

SAMPLE REPORT

Accessibility audit report

WCAG 2.2 Level AA · ADA · Section 508

SITE TESTED

Example Goods (fictional online store)

PREPARED BY

Rohit, DHS Trusted Tester (Section 508)

PAGES IN SCOPE

Home, collection, product, cart and checkout

DELIVERED AS

Excel findings spreadsheet and this report

About this sample. This is an illustrative example for a fictional store. It follows the structure of my real reports, but the store, the pages and every finding are invented. It is not based on a real client, and client names and details are never shared.

About this report

Standard	WCAG 2.2 Level AA, which covers the requirements of the ADA and Section 508
Methods	Keyboard-only testing; NVDA and JAWS on Windows; VoiceOver on macOS and iOS; TalkBack on Android; zoom to 400%; contrast checks. Automated tools (axe, WAVE) are a starting point, and every finding is confirmed by hand.
Scope	Home, collection, product, cart and checkout pages of the fictional store Example Goods
Findings	13 in total: 6 High, 5 Medium and 2 Low

6

High

5

Medium

2

Low

HOW EACH FINDING IS DOCUMENTED

Screenshot	An annotated screenshot of the issue, based on the client's website.
Issue description	What the issue is and why it matters to people using assistive technologies.
Steps to reproduce	Exactly how to see the problem, so your team can verify it.
Actual behavior	What happens on your site today.
Expected behavior / remediation	What should happen, with recommended steps and a code example.
Severity	High, Medium or Low (see below).
WCAG conformance level	A or AA, or Best practice where the issue is good practice rather than a failure.
WCAG success criterion	The WCAG 2.2 criterion the issue relates to.

SEVERITY RATINGS

- High:** prevents or seriously hinders someone from completing a key task, such as buying.
- Medium:** causes difficulty or confusion, but a workaround exists.
- Low:** a small inconvenience or best-practice improvement.

CONFORMANCE LEVELS

- Level A:** the minimum requirements. Failing them blocks access for some people.
- Level AA:** the standard referenced by the ADA and Section 508.
- Best practice:** not a WCAG failure, but recommended.

Findings summary

All 13 findings, ordered by severity. Each one is documented in full on the pages that follow.

ID	Issue	Severity	Page	WCAG level	WCAG criterion
A-01	Size options cannot be selected with the keyboard	High	Product page	A	2.1.1
A-02	Cart confirmation message is not announced	High	Product page, cart	AA	4.1.3
A-03	Focus does not move into the mini-cart drawer	High	Cart drawer	A	2.4.3
A-04	Icon-only buttons have no accessible name	High	All pages	A	4.1.2
A-05	Checkout errors are not linked to their form fields	High	Checkout	A	3.3.1, 1.3.1
A-06	Focus outline removed from links and buttons	High	All pages	AA	2.4.7
A-07	Main navigation links are not marked up as a list	Medium	All pages	A	1.3.1
A-08	Current page is not identified in the navigation	Medium	All pages	A	4.1.2
A-09	Sale price text has insufficient color contrast	Medium	Collection and product pages	AA	1.4.3
A-10	Active slide is not identified in the carousel indicators	Medium	Home page	A	4.1.2
A-11	Selected filter is not identified programmatically	Medium	Collection page	A	4.1.2
A-12	Redundant alternative text on product images	Low	Collection page	Best practice	1.1.1 (related)
A-13	Footer headings skip a heading level	Low	All pages (footer)	A	1.3.1

SEVERITY

High

PAGE

Product page

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

2.1.1

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Interactive controls must be operable with a keyboard alone. Custom controls built from non-interactive elements are invisible to the keyboard unless they are given the right role and behavior. When a required choice, such as a product size, cannot be made without a mouse, the whole purchase is blocked.

STEPS TO REPRODUCE

1. Open any product page in a desktop browser.
2. Press Tab repeatedly, starting from the product title.
3. Observe that focus moves from the image gallery straight to the Add to cart button. The size options (S, M, L and XL) are never reached.
4. Press Enter or Space while the size area is visible: no size is selected.

ACTUAL BEHAVIOR

The size options (S, M, L and XL) are styled `div` elements with mouse click handlers only. They are not in the tab order, and pressing Enter or Space has no effect. As a result, people who navigate with a keyboard, a switch device or voice control cannot choose a size and cannot complete a purchase.

