

1. ÁREA PÚBLICA (Cliente)

1.1 Home & Navegação

- **Header fixo:** Logo, Menu Categorias, Busca, Carrinho (contador), Login
- **Grid de produtos:** Últimos 20 cadastrados (renderizado via JS)
- **Footer:** Links institucionais, redes sociais, WhatsApp

1.2 Catálogo

- Grid com foto, nome, preço
- Filtro por categoria (dropdown - fetch dinâmico)
- Busca por nome (debounce pra não sobrecarregar API)

1.3 Página de Produto

- Galeria de imagens (1ª é principal) - trocar imagem ao clicar
- Nome, preço, descrição
- Seletor de tamanho (obrigatório) - botões JavaScript
- Botão "Adicionar ao carrinho" (desativado se `stock == 0` com label "Esgotado")
- **Interação:** Adicionar → Animação toast → Contador carrinho atualiza

1.4 Carrinho

- **Persistência:** `localStorage` (visitante) | Sincroniza com API (logado)
- Lista dinâmica via JS: foto, nome, tamanho, preço, quantidade (+/-), remover
- **Cálculos em tempo real:** Subtotal e Total atualizados ao mudar quantidade
- Resumo: Subtotal, "Frete a calcular", Total
- Botão "Finalizar compra"

1.5 Checkout (Tela Única - SPA Style)

Seção 1 - Identificação

- Se não logado: Formulário → Validação JS antes de enviar
- Nome, Email, CPF (máscara JS), Tel (máscara WhatsApp), Senha
- Checkbox "Criar conta"

Seção 2 - Endereço

- Campo CEP → **Busca automática ViaCEP via fetch**
- Preenche automaticamente: Rua, Bairro, Cidade, Estado
- Número e Complemento editáveis

Seção 3 - Resumo

- Produtos renderizados via JS
- Cálculos dinâmicos, Observações (textarea)

Seção 4 - Finalizar

- Checkbox termos, Botão "Enviar Pedido"
- **Loading spinner** durante processamento
- **Validação crítica:** Double-check de estoque via API antes de confirmar

1.6 Confirmação

- Página de sucesso renderizada via JS
- Número do pedido, resumo
- Botão WhatsApp (abre direto `wa.me / . . .`)

1.7 Login/Cadastro

- **Modal ou página separada**
- Login: Email, Senha → Fetch API → Salva JWT no `localStorage`
- Cadastro: Validação em tempo real (email já existe? senha fraca?)
- Recuperar senha: Email → API envia link

1.8 Minha Conta

- **SPA com navegação via JS** (sem reload de página)
- **Meus Pedidos:** Tabela dinâmica, filtros via fetch
- **Detalhes do Pedido:** Modal ou página, dados carregados via API
- **Meus Dados:** Formulário editável, salva via PUT request

1.9 Páginas Institucionais

- HTML estático ou renderizado via JS
 - Sobre, Contato (botão WhatsApp clicável), Termos, Política
-

2. ÁREA ADMINISTRATIVA (Dashboard Web)

2.1 Dashboard

- **Interface web SPA** (React Admin, Vue, ou JS puro)
- Cards: Pedidos novos, Faturamento do mês, Estoque baixo (<5)
- Últimos 10 pedidos (tabela resumida)

2.2 Gestão de Produtos

Listar: Tabela com filtros e ordenação via JS

Cadastrar/Editar:

- Formulário com validação JS
- **Upload de imagens:** Drag & drop com preview (1-5 imagens, 1ª = principal)
- Sistema de ordenação de imagens (arrastar pra reordenar)
- Tamanhos dinâmicos: Checkboxes (P, M, G, GG) com campos de estoque individual
- Status ativo/inativo

Deletar: Modal de confirmação → Soft delete (`deleted_at`)

2.3 Gestão de Categorias

- CRUD simples: Nome, descrição
- Tabela ou modal

2.4 Gestão de Pedidos

Listar:

- Tabela com filtros (status, data, cliente)
- Badges coloridos pra status

Detalhes do Pedido:

- Info cliente (nome, email, tel com **botão WhatsApp direto**, CPF)
- Produtos (foto, nome, tamanho, qtd, preço)
- Endereço completo
- **Frete editável** (admin calcula externamente e preenche)
- Status (dropdown): Novo → Confirmado → Enviado → Entregue | Cancelado
- Código de rastreio (campo texto)
- Observações do cliente + Notas internas (admin)

