The State of Web Accessibility Compliance In 2022

Navika Budhraja, Muhammad Huzaifa, Kevin Jang, Arthur Lu

Table of Contents

Abstract 2

Context 2

- Relation to Course 2
- Accessibility Standards 3
- Impact of Accessibility 3

Methodology 5

- Data Extraction Pipeline 5
- Dataset Format 6
- Sampling Bias 8

Results 9

- Key Errors 14
  - color-contrast 14
  - link-name 15
  - image-alt 16
  - aria-required-parent errors in EU Public Service Sites 17
- Trends Across Region and Website Type 17
  - Audit Types With Few Errors 17

Conclusion 19

Abstract

In this report, we also present the historical context of systematic inaccessibility of websites to determine if certain groups of people are made inaccessible from accessing the vital parts of the internet more than others. Using this context, we present a new toolchain that partially automates the processing of web accessibility data. Using this methodology, we provide updated data on the state of web accessibility in regions where web access is crucial. Specifically, we present an up-to-date dataset on the accessibility of a wide selection of public service and government websites between the United States of America and the European Union. From the data collected, we provide some commentary on the results of the data analysis and bring to light some areas in which web designers should aim to improve. Finally, we remark on potential future work with the data and the need for future work on the problem of lack of web accessibility in the field.

Context

Relation to Course

An integral aspect of the report is dispelling the idea that technology remains unbiased. By uncovering how websites remain inaccessible to disabled communities, the report will demonstrate how the historic lack of diversity has discriminatory consequences in the form of "othering" of underrepresented communities. This report will look at how websites continue to oppress individuals with disabilities, which can be attributed to the lack of disabled voices or accessibility advocates during the development process of these websites.

In *Algorithms of Oppression: How Search Engines Reinforce Racism*, Safiya Umoja Noble presents a similar concept regarding the investigation of how technology further perpetuates oppression and othering of marginalized groups. Although Noble is taking a deep dive into how the internet discriminates against race and sex, her work provides context to the report in terms of questioning and critiquing the internet as a form of systematic oppression. Similar to Noble, this report will not only investigate how essential websites fail accessibility requirements, but also take a deeper look into why this is the case, regarding the historic, cultural, and social contexts of the internet and tech industry.

The issues of poor accessibility impact the autonomy every user has to complete their tasks over the internet as websites cannot be used as intended for certain groups. This relates to the unintentional oppression of individuals with disabilities because they must rely on others to navigate the digital environment.

Accessibility Standards

The United States of America amended the Rehabilitation Act of 1973 with Section 508 about updates to website accessibility standards. In this section, Article E205.4 states that electronic content must "conform to Level A and Level AA" standards as specified in WCAG 2.0. The European Union establishes accessibility standards in EN 301 549 Clause 9.0 which states that websites should conform to WCAG 2.0 Level AA. Both organizations use the Web Content Accessibility Guidelines (WCAG) as it is the international standard for accessibility.

Level A is considered the minimum level of compliance where most users should be able to use a webpage, but there will still be components that cannot be remediated by assistive technologies. Level AA provides more accessible components such that most assistive technologies should be able to work on both desktop and mobile versions of the page. This level may not cover every user as site logic and custom components may still remain inaccessible even with assistive technology. There are newer iterations of WCAG with versions WCAG 2.1, WCAG 2.2, and WCAG 3 in development, so it is possible that these federal organizations will update their accessibility standards in the near future.

Despite these federal laws that set accessibility standards, according to the 2022 WebAim Accessibility Report over a million home pages, 96% of all websites do not meet the WCAG compliance requirements for WCAG 2.0 Level A and AA. The report will thus determine if highly visited government and public service websites in the USA and EU are compliant and can comply with the levels of accessibility in accordance with the law.

Impact of Accessibility

One of the less obvious disabilities when it comes to website development is visual impairment. As many front-end developers go by the "What you see is what you get" mentality of programming functional websites for end users, they tend to overlook how accessible their architecture is. For example, blind or partially sighted people would not be able to see the website in the way it is intended to be. Images or film media would need descriptions to tell these users what is being displayed. Although there are accessibility tools such as screen readers and tab-indexing to make navigation easier, the inaccessible architecture may send the wrong signals to the user.

