

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
- ☒ 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

- ☒ 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

Etapas

- ☒ Validação da proposta
- ☐ Trabalho final

Aluno(s) e RU(s)

Aluno	RU
Lindomar José Batista	4427651

Título

Implementar API (Django Rest Framework), App (React Native) e Web (React Js) 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

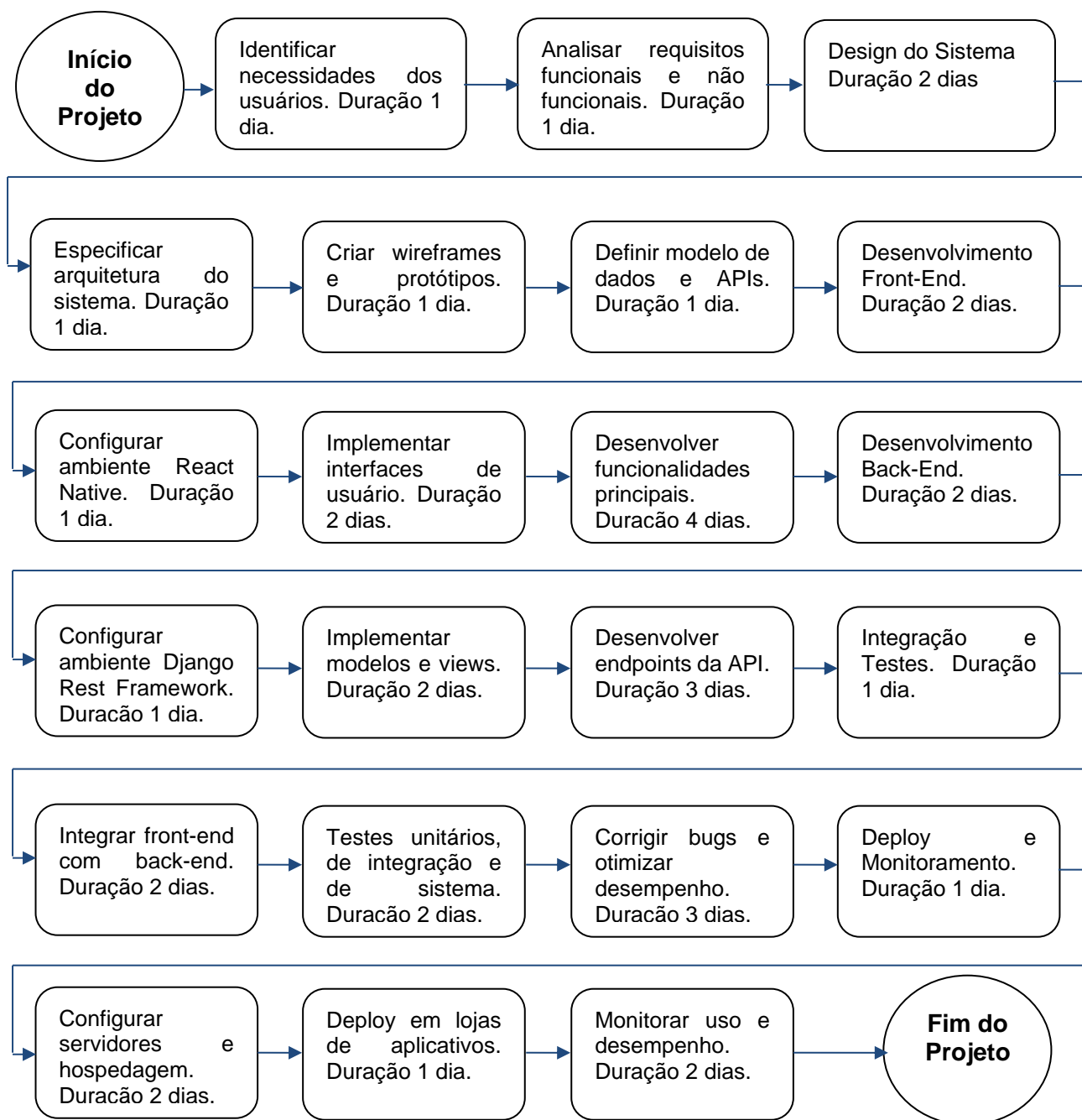
Objetivos Geral

Desenvolver uma aplicação computacional composta por uma API REST em Django Rest Framework com banco SQLite, um aplicativo móvel em React Native e uma página Web, 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.
- Desenvolver também a versão Web em React, integrada ao mesmo backend, permitindo que o sistema seja acessado via navegadores em computadores e ampliando o alcance para comunidades carentes que nem sempre possuem smartphones modernos.
- 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



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.

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.

