



TAPFLARE / RESEARCH & PRACTICAL GUIDANCE

Who Is Responsible for Website Accessibility? A WCAG 2.2 Brief

Tapflare is a flat-rate monthly subscription for graphic design and website development. A dedicated project manager assigns each request to senior designers and developers, so companies get steady creative and web output without hiring.

Published by Tapflare · 2026-10-31 · who is responsible for website accessibility · website accessibility responsibility · wcag 2.2 checklist · wcag 2.2 level aa · accessibility acceptance criteria · outsourced web development · accessibility roles and responsibilities · agency vs client accessibility · website accessibility contract clauses

Original article: <https://tapflare.com/articles/who-owns-website-accessibility-wcag-2-2-brief>



In this article

1. Executive Summary
 2. Introduction and Background
 3. Key Changes in WCAG 2.2 That Matter for Commissioned Work
 4. Who Owns What: Roles, Supplier Deliverables and the Business's Continuing Duty
 5. Implementation Considerations and Process Changes
 6. Data Analysis and Evidence
 7. Implications and Future Directions
 8. Frequently Asked Questions (FAQs)
 9. Conclusion
-

Executive Summary

Website accessibility is a shared job that no purchase order can hand to one party. The World Wide Web Consortium (W3C) draft Accessibility Roles and Responsibilities Mapping (ARRM) states that **different aspects of accessibility belong to different roles** (www.w3.org), and W3C's organizational guidance says site owners remain responsible for ensuring accessibility or providing accessible alternatives, including for third-party content (www.w3.org). For a marketing lead commissioning a landing page, the practical reading is that a supplier can deliver design and code, a content owner must supply meaning (such as what an image is for), and the business that publishes the page keeps the duty to maintain it after every edit.

The technical yardstick is **WCAG 2.2** (Web Content Accessibility Guidelines), a W3C Recommendation republished on 12 December 2024 (www.w3.org). Level AA conformance means satisfying all Level A and Level AA criteria (www.w3.org). A brief should name the version, the level, the pages and the complete journeys, including embedded third-party parts, because conformance cannot be achieved by excluding part of a page (www.w3.org).

The added value proposed here is a responsibility and evidence matrix: each selected criterion is tied to a page or journey, owners, a test method and environment, an evidence artifact, and an issue and retest owner. A mockup cannot prove keyboard behavior; Chrome's developer documentation says the only way to find keyboard and screen-reader navigation errors is to use the page that way (developer.chrome.com). A developer cannot decide what an image means; WebAIM notes the same image may need vastly different alternative text depending on how it is used (webaim.org).

Evidence has limits. Lighthouse documentation says a score of 100 does not guarantee an accessible page (developer.chrome.com), and Deque, the maker of the axe-core engine, reports that automated testing identified 57.38% of issues by volume in its audit data from first-time audits, excluding items needing manual verification (www.deque.com). In WebAIM's February 2026 analysis, 95.9% of home pages had detected WCAG 2 failures (webaim.org). Review coverage (journeys checked versus planned) should be recorded as incomplete, never as a pass rate. This report is not legal advice; W3C states that its own policy pages are not legal advice (www.w3.org).

Introduction and Background

A marketing lead who buys a landing page from an outside designer and developer faces three questions that a generic checklist does not answer. Who supplies each accessible element? What must the commission include? What evidence justifies approving a release or asking for a fuller assessment? This report answers them from published standards and guidance, then adds a criterion-to-owner-to-evidence matrix for outsourced website work. It is a commissioning guide, not a claim that any brief guarantees conformance.

Three W3C resources frame the analysis. The WCAG 2.2 Recommendation is the normative authority for criteria, levels, exceptions and conformance scope. ARRM is explicitly a draft, and it states that it does not set requirements to meet WCAG (www.w3.org). The Easy Checks page, a preliminary review method, says more robust assessment is needed to evaluate accessibility comprehensively (www.w3.org). Because ARRM already supplies role definitions, this report attributes that structure to W3C and concentrates on what a buyer adds: scoped journeys, observable checks, separated design-stage and build-stage evidence, and an issue log.

General audit guides exist. A web design audit post from the design provider Superside treats accessibility as one of five audit steps and tells readers to check that text, images and interactive components are accessible to users with disabilities (www.superside.com). It does not allocate ownership by role or specify evidence per criterion, which is the gap addressed here. All version statements are anchored to October 2026, and the 2024 republication incorporated errata, so readers should check the Recommendation's errata link before fixing a contract to specific wording.

Key Changes in WCAG 2.2 That Matter for Commissioned Work

Version, Level and Scope: What a Brief Must Name

WCAG conformance is defined for web pages, not for a company or brand. A claim may cover one page or several related pages, and a claim is optional. Scope is therefore the first commissioning decision. Three conformance conditions shape a landing-page project:

- **Complete processes.** If a page is part of a multi-step process, all pages in the process must conform at the specified level or better (www.w3.org). A lead form followed by a scheduling step is one process.
- **Full pages.** Conformance cannot be achieved if part of a page is excluded (www.w3.org), so a chat widget or embedded form is inside the boundary.
- **Non-interference.** A widget that captures keyboard focus affects the whole page, not only itself. WebAIM calls keyboard accessibility one of the most important aspects of web accessibility (webaim.org).

