KOOKIES

GUIDE

Webflow Build Standards

6 CHAPTERS

NAVIGATOR

- section_hero
  - container-large
    - hero_content
      - heading-style-h1
    - button is-primary
  - hero_image

STYLE

Class	heading-style-h1
Font	var(--display)
Size	var(--h1)
Colour	var(--ink)

Linked to variables

WEBFLOW

Build Webflow sites your team can keep running.

A good Webflow build starts from a brief with named owners and a design that passes a readiness check. It uses one written class system, variables, components and a planned CMS model. Motion stays light and respects reduced motion, the team can publish without help, and each launch check has proof. This guide sets out those standards, chapter by chapter.

Who it is for. Marketing leads, designers and Webflow developers who are planning a build, handing a design over, or inheriting a project and want a standard to hold it to.

01 Brief and Figma handover

02 Client-First, classes and variables

03 Components and CMS model

04 Interactions that stay fast

05 Editor handover

06 Pre-launch checks

Read online at www.kookies.io/guides/webflow-build-standards. Updated 2026-11-07.

CHAPTER 1 OF 6

Brief and Figma handover

01

A build should start from decisions, not guesses. A one-page brief, a pass or fail gate for the Figma file, and what to do when the design arrives in another form.

The short answer

Before a Webflow build starts, you need three things: a one-page brief with named owners, a design file that passes a readiness gate, and real content for the pages that matter most. If one is missing, the build can still start, but only when someone has written down who fills the gap and by when.

The aim is to find open questions while they are cheap. A mobile layout decided in Figma takes minutes. The same decision made in Webflow, after classes and components exist, means rework. If you want the full list of build stages, our Figma to Webflow checklist covers them. This chapter covers one moment: the go or no-go before anyone opens the Designer.

The one-page build brief

Put these answers on one page. Each one changes how the site is built, which is why they come before the first class is named.

QUESTION	WHAT IT CHANGES IN THE BUILD	A GOOD ANSWER
Who signs off?	How questions get closed	One named person, plus a backup
Which pages are in scope?	Build order and estimate	A list of page names, with final and draft marked
What repeats, and how often?	Components or CMS	"Two or three case studies a month, written by marketing"
Who edits after launch?	Roles, props and how much is locked	"Two marketers: copy and images, and new landing pages"
Does this replace a live site?	URLs, redirects and SEO work	Current URL, plus who owns the redirect list
Where do forms send?	Form setup and tests	A named inbox, CRM or sheet for each form
What needs tracking?	Events, consent and scripts	A short list of events, and who reads them
Which tools connect?	Custom code and integrations	Tool names, with an owner who has the login

DECISION RULE

If two people give different answers to "who signs off?", stop there. Every other open question waits on that one.

The Figma readiness gate

Go through the file against this table and mark each row pass or fail. "Ready" has a plain meaning here, so nobody has to argue about taste.

ITEM	READY MEANS	IF IT FAILS
Page status	Final pages are marked. Old ideas sit on a separate Figma page.	Designer marks them. Build waits on that page only.
Phone layout	Every page has a phone frame, or a written rule for how sections stack.	Designer decides the hard sections. Simple stacking can be agreed in the brief.
Text styles	Named styles cover every heading and body size.	List the one-off sizes. Designer merges or confirms them.
Colours	Named colours, with no near-duplicate greys.	Same as text styles.
Spacing	A visible scale, with sections using it.	Agree a scale before the build. See chapter 2.
Auto layout	Sections and cards use it, so intent is readable.	Ask about the sections that are hard to read.
States	Hover, focus, error and empty states are drawn for buttons, forms and lists.	Agree one rule per element, then apply it everywhere.
Repeated parts	Cards, buttons and nav are Figma components, used the same way each time.	Point out the differences. Are they on purpose?
Motion	Each animation has a note or a reference.	Static build first. Motion is added once it is agreed.
Assets	Icons export as SVG. Photos are high resolution. Fonts have a web licence.	The licence is a blocker. The rest can follow.
Real content	Long titles and real names on key pages.	Test key layouts with the longest real text you have.

Three results, three actions

A gate is only useful if the result changes what happens next. Score the file, then act on it.

RESULT	WHAT IT LOOKS LIKE	WHAT HAPPENS
Ready	Every row passes, or the gaps are small.	Start the build.
Ready with gaps	A few rows fail, each with an owner and a date.	Start with the pages that pass. Keep a shared list of gaps.
Not ready	No phone thinking, no styles, or pages still changing.	Hold the build. Send one list back to the designer, and agree a new start date.

DECISION RULE

Send gaps back as one list, once, grouped by page. Twenty questions in one message get answered in an afternoon. Twenty separate pings slow down everyone.

When the design is not a Figma file

Not every build starts from a neat file, and a different start is fine as long as you are clear about what the source is for.

WHAT YOU HAVE	TREAT IT AS	FIRST STEP	REAL EXAMPLE
A finished Figma file	The source of layout and style	Run the gate above	Petavue came with a finished design, built page by page, on desktop, tablet and phone.
A set of fixed layouts for one page	A rule the build has to follow	Ask what happens to items the layouts do not cover	Bailey shared three works grid layouts in Figma, and new projects get placed in rows automatically.
HTML prototypes	The source of layout and behaviour, not of code	Map its fonts, colours and scripts to the site's own system	CUBE sends prototypes that we rebuild as native Webflow on their existing style guide.
Content and no design	The content sets the layout	Agree the order of sections, and one class system	Baraka Impact was a single page, shaped around their content by hand in the Designer in about two hours.