A screen reader may read a mislabeled <button> label as a <a> link and thus the user will not want to engage with it because it may take them to a page they do not want to be at. Using ARIA labels incorrectly shows an effort to be accessible, but only makes it harder for people with visual impairments to know where they are on a page. In addition, blind or partially sighted people may have to rely on the keyboard instead of a mouse because it is difficult to pinpoint the positionality of the cursor.

Keyboard users, in conjunction with screen reader technologies, would utilize tab-indexing to navigate through links. Frontend developers may unintentionally prevent these users from reaching links by setting the tabindex to -1 or using tabindex on elements that do not require it. When a user is trapped in a section like this, they are unintentionally locked out of content available to people who don't experience the same extent of disabilities. Mouse users can click over any part of the website, but keyboard users must struggle with the linear logic flow over the webpage DOM.

Furthermore, digital media would be experienced differently by people with visual impairments and people with hearing impairments. While popular sites that specialize in media publishing offer alt-text and transcripts to their embedded content, most websites do not offer the same equity. These concerns may come from the various ways to interpret the content of an image or film but there are many recommendations developers can follow. Transcribing accessibility tools can help deaf viewers understand what speakers are saying, but disregards the auditory elements in the foreground. There is a tradeoff between what can be accessible and what is excluded so it is important that other areas aside from media maintain levels of inclusion in accordance with accessibility standards. In addition, multilingual media might be harder for machine learning tools to transcribe, so this is another challenge for developers to explore.

With these concerns in mind, this report determines how accessible critical governmental and public service infrastructures are to users with primarily visual impairments. The report will discuss whether more effort is needed to address accessibility on these sites. The report also discusses the aspects or users who might be the most impacted in the case of poor design. Through the results, web developers will be more conscious of the kinds of issues that prevent users from fully utilizing their web services and have the motivation and understanding to fix these problems.

Methodology

To explore this issue, we needed to collect a set of accessibility metrics from websites and store them in a dataset. We categorized the websites primarily by their region, the United States of America or the European Union, and their purpose as either government-managed or a public service. We defined these terms as a governmental site is one that is strictly for information about the government while a public service site serves the public sector. To get accessibility metrics, we used Google Lighthouse because it contains 53 specific audit types as specified within WCAG guidelines as well as how data could be saved conveniently as JSON files. Audits were performed by manually running Google Lighthouse on the website URLs with the device-type set to “Desktop”, mode set to “Navigation”, and categories only containing “Accessibility” for the runs. Then we downloaded the JSON file outputted from the audits to parse further because the data contains many sections that are irrelevant to data analysis. Using this parsed data, we inputted it into a Google Sheets dataset that contains the accessibility rating of websites separated by four sections.

Data Extraction Pipeline

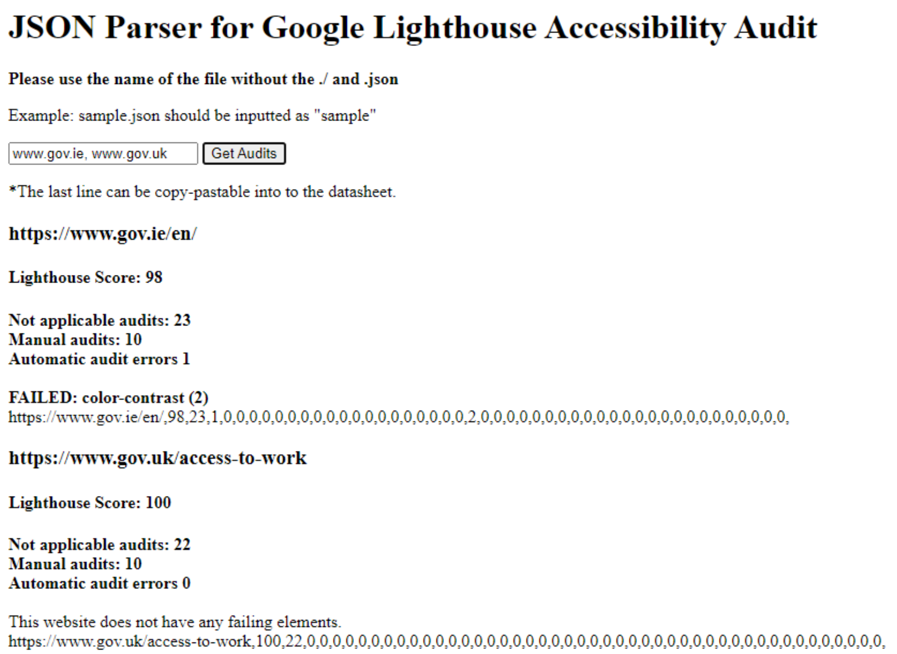
Manual data extraction would have been a slow and tedious process, so we pipelined the process to clean and format the extracted data to put into the dataset. We cleaned the name of the JSON files to represent the URL of the webpage by running a bash script to remove a time-identifier attached to the end of the file (reformat.sh). This made viewing the data easier as the files only consisted of human-readable phrases.

