Universal Access in the Information Society
https://doi.org/10.1007/s10209-022-00948-x

LONG PAPER

Usability and transparency in the design of a tool for automatic support for web accessibility validation

Nicola Iannuzzi[1] · Marco Manca[1] · Fabio Paternò[1] · Carmen Santoro[1]

Accepted: 8 November 2022

Abstract
The importance of guaranteeing accessible Web applications is becoming widely recognised, and supported by national and international legislation. This implies an increasing need for large scale validations, which can be achieved only through automatic support. Thus, there is a need for a new generation of accessibility validation tools able to check against the continuously evolving guidelines and report any issues revealed, considering that their results will be used by many people with varied backgrounds and goals. Such tools should be able to support monitoring of Web sites' accessibility, provide user-friendly information suitable for various purposes, and be transparent in terms of what they are actually able to evaluate in order to elicit the right expectations from their users. They should also consider how the technologies for implementing Web sites have evolved in recent years. In this paper, we provide a description of the requirements and design dimensions that characterise such tools in order to address emerging needs, providing indications on whether and how several existing validation tools support them, and present how we have addressed and implemented them in a specific tool. We also report on a first usability test of such tool, which has provided encouraging feedback.

Keywords Accessibility · Automatic validation tools · Accessibility monitoring

1 Introduction

The importance of providing accessible Web applications for all, including people with cognitive or physical disabilities, has become increasingly recognised. This is confirmed by the indications provided by national and international legislations to support it [1], (Lazar and Olalere, [2, 3]. A major initiative to address such aspects is the EU Directive on the "Accessibility of the Websites and Mobile Applications of Public Sector Bodies" that came into force on 26/10/2016, also known as the Web Accessibility Directive (WAD) (EU [4], which establishes accessibility requirements for the websites and mobile applications of public sector bodies. One aspect that is particularly stimulated by this directive is monitoring, which should be done more systematically in terms of the number of Web pages involved, and with some level of frequency.

In parallel, the guidelines for accessibility of Web Sites, which are developed by the W3C in the Web Accessibility Initiative, are continuously evolving considering the need for better addressing the various possible disabilities and the evolution of Web Technologies. The current version (2.1) is structured into principles, guidelines, success criteria, and techniques, which have increased compared to the previous versions. Thus, the WCAG 2.1 added 17 new success criteria to address mobile access and some disabilities better, so that now there are overall 82 success criteria.

All such aspects imply that thorough accessibility validation requires considerable effort for the number of elements and aspects that have to be checked. Thus, interest in automatic support for this activity is continuously increasing, and stimulates further research and development in this area because of its potential to support the collection and analyses of data on the effective application of the accessibility guidelines, detect non-compliance in a consistent manner,

✉ Fabio Paternò
fabio.paterno@isti.cnr.it

Nicola Iannuzzi
nicola.iannuzzi@isti.cnr.it

Marco Manca
marco.manca@isti.cnr.it

Carmen Santoro
carmen.santoro@isti.cnr.it

[1] CNR-ISTI, HIIS Laboratory, Pisa, Italy

Published online: 24 November 2022

and provide relevant information on how to address possible problems. At the same time, it is important to be aware that not all accessibility issues can be automatically detected, some of them require manual checking from accessibility experts, and subjective feedback is still important to consider [5].

In order to provide automatic support for accessibility, many proposals have been put forward. Bobby was probably the first accessibility tool widely used. Developed and released in 1996 by the Center for Applied Special Technology (CAST), Bobby was a free online accessibility tool that was used to evaluate against WCAG 1.0, WAI and US Sect. 508. In about 2006, it was acquired by IBM and then removed from service. Over time several researches and development efforts have been carried out in this area (e.g. [6–12]). As of May 2022, the W3C Web Accessibility Evaluation Tools list[1] contains 161 elements, and further tools, which are not included in it, have been put forward. However, many of them have limited impact for several reasons: some have not been able to evolve in order to address the most recent WCAG version [13, 14], some address only specific aspects, such as colour contrast or readability [15] or lexical simplification [16], some provide only information in national languages, and there is not an English version.

Abascal et al. [17] provide a set of useful criteria to analyse the support provided by accessibility tools: the type of license (free versus commercial), the platform where they can be executed; the evaluation scope (ranging from single pages to entire websites); the support provided for repairing identified issues; how the evaluation results, guidelines supported, and detected issues are rendered and exported. However, there are at least three emerging important aspects missing in such classification: accessibility monitoring, tool transparency, and support for dynamic Web sites. The first means the ability to indicate a set of Web pages and periodically check the level of accessibility in order to inform relevant stakeholders about how it evolves. The second aims to address one important problem that users of automatic validators often encounter when using multiple tools: they may provide different results. For example, a study on automatic Web accessibility evaluation (Abduganiev, 2017), which only considered support for the previous WCAG 2.0 guidelines, analysed eight popular and free online automated Web accessibility evaluation tools finding significant differences in terms of various aspects (coverage, completeness, correctness, validity, efficiency and capacity). Users of such tools are often disoriented by such differences and find them somewhat unclear. Thus, it becomes important that the tools be transparent and indicate in detail what they are actually able to validate [18]. The last aspect aims to address the increasing use of development frameworks that implement dynamic Web sites, such as Angular or Vue.js. In these cases, the Web pages that are created and sent to the browsers contain a few elements, which can be deeply modified and extended at loading time, when their JavaScripts are executed in order to obtain the actual Web page content shown to users. Thus, a traditional validator that only analyses the initial version of the page would be able to actually consider only the initial few elements. Equipping the validator with functionalities able to perform server-side rendering and then analyse the corresponding results would provide more meaningful results.

Overall, we think it is time for a new generation of automatic tools for accessibility validation characterised by the ability to address such aspects. In this paper, we indicate the requirements and design aspects that have to be considered in order to address such issues, also discussing whether and how existing validation tools support them, and present how we have addressed them to be included in a previous validation tool [19], which has a modular approach to managing guidelines (which has been adopted also in other tools [20]), it is publicly available and has a large community of users. In particular, after discussing related work, we indicate the requirements that have driven the novel work, present the corresponding design and implementation in the tool, and report on a first user test, which provided encouraging feedback. To our knowledge, such aspects have not been considered in the literature. There are some commercial tools that provide some support in this direction, but their authors have not described how they obtain it. Thus, this paper can be useful for tool and application developers in order to understand how the emerging requirements can be addressed, and for all the community interested in Web accessibility validation to better understand the actual possibilities of automatic support.