The last row shows the point of a gate: it should fit the job. A single launch page does not need eleven rows of Figma checks. It does need a sign-off, final words, a title, a description and a social image.

Content: what must be final

Waiting for every word is the wrong rule, and so is starting with none. Sort content by how much it can change the layout.

CONTENT	NEEDED BEFORE THE BUILD	CAN FOLLOW
Home and key service headings	Yes, or a near-final draft	
Pricing tables and comparison tables	Yes. Rows and columns shape the design.	
CMS fields	Yes, a sample of three real items	The remaining items
Legal pages		Final text, before launch
Blog back catalogue		Import once the CMS is agreed
Alt text, titles and descriptions	A plan and an owner	The final words, checked in chapter 6

Freeze the design, and agree how changes work

Once a page is being built, changes to it have a cost. That does not mean no changes. It means one simple route for them.

- **One list.** Changes go in one shared place, with no side messages.
- **Marked in Figma.** Changed frames get a visible note, with the date in the note.
- **Sorted by size.** Copy and image swaps go in right away. A new section or a new template is a scope question, for the person who signs off.

- **Figma stays true.** If a decision gets made in Webflow, record it in the file or the list, so the two do not drift.

DONE WHEN

Start the build when the brief has no blank owners, the gate has a result, and there is a date beside every gap. If you want us to run this review on your file, send it through our Figma to Webflow page.

TAKEAWAYS

- One page of answers comes before the first class: owners, scope, editors, forms and tracking.
- Mark the Figma file pass or fail, row by row. Each failed row needs an owner and a date.
- A font without a web licence is a blocker. Most other gaps can follow.
- HTML prototypes are a source for layout and behaviour, not code to keep.
- Size the gate to the job. A single launch page needs far fewer checks.

CHAPTER 2 OF 6

Client-First, classes and variables

02

Anyone should be able to open the project and know what a class does before they click it. The structure, the naming rules, the stacking limits and how variables fit in.

The short answer

Use one written class system from the first section. For new Webflow builds, our default is Client-First, the free naming system by Finsweet: readable names, custom classes by default, and variables for any value that repeats. Set up the variables first, build a style guide page, and then build pages from those parts.

The test is simple. A new developer opens the project in a year and can tell what a class does, and what else it will change, before touching it. If a site already has a working system, that system wins. The last section before the review says when.

The page skeleton

Every page starts from the same few layers. This is the default structure in the Client-First core structure docs:

`page-wrapper` > `main-wrapper` > `section_[name]` > `container-[size]`

LAYER	JOB	OUR RULE
`page-wrapper`	Wraps everything on the page	Style it lightly, if at all
`main-wrapper`	Holds main content in a `main` tag	Nav and footer stay outside it
`section_[name]`	Wraps one section, and holds its padding	Use a `section` tag and a name that says what it is, such as `section_pricing`
`container-[size]`	Centres content and sets the max width	Use small, medium or large. Do not invent a fourth width for one section.

The docs also describe an alternative, with `padding-global` and `padding-section-[size]` as a separate layer. Either is fine. Pick one per project, and write down which.

Naming rules

These follow the Client-First class docs. The examples use the pattern on a made-up pricing section.

RULE	WRITE THIS	NOT THIS
Custom classes have an underscore. The part before it is the folder.	`pricing-card_title`	`Heading 3 copy`
Name what the element is, in full words.	`team-list_headshot-wrapper`	`tl_img-wr`
Utility classes have no underscore.	`text-size-large`	`text_size_large`
Word order runs general to specific.	`text-size-large`	`large-size-text`
Combo classes start with `is-` .	`button is-secondary`	`button secondary-2`
Name by role, not by look or by page position.	`pricing-card_badge`	`purple-box-left`

Our additions: no default Webflow names such as `Div Block 47` survive to launch, and folder names match the section they belong to. The folders docs say each underscore adds a nested level, and warn against nesting without a reason. One level is usually enough.

Custom class, utility or combo?

The Client-First stacking docs make custom classes the default for most elements, because they are quick to make and safe to change. Use this table to choose.

SITUATION	USE	EXAMPLE
An element with its own job in a section	A custom class	`pricing-card_title`
The same size or style needs site-wide control	A utility on top of a custom class	`pricing-card_title` + `text-size-large`
A variant that keeps most of its base styles	A combo class	`button is-secondary`
A variant that shares almost nothing with its base	A new custom class	`container-large` , not `container is-large`
One property used once, such as `position: absolute`	Put it on the custom class	Do not make a global utility for it

CLASSES ON ONE ELEMENT	CLIENT-FIRST GUIDANCE
1 or 2	Common
3	Ask if it is needed
4	The absolute maximum
5 or more	Too many. Make a custom class.

DECISION RULE

Never add a style while several global classes are stacked on an element. That quietly creates a new combo class. Stop, name it on purpose, or move the style to a custom class.

