http://127.0.0.1:8081/detect_generic.html

97 Accessibility

77 Best Practices

▲ 0–49 50–89 90–100

97

Accessibility

These checks highlight opportunities to improve the accessibility of your web app. Automatic detection can only detect a subset of issues and does not guarantee the accessibility of your web app, so manual testing is also encouraged.

NAVIGATION

▲ Heading elements are not in a sequentially-descending order

Properly ordered headings that do not skip levels convey the semantic structure of the page, making it easier to navigate and understand when using assistive technologies. Learn more about heading order.

Failing Elements

These are opportunities to improve keyboard navigation in your application.

AUDIO AND VIDEO

○ `<video>` elements contain a `<track>` element with `[kind="captions"]`

When a video provides a caption it is easier for deaf and hearing impaired users to access its information. Learn more about video captions. Unscored

Failing Elements

These are opportunities to provide alternative content for audio and video. This may improve the experience for users with hearing or vision impairments.

ADDITIONAL ITEMS TO MANUALLY CHECK (10) Hide

- Interactive controls are keyboard focusable

  Custom interactive controls are keyboard focusable and display a focus indicator. Learn how to make custom controls focusable. Unscored

- Interactive elements indicate their purpose and state

  Interactive elements, such as links and buttons, should indicate their state and be distinguishable from non-interactive elements. Learn how to decorate interactive elements with affordance hints. Unscored

- The page has a logical tab order

  Tabbing through the page follows the visual layout. Users cannot focus elements that are offscreen. Learn more about logical tab ordering. Unscored

- Visual order on the page follows DOM order

  DOM order matches the visual order, improving navigation for assistive technology. Learn more about DOM and visual ordering. Unscored

- User focus is not accidentally trapped in a region

  A user can tab into and out of any control or region without accidentally trapping their focus. Learn how to avoid focus traps. Unscored

- The user's focus is directed to new content added to the page

  If new content, such as a dialog, is added to the page, the user's focus is directed to it. Learn how to direct focus to new content. Unscored

- HTML5 landmark elements are used to improve navigation

  Landmark elements (<main>, <nav>, etc.) are used to improve the keyboard navigation of the page for assistive technology. Learn more about landmark elements. Unscored

- Offscreen content is hidden from assistive technology

  Offscreen content is hidden with display: none or aria-hidden=true. Learn how to properly hide offscreen content. Unscored

- Custom controls have associated labels

  Custom interactive controls have associated labels, provided by aria-label or aria-labelledby. Learn more about custom controls and labels. Unscored

- Custom controls have ARIA roles

  Custom interactive controls have appropriate ARIA roles. Learn how to add roles to custom controls. Unscored

These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review.

PASSED AUDITS (13) Hide

- `[aria-hidden="true"]` is not present on the document `<body>`

  Assistive technologies, like screen readers, work inconsistently when `aria-hidden="true"` is set on the document <body>. Learn how `aria-hidden` affects the document body.

- `[role]`s have all required `[aria-*]` attributes

  Some ARIA roles have required attributes that describe the state of the element to screen readers. Learn more about roles and required attributes.

- `[role]` values are valid

  ARIA roles must have valid values in order to perform their intended accessibility functions. Learn more about valid ARIA roles.

- Form elements have associated labels

  Labels ensure that form controls are announced properly by assistive technologies, like screen readers. Learn more about form element labels.

- `[user-scalable="no"]` is not used in the `<meta name="viewport">` element and the `[maximum-scale]` attribute is not less than 5.

  Disabling zooming is problematic for users with low vision who rely on screen magnification to properly see the contents of a web page. Learn more about the viewport meta tag.

Background and foreground colors have a sufficient contrast ratio

Low-contrast text is difficult or impossible for many users to read. Learn how to provide sufficient color contrast.

Document has a `<title>` element

The title gives screen reader users an overview of the page, and search engine users rely on it heavily to determine if a page is relevant to their search. Learn more about document titles.

`<html>` element has a `[lang]` attribute

If a page doesn't specify a `lang` attribute, a screen reader assumes that the page is in the default language that the user chose when setting up the screen reader. If the page isn't actually in the default language, then the screen reader might not announce the page's text correctly. Learn more about the `lang` attribute.

`<html>` element has a valid value for its `[lang]` attribute

Specifying a valid BCP 47 language helps screen readers announce text properly. Learn how to use the `lang` attribute.

Links have a discernible name

Link text (and alternate text for images, when used as links) that is discernible, unique, and focusable improves the navigation experience for screen reader users. Learn how to make links accessible.

Touch targets have sufficient size and spacing.

Touch targets with sufficient size and spacing help users who may have difficulty targeting small controls to activate the targets. Learn more about touch targets.

Document has a main landmark.

One main landmark helps screen reader users navigate a web page. Learn more about landmarks.

