Tampere University

Aki Lempola

# COMPARING AUTOMATIC ACCESSIBILITY TESTING TOOLS

Faculty of Information Technology and Communication Sciences
M. Sc. Thesis
May 2023

# ABSTRACT

Aki Lempola: Comparing automatic accessibility testing tools
M.Sc. Thesis
Tampere University
Master's Degree Programme in Software Development
May 2023

---

Recently, the Web has become increasingly important as information and many essential services move to the Web. Accessibility aims to make services to users with disabilities. Web accessibility's goal is to make the Web accessible, which means disabled people can use the Web. It has been estimated that 15% of the world's population lives with some form of disability, and the aging population makes web accessibility increasingly important. Similarly, recent legislation Increasingly requires the Web to be accessible to all.

Web accessibility evaluation can be done to ensure that the website conforms to the needs of disabled people or legal requirements. There exist different accessibility evaluation methods, each with its benefits and drawbacks, and the methods often complement each other. Automatic testing tools are an important part of accessibility testing. There are many different automatic accessibility evaluation tools to choose from. And previous studies show that tools detect a different number of issues.

In this thesis, we compared three automatic accessibility testing tools in terms of how many success criteria they cover, testing speed, and the number of detected issues. Tools were used to test Finnish e-commerce sites and a test site containing a set of accessibility issues.

We found that the WAVE was the fastest tool to scan pages. IBM Accessibility Checker covered the greatest number of WCAG success criteria. The number of detected issues depends on the selected page and the type of accessibility issues present on the page. In five out of six tested pages, IBM Equal Access Accessibility Checker found the greatest number of issues, and in one out of six pages WAVE found the greatest number of issues.

Key words and terms: Web accessibility, WCAG, comparison, automatic tools.

# Contents

# 1. Introduction

Web accessibility is an essential part of making a web content usable to all. For non-disabled people, it is trivial to perceive, operate, and understand content on a website. Non-disabled people, can easily read, navigate, watch, and listen to media content, but what about disabled people who may not experience the Web similarly? If the Web is not accessible, people with disabilities may be unable to use websites and services. Web accessibility is essential as many services are available online, and sometimes only online.

Disabled people benefit from the accessibility, but also non-disabled people may benefit from better accessibility [Schmutz et al. 2016]. Aging people may experience deterioration of cognitive and or physical skills and senses, thus benefit from web accessibility [Richards and Hanson 2004]. Accessibility requirements may also improve usability for all users, especially in challenging situations, such as noisy environments, bright sunlight, or small screens [WAI 2023b].

The goal of web accessibility is to make the Web usable to disabled people. The Web is not accessible (or inaccessible) by chance; it takes an effort from developers and content producers to make the Web accessible to people with disabilities. When the Web is accessible disabled users can perceive, understand, operate, and contribute to the Web. World Wide Web Consortium (W3C) proposed Web content accessibility guidelines (WCAG) to make the Web more accessible [W3C 2018]. These guidelines have become widely used standards and are made into the legislation of some countries, for example, in European Union [European Commission 2016]. In the European Union public sector websites and mobile applications are required to conform to web content accessibility guidelines conformance level AA.

Web content accessibility guidelines [W3C 2018] provide a set of guidelines and success criteria that help to make the web more accessible for people with disabilities. WCAG 2.1 has a total of 78 testable success criteria. More specifically, these guidelines accommodate the needs of people with blindness and loss of vision, deafness and hearing loss, limited movement, speech disabilities, photosensitivity, and some coverage also for learning and cognitive disabilities [W3C 2018].

Evaluation of accessibility makes sure that the Web is usable for disabled users or that the Web content conforms to the requirements of guidelines. Automatic testing tools are an important part of evaluation because sometimes websites may contain hundreds, if not thousands, of pages. Manually evaluating all this content would be too time-consuming for expert evaluators. But it's not possible to detect all accessibility issues automatically, as some of the accessibility issues require human interpretation. And manual evaluation is needed to make sure a page is accessible. Nevertheless, more issues detected automatically lead to less work for manual evaluation.

There are many automatic accessibility evaluation tools to choose from. W3C's [2023c] Web accessibility evaluation tool list currently lists 167 tools. This list is not an essentially comprehensive list of all automatic accessibility evaluation tools. Different types of tools available include online tools, browser plugins, desktop applications, etc. Automatic accessibility evaluation tools may possibly detect a different number of issues, and different types of issues while scanning the same page [Ismailova and Inal 2022]. The use of more than one automatic accessibility evaluation tool may improve confidence in the results of the automatic evaluation tools [Padure and Pribeanu 2020; Ismailova and Inal 2022; Frazão and Duarte 2020].

This thesis aims to gain an understanding of web accessibility, different accessibility evaluation methods, and what are their pros and cons. Then learn what legal requirements for web accessibility there are in Finland. The tools are compared to answer the following research questions:

**RQ 1**: What success criteria automatic accessibility testing tools cover?

**RQ 2**: Is there a difference between selected tools features?

**RQ 3**: Do the tools detect different issues?

This thesis consists of seven chapters. in Chapters 2, 3, and 4 we go over the background of the topic. In Chapter 2 we cover the components of web accessibility, and Chapter 3 goes more in detail about web accessibility and legal requirements for web accessibility in Finland. In Chapter 4, we cover the different methods of evaluating accessibility. Next in Chapter 5, we go explain the method used to compare three automatic accessibility evaluation tools on two Finnish e-commerce websites and a test site that contains a set of accessibility issues. In Chapter 6, we go over the results of the thesis. And finally, Chapter 7 is the conclusion of the thesis.

# 2. Components of web accessibility

In the past, responsibility for web accessibility has been on the content producers (developers), but this view misses several other components of accessibility [Chisholm and Shawn 2005]. Web accessibility comprises web developers, end users, tool developers, user agents, authoring tools, and evaluation tools. Web developers produce content using authoring tools and evaluate the pages using evaluating tools [Chisholm and Shawn 2005]. End users use web-browser, media players, assistive technologies, or other user agents to access the web content [Chisholm and Shawn 2005]. And tool developers add features to authoring tools and evaluation tools according to the users' needs [Chisholm and Shawn 2005]. This chapter goes over these independent components of web accessibility in more detail.

## 2.1 Technical components

Technical components of web accessibility include technical specifications, content, and tools [Chisholm and Shawn 2005]. Technical specifications define how to use the features of the language to create web content [Chisholm and Shawn 2005]. For example, HTML (hypertext markup language) defines an <img>-tag for image content and an alt-attribute for text alternative. Then assistive technologies such as screen readers can use the alternative text to convey the same information as the image content.

Content is the information that forms web pages and web applications [Chisholm and Shawn 2005]. The web content includes technical content and natural information content [Aboy-Zahra 2008]. Technical content consists of the markup and code that describes how the content is displayed and how the user interface functions. Natural information content includes the information contained on web pages, text, multimedia, images, etc.

HTML, CSS (cascading style sheet), and JavaScript are examples of languages developers can use to present web content. HTML first became accessible in 1997 with the release of HTML 3.2, and since then, additional elements and attributes have been added [Kirkpatrick 2006]. Using semantic HTML tags makes the site more accessible, as assistive technologies can interpret the semantic information from the tags. When developers discovered that they could use JavaScript to edit the markup dynamically, accessibility problems became more common, as some of the content was only accessible with a JavaScript-enable browser [Kirkpatrick 2006].

Tools include authoring tools, evaluation tools, user agents, and assistive technologies [Chisholm and Shawn 2005]. Authoring tools include any software or service that developers can use to create and modify web content; this includes blogs, wikis, and development tools [Chisholm and Shawn 2005]. User agents are tools the end-users use to access web content, such as browsers, media players, and assistive technologies [Chisholm and Shawn 2005].

## 2.2 Human components

Web accessibility consists of a few human components. Content producers make web content, including developers, designers, writers, editors, and anyone who creates or edits web content [Chisholm and Shawn 2005]. Content producers can use guidelines and evaluation tools to check if the content is accessible. Different stages of development may require other authoring and testing tools [Aboy-Zahra 2008]. People may belong to many of these categories, and end-users may also be content producers or tool developers.

End-users use different user agents to access the web content. End-users have different abilities and disabilities; some may need assistive technologies to access web content. Assistive technologies, such as screen readers, can only read the headings and navigation links if the content is accessible and the assistive tools are developed to support the technical standards.

Tool developers design and develop tools to create and access web content [Chisholm and Shawn 2005]. Suppose the authoring tools support and promote to use of accessibility. In that case, the developers are at least more aware of the accessibility needs and may be more inclined to implement them.

# 3. Accessibility

Non-disabled people can use a mouse to navigate, see the screen and read the text on the website, listen to audio, watch flashing media content, concentrate on the page content, and not get distracted by ads. Skimming the content and headings and navigating links is a trivial task for them. But what about people with disabilities who can't access the Web similarly? They may need to use some other assistive technology, such as screen readers, voice controls, or special hardware, etc. to access the content on the Web. [Rutter et al. 2006]

The world wide web was initially designed to be used without a mouse and eyes if necessary [Rutter et al. 2006]. The world wide web has advanced a long way from the early days. Nowadays, the Web is filled with live multimedia content, audio, images, animations, videos, etc. The media content can make the Web attractive to look at and use. Making the Web accessible to all doesn't mean making the Web boring or stripping down the design of the site, as one harmful untruth myth about web accessibility claims [Rutter et al. 2006].

In this chapter, we will first go over the definition of web accessibility and some closely related terms, why accessibility is essential, and who benefits from the accessible Web. Then we will go over the legal requirements for web accessibility set in Finland and in the *European Union* (EU). Last part of this chapter, we go over the *Web Content accessibility guidelines* (WCAG), which is a widely used standard for web accessibility.

## 3.1 What is web accessibility?

The definition of web accessibility is widely discussed in the research, and there are many different definitions with different scopes and natures. These definitions differ in many ways. Some refer to a different level of interaction, some refer to equal access, some specifically refer to disabled users and some to all users, and some may focus on usability properties [Yesilada et al. 2012]. Yesilada et al. [2012] conducted a survey to understand which definition is preferred and why. The study presented five different definitions of accessibility. They found that the definition by W3C [2023a] was the most popular chosen by 45% of respondents among the given opinions.

Petrie et al. [2015] analyzed 50 different definitions of accessibility to understand the critical components of web accessibility better. The most important concepts of web accessibility definition from their study are [Petrie et al. 2015]:

- target group (users with disabilities, all users, etc.)
- what users should be able to do (access, use, navigate, understand, etc.)
- technologies used (assistive technologies)
- characteristics of the websites (usability or aspects of usability)

- design and development
- characteristics of the situations of use.

This thesis will stick to the definition by WAI [2023a], which states: "*Web accessibility* means that websites, tools, and technologies are designed and developed so that people with disabilities can use them. More specifically, people can: perceive, understand, navigate, interact with the Web, and contribute to the Web.". Web accessibility is closely related to usability and inclusion when developing a Web that works for everyone. [WAI 2023a].

*Usability* has multiple attributes: learnability, efficiency, memorability, errors, and satisfaction. Learnability measures how easily a new user can pick up and start using a new system efficiently. When the system is efficient, users who have learned to use the system have a high level of productivity. Memorable systems allow casual users who may use the system infrequently to use the system without the need to relearn how to use the system every time they need it. The system should have a low rate of errors, and when the user makes an error user should be able to recover from the error state. And lastly, the system should be satisfying to use, so the users can achieve their goals and enjoy the process. [Nielsen 1993]

While web accessibility focuses on people with disabilities, usability in research and practice doesn't specifically consider the needs of people with disabilities [WAI 2023b]. Accessibility and usability may overlap significantly, as many accessibility requirements can improve usability for all users, especially in challenging environments [WAI 2023b]. For example, when accessibility requirements are covered website is more usable with small screens in bright sunlight, users can read transcripts of audio in noisy environments where they cannot listen to the audio, and users can listen to the screen reader reading the text of the page without needing to focus on reading the screen. Accessibility and usability problems can be seen in two overlapping sets that include three categories [Petrie and Kehir 2007]:

- Problems that only affect disabled people.
- Problems that only affect non-disabled people.
- Problems that affect both disabled and non-disabled people.

