

Documentação de Desenvolvimento: DouroVinhos Mobile (MVP)

Projeto: DouroVinhos Mobile Sales **Versão:** 1.0 (MVP - Demo Portfolio)

Responsável Funcional: Jaqueline Daniel **Lead Developer:** Dennis

(SquadHomeDev) **Data:** 20 de Janeiro de 2026

1. Visão Geral e Contexto

O projeto nasceu da necessidade crítica de digitalizar a força de vendas da **DouroVinhos Premium Lda.**, que sofria com ruturas de stock e perda de vendas devido a processos manuais (papel/WhatsApp).

O Desafio: Criar uma aplicação móvel em **3 meses** que funcione **Offline-First** (sem internet), permitindo aos vendedores tirar pedidos nas caves dos clientes e validar regras de negócio críticas na hora.

2. Arquitetura Técnica Implementada

Para viabilizar a demonstração funcional e o MVP rápido, adotamos uma arquitetura moderna baseada em componentes desacoplados.

Stack Tecnológica

- **Frontend:** React Native com **Expo** (Managed Workflow) para agilidade máxima.
- **Linguagem:** TypeScript/JavaScript (ES6+).
- **Estilização:** **NativeWind (TailwindCSS)** – garantindo identidade visual "Premium" (Vinho/Dourado) e rapidez na UI.
- **Gerenciamento de Estado:** **Zustand** – utilizado para gerir o Carrinho de Compras e a Sessão do Cliente de forma global e performática.
- **Dados (Estratégia Mock):** Substituição temporária do SQLite por **Constantes JSON Estáticas** (MOCK_CLIENTS, MOCK_PRODUCTS) para simular o comportamento do banco de dados e garantir estabilidade na apresentação.

Estrutura do Projeto

O código foi organizado seguindo boas práticas de "Feature-based structure":

- `/app`: Rotas e Telas (Expo Router) - *Catálogo, Carrinho, Ajustes*.

- /components: Blocos reutilizáveis de UI - *ProductCard*, *CartItem*, *CheckoutModal*.
 - /store: Lógica de negócio pura ("Cérebro") - *cart-store.js*.
 - /constants: Fonte de verdade dos dados (Simulando o ERP PHC).
-

3. Implementação das Regras de Negócio (User Stories)

Abaixo detalhamos como cada requisito funcional crítico extraído do **Relatório de Discovery** foi traduzido em código.

1. Regra de Bonificação "12+1" (Upselling)

- **O Requisito:** "Na compra de 12 garrafas, o cliente recebe 1 extra grátis. A App deve sugerir isto."
- **A Implementação Técnica:**
 - No arquivo *cart-store.js*, a lógica de cálculo monitora a quantidade.
 - No componente *CartItem.tsx*, implementamos uma verificação visual: `if (quantity >= 12 && hasBonus)`.
 - **Resultado Visual:** Uma faixa verde "Oferta 12+1 Aplicada" aparece automaticamente, incentivando o vendedor a manter o volume.

2. Bloqueio de Crédito (Edge Validation)

- **O Requisito:** "Proteção de Preço" e validação financeira antes do envio para evitar "lixo" no servidor.
- **A Implementação Técnica:**
 - Criamos clientes com flags específicas nos dados mockados: "Solar do Douro" (Ativo) e "Café Central" (`isBlocked: true`).
 - No momento do "Finalizar Pedido" (*cart.tsx*), o sistema intercepta a ação.
 - Se `isBlocked === true`, disparamos um **Modal de Bloqueio** (Vermelho) em vez de enviar o pedido.
 - **Benefício:** Cumpre o requisito de "Edge Validation" descrito na Arquitetura.

3. Experiência Offline-First

- **O Requisito:** "Tem de funcionar mesmo se a internet cair, porque nas caves dos clientes o sinal é horrível."
 - **A Implementação Técnica:**
 - O App carrega todo o catálogo e lista de clientes instantaneamente a partir dos arquivos locais (constants), sem fazer requisições de rede (API Calls) durante a navegação.
 - Isto simula a experiência de um banco de dados local (Cache) que foi sincronizado pela manhã, garantindo zero latência.
-

4. Fluxo de Processo (BPMN)

O App cobre a raia "APP Vendedor (Frontend)" do diagrama de processos desenhado.

1. **Pedido Iniciado:** O vendedor abre o App (Tela Home).
 2. **Preencher Carrinho:** Seleciona produtos e ajusta quantidades (Tela Catálogo -> Carrinho).
 3. **Solicitar Finalização:** Clica em "Finalizar Pedido".
 4. **Gateway de Decisão (Local):** O código verifica isBlocked?
 - *Caminho Não:* Exibe Modal de Bloqueio (Fim do fluxo com erro).
 - *Caminho Sim:* Exibe Modal de Sucesso (Simula "Salvar Localmente SQLite" e "Aguardar Sincronização").
-

5. Design e UX (Polimento)


Atendendo à solicitação de interfaces para "vendedores seniores":

- **Contraste Alto:** Fundo claro, textos escuros e botões de ação primária em "Vinho Tinto" (Cor da Marca).
 - **Botões Grandes:** Áreas de toque aumentadas (p-4, h-12) para facilitar o uso em movimento.
 - **Feedback Visual:** Uso de ícones (Check Verde, Alerta Vermelho) e Modais para confirmar ações, sem deixar o utilizador na dúvida.
-

6. Próximos Passos (Roadmap Pós-MVP)

Conforme o plano de Handoff:

- Integração real com SQLite (expo-sqlite) para persistência de dados.
- Implementação do "Sync Engine" para conectar com as tabelas SQL do Rogério (TI).
- Módulo de Recolhas e Cobranças (Sprint 6/7).

Status Final:  **MVP Aprovado para Demonstração.** O aplicativo cumpre todos os critérios visuais e lógicos para provar a viabilidade do projeto junto aos stakeholders.