2 Related work

While there are several tools that offer some support to locate and visualise errors in some way, the issue of supporting monitoring functionalities has been addressed in a more limited manner until now. Indeed, interest in the possibility of monitoring Web sites on a periodical basis has increased recently, also thanks to the EU Directive that enforced recurrent monitoring of websites of public bodies and organisations (which typically include many pages). Prior work discussed and compared some automatic accessibility evaluation tools (see e.g. [17, 21]), but their analysis did not consider some emerging requirements, such as monitoring the accessibility of Web sites over time. A first exploration of how to support monitoring of Web sites accessibility at a geopolitical level was discussed in [14], but

[1] https://wave.webaim.org/

the tool presented in that paper provided only some limited representations in a tabular format of the accessibility levels detected for older WCAG versions. More recently, [22] have presented a comparative analysis of four commercial accessibility validation tools. They considered SiteImprove, PopeTech, aXe Monitoring and ARC Monitoring, which were evaluated by the authors by using trial versions sent to them by the respective vendor companies. In particular, their analysis was done in an analytical and an empirical manner: in analytical manner, by using a set of evaluation criteria (i.e. coverage of web pages, coverage of success criteria, correctness, support for localisation of errors, degree of implementing gamification patterns) with specific weights assigned to such criteria, and in empirical manner by involving 15 users who had to freely explore every tool on their own for some time (i.e. without concrete tasks to carry out). The evaluation criteria were partly based on those used in other studies (Abduganiev [23], Padure and Pribeanu [24], and Vigo et al. [21]) for comparing automatic Web accessibility evaluation tools. In addition, they also considered aspects aiming to understand how user-friendly and motivating they are to use. However, their analysis did not focus on the aspects that are addressed by our proposal (monitoring, transparency, and support for dynamic sites).

Some tools, such as Adaplugin, Accessibility Scanning & Monitoring by UserWay, and AccessiBe, offer some monitoring support and support recurrent scan. However, they are commercial ones and therefore not directly and easily exploitable by all interested and relevant stakeholders. Generally, as the only alternative to purchasing one of the available subscription packages, they provide free demos or e-tours, or in the best case, a free trial for a limited period of time (i.e. 10–14 days) with several limitations (e.g. scan just a single page, one time per month). We aim at providing better support in a tool that can be freely and easily accessed and used by different relevant stakeholders interested in monitoring the accessibility of their websites.

Generally, it is easy to see that when applying different validation tools to the same Web content, they provide different results, and users have difficulties understanding the reasons for such variability and to what extent the results are meaningful. Thus, also for improving their usability, there is a need for more transparency to help users better interpret their results [18]. In another work, [21] analysed the effectiveness of six frequently used accessibility evaluation tools in terms of coverage, completeness, and correctness concerning the WCAG 2.0 guidelines. They found that coverage was narrow as, at most, 50% of the success criteria were covered, and similarly, completeness ranged between 14 and 38%. In addition, some of the tools that exhibit higher completeness scores produced lower correctness scores (66–71%) because catching as many violations as possible can lead to an increase in false positives. Lastly, they indicated that effectiveness in terms of coverage and completeness could be boosted if the right combination of tools is employed for each success criterion. A further study on automatic Web accessibility evaluation (Abduganiev, 2017), which only considered support for the previous WCAG 2.0 guidelines, has analysed eight popular and free online automated Web accessibility evaluation tools finding significant differences in terms of various aspects (coverage, completeness, correctness, validity, efficiency and capacity). More recently, Padure and Pribeanu [24] compared five automatic tools for assessing accessibility. The result of the study indicates that the combined use of two of the considered tools would increase the completeness and reliability of the assessment. Frazão and Duarte [25] focused their analysis of accessibility on validation plugins extensions for the Chrome Web browser. They found that individual tools still provide limited and varied coverage of the success criteria. After analysing their results, they recommend using more than one tool and complementing automated evaluation with manual checking. Solutions based only on plugins are able to address issues related to validation of dynamic pages since they access directly the DOM in the browser but are not suitable for monitoring Web sites. In general, none of such studies focused on transparency aspects and how to help users understand how the accessibility evaluation tools work by providing clear information about their coverage and working. Indeed, in a survey (Yesilada [26]) respondents strongly agreed that accessibility must be grounded on user-centred practices and that accessibility evaluation is more than just inspecting source code.

3 Tool design

3.1 Requirements

In our work, we started by identifying a set of requirements (R1 – R8) that should characterise a new generation of tools for supporting accessibility validation. For this purpose, we analysed the state of the art in the scientific literature in this area, in which some studies discussed the usability problems of accessibility evaluation tools (e.g. [27–30]. In addition, in previous work [31] some requirements elicitation activities (online questionnaires, interviews and workshops) for new accessibility validation tools were carried out in the context of the WADCHER European project. We also considered our direct experience with tools in research and international projects, collaboration with the national agency for accessibility, teaching accessibility validation in HCI courses, and, more generally, with the analysis of current accessibility validation practices, and observations of feedback gathered from interaction with accessibility experts and users of the previous version of our tool. The resulting requirements are

listed below. In presenting them we also discuss whether and how current tools address them. Our analysis focuses on publicly available tools, not commercial ones. In such analysis of public tools, we note that while it is possible to find examples that address some of the requirements, there is a lack of proposals aiming to consider all of them.

R1. The tool should be able to support different types of stakeholders. Managing a public Web site typically involves several people with different backgrounds and goals. In general, we can identify Web commissioners, who are responsible for the site and determine its purpose, but often have little knowledge about the design and implementation techniques for accessibility. Web developers are those who actually implement the Web site, and often need support to understand how to address the accessibility guidelines. Sometimes there is also the involvement of User Interface Designers, who have knowledge on user experience and how to support it, but often do not know accessibility guidelines. In some cases, there are Accessibility Experts, who can manually check whether the Web elements satisfy the guidelines, but so far, there is still a rather limited number of these experts, and their work is problematic since modern Web sites contain many elements to check, which is quite difficult to do manually. This aspect has not been sufficiently considered so far. Some tools (such as Siteimprove [32] and Wave) have started to address the issue of supporting different types of stakeholders. For example SiteImprove allows users to filter the accessibility issues depending on their role (developers content writing UX design visual design) and the estimated difficulty needed to fix the issue (beginner intermediate advanced expert). However there are still several tools such as Achecker (Achecker [33]) or QualWeb (QualWeb 2022), which tend to provide the results in terms of errors and warnings lists (which can be filtered or navigated in some ways). Usually such presentations are useful for web developers who modify the implementation to improve its accessibility but are not immediately understandable or relevant for other user categories, such as Web commissioners.

R2. The tool should be able to validate single and groups of Web pages. The support provided by the tool should be flexible in terms of the granularity of what is being evaluated, also because at different times there may be different types of requests. They can range from on-the-fly validation of a single page (which for some reason is of particular interest because someone may have complained about its accessibility, or it is particularly important and requires a more complete verification), to groups of pages, up to entire Web sites. In this perspective, the EU directive 2016/2012 has indicated a methodology (EU 2018/1524) whereby the pages should be validated according to two modalities. The in-depth one is intended to thoroughly verify whether a website satisfies all the requirements identified in the standards and technical specifications referred to in the EU Directive. Therefore, it is supposed to evaluate the interaction with forms, interface controls and dialogue boxes, the confirmations for data entry, the error messages and other feedback resulting from user interaction when possible, as well as the behaviour of the website or mobile application when applying different settings or preferences. Then, there is a simplified modality, to detect instances of non-compliance with a sub-set of the requirements in the standards and technical specifications referred, and related to the requirements concerning perceivability, operability, understandability and robustness. In general, the possibility to validate multiple pages is supported only by commercial versions of some tools (such as Siteimprove and DYNO Mapper), while this kind of support is rare in publicly available tools (one of such cases is Experte [34]).