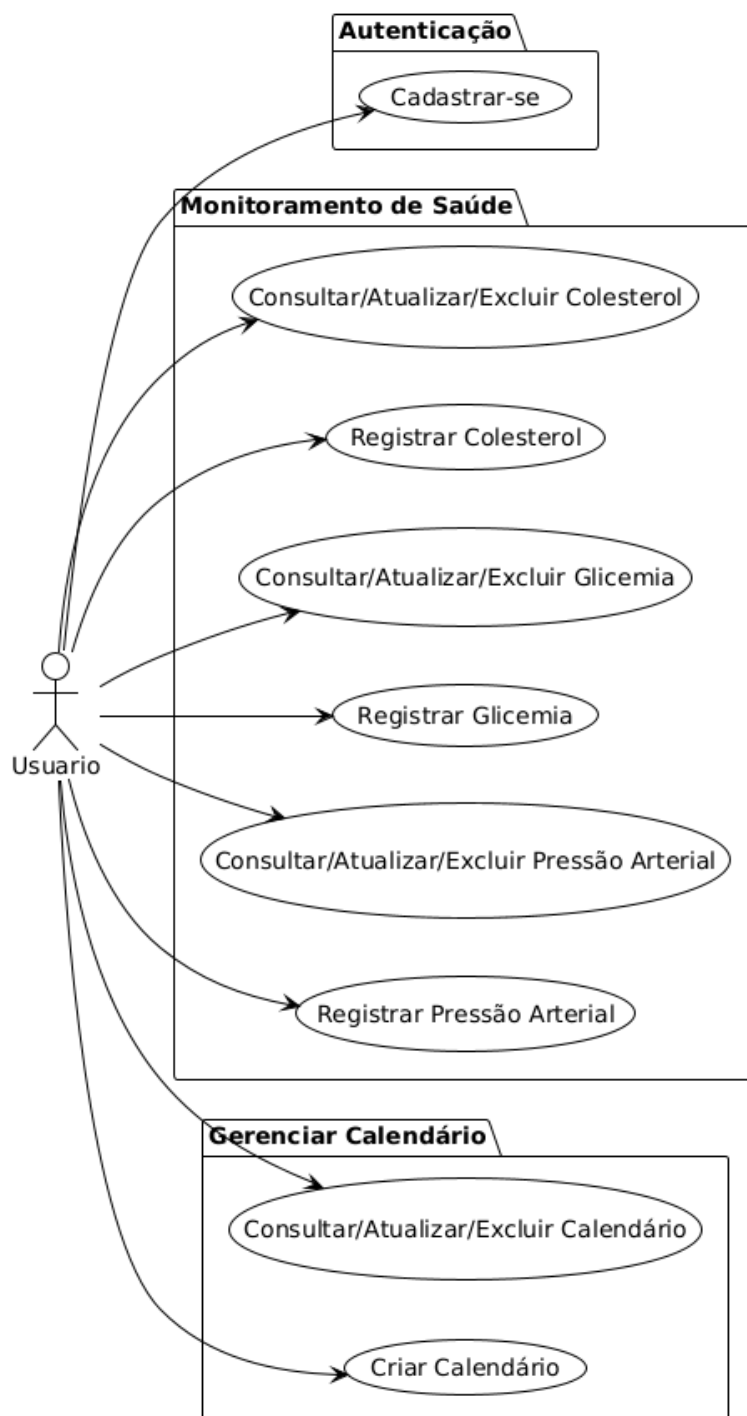


Figura 1 - Diagrama de Caso de Uso

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.

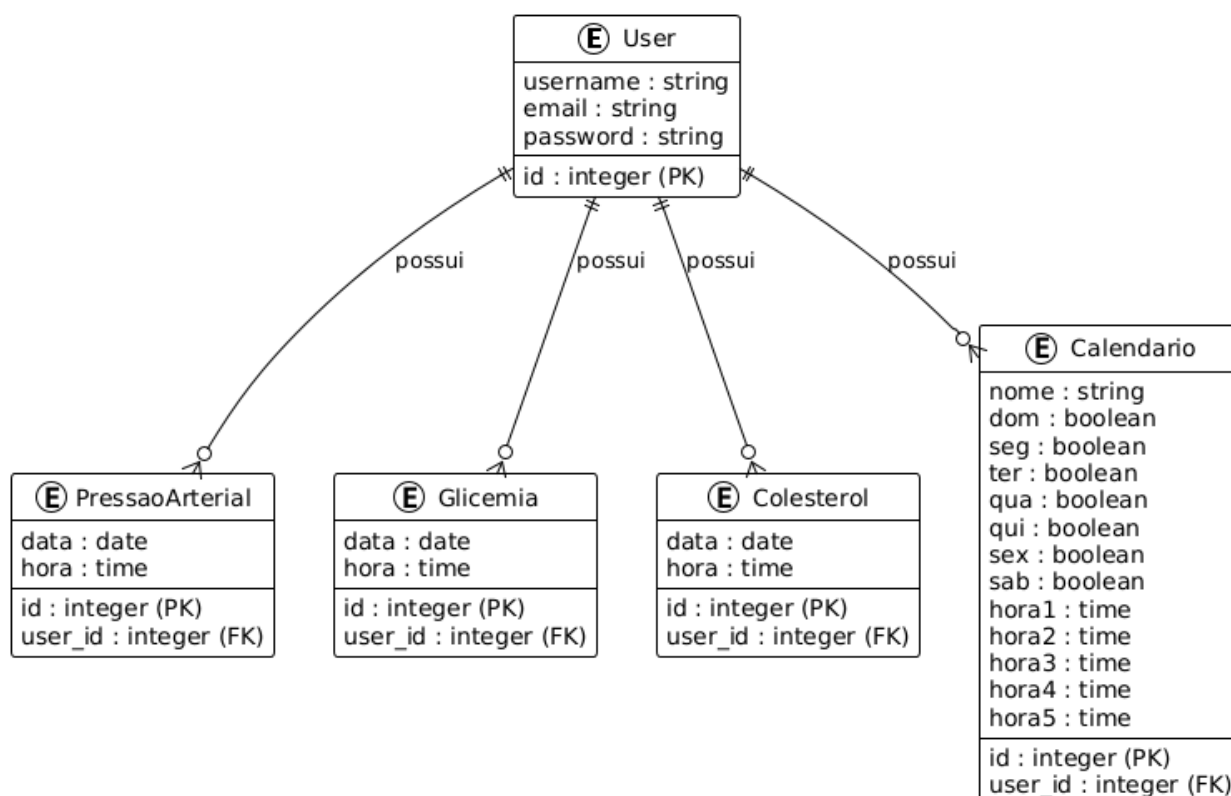


Figura 2 – Diagrama Entidade-Relacionamento (DER)

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.
2. Imagem ilustrativa
 - Mostra uma imagem localizada em assets/saude.png, que simboliza saúde e bem-estar.
 - Torna a tela mais visual e acolhedora.
3. 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.
4. 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.
5. Rodapé motivacional
 - Frase final que reforça o foco do app em autocuidado e qualidade de vida.