Variables come before classes

The stacking docs put it simply: use variables for repeated values, and classes for sets of properties. A class can read its values from variables. So the variables go in before the first section. Webflow groups variables in collections, and its Designer API docs show that one variable can be an alias of another of a compatible type. We use that for two tiers.

TIER	HOLDS	EXAMPLE NAMES	WHO USES IT
Brand	The raw values from Figma	`purple-500` , `ink-900` , `white`	Other variables only
Role	A job, pointing at a brand value	`text-primary` , `background-primary` , `border-subtle` , `button-background`	Classes

Now a rebrand is one edit in the brand tier, and a dark section changes role values only. The names above are our convention, so write yours down and keep it.

Webflow also has variable modes, where one variable holds several values, in sets such as a theme. Automatic modes attach to a breakpoint and switch on at that size. Client-First uses this approach for spacing: its spacing docs say values live in a Layout collection, with breakpoint modes, so spacing classes adapt without extra breakpoint classes. Check the current docs for mode limits before you plan around them.

Spacing: one method per job

Spacing is where projects drift first. The spacing docs offer several methods, so agree which one does which job.

JOB	USE	CLASS EXAMPLE
Space above and below a section	Section padding variables, or the padding section classes	`padding-section-large`
Even space between items in a flex or grid parent	A gap class	`gap-medium`
Space between two siblings in a stack	A spacing wrapper around one of them, or an empty spacing block	`margin-bottom-medium` , `padding-bottom-medium`
A one-off inside a component	On that component's custom class	`pricing-card_title`

Size names run from `xxtiny` to `xxhuge` . Map your Figma scale to those names once, and keep the mapping on the style guide page. If a design asks for a size not on the scale, ask if it is on purpose before adding one.

When to keep the system that is there

A class system is a means to an end. Mixing two is worse than keeping either.

SITUATION	WHAT TO DO
New build, no system	Use Client-First. Petavue runs on it, so its team can add pages from the same parts.
New pages on a site with a working style guide	Keep it. On CUBE we link every colour, heading and space to their existing variables, so new pages match the rest of the site.
Small one-page site	Keep it light. Baraka Impact needed one clean class system, not a large library.
Messy inherited site	Decide in the open: clean up in stages, or rebuild. Do not start a second system in a corner.

A class review before anything ships

- ✓ No default Webflow class names, and no names ending in numbers or "copy".
- ✓ Each section uses the agreed skeleton and has a descriptive `section_` name.
- ✓ No element has 5 or more classes, and any with 4 has a reason.
- ✓ Colours, type sizes and main spacing read from variables, with no hand-typed values for brand colours.
- ✓ The style guide page shows the type, colours, buttons, spacing and main components.
- ✓ A style change on one card updates all cards. Try it, then undo it.
- ✓ Unused classes are reviewed and removed.

DONE WHEN

Someone who did not build the site can find a section by name in the Navigator, change a button style in one place, and predict what that change will affect.

TAKEAWAYS

- ✓ Pick one written class system from the first section. Client-First is our default for new builds.
- ✓ Custom classes first. Add utilities where site-wide control matters, and combos only for true variants.
- ✓ Three classes is a question. Five is a sign you need a custom class.
- ✓ Set up variables before classes, with brand values and role values as two tiers.
- ✓ If a working style guide already exists, build on it. Never run two systems side by side.

CHAPTER 3 OF 6

Components and CMS model

03

The biggest structure choice in a build: should this be a component, a CMS collection or a one-off section? Decision rules, field standards, and real project examples.

The short answer

Use a component when the design repeats and the content is set by hand, a CMS collection when you have many items that need their own page, or need to be filtered or created from data, and a plain section when something truly appears once. Plan these choices on paper before you build, with three real items for each collection.

Get this wrong and the site still looks fine on launch day. The cost shows up later, when one small change needs dozens of manual edits. Our Webflow CMS guide explains collections, fields and references from the start. This chapter gives the rules for choosing between them.

Component, CMS or section?

QUESTION	IF YES	REAL EXAMPLE
Does it need its own URL, per item?	CMS collection with a template page	ClanX role pages all come from one CMS template.
Will pages be created from data, by a script or an API?	CMS collection	ClanX creates role pages by sending data to the CMS API.
Will people filter or search a long list of it?	CMS, usually	Petavue CMS lists use filters and load more.
Does the same item show in several places?	CMS, with references	For example, case studies, resources and people live in collections and show wherever they belong.
Is it one design used with different words and images?	Component with props	The same revamp builds every section as a component.
Will the owner add cards without a CMS?	Component with props, with code doing the rest	Bailey adds cards and sets a category prop. The filters build themselves from those categories.
Will a page get copied and filled in for each use?	Page built from components	Bailey duplicates a noindex outreach page and fills it in through props.
Does it appear once, ever?	A plain section	A one-off hero.

DECISION RULE

Ask who adds the next one, and how often. Frequent items added by a marketer belong in the CMS. A rare card added by the owner can be a component. Both are valid. Bailey is a good reminder that the CMS is not always the right answer.

Component standards

- **Name it by what it is.** `Pricing card`, not `Section 7`.
- **A prop for each thing that changes between uses.** Heading, text, image, link and visibility of optional parts.
- **Write prop names for the editor.** `Button label`, not `txt_1`.
- **Decide what is global on purpose.** Webflow says marketers and content editors can edit unbound text, links and images inside a component, and those changes reach every instance. If a value must differ per page, bind it to a prop.
- **Nav, footer, buttons and CTA blocks are components** from day one.
- **Show each one on the style guide page,** with its longest likely content.