- Histórico de alterações (timeline)

Fluxo manual:

1. Pedido chega (status "Novo")
2. Admin calcula frete externo
3. Preenche valor no painel
4. Muda status → "Aguardando Pagamento"
5. Cliente paga (Pix manual via WhatsApp)
6. Admin confirma → status "Confirmado"
7. Separa/envia → status "Enviado" + código rastreio

2.5 Configurações

- Dados da loja: Nome, email, tel, WhatsApp, logo
 - Upload de logo com preview
 - Email admin (notificações)
 - Redes sociais (opcional)
-

3. FLUXOS COMPLETOS DO SISTEMA

FLUXO 1: Compra Completa (Happy Path)

CLIENTE:

1. Acessa site
2. Navega pelo catálogo (ou busca)
3. Clica em produto
4. Seleciona tamanho
5. Clica "Adicionar ao carrinho"
 - Toast de confirmação aparece
 - Contador do carrinho atualiza
6. Continua comprando (adiciona mais produtos)
7. Clica no ícone do carrinho
8. Revisa produtos (ajusta quantidade, remove itens)
9. Clica "Finalizar compra"
10. Sistema verifica login:
 - **NÃO logado** → Mostra formulário de identificação

- **Logado** → Pula para endereço

11. Preenche dados pessoais (se não logado)

- Marca checkbox "Criar conta" (opcional)

12. Preenche CEP

- Sistema busca endereço automaticamente (ViaCEP)
- Preenche Rua, Bairro, Cidade, Estado

13. Completa com Número e Complemento

14. Revisa resumo do pedido

15. Marca checkbox de aceite dos termos

16. Clica "Enviar Pedido"

- Loading spinner aparece

BACKEND (Processamento): 17. Valida dados do formulário 18. **Double-check de estoque** (validação crítica) 19. Se novo usuário → Cria registro na tabela `users` 20. Cria pedido na tabela `orders` (status: "Novo") 21. Cria itens do pedido em `order_items` (com preço congelado) 22. **Diminui estoque** (transação atômica com criação do pedido) 23. Limpa carrinho (localStorage ou banco) 24. Envia email de confirmação pro cliente 25. Envia email de notificação pro admin 26. Retorna resposta de sucesso com número do pedido

FLUXO 2: Cliente Já Cadastrado

1. Cliente acessa site
 2. Clica em "Login"
 3. Preenche email e senha
 4. Sistema valida e cria sessão JWT
 5. Redireciona pra home (ou página anterior)
 6. Navega e adiciona produtos ao carrinho
 - Carrinho sincroniza automaticamente com o banco
 7. Vai pro checkout
 8. **Dados pessoais já preenchidos** (só revisa)
 9. **Endereço já preenchido** (pode editar ou usar salvo)
 10. Finaliza pedido
 11. Resto do fluxo igual ao Fluxo 1 (a partir do passo 16)
-

FLUXO 3: Produto Esgotado

1. Cliente navega pelo catálogo
2. Clica em produto
3. Sistema verifica estoque do tamanho selecionado
4. **Estoque = 0** para aquele tamanho:
 - Botão "Adicionar ao carrinho" fica **desabilitado** (disabled)
 - Label muda para "Esgotado"
 - Pode mostrar badge vermelho "Sem estoque"
5. Cliente **não consegue adicionar** ao carrinho
6. Pode selecionar outro tamanho (se disponível)

Variação - Todos os tamanhos esgotados:

- Badge "Produto Esgotado" aparece na galeria
 - Mensagem: "Este produto está temporariamente indisponível"
 - Botão "Avise-me quando chegar" (opcional - v2)
-

FLUXO 4: Carrinho Abandonado

Visitante (não logado):

1. Cliente adiciona produtos ao carrinho
2. Navega pelo site
3. Fecha navegador SEM finalizar
4. **Carrinho salvo no localStorage**
5. Cliente volta depois (mesmo navegador)
6. Carrinho ainda está lá
7. Pode continuar a compra normalmente

Cliente logado:

1. Cliente adiciona produtos ao carrinho
2. Carrinho sincroniza com o banco via API
3. Fecha navegador
4. Acessa de **outro dispositivo**
5. Faz login
6. **Carrinho sincronizado** aparece
7. Pode continuar de onde parou

Nota: Email de "Esqueceu algo no carrinho?" fica pra versão 2.0

FLUXO 5: Cancelamento de Pedido

CLIENTE solicita (via WhatsApp):