R3. The tool should be able to monitor over time the level of accessibility. Often Web sites are modified because the content and the functionalities need to be updated, or because some usability or accessibility problems have been signalled by end users, thus there is a need for periodically checking the accessibility level to see whether it has improved. In addition, the EU directive requires that all the European countries provide monitoring of their levels of accessibility, also indicating the number of websites to monitor depending on the number of inhabitants. The situation in this case is similar to that regarding the previous requirement. Its support in public accessibility tools is completely lacking, while a few commercial tools (such as Siteimprove, Scanning & Monitoring by UserWay, and AccessiBe) provide some support.

R4. The tool should be able to connect the errors identified to both the page code and the page user interface. Once errors are identified, web developers need immediate support to localise them in the code to understand how to fix them. However, also for stakeholders other than the technical ones (e.g. web commissioners or user interface designers), it is also useful to indicate the user interface elements affected by any errors (locate the errors within the user interface of the page) to better understand their actual impact on the user. In this case, the tools developed as browser extension plugins (such as SiteImprove and IBM Accessibility Checker (IBM 35 exploit direct browser access to satisfy such requirement. WAVE and DYNO Mapper support it by adding icons representing errors and warnings within the user interface preview of the validated page, and showing the corresponding code excerpt on a sidebar. AccessiBe shows a web page preview, but the code representing the error/warning is not connected to the page elements. Lighthouse [36], QualWeb and Expert partially support it: the first two tools show the rendering of the element causing the accessibility issue unconnected to the whole page context, while

the third one shows the code next to a small and difficult-to-understand image.

R5. The tool should be able to provide quantitative summary information on the actual level of accessibility identified. One characteristic of the accessibility validators is that often they generate long lists of issues detected; in some cases they are errors, in others they are warnings, several of them are minor issues, thus their impact can significantly vary, and in the end people have difficulties estimating the overall accessibility level. Thus, some summary quantitative estimations can be useful. Some efforts in the area of accessibility validation have been put forward; however, some of the metrics proposed are complicated and require deep technical and accessibility knowledge that often people do not have, thus resulting rather obscure. This requirement has been addressed in limited manner so far. AccessiBe and IBM Accessibility Checker provide a score based on the number of elements that return an error over the total number of analysed elements. Regarding the accessibility calculation, in Lighthouse the associated documentation,[2],[3] reports the list of the considered accessibility audits, and each audit has a weight depending on its importance: however, this information is not very clearly reported. In DYNO Mapper all the guidelines are weighted equally.

R6. The tool should be transparent indicating what it is actually able to check and what it cannot. One common issue that people who use accessibility validators encounter is that for a given Web site, different tools often provide different results. One of the main reasons for such differences is that they vary in terms of the techniques that they are able to validate, or because they differently implement the accessibility checks. However, this is not immediately understandable because usually such tools claim to support a given set of guidelines (e.g. WCAG 2.1) without indicating to what extent they actually support it. Indeed, the validation of WCAG 2.1 is implemented by analysing many techniques, and some of them cannot even be automatically checked. If we focus on those that can be checked, we still have a large number of validation techniques, and it is rather rare that the validators provide an indication of those they are able to address, and more generally how complete the validation performed is. This is one of the requirements with the weakest support. WAVE, Lighthouse and QualWeb report the implemented techniques in some way. Lighthouse reports the list of the supported checks in the document explaining the accessibility metric. QualWeb is an open source project, and in its GitHub repository it is possible to check the list of supported WCAG techniques[4] and ACT Rules.[5] WebAim, the foundation in charge of developing WAVE, provides a document[6] containing "WebAIM's interpretation of WCAG guidelines and success criteria and their own recommended techniques for satisfying those success criteria".

R7. The tool should provide practical indications about how to solve the identified problems. Once the issues are identified, their correction requires some technical knowledge: some immediate access to relevant resources would certainly be appreciated, and make more efficient the correction process. However, in this case the challenge is that sometimes there is no general one-fits-all solution for solving a specific issue, as a meaningful modification can additionally require understanding the context of the error (i.e. the content and state of the relevant part of the web page). This requirement is considered to some extent by several tools by indicating the links to the W3C WCAG documentation (e.g. QualWeb, DYNO Mapper) or to web.dev, a website managed by google to guide modern web development (e.g. Experte and Lighthouse). In some tools, examples of code respecting the considered success criteria are provided (Siteimprove, Achecker, IBM Accessibility Checker, AccessiBe and WAVE), even if sometimes they are not particularly relevant for the specific case that has generated the issue.

R8. The tool should be able to validate dynamic Web pages. Often modern Web sites are developed with JavaScript frameworks that deeply modify the content of the pages through scripts that are executed at the browser's loading time. Thus, a complete validation should be able to also address the dynamically generated content, even though this increases the complexity of the validation since the generated content is usually much larger and more varied than the initial static version. The tools implemented as a browser plugin (e.g. SiteImprove, WAVE plugin, Lighthouse and IBM Accessibility Checker) are able to satisfy this requirement, since they serialise the current DOM rendered in the browser and send it to the validator. Experte and DYNO Mapper apply some kind of server-side rendering to analyse highly dynamic pages correctly. Several tools such as Achecker, WAVE, AccessiBe are not currently able to validate such pages.

3.2 Tool functionalities and how they are presented to the users

This section reports the proposed tool characteristics, discussing how they address the requirements R1-R8 identified in the previous section. By using the tool, it is possible to

[2] https://web.dev/accessibility-scoring/

[3] https://github.com/GoogleChrome/lighthouse/blob/v6.5.0/docs/scoring.md#how-is-the-accessibility-score-calculated

