AXIARA.ID — Step-by-Step Implementation Guide

For Google Antigravity IDE with 3-Layer Agent Architecture

Version: 2.0 (Revised) | **Date:** February 2026 **Project:** axiara.id Website **Stack:** HTML + Tailwind CSS → deployed on WordPress + BeTheme **Agent Architecture:** 3-Layer (Directives → Orchestration → Execution) **Agent Workflow:** Setup → /plan → Design System → /create → Deploy

---

TABLE OF CONTENTS

---

1. How the 3-Layer Architecture Applies to This Project

Before touching any code or giving any prompts, you need to understand how your agent system works. The `AGENTS.md` file that is already loaded into your Antigravity environment defines a 3-layer architecture that governs everything the agent does. Every instruction in this guide is built around that architecture. Here is how each layer maps to the Axiara website project.

Layer 1: Directives (What to do)

Directives are Standard Operating Procedures written in Markdown. They live in the `directives/` folder. For this project, you will create directives that tell the agent exactly how to build each type of page, how to apply the Axiara brand guidelines, how to run the design system generator, and how to deploy the final output. Think of directives as instruction manuals you would give to a reliable mid-level developer — they define goals, inputs, tools to use, outputs, and edge cases.

Why this matters for Axiara: The Axiara brand has extremely strict visual rules (no bouncy animations, 0px border-radius everywhere, specific font hierarchy, Crimson left-border-only cards). If these rules only live in your head or in a single prompt, the agent will drift from them over the course of 15+ pages. By encoding them into a directive, the agent reads them every time it builds a page, ensuring consistency from the homepage to the last compliance page.

Layer 2: Orchestration (Decision making)

This is the AI agent itself. Its job is to read the relevant directive, decide what to do, call the right execution scripts in the right order, handle errors, and ask you for clarification when needed. When you

tell the agent "Build the homepage hero section," the agent should first check `directives/build-page.md` for instructions, then check `execution/` for any existing scripts it should use (like the CSS generator or the HTML template builder), then execute the work.

Why this matters for Axiara: The agent is the glue between your intent ("build the Episteme technology page") and the deterministic execution ("run the design system search, generate the HTML template, inject the correct content from the manual, apply the frosted glass card styling"). The agent decides order, handles errors, and self-anneals when something breaks.

Layer 3: Execution (Doing the work)

Execution scripts are deterministic Python scripts in the `execution/` folder. They handle the reliable, repeatable tasks: generating CSS from the design tokens, building HTML templates from a standard structure, optimizing images, validating HTML output, and deploying to WordPress. The key principle from AGENTS.md is that you push complexity into deterministic code so the agent focuses on decision-making — because 90% accuracy per step compounds into 59% success over 5 steps if the agent does everything itself, but execution scripts are 100% reliable once tested.

Why this matters for Axiara: Tasks like "generate the complete Darkroom CSS kit from the Axiara design tokens" or "create an HTML page shell with the correct font imports, Tailwind CDN, and meta tags" are deterministic. They should be scripts, not agent improvisation. Once the script works, it works every time for all 15 pages.

The Self-Annealing Loop

When something breaks — and it will — the system learns from the failure. The agent reads the error, fixes the script, tests it again, and then updates the directive with what it learned (for example: "backdrop-filter requires the -webkit prefix for Safari" or "Tailwind CDN v3.4 changed the dark mode class syntax"). Each failure makes the system stronger. This is especially important for a project with 15+ pages where the same patterns repeat — a lesson learned on the homepage should be baked into the directive so it never happens again on subsequent pages.

---

2. Phase 0: Project Setup & Folder Structure

Step 0.1 — Create or Open the Antigravity Project

Open Google Antigravity IDE. Create a new project named `axiara-id-website`, or open your existing workspace if you have already started one.

Step 0.2 — Verify AGENTS.md Is Loaded

The `AGENTS.md` file must be loaded as the foundational agent instruction set. This is typically placed at the project root or in a `.agent/` configuration folder so the agent reads it at the start of every session. Verify it is present by checking your project root. If it is not there, upload it now — it should be the very first file in the project because it tells the agent how to operate.

Step 0.3 — Create the Full Directory Structure

In the Antigravity terminal, create the complete project folder structure. This structure follows the AGENTS.md conventions exactly, with additions specific to the Axiara website project:

```bash
```

```
# Core architecture folders (from AGENTS.md)
mkdir -p directives
mkdir -p execution
mkdir -p .tmp

# Project-specific folders
mkdir -p docs
mkdir -p design-system/pages
mkdir -p src/css
mkdir -p src/components
mkdir -p src/pages/services
mkdir -p src/pages/technology
mkdir -p src/pages/industries
mkdir -p src/pages/compliance
mkdir -p src/assets/icons
mkdir -p src/assets/images
```

Here is what each folder does and why it exists:

directives/ holds all your SOPs — the instruction files the agent reads before doing any work. You will create one directive for each major concern: brand guidelines, page building, design system generation, deployment, and QA.

execution/ holds deterministic Python scripts that the agent calls. These handle repeatable tasks like generating the CSS from design tokens, building HTML page shells, validating output, and optimizing images.

.tmp/ holds all intermediate files — scraped data, temporary exports, processing artifacts. Nothing in .tmp/ is a deliverable. It can be deleted and regenerated at any time.

docs/ holds the project plan generated by /plan and the Axiara Master Manual (your content source).

design-system/ holds the output from the ui-ux-pro-max design system generator — the MASTER.md global file and page-specific overrides in the pages/ subfolder.

src/ holds all the HTML, CSS, and assets that make up the website. This is the actual build output.

Step 0.4 — Upload All Reference Documents

Place these files into the project so the agent and the directives can reference them:

```bash
# Upload into project root or docs/
# 1. The Axiara Master Architecture & Operations Manual
cp Axiara_Master_Architecture_Operations_Manual_v2_0.md docs/

# 2. The Agency 9 Creative brand guidelines
cp agency_9_creative.md docs/

# 3. The ui-ux-pro-max skill file (should already be in .agent/.shared/)
# Verify it exists:
ls .agent/.shared/ui-ux-pro-max/scripts/search.py

# 4. The plan.md command definition (should already be in .agent/commands/)
# Verify it exists:
ls .agent/commands/plan.md
```

Step 0.5 — Verify Python Environment

The design system generator and execution scripts require Python 3:

```bash
python3 --version || python --version
```

If Python is not available, install it before proceeding (see `ui-ux-pro-max.md` for OS-specific installation commands).

Step 0.6 — Set Up Environment Variables (If Needed)

If any execution scripts will need API keys or configuration values (for example, a WordPress deployment script that needs site credentials), create the `.env` file now:

```bash
# .env — Environment variables for execution scripts
# WordPress deployment (for Phase 13)
WP_SITE_URL=https://axiara.id
WP_ADMIN_USER=admin
WP_APP_PASSWORD=xxxx-xxxx-xxxx-xxxx

# Google OAuth (if using Google Sheets/Slides for deliverables)
# credentials.json and token.json should be in project root
```

You may not need all of these immediately. The important thing is that the `.env` file exists and is ready to receive credentials as execution scripts are built.

---

3. Phase 1: Create All Directives (Layer 1)

This is the most important phase of the entire project. The directives you create here will be read by the agent every time it builds a page, generates a component, or runs a QA check. Well-written directives mean the agent produces consistent, brand-compliant output across all 15+ pages without you repeating yourself.

You will create **six directive files**. Each one is a self-contained SOP for a specific concern.

Step 1.1 — Create the Brand Directive

This directive encodes every visual rule from the `agency_9_creative.md` file into an operational SOP that the agent reads before building anything. It is the single most critical directive because it prevents brand drift across pages.