To get the relevant contents of the file, we used JavaScript. JavaScript worked easily with JSON because both use similar syntax for storing data as compared to other programming languages. JavaScript could easily get the data inside the key-value pair format, so we did not need to iterate through the file line-by-line as what Java or C may have required. The script would open the JSON file and step through a few layers of keys to get to the 53 audit types of Google Lighthouse. The dataset contains only 43 of those types because ten of them require manual evaluations from the user. To access the accessibility rating of a website, we counted the number of errors detected by Google Lighthouse for each of those categories and stored them in a variable to be used in data extraction later. The implementation counts “not applicable” types as having no errors just like how some “binary” types would have no errors as well. Refer to parser.js in the included files for this code.

Since JavaScript requires HTML to be run, we created a barebones HTML page (parser.html) to execute the script. An input field would take in a comma-delimited list of website URLs that represented the names of the reformatted filenames. To get a copy-pastable list of the local repository of JSON files, another bash script would print a comma-delimited list of filenames to the console (getFilenames.sh). This would be inputted into the input field and

pressing enter or the button would execute the parsing script. We utilized the HTML page further by extracting the data as text to an HTML node. The variable that held the number of errors per type was then formatted into a comma-delimited text that we could copy and paste into the dataset.

The following screenshot of the HTML page show an example of inputting and extracting from two EU government websites.



Dataset Format

The dataset contains the following metadata that identifies the particular site in terms of region and category, along with specific countries or cities and purpose.

Region	Country	Category	Purpose	Site Name	Site URL

Example:

EU	Ireland	Government Services	Information	Government of Ireland	https://www.gov.ie/en
EU	UK	Government Services	Information	UK Gov	https://www.gov.uk/access-to-work

The quantitative data is shown below. The Lighthouse Accessibility Score is the overall score given to the website based on the weighted binary audit factors. “NA” represents the number of audits that were not applicable to the website. If a binary audit fails, it would increment the number of types that failed. For each of those failed binary audits, its type would be identified from the 43 audit types in the list below, and the total number of elements that failed would be set there. By default, the not applicable types and binary audits with no errors have values of zero. The final field counts the total number of these elements that failed to get a sense of how many cumulative errors were found in the website beyond the specific types.

Accessibility Score	Total NA Types	Total Types Failed	Total Elements Failed

Example (cont.)

https://www.gov.ie/en	98	23	1	2
https://www.gov.uk/access-to-work	100	22	0	0

Google Lighthouse Audit Types (Automatic)

accesskeys	aria-valid-attr-value	image-alt
aria-allowed-attr	aria-valid-attr	input-image-alt
aria-command-name	button-name	label
aria-hidden-body	bypass	link-name
aria-hidden-focus	color-contrast	list
aria-input-field-name	definition-list	listitem
aria-meter-name	dlitem	meta-refresh
aria-progressbar-name	document-title	meta-viewport
aria-required-attr	duplicate-id-active	object-alt
aria-required-children	duplicate-id-aria	tabindex
aria-required-parent	form-field-multiple-labels	td-headers-attr
aria-roles	frame-title	th-has-data-cells
aria-toggle-field-name	heading-order	valid-lang
aria-tooltip-name	html-has-lang	video-caption
aria-treeitem-name	html-lang-valid	

To view how the 43 types look in the dataset, refer to the **[Fig 4]** heatmap in the following Results section.

