Plan 2 — Technical Architecture Plan (Full)

Plan 2 — Technical Architecture Plan (Codex-ready)

1) Stack Decision (opinionated)

Recommendation:

Next.js (App Router) + TypeScriptTailwind CSS + CSS variables (design tokens)shadcn/ui (primitives only: accordions, tabs, dialogs)Framer Motion (selective)Content source v1: Local structured data (TS/JSON), not CMSDeployment: VercelForms: Server Actions (or API route) → email + optional CRM webhookAnalytics: Plausible or GA4 (minimal, high-signal events)

Why not CMS in v1:

CMS slows down early velocity and causes “content drift”.System-led, template-driven site is ideal for data-in-repo.Add CMS later when template is stable and updates are frequent.

2) Rendering Strategy

SSG by default (static pages + product pages)SSR only where needed (RFQ submission, optional query parsing)

Static:

Home, System, Quality, About, ContactProducts index (static + client filters)Product pages generated from dataset at build time

Benefits:

Best performance, lowest complexityLow runtime failure riskStrong caching by default

3) Routes (App Router)

app/page.tsx → Homeapp/products/page.tsx → Products indexapp/products/[slug]/page.tsx → Product detail templateapp/system/page.tsxapp/quality/page.tsxapp/about/page.tsxapp/contact/page.tsx

Optional Tier B:

app/logistics/page.tsxapp/traceability/page.tsxapp/brands/page.tsxapp/faq/page.tsxapp/privacy/page.tsxapp/terms/page.tsx

4) Data Model (Source of Truth)

4.1 Product schema (TypeScript)

File: src/data/products.ts exporting products: Product[]

Minimum fields:

slug: stringname: stringcategory: 'fruit' | 'veg'subCategory?: stringoneLiner: stringheroImage: ImageRefgallery: ImageRef[]quickSpecs: { label: string; value: string }[]pillars3: { title: string; body: string }[]grades: { grade: string; size: string; count?: string; notes?: string }[]packagingOptions: { name: string; netWeight?: string; count?: string; carton?: string; notes?: string }[]seasonality: { startMonth: number; endMonth: number; notes?: string }originRegions: string[]logistics?: { transitNotes?: string; storage?: string; reefer?: string }rfqDefaults?: { packagingPreference?: string }

4.2 Brand schema

File: src/data/brands.ts

name, logo, oneLiner, proofPoints[]

ImageRef strategy

Store images in public/images/...Reference as { src: '/images/products/watermelon/hero.jpg', alt: '...' }

5) Component Architecture

Layout:

SiteHeader, SiteFooter, Section, Container, SEO helpers

Home sections:

HeroSystem, BrandsProofModule, ProductCategoryGrid, SystemNarrativeStackProcessStepper, QualityPillars, TraceabilityTeaser, RFQForm

Product template sections:

ProductHero, SpecChips, Pillars3, SpecsTablePackagingCardGrid, MiniProcessFlow, TransitInfoPanelStickyProductCTA (mobile)

Hard rule:

No product-specific layout code. All products render from same sections.

6) Styling System (tokens)

src/styles/tokens.css defines CSS variables:colors (bg, surface, text, muted, accent, border)spacing scale, radius scale, shadowsTailwind theme reads tokens via theme.extend

Purpose:

Keeps modern structured feel without class chaos.Enables later brand/theme adjustments cleanly.

7) Imagery & Performance Rules (non-negotiable)

Images:

next/image everywhereexplicit width/heightpriority only for above-fold herolazy-load galleryavoid huge originals; pre-export sizes

Video:

only if it materially improves storymuted, short, optimized, with fallback imageremove if it hurts performance

Performance targets:

LCP < 2.5s (mobile baseline target)CLS ~ 0keep JS budget tight; selective motion

8) RFQ Form Handling (conversion backbone)

v1 approach:

Server Action or /api/rfqEmail provider: Resend or Postmark or SendGrid (choose one)Optional CRM webhook later

RFQ email must include:

product context (if from product page)user info + company + marketquantity + timeframe + packaging preference

Anti-spam:

honeypotbasic rate limitzod validation

9) SEO + Metadata

Static metadata per pageProduct metadata template: “{Product} Exporter | Company”OG defaults + optional per product laterSitemap includes product slugsrobots.txt standard

10) Analytics (minimal)

Track:

rfq_submitwhatsapp_clickproduct_viewfilter_usedcontact_click_email

11) Repo Structure (Codex must follow)

```
src/
app/
layout.tsx
page.tsx
products/
page.tsx
[slug]/
page.tsx
system/page.tsx
quality/page.tsx
about/page.tsx
contact/page.tsx
components/
layout/
sections/
product/
ui/
seo/
data/
products.ts
brands.ts
lib/
constants.ts
utils.ts
validators.ts
styles/
tokens.css
public/
images/
products/
brands/
```

12) Build Constraints

No CMS in v1 unless explicitly switched.No product-specific components.Section spacing via Section wrapper.No unverified certification claims.Forms working before polish animations.

13) Milestones

M1: Repo + tokens + layout + sample product (watermelon)

M2: Product template + products index + filters

M3: System + Quality + About pages

M4: RFQ end-to-end + WhatsApp + analytics hooks