Create the file by telling the agent:

```
Create the file directives/axiara-brand.md with the following content.
This is a Layer 1 Directive — an SOP that you must read before
building any UI component or page for the Axiara project.
```

Content for `directives/axiara-brand.md`:

```
markdown
```

Directive: Axiara Brand Guidelines

> SOP for all UI/visual decisions on the Axiara.id website.
> Read this directive BEFORE building any page or component.
> Source of truth: docs/agency_9_creative.md

Visual Concept: "The Darkroom"

The site feels like a high-end architectural portfolio crossed with
a sovereign intelligence platform. Zaha Hadid Architects meets Palantir.
Content does not "pop" (playful) — it "unveils" (premium).

Color Palette (Non-Negotiable)

Token	Value	CSS Variable	Usage
Obsidian	#0D0D0D	--axiara-obsidian	Primary background
Crimson	#C41E3A	--axiara-crimson	Accent lines, hover, "Weave"
White	#FFFFFF	--axiara-white	H1 headlines, button borders
Silver	#A8A8A8	--axiara-silver	Body text, H2 subheads
Charcoal	#1A1A2E	--axiara-charcoal	Gradient endpoint
Glass	rgba(26,26,46,0.4)	—	Frosted card backgrounds
Glass Hover	rgba(26,26,46,0.8)	—	Card hover state
Crimson Glow	rgba(196,30,58,0.05)	—	Button hover tint

No other colors are permitted unless explicitly approved.

Typography (Non-Negotiable)

Element	Font	Weight	Style	Notes
H1	Outfit	900	Normal, UPPERCASE	letter-spacing: 0.05em

H2	Playfair Display	400	ITALIC	color: Silver
H3	Outfit	700	Normal	color: White
Body	Outfit	300	Normal	18px, color: Silver
Code/Labels	JetBrains Mono	300-400	Normal	For data, badges, buttons
Buttons	JetBrains Mono	400	UPPERCASE	letter-spacing: 2px

Google Fonts import:
@import url('https://fonts.googleapis.com/css2?family=JetBrains+Mono:wght@300;400&family=Outfit:wght@300;4

UI Components

Ghost Buttons (The ONLY button style)
- background: transparent
- border: 1px solid white
- border-radius: 0px (ALWAYS — "knife-edge apex")
- font: JetBrains Mono, uppercase, letter-spacing 2px
- hover: border-color crimson, text crimson, bg crimson-glow
- transition: all 0.4s ease
- padding: 14px 32px
- ALWAYS add cursor-pointer

Frosted Glass Cards (The ONLY card style)
- background: rgba(26,26,46,0.4)
- backdrop-filter: blur(12px) + -webkit-backdrop-filter: blur(12px)
- border-LEFT only: 1px solid crimson (NO top, right, bottom borders)
- hover: bg opacity 0.8 + border-left thickens to 4px
- padding: 40px
- transition: all 0.4s ease
- Fallback for no backdrop-filter support: solid bg-charcoal

Asymmetric Image Treatment
- transform: translate(20px, 20px) on desktop

- 1px wireframe border behind (rgba(255,255,255,0.1))
- Remove offset on mobile (transform: none below 768px)

Motion Rules (STRICTLY ENFORCED)

ALLOWED:
- Fade In Up (translateY 20-30px → 0, opacity 0 → 1)
- Duration: 0.8–1.2s
- Easing: ease-out
- Staggered delays (0.1-0.2s between siblings)
- Slow parallax (0.3-0.5x scroll speed)
- Color/opacity/border transitions (0.3-0.4s)

FORBIDDEN (Never use, under any circumstances):
- Bounce, elastic, spring
- Zoom, scale transforms on hover that shift layout
- Spin, rotate entrance animations
- Any animation faster than 0.6s for entrance
- Any "playful" or "fun" motion

Icons
- USE: Lucide Icons via CDN (lucide.dev)
- NEVER: Emoji icons (🎨 🚀 ⚙️ etc.)
- Style: 48-64px, white, stroke-width 1 (thin wireframe)

Border Radius
- 0px on EVERYTHING. Buttons, cards, inputs, images, badges.
- No exceptions. This is the "knife-edge apex" brand rule.

Photography
- Black & white only, with optional Crimson color overlay
- No full-color stock photography

```
- Editorial, documentary aesthetic

## Edge Cases & Learnings
(This section gets updated by the agent via self-annealing)
- [Placeholder — agent adds learnings as it discovers them]
```

Step 1.2 — Create the Page Build Directive

This directive tells the agent how to construct any page in the project. It defines the standard structure, the required imports, and the build/verify process.

Create `directives/build-page.md`:

```
markdown
```

Directive: Build a Page

> SOP for constructing any HTML page in the Axiara.id project.
> Read directives/axiara-brand.md FIRST for all visual rules.

Pre-Build Checklist

Before writing any code:
1. Read directives/axiara-brand.md (brand rules)
2. Read design-system/MASTER.md (design system)
3. Check design-system/pages/{page-name}.md for page-specific overrides
4. Check execution/ for existing scripts that can help
5. Identify the correct Manual section for content

Standard Page Structure

Every page must include:
1. <!DOCTYPE html> with lang="en"
2. Google Fonts import (Outfit, Playfair Display, JetBrains Mono)
3. Tailwind CSS CDN
4. Link to src/css/axiara.css (the Darkroom kit)
5. Lucide Icons CDN
6. Navbar component (from src/components/navbar.html)
7. Main content sections
8. Footer component (from src/components/footer.html)
9. Scroll-triggered animation JavaScript (IntersectionObserver)
10. prefers-reduced-motion media query to disable animations

Content Source

All page copy comes from:
docs/Axiara_Master_Architecture_Operations_Manual_v2_0.md

```
Reference specific sections as noted in the project plan.
Do NOT invent content. Every claim must trace to the manual.

## Output Location

- Working files: src/pages/{category}/{page}.html
- Intermediate/temp files: .tmp/
- Final deliverables: pushed to cloud or exported for BeTheme

## Post-Build Verification

After building any page, run:
1. execution/validate_html.py {page} — checks structure & brand compliance
2. Visual review at 375px, 768px, 1024px, 1440px
3. Verify all animations are fade-in-up only (no bounce/spin/zoom)
4. Verify all border-radius is 0px
5. Verify no emojis are used as icons

## Self-Annealing Notes
(Updated as issues are discovered)
- [Placeholder]
```

Step 1.3 — Create the Design System Directive

This directive encodes the exact workflow for running the `ui-ux-pro-max` tool.

Create `directives/design-system.md`:

```
markdown
```

Directive: Generate & Manage the Design System

> SOP for running the ui-ux-pro-max design system generator.
> Tools: .agent/.shared/ui-ux-pro-max/scripts/search.py

When to Run

- ALWAYS run --design-system before building the first page
- Run page-specific overrides (--page flag) for complex pages
- Run supplementary domain searches when you need deeper detail

Master Design System Generation

```bash
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "AI governance enterprise dark luxury editorial cinematic B2B" \
  --design-system --persist -p "Axiara"
```

Output: design-system/MASTER.md

Page-Specific Overrides

```bash
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "" \
  --design-system --persist -p "Axiara" --page ""
```

Output: design-system/pages/{page-name}.md

Hierarchical Retrieval Rule

When building a page:

1. Check design-system/pages/{page-name}.md FIRST
2. If it exists, its rules OVERRIDE the Master file
3. If it does not exist, use design-system/MASTER.md exclusively

Conflict Resolution

If ui-ux-pro-max output conflicts with docs/agency_9_creative.md:
→ agency_9_creative.md ALWAYS WINS
→ The brand file is the supreme authority on visual decisions
→ ui-ux-pro-max provides supplementary intelligence (UX patterns,
landing structure, chart types) — not brand overrides

Supplementary Searches

Need	Command
UX guidelines	--domain ux "animation accessibility"
Landing structure	--domain landing "hero social-proof CTA"
Typography detail	--domain typography "serif sans editorial"
Color detail	--domain color "dark obsidian crimson"
Chart types	--domain chart "real-time dashboard"
Stack guidelines	--stack html-tailwind "responsive glassmorphism"

Self-Annealing Notes
- [Placeholder]

Step 1.4 — Create the QA Directive

Create `directives/qa-checklist.md`:

markdown

Directive: Quality Assurance Checklist

> SOP for verifying every page before delivery.
> Run AFTER building each page, not just at the end.