Sampling Bias

One flaw in the methodology was that choosing websites to audit was not a random sample. Most of the websites audited, come from well-known nations or cities. Public service data was collected by taking a subcategory within public services, such as airports or libraries, so that this data may be compared with each other. Likewise, government websites were often about main fields such as civil courts or major banks. However, the dataset consists of at least 30 samples for each region and purpose to have a large enough sample size when assessing the specific audit categories. This allows us to draw conclusions that reflect on how accessible websites with high engagement can be, but not for websites that have less engagement such as suburban government and public services. As there are many nations in the European Nation that were not sampled and each nation didn't have at least 30 samples, this paper can only make analyses based on the United States and the European Union as a whole based on their different standards of accessibility.

Results

As the dataset contains many fields to analyze, heatmaps are generated for the four combinations of region and site category over the accessibility audit types. The heat maps show the number of errors for each site and for each error category. All data points are normalized against the maximum value per feature. That is, each column representing a type of error is normalized against the maximum value in each column. Finally, the samples with zero errors per feature are colored white to bring more focus to data with non-zero errors. The heatmaps are included on the following pages and higher fidelity images can be fetched from the Heatmaps folder provided in the zip file.

To analyze our results, we looked at the data in different ways. For specific audit types, we might analyze them separately among the four categories or look at all categories together and compare them with the other audit types. We also looked for trends across the dataset among types as well as between the US and EU.

Fig 1. Heatmap of US public service sites vs number of errors reported

Fig 2. Heatmap of US government sites vs number of errors reported

Fig 3. Heatmap of EU public service sites vs number of errors reported

Fig 4. Heatmap of EU government sites vs number of errors reported

Key Errors

In the dataset, specific error types were found to appear more frequently than others across the board, or experience higher error rates than other accessibility tests.

color-contrast

Color-contrast errors were the most common across the entire dataset, as around 50% of the websites visited had at least one color-contrast error in their Lighthouse audit. Not only is there a high frequency of color-contrast errors across the dataset, but many websites experienced a significant number of color-contrast errors within the evaluation of the site itself. In fact, out of sites that experienced at least one color-contrast error, the average number of color-contrast errors was around 13 per site, which is a significant amount in comparison to other error types. To provide some context, the official Web Content Accessibility Guidelines' (WCAG) definition of color contrast is a "measure of the difference in perceived "luminance" or brightness between two colors" (WebAim). This difference is measured by a ratio that ranges from 1:1 (white on white) to 21:1 (black on white). According to WCAG, success in color-contrast is measured by three specific criteria:

I. Contrast (Minimum): The visual presentation of text and images of text has a contrast ratio of at least 4.5:1
II. Contrast (Enhanced): The visual presentation of text and images of text has a contrast ratio of at least 7:1
III. Non-text Contrast: The visual presentation of User Interface Components and Graphical Objects must have a contrast ratio of at least 3:1 against adjacent color(s)

As a result, maintaining these standard ratios for color-contrast is integral to making a website accessible, especially for those who are vision impaired. Thus, reading text or viewing images with poor color-contrast makes for an inaccessible website for those with vision-related disabilities. For instance, this graphic from Google's Web Development guide highlights the significant differences between color-contrast ratios in relation to the readability of text.

(Image from Google's Web.dev Project)

Thus, the fact that almost half of the websites visited encountered at least one color-contrast error demonstrates the frequency of color-contrast errors across the web and subsequently, how this is detrimental to those with vision impairments. Since solving color-contrast issues seem simple, it is therefore surprising how frequently these errors occur. However, it may very well be the simplicity of these concepts that contribute to color-contrast errors being overlooked.

link-name

Another common error across the dataset was link-name, with around 38% of the websites visited experiencing at least one link-name error in their Lighthouse audit. Similarly, individual websites that had at least one link-name error experienced generally high frequencies of these errors with an average of 6.5 link-name errors per site. According to Lighthouse, a link-name error occurs when a "link does not have a discernible name," which can be linked to multiple different problems, especially in the context of accessibility. More generally, when a link is missing discernible text it means that there needs to be information provided for a user to understand the purpose of the link and where it is taking them. For instance, Dequeue University, a website dedicated to teaching accessibility in development, lists out numerous issues associated with link-name errors that might need to be fixed within a website:

