-backend, em conformidade com o modelo mais comum nas aplicações web contemporâneas. Esta abordagem permite uma clara separação entre a camada de dados, a lógica da aplicação e a interface com o utilizador, promovendo uma divisão eficaz de responsabilidades, aumentando a segurança e contribuindo para uma estrutura mais organizada e previsível.

A stack tecnológica utilizada assenta em PostgreSQL para a gestão da camada de dados e num backend desenvolvido em Node.js, que expõe rotas através da framework Express.js. A comunicação entre o backend e a base de dados é realizada através de uma pool de ligações, assegurando eficiência e estabilidade. No frontend, recorre-se à framework React.js para apresentar a informação de forma clara e visualmente apelativa ao utilizador, estabelecendo comunicação com o backend através de um mecanismo de autenticação baseado em tokens, garantindo a segurança no acesso aos dados.

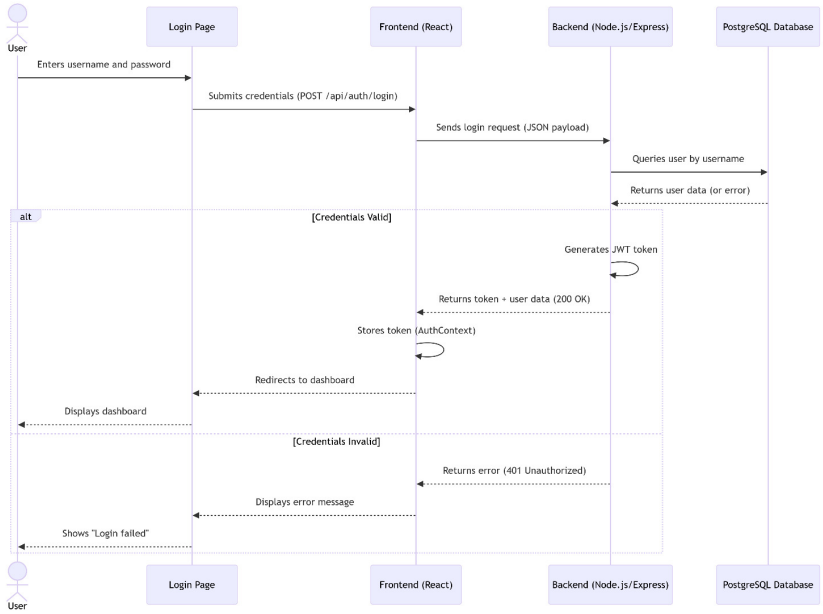

Figura 3.1: Diagrama de Sequência

4 Sistema de Base de Dados

O sistema de base de dados foi projetado para refletir as entidades principais da aplicação, permitindo a gestão de utilizadores, contactos, moradas, registos clínicos e eventos agendados. A coerência dos dados é reforçada através da utilização de tipos de dados específicos, tipos enumerados e triggers de validação.

4.1 Tabelas

A base de dados é composta por cinco tabelas principais:

- **Utilizador:** Tabela principal que armazena os dados dos utentes da aplicação.
  - `id` (integer): Identificador único do utilizador (Chave Primária).
  - `nome` (text): Nome completo do utente.
  - `data_nasc` (date): Data de nascimento do utente.
  - `altura` (numeric(5,2)): Altura do indivíduo em centímetros.
  - `peso` (numeric(5,2)): Peso do utilizador em quilogramas.
  - `genero` (sexo_tipo): Tipo enumerado que representa o género (`M`, `F` ou `O`).
  - `username` (text): Nome de utilizador para realizar o login na aplicação.
  - `password_hash` (text): Palavra-passe encriptada para autenticação segura.
- **Morada:** Permite associar uma ou mais moradas a um utilizador.
  - `id` (text): Identificação única da morada (Chave Primária).
  - `utilizador` (integer): Chave estrangeira para a tabela `Utilizador`, referenciando o `id` do utilizador.
  - `endereco` (text): Endereço da morada do utente.
  - `cidade` (text): Cidade da morada.
  - `distrito` (text): Distrito correspondente.
  - `pais` (text): País de residência.
  - `cod_postal` (text): Código-postal do utilizador.