EXPECTED BEHAVIOR / REMEDIATION

All options of a product should be selectable with the keyboard and announced with their name, role and state.

1. Use native radio buttons grouped in a `fieldset` with a `legend`, instead of styled `div` elements.
2. Style the radio buttons with CSS so the visual design stays the same.

```
<fieldset>
  <legend>Size</legend>
  <input type="radio" id="size-s" name="size" value="S">
  <label for="size-s">S</label>
  <input type="radio" id="size-m" name="size" value="M">
  <label for="size-m">M</label>
</fieldset>
```

COMMENTS

Affects people using keyboards, switch devices, voice control and screen readers. To confirm the fix, tab to the group and choose with the arrow keys. A screen reader should announce "Size, M, radio button, 2 of 4, selected".

SEVERITY

High

PAGE

Product page, cart

WCAG CONFORMANCE

LEVEL

AA

WCAG SUCCESS CRITERION

4.1.3

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

When a page changes after an action, such as an item being added to a cart, people who cannot see the change need it announced. Status messages should be exposed programmatically so assistive technologies can announce them without moving focus.

STEPS TO REPRODUCE

1. Turn on a screen reader (NVDA on Windows or VoiceOver on macOS).
2. On a product page, choose a size and activate Add to cart.
3. Observe that the cart count and the on-screen notice change, but the screen reader announces nothing.

ACTUAL BEHAVIOR

After "Add to cart" is activated, the cart count changes and a notice reading "Linen overshirt added to cart" appears on screen, but the notice is inserted into the page without a status role. As a result, screen reader users receive no confirmation and cannot tell whether the item was added.

EXPECTED BEHAVIOR / REMEDIATION

Status messages should be announced by assistive technologies without moving keyboard focus.

1. Add an empty container with `role="status"` to the page when it loads.
2. Write the confirmation text into that container after the item is added. The container must already exist, or many screen readers will not announce the text.

```
<div id="cart-status" role="status"></div>

<script>
  document.getElementById("cart-status").textContent =
    "Linen overshirt added to cart. Your cart has 2 items.";
</script>
```

COMMENTS

Affects people using screen readers. To confirm the fix, turn on NVDA or VoiceOver, add an item and listen for the message.

SEVERITY

High

PAGE

Cart drawer

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

2.4.3

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

When a drawer or dialog opens, keyboard focus must move into it and stay there until it is closed, and users must be able to close it with the keyboard. Otherwise keyboard users end up interacting with content hidden behind the overlay.

STEPS TO REPRODUCE

1. Tab to the cart button and press Enter to open the mini-cart drawer.
2. Press Tab repeatedly.
3. Observe that focus moves through links on the page behind the drawer and never enters the drawer.
4. Press Escape: the drawer stays open.

ACTUAL BEHAVIOR

When the mini-cart drawer opens, pressing Tab moves focus through the links on the covered page behind it instead of the contents of the drawer, and pressing Escape does not close it. As a result, keyboard users cannot reach the "Checkout" button or dismiss the drawer without a mouse.

EXPECTED BEHAVIOR / REMEDIATION

When a drawer opens, focus should move into it and stay there until it closes. Escape should close it and return focus to the button that opened it.

1. Move focus to the drawer heading or the close button when the drawer opens.
2. Keep Tab inside the drawer while it is open. The native `dialog` element with `showModal()` does this automatically.
3. Close the drawer on Escape and return focus to the cart button.

```
const drawer = document.querySelector("dialog#mini-cart");
drawer.showModal(); // focus moves in, Escape closes it
```

COMMENTS

Affects keyboard and screen reader users. To confirm the fix, open the drawer with the keyboard, tab through it, close it with Escape and check that focus returns to the cart button.

SEVERITY

High

PAGE

All pages

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

4.1.2

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Every button needs an accessible name so people using screen readers and voice control know what it does. Icon-only buttons need a text alternative, because the picture alone is not available to assistive technologies.

STEPS TO REPRODUCE

1. Turn on a screen reader.
2. Tab to the search, wishlist, cart and close icons in the header and on product cards.
3. Observe that each one is announced only as "button".

ACTUAL BEHAVIOR

The search, cart, wishlist and close buttons contain only an SVG icon, with no text and no `aria-label`. As a result, screen reader users hear only "button" and cannot tell what each one does.

EXPECTED BEHAVIOR / REMEDIATION

Every button should have an accessible name that describes its purpose.

