**ATIVIDADES EXTENSIONISTAS**

**Trabalho Final**

**Curso**

(    ) Bacharelado em Engenharia da Computação

(    ) Bacharelado em Engenharia de Software

( ) Bacharelado em Ciência da Computação

( ) Bacharelado em Sistemas de Informação

( X ) CST em Análise e Desenvolvimento de Sistemas

(    ) CST em Banco de Dados

(    ) CST em Ciência de Dados

(    ) CST em Desenvolvimento Mobile

(    ) CST em Gestão da Tecnologia da Informação

(    ) CST em Jogos Digitais

(    ) CST em Redes de Computadores

**Disciplina**

( X ) Atividade Extensionista I: Tecnologia Aplicada à Inclusão Digital – Levantamento

(    ) Atividade Extensionista II: Tecnologia Aplicada à Inclusão Digital – Projeto

(    ) Atividade Extensionista III: Tecnologia Aplicada à Inclusão Digital – Análise

(    ) Atividade Extensionista IV: Tecnologia Aplicada à Inclusão Digital – Implementação

**Etapa**

( X ) Validação da proposta

(    ) Trabalho final

**Aluno(s) e RU(s)**

|  |  |
| --- | --- |
| **Aluno** | **RU** |
| Lindomar José Batistão | 4427651 |

**Título**

Implementar API (DRF) e App (React Native) para indicadores de saúde e lembretes de medicação

**Setor de Aplicação**

A adesão ao tratamento e o acompanhamento de indicadores de saúde (pressão arterial, glicemia e colesterol) são desafios recorrentes em ambientes hospitalares e na atenção primária. Uma solução digital multiplataforma pode apoiar pacientes e equipes de saúde, promovendo o ODS 3 (Saúde e bem-estar), ao reduzir falhas de acompanhamento e ampliar a comunicação entre paciente e profissional.

**Objetivos de Desenvolvimento Sustentável (ODS)**

(    ) 01. Erradicação da pobreza

(    ) 02. Fome zero e agricultura sustentável

( X ) 03. Saúde e bem-estar

(    ) 04. Educação de qualidade

(    ) 05. Igualdade de gênero

(    ) 06. Água potável e saneamento

(    ) 07. Energia limpa e acessível

(    ) 08. Trabalho decente e crescimento econômico

(    ) 09. Indústria, inovação e infraestrutura

(    ) 10. Redução das desigualdades

(    ) 11. Cidades e comunidades sustentáveis

(    ) 12. Consumo e produção responsáveis

(    ) 13. Ação contra a mudança global do clima

(    ) 14. Vida na água

(    ) 15. Vida terrestre

(    ) 16. Paz, justiça e instituições eficazes

(    ) 17. Parcerias e meios de implementação

**Objetivos**

## Objetivo Geral:

Desenvolver uma aplicação computacional composta por uma API REST em Django Rest Framework (com banco SQLite) e um aplicativo móvel em React Native, capaz de registrar indicadores de saúde, gerenciar rotinas de medicamentos com lembretes e realizar autenticação segura (JWT).

**Objetivos Específicos:**

* Projetar e implementar o banco SQLite e a API REST (DRF) com endpoints de usuários, pressão arterial, glicemia, colesterol e rotinas de medicamentos (CRUD).
* Implementar autenticação e autorização com JWT, configurar CORS e armazenamento seguro do token no aplicativo.
* Desenvolver o aplicativo React Native com telas Login, Inicial, Indicadores e Medicamentos, incluindo cadastro/consulta e notificações locais para horários de doses.
* Validar o protótipo com a comunidade-alvo (mín. *N* participantes), coletando feedback de usabilidade e acessibilidade (pt-BR).
* Documentar instalação/uso e política de privacidade em conformidade com a LGPD, publicando o código no GitHub.

**Metodologia**

Analisar requisitos funcionais e não funcionais. Duração 1 dia.

Identificar necessidades dos usuários. Duração 1 dia.