- **Contacto:** Armazena os meios de contacto (telefone ou email). Cada registo corresponde a um único tipo de contacto por utilizador.
  - `utilizador` (integer): Chave primária e estrangeira para a tabela `Utilizador`.
  - `tipo_contacto` (contacto_tipo): Tipo enumerado que define o contacto como Telefone (`T`) ou Email (`E`).
  - `contacto` (text): O valor do contacto (número de telefone ou endereço de email) (Chave Primária).
- **Agenda:** Permite o registo de eventos agendados pelos utilizadores.
  - `id` (serial): Identificador sequencial da marcação (Chave Primária).
  - `utilizador` (integer): Identificador do utilizador (Chave Estrangeira para `Utilizador`).
  - `tipo_registo` (registo_tipo): Natureza do evento agendado: `G` ou `I`.
  - `data_evento` (timestamp without time zone): Data e hora do evento agendado.
  - `notas` (text): Campo opcional para observações adicionais.
  - `realizado` (boolean): Valor booleano que indica se o evento foi concretizado.
- **Registos:** Regista as entradas efetuadas pelos utilizadores, nomeadamente medições de glucose ou administrações de insulina.
  - `id` (text): Identificador único do registo (Chave Primária).
  - `utilizador` (integer): Chave estrangeira que associa o registo ao utilizador correspondente.
  - `data_registo` (timestamp without time zone): Data e hora da entrada do registo.
  - `tipo_registo` (registo_tipo): Tipo enumerado que define a natureza do registo (`G` ou `I`).
  - `dados` (jsonb): Dados brutos do formulário clínico em formato openEHR, armazenados como JSONB.

4.2 Tipos Enumerados

Foram definidos três tipos enumerados para garantir a consistência semântica nos dados introduzidos:

- **sexo_tipo:** Define o género do utilizador. Valores permitidos: {'M', 'F', 'O'}.
- **contacto_tipo:** Define o tipo de contacto. Valores permitidos: {'T', 'E'} (Telefone ou Email).
- **registo_tipo:** Define a natureza do registo clínico. Valores permitidos: {'I', 'G'}.

4.3 Triggers

Para reforçar a integridade lógica da base de dados, foram implementados três triggers com as respetivas funções de suporte:

- **check_user_id_exists():** Previne a duplicação de IDs de utilizador no momento da inserção, lançando uma exceção caso o `id` já se encontre registado.
- **update_user_weight_from_registo():** Após a inserção de um novo registo do tipo "Glucose" ou de uma atualização no perfil, este trigger atualiza automaticamente o campo `peso` na tabela `Utilizador`, caso se verifique uma alteração válida.
- **validate_contacto():** Valida automaticamente o formato dos contactos introduzidos. Para emails, verifica a sua conformidade com o formato padrão. Para telefones, valida o número de dígitos e a sua composição exclusivamente numérica.

4.4 Relações e Integridade Referencial

Todas as tabelas com associação à entidade utilizador seguem uma relação do tipo 1:N, onde um utilizador pode ter múltiplos contactos, moradas, registos e eventos agendados, mas cada registo pertence apenas a um único utilizador. As restrições de integridade referencial são aplicadas com cláusulas `ON DELETE CASCADE`, garantindo que, ao eliminar um utilizador, os seus dados associados são igualmente removidos.

5 Lógica da Aplicação e Estrutura do Backend

A lógica central da aplicação, responsável pela recuperação e manipulação de dados, autenticação de pedidos e garantia de interoperabilidade, é orquestrada através de um backend desenvolvido com recurso ao Node.js e à framework Express.js. Esta base tecnológica permitiu estruturar os diferentes componentes funcionais da aplicação, promovendo uma arquitetura modular e escalável.

No contexto da interoperabilidade, foi utilizado o Mirth Connect como sistema dummy, simulando uma entidade externa capaz de receber dados estruturados em conformidade with o protocolo FHIR. Este mecanismo permite testar e validar a capacidade da aplicação para comunicar com outros sistemas de informação em saúde.