Visual Quality Checks

- [ ] All three fonts loading (Outfit, Playfair Display, JetBrains Mono)
- [ ] Frosted glass cards render in Chrome, Firefox, Safari, Edge
- [ ] backdrop-filter fallback present (solid bg for unsupported browsers)
- [ ] Crimson is #C41E3A everywhere (no variations like #C41E3B)
- [ ] No emojis used as icons (Lucide SVGs only)
- [ ] 0px border-radius on EVERYTHING (buttons, cards, inputs, images)
- [ ] Photography is B&W with optional Crimson overlay (no full-color stock)

Brand Compliance Checks

- [ ] H1 = Outfit 900 uppercase (every page)
- [ ] H2 = Playfair Display italic silver (every page)
- [ ] All buttons are ghost style (transparent bg, white border, 0px radius)
- [ ] All cards use crimson left-border ONLY (no top/right/bottom)
- [ ] Asymmetric image offset applied (translate 20px 20px)
- [ ] No "bouncy" or "playful" animations anywhere
- [ ] All motion is fade-in-up only (0.8-1.2s, ease-out)

Responsive Checks

- [ ] 375px — text scales, grids stack, no horizontal scroll
- [ ] 768px — tablet layout, proper spacing
- [ ] 1024px — navbar full, desktop layout begins
- [ ] 1440px — full desktop, content centered
- [ ] Ghost buttons have minimum 48px tap target on mobile

- [ ] Image offsets removed on mobile (transform: none below 768px)

Accessibility Checks

- [ ] All images/icons have alt text
- [ ] All form inputs have <label> elements
- [ ] Color contrast ≥ 4.5:1 (Silver on Obsidian = 7.8:1 ✓)
- [ ] Keyboard tab navigation works on all interactive elements
- [ ] Focus states visible (crimson outline)
- [ ] prefers-reduced-motion disables all animations
- [ ] Semantic HTML (proper heading hierarchy, landmarks)
- [ ] Skip-to-content link present

Performance Checks

- [ ] Images lazy loaded
- [ ] Total page weight < 2MB
- [ ] No duplicate CSS across pages (shared axiara.css)

Run Validation Script

```bash
python3 execution/validate_html.py src/pages/{page}.html
```

Self-Annealing Notes

- [Placeholder]

Step 1.5 — Create the Deployment Directive

Create `directives/deploy-betheme.md`:

markdown

Directive: Deploy to WordPress + BeTheme

> SOP for transferring built HTML/CSS from Antigravity into
> the WordPress/BeTheme production environment.

Strategy: Section-by-Section Transfer

Transfer each page section by section, NOT page by page.
BeTheme's Muffin Builder uses a Section → Wrap → Column hierarchy
that maps directly to the HTML structure.

Mapping:
- HTML <section> → Muffin Builder "Section" block
- HTML <div.wrap> → Muffin Builder "Wrap" block
- HTML <div.column> → Muffin Builder "Column" block
- HTML content → Muffin Builder elements (text, image, icon box, etc.)

Step 1: Transfer Global CSS

Copy src/css/axiara.css → Appearance → Theme Options → Custom CSS/JS → CSS.
This single paste applies all Darkroom styling globally.

Step 2: BeTheme Options Configuration

- Layout: Full Width (100%)
- Header: Transparent Sticky, scroll bg rgba(13,13,13,0.85) + blur(10px)
- Colors: Primary #C41E3A, Background #0D0D0D, Text #A8A8A8, Headings #FFFFFF
- Entrance Animations: Fade In Up, Slow. DISABLE Zoom, Spin, Bounce.
- Typography: Disable all BeTheme defaults, custom fonts via CSS import.

Step 3: Transfer Each Page

For each page, open the Antigravity HTML and Muffin Builder side by side.
Recreate the structure section by section. Apply custom CSS classes
where needed (e.g., class "axiara-weave" for diagonal crimson lines).

Step 4: Required WordPress Plugins

- WPForms or Gravity Forms (contact form)
- WP Rocket or LiteSpeed Cache (performance)
- Yoast SEO or Rank Math (SEO)
- Smush or ShortPixel (image optimization)

Step 5: DNS & SSL

Point axiara.id DNS to the WordPress host.
Ensure SSL certificate is active (HTTPS required).

Self-Annealing Notes
- [Placeholder]

Step 1.6 — Create the Content Directive

Create `directives/content-source.md`:

markdown

Directive: Content Sourcing Rules

> SOP for pulling content from the Axiara Master Manual.
> Every piece of text on the website must trace to this document.

Source Document

docs/Axiara_Master_Architecture_Operations_Manual_v2_0.md
Version 2.0 | February 2026

Page-to-Section Mapping

Page	Manual Section(s)
Homepage	1.1 (mission), 1.2 (E2E model), 1.3 (three pillars), 1.4 (sectors)
About	1.1 (overview), 1.3 (philosophy), 6.1 (founders), 6.2 (Core 3 team)
30-Day Sprint	3.1 (Week 1), 3.2 (Week 2), 3.3 (Week 3), 3.4 (Week 4), 3.5 (summary)
E2E Transformation	1.2 (E2E model), Appendix 8.3 (client journey map)
Aksara Vector Core	2.1 (all subsections)
Episteme	2.2 through 2.2.3
Terpadu Engine	2.3 through 2.3.2
Telco	1.4 (Telecommunications row)
Energy	1.4 (Energy row)
Banking	1.4 (Banking & Insurance row)
BUMN	1.4 (BUMN row) + 5.6 (local PT advantage)
ISO 42001	5.5 + 7 (regulation table)
UU PDP	7 (regulation table, UU PDP row)
AI Decree	7 (regulation table, Decree row)
Contact	N/A (standalone form)

Content Rules

1. Never invent claims. Every statistic (99.9%, 100%, 30 days, Level 5)
   must come from the manual.
2. Preserve the voice: sovereign, precise, authoritative. Not salesy.
3. Use direct quotes from the manual for pillar descriptions and key
   differentiators — these are deliberate formulations, not paraphrases.
4. The Axiara Accuracy Standard formula (A = (T_correct - T_hallucination)
   / T_total × 100 ≥ 99.9%) can be displayed on the technology pages.
5. For industry pages, use the exact "Strategic Pain Point" and
   "E2E Entry Thesis" language from Section 1.4.

Voice Guidelines

- Do: "We assign a human Logic Owner to every automated decision node"
- Don't: "We help you leverage synergies to disrupt the AI space"
- Do: "An unaudited AI decision creates existential liability"
- Don't: "Our world-class solutions transform your business!"
- Do: "We build" / "We deploy" / "We transform"
- Don't: "We help you" (passive, weak)
- No exclamation marks. No startup jargon. Engineering tone.

Self-Annealing Notes
- [Placeholder]

---

4. Phase 2: Create Execution Scripts (Layer 3)

Now you create the deterministic Python scripts that the agent will call. These scripts handle the repeatable, reliable tasks so the agent (Layer 2) can focus on decision-making rather than doing everything itself. Remember the core principle from AGENTS.md: 90% accuracy per step = 59% success over 5 steps. Push complexity into deterministic code.

Step 2.1 — CSS Generator Script

This script generates the complete Darkroom CSS kit from the Axiara design tokens. Instead of having the agent write CSS from memory each time (which introduces drift), this script produces the exact same output every time it runs.