"WCAG 2.2 Level AA, for the listed pages and journeys" is a testable sentence; "WCAG compliant" is not. Level AAA is generally not a sensible whole-site target, so a brief should say plainly that it is out of scope.

Focus Not Obscured (2.4.11, Level AA)

Success criterion (SC) 2.4.11 requires that a focused component is not entirely hidden due to author-created content (www.w3.org). The usual culprits are fixed elements added for conversion. W3C's Understanding document says a cookie banner implemented as sticky content fails if it entirely obscures the focused component (www.w3.org). The criterion tests only that the component is not *entirely* hidden; the stricter 2.4.12 is Level AAA, so a brief citing "2.4.11" does not promise that focus is never partly covered.

Fixes are implementation choices. MDN documents the CSS (Cascading Style Sheets) property scroll-padding, which exists to prevent scrolling content from being hidden by a fixed-position header (developer.mozilla.org). web.dev describes the criterion as ensuring the focus indicator is not hidden beneath other components (web.dev). Whether the fix works is visible only on the built page with real keyboard focus, which is why the matrix assigns this check to live testing, not to design review.

Target Size (Minimum) (2.5.8, Level AA)

SC 2.5.8 sets a base requirement that pointer targets are at least 24 by 24 CSS pixels, except when an exception applies (www.w3.org). The exceptions prevent a lazy universal rule; the Understanding page counts five (www.w3.org): **spacing** (undersized targets whose 24 pixel circles do not intersect another target), **equivalent** (another compliant control on the same page does the same job), **inline** (targets in a sentence), **user agent control** (browser-determined sizes the author has not changed), and **essential** presentation.

Do not confuse this with the Level AAA criterion 2.5.5, which asks for 44 by 44 CSS pixels. Platform guidance also exceeds the WCAG minimum: MDN recommends a minimum interactive size of 44 by 44 CSS pixels for buttons (developer.mozilla.org), web.dev suggests around 48 device independent pixels for touch targets (web.dev), Android accessibility help suggests 48dp targets separated by 8dp (support.google.com), and Apple's guidelines treat spacing between controls as important as size (developer.apple.com). Padding can bring a 24 pixel icon up to a 48 pixel target (web.dev). If a brief adopts a larger figure, label it a design preference stacked on WCAG 2.2 AA.

Accessible Authentication (3.3.8, Level AA)

Gated downloads, portals and trial sign-ups involve authentication. SC 3.3.8 says a cognitive function test, such as remembering a password or solving a puzzle, is not required for any step in an authentication process unless an exception applies (www.w3.org). Exceptions cover an alternative method, an assisting mechanism (password-manager support is a named example), and two that the AAA version omits: object recognition and personal content. MDN ties valid autocomplete values to the Level AA criterion on identifying input purpose (developer.mozilla.org), and adds that disabling autocomplete removes a feature that many users depend on (developer.mozilla.org), and that valid autocomplete values help users with cognitive disabilities and motor impairments fill out forms (developer.mozilla.org).

When a third-party identity service or CAPTCHA (a challenge to tell humans from automated programs) is embedded, the business owner usually chooses the vendor, so the matrix routes this criterion to a decision as well as a build task.

Other New Criteria and Level Traps

- **2.5.7 Dragging Movements (AA)**: dragging functions need a single-pointer alternative.
- **3.3.7 Redundant Entry (A)** and **3.2.6 Consistent Help (A)** apply to multi-step forms and repeated help links.
- **2.4.13 Focus Appearance** requires a focus indicator at least as large as a 2 CSS pixel perimeter (www.w3.org). It is **Level AAA, not AA**, and some secondary sources mislabel it.
- **4.1.1 Parsing** was removed from WCAG 2.2 as obsolete.

MDN warns that removing focus styles makes keyboard navigation inaccessible for sighted users (developer.mozilla.org), and WCAG 2.2 added nine criteria in total (www.w3.org). The `:focus-visible` pseudo-class lets authors restyle the indicator without changing when it appears (developer.mozilla.org).

Who Owns What: Roles, Supplier Deliverables and the Business's Continuing Duty

Starting from the W3C Roles Draft

ARRM exists because late accessibility work lands on the wrong people, and the GOV.UK Service Manual puts the principle plainly: everyone on the team is responsible for making the service accessible (www.gov.uk). When accessibility is left until late in a project, responsibility often falls on developers, who then handle tasks outside their skillset. ARRM lets teams assign responsibilities at the success criteria level or at the task level. A **primary** owner is accountable for a task, a **secondary** owner helps complete it, and **contributors** are consulted. The decision tree states there can only be one primary owner for each task (www.w3.org), and that ownership should potentially be revisited when tasks come from resources outside the primary team, which is exactly the situation of an outsourced build.