5.1 Definição de Rotas

A definição de rotas no backend foi implementada com Express.js, permitindo responder, de forma segura e autenticada, às solicitações provenientes do frontend. Estas rotas foram organizadas em ficheiros distintos, cada um responsável por um subconjunto de funcionalidades:

- **agenda.js:** Contém rotas dedicadas à gestão da agenda do utilizador, incluindo marcação e eliminação de medições de glucose e administrações de insulina, bem como a possibilidade de marcar esses eventos como realizados ou não.
- **fhir.js:** Responsável pela interoperabilidade, este módulo permite ao frontend requisitar a exportação de registos para entidades externas, com base no protocolo FHIR. As rotas permitem testar a comunicação com o Mirth Connect, enviar registos individualmente ou em lote (bulk), garantindo a conformidade com os standards definidos para a estruturação dos dados em JSON.
- **auth.js:** Implementa as rotas de autenticação, nomeadamente o registo e login de utilizadores. O acesso às restantes rotas está condicionado à autenticação prévia do utilizador.
- **perfil.js:** Permite a visualização e edição dos dados de perfil do utilizador, disponibilizando rotas específicas para consulta e atualização dessa informação.
- **registos.js:** Agrega as rotas responsáveis pela submissão dos formulários baseados nos templates do openEHR, constituindo um dos componentes centrais da aplicação.

5.2 Acesso à Base de Dados

Para suportar a interação com a base de dados PostgreSQL, foram desenvolvidos módulos específicos responsáveis pela leitura e escrita de dados. Estes encontram-se organizados em ficheiros independentes da diretoria routes ou services, refletindo a sua função técnica distinta. As funcionalidades principais incluem:

- Armazenamento e recuperação de registos no formato JSON;
- Leitura, escrita e atualização de dados de utilizador;
- Gestão dos dados associados à agenda.

5.3 Assistente Virtual

A aplicação integra um assistente virtual inteligente que proporciona suporte personalizado aos utilizadores no contexto da gestão da diabetes. Este componente foi implementado através de uma arquitetura que combina o backend da aplicação com um modelo de linguagem local, garantindo privacidade e controlo total sobre os dados dos utilizadores.

O assistente virtual é suportado por uma rota específica no backend (**bot.js**) que atua como intermediário entre o frontend e o modelo de linguagem. Esta abordagem garante que:

- Todas as comunicações com o assistente são autenticadas através do middleware de autenticação existente;
- O acesso ao modelo de linguagem é controlado e monitorizado pelo backend;
- A integração é transparente para o utilizador final.

A funcionalidade do chatbot é suportada pelo Ollama, uma plataforma que permite executar modelos de linguagem localmente. Foi configurado um modelo personalizado denominado "DiabetesAssistant", especializado em fornecer informações e orientações relacionadas com a gestão da diabetes.

A comunicação com o Ollama processa-se por chamadas HTTP para a API local (`api/generate`), utilizando os seguintes parâmetros:

- **model:** Especifica o modelo "DiabetesAssistant"treinado para o domínio da diabetes;

- **prompt:** Contém a pergunta ou solicitação do utilizador;
- **stream:** Configurado como falso para receber respostas completas.

O assistente virtual está preparado para responder a uma variedade de questões relacionadas com a interpretação de valores de glicose e recomendações gerais, esclarecimentos sobre tipos de insulina e métodos de administração, orientações sobre alimentação e exercício físico para diabéticos, bem como explicações sobre o funcionamento da própria aplicação.

5.4 Serviços Auxiliares

Para além das rotas e do acesso à base de dados, o backend inclui diversos serviços auxiliares que asseguram a preparação, transformação e validação dos dados, essenciais para garantir a interoperabilidade e o correto funcionamento da aplicação:

- **Conversão FHIR:** Através da classe FHIRConverter, este serviço transforma os dados provenientes dos formulários openEHR em estruturas compatíveis com o protocolo FHIR. As medições de glucose são mapeadas como recursos Observation, enquanto as administrações de insulina são representadas como MedicationAdministration, de acordo com a semântica definida pelo FHIR.
- **Conexão ao Mirth Connect:** Serviço responsável pela comunicação com o sistema externo simulado (Mirth Connect), que recebe os dados FHIR através de um canal com HTTP Listener configurado para aceitar conteúdo JSON. Este módulo permite tanto o envio de dados como a realização de testes de conectividade com o endpoint externo.
- **Limpeza e Filtragem de Dados:** Serviço que extrai apenas a informação relevante dos dados submetidos pelos formulários, nomeadamente entradas textuais, datas e outros campos pertinentes. Este processo é utilizado pelo FHIRConverter para garantir que apenas os dados necessários são estruturados em conformidade com o FHIR, e também para preparar os dados a serem enviados ao frontend de forma eficiente.