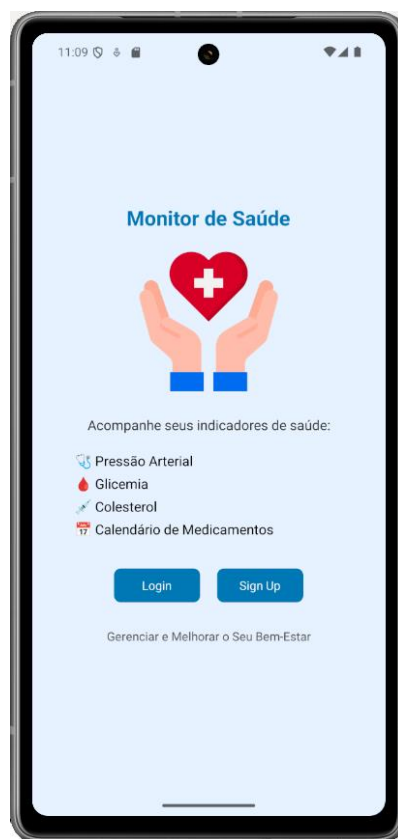


Figura 3 – Home

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.

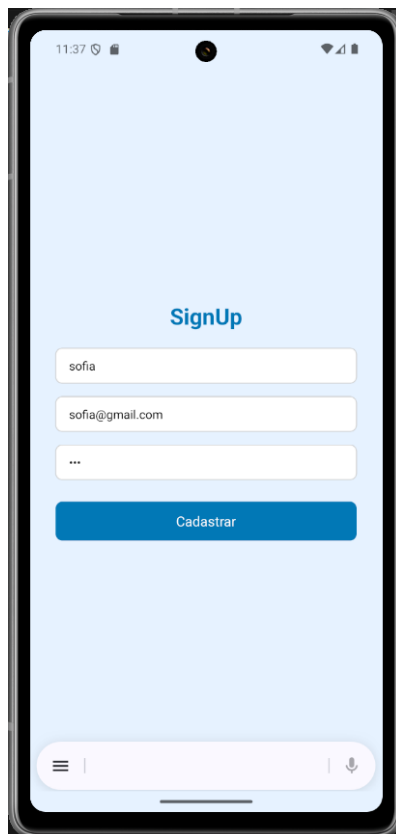


Figura 4 - Cadastro de Usuário

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.

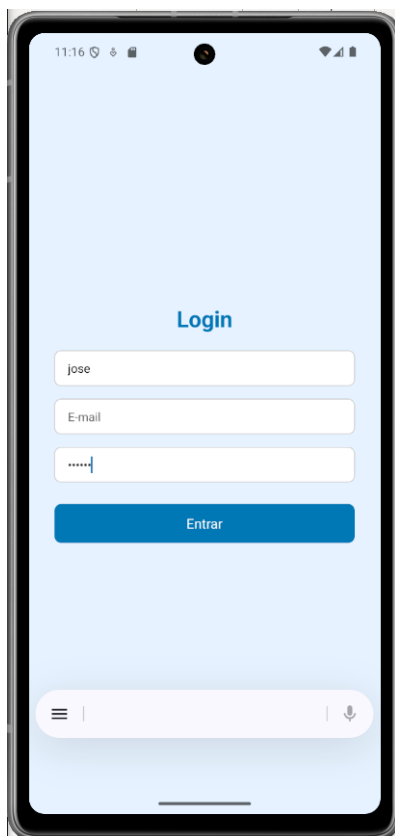


Figura 5 - Login

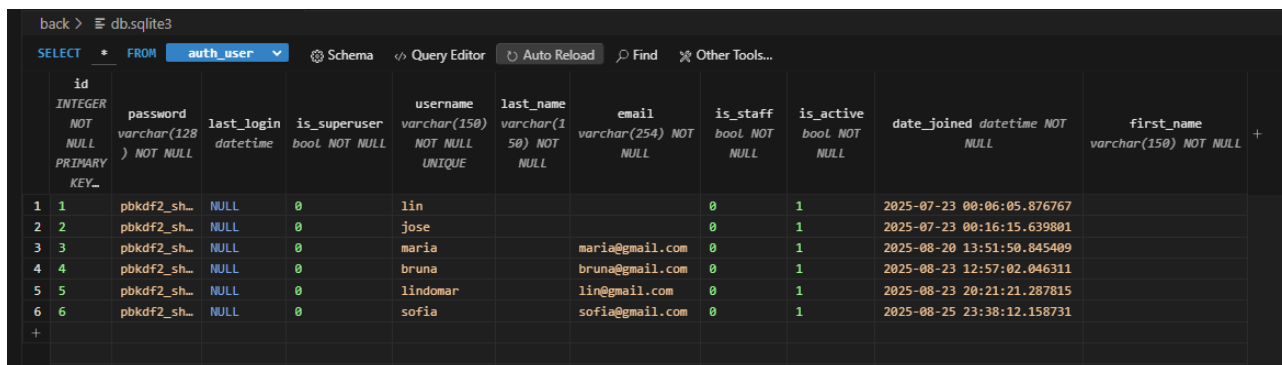
Registro de Usuário no Banco de Dados

Após o cadastro da usuária **Sofia**, foi possível verificar que o registro foi corretamente armazenado no banco de dados **SQLite** utilizado pelo Django.

A tabela `auth_user` é a tabela padrão do framework para gerenciamento de usuários. Ela armazena informações essenciais como:

- **id** → identificador único do usuário.
- **password** → senha armazenada com hash seguro (algoritmo `pbkdf2_sha256`).
- **username** → nome de usuário utilizado para login.
- **first_name** e **last_name** → dados de identificação do usuário.
- **email** → endereço eletrônico cadastrado.
- **is_staff** e **is_active** → flags de controle de acesso e status do usuário.
- **date_joined** → data e hora em que o usuário foi cadastrado no sistema.