*Inclusion* means diversity and involvement of as many people as possible [WAI 2023b]. Inclusion includes various issues, including accessibility for people with disabilities, access to software hardware and internet connectivity, computer literacy and skills, economic situation, education, geographic location, culture, age, and language [WAI 2023b]. Thus, an inclusive Web means a web usable for as many users as possible for people with all kinds of skills, needs, and backgrounds. While accessibility is focused on the needs of disabled people, it does cover part of other inclusion issues, as improvements in accessibility are not solely beneficial for disabled people.

Web accessibility is increasingly important as more essential services, such as banking, education, health care, employment, etc., are moving to the Web. And in some cases, these services may only be available online.

Making the Web accessible is best when done from the start of the design process and kept on the minds throughout the implementation process of the webpage [Rutter et al. 2006]. But this is not always possible, as making the Web accessible also includes analysing the fixing existing accessibility barriers on old sites and discovering and understanding factors that influence the accessibility of the Web [Karat et al. 2008].

### 3.2 Who benefits from accessibility?

To understand what kind of accessibility barriers users encounter, we must first consider what disabilities users may have and how these disabilities affect their way of using the Web. The Web itself may offer opportunities and services for disabled people who would otherwise have difficulty accessing them.

In the context of web accessibility and later in this document, disability means disabilities that affect access to the Web, including [WAI 2023a; Rutter et al. 2006]:

- auditory,
- cognitive,
- neurological,
- physical,
- speech, and
- visual.

Disabilities within these categories may vary significantly. For example, visual impairments may range from minor vision loss to blindness. World Health Organization [2011] estimates that about 15% of the world's population lives with some form of disability. Older adults, a large and growing part of the population [Richards and Hanson 2004], also benefit from accessibility. When we get older, most of us experience some loss in cognitive or physical skills [Karat et al. 2008].

Non-disabled people may also prefer using an accessible website over a non-accessible one, as highly accessible websites lead to better task completion time and completion rate than a site with low accessibility [Schmutz et al. 2016]. The highly accessible site also received better ratings for perceived usability, aesthetics, workload, and trustworthiness than sites with low accessibility [Schmutz et al. 2016].

### 3.3 Legal requirements in Finland

The directive on the accessibility of websites and mobile apps [European commission 2016] aims to standardize the laws about accessibility requirements of websites and mo-

bile applications of public sector bodies within member states European Union. The directive lays the foundation and schedule for the accessibility laws that member states include in their laws.

In Finland, the act on the provision of digital services [Act 306/2019] put in place the accessibility requirements for public service websites and mobile applications. The law has three requirements for the services [Act 306/2019]:

1. The service must satisfy the Web Content Accessibility Guideline 2.1 level A and AA requirements.
2. The service must keep an up-to-date accessibility statement that states:
    a. The state of the accessibility of the service, state the parts of the service that don't fulfil the accessibility requirements, and the reason why.
    b. Instructions on how the user may get the information of the digital service in an alternative way if the part of the service is not accessible.
    c. Service providers' digital contact information where the user can send accessibility feedback.
    d. Link to supervising authority, so the user can make an accessibility complaint.
3. The service must have a digital feedback channel, so the user can submit feedback about accessibility issues. The service provider must send an acknowledgment that they received the feedback and respond to the feedback within 14 days.

The main target of the act on the provision of digital services [Act 306/2019] is public sector websites and mobile applications, such as schools and authorities, but also a part of the private sector is subject to the law. This includes:

- Actors of the financial sector, i.e., banks.
- Insurance companies.
- Water and energy providers.
- Transport service providers.
- Postal service providers.

The law applies to the websites and mobile apps of the service providers listed above. *Strong electronic identification* is a service used to identify a user for digital services in Finland; banks and telecom providers offer this service, and strong electronic identification also falls in the scope of the law. [Act 306/2019]

European accessibility act [European commission 2019] further complements the web accessibility directive [European commission 2016] by expanding the scope, including computers and operating systems, smartphones, ATMs and ticketing machines, transport services, etc. And most notably, from the perspective of web accessibility, e-

commerce services, and online communication services, public and private, as this new directive applies equally to public and private sectors.

Act on the provision of digital services is not the only law guiding or obligating accessibility in Finland. For example, the constitution of Finland guarantees equality for all people [Aluehallintovirasto 2023a; Act 731/1999].

Laamanen et al. [2022] studied the landing pages of Finnish higher education institutes before and after the act on the provision of digital services [Act 306/2019] became binding. In more than half of the higher education institutions, the landing pages' number of accessibility errors decreased, but this also means that some of the landing pages' number of accessibility errors increased [Laamanen et al. 2022]. Over the transition period, the average number of errors on landing pages decreased from 145 (average errors/page) before to 119 (average errors/page) after [Laamanen et al. 2022].

### 3.4 Web content accessibility guidelines (WCAG)

World Wide Web Consortium (W3C) is an international community that develops standards for the Web to ensure the long-term growth of the Web. To help to make the Web more accessible, W3C launched a group Web Accessibility Initiative (WAI), which developed widely used Web Content accessibility guidelines (WCAG); which latest version is 2.1, which was released in 2018, while the draft for version 2.2 is currently scheduled to be published in early 2023. WCAG 2.1 extends the older 2.0 version, and content that conforms to the newer version 2.1 also conforms to the 2.0 version. Thus WCAG 2.x versions are backward compatible. [W3C 2018].

Web accessibility guidelines, checklists, and standards such as Web Content Accessibility Guidelines (WCAG) are used to evaluate accessibility and are also used in some countries' legislation. For example, in European Union uses WCAG 2.1 conformance levels A and AA as standards for web accessibility. WCAG 2.0 is also ISO (International Organization for Standardization) standard ISO/IEC 40500.

WCAG 2.1 sets up a list of how to make web content more accessible to people with disabilities. Following WCAG 2.1, web content will fulfil at least the minimum requirements for accessibility because it cannot cover all possible needs of people. The needs of people differ from person to person, and addressing the needs of everyone is almost impossible. WCAG 2.1 [W3C 2018] acknowledges that completing all success criteria does not make the website accessible to everyone, especially in cognitive language and learning areas. The WCAG may also improve the usability of the website in general. [W3C 2018]

Figure 1 shows the structure of WCAG 2.1. At the top level, WCAG 2.1 is divided into four principles that make the Web accessible. Under each principle, there is a list of guidelines that set basic goals that the authors should follow to make the content accessible. In WCAG 2.1, there is a total of 13 guidelines. These guidelines are not testable, but

each guideline has a set of testable success criteria. WCAG 2.1 has 78 success criteria (30 level A, 20 level AA, and 28 level AAA). Each success criterion belongs to one of three conformance levels A (lowest), AA, and AAA (highest). To meet a certain conformance level of WCAG 2.1 website need to satisfy all success criteria of that level and all levels below it. That means to meet conformance level AA. The site must satisfy all success criteria of levels AA and A. For each of the guidelines and success criteria in the WCAG 2.0, the document provides techniques that are either sufficient to meet the success criteria or advisory that goes beyond what is needed to pass the success criteria. Advisory techniques may address accessibility issues that are not covered by any of the success criteria. [W3C 2018]

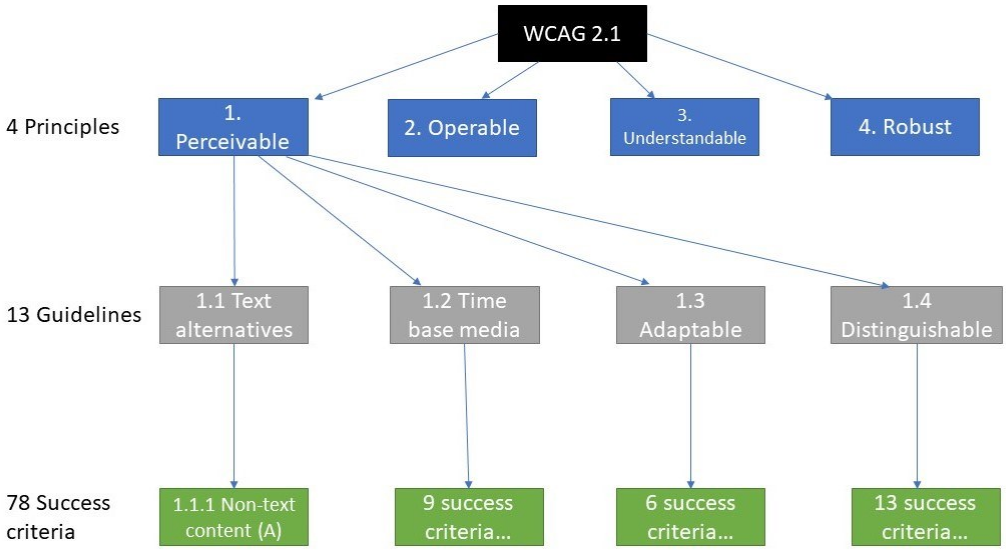


**Figure 1**. Structure of the WCAG 2.1.

The four principles of WCAG 2.1 are: *perceivable, operable, understandable, and robust*. Under the Perceivable principle, there is a total of four guidelines and 29 success criteria. Perceivable means that the content and user interface components must be presented in a way the users perceive them. This can be achieved by providing text alternatives to non-text content (Guideline 1.1), that is, alternative text to images (Success Criterion 1.1.1). And providing alternatives for time-based media (Guideline 1.2), for example, transcript to video, audio (Success Criterion 1.2.1), or mixed media content (Success Criterion 1.2.2). The content should be adaptable so it can be presented in different ways

(Guideline 1.3), for example, Success Criterion 1.3.4 requires that the content is not restricted to a single screen orientation and the layout can be changed without losing structure or content. [W3C 2018]

The operable principle includes a total of five guidelines and 29 success criteria. Operable means that all user interface components and navigation must be reachable and usable. They can't require interactions that users cannot perform. All functionalities should be available using the keyboard alone (Success Criterion 2.1.1), and users should be given enough time to use and read the content (Guideline 2.2). Content should also be functional via other input methods beyond the keyboard (Guideline 2.5). The site should be designed in a way that doesn't cause seizures and physical reactions (Guideline 2.3). For example, Success Criterion 2.3.1 requires that the content doesn’t include anything that flashes more than three times in any thee second period. [W3C 2018]

The understandable principle contains a total of three guidelines and 17 success criteria. Understandable means that the content on the site should be comprehensible for users from different backgrounds, education, and language skills. For example, Guideline 3.1 states, that the text should be readable and understandable. Guideline 3.2 requires that the webpages should work predictably, and the actions performed by users should work expectedly. Guideline 3.3 says, the site should provide users with help input information to mitigate possible errors. For example, Success Criterion 3.3.1 requires that, if errors are detected, then provide suggestions on how to fix them. [W3C 2018]

The robust principle includes one guideline and three success criteria. Robust means that the web pages should be robust enough to work on various user agents. A user agent is a software program accessing the information on the Web, for example, web browsers, bots scraping the Web, or other assistive technologies. For example, Guideline 4.1 says, the Web should work on all current and future user agents. This can be achieved for example, by following the Success Criterion 4.1.1, which requires the correct use of markup languages, elements have complete start and end tags, no duplicate attributes, and all IDs are unique. [W3C 2018]

One of the goals of WCAG 2.0 was to be more testable than its predecessor. For each guideline, a testable success criterion is provided.