ARRM's roles map onto a small project. The one-line summaries below are the author's paraphrase unless a citation is given, and "business owner" is this article's label, not a separate ARRM role:

- **Content author**: the person who authors content is responsible for making that content accessible (www.w3.org).
- **Visual designer**: owns how the product looks and feels. This article's matrix also assigns this role contrast pairs and transparency effects, which can reduce contrast; that detail is the author's addition, not ARRM wording.
- **Front-end developer**: implements the design in functional templates, preferring the correct semantic HTML element over ARIA where one exists (developer.mozilla.org).
- **Tester**: builds manual testing on top of automated checks and reports findings without owning the design decisions.
- **Project manager and business owner**: put accessibility into estimates and approve trade-offs that touch brand.

Two cautions. Most decisions that affect accessibility are made before development, so developers should not be assigned everything. And one person may fill several roles, as when a single subscription designer also builds the page, so the matrix assigns roles to tasks, not people to titles.

Assigning Roles Before Quoting

The U.S. federal Section508.gov site, cited only as an example of how one procurement body structures such requests, says accessibility requirements belong at the beginning of the procurement lifecycle (www.section508.gov), asks buyers to state how, by whom, and when accessibility testing is conducted (www.section508.gov), and to request a plan for testing during the development life cycle (www.section508.gov). Pre-quote decisions and their usual owners:

- **Target and scope:** version, level, pages, complete processes (business owner, with the supplier confirming feasibility).
- **Content meaning:** image purposes, link wording, form instructions, error wording (content owner).
- **Visual rules:** contrast pairs, focus styles, target sizes, sticky behavior (visual designer; branding trade-offs sit with the business owner).
- **Build rules:** native elements, labels, focus management, ARIA (Accessible Rich Internet Applications) only where needed (front-end developer).
- **Evidence and acceptance:** who tests, where, and which artifact is delivered (tester and accepting owner).

Supplier Deliverables Versus the Business's Ongoing Responsibility

A supplier can be asked to deliver conforming templates, documented component behavior, and test evidence for the build as delivered. A supplier cannot reasonably guarantee pages the business later edits. UK government guidance says people who edit a site after launch also have a responsibility to make content and features accessible (www.gov.uk). Section508.gov states that technology must remain accessible throughout the contract period of performance (www.section508.gov); for a subscription or retainer, the analogue is that each new section, banner or campaign page is a new item for the same checks. Where the supplier provides a content management system (CMS) or builder, Section508.gov suggests asking for samples of accessible content that the authoring tools produced (www.section508.gov).

Third-Party Forms and Embeds

Embedded forms, schedulers and consent banners are where scope disputes arise. W3C advises authors who choose a third-party embed to select products that meet WCAG requirements and to monitor content that can change. For pages with uncontrolled parts, the Recommendation offers a statement that the page does not conform but would if the named parts from uncontrolled sources were removed (www.w3.org).

Vendor documentation is evidence about the vendor's product, not about the embedding page. Cloudflare's Turnstile documentation states the widget is WCAG 2.2 AA compliant ([developers.cloudflare.com](https://developers.cloudflare.com/turnstile/)), a vendor claim about the widget only; Calendly's accessibility statement says many experiences meet WCAG 2.1 AA and excludes user-generated content (calendly.com); Google's reCAPTCHA help page documents screen reader support (support.google.com) while noting some screen readers may have difficulty getting into forms

mode (support.google.com). None replaces a test on the real page. MDN notes any iframe needs a title that concisely describes the embedded content (developer.mozilla.org), and Google's reCAPTCHA developer documentation lets the embedding site set the widget's tabindex (developers.google.com).

Legal Responsibility: What a Brief Cannot Settle

Who carries legal liability when a site falls short depends on jurisdiction, sector and contract, and a technical report cannot answer it. W3C states its policy information is not legal advice (www.w3.org). As one illustration of how regimes differ, UK guidance for public sector bodies says the commissioning organization remains legally responsible for its website meeting accessibility requirements, and that organization and supplier should work together (www.gov.uk). That is a rule for one public sector regime, not a general principle; consult a qualified professional in the relevant jurisdiction. The US Section 508 and GOV.UK sources cited elsewhere in this article are used as procurement and practice analogies; they are not obligations for private-sector commissioners.

Implementation Considerations and Process Changes

Step 1: Record Target, Scope and Journeys

Open the brief with a scope table: version (WCAG 2.2), level (AA), page templates, complete journeys (landing page, form, thank-you state, scheduling embed), supported browsers, and third-party components. GOV.UK's audit guidance recommends testing a representative sample of page templates and content types (www.gov.uk) and telling auditors if any part was built by a third party (www.gov.uk). The same logic extends to complete processes: a sample that skips the thank-you state tests half a journey.

Step 2: Build the Responsibility and Evidence Matrix

Table 1 is an authored illustrative matrix for a hypothetical trial-signup landing page. It is not copied from any W3C table and is not exhaustive. Each row names the journey, criterion and level, owners, test method and environment, evidence, and who owns the issue and retest. Row owners follow ARRM's one-primary-owner logic, listed first.