6 Interface do Utilizador e Lógica do Frontend

6.1 Dashboard

A página inicial da aplicação é representada por uma dashboard interativa que oferece ao utilizador uma visão geral e imediata dos dados mais relevantes relacionados com a sua condição de saúde. Esta secção foi concebida para reforçar o acompanhamento diário e a autoconsciência do estado clínico, promovendo a literacia em saúde e a gestão ativa da diabetes.

A interface da dashboard foi organizada em seis blocos informativos, com o objetivo de apresentar, de forma concisa e visualmente clara, os seguintes elementos:

- **Última Medição de Glucose:** Esta secção apresenta a medição de glicose mais recente, detalhando o valor (em mg/dL), o regime alimentar (por exemplo, em jejum ou pós-prandial), o peso corporal do utilizador no momento da medição, e a respetiva data e hora.

  Além disso, com base em limiares clínicos definidos, é calculado um estado da glicose, apresentado sob a forma de badge com indicação visual (cor e mensagem). Este mecanismo permite ao utilizador perceber se o valor está dentro dos parâmetros considerados normais, elevados ou perigosos, com mensagens de alerta e, quando necessário, sugestão de contacto médico imediato.

- **Última Administração de Insulina:** Este bloco mostra os dados da última administração de insulina incluindo a quantidade injetada (U).

- **Próxima Medição de Glucose Agendada:** Apresenta o evento mais próximo do tipo 'Glucose' da tabela de agenda, indicando a data e hora agendada, o tempo restante até ao evento, e quaisquer notas associadas.

- **Próxima Administração de Insulina Agendada:** Tal como no ponto anterior, mostra o próximo evento do tipo 'Insulina', com os mesmos parâmetros.

- **Estatísticas de Glucose (Últimos 7 dias):** Os seguintes indicadores são calculados e apresentados com base nas últimas medições: valor médio, valor mínimo, valor máximo e o número de medições analisadas.

- **Estatísticas de Insulina (Últimos 7 dias):** Todas as administrações de insulina dos últimos sete dias são analisadas, apresentando a média diária de unidades administradas, o total semanal de insulina injetada, e o número total de administrações registadas.

Ao aceder à página, o frontend realiza automaticamente três requisições à API da aplicação: uma para obter os registos de glucose, outra para os registos de insulina e uma terceira para os eventos da agenda. Esses dados são então processados localmente. Os registos mais recentes de glucose e insulina são identificados e apresentados nos dois primeiros blocos da interface. Em seguida, os registos restantes, relativos aos últimos sete dias, são filtrados e utilizados para calcular estatísticas agregadas. Quanto aos eventos da agenda, apenas os eventos futuros são considerados, sendo filtrados por tipo, ordenados cronologicamente e exibido aquele que estiver mais próximo de ocorrer.

O cálculo dos estados de glucose é efetuado através de uma função, que compara os valores registados com os limiares clínicos ajustados ao tipo de regime alimentar do utilizador. Esta abordagem garante ao utilizador um acesso imediato a um resumo funcional e clínico do seu estado, promovendo o autocontrolo, incentivando a adesão ao tratamento e facilitando a comunicação com os profissionais de saúde.

6.2 Registos

O componente `FormRender` atua como um mecanismo versátil para apresentar vários formulários de introdução de dados na aplicação. Em vez de ter componentes separados para cada tipo de formulário (por exemplo, entrada de glucose, entrada de insulina, registo de utilizador), o `FormRender` constrói e apresenta dinamicamente o formulário adequado com base na configuração.