Deprecated ARIA roles were not used

Deprecated ARIA roles may not be processed correctly by assistive technology. Learn more about deprecated ARIA roles.

NOT APPLICABLE (45) Hide

`[accesskey]` values are unique

Access keys let users quickly focus a part of the page. For proper navigation, each access key must be unique. Learn more about access keys. Unscored

- [aria-*] attributes match their roles

Each ARIA role supports a specific subset of aria-* attributes. Mismatching these invalidates the aria-* attributes. Learn how to match ARIA attributes to their roles. Unscored

- button, link, and menuitem elements have accessible names

When an element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to make command elements more accessible. Unscored

- ARIA attributes are used as specified for the element's role

Some ARIA attributes are only allowed on an element under certain conditions. Learn more about conditional ARIA attributes. Unscored

- Elements with role="dialog" or role="alertdialog" have accessible names.

ARIA dialog elements without accessible names may prevent screen readers users from discerning the purpose of these elements. Learn how to make ARIA dialog elements more accessible. Unscored

- [aria-hidden="true"] elements do not contain focusable descendents

Focusable descendents within an [aria-hidden="true"] element prevent those interactive elements from being available to users of assistive technologies like screen readers. Learn how aria-hidden affects focusable elements. Unscored

- ARIA input fields have accessible names

When an input field doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about input field labels. Unscored

- ARIA meter elements have accessible names

When a meter element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to name meter elements. Unscored

- ARIA progressbar elements have accessible names

When a progressbar element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to label progressbar elements. Unscored

- Elements use only permitted ARIA attributes

Using ARIA attributes in roles where they are prohibited can mean that important information is not communicated to users of assistive technologies. Learn more about prohibited ARIA roles. Unscored

- Elements with an ARIA `[role]` that require children to contain a specific `[role]` have all required children.

Some ARIA parent roles must contain specific child roles to perform their intended accessibility functions. Learn more about roles and required children elements. Unscored

- `[role]`s are contained by their required parent element

Some ARIA child roles must be contained by specific parent roles to properly perform their intended accessibility functions. Learn more about ARIA roles and required parent element. Unscored

- Elements with the `role=text` attribute do not have focusable descendents.

Adding `role=text` around a text node split by markup enables VoiceOver to treat it as one phrase, but the element's focusable descendents will not be announced. Learn more about the `role=text` attribute. Unscored

- ARIA toggle fields have accessible names

When a toggle field doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about toggle fields. Unscored

- ARIA `tooltip` elements have accessible names

When a tooltip element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to name `tooltip` elements. Unscored

- ARIA `treeitem` elements have accessible names

When a `treeitem` element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about labeling `treeitem` elements. Unscored

- `[aria-*]` attributes have valid values

Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid values. Learn more about valid values for ARIA attributes. Unscored

- `[aria-*]` attributes are valid and not misspelled

Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid names. Learn more about valid ARIA attributes. Unscored

- Buttons have an accessible name

When a button doesn't have an accessible name, screen readers announce it as "button", making it unusable for users who rely on screen readers. Learn how to make buttons more accessible. Unscored

- The page contains a heading, skip link, or landmark region

Adding ways to bypass repetitive content lets keyboard users navigate the page more efficiently. Learn more about bypass blocks. Unscored

- `<dl>`'s contain only properly-ordered `<dt>` and `<dd>` groups, `<script>`, `<template>` or `<div>` elements.

When definition lists are not properly marked up, screen readers may produce confusing or inaccurate output. Learn how to structure definition lists correctly. Unscored

- Definition list items are wrapped in `<dl>` elements

Definition list items (<dt> and <dd>) must be wrapped in a parent <dl> element to ensure that screen readers can properly announce them. Learn how to structure definition lists correctly. Unscored

- ARIA IDs are unique

The value of an ARIA ID must be unique to prevent other instances from being overlooked by assistive technologies. Learn how to fix duplicate ARIA IDs. Unscored

- No form fields have multiple labels

Form fields with multiple labels can be confusingly announced by assistive technologies like screen readers which use either the first, the last, or all of the labels. Learn how to use form labels. Unscored

- `<frame>` or `<iframe>` elements have a title

Screen reader users rely on frame titles to describe the contents of frames. Learn more about frame titles. Unscored

- `<html>` element has an `[xml:lang]` attribute with the same base language as the `[lang]` attribute.

If the webpage does not specify a consistent language, then the screen reader might not announce the page's text correctly. Learn more about the `lang` attribute. Unscored

- Image elements have `[alt]` attributes

Informative elements should aim for short, descriptive alternate text. Decorative elements can be ignored with an empty alt attribute. Learn more about the `alt` attribute. Unscored

- Input buttons have discernible text.