Journey and element	Criterion (level)	Content / design / build owner	Test method and environment	Expected evidence	Issue / retest owner
Header and sticky promo bar	2.4.11 (AA); 2.4.7 Focus Visible (AA)	Content: business owner. Design: visual designer. Build: front-end developer	Keyboard-only walk with bar showing, at desktop and 320 pixel widths; staging URL; browser and version recorded	Recording plus tab-stop list	Developer / tester
Primary call to action	2.5.8 (AA); 1.4.11 Non-text Contrast (AA); 4.1.2 Name, Role, Value (A)	Content: content owner (label). Design: designer (size, contrast). Build: developer (native element)	Measure rendered size per state; activate with Enter and Space; confirm accessible name	Measurement note per state; accessible name screenshot	Developer or designer / tester

Journey and element	Criterion (level)	Content / design / build owner	Test method and environment	Expected evidence	Issue / retest owner
Lead form	3.3.2 Labels or Instructions (A); 3.3.1 Error Identification (A); 3.3.3 Error Suggestion (AA); 4.1.3 Status Messages (AA); 1.3.5 Identify Input Purpose (AA)	Content: content owner (instructions, error wording). Design: designer (layout). Build: developer	Submit empty and invalid; complete by keyboard; confirm programmatic labels and announced success message; one named screen reader and browser pairing	Recording of error and success paths; error strings	Developer (behavior) or content owner (wording) / tester
Hero image	1.1.1 Non-text Content (A)	Content: content owner decides purpose. Design: designer flags decorative images. Build: developer	Inspect rendered markup; content owner reviews wording in context	Image inventory with decision and text	Content owner or developer / tester with content owner
Text and interface colors	1.4.3 Contrast (Minimum) (AA); 1.4.11 (AA)	Design: designer owns pairs. Build: developer keeps tokens	Design-stage contrast table; live check of rendered states including focus and error	Table plus live measurements of sampled states	Designer or developer / tester
Embedded scheduler	2.1.1 Keyboard (A); 2.1.2 No Keyboard Trap (A)	Decision: business owner (vendor). Build: developer (titled iframe)	Tab into and out of widget; operate all functions by keyboard	Recording; vendor documentation on file; decision note	Business owner / tester
Sign-in step, if present	3.3.8 (AA); 1.3.5 Identify Input Purpose (AA)	Decision: business owner (method). Build: developer must not block paste or password managers	Sign in with a password manager and with paste; confirm an alternative method	Recording; note naming the alternative	Business owner or developer / tester

Supporting guidance exists for several rows. MDN notes that buttons displaying only an icon have no accessible name unless text is supplied (developer.mozilla.org); web.dev warns never to rely only on color to show an error or success state (web.dev); and MDN recommends explicit label-to-control association for compatibility with assistive technologies (developer.mozilla.org). For forms, GOV.UK's screen reader checks include filling in and submitting a form (www.gov.uk), and its advice is to test with the actual device or technology where possible (www.gov.uk).

Table 1 is a template, not a verdict. Its purpose is to remove ambiguity before work starts: every row has one party who must act when a check fails. Delete rows for features that do not exist, add rows as scope grows, and list conditional criteria such as 3.3.4 Error Prevention only when the page modifies user data or creates legal or financial commitments. The level column keeps AAA items such as 2.4.13 out of an AA contract by mistake.

Step 3: Receive Design-Stage and Build-Stage Evidence Separately

Refuse to accept one kind of evidence for the other. Design-stage artifacts show intent; build-stage evidence shows behavior. Designers can ensure that all fields have a descriptive label adjacent to the field (www.w3.org) and show error states in frames, but the label must also be programmatically associated: MDN explains that a screen reader will read out the label when the user is focused on the form input (developer.mozilla.org). Typical design artifacts are annotated frames, a contrast table, specified sticky behavior, target dimensions and error-state frames. Figma's help center describes annotations as a way to communicate information for developers to turn design into code (help.figma.com), and its contrast checker compares entered colors against WCAG AA and AAA thresholds (www.figma.com); neither inspects a built page.

Tooling vendors are open about the boundary. Stark, a design-plugin vendor, writes that handoff is where accessibility intent either survives or gets lost (www.getstark.co).

Stark also writes that only role and aria-label can be determined from design intent without runtime state (www.getstark.co); it leaves final judgement to the teams who hold the most context (www.getstark.co).

Chrome notes that visual order may differ from source order (developer.chrome.com), web.dev lists keyboard, visual and cognitive tests as manual testing (web.dev), Firefox can overlay the tabbing order on a page (firefox-source-docs.mozilla.org), but the overlay reflects the tab order at the time the option is selected (firefox-source-docs.mozilla.org), and Accessibility Insights describes its tab stops test as covering problems such as missing tab stops, keyboard traps and incorrect tab order (accessibilityinsights.io). Acceptance rules follow, and GOV.UK recommends testing code regularly with both automated and manual testing (www.gov.uk):

- **Design stage:** accept annotated frames and a contrast table, but not as proof of keyboard order, focus visibility or announced errors. GOV.UK notes rough prototypes need not have accessible code while design choices can still be checked early (www.gov.uk).
- **Build stage:** accept evidence from the built page on the agreed staging or production URL, with browser and version recorded. WebAIM calls keyboard testing an essential part of any accessibility evaluation (webaim.org), and web.dev adds the question of whether the keyboard focus indicator is always visible (web.dev).
- **Release:** accept a completed matrix, unresolved issues and a coverage statement.