Design do Sistema

Duração 2 dias

Especificar arquitetura do sistema. Duração 1 dia.

Criar wireframes e protótipos. Duração 1 dia.

Definir modelo de dados e APIs. Duração 1 dia.

Desenvolvimento Front-End. Duração 2 dias.

Configurar ambiente React Native. Duração 1 dia.

Desenvolvimento Back-End. Duração 2 dias.

Desenvolver funcionalidades principais. Duração 4 dias.

Implementar interfaces de usuário. Duração 2 dias.

Configurar ambiente Django Rest Framework. Duração 1 dia.

Integração e Testes. Duração 1 dia.

Desenvolver endpoints da API. Duração 3 dias.

Implementar modelos e views. Duração 2 dias.

Integrar front-end com back-end. Duração 2 dias.

Configurar servidores e hospedagem. Duração 2 dias.

Monitorar uso e desempenho. Duração 2 dias.

Deploy em lojas de aplicativos. Duração 1 dia.

Deploy e Monitoramento. Duração 1 dia.

Corrigir bugs e otimizar desempenho. Duração 3 dias.

Testes unitários, de integração e de sistema. Duração 2 dias.

**Resultados Esperados/Obtidos**

Para a **modelagem de requisitos**, utilizou-se a UML (Unified Modeling Language), a fim de representar as principais funcionalidades do sistema através de **diagramas de caso de uso**. A **modelagem do banco de dados** foi realizada através de um **Diagrama Entidade-Relacionamento (DER)**, que define as tabelas, atributos e relacionamentos necessários para o armazenamento de informações.

Será adotado um processo incremental/iterativo com entregas semanais. As etapas incluem: (1) elicitação de requisitos, (2) modelagem (UML/DER), (3) implementação do back-end (DRF) e front-end (React Native), (4) testes e validação com usuários, (5) documentação e implantação do protótipo.

Protótipo funcional (API + aplicativo mobile) que permite autenticação, registro e consulta de indicadores, gerenciamento de medicamentos com lembretes e exportação simples dos dados. Relatório de testes, documentação completa e repositório público.

**1.1 Diagrama de Caso de Uso**

O diagrama da figura 1 representa as interações do **usuário autenticado** com os principais módulos do sistema:

* Cadastro de usuário,
* Registro e consulta de indicadores de saúde (pressão arterial, glicemia e colesterol),
* Gerenciamento de calendário de medicamentos.

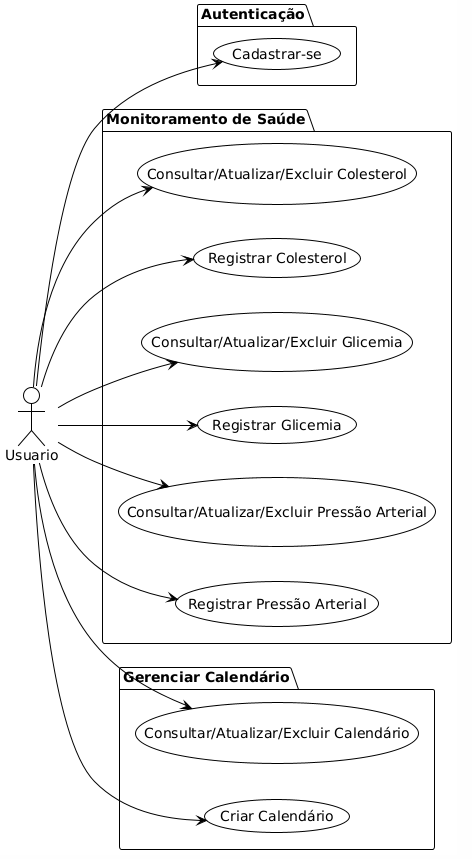


Figura 1 - Diagrama de Caso de Uso

**1.2 Modelagem de Dados – DER**

O **Diagrama Entidade-Relacionamento** (figura 2) representa as entidades do sistema e seus relacionamentos. Cada usuário pode possuir vários registros de pressão arterial, glicemia, colesterol e datas e horários para tomar medicamentos.