1. Cliente envia mensagem pedindo cancelamento
2. Admin recebe solicitação

ADMIN cancela: 3. Admin acessa painel 4. Vai em "Gestão de Pedidos" 5. Busca o pedido do cliente 6. Clica em "Ver detalhes" 7. Muda status para "Cancelado" (dropdown) 8. Adiciona observação (motivo do cancelamento) 9. Clica "Salvar"

BACKEND processa: 10. Marca pedido como cancelado no banco 11. **Devolve produtos pro estoque** (reverte a baixa) 12. Envia email automático de cancelamento pro cliente 13. Registra no histórico do pedido

Se cliente já pagou:

- Admin combina devolução via WhatsApp
- Anota nos campos internos do pedido

FLUXO 6: Alteração de Estoque

VENDA (automático):

1. Cliente finaliza pedido
2. Sistema cria pedido + order_items
3. **Diminui estoque automaticamente** na mesma transação
 - Ex: Produto X, tamanho M tinha 10 → agora tem 9
4. Se falhar em alguma etapa, **rollback completo** (nada é salvo)

ADMIN ajusta manualmente:

1. Admin entra em "Gestão de Produtos"
2. Clica em "Editar" no produto desejado
3. Altera quantidade de estoque de um tamanho
 - Ex: Tamanho G: 5 → 20 (recebeu reposição)
4. Clica "Salvar"
5. Sistema registra no log:
 - `admin_id`, `timestamp`, `produto_id`, `tamanho`, `estoque_anterior`, `estoque_novo`

6. Estoque atualizado imediatamente

CANCELAMENTO (automático):

1. Admin cancela pedido (Fluxo 5)
2. Sistema identifica produtos do pedido
3. **Devolve quantidades pro estoque**
 - Ex: Produto X, tamanho M estava 9 → volta pra 10
4. Registra no log de alterações

CORREÇÃO de erro:

1. Admin percebe discrepância de estoque
 2. Acessa logs de alteração
 3. Verifica histórico:
 - Quem alterou
 - Quando alterou
 - Valores antes/depois
 4. Faz ajuste manual se necessário
-

FLUXO 7: Recuperação de Senha

CLIENTE esqueceu a senha:

1. Cliente vai pra tela de login
2. Clica em "Esqueci minha senha"
3. Página de recuperação abre
4. Cliente digita seu email
5. Clica "Enviar link de recuperação"

BACKEND: 6. Valida se email existe no banco

- **NÃO existe** → Mostra mensagem genérica (segurança)
 - **Existe** → Continua
7. Gera token temporário único
 8. Salva token no banco (tabela `password_resets` com expiração 1h)
 9. Envia email com link:
 - `https://seusite.com/redefinir-senha?token=abc123xyz`

CLIENTE: 10. Recebe email 11. Clica no link 12. Página de redefinição abre 13. Sistema valida token: - **Expirado** → Mostra erro "Link expirado, solicite novamente" - **Inválido** → Mostra erro "Link inválido" - **Válido** → Mostra formulário 14. Cliente digita nova senha 15. Cliente confirma nova senha 16. Clica "Redefinir senha"

BACKEND: 17. Valida senhas (iguais? mínimo 6 caracteres?) 18. Criptografa nova senha (BCrypt) 19. Atualiza senha no banco 20. **Invalida o token** (não pode ser usado de novo) 21. Retorna sucesso

CLIENTE: 22. Vê mensagem "Senha alterada com sucesso!" 23. Redireciona pra tela de login 24. Faz login com nova senha

FLUXO 8: Login Admin

1. Admin acessa `/admin` (URL direta)
2. Sistema verifica se há sessão ativa:
 - **SIM** → Redireciona pro dashboard
 - **NÃO** → Mostra tela de login
3. Admin digita email e senha
4. Clica "Entrar"
5. Sistema valida credenciais
6. **Credenciais inválidas:**
 - Mostra erro "Email ou senha incorretos"
 - Não especifica qual está errado (segurança)
 - Limita tentativas (rate limiting)
7. **Credenciais válidas:**
 - Cria sessão JWT
 - Salva token no localStorage
 - Redireciona pro dashboard
8. Admin vê painel administrativo completo

Sessão expira:

- Após X horas de inatividade
- Sistema detecta token expirado
- Redireciona pro login automaticamente
- Mostra mensagem "Sessão expirada, faça login novamente"