Na imagem, é possível observar que o usuário **Sofia** recebeu o **id 6**, está **ativo** (`is_active = 1`), e foi criado em 2025-08-23 23:38:12. Isso confirma que o processo de cadastro funciona corretamente, inserindo os dados no banco de forma automática por meio da API de autenticação e do modelo `User` do Django.



	id	password	last_login	is_superuser	username	last_name	email	is_staff	is_active	date_joined	first_name
1	1	pbkdf2_sh...	NULL	0	lin			0	1	2025-07-23 00:06:05.876767	
2	2	pbkdf2_sh...	NULL	0	jose			0	1	2025-07-23 00:16:15.639881	
3	3	pbkdf2_sh...	NULL	0	maria		maria@gmail.com	0	1	2025-08-20 13:51:50.845409	
4	4	pbkdf2_sh...	NULL	0	bruna		bruna@gmail.com	0	1	2025-08-23 12:57:02.046311	
5	5	pbkdf2_sh...	NULL	0	lindomar		lin@gmail.com	0	1	2025-08-23 20:21:21.287815	
6	6	pbkdf2_sh...	NULL	0	sofia		sofia@gmail.com	0	1	2025-08-25 23:38:12.158731	

Figura 6 - Tabela de usuários

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 arterialregistrar e visualizar medições de pressão arterial
- ✓ BloodGlucose glicemiaregistrar os níveis de glicose no sangue
- ✓ Cholesterol colesterol.....registrar o colesterol
- ✓ Medications medicamentos.....agendar 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 7 - 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).

4. 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.

5. 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.

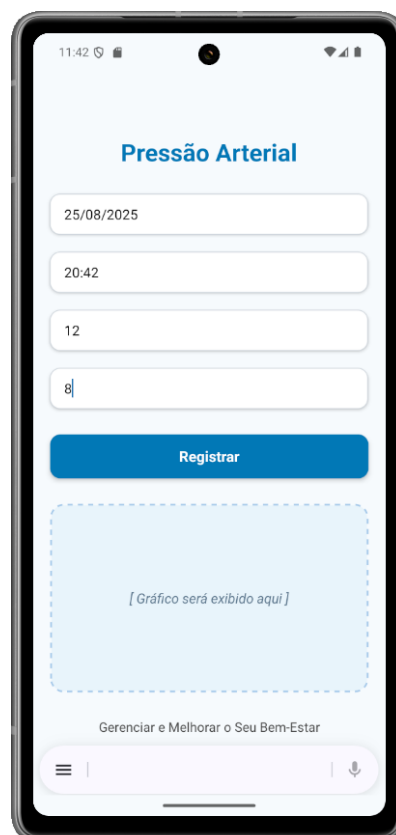


Figura 8 - Pressão Arterial

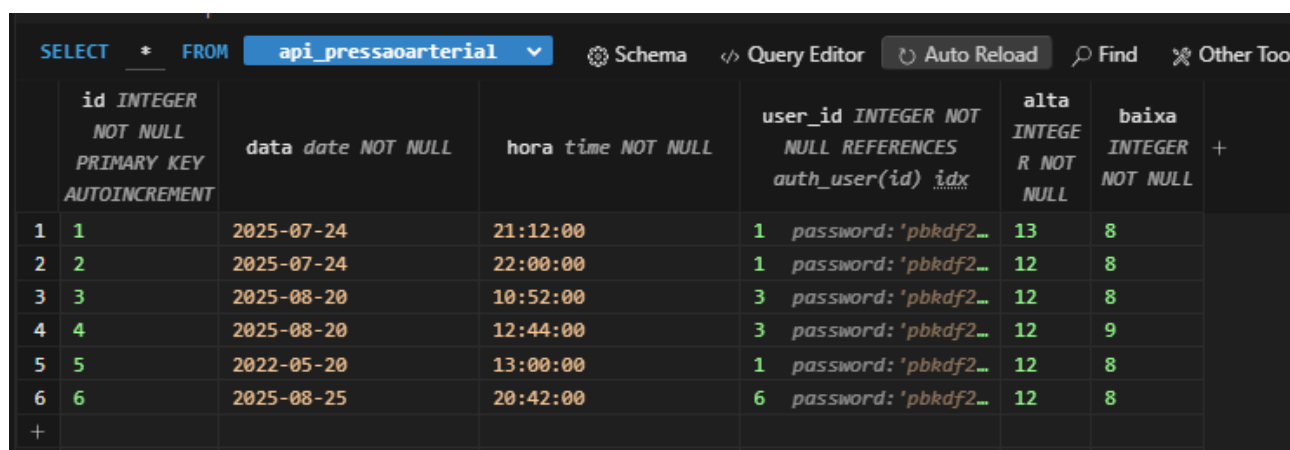
Registro de Pressão Arterial no Banco de Dados

A tabela `api_pressaoarterial` foi criada a partir do **modelo do Django** responsável por armazenar os dados de pressão arterial dos usuários. Cada registro contém:

- **id** → identificador único (chave primária, autoincremento).
- **data** → data da medição registrada.
- **hora** → horário da medição.
- **user_id** → referência ao usuário que realizou o registro (chave estrangeira ligada à tabela `auth_user`).
- **alta** → valor da pressão sistólica (pressão “máxima”).
- **baixa** → valor da pressão diastólica (pressão “mínima”).

Na imagem apresentada, é possível observar:

- O usuário com **id 1** possui múltiplas medições armazenadas em diferentes datas e horários, o que demonstra o acompanhamento contínuo.
- O usuário **id 6 (Sofia)** registrou uma medição em 2025-08-25 às 20:42, com pressão **12/8**, confirmando que o relacionamento entre usuários e medições funciona corretamente.
- Todas as inserções foram feitas de forma automatizada através da **API Django REST**, que valida os dados e garante a persistência no banco.



	id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT	data date NOT NULL	hora time NOT NULL	user_id INTEGER NOT NULL REFERENCES auth_user(id) idx	alta INTEGE R NOT NULL	baixa INTEGER NOT NULL	+
1	1	2025-07-24	21:12:00	1 password: 'pbkdf2...	13	8	
2	2	2025-07-24	22:00:00	1 password: 'pbkdf2...	12	8	
3	3	2025-08-20	10:52:00	3 password: 'pbkdf2...	12	8	
4	4	2025-08-20	12:44:00	3 password: 'pbkdf2...	12	9	
5	5	2022-05-20	13:00:00	1 password: 'pbkdf2...	12	8	
6	6	2025-08-25	20:42:00	6 password: 'pbkdf2...	12	8	
+							

Figura 9 - Tabela de 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 10 - 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.