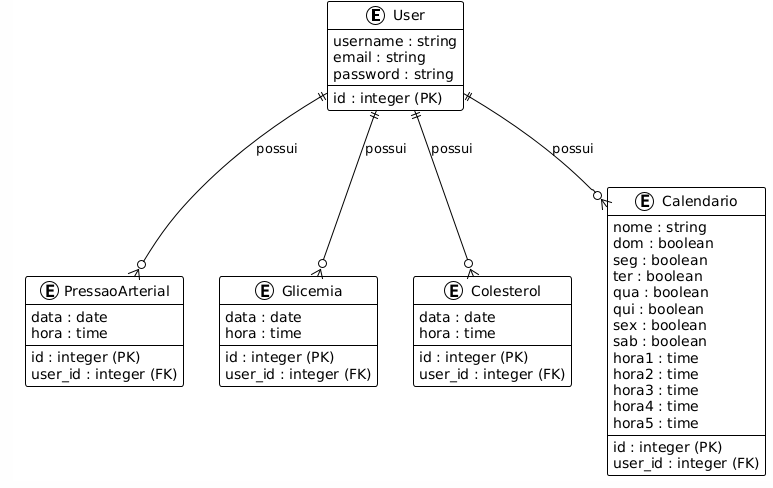


Figura 2 - – Diagrama Entidade-Relacionamento (DER)

**Desenvolvimento do Front-end**

O front-end do sistema foi desenvolvido utilizando **React Native**, com o objetivo de fornecer uma interface mobile responsiva e intuitiva. A escolha dessa tecnologia permite a criação de aplicações móveis multiplataforma (Android e iOS) com base em **JavaScript e JSX**, garantindo performance e uma boa experiência do usuário.

O projeto está estruturado em **componentes funcionais**, que utilizam o **React Hooks** (useState) para gerenciamento de estado local. A comunicação com o backend é realizada por meio da biblioteca **Axios**, que facilita o consumo de APIs REST. Além disso, o **AsyncStorage** é utilizado para o armazenamento local do **token JWT**, garantindo que as requisições autenticadas ao servidor sejam feitas de forma segura.

**Principais Telas do Aplicativo**

**Tela de Boas-Vindas (Home)**

Esta é a primeira tela exibida ao abrir o app. Ela serve como uma introdução amigável ao propósito do aplicativo e fornece acesso às telas de autenticação (Login e Cadastro).

#### Objetivo da Tela

A tela Home tem como função principal:

* Apresentar o nome e identidade visual do aplicativo.
* Informar de forma simples o que o app oferece.
* Permitir que o usuário entre ou crie uma conta.

#### Funcionalidades e estrutura

1. Cabeçalho: Título do app

* Exibe o nome do aplicativo em destaque, promovendo o propósito de monitoramento da saúde.

1. Imagem ilustrativa

* Mostra uma imagem localizada em assets/saude.png, que simboliza saúde e bem-estar.
* Torna a tela mais visual e acolhedora.

1. Lista de funcionalidades disponíveis

* Explica de forma simples e com ícones quais funcionalidades estarão disponíveis após o login.
* Serve como um “preview” das capacidades do app.

1. Ações de autenticação

* Dois botões direcionam o usuário:
  + Para a tela de login existente (Login)
  + Para o cadastro de uma nova conta (SignUp)
* Usam o React Navigation para navegação entre as rotas.

1. Rodapé motivacional

* Frase final que reforça o foco do app em autocuidado e qualidade de vida.

