Plan 4 — Content & Data Loading Plan (Full)

Plan 4 — Content & Data Loading Plan (SOP)

Goal of Plan 4

Ensure content enters the system without chaos:

consistent product onboardingconsistent imageryno drift from canonical templateeasy additions later without developer rewrites

Content Source of Truth (v1)

Products and brands live in repo (TypeScript/JSON).One schema for all products.No “random content in components.”

Product Onboarding Workflow

Step 1: Create product entry

Add product object to src/data/products.tsMust include required fields:slug, name, category, oneLiner, heroImage, galleryquickSpecs, pillars3, grades, packagingOptionsseasonality, originRegionslogistics optional

Step 2: Add imagery to /public

Folder pattern:

public/images/products/{slug}/hero.jpg01.jpg, 02.jpg, 03.jpg (gallery)pack-01.jpg, pack-02.jpg (if needed)

Step 3: Validate

Build passes TypeScript checksProduct page renders with zero layout editsNo missing alt text

Step 4: Review checklist (buyer-facing)

Are claims factual and verifiable?Are specs procurement-ready?Does packaging match buyer mental model?

Copy Rules (procurement-friendly)

Prefer measurable facts (size, count, pack, season window).Avoid “best / world-class / premium” unless supported.Use short paragraphs, structured bullets, tables where needed.Every “trust claim” must map to a process step or proof.

Imagery Rules (strategy aligned)

Abstract system visuals first (home/system hero).Reality photos used as proof tied to steps (farm, sorting, packing, cold chain).No random lifestyle photos.Each image must have:filename conventionalt textplacement rationale (which section/step it supports)

Image Optimization SOP

Export sizes:Hero: ~2000px wideGallery: ~1400px wideCompress with modern codecs (WebP/AVIF if possible)Avoid uploading huge originals to repoAlways use next/image

Ownership & Roles

Content owner: decides copy/specs (procurement correctness)Brand owner: approves tone and proof modulesDev owner: ensures schema correctness, build integrityQA owner: checks rendering, mobile, performance

Release Process

Add/modify product dataAdd imagesRun local checksDeploy previewApprovePromote to production

When to introduce a CMS (later)

CMS is justified only when:

product template is stable for weeksfrequent updates are needed (weekly)a non-technical owner will maintain contentyou have enough content volume to justify workflow overhead

Until then:

repo data is faster, safer, and keeps system consistency.