Tell the agent:

```
Create execution/generate_css.py

This script generates the complete Axiara "Darkroom" CSS kit.
It reads the design tokens (colors, fonts, variables) and outputs
a valid CSS file at src/css/axiara.css.

The CSS must include all 5 blocks from docs/agency_9_creative.md:
A. Google Fonts @import
B. :root CSS custom properties (Darkroom Variables)
C. Typography overrides (H1 Outfit 900 uppercase, H2 Playfair italic, etc.)
D. UI element styles (ghost buttons, frosted glass cards)
E. Layout styles (asymmetric image offset, body background, weave lines)

The script should be deterministic — same input, same output, every time.
Include comments referencing the brand guideline source for each section.
Well-commented Python. No external dependencies beyond standard library.
```

Step 2.2 — HTML Page Shell Generator

This script generates the standard HTML shell for any page, so the agent does not have to recreate the boilerplate each time.

Tell the agent:

```
Create execution/generate_page_shell.py

This script generates a standard HTML page shell for the Axiara.id site.

Usage: python3 execution/generate_page_shell.py --title "Page Title"
       --slug "page-slug" --output "src/pages/category/page.html"

The shell includes:
- <!DOCTYPE html> with lang="en"
- <meta> tags (charset, viewport, description)
- Google Fonts import link
- Tailwind CSS CDN link
- <link> to src/css/axiara.css
- Lucide Icons CDN script
- Navbar HTML (read from src/components/navbar.html or inline)
- <main> wrapper with id="content"
- Footer HTML (read from src/components/footer.html or inline)
- IntersectionObserver script for scroll-triggered fade-in-up animations
- prefers-reduced-motion media query that disables all animations
- Skip-to-content link for accessibility

Arguments:
--title   : Page <title> tag text (e.g., "NTRJ Episteme — Axiara")
--slug    : URL slug (e.g., "episteme")
--output  : Output filepath
--desc    : Meta description text (optional)

Well-commented Python. Use string templates, no external dependencies.
```

Step 2.3 — HTML Validation Script

This script checks a built page against the brand rules encoded in the directive. It is the automated

enforcement layer for the QA checklist.

Tell the agent:

```
Create execution/validate_html.py

This script validates an Axiara HTML page for brand compliance and
structural correctness.

Usage: python3 execution/validate_html.py src/pages/services/sprint.html

Checks:
1. STRUCTURE: Valid HTML5 doctype, lang attribute, meta viewport
2. FONTS: Google Fonts import URL contains Outfit, Playfair Display,
   JetBrains Mono
3. CSS: Links to axiara.css
4. NO EMOJIS: Scan for common emoji unicode ranges in visible text
5. BORDER RADIUS: Scan for any border-radius value > 0
   (flag "rounded-" Tailwind classes except "rounded-none")
6. ANIMATIONS: Flag any CSS animation names containing "bounce",
   "spin", "zoom", "elastic", "scale"
7. HEADINGS: Verify heading hierarchy (no skipping from H1 to H3)
8. ACCESSIBILITY: Check for alt="" on images, <label> on form inputs
9. REDUCED MOTION: Check for prefers-reduced-motion media query

Output: PASS/FAIL per check, with line numbers for failures.

Well-commented Python. Parse HTML with standard library (html.parser).
No external dependencies.
```

Step 2.4 — Image Optimizer Script (Optional)

Tell the agent:

```
Create execution/optimize_images.py

This script processes images for the Axiara site:
1. Converts color photos to grayscale (B&W requirement)
2. Optionally applies a Crimson (#C41E3A) duotone overlay
3. Compresses output to WebP format
4. Generates multiple sizes (375w, 768w, 1440w) for responsive srcset

Usage: python3 execution/optimize_images.py
    --input src/assets/images/raw/photo.jpg
    --output src/assets/images/
    --crimson-overlay (optional flag)

Requires Pillow: pip install Pillow
```

Step 2.5 — Verify All Scripts Work

Run each script once to confirm it executes without errors:

```
bash
```

```
# Generate the CSS
python3 execution/generate_css.py

# Generate a test page shell
python3 execution/generate_page_shell.py --title "Test Page" --slug "test" --output .tmp/test.html

# Validate the test page
python3 execution/validate_html.py .tmp/test.html
```

If any script fails, this is where the self-annealing loop kicks in. The agent reads the error, fixes the script, tests again, and updates the corresponding directive with what it learned. For example, if the CSS generator outputs an invalid `@import` placement, the agent fixes the script and adds a note to `directives/axiara-brand.md` under "Self-Annealing Notes": "CSS @import must be the first rule in the file — no comments or rules before it."

---

5. Phase 3: Generate the Project Plan

Now that the architecture is in place (directives written, execution scripts ready), you generate the formal project plan using the `/plan` command.

Step 3.1 — Run the /plan Command

```
/plan Axiara.id corporate website — dark luxury AI governance brand,
15 pages including homepage, about, services (30-day sprint, E2E,
aksara vector core), technology (episteme, terpadu), industries
(telco, energy, banking, BUMN), compliance (ISO 42001, UU PDP,
AI decree), insights blog, and contact page.
Stack: HTML + Tailwind CSS.
```

Brand: Agency 9 Creative "Darkroom" aesthetic.
Architecture: 3-layer (directives in directives/, scripts in execution/).

Step 3.2 — Answer the Socratic Gate

The agent will ask clarifying questions. Here are the answers you should give, tailored to the 3-layer architecture:

Q: "What is the primary goal?" A: "Convert C-suite executives (CEO, CRO, DPO, CTO) of regulated Indonesian enterprises into 30-Day Sprint assessment leads. The site must communicate that AI without governance is a Board-level liability."

Q: "What technology stack?" A: "HTML + Tailwind CSS as standalone pages. Final deployment is WordPress + BeTheme. Directives are in directives/, execution scripts in execution/, and the agent orchestrates between them per AGENTS.md."

Q: "Any existing brand assets?" A: "Yes. docs/agency_9_creative.md has the complete brand CSS kit. directives/axiara-brand.md has the operational SOP version. The agent must read the directive before building any page."

Q: "How many pages?" A: "15 pages. See directives/content-source.md for the complete page-to-manual-section mapping."

Step 3.3 — Review the Plan

The agent creates `docs/PLAN-axiara-website.md`. Open it and verify that the task breakdown references the directives and execution scripts. A good plan should look like this at the task level:

```
Task 4.1: Build Homepage Hero
- Read: directives/axiara-brand.md, directives/build-page.md
- Run: execution/generate_page_shell.py --title "Axiara.ai" --slug "home"
- Content from: Manual Section 1.1 (mission statement)
- Design system: design-system/pages/homepage.md
- Build hero section with H1, H2, body, CTAs
- Run: execution/validate_html.py src/pages/index.html
```

If the plan does not reference directives and scripts, tell the agent to revise it: "The plan should reference the specific directive files and execution scripts for each task, per our 3-layer architecture in AGENTS.md."

---

6. Phase 4: Generate the Design System

Run the `ui-ux-pro-max` tool to generate the design system. The agent should read `directives/design-system.md` first, which tells it exactly what commands to run and how to handle conflicts with the brand guidelines.

Step 4.1 — Generate Master Design System

```bash
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "AI governance enterprise dark luxury editorial cinematic B2B" \
  --design-system --persist -p "Axiara"
```

This creates `design-system/MASTER.md`.