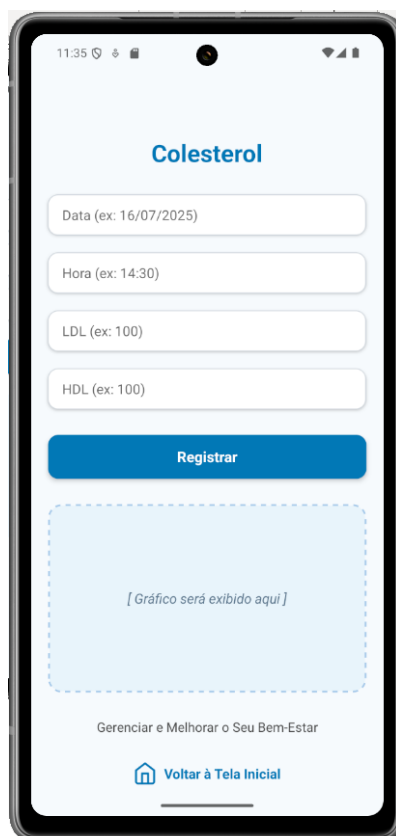


Figura 11 - Colesterol

Registro de Glicemia no Banco de Dados

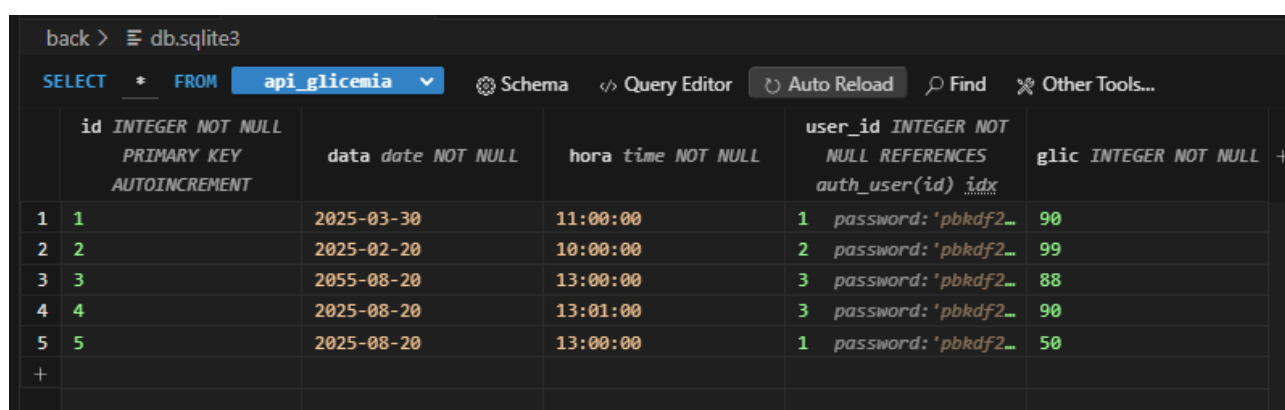
A tabela `api_glicemia` foi criada pelo modelo do Django para armazenar os valores de glicemia informados pelos usuários. Os campos principais são:

- **id** → identificador único do registro (chave primária, autoincremento).
- **data** → data em que a medição foi realizada.
- **hora** → horário exato da medição.
- **user_id** → chave estrangeira que referencia o usuário na tabela `auth_user`, vinculando cada medição ao responsável.
- **glic** → valor da glicemia em mg/dL informado pelo usuário.

Na imagem observamos:

- O usuário **id 1** possui medições em diferentes dias e horários (valores como 90 e 50 mg/dL).
- O usuário **id 3** também registrou medições (ex.: 88 e 90 mg/dL).
- Cada entrada está vinculada ao usuário correto por meio do campo `user_id`.
- Os dados foram gravados de forma consistente via **API Django REST**, garantindo que as medições feitas no aplicativo sejam persistidas no banco.

Isso demonstra que o sistema permite o **registro histórico da glicemia**, essencial para o acompanhamento de pacientes e análise da evolução ao longo do tempo.



	<code>id</code> INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT	<code>data</code> date NOT NULL	<code>hora</code> time NOT NULL	<code>user_id</code> INTEGER NOT NULL REFERENCES <code>auth_user(id)</code> <code>idx</code>	<code>glic</code> INTEGER NOT NULL
1	1	2025-03-30	11:00:00	1 password: 'pbkdf2...	90
2	2	2025-02-20	10:00:00	2 password: 'pbkdf2...	99
3	3	2055-08-20	13:00:00	3 password: 'pbkdf2...	88
4	4	2025-08-20	13:01:00	3 password: 'pbkdf2...	90
5	5	2025-08-20	13:00:00	1 password: 'pbkdf2...	50

Figura 12 - Tabela de glicemia

Tela de Medicamentos

Esta tela tem como objetivo central permitir ao usuário **cadastrar, visualizar e atualizar rotinas de medicamentos** de forma prática e segura.

Funcionalidades principais:

- **Nome do medicamento:** campo de texto para identificar claramente o remédio que será controlado.
- **Dias da semana:** botões interativos que permitem selecionar visualmente em quais dias o medicamento deve ser tomado. Os botões ficam destacados quando selecionados, indicando a escolha do usuário.
- **Horários de administração:** até **cinco campos de horários diferentes**, com formatação automática no padrão **HH:MM**. Isso possibilita organizar tratamentos mais complexos, como medicamentos que precisam ser ingeridos em diferentes momentos do dia.
- **Integração com API:**
 - **GET:** ao carregar a tela, é feita uma requisição para buscar as rotinas já registradas do usuário autenticado.
 - **POST:** se não existir rotina salva, o sistema cria um novo registro.

