SAMPLE REPORT - a made-up demo cafe website with planted problems. In a real order every finding is checked by a person before delivery. Automated checks find only part of accessibility problems; this is not a legal certificate or a full audit.

Accessibility check - Demo cafe (made-up)

2 page(s) checked on 2026-10-29 03:50 UTC, on a desktop screen and a 320 px phone screen. **22 definite problems** and **3 things to check by hand**.

Automated testing finds only part of the accessibility problems on a site - usually about 30-50% of WCAG issues. This report is the automated part plus a list of things a person must check. It is not a full WCAG 2.2 audit and not a statement that the site is or is not "compliant".

Fix these first

1. **Keyboard focus is invisible** - 4 place(s). Keyboard users cannot see where they are on the page.
2. **Clickable item cannot be used with a keyboard** - 3 place(s). Keyboard, switch and many screen-reader users cannot use it at all.
3. **Button has no name** - 1 place(s). Screen-reader and voice users hear "button" and cannot say or know which one.
4. **Form field has no label** - 1 place(s). Screen-reader and voice-control users cannot tell what to type; placeholder text disappears.
5. **Link has no name** - 1 place(s). Screen-reader users hear "link" with nothing else (common with icon/logo links).

Definite problems (automated test failed)

Keyboard focus is invisible

Blocker | WCAG 2.4.7 | 4 place(s)

Who it hurts: Keyboard users cannot see where they are on the page.

How to fix: Do not remove the outline; if you restyle it, add a clear :focus-visible style (e.g. outline:3px solid #1a5fb4; outline-offset:2px).

▶ Where (first 20)

Clickable item cannot be used with a keyboard

Blocker | WCAG 2.1.1 | 3 place(s)

Who it hurts: Keyboard, switch and many screen-reader users cannot use it at all.

How to fix: Use a real <button> or <a href>. If you must use a div, add role, tabindex="0" and Enter/Space key handling.

▶ Where (first 20)

Button has no name

Blocker WCAG 4.1.2 1 place(s)

Who it hurts: Screen-reader and voice users hear "button" and cannot say or know which one.

How to fix: Put text inside the button, or for icon buttons add aria-label="Close" (or visually hidden text).

▶ Where (first 20)

Form field has no label

Blocker WCAG 1.3.1 1 place(s)

Who it hurts: Screen-reader and voice-control users cannot tell what to type; placeholder text disappears.

How to fix: Add a visible <label for="field-id">Name</label> (or aria-label if a visible label is impossible).

▶ Where (first 20)

Link has no name

Blocker WCAG 2.4.4 1 place(s)

Who it hurts: Screen-reader users hear "link" with nothing else (common with icon/logo links).

How to fix: Give the link text, or alt text on its image, or aria-label="Facebook page".

▶ Where (first 20)

Text is too faint against its background

Serious WCAG 1.4.3 1 place(s)

Who it hurts: People with low vision, older users, anyone on a phone in sunlight.

How to fix: Darken the text or lighten the background until the contrast is at least 4.5:1 (3:1 for large text 24 px, or 19 px bold).

▶ Where (first 20)

Two fields share one id, so a label points to the wrong field

Serious WCAG 1.3.1 1 place(s)

Who it hurts: Screen-reader users hear the wrong label; clicking the label focuses the wrong box.

How to fix: Give every field a unique id and make each label's for="" match its own field.

▶ Where (first 20)

Image has no text alternative

Serious WCAG 1.1.1 1 place(s)

Who it hurts: Blind screen-reader users hear the file name or nothing.

How to fix: Add alt="what the image shows or does". If it is only decoration, use alt="" so screen readers skip it.

▶ Where (first 20)

Page scrolls sideways on a narrow screen / 400% zoom

Serious WCAG 1.4.10 1 place(s)

Who it hurts: People who zoom in, and small phones, must scroll left-right on every line.

How to fix: Remove fixed widths (use max-width:100%), let tables/code scroll inside their own box (overflow-x:auto), wrap long words.

▶ Where (first 20)

Pinch-zoom is blocked on phones

Serious WCAG 1.4.4 1 place(s)

Who it hurts: People with low vision cannot enlarge text.

How to fix: Remove user-scalable=no and maximum-scale from the viewport meta tag.

▶ Where (first 20)

Tap target is too small and too close to others

Moderate WCAG 2.5.8 3 place(s)

Who it hurts: People with shaky hands or big fingers tap the wrong thing.

How to fix: Make the clickable area at least 24 x 24 CSS pixels (44 x 44 is better), or add spacing so 24 px circles do not overlap.

▶ Where (first 20)

Page language is not set

Moderate WCAG 3.1.1 1 place(s)

Who it hurts: Screen readers may read the page in the wrong accent or language.

How to fix: Add lang to the html tag, e.g. <html lang="en">.

▶ Where (first 20)

Page has no title

Moderate WCAG 2.4.2 1 place(s)

Who it hurts: Screen-reader users and anyone with many tabs open cannot tell pages apart.

How to fix: Add a <title> that says what the page is, e.g. "Book a table - Rosie's Cafe".

▶ Where (first 20)

Heading is empty

Minor WCAG 1.3.1 1 place(s)

Who it hurts: Screen-reader users who jump by headings land on nothing.

How to fix: Put text in the heading or remove the empty heading tag.

▶ Where (first 20)

List is built incorrectly

Minor WCAG 1.3.1 1 place(s)

Who it hurts: Screen readers miscount or do not announce the list.

How to fix: Only put <li> directly inside /<ol>, and only <dt>/<dd> inside <dl>.

▶ Where (first 20)

Please check by hand

Page scrolls sideways on a narrow screen / 400% zoom

Serious | WCAG 1.4.10 | 1 place(s)

Who it hurts: People who zoom in, and small phones, must scroll left-right on every line.

How to fix: Remove fixed widths (use max-width:100%), let tables/code scroll inside their own box (overflow-x:auto), wrap long words.

▶ Where (first 20)

Contrast could not be measured automatically (please check)

Minor | WCAG 1.4.3 | 2 place(s)

Who it hurts: Text over images or gradients may be unreadable for low-vision users.

How to fix: Check the text against the darkest/lightest part of the background; add a solid overlay if needed.

▶ Where (first 20)

What an automated check cannot see

- Whether alt text, link text, labels and headings are meaningful (we can only see if they exist).
- Whether captions and audio descriptions are correct.
- Whether the keyboard order makes sense and every task can be finished with a keyboard only.
- How it sounds in a real screen reader (NVDA, JAWS, VoiceOver, TalkBack).
- Errors, time limits, moving content, PDFs and pages behind a login we were not given.
- Colour used as the only way to show meaning (except links in text).

Our human review add-on covers these with keyboard and screen-reader testing of your key pages.

Context, not legal advice: in the US the ADA is used for website accessibility claims; in the EU the European Accessibility Act applies to many consumer services from 28 June 2025; in Australia the Disability Discrimination Act 1992 covers websites and WCAG is the usual reference. Ask a lawyer about your own duties.

Checked with our own checking tools, then reviewed by a person.