When designers and developers focus on accessibility through WCAG checklists and automatic tests, the focus is solely on the technological implementation of accessibility, and the usability aspect of the accessibility may be neglected. Combining the accessibility standards and usability processes with real people ensure that the website is technologically accessible and usable. This is referred to as *usable accessibility* or *accessible user experience*. [WAI 2023b]

Roemen and Svanaes [2012] empirically validated the performance of WCAG 1.0 and 2.0 compared to the usability test of two websites with disabled users. They found

that only 27% of accessibility issues encountered by users during the study were covered by WCAG 1.0, and the rate for WCAG 2.0 was 32%. While the combination of both WCAG 1.0 and 2.0 covered 38% of accessibility issues. The study also found no correlation between the severity of accessibility issues encountered and the conformance level of WCAG. Using WCAG alone does not guarantee that the website is accessible [Roemen and Svanaes 2012].

# 4. Evaluating web accessibility

Web accessibility evaluation is a process that evaluates how well users with disabilities can use the Web. This process aims to find accessibility problems and possibly assess the level of accessibility [Brajnik et al. 2011]. Accessibility evaluation is an important part of assuring that the website is usable for disabled people. Ideally, the site is evaluated multiple times during the design and development, but this is not always possible. Some old sites may need to be retrofitted to conform to accessibility requirements. There are a few things to consider in the context of web accessibility evaluation, such as what is web content and how does it affect the evaluation? What are the requirements for accessibility, and what parties are involved in web accessibility evaluation? What are different accessibility evaluation methods available for web development? In this chapter, we aim to answer these questions.

When evaluating web accessibility, we need to evaluate the web content. Web content can be seen as having two parts, technical content and natural information content. The technical content is the technical implementation of the website; it determines how the information is displayed and how the site functions, common technologies are HTML, CSS, and JavaScript. Technical persons produce technical content during the development of the site. Natural content is information displayed on the Web, including text, images, audio, video, and multimedia.

Some accessibility requirements for technical content are easy to evaluate automatically with software. For example, WCAG [W3C 2018] success criterion 1.4.3 for minimum contrast sets minimum requirements for contrast between foreground text and background. It is easy to check the HTML and CSS for contrast because it is easy to know algorithmically what foreground and background are. But the same rule is not trivial to evaluate when evaluating text in images. It is hard to differentiate what is foreground text and what is background in image data, and often human input is needed. Evaluating natural information content for accessibility is equally important as technical content, even though it is often neglected [Aboy-Zahra 2008].

Evaluating technical content bears similarities to software quality assurance, where the correct behavior of software is confirmed with specific test cases in a controlled environment. Still, there are some key differences [Aboy-Zahra 2008]. Web content tends to change frequently, while the software is often released in discrete versions that don't change much over time [Aboy-Zahra 2008]. A large part of web content is published by non-technical authors, with content management systems (for example, WordPress), wikis, blogging sites, social media, etc. Thus quality control of such content is difficult [Aboy-Zahra 2008].

Evaluating the accessibility of navigation and other user interface elements and natural information content more often reassembles software usability evaluation (also known

as usable accessibility), and evaluating technical content bears some similarities to the software quality assurance methods (also known as technical accessibility), web accessibility evaluation lies somewhere between them [Aboy-Zahra 2008; Bai et al. 2017]. Web accessibility evaluation has evolved into its own field with tools, guidelines, and standards [Aboy-Zahra 2008].

There are different stages of the web development life cycle: requirements, design, implementation, and operations. And different roles working on web development projects include designers, developers, and testers. Different roles can use different evaluation methods at different stages of development. For example, automatic testing tools need a working prototype that can be tested; thus, automatic tools are not always usable during the early design phase of development. Different roles may prefer different testing methods; teams should not choose a single method for all members [Bai et al. 2019].

Correct evaluation method should be based on the reasons and goals of evaluation, it does not make sense to do user testing if the goal of the evaluation is to measure the conformance to a guideline, and it is not optimal to evaluate WCAG walk-through when measuring the difficulties disabled users may face when using the website [Brajnik 2006]. There is no one true method of evaluating accessibility, and often a combination of evaluation methods leads to the best result.

Evaluation methods should be valid, reliable, useful, and efficient [Brajnik 2006]. *Validity* measures the relevance of the discovered issues, that is, how relevant is the issue to real-world use of the system and how likely they show up in real use of the system [Brajnik 2006]. *Reliability* means how reliably the independent evaluators produce the same results [Brajnik 2006]. *Usefulness* means to what extent the results produced to whom may use the results to assess, fix, or otherwise assess the accessibility of a website [Brajnik 2006]. *Efficiency* measures the cost of the method's resources (time, persons, skill level, etc.) [Brajnik 2006]. The evaluation method should also be user-friendly because practitioners are more motivated to keep using it if it is user-friendly [Bai et al. 2019].

Bai et al. [2019] surveyed what accessibility evaluation methods software teams prefer. They investigated six methods: Cambridge Glasses, personas, WCAG walk-through, screen reader, Dyslexia simulator, and automatic tool SiteImprove. Their study included 57 members from software development teams from different roles, such as developers, designers, managers, and testers. The study participants rated the tools for usefulness, ease of use, ease of learning, and satisfaction. Bai et al. [2019] found that almost all methods scored fairly high, but the WCAG walk-through scored significantly lower than the rest. [Bai et al. 2019]

Evaluation of accessibility testing tools can be evaluated in at least two ways: using a test suite or selecting a representative sample of websites [Vigo et al. 2013]. Test suites

are a set of tests where each success criterion is tested in a way that checks if the automated tool catches the intentionally made error. This is called *true positive*, and if the tool fails to detect the error, that will result in *a false negative*. And on the test that conforms to the success criteria should not find any errors. The result is called *a true negative*. And if the tool reports errors when it should not, those cases are called *false positives*. [Vigo et al. 2013]

Test suites are suited for testing tool validity and reliability. Still, the limitation of the test suites is that they test is often isolated pieces of code that just test each success criterion one by one, and the tests are highly specific. Therefore, these tests don't necessarily represent the real issues found in real websites, which may be combinations and variations of different issues and may appear in unexpected ways. [Vigo et al. 2013]

Testing the tool performance on real websites can be beneficial because the accessibility issues on the real websites may appear in unexpected ways, in ways the test suites cannot predict. In an ideal case, the selected sites cover a wide range of success criteria, and the selected pages of the websites should be representative enough of the website and the overall use cases. [Vigo et al. 2013]

### 4.1 Automated testing

Automatic testing is carried out without human intervention, it is a cost-effective way to evaluate many sites and can be done periodically. Automatic evaluation tools are software programs or online services that can help evaluate the accessibility of web content [WAI 2023c]. These automatic accessibility evaluation tools can process a large number of sites fast compared to manual human evaluation. Automatic tools are available to everyone and do not need any special skill; hence the evaluator doesn't have to be an expert to use them [Laamanen et al. 2022].

There are a lot of automatic tools to choose from. W3C [2023c] lists 139 automatic tools for evaluating against WCAG 2.0 guidelines and 85 tools for WCAG 2.1 guidelines. The list can filter the tools by language, type of tools, supported formats, assists by, scope of what the tool checks, and license. Available tools include application programming interface (API), authoring tools and browser plugins, command line tools, desktop tools, and online services.

While the automatic testing tools often check the content for the guideline success criteria, it is important to understand that the tools do not provide complete coverage of the guidelines. The WCAG provides minimum requirements for web accessibility [Aluehallintovirasto 2023b] and cannot cover all needs of all disabled people. With that in mind, the automatic tools can check only a subset of WCAG requirements [Aboy-Zahra 2008]. For example, WCAG guideline 1.1 states that all non-text content needs a text alternative [W3C 2018]. While an automatic tool can easily check the presence of an

alternative text, it is difficult to assess if the text alternative accurately conveys the same information as the non-text content.

Errors reported by automatic testing tools can greatly differ when testing the same page [Ismailova and Inal 2022]. Some tools may report the same error multiple times, thus inflating the number of errors [Padure and Pribeanu 2020]. Previous studies on automatic testing tools find that it is recommended to use multiple automatic tools to increase confidence in results [Padure and Pribeanu 2020; Ismailova and Inal 2022; Frazão and Duarte 2020].

Vigo et al. [2013] tested six state-of-the-art automatic testing tools on the tool *coverage*, *completeness*, and *correctness*. Coverage measures how many success criteria the tool covers out of all the success criteria. They found that all tools covered less than half of the success criteria, and the tool performing worst only covered 1 out of 4 success criteria [Vigo et al. 2013]. Completeness measures the ratio of violations reported by the tool and the actual number of violations found by expert evaluators. The best-performing tool got a score of 38%. Correctness measures do the tool report incorrect results, and the study found that the tools evaluated produced a high level of correctness [Vigo et al. 2013]. That means if a site passes all automatic checks, it does not mean it fully satisfies all WCAG requirements or is usable for people with disabilities.

Automatic testing could be differentiated into three categories: syntactic, heuristic, and indicative checks [Aboy-Zahra 2008]. Syntactic checks analyze the syntactic structure of the web page. For example, an automatic testing tool can check that the image elements have an alt attribute [Aboy-Zahra 2008]. The alt-attribute is used to describe the image content so that assistive technologies, for example, a screen reader, can describe the image content. While the automatic tool can check that the alt-attribute exists, it cannot check if it correctly describes the content. The automatic tools are otherwise accurate in checking syntactic errors [Aboy-Zahra 2008].

Heuristic checks analyze some semantics of the web content, for example, layout and markup or the natural language information [Aboy-Zahra 2008]. These checks cover a wider range of accessibility requirements, but they do it at the cost of accuracy; the tools often only give warnings of these errors so the human evaluator can manually go over these warnings [Aboy-Zahra 2008].

Indicative checks use statistical metrics and profiling techniques to estimate the state of accessibility of the whole website or a large collection of web content [Aboy-Zahra 2008]. These checks are too imprecise to evaluate the accessibility. Still, these methods can be used to monitor the state of accessibility of a website over time or monitor the overall development of accessibility in a specific sector [Aboy-Zahra 2008].

### 4.2 Manual testing

Manual testing is a process where expert human evaluators do the evaluation process. Some of the manual evaluations include guideline conformance, simulation kits, assistive technologies, and personas [Bai et al. 2017]. Guideline conformance review is one of the most widely used methods for evaluating accessibility [Bracnik 2008].

*Guideline conformance* is a method where an expert evaluator checks a guideline success criterion and decides if the evaluated page passes the success criteria [Brajnik 2008]. WCAG is used in some country's legislation as a standard for web accessibility and is the most widely used standard. Other standards and guidelines also exist. The benefits of WCAG walk-throughs or other guideline conformance reviews are that this method usually covers a wide range of accessibility issues and is relatively cost-effective, especially when combined with automatic testing tools. The guideline conformance method can identify reasons why a success criterion fails [Brajnik 2008].

Website accessibility conformance evaluation methodology by W3C [2014] comprises 5 steps:

1. Define the evaluation scope
2. Explore the target website
3. Select representative sample
4. Audit the selected sample
5. Report the findings.

These steps are not necessarily done in order, and the sequence depends on the website, the evaluation's purpose, and the evaluator's process [W3C 2014]. In the process, the evaluator moves from one step to the next. Also, the evaluator may return to any previous step as new information about the evaluation is discovered during the evaluation process [W3C 2014].

*Simulation kits* or *screening techniques* include equipment and software that simulate the effects of disabilities or techniques where some sensory, motor or cognitive capability is artificially reduced [Bai et al. 2017; Brajnik et al. 2011]. For example, Cambridge Simulation Glasses [University of Cambridge 2023] simulate the effects of vision loss. These glasses can be used to test the website and provide the evaluator insight into what it might feel like to use a website with a loss of vision. And to help design a more accessible website, the glasses might help to find issues with contrast and text sizes [Bai et al. 2017].