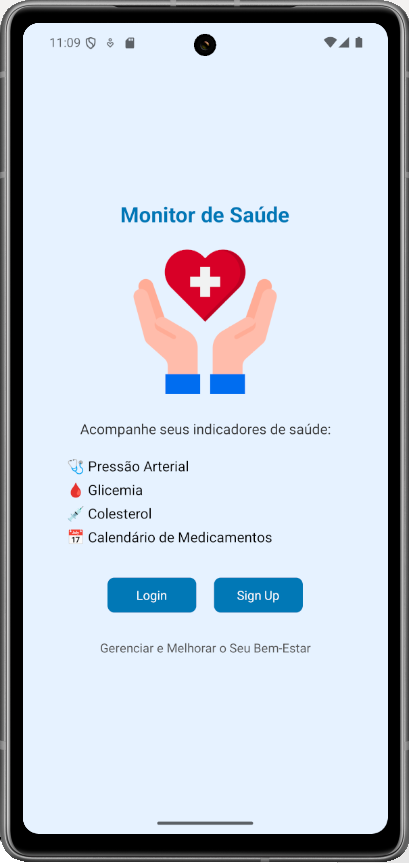


Figura 3 – Home

**Tela de Cadastro de Usuário**

A tela SignUp permite que um novo usuário crie uma conta preenchendo nome de usuário, e-mail e senha. Esse formulário envia os dados para o backend via uma requisição POST, que retorna um token JWT. Esse token é armazenado localmente usando AsyncStorage.

Os estados username, email e password armazenam os dados inseridos pelo usuário nos campos do formulário.

Uma função (handleSignUp) executa a requisição HTTP POST para o endpoint de cadastro. Em caso de sucesso, o token retornado (response.data.access) é salvo no dispositivo com AsyncStorage, e o usuário é redirecionado para a tela inicial.

A interface é composta por três campos de texto e um botão. Cada campo é controlado via estado, e o botão dispara o processo de cadastro quando pressionado.

Resumo

* O usuário insere nome, e-mail e senha;
* O app envia esses dados para o backend via axios;
* O backend responde com um token;
* O token é armazenado com AsyncStorage;
* O app redireciona para a tela inicial.

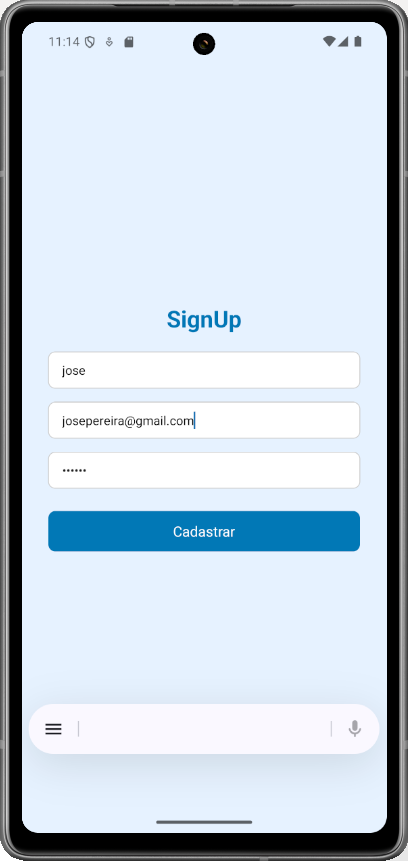
**

Figura 4 - Cadastro de Usuário

**Tela de Login**

Essa tela permite que o usuário acesse o aplicativo informando suas credenciais. Ao submeter os dados, o app envia uma requisição para a API de autenticação do backend. Se os dados estiverem corretos, um token JWT é salvo no AsyncStorage para autenticação futura.

O campo de e-mail está presente, mas não é utilizado no processo de login — apenas username e password são enviados à API. O campo de e-mail pode ser removido para evitar confusão, ou ser usado no futuro.