[4] https://github.com/qualweb/wcag-techniques

[5] https://github.com/qualweb/act-rules

[6] https://webaim.org/standards/wcag/checklist

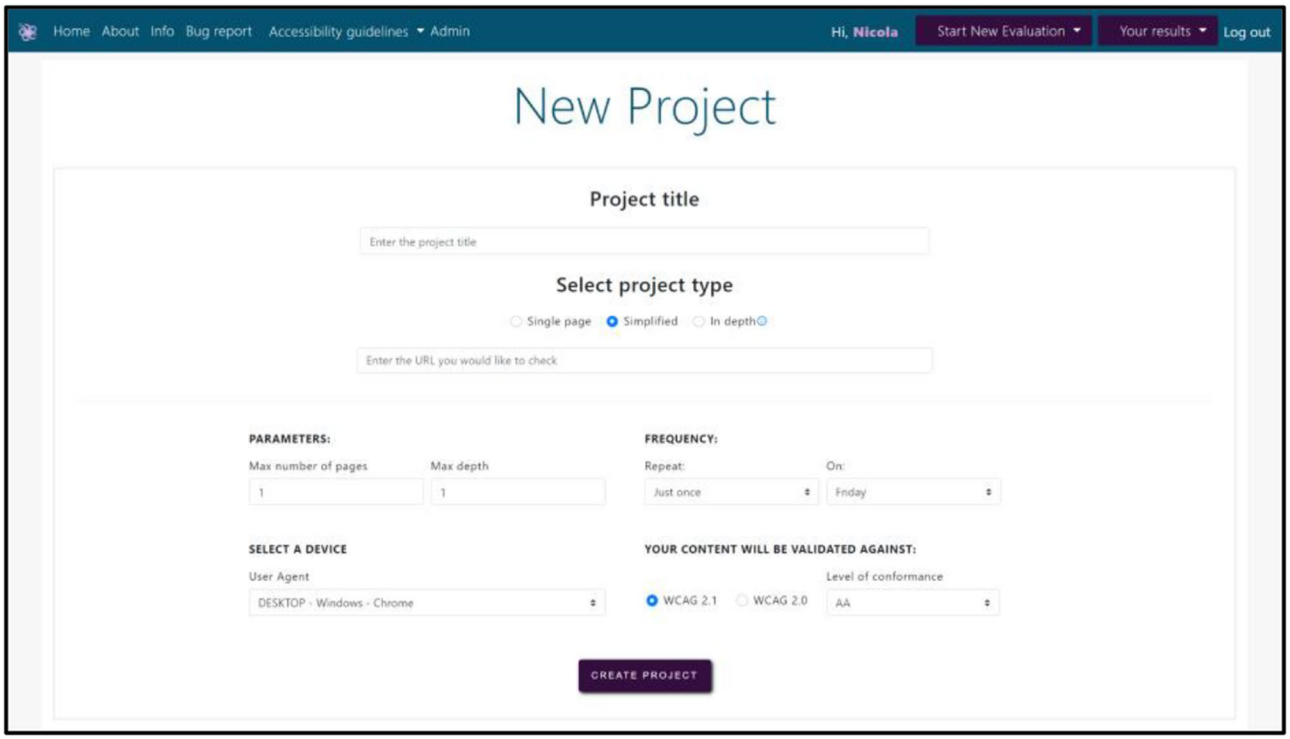

Fig. 1 Creating a Project

evaluate a single page or create a project, which represents one page or a set of pages to validate using the same validation settings within an "audit" (R2). An audit is an accessibility inspection that can be done either once or multiple times, which can be scheduled for being periodically and automatically activated at a defined interval of times decided by the user. The introduction of "projects" allows users to request multiple evaluations of the same page or the same group of web pages (belonging to the project), to be able to compare the obtained results and monitor the evolution of accessibility over time (R3). Users can configure a project according to a set of parameters based on the type of evaluation to carry out. We defined three project types:

- *Single Page Project*. This is the simplest case of project, and it should be selected when the user wants to monitor the evolution of accessibility over time of just one web page. For creating a Single Page Project, it is necessary to specify just its URL, the level of conformance requested, the type of device (desktop, smartphone, tablet) and the user agent (operating system and browser) to simulate when accessing the Web page. It is worth noting that it is also possible to carry out the validation of a single page *without creating an associated project*: in this case it is possible to choose whether the validation will consider only static content or also the dynamic one (by using the server-side rendering validation). Instead, when validating a *project*, the tool always considers the dynamic content obtained through server-side rendering (R8), since this takes longer time, the results are provided asynchronously.
- *Simplified Project*. It evaluates a subset of pages belonging to a website (see Fig. 1) starting from a base URL, and selected according to two parameters: the maximum number of pages to consider, and the maximum depth to reach when selecting the pages starting from the home page in the website domain (R2).
- *In-depth Project*. It corresponds to the most thorough validation: in this case the tool evaluates a list of representative pages (R2) such as Home, Login, Sitemap, Accessibility Statement and verifies whether the website satisfies all the requirements identified in the standards and technical specifications referred to in Article 6 of EU Directive 2016/2102.

For all types of projects, it is also possible to schedule a specific frequency by selecting the days to automatically perform the evaluation (e.g. Sundays, Mondays), and whether it should be performed every week, every two weeks, or four weeks. It is possible to select the particular set of guidelines to use (currently WCAG 2.1 or WCAG 2.0), the conformance level, and also the device and user agent to consider for the validation.

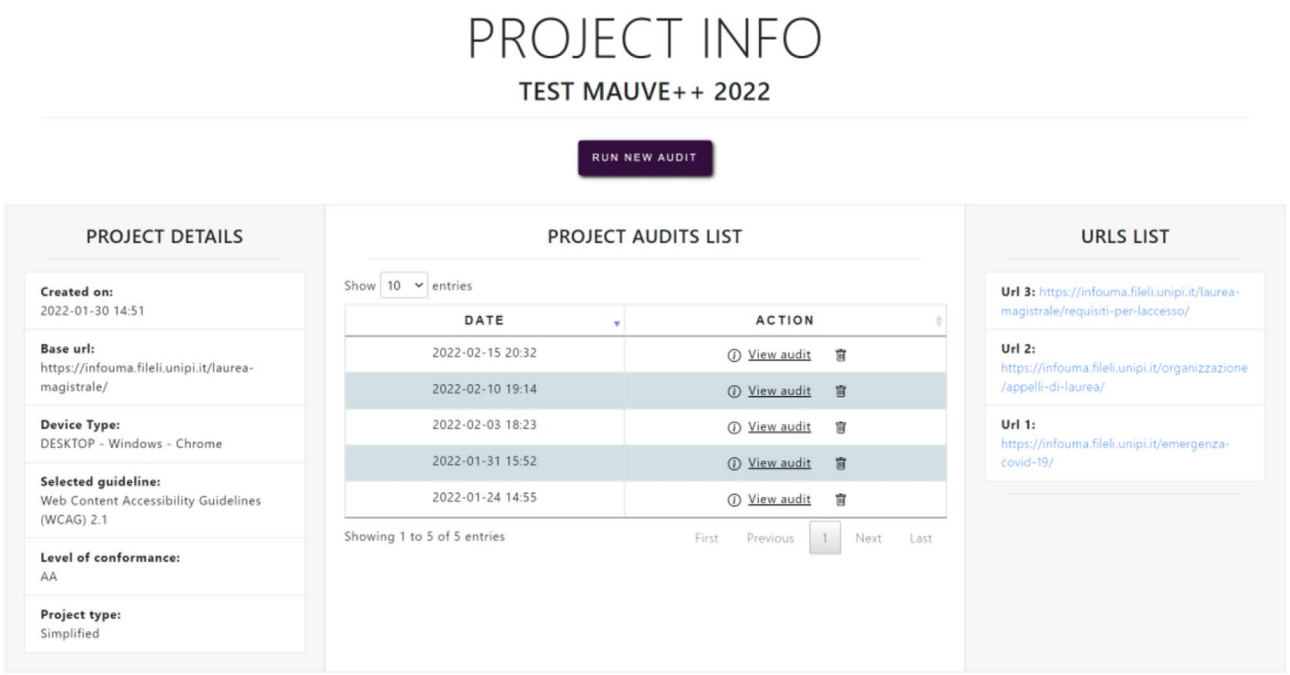

Fig. 2 The top part of the page dedicated to showing the information about a project (summary information)

In addition, on a page called "Info", the tool reports the list of Success Criteria and the associated Techniques that it is able to evaluate, for providing transparent information regarding what it is actually able to validate (R6). It also provides access to the XML-based specification of how each technique is interpreted by the tool for validation purposes. Thus, its users can better understand how and when the tool determines erroneous cases.