- **PUT**: se já houver um registro existente, a rotina é atualizada.
- Todas as requisições utilizam **autenticação JWT**, garantindo segurança no acesso aos dados pessoais de saúde.

Interface do usuário:

- A parte superior concentra o **formulário de cadastro** com os campos de medicamento, dias da semana e horários.
- O botão **Cadastrar** executa o envio ao backend, exibindo feedback visual para o usuário.
- Logo abaixo, é exibida a seção **Meus medicamentos**, listando de forma organizada as rotinas já salvas.
 - Cada item da lista mostra:
 - Nome do medicamento.
 - Dias da semana selecionados (destacados com cores).
 - Horários configurados.
 - **Ações rápidas**:
 - Ícone de lápis para editar a rotina.
 - Ícone de lixeira para excluir o registro.
- Na parte inferior, há um botão de navegação com ícone de **home**, permitindo retornar para a tela inicial.

Benefícios e usabilidade:

- **Interface intuitiva**: organizada em seções bem definidas, facilita a compreensão mesmo para usuários com pouca experiência tecnológica.
- **Controle visual**: dias e horários destacados melhoram a memorização da rotina e reduzem a chance de erro.
- **Gerenciamento completo**: permite não só cadastrar, mas também **editar e excluir** rotinas existentes.
- **Expansibilidade**: a tela foi projetada para futuras melhorias, como integração com alarmes e notificações locais para lembrar o horário dos medicamentos.

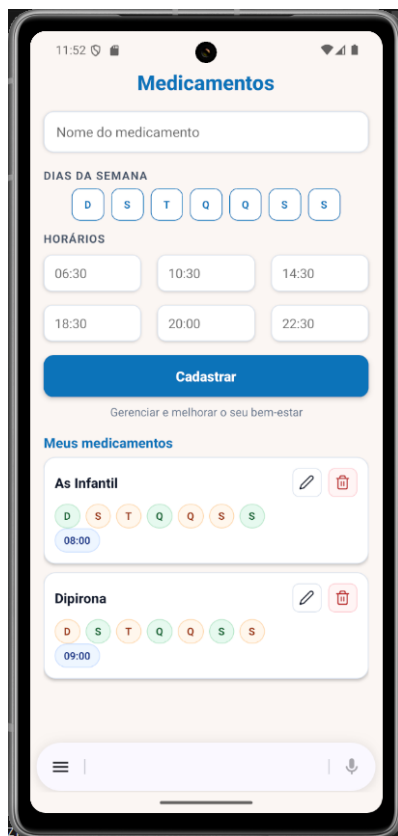


Figura 133 - Página de controle de datas e horários de medicamentos

Registro de Rotinas de Medicamentos no Banco de Dados

A tabela `api_calendario` foi projetada para guardar as rotinas de medicamentos configuradas pelos usuários. Os campos principais são:

- **id** → identificador único da rotina (chave primária).
- **nome** → nome do medicamento cadastrado (ex.: “AS Infantil”, “Dipirona”).
- **dom a sab** → colunas booleanas que representam os dias da semana em que o medicamento deve ser tomado (1 = ativo, 0 = inativo).
- **hora1 ... hora5** → até cinco horários distintos em que o medicamento deve ser administrado.
- **user_id** → chave estrangeira que vincula a rotina ao usuário responsável (referência à tabela `auth_user`).

Na imagem apresentada:

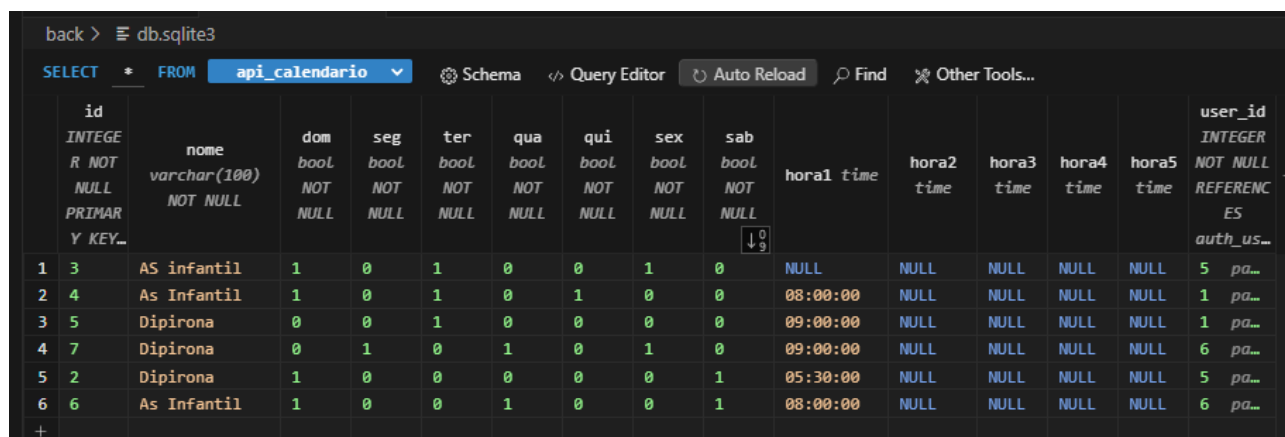
- O usuário **id 5 (lindomar)** possui rotinas de “AS Infantil” e “Dipirona”, com diferentes combinações de dias e horários (ex.: 05:30, 09:00).
- O usuário **id 6 (Sofia)** também cadastrou “AS Infantil”, com horário definido às 08:00.

- O usuário **id 1** tem a rotina de “Dipirona” às 09:00, mostrando que múltiplos usuários podem cadastrar medicamentos distintos ou até o mesmo medicamento em horários diferentes.