O `FormRender` opera através do carregamento da sua estrutura de formulário, definições de campo, regras de validação e estilo a partir de ficheiros de configuração JSON externos dedicados. Para qualquer tipo de formulário específico solicitado (como 'glucose', 'insulina' ou 'individuo'), o `FormRender` recupera o modelo de dados JSON (JDT) e as informações de estilo relevantes. As tarefas complexas de exibir a interface do utilizador do formulário e garantir a validade dos dados são então entregues à biblioteca `protected-aidaforms`.

Para entradas de dados rotineiras, como registos de glucose ou insulina, o registo comunica diretamente com um endpoint da API de backend (`/api/registos/compositions`). Após o preenchimento do formulário, o `FormRender` obtém um token de autenticação, empacota os dados do formulário e envia-os para o servidor para armazenamento. O backend valida os dados e guarda-os sob o perfil do

utilizador autenticado. O `FormRender` fornece feedback imediato ao utilizador, exibindo mensagens de sucesso ou erro e frequentemente redirecionando-o após uma entrada bem-sucedida.

O processo de registo de utilizador emprega o `FormRender` de forma diferente devido à sua natureza de várias etapas. Inicialmente, as credenciais (nome de utilizador/palavra-passe) são recolhidas e temporariamente armazenadas por um componente separado. Posteriormente, o componente é utilizado com `type="individuo"` para exibir um formulário para detalhes pessoais. Crucialmente, após a submissão destes detalhes pessoais, a API de registo final não é contactada diretamente. Em vez disso, utiliza um mecanismo de callback para passar as informações pessoais recolhidas para o seu componente pai (`Register.jsx`).

Esta delegação permite que o componente consolide todas as informações necessárias: as credenciais da primeira etapa e os detalhes pessoais do `FormRender`. Só então o `Register.jsx` faz o pedido de registo completo para um endpoint de backend dedicado (`/api/auth/register`).

6.3 Painel de Estatísticas

O Painel de Estatísticas foi concebido para fornecer aos utilizadores uma análise visual abrangente dos seus dados de saúde registados. A sua principal função é transformar dados numéricos brutos em tendências compreensíveis, permitindo aos utilizadores monitorizar os seus níveis de glicose, administrações de insulina e alterações de peso ao longo do tempo.

Este componente é responsável pela recuperação e apresentação de métricas históricas de saúde. Obtém registos de glicose e insulina a partir do backend e processa os dados de peso associados. A evolução temporal da glucose, da insulina e do peso é apresentada em gráficos de linhas individuais para uma análise mais focada. Além disso, o componente calcula e apresenta estatísticas resumidas - como valores mínimos, máximos e médios - para cada métrica de saúde dentro de um período de tempo selecionado, oferecendo uma visão geral estatística imediata.

O sistema fornece funcionalidade para ajustes dinâmicos da janela de tempo, permitindo aos utilizadores analisar dados durante vários períodos. As predefinições disponíveis incluem '24 horas', '7 dias', '14 dias' e '28 dias', bem como uma opção para apresentar "todos os dados", facilitando a avaliação de tendências a curto e a longo prazo. Além disso, os utilizadores podem efetuar uma análise mais granular através de filtros de dados específicos. As entradas de glicose podem ser separadas por contexto relacionado com a refeição (por exemplo, 'Jejum', 'Pós-prandial'), e os registos de insulina podem ser filtrados pela via de administração (por exemplo, 'Subcutânea'), refinando assim os dados para satisfazer

necessidades analíticas específicas.

6.4 Agenda

Com o objetivo de auxiliar os utilizadores na organização da sua rotina de medições de glicose e administrações de insulina, foi implementada uma funcionalidade de agenda que permite a marcação destes eventos de forma prática e intuitiva. Embora a funcionalidade se encontre ainda numa fase prototípica — não suportando, por exemplo, a definição de eventos recorrentes —, representa uma base sólida para um sistema de agendamento com elevado potencial de evolução.

A agenda realiza o carregamento inicial (fetch) dos eventos armazenados na tabela `agenda` da base de dados, mantendo-os em cache local para evitar acessos repetitivos ao servidor, sempre que não existam alterações. Os eventos são então apresentados ao utilizador de acordo com o modo de visualização selecionado, sendo possível consultar as marcações em formato diário, semanal ou mensal, consoante a preferência.