A useful test: drop the component on a blank page, fill in the props with long, awkward text, and shrink the window down to phone width. If it breaks, fix the component, not the page it is on.

Plan the CMS on paper first

Do this before you make any collection. Changing a model after items are in is slower than getting it close the first time. Here is a filled-in example, for a case study collection.

FIELD	TYPE	WHY	NOTE FOR EDITORS
Name	Plain text	Card title, and the default title	Keep under 60 characters
Slug	Default	The URL	Do not change after publishing without a redirect
SEO title	Plain text	Differs from the card title	Unique per item
SEO description	Plain text	Search and social	One or two plain sentences
Client logo	Image	Card and page header	SVG if possible, with alt text
Summary	Plain text	Cards and lists	One sentence
Body	Rich text	The story	Headings start at H2
Service	Reference	Filter and related lists	Pick one from the list
Featured	Switch	Home page list	Turn on for no more than the list shows

Write a note on every field that could confuse someone, and keep the note in the field's help text where Webflow lets you.

Field standards

CHOICE	USE THIS	BECAUSE
Short fixed list that rarely changes	Option field	No extra collection to look after
A list that has its own details or its own page	Reference field	One source for that thing, such as an author with a photo and a bio

CHOICE	USE THIS	BECAUSE
Many of them per item	Multi-reference	Tags and related services
Text that needs headings, links or lists	Rich text	Editors can format it safely
Text that must fit a design	Plain text	Editors cannot add a heading that breaks a card
Sort order set by hand	Number	Clear and easy to change

On limits, the Webflow CMS API guide says a collection schema can have up to 60 fields, along with the default name and slug. Reference field limits and item counts depend on your plan and can change, so check the current docs before you commit to a model that needs a lot of them. Nearing a limit is usually a sign that one collection is doing two jobs.

Lists, filters and load more

On Petavue, CMS lists use Finsweet Attributes for filters and load more, so the team does not have to maintain custom code. If you use them, these points come from Finsweet's own docs for List Filter and List Load.

- Filters use `fs-list-element` and `fs-list-field` attributes, and the values on the filter and on the item must match.
- They do not run in Design, Edit or Preview mode. Test on the published site after each change.
- Load More needs native "Paginate Items" turned on, and items per page sets how many load per click.
- Render All can hurt performance on large collections, especially on phones. Avoid it on big lists.
- Interactions will not fire on newly loaded items unless you add the reset attribute, which also restores elements to their initial state. Test it.
- Loaded pages are cached until you republish. If content changes through the API without publishing, read the cache setting.

Use the current version of Attributes, and read its docs for each feature rather than copying old snippets.

Pages from data

ClanX shows the pattern: role data goes in as CMS items, and one template turns each into a full landing page, so a design change on the template updates every role page. Our post on Webflow CMS API landing pages covers the how. These are the standards to hold to.

- Design the template with the longest and the shortest real items first.
- Each page must say something unique. Check that fields hold real differences. A swapped job title is not enough.
- Create items as drafts first, and read a few as a person before they go live. The API guide covers staged and live items, and draft flags.
- Decide who owns the data source, and what happens to a page when a role closes.

- Rate limits and publishing rules can change, so check the current docs before you plan a large run.

Check the model before filling it

Load three real items, and then try the awkward cases.

- An item with no image. An item with a very long title. An item with an empty optional field.
- Empty lists show something sensible, or the block hides itself.
- A related list shows the right items, and does not include the current page.
- A marketer can add one item from start to finish, without help.

DONE WHEN

A person who was not on the build can add an item and read back the rule for what goes where. If you would like us to plan the model, that is part of our Webflow CMS work.

TAKEAWAYS

- CMS for items with their own URL, filters or data sources. Components for one design with different content. Plain sections for true one-offs.
- Ask who adds the next one, and how often. That usually settles it.
- Bind anything that changes per page to a prop. Unbound component content changes everywhere.
- Model the CMS on paper with three real items before you create a collection.
- Attributes only run on the published site, so test there after each change.

CHAPTER 4 OF 6

Interactions that stay fast

04

Every animation should earn its place. A ladder for choosing the lightest tool, a one-line spec for each interaction, reduced motion rules, and the numbers to test against.

The short answer

Give each animation a job, use the lightest tool that can do it, and write down what it does on a phone and with reduced motion turned on. Animate transform and opacity where you can, never hide the main content while a page loads, and test with real numbers on a real phone.

Our post on keeping Webflow interactions fast explains the performance rules, and why they matter. This chapter is how to decide and document motion during a build, so it stays under control as the site grows.

Spend motion where it helps

A site with motion everywhere has no focus, and it costs more to test. Decide where it goes before you build it.

GOOD REASONS	WEAK REASONS
It explains how something works	The design felt a bit plain
It shows a change of state: open, loading, sent	Every section fades in because the first one did
It draws the eye to one product moment	A competitor has it
It carries the brand, on a site where motion is the work	A library was already loaded

On a content-heavy site, keep custom animation to the moments that help, such as the hero, the proof cards and the product visuals, and leave the rest as plain CSS. Bailey is the other end of the scale. A creative studio sells motion, so the works page has page transitions, a custom cursor and smooth scrolling. Both are right for what the site is for.