Step 4.2 — Generate Page-Specific Overrides

```bash
```

```
# Homepage
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "AI governance hero landing page CTA enterprise dark" \
  --design-system --persist -p "Axiara" --page "homepage"

# 30-Day Sprint (primary conversion page)
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "timeline process steps enterprise service dark" \
  --design-system --persist -p "Axiara" --page "sprint"

# Episteme (technical product page)
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "product feature comparison dashboard dark tech" \
  --design-system --persist -p "Axiara" --page "episteme"

# Contact
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "contact form enterprise minimal dark" \
  --design-system --persist -p "Axiara" --page "contact"
```

Step 4.3 — Run Supplementary Domain Searches

bash

```
# UX: animation guidelines (critical for Axiara's "unveil" rule)
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "animation slow reveal parallax premium" --domain ux

# Landing page: structure patterns
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "hero social-proof CTA enterprise B2B" --domain landing

# Typography: confirm Outfit + Playfair pairing
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "serif sans editorial luxury professional" --domain typography

# Stack: Tailwind-specific implementation guidelines
python3 .agent/.shared/ui-ux-pro-max/scripts/search.py \
  "layout responsive form glassmorphism dark-mode" --stack html-tailwind
```

Step 4.4 — Verify Design System Folder

```
design-system/
├── MASTER.md
└── pages/
    ├── homepage.md
    ├── sprint.md
    ├── episteme.md
    └── contact.md
```

---

7. Phase 5: Build Global Foundation

Now the agent begins actual implementation. For every step from here forward, the agent should follow

this loop based on the 3-layer architecture:

The Build Loop (repeat for every task):

1. Agent reads the relevant directive(s)
2. Agent checks `execution/` for existing scripts that can help
3. Agent calls scripts first for deterministic work (CSS generation, page shell)
4. Agent writes the custom/creative code (section content, layout decisions)
5. Agent runs `execution/validate_html.py` to verify
6. If validation fails → self-annealing loop (fix → test → update directive)

Step 5.1 — Generate the Global CSS

```
Read directives/axiara-brand.md, then run execution/generate_css.py
to create src/css/axiara.css. Verify the output matches the brand
directive exactly.
```

Step 5.2 — Build Navbar Component

Tell the agent:

```
Read directives/axiara-brand.md and directives/build-page.md.

Build src/components/navbar.html — the global navigation component.

BEHAVIOR:
- Starts transparent (background: transparent) at page top
- On scroll past 50px, transitions to frosted glass:
  background rgba(13,13,13,0.85) + backdrop-filter blur(10px)
- Fixed position, z-index 50

CONTENT:
- Left: "AXIARA" wordmark (Outfit 700, white, letter-spacing 0.05em)
- Center: Nav links — About | Services | Technology | Industries
  | Compliance | Insights | Contact
  (JetBrains Mono, 12px, uppercase, letter-spacing 2px, white,
   hover: crimson, transition 0.3s)
- Right: Ghost button CTA "REQUEST ASSESSMENT"

MOBILE (below 1024px):
- Hamburger icon (three thin white lines, cursor-pointer)
- Opens full-screen overlay: bg-obsidian, nav links stacked vertically
  centered, with a diagonal crimson line at ~15deg as decoration
- Close button: thin "X" in white, top-right

Use HTML + Tailwind CSS. Include the scroll-detection JavaScript inline.
Add cursor-pointer to all clickable elements.
0px border-radius on everything.
```

Step 5.3 — Build Footer Component

```
Read directives/axiara-brand.md.
```

Build src/components/footer.html — the global footer.

CONTENT:
- Logo: "AXIARA" (Outfit 700, white)
- Tagline: "Foundational Logic. Integrated Innovation."
  (Playfair Display italic, silver, 16px)
- Company: "PT Aksara Inovasi Terpadu" (Outfit 300, silver, 14px)
- Navigation columns: Services | Technology | Company | Legal
- Copyright: "© 2026 PT Aksara Inovasi Terpadu. All rights reserved."
  (JetBrains Mono, 11px, silver/60%)

STYLE:
- Background: axiara-gradient (135deg, obsidian → charcoal)
- Top border: 1px solid rgba(255,255,255,0.05)
- Padding: 80px top, 40px bottom
- 0px border-radius everywhere

Step 5.4 — Create Tailwind Config Extension

Create an inline Tailwind config (or tailwind.config.js if using
build step) that extends Tailwind with Axiara design tokens:

Colors: obsidian #0D0D0D, crimson #C41E3A, charcoal #1A1A2E, silver #A8A8A8
Fonts: font-eng (Outfit), font-phil (Playfair Display), font-code (JetBrains Mono)

So I can write: bg-obsidian, text-crimson, font-eng, font-phil, font-code
instead of arbitrary values throughout the HTML.

---

8. Phase 6: Build Homepage

The homepage has 7 sections. Build each one as a separate prompt, then assemble. The agent should read `directives/build-page.md` and `directives/axiara-brand.md` before starting, and run `execution/generate_page_shell.py` first to create the HTML shell.

Step 6.0 — Generate the Page Shell

```
Run: python3 execution/generate_page_shell.py \
  --title "Axiara.ai — End-to-End AI Transformation Partner" \
  --slug "home" \
  --output src/pages/index.html \
  --desc "Axiara.ai moves enterprise clients from experimental AI to governed AI excellence. End-to-End AI
Transformation Partner for regulated Indonesian enterprises."
```

Step 6.1 — Hero Section

```
Read directives/axiara-brand.md and directives/content-source.md.
Content source: Manual Section 1.1 (mission statement).
Design reference: design-system/pages/homepage.md.

Build the hero section inside src/pages/index.html <main>.

LAYOUT: Full viewport height (min-h-screen). Background: Nexus Grid
texture (create a subtle SVG hex pattern as inline background, or
use a CSS repeating pattern) with slow parallax (background-attachment: fixed).
Three diagonal crimson lines at 5% opacity as ::before pseudo-elements.

CONTENT (left side, max-w-xl):
- Pre-headline: "END-TO-END AI TRANSFORMATION PARTNER"
  (font-code text-xs text-crimson uppercase tracking-[3px])
```

- H1: "YOUR AI IS A LIABILITY."
 "WE MAKE IT AN ASSET."
  (font-eng font-black text-7xl text-white uppercase tracking-wide)
- H2: "Foundational Logic. Integrated Innovation."
  (font-phil text-2xl italic text-silver)
- Body: Mission statement from Manual Section 1.1.
  (font-eng font-light text-lg text-silver max-w-lg)
- CTA: Ghost button "SCHEDULE YOUR 30-DAY SPRINT →"
- Secondary: "See How It Works ↓" (text-silver hover:underline)

RIGHT SIDE (hidden on mobile): SVG wireframe showing a dark opaque
cube transforming into a transparent "glass box" with internal
crimson logic pathways. CSS animation, 4s infinite loop, ease-in-out.

ANIMATION: Staggered fade-in-up on all text elements.
Pre-headline 0.2s delay, H1 0.4s, H2 0.7s, body 0.9s, CTAs 1.1s.
Each: translateY(30px)→0, opacity 0→1, 0.8s ease-out.
Use IntersectionObserver for scroll trigger OR load trigger for hero.

Step 6.2 — E2E Timeline Section

Build the "End-to-End Arc" section.
Content: Manual Section 1.2 (E2E Transformation Model).

H2: "No Handoff Points. No Vendor Gaps."

Five-node horizontal timeline (overflow-x-auto, scroll-snap-x mandatory
on mobile, full row on desktop). Connected by a thin white line.

Nodes (frosted glass cards, min-w-[240px]):
1. "DIAGNOSE" — Lucide: radar — "Shadow AI Audit & Risk Mapping"
2. "ARCHITECT" — Lucide: layers — "Sovereign Data Foundation & Vector Core"
3. "ILLUMINATE" — Lucide: eye — "Transparency Layer & Logic Mapping"

4. "INTEGRATE" — Lucide: puzzle — "ERP/CRM Embedding & Agentic Workflows"
5. "SUSTAIN" — Lucide: shield-check — "Continuous Governance & Certification"

Below: Playfair Display italic centered:
"One accountable partner from boardroom strategy to production deployment."

Step 6.3 — Three Pillars Section

Build the "Three Pillars" section.
Content: Manual Section 1.3 (Axiara Philosophy).

H2: "The Architecture of Certainty"

Three-column grid (lg:grid-cols-3). Each pillar = tall frosted glass card.