Para facilitar a criação de novos eventos, é disponibilizado um modal de marcação, com sugestões pré-definidas de horários comuns para medições de glicose e administrações de insulina. Após agendados, os eventos podem ser marcados como realizados ou não realizados, sendo também possível proceder à eliminação dos mesmos. A aplicação destaca automaticamente os eventos em atraso e os que devem ser realizados no dia corrente, apresentando de forma clara o número total de tarefas pendentes e concluídas no próprio dia.

6.5 Histórico

A página do histórico é uma componente vital para a gestão e visualização dos registos de saúde do utilizador. O seu propósito principal é exibir uma tabela abrangente com todas as medições de glicose e injeções de insulina realizadas pelo utilizador. Esta tabela organiza os dados por data e tipo de registo (glicose ou insulina), apresentando o valor correspondente de cada medição ou injeção. Além da visualização tabular, a página oferece a funcionalidade de enviar registos individuais ou em lote para sistemas externos.

Para a obtenção dos dados, a página de histórico realiza duas chamadas distintas à API. Uma requisição é feita para obter os registos de glicose, e outra para os registos de insulina. Ambas as requisições são assíncronas e utilizam um token de autenticação para garantir a segurança dos dados. Antes da informação ser apresentada ao utilizador, os dados são processados e combinados, sendo posteriormente

ordenados cronologicamente para facilitar a leitura e análise. A página também calcula e exibe o número total de registos para cada tipo (glicose e insulina), fornecendo um resumo estatístico rápido ao utilizador.

Um dos recursos chave da página de histórico é a capacidade de visualizar os detalhes de cada registo individualmente. Ao selecionar um registo específico, um pequeno modal é exibido, contendo todos os campos e valores associados que foram preenchidos no formulário de registo original, oferecendo uma visão aprofundada dos dados.

Para a funcionalidade de envio de registos, a aplicação integra-se com o Mirth Connect, uma plataforma de integração de dados de saúde. A comunicação com o Mirth é realizada utilizando o padrão HL7 FHIR, que facilita a interoperabilidade e a troca segura de informações de saúde entre diferentes sistemas. Todas estas operações comunicam com o backend através de endpoints dedicados (`/api/fhir/send/`, `/api/fhir/send-bulk`, `/api/fhir/test-connection`), garantindo que os dados são formatados e enviados de acordo com as especificações FHIR para uma integração eficaz. As respostas das operações de envio incluem mensagens de sucesso ou erro para o utilizador, indicando o estado da transmissão dos dados.

Por fim, a página de histórico oferece funcionalidades de filtragem robustas. O utilizador pode selecionar quais os registos que deseja visualizar, ajustando a tabela de acordo com a sua preferência.

6.6 Perfil

A página de perfil permite aos utilizadores visualizar e modificar as suas informações pessoais e médicas. Os dados são carregados assincronamente a partir de uma API segura, com um indicador de carregamento e gestão de erros para uma experiência de utilizador fluida.

Nesta secção, são exibidos dados como nome completo, número de utente, data de nascimento, género e morada. Apenas os contactos (emails e telefones) podem ser editados. Os utilizadores podem adicionar, remover e modificar múltiplos endereços de email e números de telefone, sendo o primeiro contacto designado como "Principal".

Ainda na mesma página, são apresentados a altura e o peso do utilizador, ambos editáveis. O Índice de Massa Corporal (IMC) é calculado e exibido dinamicamente com base nestes valores, acompanhado de uma interpretação (por exemplo, "Peso Baixo", "Normal", "Sobrepeso", "Obesidade"), categorizada por cores para fácil compreensão.

Quando o utilizador decide editar os seus dados, os campos editáveis tornam-se ativos e são apresentadas as opções para "Guardar"ou "Cancelar"as alterações. Ao guardar, as modificações na altura, peso,

emails e telefones são enviadas para a API. Em caso de sucesso, o perfil é atualizado e uma mensagem de confirmação é exibida. Se ocorrer um erro, uma mensagem de alerta é apresentada.