Evaluation with assistive technology, the expert evaluates the site in a way disabled users might use the site. For example, if blind or visually impaired users cannot see the information on the page, they can use screen readers to browse the content [Bai et al. 2017].

Personas are another way to evaluate accessibility. Personas are fictional people that can be used in the specification (requirements) development process to better understand

the users' needs but can also be used in development and testing [Bai et al. 2017]. Personas represent the target users, and different personas can represent different user groups.

*Barrier walk-through* is a technique where the context of website usage is explicitly considered [Brajnik 2008]. In a barrier walk-through, an evaluator accesses predefined barriers that are interpretations and extensions of well-known accessibility principles [Brajnik 2008]. *Accessibility barriers* are conditions that make using the website difficult for disabled users [Brajnik 2008]. The context for barrier walk-through comprises user categories such as blind person, hearing impairment, cognitive impairment, etc. [Brajnik 2008]. Usage scenarios mean the way of using the website, for example, using the website with a screen reader or using a mobile device, and user goals are use cases of why the user is using the website [Brajnik 2008].

The *evaluator effect* is the "observation that usability evaluators report substantially different sets of usability problems when applying the same evaluation technique on the same application" [Hornbæk and Frøkjær 2008]. The evaluator effect has been studied in usability evaluation methods [Hornbæk and Frøkjær 2008]. Brajnik et al. [2011] studied the evaluator effect in the barrier walk-through method. They found that a single expert evaluator found about 72% of accessibility barriers, two experts found 94% of problems on average, and three experts found all the accessibility problems [Brajnik et al. 2011]. One non-expert evaluator found about 50% of the true problems, and three non-experts found about 80% of true problems. While using the walk-through barrier method, 14 non-experts were needed to find all problems [Brajnik et al. 2011].

### 4.3 User testing

User testing is an evaluation method done with real end users. Users then are asked to perform goal-free or goal-orientated tasks, and their performance is observed by the evaluator [Brajnik et al. 2011]. The benefit of user testing as an accessibility evaluation method is that it identifies real usability problems experienced by real users [Brajnik 2008].

Drawbacks of user testing include relatively low-cost efficiency, inability to highlight the defects. These problems may be missed if predefined scenarios are not well chosen, it will not identify low-severity problems [Brajnik 2008]. It is also complicated to set up testing scenarios with disabled participants, especially when assistive technologies are needed [Brajnik 2008]. User testing methods might result in usability problems rather than accessibility problems affecting disabled users. While still relevant, it's not what the goal of the evaluation was [Brajnik 2008]. The experience of the user also affects the results. A user who is experienced in using technology (such as a screen reader) might not find accessibility problems affecting novice users [Brajnik et al. 2011].

The crowdsourcing method is another possible way to evaluate accessibility [Song et al. 2018]. In the crowdsourcing method, work is divided into small tasks that can be divided by workers to solve them in parallel remotely [Song et al. 2018]. Song et al. [2018] conclude that the crowdsourcing method with non-expert evaluators yielded a reliable way to assess accessibility with their novel truth algorithm.

# 5. Comparison of accessibility testing tools

In this chapter, we cover the method used to compare three automatic accessibility testing tools. First, we present the research questions and research method. Then, we go over the method used to select the tools and introduce the selected testing tools. Next, we discuss the reasons for selecting websites, how the sites were sampled, and how the pages were tested. Lastly, we go over the metrics used for the comparison.

## 5.1 Research questions and research method

In this thesis, we compared three different automatic accessibility testing tools. The thesis aims to answer the following research questions:

- RQ 1 What success criteria do automatic accessibility testing tools test?
- RQ 2 Is there a difference between selected tool features?
- RQ 3 Do the tools detect different issues?

To answer RQ 1 and RQ 2, we investigated the tool documentation if the tools are transparent about what success criteria they test. Success criteria coverage was collected from the available tool documentation. Coverage means that the tool maps at least one issue to the success criteria. Coverage does not mean that all possible issues of the success criteria are detected. And one issue can also be mapped to multiple success criteria.

To answer RQ 3, tools were tested on three different websites, and we were interested in only automatically detected issues, which means that warnings and potential accessibility issues that needed a human review were discarded, also recommendations or best practices which are issues that do not correspond to WCAG 2.1 success criteria were discarded. The study assumes that all the accessibility issues reported by the tools were real issues.

## 5.2 Selecting tools

Automatic accessibility evaluation tools were selected from the WAIs [2023c] web accessibility evaluation tools list page. Web accessibility evaluation tools list [WAI 2023c] provides a list of evaluation tools. Currently, the page lists 167 tools. The vendors and others provide information about the tools on the page. W3C does not endorse specific tools listed on the page. The page can assist in selecting evaluation tools by allowing users to filter according to a wanted tool feature. [WAI 2023c]

When we selected the tools, firstly, the list was filtered by selecting the tools that check WCAG 2.1 guidelines. Then the type of the tool was set to browser plugins, and supported formats were set to CSS, HTML, and images. The list of tools was further filtered down by selecting tools that generate evaluation results reports, and the license was set to free software. These filters resulted in a list of five tools. From the result list, three tools were selected for this study. These tools are IBM Equal access accessibility checker [IBM 2023a], LERA [AdvancedBytez 2023], and WAVE [WebAIM 2023a].

### 5.2.1 Google Chrome

Google Chrome is a web browser developed by Google. Chrome is based on the Chromium open-source project. Google Chrome was selected because it is currently the most popular web browser, and most of the browser plugin accessibility evaluation tools were available for Google Chrome.

All the accessibility testing tools are Chrome extensions. Accessibility testing extensions are installed from the Chrome web store. Browser extension accessibility testing tools are easy to install and relatively easy to use because the use of the tools does not necessarily require technical knowledge, or the user doesn't need to be an accessibility expert to use the. Even though the accessibility knowledge makes using the tools more effective.

Browser extensions also provide some benefits over online services. Browser extensions run the accessibility tests on the browser, so it is possible to test locally hosted sites and password-protected content.

In this study we are using Google Chrome Version 111.0.5563.65 (Official Build) (64-bit) [Google 2023].

### 5.2.2 IBM Equal Access Accessibility Checker

IBM Equal Access Accessibility Checker is an open-source automatic accessibility checking tool for Chrome and Firefox browsers developed by IBM Accessibility [IBM 2023a]. The tool utilizes IBM's accessibility rule engine, which detects accessibility issues for web pages and web applications and creates exportable accessibility reports. The extensions are integrated into the web browser's development tools. [IBM 2023a]

Settings of the IBM accessibility checker allow selecting a rule set by deployment date. The latest ruleset version 7.2, was released 03/30/2021, including WCAG 2.1 ruleset, among other guidelines. It is also possible to select the guideline between IBM Accessibility, WCAG 2.0 (A, AA), and WCAG 2.1 (A, AA). This study uses the Google Chrome extension version 3.1.46.9999, released in the Chrome web store on 11/03/2023.

The IBM accessibility checker reports accessibility errors in violations, needs review, and recommendations categories. Figure 2 shows a result of an accessibility scan. Violations are errors detected by the tool automatically, needs review are possible violations that need manual review, and recommendations are opportunities to apply best practices. In this study, we are only interested the errors detected by the tool. Thus, we are only looking into the issues reported as violations. The tool can report the issues sorted differently in different tabs, including requirements, element roles, and rules. The issues are mapped to the most relevant IBM requirement in the requirements view, corresponding to the WCAG 2.1 guidelines. In the element role's view, the issues are organized by the roles of the DOM (document object model) elements. And in the rules view the issues are sorted by the rules in the rule set.

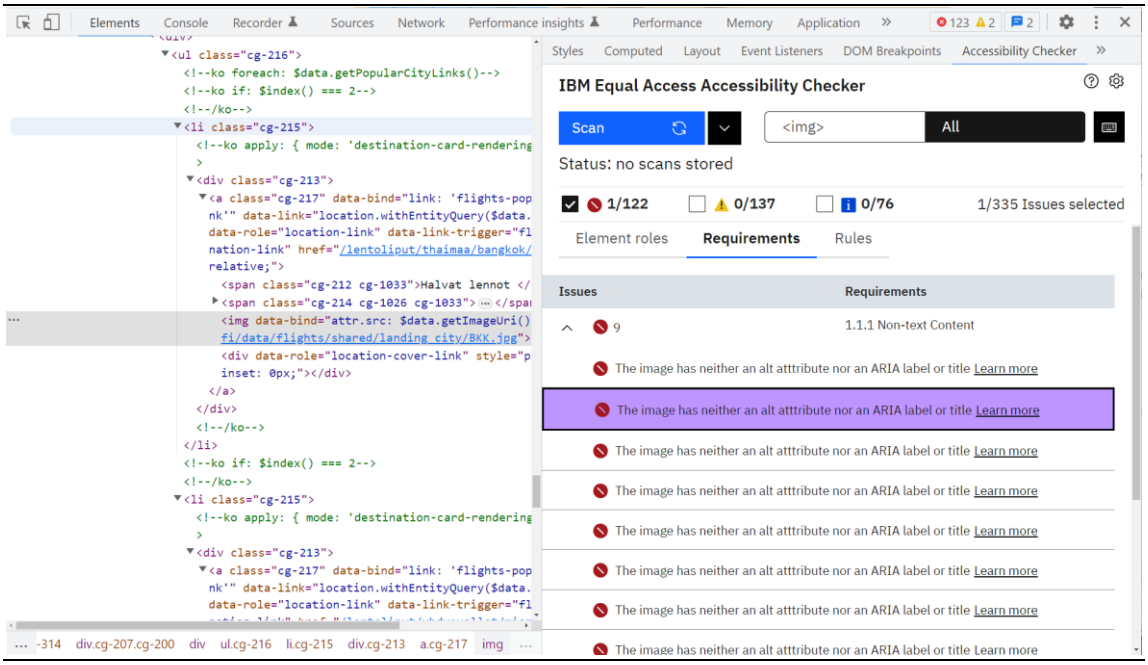


**Figure 2.** Accessibility test result using IBM Accessibility Checker [IBM 2023a].

### 5.2.3 LERA

LERA is a free automatic web-based accessibility testing and reporting tool developed by AdvancedBytez [AdvancedBytez 2023]. LERA utilizes the axe-core library, which is an open-source accessibility testing engine for websites and other HTML-based user interfaces [Deque Labs 2023a]. According to the axe-core GitHub page, on average, axe-core can find 57% of WCAG issues automatically [Deque Labs 2023a]. In this thesis, we are using version 0.5.2 of LERA, installed from the Chrome Web Store.

LERA scan audits one page at a time against the WCAG 2.1 guideline levels A and AA success criteria. LERA only reports automatically detected accessibility errors, therefore all the errors reported by the tool are used in this study. Errors are categorized and can be filtered by impact and success criteria. The severity impact categories include critical, serious, moderate, and minor. LERA documentation did not specify what these categories mean, but these categories are defined in axe-core documentation as [Deque Labs 2023b]:

- *Minor*: considered to be a nuisance or an annoying bug
- *Moderate:* results in some difficulty for people with disabilities, but will generally not prevent them from accessing fundamental features or content.
- *Serious:* results in serious barriers for people with disabilities, and will partially or fully prevent them from accessing fundamental features or content.
- *Critical:* Results in blocked content for people with disabilities, and will definitely prevent them from accessing fundamental features or content.

Using LERA pages can be scanned by clicking the LERA extension, and then clicking the automatic audit button. LERA then opens the dashboard in a new browser window.