Adding discernable and accessible text to input buttons may help screen reader users understand the purpose of the input button. Learn more about input buttons. Unscored

- `<input type="image">` elements have `[alt]` text

When an image is being used as an `<input>` button, providing alternative text can help screen reader users understand the purpose of the button. Learn about input image alt text. Unscored

- Links are distinguishable without relying on color.

Low-contrast text is difficult or impossible for many users to read. Link text that is discernible improves the experience for users with low vision. Learn how to make links distinguishable. Unscored

- Lists contain only `<li>` elements and script supporting elements (`<script>` and `<template>`).

Screen readers have a specific way of announcing lists. Ensuring proper list structure aids screen reader output. Learn more about proper list structure. Unscored

- List items (`<li>`) are contained within `<ul>`, `<ol>` or `<menu>` parent elements

Screen readers require list items (`<li>`) to be contained within a parent `<ul>`, `<ol>` or `<menu>` to be announced properly. Learn more about proper list structure. Unscored

- The document does not use `<meta http-equiv="refresh">`

Users do not expect a page to refresh automatically, and doing so will move focus back to the top of the page. This may create a frustrating or confusing experience. Learn more about the refresh meta tag. Unscored

- `<object>` elements have alternate text

Screen readers cannot translate non-text content. Adding alternate text to `<object>` elements helps screen readers convey meaning to users. Learn more about alt text for `object` elements. Unscored

- Select elements have associated label elements.

Form elements without effective labels can create frustrating experiences for screen reader users. Learn more about the `select` element. Unscored

- Skip links are focusable.

Including a skip link can help users skip to the main content to save time. Learn more about skip links. Unscored

- No element has a `[tabindex]` value greater than 0

A value greater than 0 implies an explicit navigation ordering. Although technically valid, this often creates frustrating experiences for users who rely on assistive technologies. Learn more about the `tabindex` attribute. Unscored

- Cells in a `<table>` element that use the `[headers]` attribute refer to table cells within the same table.

  Screen readers have features to make navigating tables easier. Ensuring `<td>` cells using the `[headers]` attribute only refer to other cells in the same table may improve the experience for screen reader users. Learn more about the `headers` attribute. Unscored

- `<th>` elements and elements with `[role="columnheader"/"rowheader"]` have data cells they describe.

  Screen readers have features to make navigating tables easier. Ensuring table headers always refer to some set of cells may improve the experience for screen reader users. Learn more about table headers. Unscored

- `[lang]` attributes have a valid value

  Specifying a valid BCP 47 language on elements helps ensure that text is pronounced correctly by a screen reader. Learn how to use the `lang` attribute. Unscored

- Tables have different content in the summary attribute and `<caption>`.

  The summary attribute should describe the table structure, while `<caption>` should have the onscreen title. Accurate table mark-up helps users of screen readers. Learn more about summary and caption. Unscored

- All heading elements contain content.

  A heading with no content or inaccessible text prevent screen reader users from accessing information on the page's structure. Learn more about headings. Unscored

- Uses ARIA roles only on compatible elements

  Many HTML elements can only be assigned certain ARIA roles. Using ARIA roles where they are not allowed can interfere with the accessibility of the web page. Learn more about ARIA roles. Unscored

- Image elements do not have `[alt]` attributes that are redundant text.

  Informative elements should aim for short, descriptive alternative text. Alternative text that is exactly the same as the text adjacent to the link or image is potentially confusing for screen reader users, because the text will be read twice. Learn more about the `alt` attribute. Unscored

- Identical links have the same purpose.

  Links with the same destination should have the same description, to help users understand the link's purpose and decide whether to follow it. Learn more about identical links. Unscored

77

Best Practices

GENERAL

▲ Uses deprecated APIs — 1 warning found

Deprecated APIs will eventually be removed from the browser. Learn more about deprecated APIs.

Deprecation / Warning	Source
JSDelivr CDN Cdn	
`SharedArrayBuffer` will require cross-origin isolation. See https://developer.chrome.com/blog/enabling-shared-array-buffer/ for more details.	transformers@3.0.0:100

▲ Browser errors were logged to the console

Errors logged to the console indicate unresolved problems. They can come from network request failures and other browser concerns. Learn more about this errors in console diagnostic audit

Source	Description
JSDelivr CDN Cdn	
transformers@3.0.0:100	[0;93m2026-02-09 15:38:12.136799 [W:onnxruntime:, session_state.cc:1168 VerifyEachNodeIsAssignedToAnEp] Some nodes were not assigned to the preferred execution providers which may or may not have an negative impact on performance. e.g. ORT explicitly assigns shape related ops to CPU to improve perf. [m
transformers@3.0.0:100	[0;93m2026-02-09 15:38:12.138099 [W:onnxruntime:, session_state.cc:1170 VerifyEachNodeIsAssignedToAnEp] Rerunning with verbose output on a non-minimal build will show node assignments. [m

