README - Plataforma de Agronegócio (Front-End)

Este documento descreve a arquitetura e as funcionalidades do front-end da nossa plataforma de agronegócio, desenvolvida para conectar agricultores e fornecedores de forma eficiente e intuitiva.

Visão Geral do Projeto

O front-end foi desenvolvido em **React Native**, um framework de código aberto que permite a criação de aplicativos móveis multiplataforma (iOS e Android) com uma única base de código. A escolha foi estratégica para reduzir custos e tempo de desenvolvimento, atendendo às necessidades do agronegócio, onde a agilidade é crucial.

A aplicação foi projetada para garantir um desempenho robusto, comparável a aplicativos nativos, mesmo em áreas rurais com baixa conectividade.

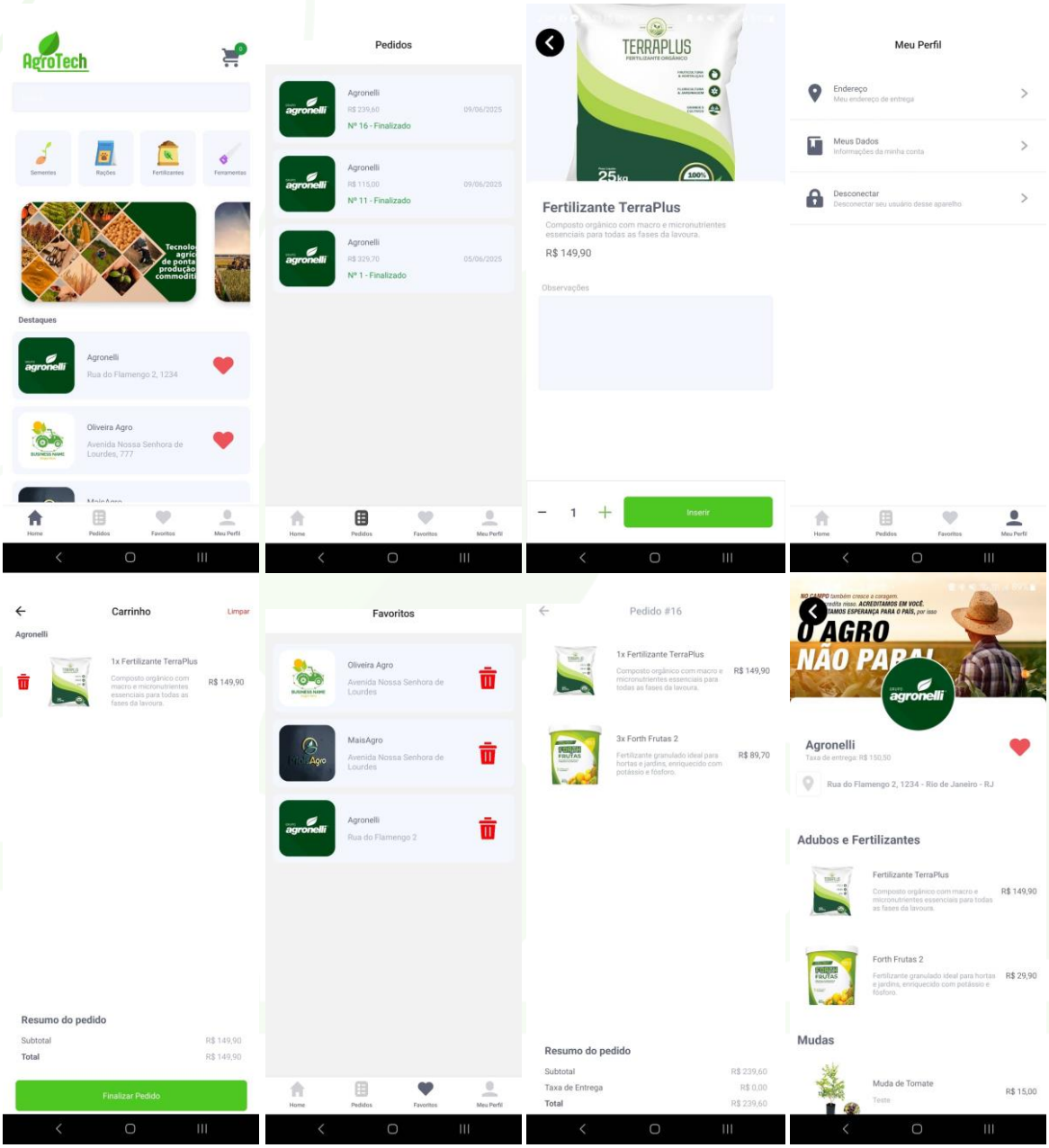
Tecnologias e Bibliotecas Principais

- **Framework:** React Native
- **Linguagem:** JavaScript