1. Add an `aria-label` (or visually hidden text) to each icon button.
2. Hide the decorative SVG from assistive technologies with `aria-hidden="true"`.

```
<button type="button" aria-label="Cart, 2 items">
  <svg aria-hidden="true" focusable="false">...</svg>
</button>
```

COMMENTS

Affects people using screen readers and voice control. To confirm the fix, tab to each icon button and listen to its name.

SEVERITY

High

PAGE

Checkout

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

3.3.1, 1.3.1

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Form errors must be identified in text and connected to the field they belong to, so that people using assistive technologies know that an error occurred, where it is and how to fix it.

STEPS TO REPRODUCE

1. Add an item to the cart and go to checkout.
2. Leave the Email field empty and activate Continue to shipping.
3. Observe the red message under the field. A screen reader announces nothing, and focus stays on the button.

ACTUAL BEHAVIOR

When the email field is left empty, a red message appears under it, but the message is not connected to the field and focus stays on the button. As a result, screen reader users are not told that an error occurred or which field it relates to.

EXPECTED BEHAVIOR / REMEDIATION

Errors should be identified in text, connected to the field they belong to, and announced to assistive technologies.

1. Add `aria-invalid="true"` to each field in error.
2. Connect the error message to its field with `aria-describedby`.
3. Move focus to an error summary at the top of the form that links to each problem.

```
<label for="email">Email</label>
<input id="email" type="email"
  aria-invalid="true"
  aria-describedby="email-error">
<p id="email-error">
  Enter your email address, for example name@example.com.
</p>
```

COMMENTS

Affects people using screen readers and people with cognitive or learning disabilities. To confirm the fix, submit the empty form. Focus should move to the summary, and each field should be announced as invalid with its message.

SEVERITY

High

PAGE

All pages

WCAG CONFORMANCE
LEVEL

AA

WCAG SUCCESS CRITERION

2.4.7

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Keyboard users need to see which element has focus. Removing the browser's default focus outline without replacing it leaves them unable to tell where they are on the page.

STEPS TO REPRODUCE

1. Open the home page and press Tab repeatedly.
2. Watch the links and buttons as focus moves through them.
3. Observe that no outline or other visible change marks the focused element.

ACTUAL BEHAVIOR

The stylesheet sets `outline: none` on links and buttons and adds no other visible focus style. As a result, keyboard users cannot see where they are on the page.

EXPECTED BEHAVIOR / REMEDIATION

Every interactive element should have a clearly visible focus indicator.

1. Remove `outline: none` from the global stylesheet.
2. Add a `:focus-visible` style with at least 3:1 contrast against the background.

```
:focus-visible {  
  outline: 3px solid #1A1A1A;  
  outline-offset: 2px;  
}
```

COMMENTS

Affects everyone who navigates with a keyboard. To confirm the fix, tab through each page and check that the focused item is always visible.

SEVERITY

Medium

PAGE

All pages

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

1.3.1

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Groups of related links should be marked up as lists. List markup lets assistive technologies announce how many items there are, and lets users skip past the whole group.

STEPS TO REPRODUCE

1. Inspect the main navigation with the browser's developer tools.
2. Observe that the links are a sequence of anchor elements with no `ul` or `li` elements.
3. With a screen reader running, move to the menu and note that no list is announced.

ACTUAL BEHAVIOR

The main navigation links ("Home", "Shop", "Journal" and "About") are plain `a` elements separated by spaces, with no list markup. As a result, screen reader users are not told that the links are related or how many there are.

EXPECTED BEHAVIOR / REMEDIATION

Related links should be structured with list markup so assistive technologies can identify the list and its items.

1. Use `ul` and `li` elements to mark up the navigation links.
2. Keep the visual presentation with CSS.

```
<nav aria-label="Main">
  <ul>
    <li><a href="/">Home</a></li>
    <li><a href="/shop/">Shop</a></li>
    <li><a href="/journal/">Journal</a></li>
    <li><a href="/about/">About</a></li>
  </ul>
</nav>
```

COMMENTS

Affects people using screen readers. To confirm the fix, navigate to the menu with a screen reader and listen for "list, 4 items".

SEVERITY

Medium

PAGE

All pages

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

4.1.2

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Navigation should tell people using assistive technologies which page they are on. A visual highlight alone is not available to people who cannot see it.