Figure 3 shows the LERA dashboard. The dashboard shows the number of issues found on the page and the distribution of issues by severity. The automated issues tab shows the issues in a list. After clicking an issue, LERA shows details of the issue, including code snippet, impact, issue tags that show references to guidelines, and recommendations on how to fix the issue. Clicking the eye icon highlights the issue location on the page.

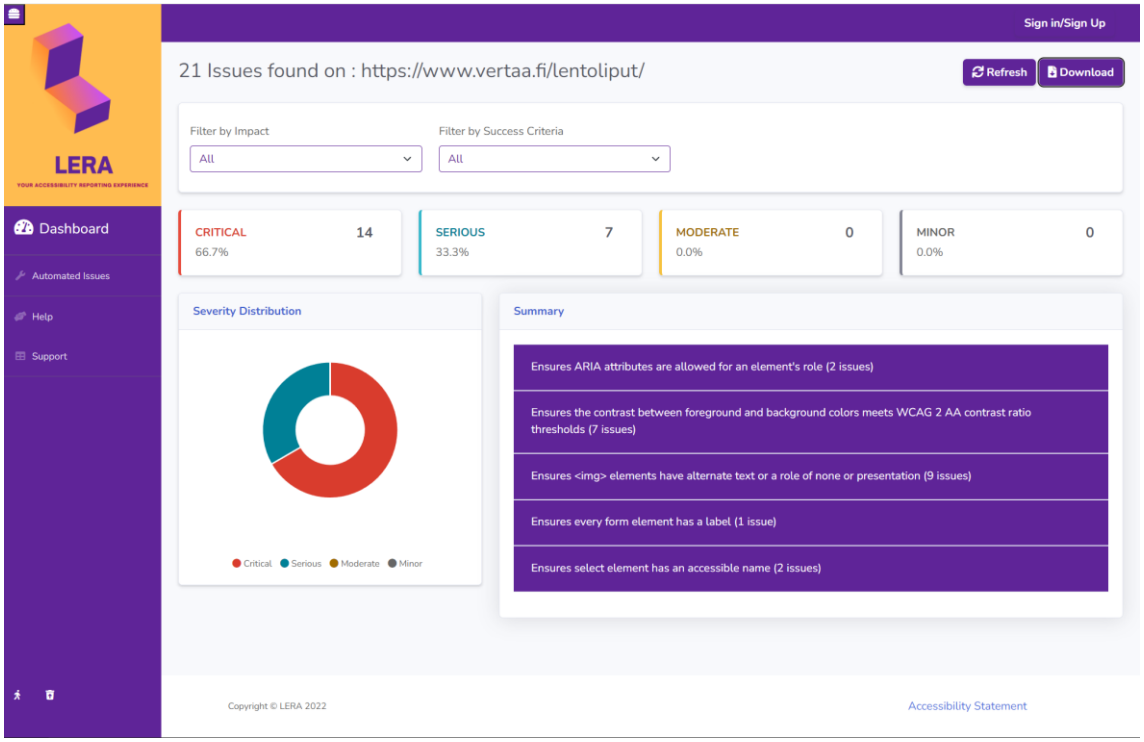


**Figure 3.** LERA dashboard [AdvancedBytez 2023].

### 5.2.4 WAVE

WAVE automatic web accessibility evaluation tool is available for free as an online service and browser extension [WebAIM 2023a]. WAVE is developed by WebAIM, which is a non-profit organization. WAVE focuses on issues that impact end users, facilitate human evaluation, and educate about web accessibility.

WAVE browser extension scans a page by going to the page and clicking the browser extension icon or right-clicking the page and then selecting 'WAVE this page' from the dropdown menu.

WAVE presents the page with embedded icons and indications. These icons and indicators present some information about the accessibility of the page. Figure 4 shows an example scan result with a missing text alternative issue selected. The WAVE side panel provides a summary of the scan results. Accessibility issues are reported as errors and alerts. WAVE also reports features, structure elements, and ARIA labels detected, that way evaluator can manually review that the features are implemented correctly. Errors

are separated into two categories errors and contrast errors. Alerts are possible accessibility issues that need manual evaluation. In this thesis, we are only interested in the automatically found issues that are reported as errors and contras errors.

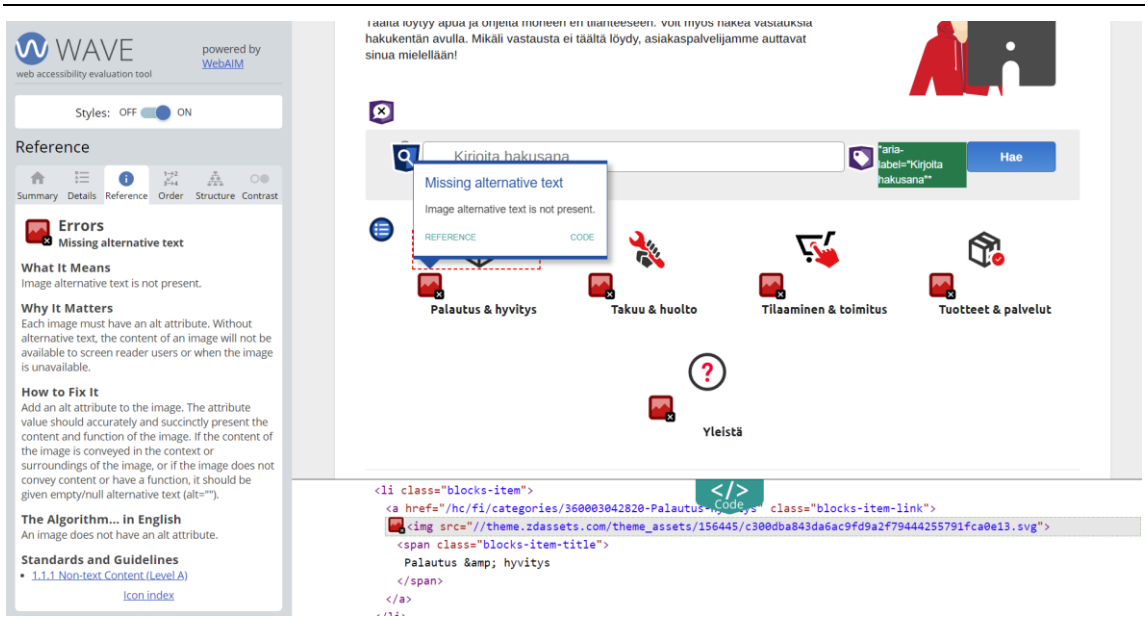


**Figure 4.** WAVE detecting an image with a missing alternative text [WebAIM 2023a].

The details tab lists accessibility issues in the side panel. The user can then click icons in the details view to highlight the issue on the page. Clicking the issue icon on the page shows the details of the issue and links to the guideline reference and to the source code.

## 5.3 Selecting websites

We are interested in the performance of the tools. So, we are going to use a combination of real sites and a test site. We are mainly interested in the automatic testing tool performance, so we try to select sites that cover a wide range of content and possible issues such as images, text, videos, input forms, etc. e-commerce sites were selected for that reason. The state of the accessibility of the e-commerce sites is also interesting because the e-commerce sites don't have specific legal accessibility requirements yet.

The test site has several intentionally made accessibility issues, and it was selected to see if the tools detect the issues on the site. While the site does not have a test for all possible accessibility issues, it does have a list of issues we can use to check if the tools detect them.

### 5.3.1 Vertaa.fi

Vertaa.fi is the most popular price comparison site in Finland [Vertaa 2023]. Vertaa.fi offers a way to compare prices of products and services but doesn't sell any products themselves. Instead, they collect the price information from 243 stores and allow users to search for products and find a store that sells the product for the lowest price.

### 5.3.2 Verkkokauppa.com

According to Verkkokauppa.com [2023], Verkkokauppa.com is the most visited and well-known retail e-commerce store in Finland. They sell computers, electronics, toys, games, sports products, etc. Verkkokauppa.com was founded in 1992, and in addition to their e-commerce store, they have 4 retail stores.

### 5.3.3 Test suite

Government digital services [2023] developed an open-source test suite to test automatic accessibility testing tools. The test site has 142 accessibility issues, in this study we use 140 tests, because the embedded YouTube video for the flashing content test is no longer available, and the audio file is missing for the embedded audio file is missing text alternative test.

## 5.4 Sampling pages

Pages were selected to increase the number of different elements, and so the increase the number of different possible accessibility issues. Instead of just scanning the landing pages of multiple different sites, we decided to scan the landing page and a couple of other pages per site. Because we assume that the landing page might have the least errors on the site.

The landing page is the first site most users see when they visit the site and use to navigate to the different parts of the site. We assume that it makes the developers most sense to focus the accessibility efforts on the landing page first. Thus, the landing page might have the least number of accessibility issues.

Table 1 shows the selected pages and the dates when the page was tested. On Verkkokauppa.com we tested the landing page, account creation page, and customer service page. On the Vertaa.fi site we tested the landing page and from the popular categories, we selected the first category which was cheap flights (halvat lennot).

On the test suite, most of the tests were on the main page, but some of the tests were linked to different pages. We tested the test page and any additional linked test pages.

| **Page** | **Date when the site was tested** |
|---|---|
| **Verkkokauppa.com** | 6.4.2023 |
| **Create account page. (verkkokauppa)** | 6.4.2023 |
| **Customer service page. (verkkokauppa)** | 6.4.2023 |
| **Vertaa.fi** | 6.4.2023 |
| **Cheap flights (vertaa)** | 6.4.2023 |
| **Test suite** | 6.4.2023 |

**Table 1**. Selected pages and dates when the pages were tested.

### 5.5 How the pages were tested

We tested each page with each tool one by one. First, we waited for the page to fully load, then we scrolled to the bottom of the page, to make sure that the page was fully loaded. Then we tested the same page in the same browser window, with each tool one by one. That way we hope to minimize the probability of dynamic content changing between the test of different tools.

Lastly, we repeated the steps one more time, to see if the accessibility testing tools produced consistent results, and to collect the data of scan speed of the tools. All the tools reported identical results on both scans on every page we tested.

Tools report the detected accessibility issues in different ways. Each tool checks accessibility rules, a rule may correspond to one or more WCAG 2.1 success criteria. We were interested to see how well the automatic testing tools can check the conformance to WCAG. For that reason, we collected the total number of accessibility issues and the number of issues in each WCAG success criterion. In some cases, the number of accessibility issues may be less than the sum of WCAG violations, because an issue detected by the tool may be mapped to multiple success criteria by the tool.

### 5.6 Comparison metrics

Comparison metrics used to compare the tools are efficiency, completeness, and the number of detected issues. Efficiency means how fast the tool scans the page. Scan time is important because the scan time can quickly add up to the tool use used to scan hundreds of pages.

Completeness means how completely the tool covers the WCAG success criteria. The tool is considered to cover a success criterion if the tool performs at least one test on that success criterion.

The number of detected issues is the number of automatically detected issues on a page. Issues that needed a human review and recommendations were discarded because we were interested in how well the tools detect issues automatically.

Features use to compare the tools were the features we found useful while testing the sites. List of features is not complete list of the features of a tool. But it is a set of features we found useful while using the tools.

# 6. Results

In this chapter, we go over the results of the comparison of accessibility testing tools. First, we go over the success criteria coverage of the tools. Then we present the analysis of the detected issues by the accessibility testing tools. And lastly, we present the comparison of tool features.

## 6.1 Success criteria coverage

Success criteria coverage is collected from the tool's documentation, to figure out the transparency of the tools. That means we want to know if the tools inform what success criteria they test. Table 2 shows the success criteria covered by the tools according to the documentation [Deque Labs 2023c; IBM 2023b; WebAIM 2023b]. In Table 2, F stands for failure, A for an alert, N stands for needs review, and X means that the tool didn't specify if the tests produce errors or alerts of possible errors.