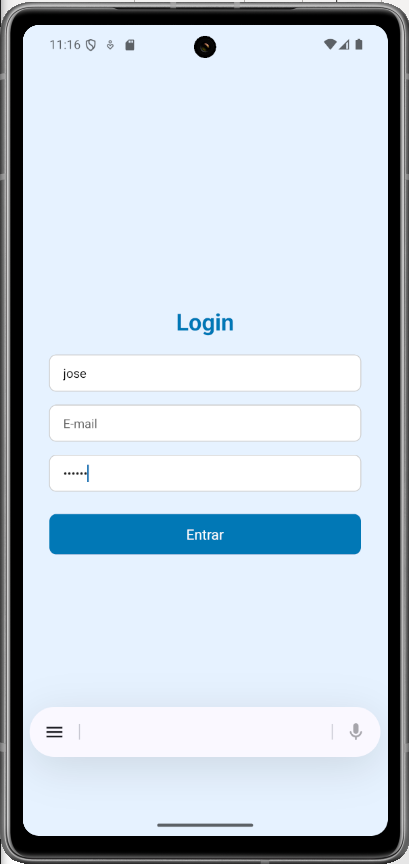


Figura 5 - Login

**Tela Inicial**

Esta tela é a central de navegação do app. Ela dá acesso a todas as principais funcionalidades de monitoramento da saúde do usuário.

Funcionalidades principais:

**1. Visual e apresentação**

* A tela exibe o título Monitor de Saúde com um subtítulo “Acompanhe seus indicadores de saúde”, reforçando a proposta do aplicativo.
* Logo abaixo, é apresentada uma grade com quatro ícones interativos, cada um levando a uma funcionalidade específica do app.

**2. Navegação entre telas**

Cada botão da grade possui um ícone e uma legenda, e quando pressionado leva o usuário para outra seção:

* HeartPulse Pressão arterial Registrar e visualizar medições de pressão arterial
* BloodGlucose Glicemia Registrar os níveis de glicose no sangue
* Cholesterol Colesterol Registrar o colesterol
* Medications Medicamentos Agendamento e controle de medicamentos

Todos esses botões usam navigation.navigate() para mover o usuário entre as rotas do React Navigation.

3. Logout (voltar à tela de login)

* Um botão “Voltar” com o ícone de login invertido é exibido ao final da tela.
* Ao ser pressionado, o app executa o logout:
  + Remove o token de autenticação do AsyncStorage;
  + Redireciona o usuário de volta para a tela “Home” (presumivelmente a tela de login ou boas-vindas).

4. Frase motivacional

Abaixo dos ícones há um texto de incentivo: Gerenciar e Melhorar o Seu Bem-Estar, promovendo a proposta de autocuidado do app.



Figura 6 - Inicial

**Tela de Pressão Arterial**

Esta tela tem como objetivo permitir que o usuário registre manualmente suas medições de pressão arterial. Ela funciona da seguinte forma:

Funcionalidades principais:

1. **Entrada de dados**  
   O usuário pode preencher:
   * A data da medição (no formato dd/mm/aaaa);
   * O horário da medição (no formato hh:mm);
   * O valor da pressão arterial sistólica (pressão "alta", ex: 120);
   * O valor da pressão arterial diastólica (pressão "baixa", ex: 80).
2. **Validação dos campos**  
   O app valida se:
   * A data e hora estão em formatos válidos;
   * Os valores de pressão são numéricos.
3. **Envio dos dados para o backend (API)**

Ao clicar em “Registrar”, os dados são:

* + Convertidos para o formato apropriado (ex: data → aaaa-mm-dd);
  + Enviados para o servidor via uma requisição HTTP POST;
  + A requisição inclui um cabeçalho Authorization com um token JWT, que é recuperado do AsyncStorage (ou seja, o envio é autenticado).

1. **Resposta da API e feedback**
   * Se o servidor responder com sucesso, o app exibe uma mensagem de confirmação ("Pressão registrada com sucesso!");
   * Em caso de erro (problemas na API ou nos dados), uma mensagem de alerta é exibida ao usuário.
2. **Visualização futura dos dados**
   * Embora o gráfico ainda não esteja implementado, a tela já reserva um espaço com o texto "[ Gráfico será exibido aqui ]", indicando que será possível futuramente visualizar a evolução da pressão ao longo do tempo.