After having created a project, it will be added to the user's workspace together with the other projects available for that user. The tool contains a specific section showing the information associated with each specific project. Figure 2 shows the top-most part, namely the one dedicated to providing a summary information of the selected project, from left to right: the details of the project (creation date and main configuration parameters), the list of audit(s) associated with it, and the URLs of the web pages considered in this project.

When the user selects a specific audit of a project (through the link "View audit" in the central panel of the window shown in Fig. 2), in a new page it is possible to get the information associated with that audit. The details are shown divided into the following sections:

- **Results**: a summary of the results produced by the considered audit, in terms of the average number of techniques (calculated over the pages belonging to that audit) with error/warning/success/non-applicable results;
- **Page Results:** a graph visualising, for each page, the number of errors (or warnings) found: the X axis corresponds to the pages belonging to the audit, the Y axis to the number of occurrences of the different types of issues found (errors/warnings). By hovering the mouse over each dot shown in the graph, it is possible to display the corresponding information (the URL of the page considered, number of errors or warnings);
- **List of Evaluated pages:** the list of pages assessed in the audit is provided in the last section. Further information on the evaluation of each page can be seen by selecting the corresponding link.

When the user selects a specific page evaluated by the tool (through one of the links provided in the section "List of Evaluated Pages"), different pieces of information are provided via multiple tabs (see Fig. 3):

- **Evaluation Summary (R5)**, which provides the parameters specified by the user at validation creation time, the summary of the corresponding results in terms of number of errors, warnings, successes and not applicable techniques, the possibility to download the evaluation report in PDF or in EARL[7] format [[37]] (the W3C standard to represent test results), and two metrics that provide overall information about the accessibility level of the evaluated pages;

[7] https://www.w3.org/WAI/standards-guidelines/earl/

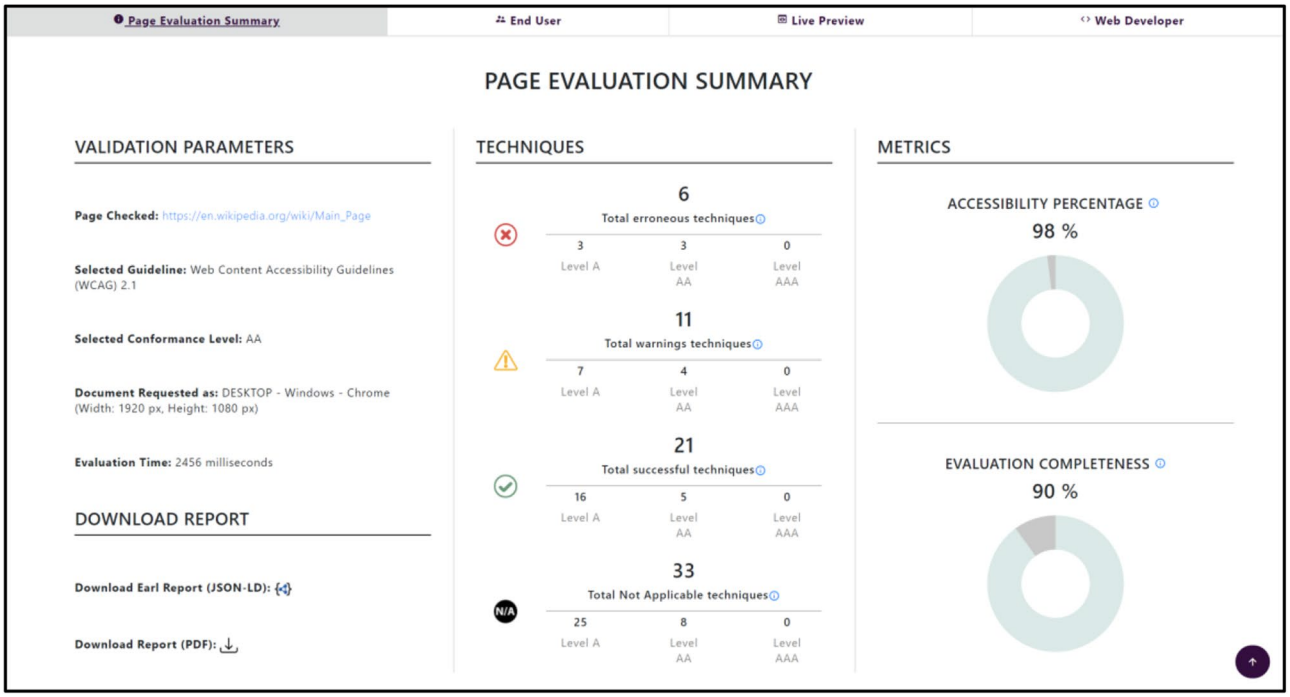

Fig. 3 Page Evaluation Summary

- **End User View (R1)**, which provides the results grouped according to various parameters, i.e. principles (e.g. perceivable, operable), categories (e.g. ARIA, content) and code type (e.g. HTML, CSS), which can be understood even by people with limited technical knowledge;
- **Live Preview (R1, R4)**, which allows users to visually locate the identified issues directly within the web page user interface, and connect them to the corresponding code;
- **Web Developer View (R1)**, which highlights the errors/warnings within the code, particularly useful for Web developers.

While the Page Evaluation Summary provides summary information about the evaluation of a page, which can be of interest for different categories of users, the goal of the last three views (End-User, Live Preview and Web Developer views) is to support different stakeholders that need to access accessibility information for different purposes (R1). For instance the Web Developer view mainly targets technical people, whereas the End-User view aims to support end users through the provision of summative tables and charts.

The Page Evaluation Summary (see Fig. 3) in its central part provides an overview of the issues found in the validations of the various techniques applied to the considered page. In particular, it groups them into four categories, counting the number of: i) techniques that in at least some cases resulted as violated (errors); ii) techniques that were applied but their evaluation did not give a definitive answer; therefore, a human check is needed (warnings); iii) techniques that resulted in successful application (successes); iv) techniques that was not possible to apply because not relevant for the considered web page (non-applicable). It also provides such numbers according to the conformance levels (https://www.w3.org/TR/WCAG21/#cc1).