The tool ladder

Start at the top. Move down only when the step above cannot do the job.

STEP	TOOL	GOOD FOR	COST
1	CSS transitions	Hover, focus, simple state changes	Lowest. No script of your own.
2	Webflow interactions	Reveals, tabs, menus, scroll effects editors can see in the Designer	Low to medium. Stays editable without code.
3	Custom GSAP in your own script	Sequences and timing that the visual tools cannot express	Medium. Someone has to maintain the code.

STEP	TOOL	GOOD FOR	COST
4	Heavier plugins, Lottie, video or 3D	One moment that is the point of the page	High. Needs a size limit and a fallback.

Webflow also offers Interactions with GSAP in the Designer. Its update note lists support for breakpoints, prefers reduced motion settings, and page scoping, which controls the pages an interaction loads on. For most new work, check the current docs and compare this with Classic interactions before choosing.

DECISION RULE

If the team will need to change it without a developer, keep it in the Designer. On CUBE, copy, logos and steps in the home page interactions change in the Designer without touching code. That was a build requirement, not a nice extra.

A one-line motion spec

For each interaction, write one row. It takes a minute, and it means the designer, developer and tester are all checking the same thing.

FIELD	EXAMPLE: HOME PAGE TABS
Job	Show four product areas without a long page
Trigger	Auto-advance, with a timer bar. Stops on click.
What moves	Panel opacity and a small transform, timer bar scale
Tool	Webflow interaction, or a small script if timing needs it
On phones	Same logic, swipe enabled, no hover
Reduced motion	No auto-advance. Tabs change on click, with no slide.
JS off or failed	First panel visible, with its text readable
Done when	Keyboard works, no layout shift, smooth on a mid-range phone

The example is made up, though auto-playing tabs with timer bars are a real thing on the CUBE home page. Keep all rows in one place, such as the handover document in chapter 5.

Standards for every interaction

STANDARD	WHY	SOURCE
Prefer transform and opacity	They are the compositor-only properties, so they skip layout and paint	web.dev
Use `will-change` sparingly	Each layer uses memory, which hurts low-memory devices	web.dev
Keep the main heading and hero image visible at load	An element hidden until scripts finish adds render delay	web.dev

STANDARD	WHY	SOURCE
Never lazy-load the LCP image	It starts the request later than needed	web.dev
Reserve space for anything that animates in	Late content shifts the layout	Our rule
Load heavy scripts only on pages that use them	Other pages should not pay for one effect	Our rule
Offer a non-gesture route	On the CUBE calculator, seat counts can be typed as well as dragged	Our rule

Reduced motion is part of the build

The prefers-reduced-motion setting tells a site that a visitor wants less non-essential motion, for example because of a vestibular disorder. Decide per type, up front.

TYPE OF MOTION	WITH REDUCED MOTION ON
Fade or slide on scroll	Show content in place, with no movement
Auto-playing tabs or sliders	No auto-advance. Keep user controls.
Looping background video	Show a still poster, with a control to play
Page transitions	A plain page change
Smooth scrolling libraries	Turn off, and use native scrolling
Hover colour changes	Keep. These are fine.

In Interactions with GSAP, use the reduced motion settings. In custom scripts, check the media query in code before you start any timeline. Then test it for real, by turning the setting on in your device or browser.

Heavy media and custom code

On Bailey, every page is full of large video, so the pages needed a plan for it. Video and images are served from Bunny CDN, video plays on hover, and animation runs on GSAP with ScrollTrigger and SplitText, plus Lenis for scrolling. Your site may not need any of that, but the same questions apply.

- **Why is it here?** Name the job, using the one-line spec.
- **What is the fallback?** Every video has a poster. Every 3D scene has a still.
- **When does it load?** Off-screen media waits. The one hero element does not.
- **Who hosts it?** Large video does not belong in a plain background upload if a proper host would serve it better.
- **Who maintains the script?** Keep custom code in one known place, with comments and a named owner.
- **What is the licence?** Check the terms for any library or plugin, and check again when you add one.

If animation still matters after launch, a short note in the handover saying which script drives which effect will save your next developer a day.

Test against numbers

Google's Core Web Vitals have published targets, and web.dev says to measure them at the 75th percentile of page loads, split by mobile and desktop.

METRIC	GOOD IS	HOW MOTION HURTS IT
LCP	2.5 seconds or less	A hero hidden by an interaction
INP	200 milliseconds or less	Heavy scripts during taps and scrolling
CLS	0.1 or less	Elements that move others as they appear

- ✓ Run Lighthouse on mobile on each template. Our build target is 95 or above. That is a target, not a promise.
- ✓ Run it with and without the animated sections, if you are unsure what the effect costs.
- ✓ Scroll the page on a real mid-range phone, with no cache.
- ✓ Turn on reduced motion, and check again.
- ✓ Use the keyboard to reach each control, and see it in focus.
- ✓ Publish, and test on the published site. Field data takes time to arrive, so keep checking after launch.

DECISION RULE

If an effect is the reason the page misses its target, it gets simpler, or it goes. Nothing is exempt. The site has to work first. Our Webflow speed guide and Core Web Vitals on Webflow cover what to try.