I. Link Text is hidden to a screen reader (including inner link text)
II. A Link does not receive programmatic focus (i.e. using device-specific events to program behavior)
III. Missing aria-labels on the link

As mentioned before, link-name has a multitude of accessibility implications, especially since links are integral elements of a website. 2 important implications to missing data surrounding link-name include

I. Users who exclusively rely on a keyboard to navigate through a page may not be able to access links that do not have programmatic focus
II. Users who rely on the screen reader to navigate through a website will not be able to know where the link is pointing to.

Therefore, ensuring the accessibility of links is integral in developing a website that is easy to navigate for all users no matter which modes they use to access the site. Thus, the high frequency of link-name errors demonstrates how accessibility is once again overlooked either in the desire to develop a product efficiently or a company and/or developer's lack of understanding that there are multifarious ways in which websites can be navigated.

image-alt

Another common error was image-alt. This error occurred 201 times in total with public service sites accounting for 141 or around 70% of all image-alt errors. According to Dequeue University, this rule checks if images have alt text. Alt text is used by screen readers to convey image contents and purpose to users. This is critical for users who cannot see or have impaired vision and rely on screen readers to access web resources. For these people, visual information in images needs to be provided by alt text so that users can understand the information conveyed. Dequeue highlights a few methods to attach alt text to an image:

I. Using an alt attribute
II. Using an aria-label
III. Using an aria-label led by attribute

Dequeue also provides some guidelines on what alt text should contain:

I. Why is the non-text content here?
II. What information is it presenting?
III. What purpose does it fulfill?
IV. If I could not use the non-text content, what words would I use to convey the same information or function?

Providing the answers to the questions by developers is integral in providing the context and purpose of an image to those who are vision impaired. The high number of errors relating to missing alt text demonstrates a lack of consideration of accessibility for vision-impaired users. These errors occur when the alt is not specified, which indicates that the failures did not have time or effort to populate the alt text information. Thus, the frequency of errors suggests again the overlooking of accessibility in favor of quickly making a product.

aria-required-parent errors in EU Public Service Sites

An interesting trend found in the data was that European Union public services websites had a high frequency of aria-required-parent in comparison to other types of websites collected and analyzed. In fact, out of all of the EU public service sites visited, there were a total of 61 aria-required-parent errors as opposed to 2 in EU government sites, 10 in US government sites, and 1 in US public service sites. To provide background, aria-required-parent errors occur because an ARIA role is not contained by its associated parent element. This error is integral to accessibility because defining proper ARIA roles allow for assistive technologies (screen readers, text-to-voice, etc.) to understand website elements and interactions. As to why EU public service sites have more aria-required-parent errors than the other categories, the justification seems inconclusive. According to the European Union, websites affiliated with EU institutions follow the Web Content Accessibility Guidelines, which are generally accepted by most countries, including the United States.

Trends Across Region and Website Type

Overall, the trends in errors were similar between European countries and the US. This is somewhat expected as these countries must comply with the same WCAG 2.0 standard for websites in the public sector. According to eSSENTIAL Accessibility, European Union websites aim to “improv[e] digital accessibility across Europe”. Although the regulations stated in EN 301 549 may have improved accessibility there were still many errors in the websites. As for the US, Section 508 of the Rehabilitation Act requires a minimal level of accessibility. Overall, the accessibility of both US and countries in the European Union regarding government and public services sites could still use further improvement.

Audit Types With Few Errors

A similar trend between all four categories was how certain audit types did not have many errors. A threshold value of 5% of sites reported was used to determine if the category type was rarely at fault. For example, the dataset for US government sites had 39 websites and the threshold would be at most 1.95 websites with errors. Only one website for aria-valid-attr had an error, so it passed this threshold. However, aria-required-children had errors in two websites so it was above this threshold.