The two metrics are visualised using a doughnut chart representation. They are calculated based on the following values: S = number of validation techniques applied successfully; E = Number of validation techniques applied unsuccessfully; W = Number of validation techniques applications that require a manual evaluation. They are both percentages, defined as follows:

- **Accessibility Percentage = S/(S + E)**: Number of distinct techniques successfully evaluated out of the total number of techniques for which the tool was able to make a successful or unsuccessful evaluation;
- The sum of unsuccessful evaluations is weighted by taking into account the conformance level of the corresponding validation techniques: a “Level A” technique is considered with weight 1, “Level AA” with weight 0.6 and “Level AAA” with weight 0.2. This has been done because errors generated by techniques of level AAA are considered less “critical” than “Level AA” techniques, which in turn are less critical then “Level A” techniques

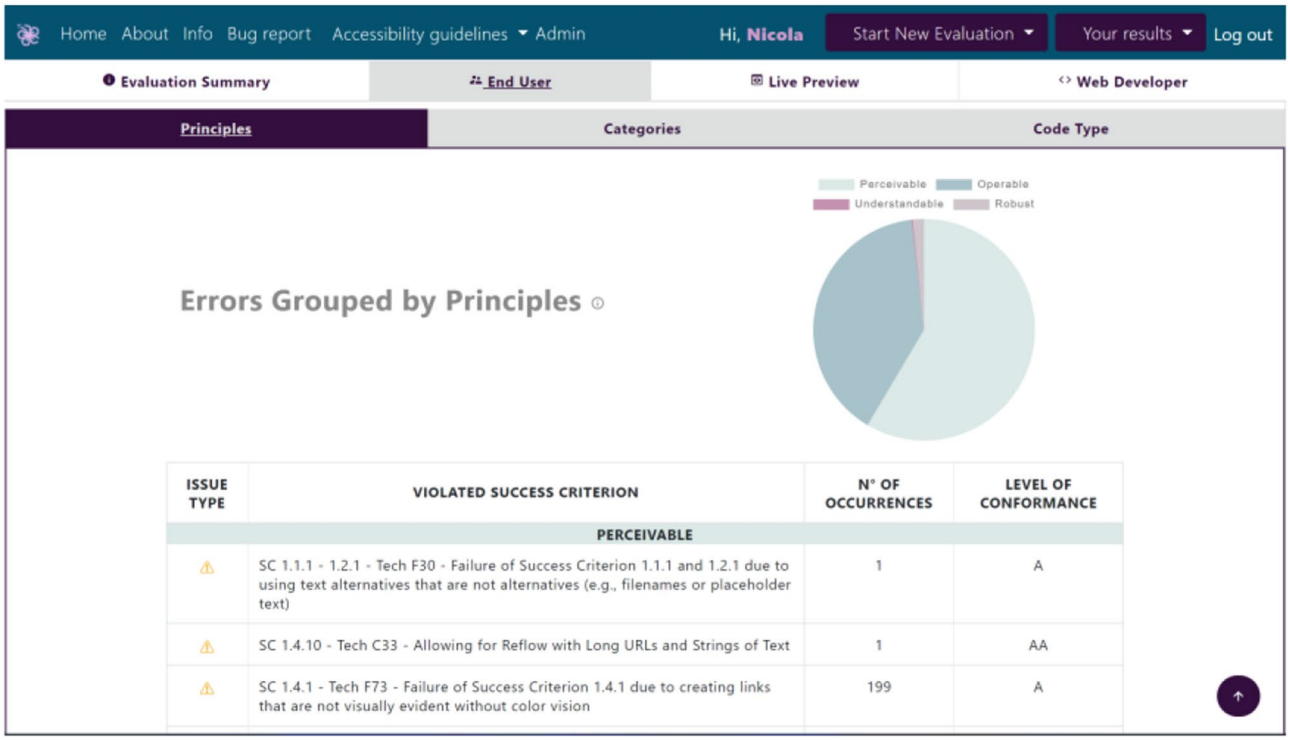

Fig. 4 End User View with errors grouped by principles

and consequently their impact on the accessibility level calculation is lower.

- **Evaluation Completeness = (S + E)/(S + E + W)**: Number of distinct validation techniques applications for which the tool was able to carry out a successful or unsuccessful evaluation compared to the total number of validation techniques applications. We introduced this metric to make it more transparent to the users that the tool is not able to decide on the accessibility of all the web elements analysed. Thus, even if the accessibility percentage score is 100%, they still have to check the evaluation completeness to understand whether the automatic validation has been able to decide on the correct application of all the validation techniques.

Such metrics are helpful to provide a compact indication of the current level of accessibility (accessibility percentage—R5), but also remind their users that the automatic evaluation may not be complete, and provide an indication of the level of completeness (evaluation completeness).

The goal of the "End User" view (see Fig. 4) is to provide information useful for Web commissioners, or in general people with limited technological knowledge but still interested in understanding the aspects more problematic for their website from an accessibility perspective. In this view the results are provided according to different aspects, which the user can select among different tabbed panels (see Fig. 4): i) the involved WCAG 2.1 accessibility principles (namely: perceivable, understandable, robust, operable); ii) the type of elements involved in the issues found (see the "Categories" tab in Fig. 4); iii) the "Code Type" tab, to see which ones are HTML or CSS-related errors. Figure 4 shows the tab dedicated to the accessibility principles: for each type of issue found (among errors and warnings) the technique that was involved is reported as well as the corresponding success criterion, and this information is grouped under the specific WCAG 2.1 principle involved.

The "Live Preview" aims to provide end users with the possibility to easily localise a specific error directly within the user interface of the considered page, to have more information about the error, and better understand its potential impact on the user (Fig. 5). There is a left-hand panel with two parts. One is dedicated to listing and filtering the elements to analyse: errors or warnings, HTML or CSS elements, errors related to specific WCAG principles. The other part shows the filtered elements: for each type of error, the number of occurrences and the list of such occurrences are indicated. The user can select a specific occurrence of an error (by clicking on the associated eye-shaped icon), and then automatically the associated part of the web page is highlighted in red within the page shown in the central panel. If the user hovers the mouse over that highlighted element, guidance information about how to solve the associated issue is displayed (R7—see the "How to solve?" tooltip

