🚀 Kazen: Plano Técnico para Startup Challenge (MVP)

Foco: Velocidade, Impacto Visual e Estabilidade na Demo. **Estratégia:** "Demo-Driven Development"

1. Tech Stack (Otimizada para Velocidade de Desenvolvimento)

Para um challenge, precisamos de zero configuração e resultados visuais imediatos.

Componente	Tecnologia	Justificativa "Challenge Mode"
Framework	**Next.js 14 (App Router)**	Rápido de iniciar, routing fácil, e permite criar APIs "falsas" (Mock API) dentro do próprio projeto se o backend atrasar.
Linguagem	**TypeScript**	Evita bugs estúpidos de última hora durante a demo (ex: `undefined is not a function` no palco).
Styling	**Tailwind CSS + Shadcn/UI**	**Crítico:** Usa componentes prontos (Shadcn) para parecer profissional sem desenhar do zero. Já temos a paleta `kazen_color_palette.js` .
Backend/DB	**Supabase (Baas)**	Zero setup de servidor. O banco Postgres real dá credibilidade técnica se os juízes perguntarem.
Imagens	**Cloudinary (ou Public Folder)**	Não compliques com S3. Para o MVP, mete as 50 imagens de produtos na pasta `/public` do Next.js para carregamento instantâneo e zero latência.
PWA	**Vite PWA Plugin / Serwist**	Cache agressivo. A app deve abrir instantaneamente.
Mapas	**Imagem Estática / Leaflet**	**Truque:** Não gastes tempo com API do Google Maps agora. Usa uma biblioteca leve como `Leaflet` ou até uma imagem estática interativa para simular a funcionalidade de mapa sem o risco de custos ou erros de API key.

2. Arquitetura de Dados (Simplificada)

Para o challenge, não precisamos de um sistema complexo de gestão de preços históricos. Precisamos de um snapshot do "agora".

Esquema da Base de Dados ("Seed Data" Orientado)

Em vez de criar um painel de admin para inserir produtos, vamos criar um **Script de Seed** robusto que limpa e repopula a base de dados com dados perfeitos para a demo.

```
erDiagram
    PRODUCT {
        string id PK
        string name "Ex: Picanha Fresca"
```

```
    string image_url "Local asset path"
    string category "Churrasco"
    string brand
}

STORE {
    string id PK
    string name "Ex: Kero Talatona"
    string logo_url
    string color_hex
}

%% A tabela mágica para a demo
PRICE_SNAPSHOT {
    string id PK
    string product_id FK
    string store_id FK
    float price "Preço fixo para a demo"
    boolean is_promo "Para mostrar UI de desconto"
    boolean in_stock "Para testar fluxo de substituição"
}

SHOPPING_LIST {
    string id PK
    jsonb items "Array de produtos selecionados"
    float total_saved "Cálculo fake de poupança"
}
```

Nota Tática:

- Não normalizes demasiado.
- Não faças scraping real. Inventa os preços para garantir que a **Loja A** seja sempre mais barata que a **Loja B** na tua demo, para provar o valor da poupança inequivocamente.

3. Estratégia PWA & "Fake-Offline"

Numa competição, a performance percebida é tudo.

1. **Estratégia de Assets:**
   - Coloca todos os ícones, logos e imagens de produtos na pasta `public/` .
   - Configura o `next/image` para priorizar o carregamento (`priority={true}`) das imagens da "Homepage".
2. **Manifesto Robusto:**
   - Nome curto: "Kazen".
   - `display: standalone` (remove a barra de URL, parece nativo).
   - `theme_color : #14B8A6` (Brand Primary).
   - Ícones gerados para iOS e Android (use `maskable`).
3. **O Truque do "Loading Skeleton":**

- Nunca mostres um "spinner" branco. Desenha esqueletos (rectângulos cinzentos a pulsar) que imitam exatamente o layout dos produtos. Isso faz a app parecer 2x mais rápida.

4. Roadmap de Implementação ("Sprint de 7 Dias")

Dias 1-2: O "Palco" (Setup & UI)

- [ ] Setup Next.js + Tailwind + Shadcn.
- [ ] Configurar a `kazen_color_palette.js` no tailwind.
- [ ] **Criar a Componentes Chave:**
  - Card de Produto (Imagem grande, Preço destaque, Preço comparação pequeno).
  - Barra de Navegação Mobile (Bottom Tab Bar).
  - Header com saudações ("Olá, Willfredy").

Dias 3-4: O "Ator" (Dados & Lógica)

- [ ] **Criar o "Golden Dataset":**
  - Escolher 1 Categoria (ex: "Churrasco de Domingo").
  - Selecionar 10 Produtos chave.
  - Criar JSON estático com preços para 3 lojas (Loja A Barata, Loja B Cara, Loja C Média).
- [ ] Implementar a lógica de comparação (Frontend only): Somar os preços do JSON e mostrar a diferença.

Dia 5: O "Clímax" (Fluxo Principal)

- [ ] **Tela de Lista:** O utilizador clica nos produtos sugeridos.
- [ ] **Tela de Comparação:** Animação de barras a crescer mostrando o preço total em cada loja.
- [ ] **Tela de "Checkout":** Um botão "Reservar no Kero" que abre um modal de sucesso com confetes (biblioteca `canvas-confetti`). *Não implementes pagamentos reais.*

Dias 6-7: O "Polimento" (UX & PWA)

- [ ] Configurar PWA (Manifest).
- [ ] Testar instalação no telemóvel.
- [ ] Adicionar micro-interações (feedback tátil ao clicar em botões).
- [ ] **Ensaio Geral:** Fazer a demo completa 10 vezes seguidas para garantir que nada quebra.

5. Funcionalidades "Mockadas" (Não Implementar de Verdade)

Para poupar tempo e reduzir risco, vamos simular estas partes:

- **Login:** Não faças autenticação real complexa. Faz um login "fake" ou um botão "Entrar como Convidado" que já loga num perfil pré-configurado com foto e nome.
- **Pagamento:** O botão "Pagar com MCX" deve apenas mostrar um "loading" de 2 segundos e depois "Sucesso!".
- **Mapa:** Usa uma imagem de um mapa de Luanda com pinos desenhados em cima. Ao clicar, abre um modal com os detalhes da loja.

6. Diferenciais para o Júri (O "Secret Sauce")

1. **Onboarding Incrível:** Uma tela inicial animada que explica o valor em 3 slides ("Poupe Tempo", "Poupe Kwanzas", "Compre Melhor").
2. **Modo Escuro:** Se implementares com Tailwind (`dark:`), ganhas pontos de design e modernidade.
3. **QR Code:** No final da apresentação, mostra um QR Code para os juízes testarem a app no