Figura 7 - Pressão Arterial

**Tela de Glicemia**

Essa tela permite ao usuário registrar sua taxa de glicemia em uma data e horário específicos. Os dados são validados e enviados via requisição POST para a API, com autenticação por token JWT.

Essa tela:

* Valida data, hora e valor da glicemia;
* Envia os dados de forma segura à API;
* Usa token armazenado no AsyncStorage;
* Limpa os campos após o envio;
* É uma interface intuitiva para o usuário registrar suas medições.



Figura 8 - Glicemia

**Tela de Colesterol:**

Essa tela permite ao usuário registrar os níveis de colesterol LDL e HDL com data e hora. O código está estruturado para fazer validações antes de exibir uma confirmação.

As entradas de data e hora são automaticamente formatadas conforme o usuário digita.

Atualmente, o botão “Registrar” apenas faz as validações e exibe os dados no console como forma de simulação.

A tela de Colesterol:

* Permite digitar e validar data, hora e valores de colesterol;
* Mostra mensagem de sucesso após validação;
* Pode ser facilmente conectada à API com um POST;
* Está pronta para ser estendida com gráficos e histórico.

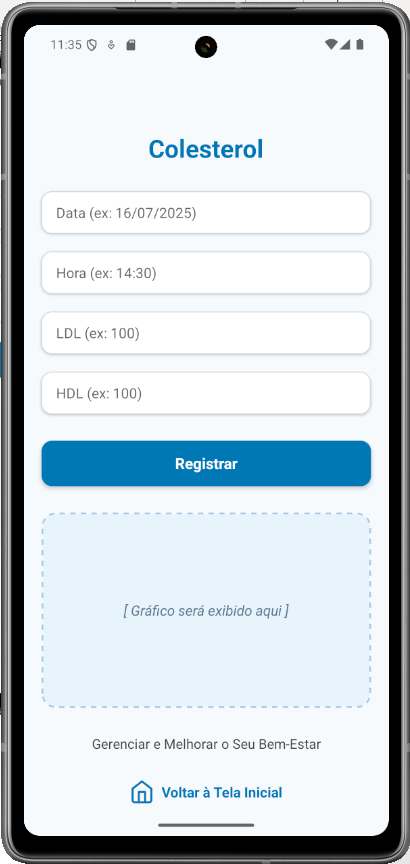


Figura 9 - Colesterol

**Tela de Medicamentos**

Esta tela permite que o usuário registre e atualize a rotina de medicamentos, incluindo:

* Nome do medicamento
* Dias da semana em que será tomado
* Até 5 horários diferentes

Ela é integrada com a API via métodos POST e PUT, utilizando autenticação por token JWT.

1. Estado inicial e carregamento de dados

Ao abrir a tela, é feita uma requisição GET para buscar um calendário já existente.

1. Seleção dos dias da semana

Cada dia pode ser marcado/desmarcado tocando no botão correspondente, isso afeta visualmente a interface e também o payload enviado.

1. Cadastro ou atualização de rotina

O botão "Registrar" ou "Atualizar" envia os dados para o backend. Se já existe um ID salvo (via useEffect), é feito um PUT, caso contrário, um POST.

1. Formatação dos horários

Cada campo de hora usa um manipulador que formata automaticamente para o formato HH:MM

1. Interface do usuário

Exibe:

* Campo para nome do remédio
* Seleção visual dos dias
* Campos de entrada para horários
* Botões de ação

1. Navegação

O botão com ícone "home" retorna à tela inicia

A tela de medicamentos:

* Salva e atualiza uma rotina com nome, dias e horários;
* Carrega dados do backend e atualiza se já existir;
* Utiliza token JWT para autenticação;
* Tem interface intuitiva e limpa;
* Está pronta para ser usada e expandida (como com alarmes ou lembretes).