Step 4: Assign Roles to Actions

Table 2 translates five roles into actions across the project timeline. It is an authored synthesis, not a W3C table.

Role	Before quoting	Design stage	Build stage	Release decision	After launch edits
Content owner	Lists images, forms, links, headings; names who writes alternative text and errors	Supplies copy that fits label and error space	Reviews alternative text and link wording on staging	Signs off wording rows	Applies the same rules to every edit

Role	Before quoting	Design stage	Build stage	Release decision	After launch edits
Visual designer	Confirms palette, sizes, sticky elements can meet the level	Delivers contrast table, focus style, state frames	Answers drift questions	Confirms rendered states match spec	Updates spec when brand changes
Front-end developer	Confirms native-element approach; flags third-party limits	Reviews design feasibility	Implements labels, focus, keyboard behavior, titled embeds	Fixes assigned issues	Fixes regressions
Tester	Agrees methods and environments	Notes what artifacts cannot show	Runs the matrix on staging; records evidence	Reports findings and coverage	Retests changed journeys
Accepting business owner	Sets version, level, scope; assigns owners	Approves brand trade-offs	Decides on third-party components	Accepts, defers, or requests further assessment	Owens content and any accessibility statement

Table 2 keeps the tester as a reporter rather than a decision maker, consistent with ARRM's position that testing is not a role that owns accessibility decisions.

Step 5: Three Fictional Review Cases (Hypothetical Examples)

These invented cases show how ownership resolves; they name no real organization.

Case A (Hypothetical Example): the sticky promo bar. Design frames show a promo bar fixed to the viewport top. In the build, tabbing through the lead form moves focus to a field that sits entirely behind the bar. The design file passed every contrast check and could not reveal this. Issue owner: front-end developer, who applies scroll padding (developer.mozilla.org) or removes stickiness on small screens. Retest owner: tester, repeating the keyboard walk at both widths.

Case B (Hypothetical Example): the hero image. The developer inserts the hero image using the file name as its text alternative because no purpose was supplied. Automated tests can tell whether alternative text exists, but WebAIM says the best alternative text depends on the context and intended content of the image (webaim.org), and it suggests asking whether important content would be lost if the image were deleted (webaim.org). Primary owner: content owner, who decides informative, functional or decorative. Secondary owner: developer, who implements the decision. The tester checks the inventory and does not choose the wording.

Case C (Hypothetical Example): the embedded scheduler. A third-party scheduling widget lets keyboard focus enter but not leave. Issue owner: the business owner, who chose the vendor. Options: ask the vendor for a fix, replace the widget, or exclude it from the claimed scope with a statement that the page would conform if the parts from uncontrolled sources were removed. The developer embeds it with a descriptive iframe title and documents the problem but cannot repair the vendor's code. Retest owner: tester, after the decision.

Step 6: Record Unresolved Issues, Remediation, Retest and Decisions

An acceptance record must survive staff changes and later edits. *Table 3* is a hypothetical log, with observed review coverage stated beneath it.

ID	Journey / element	Criterion (level)	Finding (Hypothetical)	Severity	Issue owner	Remediation	Retest	Decision
A-01	Lead form	2.4.11 (AA)	Focused field fully hidden by sticky bar	Blocking	Developer	Scroll padding	Pending	Release held
B-02	Hero image	1.1.1 (A)	Alternative text is a file name	Major	Content owner	Supply purpose and wording	Passed	Accepted
C-03	Scheduler embed	2.1.2 (A)	Focus enters but cannot leave	Blocking	Business owner	Vendor fix or replace	Not retested	Decision due
D-04	Footer links	2.5.8 (AA)	Adjacent links under 24 pixels with intersecting spacing circles	Minor	Designer	Adjust spacing	Passed	Accepted
E-05	Sign-in step	3.3.8 (AA)	Not reviewed	Unknown	Tester	Schedule review	n/a	Open

Observed review coverage (Hypothetical): 6 of 8 planned journeys reviewed; sign-in step and consent banner not yet reviewed. This describes the review plan only. It is incomplete, it contains two blocking issues, and it does not support a statement of conformance. No aggregate pass percentage is reported, because it would hide the blocking barriers. The same discipline appears in W3C's methodology, which says sample-based evaluation alone does not result in being able to make WCAG 2 conformance claims (www.w3.org).

If the business owner accepts a known barrier, the record should say so. An accessibility statement should list known limitations and give contact information, but such a statement is not a technical assessment or declaration of conformity.

Step 7: Decide When to Get a Fuller Evaluation and Involve Users

A short acceptance sample answers a narrower question than "is this site accessible?" W3C tells evaluators that tools cannot check all accessibility aspects automatically and that human judgement is required (www.w3.org), while Microsoft's Accessibility Insights says its assessment includes manual requirements where the tool provides instructions and the tester does the evaluating (accessibilityinsights.io). GOV.UK says detailed WCAG checking requires a large amount of technical expertise (www.gov.uk). Signals that a specialist evaluation is warranted:

- The business wants to publish a conformance claim or accessibility statement; GOV.UK says an audit gives the information needed for a statement (www.gov.uk).
- The scope includes sign-in, payment, multi-step processes or heavily scripted components.
- The record has unresolved blocking issues or large unreviewed scope.