Card 1 — AXIOM: Lucide "scale" | "TRUTH IN LOGIC" | "Explainability & Audit"
  Quote: "If the Board cannot audit it in 30 seconds, it is not ready
  for production."

Card 2 — AKSARA: Lucide "database" | "FOUNDATIONAL SCRIPT" | "Data Sovereignty"
  Quote: "Without the Aksara, AI is guessing. With it, AI is grounded in
  verified institutional truth."

Card 3 — TERPADU: Lucide "git-merge" | "INTEGRATED INNOVATION" | "Seamless Embedding"
  Quote: "If a solution is not embedded into the marrow of the client's
  operations, it is not finished."

Staggered fade-in-up on scroll (0s, 0.15s, 0.3s delays).

Step 6.4 — Statistics Section

Build the "Governance Gap" stats section.
Content: Manual Sections 2.1, 3.5, 8.1.

H2: "What Your Board Doesn't See"

Split: Left = 2×2 stat grid, Right = Episteme dashboard wireframe mockup.

Stats (Outfit 900, crimson numbers):
- 99.9% — Axiara Accuracy Standard / Zero Hallucination Pipeline
- 100% — Data Residency / Within Indonesian Borders
- 30 DAYS — Black Box to Glass Box / Complete Governance Sprint
- Level 5 — Target Excellence / ISO 42001 Certification Pathway

Numbers count-up animate on scroll (IntersectionObserver).

Right: Frosted glass frame containing a simplified wireframe dashboard
(thin white/silver lines, small node dots, horizontal bars — all CSS/SVG).

Step 6.5 — Industry Authority Section

Build the "Industry Authority" section.
Content: Manual Section 1.4 (Target Sectors).

H2: "Built for Regulated Enterprise"

2×2 grid of frosted glass cards:
1. Telecommunications — Lucide "radio-tower" — AI routing without audit
2. Energy & Utilities — Lucide "zap" — Life-safety AI decisions
3. Banking & Insurance — Lucide "landmark" — Automated credit/claims

4. BUMN — Lucide "building-2" — Presidential Decree compliance

Each card links to /industries/{slug}. "Explore →" link in crimson.

Step 6.6 — Regulatory Stack Section

Build the "Regulatory Stack" section.
Content: Manual Section 7.

H2: "Three Mandates. One Partner."

Three horizontal frosted glass bars stacked vertically:
1. UU PDP — "100% data residency, automated PII scrubbing, AES-256"
2. Presidential AI Decree — "Episteme audit trails + human Logic Owners"
3. ISO 42001 — "30-Day Sprint → Clause 4-10 alignment"

Below: "AXIARA" centered in Outfit 900, crimson — the convergence point.

Step 6.7 — CTA Footer Section

Build the final CTA section.

H2: "Your AI governance gap is a ticking clock."
Body: "The 30-Day Sprint moves your organization from Level 1 to Level 4
maturity — from Shadow AI exposure to Glass Box auditability — in four weeks."
CTA: "REQUEST YOUR SPRINT ASSESSMENT →" (ghost button, larger padding)
Sub-text: "CEO-to-CEO engagement. Board-level outcomes." (font-code 11px)

Background: axiara-gradient. Padding: 120px vertical.

Step 6.8 — Validate the Homepage

```
Run: python3 execution/validate_html.py src/pages/index.html

Review the output. Fix any failures. If the validation script itself
has a bug, fix the script, test it, and update directives/qa-checklist.md
with what you learned (self-annealing loop per AGENTS.md).
```

---

9. Phase 7: Build About Page

```
Read directives/build-page.md, directives/axiara-brand.md,
directives/content-source.md.

Run: python3 execution/generate_page_shell.py \
  --title "About — Axiara.ai" --slug "about" \
  --output src/pages/about.html

Content source: Manual Sections 1.1, 1.3, 6.1, 6.2.

SECTIONS:

1. HERO: H1 "AXIARA" (80px centered) + H2 "Every syllable is an
   operational mandate."

2. PHILOSOPHY: Three frosted glass cards explaining the etymology:
   AXI → AXIOM, AK → AKSARA, TERPADU. Full operational mandate text
   from Section 1.3.

3. FOUNDERS: Two-column asymmetric image cards.
```

Left: Erik Gunawan Supriatna (CEO, 45% equity, 51% voting,
operational & commercial authority).
Right: Hadi Hendrawan (CTO, 45% equity, veto on technical changes,
IP architecture authority).
Use placeholder image boxes (dark gray, "PHOTO" label).

4. CORE 3 TEAM: Three-column layout.
Lead Data Architect (Aksara, Phases 2+4)
| Compliance Lead (Axiom, Phases 3+5)
| Technical PM (Terpadu, Phases 4+5).

5. COMPANY: PT Aksara Inovasi Terpadu. Indonesian-incorporated.
Mission statement. Tagline.

Run validation after build.

---

10. Phase 8: Build Service Pages

Step 8.1 — 30-Day Sprint Page (Primary Conversion Page)

Read all directives.
Run execution/generate_page_shell.py for "30-Day Sprint".
Design reference: design-system/pages/sprint.md.
Content: Manual Sections 3.1 through 3.5.

HERO: H1 "THE 30-DAY SPRINT" / H2 "From Black Box to Glass Box in four weeks."

MAIN: Vertical timeline with crimson center line that "grows" on scroll.

WEEK 1 — "THE DIAGNOSTIC" (Node label: "WEEK 01" in font-code crimson)

Title: "Shadow AI Audit & Risk Mapping"
Badges: NIST: MAP | OWASP: Governance, Responsible AI | ISO: Clause 4
Pillar: Shield | Phase 1: Diagnose | Level 1-2 → Level 2
Activities: Shadow AI tool identification, data ingestion risk assessment,
output dependency mapping, regulatory cross-referencing.
Deliverables (sub-card): Landscape Report, Exposure Assessment,
Maturity Score, Executive Briefing.

WEEK 2 — "THE AKSARA FOUNDATION" (WEEK 02)
Title: "Data Sovereignty & Vector Core"
NIST: GOVERN | OWASP: Data Mgmt, Privacy | ISO: Clause 6-8
Pillar: Aksara | Phase 2: Architect | Level 2 → Level 2+
Activities from Section 3.2.1.
Deliverables: Migration Plan, Security Audit, Draft Logic Map,
Accountability Map draft.

WEEK 3 — "THE AXIOM TRANSITION" (WEEK 03)
Title: "Logic Mapping & Stress-Testing"
NIST: MEASURE | OWASP: Verification, Design | ISO: Clause 8-9
Pillar: Axiom | Phase 3: Illuminate | Level 2+ → Level 3+
The Logic Mapping Workshop breakdown:
  Glass Box Reveal (15min), Decision Trace (30min),
  Accountability Finalization (25min), DPO Red-Team (20min)
Deliverables: Logic Maps, Integrity Report, Accountability Map,
DPO Sign-Off.

WEEK 4 — "THE BLUEPRINT" (WEEK 04)
Title: "Integration & Transformation Roadmap"
NIST: MANAGE | OWASP: Governance, Operations | ISO: Clause 9-10
Pillar: Terpadu | Phases 4-5 Blueprint | Level 3+ → Level 4
Nexus Roadmap: Phase I (Months 2-6), Phase II (7-12), Phase III (Year 2)
Deliverables: Executive Blueprint, 24-Month Roadmap, ROI Projections,

Board Deck, MSA.

CTA: "START YOUR SPRINT →"

Run validation after build.

Step 8.2 — E2E Transformation Page

Build src/pages/services/e2e.html.
Content: Manual Sections 1.2, Appendix 8.3.

HERO: H1 "END-TO-END TRANSFORMATION"
H2: "Five phases. Zero vendor gaps. One accountable partner."