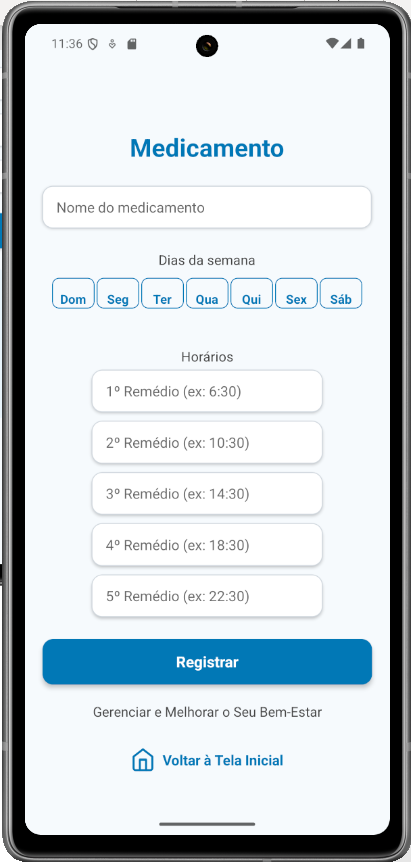
****

Figura 10 - Medicamento

**Considerações**

O desenvolvimento do aplicativo "Monitor Saúde: Gerenciar e Melhorar o Seu Bem-Estar" representa um passo significativo no uso da tecnologia para promover a saúde e o bem-estar pessoal, alinhando-se aos Objetivos de Desenvolvimento Sustentável (ODS) da ONU, especialmente o ODS 3. Este projeto visa oferecer uma ferramenta acessível, eficiente e inclusiva para a gestão de saúde, com funcionalidades que atendem tanto a necessidades individuais quanto coletivas.

#### **Inovação e Impacto**

Este aplicativo não só inovará na maneira como as pessoas gerenciam sua saúde, mas também terá um impacto positivo em diversas áreas:

* **Pessoal**: Melhor adesão ao tratamento, maior conscientização sobre saúde e bem-estar, e uma rotina mais saudável.
* **Profissional**: Melhoria na comunicação entre pacientes e profissionais de saúde, proporcionando um acompanhamento mais preciso e personalizado.
* **Coletivo**: Contribuição significativa para programas de saúde pública, empresas e organizações que visam promover o bem-estar de suas comunidades.

#### **Metodologia e Desenvolvimento**

A metodologia adotada assegura um desenvolvimento estruturado e focado no usuário. Desde a definição clara de objetivos até a coleta de feedback para melhorias contínuas, cada etapa foi projetada para garantir a criação de um aplicativo robusto, eficaz e fácil de usar.

#### **Desenho Universal**

A aplicação dos sete princípios do Desenho Universal no desenvolvimento do aplicativo garante acessibilidade para todos os usuários, incluindo aqueles com deficiências. Isso promove a inclusão digital e assegura que o aplicativo possa ser utilizado por um público diverso.

#### **Sustentabilidade e Futuro**

O projeto não termina com a implementação inicial do aplicativo. A coleta contínua de feedback e a análise de dados de uso permitirão que o aplicativo evolua constantemente, adaptando-se às novas necessidades e incorporando avanços tecnológicos e médicos. Isso garante que o aplicativo permaneça relevante e eficaz a longo prazo.

**Repositório**

Ao acessar o repositório (https://github.com/lindomarbatistao/bemestar.git) leia os README, pois terá um passo a passo de como baixar, instalar e executar o projeto.

Endereço do repositório GitHub

* <https://github.com/lindomarbatistao/bemestar.git>

**Conclusão**

O escopo e os objetivos agora tornam explícita a natureza computacional do projeto, o que atende aos requisitos da disciplina de Atividade Extensionista I e alinha-se ao ODS 3. O plano de desenvolvimento, entregáveis e cronograma permitem evoluir o protótipo para o Trabalho Final.