ARCHITECTURE NOTES · RESOURCE

The five-phase Webflow CRO architecture

A reference companion to the build that runs on /websites-cro. Audit, design system, build, experimentation, manage. What each phase ships and what gets skipped at your peril.

Companion to: The five-phase Webflow CRO architecture we ship to every client.

A Lighthouse 100 score is the table-stakes deliverable, not the work

The interesting work starts after the page is fast. The Architecture Build phase ships the page. The experimentation layer is what decides whether the page earns its place in the funnel. This deck is the section-by-section reference for the published five-phase build, with the trade-offs and metrics named out loud. Use it before scoping a CRO engagement so you know what you are buying and what you should refuse to skip.

- For founders and brand operators above 30k a month in revenue scoping a Webflow CRO build.
- Each phase has a specific deliverable, a specific failure mode, and a specific reason it cannot be moved later in the order.
- Numbers cited are pulled directly from the article. No fabricated case studies.
- Use this as the reading guide before you read the architecture note end-to-end.

The five phases, in the order they have to run

01 **Phase 1: Audit and frame**

Before any pixels move. The current site, analytics, heatmaps, support tickets, and sales-call transcripts where customers describe what they were trying to do when they hit the page. Output is a written hypothesis stack with three to five testable bets, each tied to a measurable

02 **Phase 2: Design system setup**

A reusable component library inside Webflow the operator can manage without engineering involvement. Brand tokens (color, type, spacing) defined once and referenced everywhere. Component variants for cards, buttons, hero sections. CMS templates for collection pages.

03 **Phase 3: Build and tracking**

The site itself, instrumented from the first pixel. Heatmap, scroll-depth, click tracking on every conversion-load-bearing page. Event tracking on every meaningful state change. Lighthouse 100 enforced on real-world conditions, not synthetic fast-laptop runs.

04 **Phase 4: Experimentation layer**

The phase that distinguishes a CRO build from a redesign. Each test runs through a structured shape: hypothesis written before the test, sample size calculated up front, win and loss criteria defined before reading any data. Wins roll forward. Losses update the hypothesis stack.

05 **Phase 5: Manage and improve**

The ongoing engagement. Monthly funnel review, quarterly hypothesis-stack review, ad-hoc work on whatever shifted in the brand context. The page does not get redesigned unless the data demands it. Iterative changes accumulate inside the existing architecture.

Why Phase 1 is the cheapest phase and the most often skipped

What teams do when they skip the audit

- Open Figma and start moving pixels.
- Write a hypothesis stack after the redesign ships.
- Anchor decisions on what the founder thinks looks dated.
- Treat heatmaps and sales transcripts as nice-to-haves.
- Discover the conversion drop only after the launch settles.

What teams do when they run the audit first

- Start with the hypothesis stack, three to five testable bets.
- Anchor the visual decisions on what the data and transcripts say.
- Treat the audit output as the spec the design and build phases serve.
- Skip redesigns where the data does not support the bet.
- Ship redesigns that look better and convert better, not one or the other.

The experimentation layer, in the shape every test runs through

01 **Hypothesis written first**
A single sentence with the change, the audience, and the expected effect. Written before the design moves. Pinned to the hypothesis stack from Phase 1.

02 **Sample size calculated up front**
Before the test ships, the team agrees on how long it has to run and how many conversions per arm it needs. The article cites 800 conversions per arm as the floor below which the test produces noise that masquerades as signal.

03 **Win and loss criteria locked**
Both criteria written before any data is read. No moving the goalpost mid-test. Tests that meet the win criteria roll forward into the production page.

04 **Cadence held at three to four per quarter**
The realistic cadence at this stage, per the article. The cap comes from statistical power, not from team capacity.

05 **Losses update the stack**
A loss is a documented update to the hypothesis stack, not a quiet abandonment. The stack is the durable artifact across quarters.

The build and tracking phase, instrumented from the first pixel

- **Heatmap, scroll-depth, and click tracking on every conversion-load-bearing page.**
  Wired before the page goes live, not after the conversion drop arrives.
- **Analytics events for every meaningful state change.**
  Form started, form completed, CTA hovered, video played past 30 seconds.
- **Conversion attribution wired into the brand existing analytics stack.**
  No parallel attribution model. The brand owns its own truth.
- **Lighthouse 100 enforced on mobile 4G simulated, mid-tier device.**
  Synthetic 100 on a fast-laptop run is reported as a different number in the build doc.
- **Design system tokens referenced, not hardcoded.**
  Brand color, type, spacing defined once and referenced everywhere.
- **CMS templates wired for collection pages the operator owns post-launch.**
  Phase 2 work that makes Phase 5 cheap.

When this build is wrong, named out loud

When the structural work does not pay back

- Brand revenue is below the 30k a month threshold the article cites.
- The team wants a redesign and is calling it CRO.
- There is no appetite for losses inside the experimentation layer.
- The design system is treated as polish rather than as Phase 5 leverage.
- Audit findings get shelved because they conflict with a Figma direction.

When the structural work pays back fastest

- Revenue is past the threshold and the funnel has measurable holes.
- The team can articulate the testable bets before the build.
- Engineering will not be the bottleneck on operator-led changes post-launch.
- The brand is willing to hold tests at three to four per quarter.
- Losses are treated as updates to the stack, not failures to bury.

What the article documents as the published outcome band

LIGHTHOUSE TARGET	CONVERSION UPLIFT BAND	TEST CADENCE	SAMPLE SIZE FLOOR	ENGAGEMENT THRESHOLD
100	**20-40%**	**3-4 / quarter**	**800 / arm**	**~30k / month**
Enforced on real-world conditions (mobile 4G simulated, mid-tier device),	Published outcome on the Architecture Build engagement, per the	Realistic cadence cap, set by statistical power, per the article.	Below this, the test produces noise that masquerades as signal,	Brand monthly revenue floor below which the structural work does not

A pre-engagement gate before scoping the build

- **Confirm the brand is past the 30k a month revenue threshold.**
  Below the threshold the structural work does not pay back inside a reasonable horizon, per the article. The page says so out loud.
- **Confirm the team can articulate three to five testable bets.**
  If the hypothesis stack does not exist after the audit pass, the experimentation layer has nothing to run.
- **Confirm the existing analytics stack is the source of truth.**
  No parallel attribution model. Conversion attribution wires into the brand existing stack at Phase 3.
- **Confirm appetite for documented losses inside the experimentation cadence.**
  Tests that miss criteria are documented and update the stack. A team that cannot publish losses internally cannot run this layer.
- **Confirm the design system will be operator-managed post-launch.**
  The Phase 5 cost depends on it. Phase 2 is what makes manage cheap, per the article.
- **Confirm Lighthouse 100 is enforced on real-world conditions.**
  Synthetic 100 on a fast-laptop run is a different number and is reported as such in the build doc.

NEXT STEP

A CRO build is the experimentation layer, not the redesign

Run the audit first. Hold the experimentation cadence. Let the design system carry the manage phase. Skip Phase 1 and the rest of the build will work harder for less.

Read the full architecture note ->