MAIN: Interactive Client Journey Map (horizontal scroll timeline):
Day 1-7: Pre-Engagement → Week 1: Diagnose → Week 2: Architect
→ Week 3: Illuminate → Week 4: Blueprint → Months 2-6: Integrate
→ Months 7-12: Scale → Year 2: Sustain

Each node is expandable (click/hover reveals detail from Appendix 8.3).

Step 8.3 — Aksara Vector Core Page

Build src/pages/services/aksara.html.
Content: Manual Section 2.1 (all subsections).

HERO: H1 "THE AKSARA VECTOR CORE" / H2 "Your sovereign data foundation."

MAIN: Vertical architecture flow diagram.
Each component = frosted glass card with crimson connecting arrows:
Legacy Data Connectors → Embedding Engine → PII Scrubbing Layer

→ Encryption Envelope → Vector Storage → RAG Pipeline

Each card reveals technical specs from Section 2.1.1 on hover.

---

11. Phase 9: Build Technology Pages

Step 9.1 — Episteme Page

Build src/pages/technology/episteme.html.
Design reference: design-system/pages/episteme.md.
Content: Manual Sections 2.2 through 2.2.3.

HERO: H1 "NTRJ EPISTEME" / H2 "The Glass Box. Not the Black Box."

SECTION 1 — SPLIT COMPARISON:
Left (muted, silver border): "Model Explainability (xAI)"
- SHAP plots, feature attribution, model internals
- "A research tool for data scientists"

Right (full crimson): "Workflow Transparency (Episteme)"
- Logic Maps, document traceability, human Logic Owners
- "A liability shield for the Board"

Pull quote: "A SHAP plot tells a data scientist which variable mattered most. An Episteme Logic Map tells a CRO which company document was used."

SECTION 2 — SIX CAPABILITIES (3×2 grid of glass cards):
From Section 2.2.1: Input Visibility, Context Control, Reasoning Steps, Output Tracking, Decision Lineage, Human Oversight.

SECTION 3 — ZERO-DISRUPTION: From Section 2.2.3. Episteme works around
existing AI tools, no model replacement needed.

Step 9.2 — Terpadu Engine Page

Build src/pages/technology/terpadu.html.
Content: Manual Sections 2.3 through 2.3.2.

HERO: H1 "THE TERPADU ENGINE" / H2 "From lab experiment to operational organ."

Four glass cards:
1. ERP/CRM API Bridge (SAP, Oracle, Salesforce, Odoo, Jubelio)
2. Agentic Workflow Orchestrator (multi-step autonomous execution)
3. Event-Driven Trigger System (real-time CRM/ERP/IoT listeners)
4. Feedback & Drift Monitor (continuous accuracy monitoring)

Include the E2E Technical Stack table from Section 2.3.2.

---

12. Phase 10: Build Industry & Compliance Pages

Step 10.1 — Industry Pages (Reusable Template)

First, create a reusable industry page directive.
Add to directives/build-page.md under "Templates":

Industry Page Template Structure:
1. Hero (industry H1 + icon)
2. "Strategic Pain Point" section (what keeps the CxO awake)
3. "Axiara E2E Entry Thesis" section (how we solve it)

4. "Sprint Application" section (how the 30-Day Sprint applies)
5. CTA: "Explore the 30-Day Sprint for [Industry] →"

Then build four pages using this template.
Content: Manual Section 1.4 (exact row text for each sector).

1. src/pages/industries/telco.html — Telecommunications
   Icon: Lucide "radio-tower"
   Pain: AI routing without auditable logic
   Thesis: Episteme Glass Box for routing + CRM integration

2. src/pages/industries/energy.html — Energy & Utilities
   Icon: Lucide "zap"
   Pain: Life-safety AI decisions without traceability
   Thesis: Vectorized safety protocols + human approval chain

3. src/pages/industries/banking.html — Banking & Insurance
   Icon: Lucide "landmark"
   Pain: Automated credit/claims without governance
   Thesis: Glass Box for financial AI + regulatory compliance

4. src/pages/industries/bumn.html — State-Owned Enterprises
   Icon: Lucide "building-2"
   Pain: Presidential Decree compliance + public accountability
   Thesis: ISO 42001 certification + local PT advantage (Section 5.6)

Run validation on each page after build.

Step 10.2 — Compliance Pages

Build three compliance pages.
Content: Manual Sections 5.5 and 7.

1. src/pages/compliance/iso-42001.html
   H1: "ISO 42001 CERTIFICATION"
   Three-way comparison: NIST (voluntary, not certifiable) vs
   EU AI Act (binding, European) vs ISO 42001 (certifiable, global)
   Axiara's 30-Day Sprint as certification accelerator.

2. src/pages/compliance/uu-pdp.html
   H1: "UU PDP COMPLIANCE"
   Data residency, PII protection, consent management.
   How Aksara Vector Core satisfies all requirements.

3. src/pages/compliance/ai-decree.html
   H1: "2026 PRESIDENTIAL AI DECREE"
   Formal governance mandate for automated decisions.
   How Episteme provides audit trails and Logic Owners.

All pages: frosted glass cards, crimson accents, ghost CTAs.
Run validation after each build.

---

13. Phase 11: Build Contact Page

Read directives/build-page.md, directives/axiara-brand.md.
Design reference: design-system/pages/contact.md.

Build src/pages/contact.html.

HERO: H1 "BEGIN YOUR TRANSFORMATION" / H2 "CEO-to-CEO engagement starts here."

FORM (centered, max-w-lg):

Input styling: bg glass, border 1px silver/30%, focus border crimson,
text white, font-eng. Border-radius: 0px. Labels: font-code uppercase.

Fields:
1. Full Name (text)
2. Title / Role (text)
3. Company Name (text)
4. Email (email)
5. Phone (tel)
6. Primary Interest (select): 30-Day Sprint | E2E Transformation
   | Technology Assessment | ISO 42001 Certification | Custom Inquiry
7. Message (textarea, 4 rows)

Submit: Ghost button "SUBMIT →" (full-width)
Below: "PT Aksara Inovasi Terpadu | axiara.ai" (font-code 11px silver/60%)
"All engagements begin with a Board-level governance assessment."

No map. No social icons. Form only. Zero distraction.
Run validation after build.

---

14. Phase 12: Responsive, Accessibility, & QA

Step 12.1 — Run the QA Directive Across All Pages

Tell the agent:

Read directives/qa-checklist.md.

Run execution/validate_html.py against every page in src/pages/.
Compile results into .tmp/qa-report.md.

For any failures:
1. Fix the page
2. Re-run validation to confirm fix
3. If the failure revealed a pattern (e.g., a common mistake),
   update the relevant directive under "Self-Annealing Notes"
   so it never happens again (per AGENTS.md self-annealing loop)

Step 12.2 — Responsive Testing

Review all pages at 375px, 768px, 1024px, 1440px.

Fix these common issues:
- Hero H1 scaling: 40px mobile → 56px tablet → 72px desktop
- All grids collapse to single column below 768px
- Ghost buttons: min 48px tap target on mobile
- Horizontal timelines → vertical on mobile
- Image offsets removed on mobile (translate(0,0) below 768px)
- Navbar → hamburger below 1024px
- No horizontal scroll at any breakpoint

If you discover a responsive pattern that should be standardized,
update directives/axiara-brand.md under "Self-Annealing Notes"
AND update execution/validate_html.py to check for it.

Step 12.3 — Accessibility Audit

```
Audit all pages for accessibility:
- alt text on all images/icons
- <label> on all form inputs
- Color contrast ≥ 4.5:1 (test Crimson on Obsidian for small text)
- Keyboard tab navigation on all interactive elements
- Visible focus states (crimson outline)
- prefers-reduced-motion disables all animations
- Semantic HTML (heading hierarchy, ARIA landmarks)
- Skip-to-content link
- <html lang="en">

Update directives/qa-checklist.md if any new checks should be added.
```

---

15. Phase 13: Export & Deploy to BeTheme

Follow `directives/deploy-betheme.md` for the complete deployment SOP.