- **Navegação:** `@react-navigation/bottom-tabs` (para a barra de navegação principal) e React Navigation (para o gerenciamento geral de telas).
- **Comunicação com API:** `Axios` (para requisições ao back-end).
- **Gerenciamento de Estado:** Hooks nativos do React (`useState`, `useEffect`, `useContext`) para controle de estado local e global.
- **Armazenamento Local:** `AsyncStorage` para persistência de dados no dispositivo, garantindo funcionalidade offline.
- **Componentes de UI:** `FlatList` (para renderização otimizada de listas), `ScrollView`, `Modal`, `TextInput`, entre outros componentes nativos.
- **Estilização:** `StyleSheet` e `Flexbox` para criar layouts responsivos e consistentes.

Funcionalidades

A plataforma é dividida em duas áreas principais: a **Área do Cliente** (agricultor) e a **Área da Loja** (fornecedor).

Telas Cliente



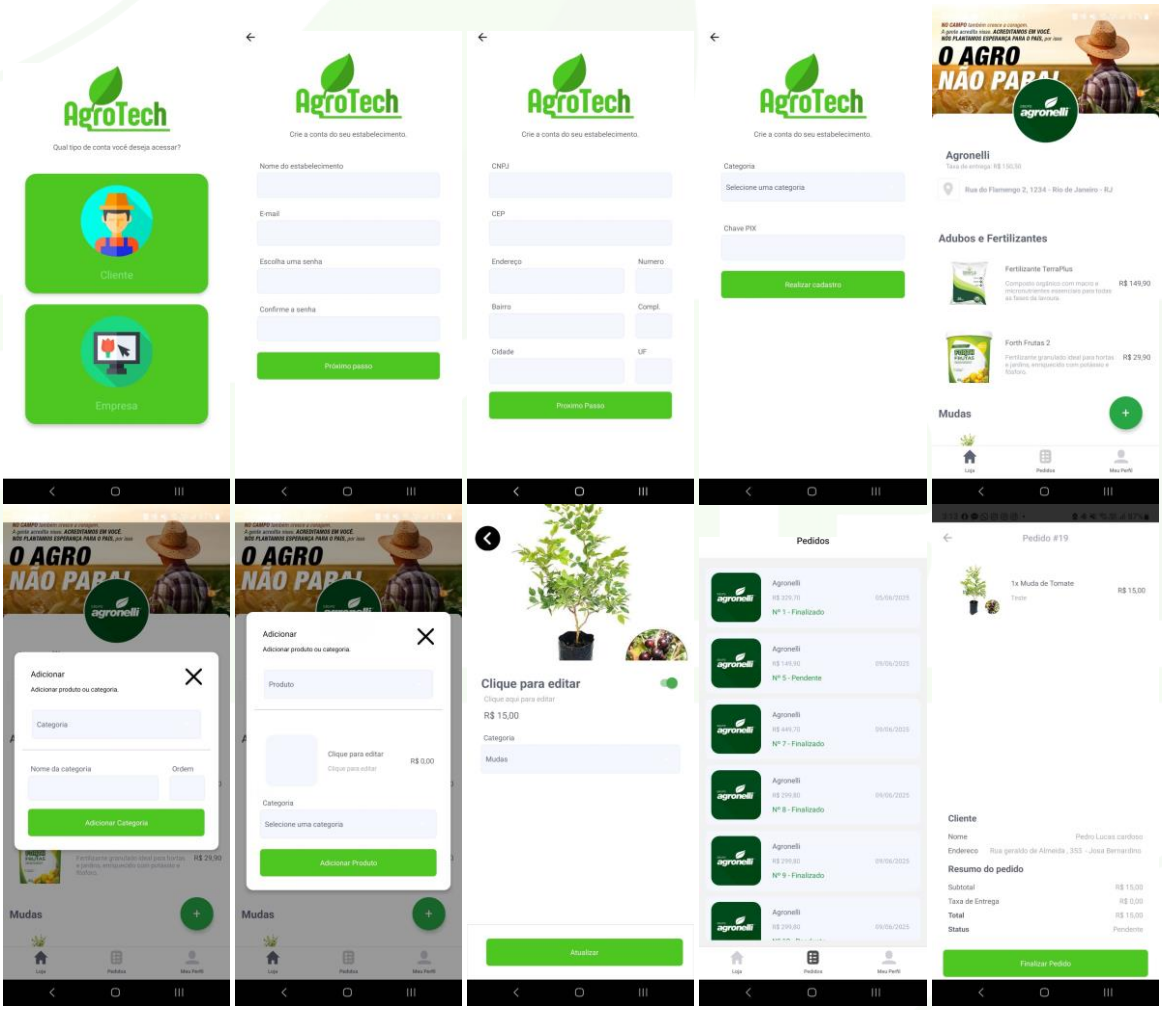
Área do Cliente

Interface focada na experiência de compra do agricultor.