When commissioning one, GOV.UK advises checking that auditors do not rely exclusively on automated tools (www.gov.uk), and Section508.gov advises buyers to conduct independent validation testing to verify vendor conformance claims (www.section508.gov), to evaluate each proposal to validate vendor claims against stated requirements (www.section508.gov), and to ask offerors to describe the evaluation methods they will use to validate conformance (www.section508.gov); it also says rigorous testing is required to validate vendor claims (www.section508.gov). It describes an Accessibility Conformance Report (ACR) as the document buyers use to assess a supplier's product (www.section508.gov).

Involving users with disabilities is complementary. W3C says user evaluation alone cannot determine if a website is accessible (www.w3.org), and its guidance is to fix significant barriers in an initial review before evaluating with users.

Commissioning Questions to Copy

These authored questions can be pasted into a request for quotation:

- Which WCAG version and level does the quote assume, for which pages and complete journeys?
- Who supplies or reviews alternative text, link wording, form instructions and error messages?
- Which design-stage artifacts will be delivered (contrast table, focus style, target sizes, sticky behavior, error frames)?
- Which build-stage evidence is delivered per matrix row, from which environment, with which browser and version?
- Which third-party embeds are proposed, what accessibility documentation does each vendor publish, and who resolves an issue one of them causes?
- What applies to edits after launch, including new sections and campaign pages?

Data Analysis and Evidence

Quantitative evidence falls into two groups: how common detectable failures are, and how much tools can see. Neither supports a pass-rate claim for a single project.

Prevalence. WebAIM's analysis of one million home pages (data from February 2026) found 95.9% with detected WCAG 2 failures (webaim.org), averaging 56.1 errors per page (webaim.org). Six error types accounted for 96% of detected errors (webaim.org). Low-contrast text appeared on 83.9% of home pages (webaim.org); 16.2% of home page images lacked alternative text (webaim.org); one third of form inputs were not properly labeled (webaim.org). WebAIM states that absence of detected errors does not indicate a page is accessible or conformant (webaim.org), and that, because only detectable failures were counted, full WCAG 2 A/AA conformance was certainly lower than 4.1% (webaim.org).

Complexity and third-party code. The average home page had 1437 elements (webaim.org). 82.7% of pages used ARIA (webaim.org); pages with ARIA averaged 59.1 detected errors (webaim.org), though WebAIM cautions the association does not mean ARIA introduced them (webaim.org). MDN's guidance is to prefer the correct semantic HTML element over ARIA (developer.mozilla.org). Most pages using a common CMS had fewer errors than average, but the effect of CMS choice varied widely (webaim.org). Ads were

among the strongest harbingers of errors (webaim.org). WebAIM warns these are correlations that cannot always be attributed to the technology (webaim.org). The implication for a buyer is modest: keep the embed list short and test the embeds.

What tools can see. Deque's coverage report found that 57.38% of total issues in its first-time audit data were identified by automated testing, excluding items needing manual verification (www.deque.com); Deque describes the report's sample as 13,000+ pages/page states and nearly 300,000 issues (www.deque.com); a related Deque blog post contrasted its result with the commonly cited 20 to 30 percent measured by success criteria (www.deque.com). These are vendor-originated figures from audit data, published on a page that shows no visible publication date, so treat their currency with caution.

The UK Government Digital Service built a test page with 143 deliberate barriers in 2017 and found that 42 were missed by all of the tools tested (accessibility.blog.gov.uk), comparing a testing tool to a spellchecker (accessibility.blog.gov.uk); that study is old and reflects the tools of its time.

Lighthouse documentation says automatic detection can only detect a subset of issues (raw.githubusercontent.com), and Microsoft's Accessibility Insights says most accessibility problems can only be discovered through manual testing (accessibilityinsights.io), and notes that its automated color-contrast rule cannot determine the ratio of text superimposed on an image or gradient (accessibilityinsights.io), and that flagged items must be reviewed by a human to determine whether they pass or fail (accessibilityinsights.io).

Deque states that the remaining coverage is achieved with manual testing (www.deque.com), and GDS concluded that automated tools cannot be relied on by themselves to check the accessibility of a website (accessibility.blog.gov.uk). Chrome notes that its Lighthouse panel and the axe extension generally provide the same information because axe is the underlying engine (developer.chrome.com).

Test environment. WebAIM's Screen Reader User Survey 11, run in July and August 2026 (webaim.org), found JAWS the most commonly used screen reader at 69.2% of respondents, outpacing NVDA at 59.3% (webaim.org), with JAWS and Chrome the most common pairing by a substantial margin (webaim.org). It warns the sample was not controlled and may not represent all screen reader users (webaim.org). Name the browser and assistive technology used. No hours-saved, conversion or legal-risk figures are offered, because no verifiable primary source supports them.