The selected tools do some test for issues and warnings in 37 success criteria out of 78, which means 47% of the success criteria is covered by the selected three tools. While the success criteria covered, it does not mean that the success criteria are completely tested, but rather that the tool can detect at least one issue mapped to the success criterion. Tools also map some detected issues to multiple success criteria.

IBM Equal Access Accessibility Checker did not mention if the automatic test for specific WCAG success criteria produces issues or alerts of possible errors. They only reported that the success criteria were automatically tested. But when comparing the coverage of issues and alerts, IBM Accessibility Checker covers the most WCAG 2.1 success criteria 31 out of 78, while WAVE and LERA both cover 22 success criteria when considering both automatically detected issues and alerts of possible issues. According to the tool's documentation, LERA covers the most success criteria with a fully automatic test with 20 out of 78 success criteria, while WAVE detects automatic issues for 13 out of 78 success criteria.

The Union of the success criteria covered by the tools shows that the selected tools cover different success criteria. Thus complementing each other. The Union of success criteria covered by issues and alerts is 37, while the single tool with the widest coverage covered 31 success criteria.

| Success Criteria | IBM | WAVE | LERA (axe-core 4.3.5) |
|---|---|---|---|
| **WCAG 1.1.1** | X | F, A | F, N |
| **WCAG 1.2.1** | X | A | N |
| **WCAG 1.2.2** | X | A | N |
| **WCAG 1.2.3** | | A | |
| **WCAG 1.2.4** | X | | |
| **WCAG 1.2.5** | X | A | |
| **WCAG 1.3.1** | X | F, A | F, N |
| **WCAG 1.3.2** | X | A | |
| **WCAG 1.3.3** | X | | |
| **WCAG 1.3.5** | X | | F |
| **WCAG 1.4.1** | X | | F, N |
| **WCAG 1.4.2** | X | A | F, N |
| **WCAG 1.4.3** | X | F | F, N |
| **WCAG 1.4.4** | X | | |
| **WCAG 1.4.12** | | | F |
| **WCAG 2.1.1** | X | F, A | F, N |
| **WCAG 2.1.2** | X | A | |
| **WCAG 2.2.1** | X | F | F |
| **WCAG 2.2.2** | X | F | F |
| **WCAG 2.2.4** | | | F |
| **WCAG 2.4.1** | X | F, A | F, N |
| **WCAG 2.4.2** | X | F | F |
| **WCAG 2.4.3** | | A | |
| **WCAG 2.4.4** | X | F, A | F, N |
| **WCAG 2.4.6** | X | F, A | |
| **WCAG 2.4.7** | X | | |
| **WCAG 2.4.9** | | | N |
| **WCAG 2.5.3** | X | | F |
| **WCAG 3.1.1** | X | F, A | F |
| **WCAG 3.1.2** | X | | F |
| **WCAG 3.2.1** | X | | |
| **WCAG 3.2.2** | X | A | |
| **WCAG 3.2.5** | | | F |
| **WCAG 3.3.1** | X | | |
| **WCAG 3.3.2** | X | F, A | N |
| **WCAG 4.1.1** | X | | F |
| **WCAG 4.1.2** | X | F | F, N |
| **Union of covered success criteria: 37** | 31 | 22 | 22 |
| **Union of Issues: 22** | | 13 | 20 |
| **Union of Warnings: 22** | | 17 | 13 |

**Table 2**. WCAG Success criteria covered by the tools according to the documentation [Deque Labs 2023c; IBM 2023b; WebAIM 2023b].

### 6.2 Detected accessibility issues

In this section, we go over the accessibility issues detected by the tools. We go over the differences between the detected issues, we found that the selected tools find different number of issues reported to different success criteria. One tool may be better at detecting issues on one success criterion than another.

We found that every tool we tested detected accessibility issues on every tested page. The performance of the tools seems to depend on the page and the type of accessibility issues present on the page. One tool may find the greatest number of issues on one page but the least number of issues on another page.

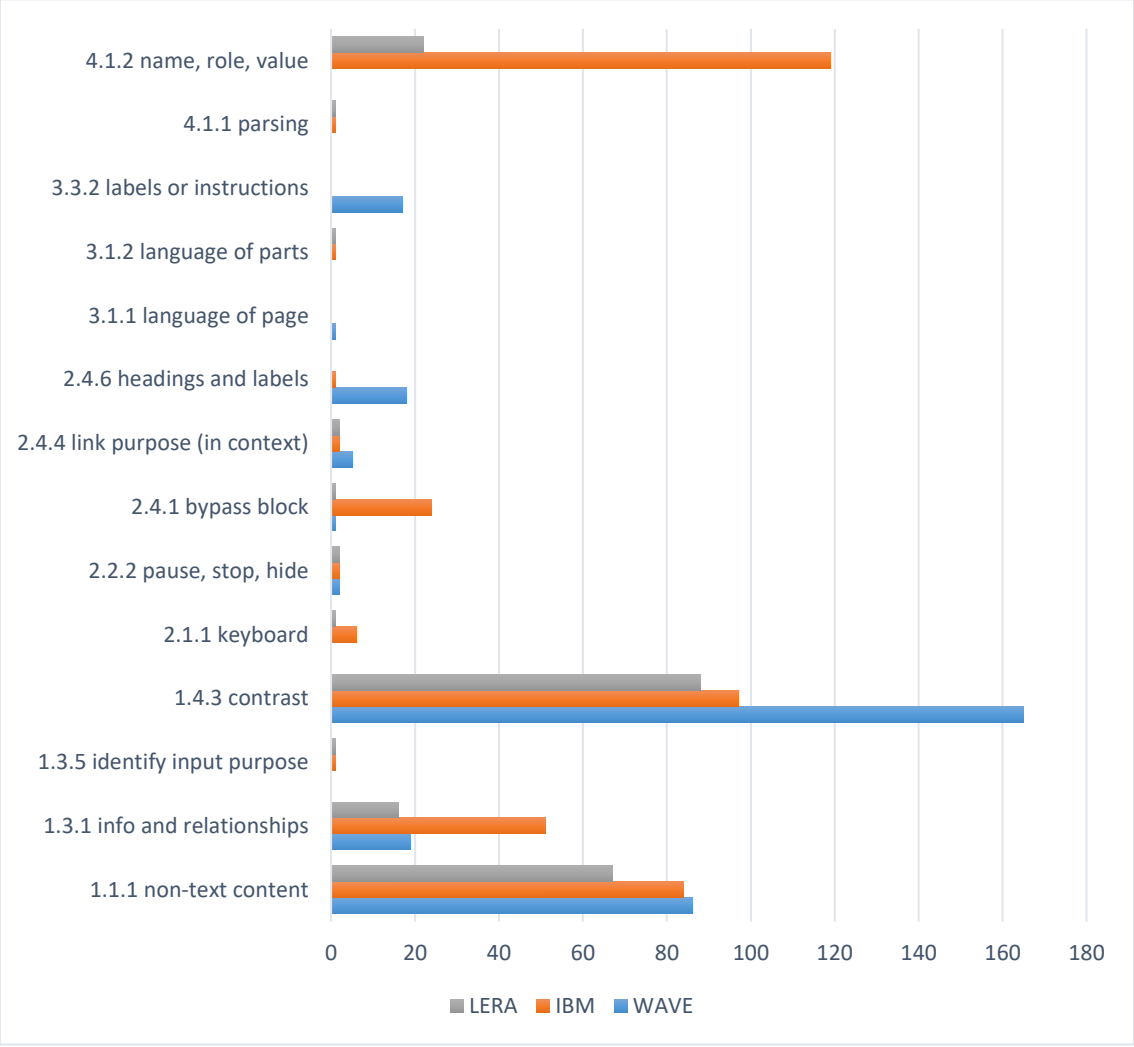


**Figure 5**. The total number of issues for each success criterion on all tested pages.

Figure 5 shows the total number of issues for each success criterion on all tested pages. IBM Accessibility checker reported the greatest number of issues for success criteria 4.1.2 (name, role value), 2.4.1 (bypass block), 2.1.1 (keyboard), and 1.3.1(info and relationships). While WAVE reported the greatest number of issues for success criteria

1.4.3 (contrast), 1.1.1 (non-text content), 2.4.4 (link purpose in context), 2.4.6 (headings and labels), and 3.3.2 (labels or instructions).

### 6.2.1 Verkkokauppa.com

Tables 3, 4, and 5 show the accessibility issues detected on the tested pages on verkkokauppa.com. Every tool found accessibility issues on all these pages. WAVE found a total of 25 issues, IBM 69 issues, and LERA 17 issues.

Table 3 shows the accessibility issues detected on the verkkokauppa.com landing page. IBM Accessibility Checker found the most errors, that is 41 accessibility issues were detected, while WAVE detected 12 issues and LERA 3 issues.

| Success criteria | WAVE | IBM | LERA |
|---|---|---|---|
| **1.1.1 non-text content** | | 14 | |
| **1.4.3 contrast** | 12 | 11 | 1 |
| **2.1.1 keyboard** | | 2 | |
| **2.4.1 bypass blocks** | | 10 | |
| **4.1.2 name, role, value** | | 4 | 2 |
| **total issues** | **12** | **41** | **3** |
| **Scan time** | **< 1 sec** | **21 sec** | **8 sec** |

**Table 3**. Accessibility issues detected on the verkkokauppa.com landing page.

IBM Accessibility Checker also found issues in the greatest number of success criteria, finding issues in 5 different success criteria. While IBM found the most issues on the greatest number of issues on the landing page, every tool found issues in the success criterion 1.4.3 low contrast, WAVE found the greatest number of issues for this success criteria 12, IBM 11, and LERA 1 issue. The left side of Figure 6 shows a low contrast ratio issue present on the landing page (value 3.46) and the same text with a sufficient contrast ratio of 4.61 on the right.


**Figure 6**. Low contrast on left and sufficient contrast on the right [Verkkokauppa 2023].

Accessibility issues found on the account creation page can be seen in Table 4. Again, IBM Accessibility Checker found the greatest number of issues. In detail, IBM Accessibility Checker found 17 issues, and WAVE and LERA both found 3 accessibility issues. On this page, WAVE found the most issues violating success criteria 1.4.3 for low contrast. WAVE also maps the empty form label rule to 4 different WCAG success criteria, success criteria 1.1.1, success criteria 1.3.1, success criteria 2.4.6, and success criteria 3.3.2, while IBM Accessibility Checker maps this issue to success criteria 4.1.2.

LERA and IBM Accessibility Checker found the same number of issues for success criteria 1.3.5 and 1.4.3. Again, WAVE found the greatest number of contrast issues 2, while IBM and LERA found 1.

| Success criteria | WAVE | IBM | LERA |
|---|---|---|---|
| **1.1.1 non-text content** | 1 | | |
| **1.3.1 info and relationships** | 1 | | |
| **1.3.5 identify input purpose** | | 1 | 1 |
| **1.4.3 contrast** | 2 | 1 | 1 |
| **2.1.1 keyboard** | | 1 | |
| **2.4.1 bypass block** | | 10 | |
| **2.4.6 headings and labels** | 1 | | |
| **3.3.2 headings and labels** | 1 | | |
| **4.1.2 name, role, value** | | 4 | 1 |
| **total issues** | **3** | **17** | **3** |
| **Scan time** | **< 1 sec** | **4 sec** | **2 sec** |

**Table 4**. Accessibility issues detected on the verkkokauppa.com account creation page.

Table 5 shows the accessibility issues found on the verkkokauppa.com customer service page. On this page, the tools produced the most similar results. All three tools found the same number of violations for success criteria 1.1.1 non-text content, each tool found 10 issues. In addition of that LERA and IBM Accessibility Checker produced an identical report of errors on the customer service page. Both tools found 11 issues, and both tools mapped the found to the same success criteria.

| Success criteria | WAVE | IBM | LERA |
|---|---|---|---|
| **1.1.1 non-text content** | 10 | 10 | 10 |
| **1.3.1 info and relationships** | | 1 | 1 |
| **4.1.2 name, role, value** | | 1 | 1 |
| **total issues** | **10** | **11** | **11** |
| **Scan time** | **< 1sec** | **< 1sec** | **< 1sec** |

**Table 5**. Accessibility issues detected on the verkkokauppa.com customer service page.

Every tool scanned the customer service page under a second, because the page was the smallest page of the three tested pages.