Across all four categories, these are the types that were below the threshold value. Types in which at least one of the categories did not pass the threshold were not included. From the

example above, aria-valid-attr passed the individual thresholds for each of the categories, while aria-required-children failed for US government and US public sites.

accesskeys	aria-treeitem-name	html-lang-valid
aria-command-name	aria-valid-attr	input-image-alt
aria-hidden-body	bypass	meta-refresh
aria-progressbar-name	definition-list	object-alt
aria-required-attr	dlitem	td-headers-attr
aria-required-parent	document-title	th-has-data-cells
aria-toggle-field-name	form-field-multiple-labels	valid-lang
aria-tooltip-name		video-caption

This result may infer that because these types rarely had errors, they might be generally not present on the website. Many of these types such as aria-progressbar-name and aria-toggle-field-name would only be used in specific use cases that a majority of websites may not have. It is expected that document-title rarely have errors because by definition they would need to lack a title element, and all of the websites surveyed have had their own titles. Deque University states that “Navigating through pages can quickly become difficult and confusing for screen reader users if the pages are not marked with a title. The page title element is the first thing screen reader users hear when first loading a web page.” This is a positive outcome as the visually impaired can find these important government and public service websites during a search query rather quickly.

In addition, valid-lang may suggest that the website developers understand what kinds of language values they can use. This should be expected as the websites are likely built by local employees in the region and many of the websites have options to change the page’s language either automatically by detecting the browser or by an option on the website. With valid-lang as one of the types that don’t encounter errors in the multiple languages across the EU means that screen readers would be able to read the pages in the appropriate language because “Screen readers can switch between these language libraries easily, but only if the documents specify which language(s) to read”, according to Deque University.

Other notable types that might be rarely used or at fault are aria-input-field-name, aria-roles, and frame-title where only one category did not pass the threshold. The titles of these types are self-explanatory yet it is interesting how they might be rarely at fault because search bars and embedded iframes are common on websites. Screen readers should be able to understand what most input fields are about and can insert data accordingly, while embedded frames would be easily identifiable on most websites.

Conclusion

In conclusion, this project audited, analyzed, and evaluated the current state of accessibility in informational websites in the US and the EU, uncovering the ways in which websites can improve accessibility across the board.

While this data has been taken over the more engaged sites as compared to suburban or local areas of service, anyone can benefit by understanding what the most common error is and reevaluating their website for accessibility improvements. In addition, even though the US and the EU have enacted laws and regulations regarding accessibility standards of websites, it is nevertheless important to review the relevance of these standards through audits such as the ones being conducted in this project.

While there have been significant improvements made toward accessibility in digital spaces, more can always be done to improve accessibility on websites. In the analysis of the dataset, a few key errors were found across all categories of websites that continued to experience a high frequency of errors, demonstrating that there are still areas of improvement surrounding accessibility. For example, the high frequency of color-contrast errors, link-name errors, and image-alt demonstrates seemingly minor changes to a website, but these design decisions with a major impact on accessibility have yet to be made properly on most websites. Although WCAG guidelines are quite comprehensive and are universally enforced in the EU and US, minor errors such as those listed in the results section still persist, demonstrating the lack of prioritization for fixing accessibility errors. Many of these errors have quite simple fixes to them, especially since auditing tools like Lighthouse provide supporting documentation surrounding the errors, so there is not much overhead for a developer/company to make widespread fixes to accessibility errors. Thus, making these minor changes to a website goes a long way in terms of accessibility.

In regards to the project itself, there are several improvements to the methodology that could be done in the future. For example, a random sampling of websites rather than letting personal and algorithmic bias affect what websites were chosen could improve data accuracy. A list of sites could be sourced from other researchers or governmental services that may have a list of public service and government websites and their domains. Ideally, this list would have all known domains used so there is no personal bias over the purpose of the website to look for nor algorithmic bias that comes from picking the first option that comes up in a search query. Using this list of domains, the data could be categorized and then a randomizer could iterate over the dataset to pick any number of domains. This method may help with auditing websites that get less engagement, such as suburban services that would likely be known locally instead of by global researchers.

Additionally, not applicable types were counted the same way as having no errors. This may have affected the results for rarely faulted audit types because it could be that many of those

types discovered were actually not applicable to a majority of the websites. In the future, not applicable types should be marked as separate from binary types with no errors to check the validity of this result.