Coverage. The only calculation proposed is review coverage: journeys checked divided by journeys planned, reported with the count of unreviewed items and any blocking issues. In the hypothetical log, 6 of 8 describes the review plan, not a pass rate. W3C's methodology recommends evaluating the entire product where feasible.

Implications and Future Directions

For buyers. The matrix shifts the conversation from "is it compliant?" to "who does what, and how will we know?" It does not make a project accessible by itself, but it makes ambiguity visible before it becomes rework. Documented evidence per row is checkable; a badge or overall score is not. GOV.UK tells suppliers they must be accurate about how far a product meets accessibility standards (www.gov.uk).

For agencies and practitioners. Designers should annotate more and certify behavior less. Developers should default to native elements; MDN notes that replacing a button with a styled div loses native keyboard accessibility (developer.mozilla.org). Testers should record browser, version and assistive technology, because results depend on them.

Tooling. Accessibility Insights positions its FastPass as a check for common, high-impact issues in less than five minutes (accessibilityinsights.io) and says the best way to evaluate compliance is to complete an assessment (accessibilityinsights.io). WebAIM describes WAVE as facilitating human evaluation of web content (wave.webaim.org).

Open questions. How much evaluation a specific site needs depends on scope, complexity and legal context. Public data on how outsourced handoffs fail, as distinct from how pages fail in aggregate, is thin, so treat the role assignments as a hypothesis to refine against project records. Draft resources such as ARRM may change; copy the roles into the project record with a retrieval date.

Frequently Asked Questions (FAQs)

Who is responsible for website accessibility? Several parties, by task. W3C's draft guidance says different aspects belong to different roles, and the site owner remains responsible for third-party content or for providing accessible alternatives, as the body of this report sets out.

Is it the agency's or the client's responsibility? It depends on what the contract assigns. ARRM recommends one primary owner per task. In practice the agency owns the templates and components it builds, the client owns content authored after launch (www.gov.uk), and third-party component decisions need an explicit owner.

What is new in WCAG 2.2? Nine additional success criteria and the removal of 4.1.1 Parsing. For landing pages the most relevant are 2.4.11 Focus Not Obscured (Minimum), 2.5.8 Target Size (Minimum) and 3.3.8 Accessible Authentication (Minimum), all Level AA, plus 2.5.7, 3.3.7 and 3.2.6.

What are the WCAG 2.2 requirements for outsourced web development? None are outsourcing-specific. The Recommendation sets criteria and conformance conditions, such as complete processes and full pages, and the brief translates them into deliverables.

What does a WCAG 2.2 Level AA checklist look like? A list of Level A and AA criteria that apply to the page, each with an observable check. No short list is complete; W3C's Easy Checks are explicitly not definitive. The matrix above is a scoped extract.

What should accessibility acceptance criteria say? Each names the page or journey, criterion and level, test method and environment, evidence artifact, and issue and retest owner. Authored example: "With the promo bar visible on staging, every focusable lead-form element is at least partly visible when focused (2.4.11); evidence is a recording and tab-stop list."

What accessibility clauses belong in a web development contract? Authored suggestions, not legal drafting: target version, level and scope; named owners; evidence per row; a process for third-party components; remediation and retest responsibility; post-launch editing duties. W3C suggests including criteria

in requests for quotations and contracts, and GOV.UK suggests considering regular accessibility reviews in the contract (www.gov.uk). Have a qualified lawyer review actual contract language.

Who is liable if a site is challenged? This report does not answer that: it depends on jurisdiction and contract, and a technical record does not determine legal outcomes. Consult a qualified professional.

Conclusion

Website accessibility is owned by role and by task, and the commissioning brief is where that ownership can be made explicit. The business sets the target and scope, supplies the content that carries meaning, decides on third-party components, and owns every edit after launch. The designer supplies artifacts that show intent, the developer supplies implementation that can be tested, the tester reports what the built page does in a named environment, and the accepting owner decides, on the record.

Three practices carry most of the value. First, write a scoped, testable target: WCAG 2.2, Level AA, named pages and complete journeys, with third-party parts inside the boundary. Second, keep design-stage artifacts and build-stage evidence separate, because a mockup cannot prove keyboard behavior and a developer cannot decide what an image means. Third, report review coverage as incomplete, log unresolved issues with owners and decisions, and escalate to a specialist evaluation and to users with disabilities when stakes or scope exceed a short acceptance sample.

The limits are real. A criterion-by-owner matrix is a coordination tool, not a conformance claim, and the legal consequences of any gap depend on facts outside this report. Treat the matrix as the minimum honest record of what was asked, who did it, what was seen, and what remains open.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. <https://www.w3.org/WAI/planning/arm>
2. <https://www.w3.org/WAI/planning/org-policies>
3. <https://www.w3.org/TR/WCAG22>
4. <https://developer.chrome.com/docs/devtools/accessibility/reference>
5. <https://webaim.org/techniques/alttext>
6. <https://developer.chrome.com/blog/lighthouse-11-0>
7. <https://www.deque.com/automated-accessibility-coverage-report>
8. <https://webaim.org/projects/million>
9. <https://www.w3.org/WAI/policies>
10. <https://www.w3.org/WAI/test-evaluate/preliminary>
11. <https://www.superside.com/blog/web-design-audit>
12. <https://webaim.org/techniques/keyboard>
13. <https://www.w3.org/WAI/WCAG22/Understanding/focus-not-obsured-minimum.html>
14. <https://developer.mozilla.org/en-US/docs/Web/CSS/scroll-padding>