Importância

Esse registro no banco demonstra que:

- O sistema não apenas exibe as rotinas na tela, mas também **grava de forma persistente** no banco de dados.
- É possível que cada usuário mantenha várias rotinas de medicamentos de acordo com sua necessidade.
- A estrutura está preparada para futuras melhorias, como **alertas automáticos** e **notificações push**, aproveitando os dados já armazenados.



	id	nome	dom	seg	ter	qua	qui	sex	sab	hora1	hora2	hora3	hora4	hora5	user_id
1	3	AS infantil	1	0	1	0	0	1	0	NULL	NULL	NULL	NULL	NULL	5
2	4	As Infantil	1	0	1	0	1	0	0	08:00:00	NULL	NULL	NULL	NULL	1
3	5	Dipirona	0	0	1	0	0	0	0	09:00:00	NULL	NULL	NULL	NULL	1
4	7	Dipirona	0	1	0	1	0	1	0	09:00:00	NULL	NULL	NULL	NULL	6
5	2	Dipirona	1	0	0	0	0	0	1	05:30:00	NULL	NULL	NULL	NULL	5
6	6	As Infantil	1	0	0	1	0	0	1	08:00:00	NULL	NULL	NULL	NULL	6

Figura 14 - Tabela de calendário

Página Web – Home

A página inicial da aplicação web **Monitor de Saúde** foi desenvolvida com foco em simplicidade, acessibilidade e comunicação direta da proposta do sistema.

Estrutura da Home:

- **Título e slogan:** O título “Monitor de Saúde” é acompanhado por uma frase explicativa que reforça a importância de acompanhar indicadores de saúde para melhorar o bem-estar.
- **Indicadores principais:** A página destaca quatro áreas fundamentais de monitoramento:
 1. **Pressão Arterial** → enfatiza a importância de acompanhar a pressão para prevenir doenças cardíacas, AVC e complicações renais.

2. **Glicemia** → destaca o controle da glicose como medida essencial na prevenção e no gerenciamento do diabetes.
 3. **Colesterol** → alerta sobre os riscos de níveis elevados, que podem obstruir artérias e aumentar as chances de infarto.
 4. **Medicamentos** → reforça a necessidade da adesão correta ao tratamento como fator crucial no controle de doenças crônicas.
- **Ações principais:** Na parte inferior, há botões de **Conecte-se** (login) e **Cadastro**, que direcionam o usuário para iniciar o uso do sistema.
 - **Rodapé motivacional:** A página encerra com a frase “Gerencie e melhore sua qualidade de vida com conhecimento e acompanhamento.”, reforçando o objetivo central do sistema.

Integração com o Backend

A versão Web foi construída reutilizando o mesmo **backend em Django Rest Framework** que já atende o aplicativo mobile. Isso garante:

- Consistência nos dados (informações registradas na web ficam disponíveis no mobile e vice-versa).
- Autenticação segura via JWT.
- Centralização em uma API única, facilitando manutenção e escalabilidade.

Inclusão Digital

Um diferencial desta página é que ela foi **desenvolvida especialmente para ampliar o acesso da comunidade carente**. Muitas pessoas não possuem smartphones modernos ou com espaço suficiente para instalar aplicativos, mas podem acessar o sistema pelo computador de casa, em escolas públicas, telecentros, bibliotecas ou lan houses. Assim, o projeto contribui para a **inclusão digital** e assegura que um número maior de pessoas possa cuidar de sua saúde utilizando tecnologia acessível.



Figura 15 - Página inicial da web

Página Web – Login

A tela de **Login** da aplicação Web permite que usuários previamente cadastrados no sistema acessem suas informações de saúde de forma segura.

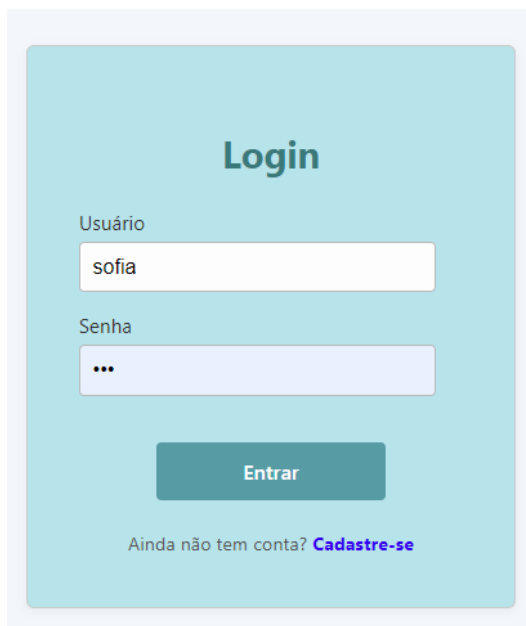
Estrutura da tela:

- **Campos de autenticação:** o usuário deve inserir seu **nome de usuário** e **senha**.
- **Botão de ação:** o botão **Entrar** envia os dados para a API, que valida as credenciais e, em caso de sucesso, libera o acesso ao painel principal.
- **Link de cadastro:** para novos usuários, há a opção “Ainda não tem conta? Cadastre-se”, que direciona para a tela de criação de conta.

Integração com o mesmo banco de dados

Um ponto de destaque é que a autenticação da versão Web utiliza **exatamente o mesmo backend em Django Rest Framework** e o mesmo banco de dados do aplicativo mobile. Na imagem, por exemplo, é utilizada a usuária “**sofia**”, previamente cadastrada pelo aplicativo mobile. Como os dados foram armazenados no banco `auth_user` do Django, ela pode acessar normalmente a aplicação web sem necessidade de novo cadastro.

Na parte inferior da tela de Login foi adicionado um **QR Code**, que permite ao usuário baixar diretamente o aplicativo Android. Essa funcionalidade facilita o acesso, especialmente para pessoas da comunidade que preferem utilizar o sistema no celular em vez do navegador, ampliando ainda mais a inclusão digital.