- **Navegação Principal (Figura 1):** Uma `bottom tab bar` permite acesso rápido às telas de Início, Pedidos, Favoritos e Perfil.
- **Tela de Produtos (Figura 3):** Exibição dos produtos em formato de *cards*, com imagem, nome, preço e botão de compra. A tela é construída com `ScrollView` e `Flexbox` para garantir responsividade.
- **Carrinho de Compras (Figura 5):** Gerenciamento dos itens adicionados, com funcionalidades para ajustar quantidade ou remover produtos. O estado é gerenciado com `useState` e `useContext` para atualizações em tempo real.
- **Tela de Pedidos (Figura 2):** Lista o histórico de pedidos do usuário, consumindo dados via API e renderizando-os de forma eficiente com `FlatList`.
- **Tela de Favoritos (Figura 6):** Permite ao usuário salvar lojas favoritas para acesso rápido, fortalecendo o relacionamento com fornecedores.

- **Resumo do Pedido (Figura 7):** Apresenta um resumo claro da compra antes da finalização, incluindo produtos, subtotal, entrega e valor total.
- **Tela de Perfil (Figura 4):** Exibe os dados do usuário e permite o gerenciamento de informações e logout. Utiliza `AsyncStorage` para persistir os dados localmente.
- **Vitrine da Loja (Figura 8):** Permite ao cliente visualizar a página de um vendedor específico, com seus detalhes e lista de produtos.

Telas Loja



Área da Loja

Ferramentas completas para o fornecedor gerenciar sua vitrine e operações.

- **Seleção de Tipo de Conta (Figura 9):** Tela inicial onde o usuário escolhe se o acesso será como "Cliente" ou "Empresa".
- **Cadastro de Empresa (Figuras 10, 11 e 12):** Um fluxo de cadastro dividido em três etapas para coletar informações essenciais da empresa (dados básicos, endereço, categoria e chave Pix).
- **Painel da Loja (Figura 13):** Tela principal do fornecedor, exibindo informações da loja, vitrine de produtos organizados por categoria e um botão de ação flutuante (`FAB`) para adicionar novos produtos ou categorias.

- **Gerenciamento de Categorias (Figura 14):** Um modal permite ao fornecedor criar novas categorias para organizar seus produtos.
- **Gerenciamento de Produtos (Figuras 15 e 16):** Modais e telas dedicadas para adicionar novos produtos (com imagem, nome, descrição e preço) e editar produtos já existentes.
- **Resumo dos Pedidos (Figura 17, Figura 18):** Interface para o lojista visualizar os pedidos recebidos e os detalhes, incluindo informações do cliente e status da compra.

Destaques da Arquitetura

- **Foco em Desempenho:** A utilização do `FlatList` em vez do `ScrollView` para listas dinâmicas foi uma decisão consciente para otimizar o uso de memória e garantir fluidez, especialmente em dispositivos de baixo custo.
- **Experiência do Usuário (UX) no Contexto Rural:** O design da interface prioriza a simplicidade, clareza e acessibilidade, com botões grandes e fluxos intuitivos, considerando as necessidades de usuários em ambientes rurais.
- **Resiliência à Conectividade:** O uso de `AsyncStorage` para persistir dados localmente reduz a dependência de uma conexão constante com a internet, um fator crítico para o público-alvo.
- **Gerenciamento de Estado Escalável:** O `useContext` é utilizado para gerenciar o estado global (como o carrinho de compras), promovendo consistência de dados em toda a aplicação e facilitando a escalabilidade.

Referências Citadas no Desenvolvimento

O desenvolvimento seguiu boas práticas e foi embasado em estudos e fontes reconhecidas na área, conforme citado no documento original de desenvolvimento: Bolfe (2020), Deponti (2024), DIO (2024), Escudelario; Pinho (2021), Fagundes (2023), Gonçalves (2025) e Oliveira et al. (2022).