15. <https://web.dev/learn/accessibility/focus>
16. <https://www.w3.org/WAI/WCAG22/Understanding/target-size-minimum.html>
17. <https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Elements/button>
18. <https://web.dev/articles/accessible-tap-targets>
19. <https://support.google.com/accessibility/android/answer/7101858?hl=en>
20. <https://developer.apple.com/design/human-interface-guidelines/accessibility>
21. <https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Attributes/autocomplete>
22. <https://developer.mozilla.org/en-US/docs/Web/CSS/:focus-visible>
23. <https://www.w3.org/WAI/standards-guidelines/wcag/new-in-22>
24. <https://www.gov.uk/service-manual/helping-people-to-use-your-service/making-your-service-accessible-an-introduction>
25. <https://www.w3.org/WAI/planning/arm/decision-tree>
26. <https://www.w3.org/WAI/planning/arm/roles>
27. <https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA>
28. <https://www.section508.gov/buy>
29. <https://www.gov.uk/guidance/make-your-website-or-app-accessible-and-publish-an-accessibility-statement>
30. <https://www.section508.gov/buy/request-accessibility-information>
31. <https://developers.cloudflare.com/turnstile>
32. <https://calendly.com/legal/web-accessibility-statement>
33. <https://support.google.com/recaptcha>
34. <https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Elements/iframe>
35. <https://developers.google.com/recaptcha/docs/display>
36. <https://www.gov.uk/guidance/accessibility-requirements-for-public-sector-websites-and-apps>
37. <https://www.gov.uk/service-manual/helping-people-to-use-your-service/getting-an-accessibility-audit>
38. <https://web.dev/learn/forms/accessibility>
39. <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/label>
40. <https://www.gov.uk/service-manual/technology/testing-with-assistive-technologies>
41. <https://www.w3.org/WAI/tips/designing>
42. <https://help.figma.com/hc/en-us/articles/20774752502935-Add-measurements-and-annotate-designs>
43. <https://www.figma.com/color-contrast-checker>
44. <https://www.getstark.co/blog/platform-aware-annotations>
45. <https://web.dev/learn/accessibility/test-manual>
46. https://firefox-source-docs.mozilla.org/devtools-user/accessibility_inspector/index.html
47. <https://accessibilityinsights.io/docs/web/overview>
48. <https://www.gov.uk/service-manual/helping-people-to-use-your-service/testing-for-accessibility>
49. <https://developer.mozilla.org/en-US/docs/Web/CSS/scroll-padding-top>
50. <https://www.w3.org/TR/WCAG-EM>
51. <https://www.w3.org/WAI/test-evaluate/tools/selecting>

52. <https://accessibilityinsights.io/docs/web/getstarted/assessment>
53. <https://www.gov.uk/government/publications/doing-a-basic-accessibility-check-if-you-cant-do-a-detailed-one/doing-a-basic-accessibility-check-if-you-cant-do-a-detailed-one>
54. <https://www.section508.gov/buy/understand-claims>
55. <https://www.section508.gov/sell/acr>
56. <https://www.w3.org/WAI/test-evaluate/involving-users>
57. <https://www.deque.com/blog/automated-testing-study-identifies-57-percent-of-digital-accessibility-issues>
58. <https://accessibility.blog.gov.uk/2017/02/24/what-we-found-when-we-tested-tools-on-the-worlds-least-accessible-webpage>
59. <https://raw.githubusercontent.com/GoogleChrome/lighthouse/main/core/config/default-config.js>
60. <https://accessibilityinsights.io/docs/web/getstarted/fastpass>
61. <https://webaim.org/projects/screenreadersurvey11>
62. <https://www.gov.uk/guidance/make-things-accessible>
63. https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Accessibility/HTML
64. <https://wave.webaim.org>

THE PUBLISHER

About Tapflare

Tapflare gives companies an on-demand design and web team for a flat monthly fee.

How it works

Each subscriber has a dedicated project manager who learns the brand, assigns every request to a senior specialist in that kind of work, and reviews the result before it is delivered. Requests are submitted and discussed in the Tapflare portal, each plan includes a daily allotment of design or development time, and the client keeps full ownership of the export and source files.

What it covers

Graphic design, custom illustration, presentation design, logos and brand guides, video editing and motion graphics, website and mobile-app mockups, and website development, from design mockups to HTML, to drag-and-drop builders or custom code.

What it is not

Tapflare is not a marketing agency, a media-buying, SEO or copywriting service, a software product or a freelance marketplace. Where a reader needs one of those, the honest answer is to say so.

Visit

See [Tapflare](#) and its [pricing](#).

Articles published here explain general practice and cite their sources. Prices and plan contents, including Tapflare's own, change; check the provider's current pricing page before deciding.

Website: <https://tapflare.com>

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.