STEPS TO REPRODUCE

1. Open the Shop page.
2. Inspect the Shop link in the main navigation.
3. Observe that it is styled in bold but has no aria-current attribute, and that a screen reader does not announce it as the current page.

ACTUAL BEHAVIOR

The link for the current page is shown in bold, but no attribute identifies it as current. As a result, screen reader users are not told which page they are on.

EXPECTED BEHAVIOR / REMEDIATION

The current state of the active link should be defined programmatically and announced.

1. Add `aria-current="page"` to the link of the current page.
2. Make sure the attribute moves to the right link on every page.

```
<a href="/shop/" aria-current="page">Shop</a>
```

COMMENTS

Affects people using screen readers. To confirm the fix, tab to the link and listen for "current page".

SEVERITY

Medium

PAGE

Collection and product pages

WCAG CONFORMANCE
LEVEL

AA

WCAG SUCCESS CRITERION

1.4.3

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Text needs enough contrast against its background to be readable by people with low vision and in poor viewing conditions. WCAG requires a ratio of at least 4.5:1 for normal-sized text.

STEPS TO REPRODUCE

1. Open a collection page that shows sale items.
2. Pick the text color and the background color of a sale price with a color picker.
3. Enter both values in a contrast checker and observe a ratio of 2.99:1.

ACTUAL BEHAVIOR

Sale prices are rendered with foreground color #E57373 against a white #FFFFFF background, giving a contrast ratio of 2.99:1. As a result, users with low vision cannot adequately read the price.

EXPECTED BEHAVIOR / REMEDIATION

Normal text should have a contrast ratio of at least 4.5:1 against its background.

1. Change the foreground color from #E57373 to #C0392B (a ratio of 5.44:1) or darker.
2. Apply the new value in the stylesheet rule for the sale price.

```
.price--sale {  
  color: #C0392B; /* 5.44:1 on white */  
}
```

COMMENTS

Affects people with low vision and people using a screen in bright light. To confirm the fix, measure the new color against its background with a contrast checker.

SEVERITY

Medium

PAGE

Home page

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

4.1.2

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Carousel controls must have accessible names, and the currently active slide must be identifiable by people who cannot see the visual indicator.

STEPS TO REPRODUCE

1. Open the home page and tab to the carousel indicators.
2. With a screen reader running, move through the five dots.
3. Observe that each is announced as "button", with no name and no indication of which slide is active.

ACTUAL BEHAVIOR

The five round indicators under the homepage carousel are `button` elements with no text. The active slide is shown only by a filled dot. As a result, screen reader users hear "button" five times and cannot tell which slide is active.

EXPECTED BEHAVIOR / REMEDIATION

Each indicator should have an accessible name, and the active one should be identified programmatically.

1. Give each button a name, such as `aria-label="Go to slide 2 of 5"`.
2. Add `aria-current="true"` to the active indicator, and update it when the slide changes.

```
<button type="button" aria-label="Go to slide 1 of 5" aria-current="true"></button>  
<button type="button" aria-label="Go to slide 2 of 5"></button>
```

COMMENTS

Affects people using screen readers. Note: `aria-selected` is only valid on elements with the `tab`, `option`, `row` or `gridcell` role, so it should not be added to plain buttons. If the indicators follow the tabs pattern, use `role="tablist"`, `role="tab"` and `aria-selected` instead.

SEVERITY

Medium

PAGE

Collection page

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

4.1.2

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Toggle buttons, such as filters, must expose their on or off state to assistive technologies. A change of color alone is not enough.

STEPS TO REPRODUCE

1. Open the collection page and apply the Linen filter.
2. Tab to the Linen button with a screen reader running.
3. Observe that it is announced only as "button", with no pressed state.

ACTUAL BEHAVIOR

On the collection page, the selected "Linen" filter is highlighted with a background color only, and the filter button has no state. As a result, screen reader users are not aware that "Linen" is the filter currently applied.

EXPECTED BEHAVIOR / REMEDIATION

The selected filter should be announced as pressed.

1. Add `aria-pressed="true"` to the selected filter button and `aria-pressed="false"` to the others.
2. Update the values whenever the user selects a different filter.

```
<button type="button" aria-pressed="true">Linen</button>  
<button type="button" aria-pressed="false">Cotton</button>
```

COMMENTS

Affects people using screen readers. To confirm the fix, tab to a filter and listen for "pressed" or "not pressed".