Furthermore, the ten manual audit types of Google Lighthouse were not included in the dataset because the audit does not identify if the site has issues with these. The ten manual types are are custom-controls-labels, custom-controls-roles, focus-traps, focusable-controls, interactive-element-affordance, logical-tab-order, managed-focus, offscreen-content-hidden, use-landmarks, and visual-order-follows-dom. As a follow-up to this project, someone could develop ways to audit these dynamic or customized types. Focus traps and logical tab order are highly important when it comes to navigation because they could limit how keyboard users can get around the page. As a possible solution to focus traps, a custom audit that uses a timeout system to reach all known tabbable elements in the website would be an interesting field to explore and propose to the developers at Google Lighthouse to improve this accessibility technology.

Related work that can be performed using this data is the focus on development of assistive technologies for users with disabilities. Developers are bound to miss some accessibility everywhere when there are hundreds of pages beyond the webpage. It could be a more affordable option to develop existing assistive technologies as many of these errors usually directly affect people with disabilities. While Google Lighthouse audits identify what the errors are passively, it might be possible for an assistive technology to dynamically fix some of those errors as a user navigates the page. With the results of this data, a technology that enhances the color contrast by reading the page's colors and changing them based on the user's disability can help to resolve many of these errors. In addition, machine learning developments can be applied to images to create functional alt text, although this is still limited by what is intended to be shown by the publisher.

Future work is still required in this area. Focusing on the United States and the European Union, larger datasets with periodic audit reports taken every month would provide insight into the developing progress of web accessibility in these two regions. This would be a useful way to hold the US and nations in the EU accountable for their design decisions as it comes to their accessibility. Automated processing could also be implemented which gives common errors, which can inform developers of current common shortcomings in order to solve the problems which are the most prevalent first. Such a dataset would be useful for government officials continue to report on the state of web accessibility. With more research into this development, other researchers can view accessibility in other regions and areas of service to determine the next steps in improving accessibility for all users. Implementation of universal compliance will make the internet more inclusive and will end up educating developers on why accessibility should always matter when building websites.

Works Cited

Commented [1]: gonna go with MLA for now

"About the ICT Accessibility 508 Standards and 255 Guidelines." *Revised 508 Standards and 255 Guidelines*, U.S. Access Board, https://www.access-board.gov/ict/#E205.4.

"Accessibility Requirements for ICT Products and Services - Clause 9.0." *EN 301 549 V3.2.1*, ETSI, https://www.etsi.org/deliver/etsi_en/301500_301599/301549/01.01.02_60/en_301549v010102p.pdf.

"Contrast and Color Accessibility understanding WCAG 2 Contrast and Color Requirements."*WebAIM*, https://webaim.org/articles/contrast/.

"Documents Must Contain a Title Element To Aid In Navigation." *Deque University*, Deque University, https://dequeuniversity.com/rules/axe/4.4/document-title

eSSENTIAL Accessibility. "EN 301 549: The European Standard for Digital Accessibility." *ESSENTIAL Accessibility*, 5 Sept. 2022, https://www.essentialaccessibility.com/blog/en-301-549.

Gash, Dave, et al. "Color and Contrast Accessibility." *Web.dev*, https://web.dev/color-and-contrast-accessibility/.

"Images Must Have Alternate Text." *Deque University*, Deque University, https://dequeuniversity.com/rules/axe/4.3/image-alt.

"Lang Attribute Must Have a Valid Value." *Deque University*, Deque University, https://dequeuniversity.com/rules/axe/4.4/valid-lang.

"Links Must Have Discernible text" *Deque University*, Deque University, https://dequeuniversity.com/rules/axe/4.4/link-name.

Noble, Safiya. *Algorithms of Oppression*. New York, New York University Press, 2018.

"The WebAIM Million: The 2022 Report on the Accessibility of the Top 1,000,000 Home Pages." *WebAIM*, https://webaim.org/projects/million/.

"Web Content Accessibility Guidelines (WCAG) 2.1." *W3C*, https://www.w3.org/TR/WCAG21.