Step 13.1 — Transfer Global CSS

Copy `src/css/axiara.css` into BeTheme → Appearance → Theme Options → Custom CSS/JS → CSS. This single paste applies all Darkroom styling globally.

Step 13.2 — Configure BeTheme Options

Following the directive exactly: Full Width layout, Transparent Sticky header, Fade In Up animations (Slow), all default fonts disabled, custom Google Fonts via CSS import.

Step 13.3 — Transfer Pages Section by Section

For each page, open the Antigravity HTML and Muffin Builder side by side. Map HTML `<section>` to

Muffin "Section" blocks, `<div>` wraps to "Wrap" blocks, columns to "Column" blocks. Paste content, apply CSS classes where needed.

Step 13.4 — Install WordPress Plugins

Per the deployment directive: WPForms/Gravity Forms, WP Rocket/LiteSpeed Cache, Yoast/Rank Math, Smush/ShortPixel.

Step 13.5 — Final Verification

Run the full QA checklist one more time in the live WordPress environment. Verify fonts load from Google CDN, frosted glass renders across browsers, and all responsive breakpoints work.

Appendix A: Complete CSS Variables

```css
```

```
:root {
    --axiara-obsidian: #0D0D0D;
    --axiara-crimson: #C41E3A;
    --axiara-white: #FFFFFF;
    --axiara-silver: #A8A8A8;
    --axiara-charcoal: #1A1A2E;
    --axiara-gradient: linear-gradient(135deg, #0D0D0D 0%, #1A1A2E 100%);
    --axiara-glass: rgba(26, 26, 46, 0.4);
    --axiara-glass-hover: rgba(26, 26, 46, 0.8);
    --axiara-glow: rgba(196, 30, 58, 0.05);
    --axiara-wire: rgba(255, 255, 255, 0.1);
    --font-eng: 'Outfit', sans-serif;
    --font-phil: 'Playfair Display', serif;
    --font-code: 'JetBrains Mono', monospace;
    --transition-smooth: all 0.4s ease;
    --transition-fast: all 0.2s ease;
    --section-padding: 120px;
    --card-padding: 40px;
    --content-max-width: 1200px;
}
```

Appendix B: Component Library Reference

Ghost Button

```
html
```

```
<a href="#" class="inline-block px-8 py-4 border border-white
   text-white font-code text-xs uppercase tracking-[2px]
   hover:border-crimson hover:text-crimson hover:bg-[rgba(196,30,58,0.05)]
   transition-all duration-[400ms] cursor-pointer">
    BUTTON TEXT →
</a>
```

Frosted Glass Card

```
html

<div class="bg-[rgba(26,26,46,0.4)] backdrop-blur-[12px]
   border-l border-l-crimson p-10
   hover:bg-[rgba(26,26,46,0.8)] hover:border-l-4
   transition-all duration-[400ms] cursor-pointer">
    <!-- content -->
</div>
```

Pre-headline Label

```
html

<span class="font-code text-xs text-crimson uppercase tracking-[3px]">
    LABEL TEXT
</span>
```

Stat Number

```
html
```

```
<div>
  <span class="font-eng font-black text-6xl text-crimson">99.9%</span>
  <p class="font-eng font-light text-silver mt-2">Description</p>
  <p class="font-code text-[11px] text-silver/60 mt-1">Sub-label</p>
</div>
```

Framework Badge

html

```
<span class="inline-block px-3 py-1 border border-silver/30
  font-code text-[10px] text-silver uppercase tracking-wider">
  NIST: MAP
</span>
```

Asymmetric Image

html

```
<div class="relative">
  <div class="absolute inset-0 border border-white/10
    translate-x-5 translate-y-5 z-0"></div>
  <img src="..." alt="..." class="relative z-10 grayscale" />
</div>
```

Appendix C: Self-Annealing Playbook

This appendix documents the self-annealing process from AGENTS.md as it applies to the Axiara project. When something breaks, follow this loop:

Step 1 — Read the error. Look at the exact error message and stack trace. Do not guess at the cause.

Step 2 — Fix the script. If the error is in an execution script, fix the Python code. If the error is in the HTML/CSS output, fix the generation logic.

Step 3 — Test the fix. Run the script again. Run the validation script. Confirm the fix works.

Step 4 — Update the directive. Add what you learned to the relevant directive under "Self-Annealing Notes." For example:

- "backdrop-filter requires -webkit prefix for Safari" → add to `directives/axiara-brand.md`
- "Tailwind CDN v3.4 requires `<script>` tag with config, not CSS import" → add to `directives/build-page.md`
- "Crimson (`#C41E3A`) on Obsidian (`#0D0D0D`) has contrast ratio 4.2:1 — fails WCAG AA for text below 18px. Use white for small text, crimson only for large text and accents." → add to `directives/qa-checklist.md`

Step 5 — The system is now stronger. The same mistake will never happen again because the directive now encodes the lesson. Over 15+ pages, this compounds into a highly reliable build process.

Common Annealing Patterns for This Project:

Error Pattern	Likely Fix	Update Which Directive
backdrop-filter not working in Safari	Add -webkit-backdrop-filter	axiara-brand.md
Font not loading	Check @import URL, verify font names	axiara-brand.md
Border-radius found on element	Remove rounded-* class, add rounded-none	qa-checklist.md
Emoji found in UI	Replace with Lucide icon	axiara-brand.md
Animation too fast/bouncy	Change to fade-in-up 0.8s ease-out	axiara-brand.md
Content doesn't match manual	Re-read correct section, fix text	content-source.md
Image offset breaks mobile layout	Add md:translate-x-5 (responsive)	build-page.md
Heading hierarchy skip (H1→H3)	Insert missing H2	qa-checklist.md

---

Appendix D: All Directive Files (Summary)

File	Purpose	When the Agent Reads It
directives/axiara-brand.md	Visual rules, colors, fonts, motion, components	Before building ANY page or component
directives/build-page.md	Page structure, build process, verification steps	Before building each page
directives/design-system.md	How to run ui-ux-pro-max, conflict resolution	Before generating design system

File	Purpose	When the Agent Reads It
directives/qa-checklist.md	All quality checks, responsive, accessibility	After building each page
directives/deploy-betheme.md	WordPress/BeTheme transfer process	During deployment phase
directives/content-source.md	Page-to-manual-section mapping, voice rules	When writing any page content

Complete Workflow Summary

```
PHASE 0   Setup project structure (folders, reference docs, .env)
PHASE 1   Create all 6 directives (Layer 1 — the SOPs)
PHASE 2   Create execution scripts (Layer 3 — deterministic tools)
          → generate_css.py, generate_page_shell.py, validate_html.py
PHASE 3   /plan → generates docs/PLAN-axiara-website.md
PHASE 4   Run ui-ux-pro-max → generates design-system/MASTER.md + pages/
PHASE 5   Build global foundation (CSS, navbar, footer, Tailwind config)
PHASE 6   Build homepage (7 sections, one prompt each)
PHASE 7   Build about page
PHASE 8   Build service pages (sprint, e2e, aksara)
PHASE 9   Build technology pages (episteme, terpadu)
PHASE 10  Build industry (×4) + compliance (×3) pages
PHASE 11  Build contact page
PHASE 12  QA: responsive + accessibility + brand compliance
PHASE 13  Export → deploy to WordPress + BeTheme
```

```
THROUGHOUT: Self-annealing loop on every error
        (fix → test → update directive → system gets stronger)
```

The agent reads directives before every task, calls execution scripts for deterministic work, writes creative code for the variable parts, validates after every page, and self-anneals when anything breaks. Each failure makes the system more reliable for the remaining pages.

— END OF IMPLEMENTATION GUIDE v2.0 — PT Aksara Inovasi Terpadu | Axiara.ai Foundational Logic. Integrated Innovation.