TAKEAWAYS

- ✓ Each animation needs a job. Write down its trigger, phone behaviour and reduced motion behaviour.
- ✓ Use the lightest tool that can do the job. Keep what editors need to change in the Designer.
- ✓ Prefer transform and opacity. Keep the hero visible at load, and never lazy-load the LCP image.
- ✓ Reduced motion is a build requirement, tested on the device, not a late add-on.
- ✓ Test with numbers on a real phone. An effect that makes a page miss its target gets simpler or goes.

CHAPTER 5 OF 6

Editor handover

05

A handover is finished when your team has published something without help. Choosing roles, building for editors, a safe and ask-first table, and the handover pack.

The short answer

Handover starts during the build, not at the end. Give each person the right Webflow role, expose what they should change as component props and CMS fields, lock what they should not, write a short guide, and then have them publish a real change while you watch. A handover is finished when they can do it without you.

That last part is the one people skip. A recorded tour feels complete, but only a real edit shows whether the site can be run by the people who own it.

Choose the right role

Webflow's guide to building and editing content describes the first two roles below. Roles and plans can change, so check the current docs before you set up seats.

PERSON	ROLE	WHAT IT IS FOR
Copywriter, content manager, translator	Content editor	Updating content, translating, and updating site assets
Marketer who builds landing pages	Marketer	Drag-and-drop page building from pre-built, approved components
Designer or developer changing the system	A designer or site manager role	Changes to layout, classes and components

Give people the lowest role that does their job. It protects them as well as the site, since they can work freely without worrying about what they may break.

If you came from an older Webflow site, note that Webflow has announced the end of the legacy Editor, and it describes client seats as the replacement for clients of freelancers and agencies. Our Webflow Editor entry covers the basics. Check the current docs for the dates and for what applies to your plan and workspace.

Build for the person editing

Most of the editing experience is decided while you build. These are the standards to hold to.

STANDARD	WHAT IT LOOKS LIKE	REAL EXAMPLE
Pages are made from parts	New pages use existing components, with no new classes	For example, your team adds pages from components.

STANDARD	WHAT IT LOOKS LIKE	REAL EXAMPLE
Repeat use is cheap	Duplicate the page, fill the props, publish	Bailey duplicates the outreach page for every pitch, with client name, video and links set through props.
Rules live in the content	Tag a card, and the filter updates by itself	Bailey's filters build from each card's category.
Effects do not need a developer	Copy, logos and steps change in the Designer	The same is true on CUBE.
Bulk work has a route	New pages come from sending data	ClanX adds role pages through the CMS API, with no Designer work per page.
Simple sites stay simple	One clean class system on one page	Baraka Impact is easy for the team to edit later.

Webflow notes that unbound content in a component changes across all instances, and that the canvas shows when an edit is global. Tell editors what that notice means before they first see it. Webflow also has settings for restricting what editors can change on an element, and for whether they can publish. Check the current docs for how those work on your plan.

What editors can do alone

Put this table in the guide, using your own site's details. It removes most of the "am I allowed to?" messages.

SAFE TO DO ALONE	ASK FIRST	LEAVE TO A DEVELOPER
Change copy, images and links	Change a page slug, because it needs a redirect	Change classes or variables
Add or edit CMS items	Delete a page or a CMS item with a URL	Change custom code or scripts
Build a page from approved components	Unpublish a page that gets traffic	Edit interactions
Change SEO titles and descriptions	Change a form's destination	Change the CMS field structure
Add alt text	Add a new tracking script	Change DNS or hosting settings

DECISION RULE

If a change could break a URL, a form, tracking or the design on other pages, it is not a solo edit. Everything else should be, so the team does not wait on anyone for small things.

The handover pack

Keep it short enough that people will read it, and in a place the team already uses.

ITEM	WHAT IT HOLDS
One-page guide	How to log in, how to edit, how to publish, the table above, and who to ask
Style guide page	Live in the project: type, colours, buttons and components, each with its name
Component list	Each component, what its props do, and where it is used
CMS list	Each collection, who owns it, and a filled-in sample item
Motion list	The specs from chapter 4, and which script drives which effect
Integrations list	Forms, tools and scripts, with where each one sends data and who holds the account
Access list	Who holds what: Webflow, DNS, analytics, Search Console, and any CDN or API host. Names only, no passwords in the document.
Short recordings	Editing a page, adding a CMS item, and publishing

If you are on a client's workspace, agree ownership before launch: who pays for the site plan, and who owns the workspace once the build ends.

The publish test

Run this live with the person who will do the work. They drive. You watch and stay quiet.

1. Change a line of copy on the home page, and publish.
2. Swap an image, and add alt text.
3. Add a CMS item with an SEO title and description, and find it on the live site.
4. Build a short landing page from components.
5. Find the right person to ask about a slug change.
6. Undo a mistake, or say how they would get it fixed.

Each time they hesitate, write down why. It is usually a poor prop name, a missing field note, or a gap in the guide. Fix those, because they are bugs in the handover. They are not the editor's fault.

Avoid the common handover problems

WHAT HAPPENS	WHY	STANDARD
Editors break a layout with long text	The design was only tested with short copy	Test components with the longest real content
A heading in rich text breaks page structure	Field notes are missing	Say which heading levels to use, and where
Two people overwrite each other	No agreed process	Agree who edits what. Check the current docs for how Webflow handles two people editing at once.