6.7 Opções

A página de opções da aplicação foi pensada para que os utilizadores possam personalizar a sua experiência, especialmente no que diz respeito à gestão de notificações e alertas de glicose.

A aplicação verifica o estado das permissões de notificação do navegador e informa o utilizador se estão ativas, negadas ou pendentes. Se necessário, há uma opção para solicitar ou reativar as notificações.

Os utilizadores podem definir até três tempos de aviso antes das medições de glicose agendadas:

- **Primeiro Aviso:** 30 minutos antes.
- **Segundo Aviso:** 15 minutos antes.
- **Terceiro Aviso:** 5 minutos antes.

Além destes, a aplicação também garante notificações no momento exato da medição e em caso de atraso.

É possível personalizar os limites de referência da glicose para diferentes regimes de medição: Jejum, Pós-prandial (após as refeições), Pré-prandial (antes das refeições) e Aleatório.

Para cada regime, o utilizador pode ajustar os valores "Mínimo Normal" e "Máximo Normal", que devem estar entre 30 e 400 mg/dL. Com base nestes limiares, as medições de glicose são categorizadas visualmente:

- **Normal:** Valores dentro dos limites (verde).
- **Cuidado:** Valores ligeiramente fora dos limites (amarelo).
- **Perigo:** Valores muito fora dos limites (vermelho).

As configurações são carregadas automaticamente ao abrir a página e são guardadas de forma persistente, garantindo que as lógicas de notificação e categorização de glicose utilizem sempre as definições mais recentes. Mensagens de sucesso ou erro são exibidas para manter o utilizador informado sobre o estado das suas configurações.

6.8 Interface do Assistente Virtual

No frontend, o assistente virtual foi implementado como um widget interativo que proporciona uma experiência de chat intuitiva e acessível. Este componente foi desenvolvido utilizando React e integra-se perfeitamente com o design da aplicação.

O assistente é materializado através do componente `ChatWidget.jsx`, que implementa uma interface de chat flutuante posicionada no canto inferior direito da aplicação. Esta localização estratégica garante que o assistente está sempre acessível sem interferir com o conteúdo principal da aplicação.

O assistente virtual está intrinsecamente ligado ao sistema de autenticação da aplicação. Apenas utilizadores autenticados podem aceder ao chatbot, e todas as comunicações são realizadas utilizando tokens JWT válidos. Esta integração garante que o widget apenas é apresentado a utilizadores com sessões ativas e que todas as requisições ao backend incluem credenciais de autenticação adequadas.

Esta implementação frontend complementa a infraestrutura backend, criando uma experiência de assistente virtual que enriquece significativamente a utilidade da aplicação de gestão da diabetes.

6.9 Login

A interface de login permite que um utilizador previamente registado aceda à aplicação através da inserção do seu nome de utilizador e respetiva palavra-passe. Esta funcionalidade foi implementada com o suporte de contexto de autenticação (`AuthContext`) para gerir o estado do utilizador autenticado.

Ao submeter o formulário, é feita uma requisição à API (`/api/auth/login`), onde são enviados os dados introduzidos pelo utilizador. O endpoint da API é responsável por validar as credenciais fornecidas, comparando-as com os dados existentes na base de dados. Caso a autenticação seja bem-sucedida, os dados do utilizador e o respetivo token de acesso são armazenados através do método `login` do contexto de autenticação, permitindo o acesso às restantes funcionalidades da aplicação.

Em caso de falha, é apresentada ao utilizador uma mensagem de erro apropriada. Adicionalmente, é implementado um estado de carregamento para informar o utilizador que o pedido está a ser processado.

A interface inclui também um link para a página de registo, direcionando os utilizadores que ainda não possuem conta para criarem um novo registo.

7 Potenciais Melhorias da Aplicação

Tal como em qualquer projeto de desenvolvimento de software, é essencial identificar oportunidades de melhoria que contribuam para a evolução da aplicação. Embora a solução desenvolvida integre um conjunto relevante de funcionalidades úteis para pessoas com diabetes, existem áreas que, se aprimoradas, poderiam melhorar significativamente a experiência de utilização, a eficiência do sistema e a sua adoção em contextos reais.