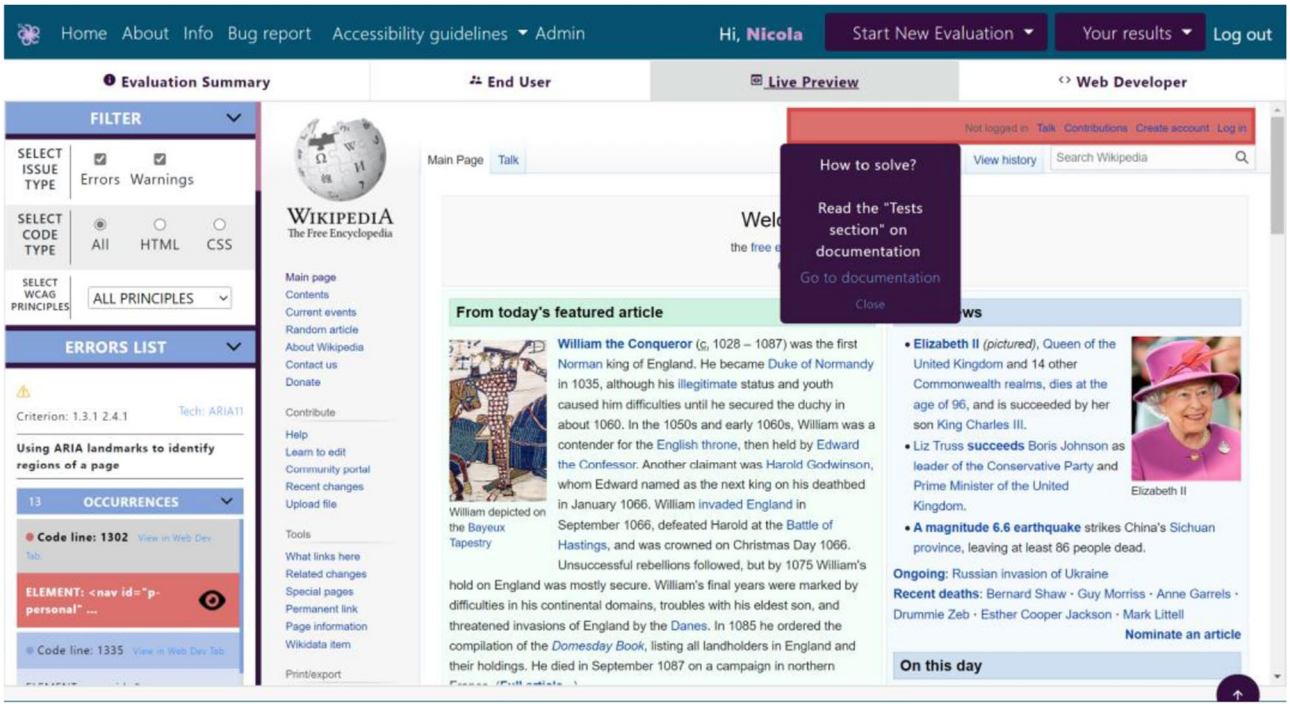

Fig. 5 The Live Preview showing (left part) the list of accessibility issues found by the tool and (main panel) a visualisation of the involved page