WHAT HAPPENS	WHY	STANDARD
A publish sends half-done work live	Nobody agreed who publishes, or when	Name who publishes. Check the current docs for your plan's publishing options.
A slug changes, and old links return 404	Nobody knew it needed a redirect	Put it in the ask-first column, and link the redirect map tool.
A new tool gets added and speed drops	Scripts are added with no owner	New scripts are an ask-first item, and they get listed.

After the handover

Plan for who looks after the site next, before the project ends. Our work includes 30 days of support after launch for safe changes: copy, image, colour and link updates, and fixes to what we built. It does not cover new sections or redesigns. After that, you can run the site yourself using the guide, hire Webflow development by the hour or the project, or use a care plan.

Our post on what comes after launch lists the kind of upkeep to plan for.

DONE WHEN

The person who will run the site has published a real change, on their own, using the guide. Petavue was a one-time build, handed over ready for the team to edit. That is the standard for a project with no ongoing plan.

TAKEAWAYS

- Handover is designed during the build. Props and fields decide what editing feels like.
- Give each person the lowest role that does their job, and check the current docs for what each role includes.
- Write down what is safe alone, what to ask first, and what is for a developer.
- If a change can break a URL, form, tracking or other pages, it is not a solo edit.
- Handover is finished when the team publishes a real change without help.

CHAPTER 6 OF 6

Pre-launch checks

06

The last gate. A table of checks, each with its proof, owner and severity, plus the Webflow traps worth testing by hand and a clear rule for when a problem blocks launch.

The short answer

A Webflow site is ready to launch when each check below has proof beside it, and no blocker is open. Proof means a result someone can look at: a crawl export, a form entry that arrived, or a Lighthouse run. "Looks fine to me" does not count.

Our website launch checklist is the full list for any platform, and it is worth copying. This chapter is the sign-off layer for a Webflow build: the table that ties the standards in this guide to a go or no-go decision. If the site replaces another one, the Site Move Playbook has the launch-day run sheet and the redirect work.

The sign-off table

Copy it into a shared sheet, and add a date and a link to the proof for each row.

AREA	CHECK	PROOF	OWNER	SEVERITY
Brief	Scope is met: every agreed page exists	Page list compared with the sitemap	Project lead	Blocker
Brief	Forms send to the agreed places	A real entry from each form arrives, confirmed by a person	Project lead	Blocker
Build	Matches the design at each breakpoint	Side by side at desktop, tablet and phone sizes	Developer	Blocker on main pages
Classes	Chapter 2 review passes	Style list and Navigator read through	Second developer	Fix now if easy
Components	Long and missing content holds up	Test with awkward text, on a phone	Developer	Blocker on main pages
CMS	Templates work for every item type	Long, short and empty items checked	Developer	Blocker
CMS	Filters and load more work	Tested on the published site	Developer	Blocker if used
Motion	Hero visible right away	Slow network test, on a phone	Developer	Blocker
Motion	Reduced motion works	Device setting on, pages read through	Developer	Blocker
Speed	Core Web Vitals targets, and our Lighthouse target	Lighthouse on mobile per template	Developer	Fix or log

AREA	CHECK	PROOF	OWNER	SEVERITY
SEO	Unique titles and descriptions, one H1	Crawl export	SEO owner	Blocker on main pages
SEO	Staging blocks are gone from live	Crawl for `noindex`, and read robots.txt	SEO owner	Blocker
SEO	Canonicals and sitemap agree	Crawl compared with the sitemap	SEO owner	Blocker
SEO	Schema is valid per template	Rich Results Test	SEO owner	Fix or log
Move	Every old URL has an answer	Whole old URL list tested	SEO owner	Blocker
Tracking	Events arrive after consent	Real-time or debug view	Project lead	Blocker if reported on
Access	Keyboard reaches every control	Tab through main pages	Developer	Fix or log
Access	Images have alt text, and text has contrast	Crawl export, and a contrast check	Developer	Fix or log
Content	No placeholder or test wording	Search of the built pages	Content owner	Blocker
Handover	Publish test passed	Editor publishes a real change	Project lead	Blocker

Blocker, fix now, or log

Without a rule, the last week becomes a debate about every small thing. Agree this before the checks start.

SEVERITY	MEANS	EXAMPLES	WHAT HAPPENS
Blocker	Loses leads, breaks main pages, hides the site, or tells a lie	A form that does not send. A `noindex` on live. Wrong prices.	No launch until it is fixed
Fix now	Visible, and quick to fix	A broken link. A cropped image. A class left with a default name.	Fix on the day it is found
Log	Real, small, and does not touch the main journey	A minor spacing issue on a rare screen width	Goes on a list, with an owner and a date

DECISION RULE

One person decides the severity when people disagree: the person named in the brief as signing off. Every logged item gets a date. A list with no dates is just a nicer way of forgetting.

Webflow traps to check by hand

These are the ones a general checklist tends to miss, because they come from how the tools work.