Do ponto de vista da usabilidade, destaca-se a possibilidade de introduzir opções adicionais de personalização ao nível das definições da aplicação. A seleção de perfis predefinidos com base em características clínicas e demográficas — por exemplo, distinguindo entre utilizadores do sexo masculino ou feminino, ou entre pessoas com diabetes tipo 1 ou tipo 2 — permitiria a configuração automática de `thresholds` clínicos recomendados. Embora estes valores possam ser ajustados manualmente, a pré-configuração com base em orientações médicas reconhecidas facilitaria a utilização da aplicação, especialmente por utilizadores menos experientes.

Relativamente à funcionalidade de agenda, esta constitui atualmente apenas um protótipo inicial. Uma das suas principais limitações é a ausência de suporte para eventos recorrentes — como medições de glicose agendadas para se repetirem diariamente — o que obriga o utilizador a criar cada evento manualmente. A inclusão de recorrência representaria uma melhoria substancial, tornando a agenda mais eficiente, intuitiva e alinhada com as rotinas dos utilizadores, funcionando como um verdadeiro assistente terapêutico.

Verificaram-se também alguns constrangimentos de desempenho, nomeadamente `reloads` completos da página causados pela aplicação de estilos provenientes dos formulários `OpenEHR`, que afetam a fluidez e reatividade esperadas numa aplicação `React`. Uma abordagem mais robusta à integração e renderização destes formulários, evitando alterações colaterais aos estilos globais, permitiria preservar o ciclo de `rendering` do `React` e reduzir recarregamentos desnecessários.

Por fim, importa reforçar a necessidade de uma maior portabilidade para dispositivos móveis. Dado o papel central dos `smartphones` na gestão diária da saúde, uma versão responsiva ou aplicação `nativa/híbrida` seria fundamental para aumentar a adoção. Tal permitiria ao utilizador receber notificações, consultar a agenda e registar dados em tempo real, sem depender do acesso a um computador pessoal. Neste sentido, um `design mobile-first` representa um dos maiores incrementos possíveis na utilidade prática da aplicação.

8 Conclusão

O presente trabalho permitiu ao grupo aplicar e consolidar conhecimentos no domínio do desenvolvimento web full-stack, através da construção de uma aplicação integrada no contexto do processo clínico eletrónico. Esta experiência revelou-se fundamental para compreender como os formulários openEHR facilitam a criação de interfaces uniformes e alinhadas com os standards mais recentes da informática em saúde, bem como para explorar como a estruturação de dados segundo o protocolo FHIR garante uma comunicação interoperável, segura e coerente entre sistemas heterogéneos.

Durante o desenvolvimento, foi possível explorar as vantagens das tecnologias adotadas, nomeadamente a responsividade e modularidade do React.js na construção de interfaces ricas e eficientes, e a versatilidade do Express.js na definição clara e escalável de rotas para o backend. Esta combinação revelou-se eficaz na promoção de uma separação clara de responsabilidades entre camadas da aplicação, simplificando o desenvolvimento e reforçando boas práticas de segurança e organização do código.

Adicionalmente, foi estudada a utilização de uma base de dados híbrida, que combina a estrutura de um sistema relacional com a flexibilidade de campos em JSON, permitindo o armazenamento eficiente dos registos clínicos submetidos pelos utilizadores. Esta abordagem mostrou-se particularmente útil para manter a integridade dos dados clínicos estruturados de forma dinâmica, como os provenientes dos formulários openEHR.

Em fase de encerramento, importa salientar o impacto transformador que soluções digitais como a aqui desenvolvida podem ter na gestão de doenças crónicas, como a diabetes. Ao aproximar o utilizador do seu próprio processo terapêutico, estas ferramentas não só facilitam a monitorização e o cumprimento dos regimes clínicos, como também empoderam o doente, tornando-o um agente ativo no controlo da sua condição. Continuar a investir no desenvolvimento de aplicações deste tipo é um passo essencial para a construção de um futuro onde doenças outrora incapacitantes são geridas com autonomia, precisão e dignidade.