TRUST AND SAFETY

○ Ensure CSP is effective against XSS attacks

A strong Content Security Policy (CSP) significantly reduces the risk of cross-site scripting (XSS) attacks. Learn how to use a CSP to prevent XSS Unscored

Description	Directive	Severity
No CSP found in enforcement mode		High

Use a strong HSTS policy

Deployment of the HSTS header significantly reduces the risk of downgrading HTTP connections and eavesdropping attacks. A rollout in stages, starting with a low max-age is recommended. Learn more about using a strong HSTS policy. Unscored

Description	Directive	Severity
No HSTS header found		High

Ensure proper origin isolation with COOP

The Cross-Origin-Opener-Policy (COOP) can be used to isolate the top-level window from other documents such as pop-ups. Learn more about deploying the COOP header. Unscored

Description	Directive	Severity
No COOP header found		High

Mitigate clickjacking with XFO or CSP

The `X-Frame-Options` (XFO) header or the `frame-ancestors` directive in the `Content-Security-Policy` (CSP) header control where a page can be embedded. These can mitigate clickjacking attacks by blocking some or all sites from embedding the page. Learn more about mitigating clickjacking. Unscored

Description	Severity
No frame control policy found	High

Mitigate DOM-based XSS with Trusted Types

The `require-trusted-types-for` directive in the `Content-Security-Policy` (CSP) header instructs user agents to control the data passed to DOM XSS sink functions. Learn more about mitigating DOM-based XSS with Trusted Types. Unscored

Description	Severity
No `Content-Security-Policy` header with Trusted Types directive found	High

PASSED AUDITS (11) Hide

Uses HTTPS

All sites should be protected with HTTPS, even ones that don't handle sensitive data. This includes avoiding mixed content, where some resources are loaded over HTTP despite the initial request being served over HTTPS. HTTPS prevents intruders from tampering with or passively listening in on the communications between your app and your users, and is a prerequisite for HTTP/2 and many new web platform APIs. Learn more about HTTPS.

Avoids third-party cookies

Third-party cookies may be blocked in some contexts. Learn more about preparing for third-party cookie restrictions.

Allows users to paste into input fields

Preventing input pasting is a bad practice for the UX, and weakens security by blocking password managers.Learn more about user-friendly input fields.

Avoids requesting the geolocation permission on page load

Users are mistrustful of or confused by sites that request their location without context. Consider tying the request to a user action instead. Learn more about the geolocation permission.

Avoids requesting the notification permission on page load

Users are mistrustful of or confused by sites that request to send notifications without context. Consider tying the request to user gestures instead. Learn more about responsibly getting permission for notifications.

Displays images with correct aspect ratio

Image display dimensions should match natural aspect ratio. Learn more about image aspect ratio.

Serves images with appropriate resolution

Image natural dimensions should be proportional to the display size and the pixel ratio to maximize image clarity. Learn how to provide responsive images.

Page has the HTML doctype

Specifying a doctype prevents the browser from switching to quirks-mode. Learn more about the doctype declaration.

Properly defines charset

A character encoding declaration is required. It can be done with a `<meta>` tag in the first 1024 bytes of the HTML or in the Content-Type HTTP response header. Learn more about declaring the character encoding.

No issues in the `Issues` panel in Chrome Devtools

Issues logged to the `Issues` panel in Chrome Devtools indicate unresolved problems. They can come from network request failures, insufficient security controls, and other browser concerns. Open up the Issues panel in Chrome DevTools for more details on each issue.

Page has valid source maps

Source maps translate minified code to the original source code. This helps developers debug in production. In addition, Lighthouse is able to provide further insights. Consider deploying source maps to take advantage of these benefits. Learn more about source maps. Unscored

URL	Map URL
JSDelivr CDN Cdn	
...@huggingface/transformers@3.0.0 (cdn.jsdelivr.net)	...@huggingface/transformers.min.js.map (cdn.jsdelivr.net)
`Error: Failed fetching source map (404)`	
...js/bootstrap.bundle.min.js (cdn.jsdelivr.net)	...js/bootstrap.bundle.min.js.map (cdn.jsdelivr.net)

NOT APPLICABLE (2) Hide

Redirects HTTP traffic to HTTPS

Make sure that you redirect all HTTP traffic to HTTPS in order to enable secure web features for all your users. Learn more. Unscored

Detected JavaScript libraries

All front-end JavaScript libraries detected on the page. Learn more about this JavaScript library detection diagnostic audit. Unscored

Captured at Feb 9, 2026, 3:38 PM GMT+1
Initial page load

Emulated Moto G Power with Lighthouse 13.0.1
Slow 4G throttling

Single page session

Using Chromium 145.0.0.0 with devtools

Generated by **Lighthouse** 13.0.1 | File an issue