TRAP	WHY IT HAPPENS	CHECK
Attributes work only once published	Finsweet says they do not run in Design, Edit or Preview mode	Test every filter, search and load more on the published site

TRAP	WHY IT HAPPENS	CHECK
Interactions stop on loaded items	Items added by load more can miss interactions, according to Finsweet	Press load more, then hover and click the new items
Old cached items after API changes	Finsweet says loaded documents are cached until republish	Republish, or read the cache setting if your data comes in through the API
A global component edit reached too far	Unbound content changes everywhere	Read all pages that use a recently edited component
CMS template with thin data	The template looks good with the sample item	Open the emptiest item on the live template
Hidden elements still load	Something hidden on a breakpoint can still be in the page	Check what the phone downloads, not only what it shows
Staging settings carried over	Password or `noindex` settings stay on until someone changes them	Crawl the live domain, and check robots.txt on launch day
Custom code in the wrong scope	A script for one page is set site-wide	List each script, where it loads, and why

Run it on the published site

Designer previews and the live site are not the same test. Run the final pass on a published staging address such as the `webflow.io` one, then again on the real domain, because forms, scripts and security rules can behave differently there.

- ✓ Publish to staging, with no cached session.
- ✓ Check the site on a real phone, on a mobile connection.
- ✓ Keep staging out of search results, and check the current docs for how to do this on your plan.
- ✓ Check `www` and non-`www`, and that one sends cleanly to the other.
- ✓ After launch, repeat the form, tracking and `noindex` checks on the real domain.

For search and answer engines, use our free tools for quick checks on key pages: the meta tags tool, the schema generator and the AEO score checker. For a deeper pass, see the Webflow SEO checklist, our schema guide and accessibility basics.

Two reviewers, for two kinds of blindness

The builder cannot see what they already know. Use a second pair of eyes, and split the job.

REVIEWER	LOOKS FOR	WORKS FROM
Second developer	Class names, components, code, breakpoints, speed	The Designer, a crawl and Lighthouse
Someone outside the build	Words, flow, wrong facts, buttons that surprise them	A phone, with no briefing

Give the outside reviewer two tasks, such as "find what this costs" and "ask for a call". Watching someone try is more useful than a list of opinions. On smaller builds, one person can do both, as long as a little time has passed since building.

Sign off, and keep a record

Keep the record short. It answers the question "what did we check, and who said yes?" when something odd turns up in three months.

- ✓ The sign-off table, with dates and proof links.
- ✓ The list of logged items, with owners and dates.
- ✓ A dated export of titles, descriptions and URLs.
- ✓ The Lighthouse results by template, as a before picture for later.
- ✓ A note of which scripts run, where, and why.
- ✓ The name of the person who said go, and when.

Know how to restore an earlier version of the site before launch day. Check the current docs for how backups work on your plan, and make sure the person on call can do it.

The first days after launch

Launch day is the start of the watch. In the first week, send a real form again, read the Search Console reports as they fill in, look through the logged items, and ask your team to report anything odd. The Site Move Playbook has a full 30-day schedule if URLs changed.

If you would rather have someone else run this gate, it is part of how we handle Webflow development and Figma to Webflow projects. You can also see how the steps fit together on our process page.

DONE WHEN

Every row has proof, no blocker is open, every logged item has a date, and one named person has said go.

TAKEAWAYS

- ✓ Each check needs proof someone can look at. "Looks fine" is not a result.
- ✓ Agree blocker, fix now and log before the checks start, with one person who decides.
- ✓ Test Attributes, load more and interactions on the published site.
- ✓ Run the final pass on staging, then repeat key checks on the real domain.
- ✓ Keep a dated record of what was checked and who said go.

QUESTIONS

Questions people ask.

Do I have to use Client-First?

No. What matters is one written system that the whole project follows. Client-First is our default for new builds because it is free, well documented and readable. If your site already has a working style guide, build on that, as we do for CUBE.

What should be ready before a Webflow build starts?

A one-page brief with named owners, final pages marked in the design, phone layouts or stacking rules, named text and colour styles, licensed fonts, and real content for key pages. Chapter 1 has the full gate.

Can a Webflow build start without a Figma file?

Yes. CUBE sends HTML prototypes, and Baraka Impact had content and no design. The source changes, but the need for clear decisions and one class system does not.

When should I use the CMS instead of a component?

Use the CMS when items need their own URL, need filters, or are created from data. Use a component when one design takes different content and the items are added by hand. Ask who adds the next one, and how often.

Does motion make a Webflow site slow?

It can, depending on how it is built. Keep the hero visible at load, prefer transform and opacity, load heavy scripts only where they are used, and test on a real phone. Our post on interactions and performance goes deeper.

How do we stop editors from breaking the site?

Give each person the lowest role that fits, put what changes into props and CMS fields, write down what needs asking first, and run a live publish test. Chapter 5 has a table you can copy.

Can we use this guide with our own developer?

Yes. It is written so that any team can use it. If you want us to do the build or review one, see Webflow development.

Keep going

- Figma to Webflow
- Webflow development
- Webflow CMS
- Webflow care plan
- Figma to Webflow checklist
- Keep Webflow interactions fast
- Website launch checklist
- Webflow CMS guide
- The Site Move Playbook
- Petavue case study
- CUBE case study
- What is Client-First?

Planning a Webflow build? Book a free 20 minute call

Kookies. www.kookies.io. hello@kookies.io