SEVERITY

Low

PAGE

Collection page

WCAG CONFORMANCE
LEVEL

Best practice

WCAG SUCCESS CRITERION

1.1.1 (related)

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Images that only repeat adjacent text add noise for screen reader users. Decorative or redundant images should be hidden from assistive technologies by giving them an empty alt attribute.

STEPS TO REPRODUCE

1. Open a collection page and listen to a product card with a screen reader.
2. Observe that the product name is read twice: once from the image alt text and once from the link text.

ACTUAL BEHAVIOR

Each product image has alt text that repeats the product name shown in the adjacent visible link (for example alt="Linen overshirt" beside the link "Linen overshirt"). As a result, screen reader users hear the same name twice.

EXPECTED BEHAVIOR / REMEDIATION

Images that only repeat adjacent text should be ignored by assistive technologies.

1. Set the alt attribute to an empty string on images that repeat adjacent text.
2. Alternatively, place the image and the text inside the same link.

```
<a href="/shop/linen-overshirt/">  
    
  Linen overshirt  
</a>
```

COMMENTS

A small annoyance rather than a barrier. To confirm the fix, listen to a product card with a screen reader and check that the name is read once.

SEVERITY

Low

PAGE

All pages (footer)

WCAG CONFORMANCE
LEVEL

A

WCAG SUCCESS CRITERION

1.3.1

SCREENSHOT

An annotated screenshot of this issue is added here in a real report, based on the client's website.

ISSUE DESCRIPTION

Headings should follow a logical outline without skipping levels, so that people who navigate by headings can understand the structure of the page.

STEPS TO REPRODUCE

1. Open any page and list its headings with a screen reader or a browser extension.
2. Observe that the footer headings are level 4 and follow a level 2, with no level 3 between them.

ACTUAL BEHAVIOR

The footer headings "Shop" and "Help" are `h4` elements that follow an `h2` heading, skipping level 3. As a result, screen reader users who navigate by headings encounter a confusing jump in the page outline.

EXPECTED BEHAVIOR / REMEDIATION

Heading levels should not skip, so the structure reflects the page outline.

1. Change the `h4` elements to `h3`.
2. Keep the appearance by reusing the existing CSS classes on the new elements.

```
<h2>Customer care</h2>
<h3>Shop</h3>
<h3>Help</h3>
```

COMMENTS

Affects people who navigate by headings. To confirm the fix, list the page headings with a screen reader and check that no level is skipped.

Recommended fix order

Most findings are fixed once in the theme templates and then apply across the whole site. Fixing them in this order removes the biggest barriers first.

Fix first

High

These block key tasks such as choosing a size, adding to cart and checking out.

- **A-01** Size options cannot be selected with the keyboard
- **A-02** Cart confirmation message is not announced
- **A-03** Focus does not move into the mini-cart drawer
- **A-04** Icon-only buttons have no accessible name
- **A-05** Checkout errors are not linked to their form fields
- **A-06** Focus outline removed from links and buttons

Fix next

Medium

These cause difficulty or confusion, and most are quick template-level changes.

- **A-07** Main navigation links are not marked up as a list
- **A-08** Current page is not identified in the navigation
- **A-09** Sale price text has insufficient color contrast
- **A-10** Active slide is not identified in the carousel indicators
- **A-11** Selected filter is not identified programmatically

Fix when convenient

Low

Small inconveniences and best-practice improvements.

- **A-12** Redundant alternative text on product images
- **A-13** Footer headings skip a heading level

WHAT A FULL AUDIT INCLUDES

A findings spreadsheet your team can filter by page, severity or WCAG criterion. For every issue: the page, a screenshot, the issue description, steps to reproduce, the actual and expected behavior with a code-level fix, the severity, and the WCAG level and criterion. A walkthrough call with your developers, and a free retest after your fixes.

HOW THE RETEST WORKS

1. Your developers fix the issues, starting with the High findings.
2. Within 60 days, I recheck every issue from the original audit with the same keyboard and screen reader tests.
3. You receive an updated findings spreadsheet showing which issues are fixed and which are still open.

Want a report like this for your site?

Request a free one-page accessibility check at adaauditpro.org/free-accessibility-check/

Or see audit pricing at adaauditpro.org/pricing/

ADA Audit Pro · hello@adaauditpro.org