### 6.2.2 Vertaa.fi

Tables 6 and 7 show the results of pages tested on vertaa.fi. The total number of errors on the tested pages was: WAVE 205, IBM Accessibility Checker 234, and LERA 139 accessibility issues between the two tested pages.

Table 6 shows the accessibility issues detected with each tool on the vertaa.fi landing page. WAVE detected the greatest number of issues on the landing page with a total of 139 issues detected, IBM Accessibility Checker detected 112 issues and LERA 118 issues.

| Success criteria | WAVE | IBM | LERA |
|---|---|---|---|
| **1.1.1 non-text content** | 47 | 44 | 44 |
| **1.3.1 info and relationships** | 3 | | |
| **1.4.3 contrast** | 92 | 67 | 74 |
| **2.4.1 bypass blocks** | | 1 | |
| **2.4.6 headings and labels** | 3 | | |
| **3.3.2 labels or instructions** | 3 | | |
| **total issues** | **139** | **112** | **118** |
| **Scan time** | **< 1sec** | **7 sec** | **6 sec** |

**Table 6**. Accessibility issues detected on the vertaa.fi landing page.

Table 6 also shows that the WAVE detected the greatest number of accessibility violations in the success criteria 1.4.3 contrast. Figure 7 shows some contrast issues detected on the vertaa.fi landing page, only the dark blue text has sufficient contrast. IBM Accessibility Checker was the only tool to detect any violations of success criteria 2.4.1. Every tool found 44 violations for the success criteria 1.1.1, but WAVE also maps missing labels to this category, for that reason WAVE reported more success criteria 1.1.1 violations than the two other tools.


**Figure 7**. Elements with low contrast on the vertaa.fi landing page [Vertaa 2023].

Table 7 shows the accessibility issues detected on the vertaa.fi flight search page. IBM Accessibility Checker found the greatest number of errors on the flight search page, with a total of 122 accessibility issues found, WAVE found 66 issues, and LERA 21. Similarly, to the vertaa.fi landing page, all the tools found the same number of success criteria 1.1.1 violations, but WAVE mapped additionally 3 missing label issues to this category. Hence the larger number of issues for success criteria 1.1.1. WAVE detected the greatest number of success criteria 1.4.3 violations. IBM Accessibility Checker was the only tool to detect issues for success criteria 2.1.1 and 2.4.1. IBM Accessibility checker also detected the greatest number of violations for the success criteria 4.1.2: IBM 97, LERA 5, WAVE 0.

| Success criteria | WAVE | IBM | LERA |
|---|---|---|---|
| **1.1.1 non-text content** | 12 | 9 | 9 |
| **1.3.1 info and relationships** | 3 | | 3 |
| **1.4.3 contrast** | 54 | 13 | 7 |
| **2.1.1 keyboard** | | 2 | |
| **2.4.1 bypass block** | | 1 | |
| **2.4.6 headings and labels** | 3 | | |
| **3.3.2 labels or instructions** | 3 | | |
| **4.1.2 name, role, value** | | 97 | 5 |
| **total issues** | **66** | **122** | **21** |
| **Scan time** | **< 1 sec** | **5 sec** | **3 sec** |

**Table 7**. Accessibility issues detected on the vertaa.fi flight search page.

### 6.2.3 Test suite

Accessibility issues detected on the test are shown in Table 8. IBM Accessibility Checker detected the greatest number of issues in the test suite, most of the issues were in the success criteria 1.3.1 and these issues were about data Table cells missing header or scope, IBM Accessibility checker was only tool that detected these issues.

| Success criteria | WAVE | IBM | LERA |
|---|---|---|---|
| **1.1.1 non-text content** | 16 | 7 | 4 |
| **1.3.1 info and relationships** | 12 | 50 | 12 |
| **1.4.3 contrast** | 5 | 5 | 5 |
| **2.1.1 keyboard** | | 1 | 1 |
| **2.2.2 pause, stop, hide** | 2 | 2 | 2 |
| **2.4.1 bypass block** | 1 | 2 | 1 |
| **2.4.4 link purpose (in context)** | 5 | 2 | 2 |
| **2.4.6 headings and labels** | 11 | 1 | |
| **3.1.1 language of page** | 1 | | |
| **3.1.2 language of parts** | | 1 | 1 |
| **3.3.2 labels or instructions** | 10 | | |
| **4.1.1 parsing** | | 1 | 1 |
| **4.1.2 name, role, value** | | 13 | 13 |
| **total issues** | **26** | **85** | **31** |

**Table 8**. Accessibility issues detected on the test suite.

All the tools found same number of issues for success criteria 1.4.3 and 2.2.2. LERA and IBM Accessibility Checker found the same number of issues for 7 out of 11 success criteria. For the test of usage of lang attribute for change of language with an invalid value, IBM accessibility checker and LERA mapped the issue to success criteria 3.1.2, while WAVE mapped this issue to success criteria 3.1.1.

### 6.2.4 Summary

WAVE reported the greatest number of issues for the success criterion 1.4.3 low contras on every page we tested. While testing the test suite, all the tools detected all the contrast tests in the test suite. This shows that the use of test suites doesn't necessarily imitate the real issues on real pages. IBM Accessibility Checker found the most issues for 5 out of 6 tested pages. And WAVE found the greatest number of issues for 1 out of 6 pages tested.

IBM Accessibility Checker was the only tool to report issues for success criteria 2.1.1 and 2.4.1 on the real pages, while LERA and IBM Accessibility Checker detected issues for these categories in the test suite, and WAVE detected issues for the 2.4.1 in the test site.

The tools also map the same issues differently to the WCAG success criteria. WAVE maps an issue of an empty or missing form label to four success criteria 1.1.1 non-text content, 1.3.1 info and relationships, 2.4.6 headings and labels, and 3.3.2 labels and instructions. While IBM Accessibility Checker maps this same issue to a success criterion 4.1.2 name, role, value. And LERA maps missing form label to two success criteria 1.3.1 info and relationships and 4.1.2 name, role, value. WAVE maps an image link with no

alternative text to two success criteria 1.1.1 non-text content and 2.4.4 link purpose. And IBM Accessibility checker maps this issue to success criterion 2.4.4 link purpose. And LERA maps this issue to two success criteria 2.4.4 link purpose and 4.1.2 name, role, value. IBM Accessibility Checker and LERA mapped an issue where the lang attribute is used to identify a change of language, but with an invalid value to success criterion 3.1.2 language of parts, and WAVE mapped this issue to success criterion 3.1.1 language of page. WAVE maps image button missing alternative text issue to two success criteria 1.1.1 non-text content and 2.4.4 link purpose. IBM Accessibility Checker and LERA map this to success criteria 4.1.2 name, role, value. While it is most important that the tools detect the issue, it may be confusing for the user if the tools map the same issue to different success criteria.

The number of accessibility issues detected by each tool seems to depend on the pages selected. One tool might detect more accessibility issues on one page than another and fewer issues on another page, depending on the type of accessibility issues on the page. Out of the selected tools WAVE appears to be best at detecting issues for success criterion 1.4.3 low contrast, while IBM accessibility Checker appears to detect most issues for success criteria 4.1.2 name, role, value, 2.4.1 bypass block, and 2.1.1 keyboard on the tested pages. If one tool is better at detecting one type of accessibility issue than other tools, and then if this type of issue is prominent on the page, then that tool is going to detect more issues on the page. As can be seen in Tables 6 and 7 vertaa.fi pages, WAVE detected more issues on the landing page and IBM on the flight search page. For that reason, using multiple automatic testing tools is recommended.

As for the scan time WAVE was clearly the fastest tool, LERA was the second fastest, and IBM Accessibility Checker was the slowest of the selected tools. Average scan time per tested pages, IBM Accessibility checker was over 7 time slower than WAVE and LERA was 4 times slower than WAVE.

### 6.3 Tool features

In this section, we go over the tool features. Tool features were gathered while using the tools to scan pages. Features listed in Table 9, are not a comprehensive list of all the features of the tools, rather it includes features that we identified as useful while using the tools.

The tools showing navigation order can be a useful feature. Especially the way WAVE implemented this feature. WAVE shows what the screen reader says. This can be useful to understand the functionality of screen readers, without the need of installing and learning to use a screen reader. Another useful feature implemented by WAVE is to toggle the styles, this feature can help to find accessibility issues hidden with styles.

IBM Accessibility Checker is the only tool to allow changing between rulesets. IBM Accessibility Checker is the only tool that allows one to select an element on the page

and show the issues on the selected element. This feature can be useful if the page large number of issues. It may be easier to select a part of the page and fix issues that way, instead of going over an overwhelming number of issues.

All three selected tools map the detected issues to the WCAG 2.1 guidelines, this allows the user to seek more information about the issue. And all the selected tools highlight issues on the page. All the tools also provide instructions on how to fix accessibility issues.

At the time of writing this, the latest WAVE Google Chrome browser plugin update is almost a month old. IBM Accessibility Checker's latest update for the Google Chrome browser plugin is a bit over a week old. And LERA's Google Chrome plugin was last updated almost a year ago. IBM Accessibility Checker and WAVE seem to be more regularly updated.

| Feature | WAVE | IBM | LERA |
|---|---|---|---|
| **Show navigation order** | Yes, shows what screen reader reads | Yes | no |
| **Toggle styles** | Yes | no | no |
| **change rule set** | no | yes | no |
| **map issues to WCAG 2.1 standards** | yes | yes | yes |
| **Highlight issues on the page** | yes | yes | yes |
| **Highlight issue in the code** | Yes | Yes, in dev tools | Code snippet |
| **Instructions how to fix the issue** | yes | yes | yes |
| **Show issues on selected part of the page** | no | yes | no |
| **Semiautomatic checks** | yes | yes | no |
| **Last update** | 17.3.2023 | 5.4.2023 | 11.5.2022 |

**Table 9.** Tool features.

# 7. Conclusion

In this thesis we compared three automatic accessibility evaluation plugins for Google Chrome in terms of efficiency, WCAG success criteria covered, and issues detected. The selected tools were WAVE, IBM Equal Access Accessibility Checker, and LERA. With the comparison of the tools, we hope to gain an understanding of the automatic accessibility evaluation tools, what these tools can test in terms of WCAG, what issues they found in Finnish e-commerce sites, and whether are there differences among the selected tools.

Regarding WCAG success criteria covered, we found that the combination of the tools covered 37 success criteria out of 78. This is more than any single tool, alone IBM Accessibility Checker covered 31 success criteria, and WAVE and LERA each covered 22 success criteria. From this, we can see that the tools cover not only a different number of success criteria but also different success criteria. Thus, the tools complement each other.

In terms of the number of issues detected, results depend on the scanned page. More precisely the number of issues detected by the tool depends on the types of accessibility issues present on the page. Most of the issues are in perceivable principle. Out of the scanned pages, success criterion 1.4.3 low contrast issues have the greatest impact on the results. IBM Accessibility Checker detected the greatest number of accessibility issues on five out of six scanned pages, while WAVE detected the most issues on one out of the six scanned pages. IBM Accessibility Checker detected the greatest number of issues on the most of tested pages, it also detected the least number of issues on one tested page.