The image shows a login form titled "Login" on a light blue background. It contains two input fields: "Usuário" with the text "sofia" and "Senha" with three dots indicating a password. Below the fields is a dark blue button labeled "Entrar". At the bottom, there is a link that says "Ainda não tem conta? Cadastre-se".

Figura 16 - Página de login

Página Web – Medicamentos

A tela de **Medicamentos** da versão Web foi desenvolvida para permitir que o usuário registre, visualize e atualize sua rotina de medicação diretamente pelo navegador, de forma simples e acessível.

Estrutura da tela:

- **Formulário de cadastro:** inclui campo para nome do medicamento, seleção dos dias da semana e até cinco horários distintos para administração.
- **Botão de ação:** o botão **Cadastrar** envia as informações para o backend, utilizando a API central do sistema.
- **Listagem de medicamentos:** exibe os medicamentos já cadastrados pelo usuário, incluindo:
 - Nome do medicamento.
 - Dias da semana destacados em que deve ser tomado.
 - Horários configurados.
 - Ícones de ação para **editar** ou **excluir** a rotina.

Integração com o backend

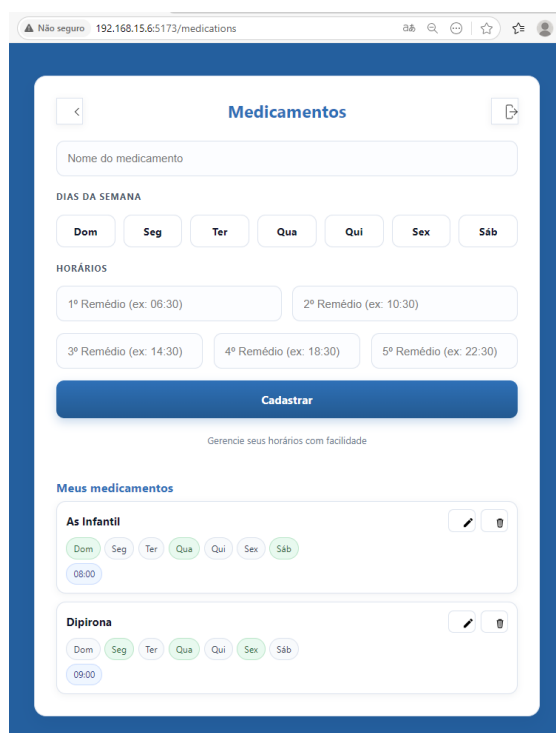
Um ponto essencial é que esta tela consome os **mesmos dados armazenados no backend Django Rest Framework**.

Na imagem é possível observar que os medicamentos “**As Infantil**” (08:00) e “**Dipirona**” (09:00) aparecem exatamente como foram cadastrados anteriormente no banco de dados. Isso comprova que:

- A versão Web está totalmente integrada com o **mesmo backend** utilizado pelo aplicativo mobile.
- Os dados são compartilhados em tempo real, garantindo consistência entre diferentes plataformas.
- Alterações feitas na Web refletem no App e vice-versa, pois ambos consomem a mesma API.

Inclusão Digital

Essa tela também reforça o caráter de **inclusão digital do projeto**: mesmo que o usuário não tenha acesso a um smartphone moderno, ele pode gerenciar seus medicamentos pela versão Web, disponível em computadores de escolas, bibliotecas ou telecentros comunitários.



A imagem mostra a interface web do sistema de gerenciamento de medicamentos. No topo, há um formulário para cadastrar um novo medicamento, com campos para o nome, dias da semana (Dom, Seg, Ter, Qua, Qui, Sex, Sáb) e horários (1º a 5º Remédio). Abaixo do formulário, há uma seção intitulada 'Meus medicamentos' que exibe uma lista de medicamentos cadastrados. A lista contém dois itens: 'As Infantil' com horário 08:00 e 'Dipirona' com horário 09:00. Cada item na lista possui botões para editar e excluir.

Figura 17 - Página de controle de datas e horários de medicamentos

Considerações

O desenvolvimento do aplicativo "Monitor Saúde: Gerenciar e Melhorar o Seu Bem-Estar" representa um passo importante 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.

Conclusão

O desenvolvimento do projeto Monitor de Saúde representa uma contribuição significativa tanto do ponto de vista tecnológico quanto social. Do lado técnico, a solução integra três camadas fundamentais: um backend robusto construído em Django Rest Framework, responsável pela segurança, persistência e centralização dos dados; um aplicativo móvel em React Native, que possibilita ao usuário acompanhar seus indicadores de saúde de forma prática e direta; e uma aplicação web em React, que amplia ainda mais o alcance do sistema ao permitir acesso em computadores, escolas e telecentros.

Essa arquitetura integrada garante consistência dos dados, já que todas as plataformas consomem a mesma API e compartilham o mesmo banco, evitando duplicidade e assegurando que a experiência do usuário seja contínua, independentemente do dispositivo utilizado.

Além do aspecto técnico, o trabalho tem um forte caráter social, pois promove inclusão digital e contribui para a democratização do acesso à saúde. A possibilidade de cadastrar, consultar e acompanhar indicadores como pressão arterial, glicemia, colesterol e rotinas de medicamentos pode fazer diferença no cotidiano de pessoas que, muitas vezes, têm dificuldade em manter um acompanhamento sistemático de sua saúde. Ao oferecer uma versão web acessível em locais públicos, o projeto se torna especialmente útil para comunidades carentes, que nem sempre dispõem de celulares modernos ou de acesso constante a serviços de saúde.

Dessa forma, o trabalho não apenas atende aos requisitos acadêmicos e técnicos da disciplina, mas também reforça a importância da tecnologia como ferramenta de transformação social. Ele demonstra como uma solução computacional bem estruturada pode gerar impacto real, unindo inovação, acessibilidade e compromisso com a melhoria da qualidade de vida.

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>

Caso consiga hospedar a tempo:

- www.lintelecom.com.br