As to types of accessibility issues detected, WAVE detected the greatest number of issues for four success criteria and IBM Accessibility Checker for four success criteria. Although the accessibility issues detected for each success criterion are maybe not a reliable metric for measuring the tool performance, the tools seem to map detected issues to the WCAG differently. WAVE tends to map an issue from one to four success criteria, while IBM Accessibility Checker maps these issues to one success criterion, and LERA maps issues one to two success criteria.

The results of the thesis align with the previous studies of automatic accessibility evaluation tools [Padure and Pribeanu 2020; Ismailova and Inal 2022; Frazão and Duarte 2020]. The tools selected for this thesis cover different success criteria and complement each other. The usage of a combination of different automatic accessibility testing tools yields better results than using a single tool. Thus, it is recommended to use more than one automatic accessibility evaluation tool. It is also important to keep in mind, that the automatic accessibility testing tools cannot detect all accessibility issues. Many accessibility requirements need human interpretation. It’s not possible to determine conformance to the guidelines with automatic tools alone.

The method of this thesis has its limitations. Firstly, we used only automatic accessibility evaluation tools, these tools can only detect a part of accessibility issues present on a page. And in this thesis, we were only interested in the automatically detected issues. Further limiting the number of the issues these tools can detect, as we discarded all issues that needed manual review. Secondly, we assumed that all the issues reported by the tools are true positives. These limitations may reward a tool that reports more issues with a cost of accuracy and penalizes tools that are more conservative and attempt to report only real accessibility issues.

In this thesis, we covered three automatic accessibility evaluation tools. We tested them on two different websites and a test site. Future studies could expand the number of tools and the number of tested pages and include more different types of websites, to gain more confidence in the results. Different types of automatic accessibility tools could be included. Future studies could also analyse alerts of potential issues, to find out if there are differences between tools. Does one tool report an accessibility issue as a detected issue, and do other tools then report the same issue as an issue that needs manual review?

Future studies could also manually analyse the automatically detected issues to compare the accuracy of the tools. A comparison against a manual conformance review of the page could also be made. To analyse how well the automatic tools detect issues compared to an expert evaluator.

## References

Aboy-Zahra, Shadi. 2008. Web accessibility and Guidelines. In Harper Simone and Yesilada Yeliz (esd.), *Web Accessibility a Foundation for Research*. Springer Science & Business media, 79-106.

Act 306/2019. 2019. Laki digitaalisten palveluiden tarjoamisesta. Valtiovarainministeriö. *https://www.finlex.fi/fi/laki/smur/2019/20190306.* (Accessed 8.5.2023)

Act 731/1999. 1999. Suomen perustuslaki. Oikeusministeriö. *https://www.finlex.fi/fi/laki/smur/1999/19990731*.

AdvancedBytez. 2023. LERA. *https://advancedbytez.com/lera/* (Accessed 20.3.2023)

Aluehallintovirasto. 2023a. Muita lakeja. *https://www.saavutettavuusvaatimukset.fi/digipalvelulain-vaatimukset/muita-lakeja/.* (Accessed 12.2.2023)

Aluehallintovirasto. 2023b. Tietoa WCAG-ohjeistuksesta. *https://www.saavutettavuusvaatimukset.fi/digipalvelulain-vaatimukset/tietoa-wcag-kriteereista/.* (Accessed 31.1.2023).

Bai, Aleksander, Heidi Camilla Mork, and Viktoria Stray. 2017. A Cost-Benefit Analysis of Accessibility Testing in Agile Software Development: Results from a Multiple Case Study. *International journal on advances in software* (2017).

Bai, Aleksander, Viktoria Stray, and Heidi Mork. 2019. What Methods Software Teams Prefer When Testing Web Accessibility. *Advances in Human-Computer Interaction*, vol. 2019, 2019, pp. 1–14, https://doi.org/10.1155/2019/3271475.

Brajnik, Giorgio. 2006. Web Accessibility Testing: When the Method Is the Culprit. In *Computers Helping People with Special Needs,* 156–163. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

Brajnik, Giorgio. 2008. A Comparative Test of Web Accessibility Evaluation Methods. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, 113–120. ACM, 2008.

Chisholm, Wendy, and Shawn Henry. 2005. Interdependent Components of Web Accessibility. In *ACM International Conference Proceeding Series; Vol. 88: Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A); 10-10 May 2005,* 31–37. ACM, 2005.

Deque Labs. 2023a. Axe-core. *https://github.com/dequelabs/axe-core.* (Accessed 20.3.2023)

Deque Labs. 2023b. Issue Impact. *https://github.com/dequelabs/axe-core/blob/develop/doc/issue_impact.md.* (Accessed 20.3.2023)

Deque Labs. 2023c. Rule descriptions. *https://github.com/dequelabs/axe-core/blob/4937bfa4f8d689f81fb89c71d6a292fcbdba767b/doc/rule-descriptions.md.* (Accessed 10.4.2023)

European commission. 2016. Directive (EU) 2016/2102 of the European Parliament and of the Council of 26 October 2016 on the accessibility of the websites and mobile

applications of public sector bodies. *https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016L2102*. (Accessed 8.5.2023)

European commission. 2019. Directive (EU) 2019/882 of the European Parliament and of the Council of 17 April 2019 on the accessibility requirements for products and services. *https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:32019L0882* (Accessed 8.5.2023)

Frazão, Tânia, and Carlos Duarte. 2020. Comparing Accessibility Evaluation Plug-Ins. *Proceedings of the 17th International Web for All Conference*, ACM, 2020, pp. 1–11, *https://doi.org/10.1145/3371300.3383346*.

Google. 2023. Google Chrome. *https://www.google.com/chrome/.* (Accessed 21.3.2023)

Government Digital Services. 2023. Accessibility tool audit. *https://alphagov.github.io/accessibility-tool-audit/test-cases.html*. (Accessed 3.4.2023)

Hornbæk, Kasper, and Erik Frøkjær. 2008. A Study of the Evaluator Effect in Usability Testing. *Human-computer interaction* 23, no. 3 (2008): 251–277.

IBM. 2023a. Tools. *https://www.ibm.com/able/toolkit/tools.* (Accessed 18.3.2023)

IBM. 2023b. IBM Accessibility Requirements. *https://www.ibm.com/able/requirements/requirements/.* (Accessed 10.4.2023)

Ismailova, Rita, and Yavuz Inal. 2022. Comparison of online accessibility evaluation tools: an analysis of tool effectiveness. *IEEE Access* 10 (2022): 58233-58239.

Karat, John, Gregory Abowd, Gaëlle Calvary, John Carroll, Gilbert Cockton, Mary Cz, and Jean Vanderdonckt. 2008. *Web Accessibility: A Foundation for Research*. 1. Aufl. London: Springer Netherlands, 2008.

Kirkpatrick, Andrew. 2006. Overview of Accessible Technologies. In *Web Accessibility*, 85–100. Berkeley, CA: Apress, 2006.

Laamanen, Merja, Tarja Ladonlahti, Hannu Puupponen, and Tommi Kärkkäinen. 2022. Does the Law Matter? An Empirical Study on the Accessibility of Finnish Higher Education Institutions' Web Pages. *Universal access in the information society* (2022): 1–17.

Nielsen, Jakob. 1993. *Usability Engineering*. 1st edition. Cambridge, Mass: AP Professional, 1993.

Padure, Marian, and Costin Pribeanu. 2020. Comparing Six Free Accessibility Evaluation Tools. *Informatica economica* 24, no. 1/2020 (2020): 15–25

Petrie, Helen, and Omar Kheir. 2007. The Relationship Between Accessibility and Usability of Websites. In *Conference on Human Factors in Computing Systems: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems; 28 Apr.-03 May 2007*, 397–406. ACM, 2007.

Petrie, Helen, Andreas Savva, and Christopher Power. 2015. Towards a unified definition of web accessibility. Proceedings of the 12th International Web for All Conference. 2015.

Richards, John T., and Vicki L. Hanson. 2004. Web accessibility: a broader view. *Proceedings of the 13th international conference on World Wide Web*. 2004.

Roemen, Dagfinn, and Dag Svanaes. 2012. Validating WCAG Versions 1.0 and 2.0 through Usability Testing with Disabled Users. *Universal access in the information society 11, no. 4* (2012): 375–385.

Rutter, Richard, Patrick H Lauke, Cynthia Waddell, Jim Thatcher, Shawn Lawton Henry, Bruce Lawson, Andrew Kirkpatrick, Christian Heilmann, Michael R Burks, and Bob Regan. 2006. *Web Accessibility: Web Standards and Regulatory Compliance*. Berkeley, CA: Apress L. P, 2006.

Schmutz, Sven, Andreas Sonderegger, and Juergen Sauer. 2016. Implementing recommendations from web accessibility guidelines: would they also provide benefits to non-disabled users. *Human factors 58, no. 4* (2016): 611–629.

Song, Shuyi, Jiajun Bu, Andreas Artmeier, Keyue Shi, Ye Wang, Zhi Yu and Can Wang. 2018. Crowdsourcing-Based Web Accessibility Evaluation with Golden Maximum Likelihood Inference. In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 2, CSCW, Article 163 (November 2018), 21 pages. https://doi.org/10.1145/3274432

University of Cambridge. 2023. Cambridge Simulation Glasses *Cambridge Simulation Glasses (inclusivedesigntoolkit.com)*. (Accessed 7.2.2023)

Verkkokauppa. 2023. Yritystiedot. *https://www.verkkokauppa.com/fi/yritystiedot.* (Accessed 3.4.2023)

Vertaa. 2023. Vertaa.fi. *https://www.vertaa.fi/info/info/.* (Accessed 3.4.2023)

Vigo, Markel, Justin Brown, and Vivienne Conway. 2013. Benchmarking Web Accessibility Evaluation Tools: Measuring the Harm of Sole Reliance on Automated Tests. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility,* 1–10. ACM, 2013.

W3C. 2018. Web content accessibility guidelines (WCAG 2.1) *https://www.w3.org/TR/WCAG21/.* (Accessed 8.5.2023)

W3C. 2014. Website Accessibility Conformance Evaluation Methodology (WCAG-EM) 1.0. *https://www.w3.org/TR/WCAG-EM/.* (Accessed 6.2.2023)

WAI. 2023a. Accessibility intro. *https://www.w3.org/WAI/fundamentals/accessibility-intro/.* (Accessed 15.1.2023)

WAI. 2023b. Accessibility, Usability, and Inclusion. *https://www.w3.org/WAI/fundamentals/accessibility-usability-inclusion/.* (Accessed 8.5.2023)

WAI. 2023c. Web accessibility evaluation tools list. *https://www.w3.org/WAI/ER/tools/.* (Accessed 31.1.2023).

WAI. 2023d. W3C accessibility standards overview. *https://www.w3.org/WAI/standards-guidelines/.* (Accessed 13.2.2023)

WebAIM. 2023a. WAVE web accessibility evaluation tools. *https://wave.webaim.org/.* (Accessed 20.3.2023)

WebAIM. 2023b. WAVE-WCAG Mappings. *https://docs.google.com/spreadsheets/d/1oSyK4QiyHf1zx-xrc4P9M7efYVv0vohNxVWjeHan8Iw/edit#gid=902071383.* (Accessed 10.4.2023)

World Health Organization. 2011. World report on disability summary. WHO/NMH/VIP/11.01 *https://www.who.int/publications/i/item/WHO-NMH-VIP-11.01*

Yesilada, Yeliz, Giorgio Brajnik, Markel Vigo, and Simon Harper. 2012. Understanding Web Accessibility and Its Drivers. In *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility*, 1–9. ACM, 2012.