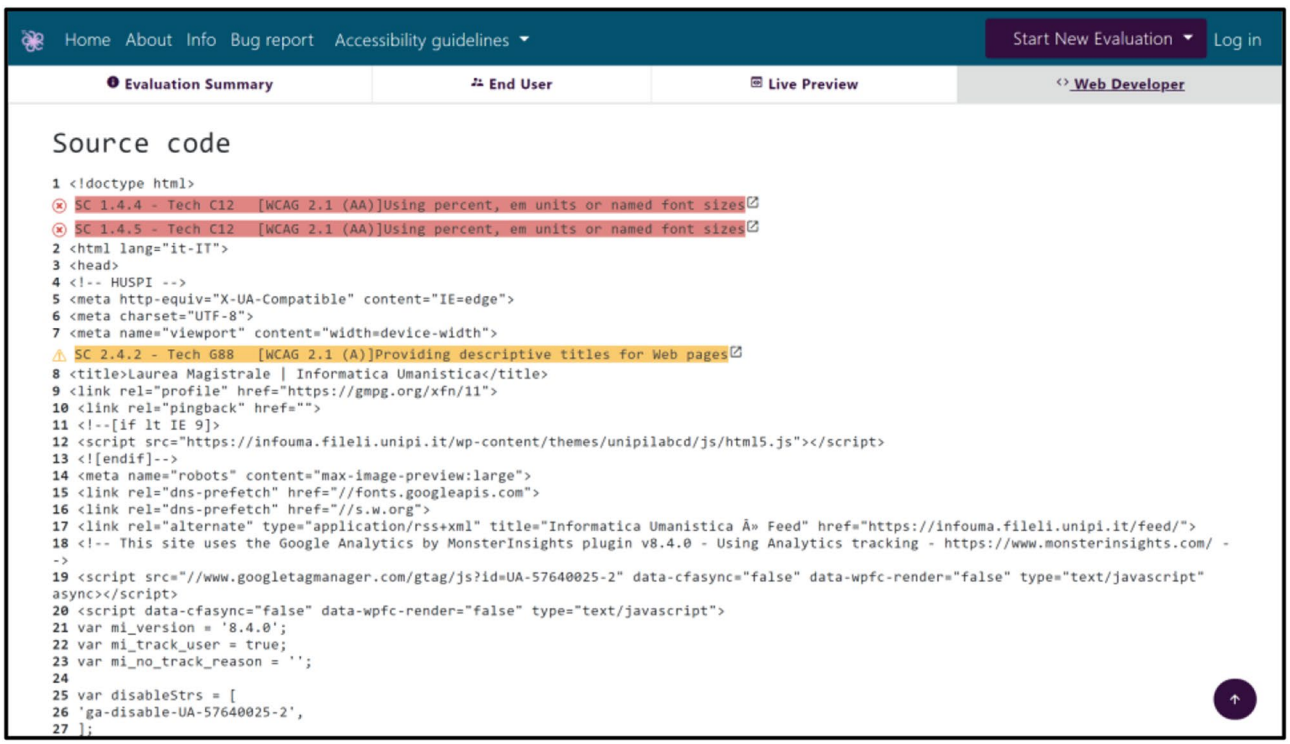

Fig. 6 The Web Developer View

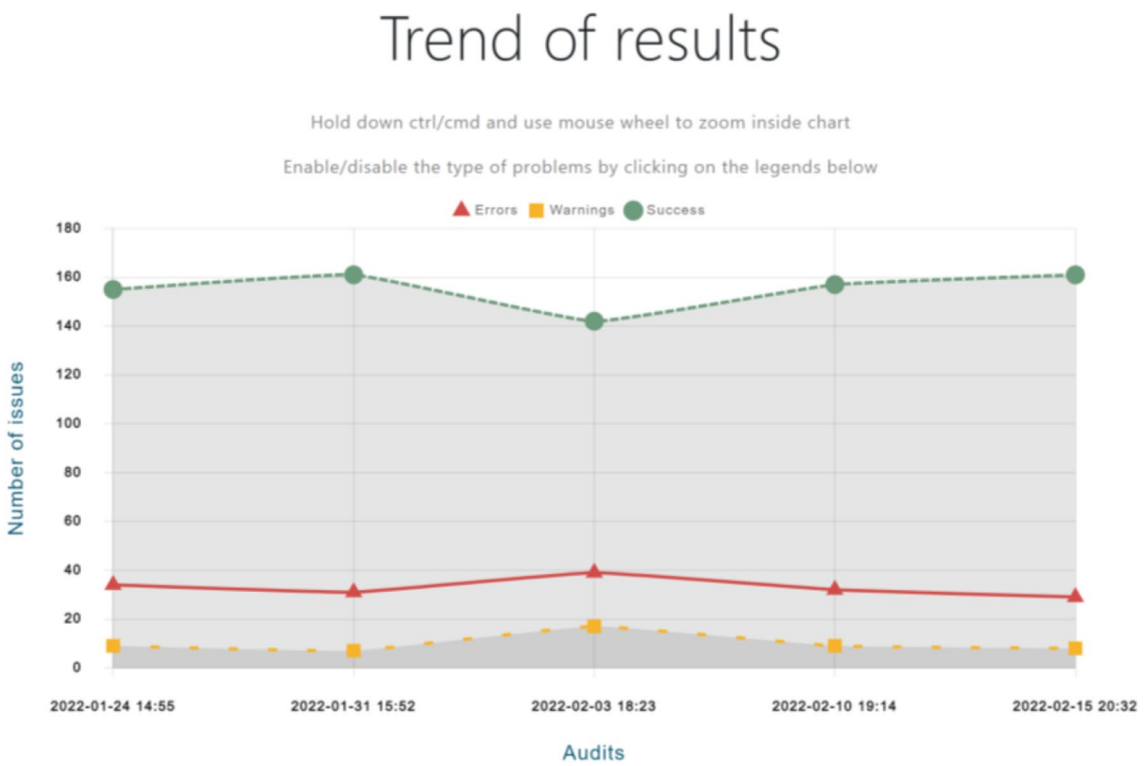

Fig. 7 Trend of results diagram for monitoring a project

in Fig. 5). In the left-hand panel, by clicking the code line link, it is also possible to automatically move to the Web Developer view and see the corresponding code line which caused the issue.

The Web Developer View (Fig. 6) presents the HTML code listing, highlighting in different colours the rows that have been affected by accessibility issues: red for errors and yellow for warnings. The rows that are affected by accessibility issues are interactive: by selecting one of them, the user is redirected to the corresponding W3C page referring to the involved WCAG technique. In the right-bottom part of the page, there is a small interactive arrow linking directly to the top of the page (for easier access to the various panels).

Figure 2 shows the top-most part of the page dedicated to the information about each project, namely the one providing summary information on that project. However, in that page also additional pieces of more detailed information are provided, namely:

- **Trend of results (R3)**: This chart supports monitoring over time the accessibility of a page or a group of pages included in a project, by displaying a line chart (see Fig. 7) with three lines indicating the number of validations performed, with results shown, respectively, as errors, warnings and successes (Y axis) for each of the audits performed over time (the X axis indicates the dates of the audits). It is also possible to interactively exclude one or more lines by acting on the respective legends, and also to zoom in/out the graph;
- **Accessibility percentage and evaluation completeness:** A bar chart in which two bars are shown for each audit, one for each metric. In a way similar to the previous graph, the user can exclude some data from the graph, and also zoom in/out.

4 Tool architecture

We aim to a solution that is open, supports standard semantic interpretations of the accessibility rules, extensible with limited effort to new guidelines, flexible in defining what to validate (single pages, groups of selected pages, entire web sites) and in reporting results tailored for different relevant roles, and able to support validation according to the hierarchy of accessibility requirements (principles, guidelines, success criteria, techniques) and for different types of devices. In addition, beyond the possibility to carry out a validation just once, the tool offers the possibility to monitor the accessibility of a web site over time, which means automatically scheduling and running accessibility evaluations of a Web site (i.e. audits) at specific times, so that users can follow its evolution over time. To support this, we

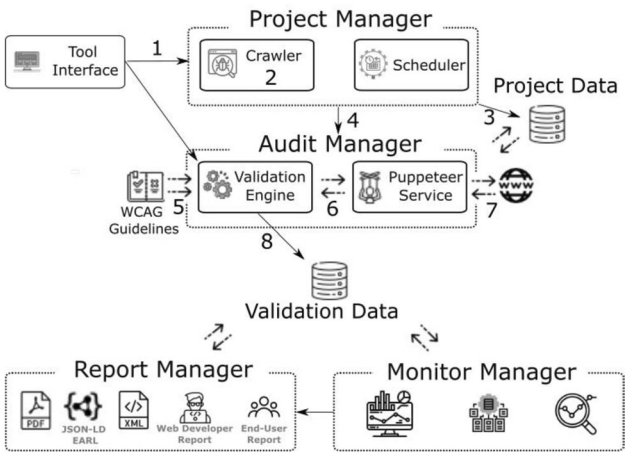

Fig. 8 The Tool Architecture

introduced the possibility to create "projects" in the tool: a project groups together the various audits conducted at different times on a specific group of pages, where each audit corresponds to the evaluation of the web pages belonging to that project at a specific time.

Figure 8 describes the architecture of the new tool to support accessibility evaluation and monitoring, also using server-side rendering. It is mainly composed of:

- a *Project Manager*, which discovers the actual pages of interest for the validation (using a Crawler) according to the parameters specified by the user, and, through the Scheduler, allows for planning future audits in such a way that they will automatically run according to the information specified by the user at project's creation time;
- an *Audit Manager*, which carries out the actual validation of the selected pages according to a specific set of guidelines (e.g. WCAG 2.1), also managing the validation when such pages have dynamic content;
- a *Monitor Manager:* this module takes the results associated with the audits belonging to a project, aggregates them and provides interactive representations of such information within the tool, to allow the user to analyse such results and have an overview of the evolution of accessibility of the considered web pages.
- a *Report Manager*, which generates validation results in various formats (PDF, Web, EARL), aimed to various types of users.

By using the tool's Web interface, users can specify some parameters (1) that will be used to create different types of projects (i.e. simplified or in-depth). In the case of a "simplified" project, the pages included in the project are those belonging to a specific Web site and identified by a crawler (2) starting from a base URL specified by the user at the project's creation time. It is worth noting that in the case of an in-depth project the user will specifically indicate the pages that should be involved in the validation. The Project Data such as the crawler configuration and the evaluation settings (WCAG version, target device and user agent, etc.) are stored in a database (3) and are used by the Project Manager and the Audit Manager modules.

To implement the crawler, we exploited crawler4j[8], an open-source multi-thread Web crawler for Java that provides a simple interface for crawling the Web. The crawler needs a base URL that is used as a seed: this is the first page fetched, and from such URL, the crawler will start following the links discovered within the page and iteratively analyse the newly discovered pages. The crawler takes as input two main configuration parameters provided by users during the project creation: Crawler depth and Maximum Number of Pages that should be fetched. The depth of crawling describes the extent to which this module discovers the target pages. For example, if we have the seed page "A", which links to page "B", which links to page "C", then we obtain the following link structure: A—>B—>C—>[…]; since "A" is a seed page, it will have a depth of 0, "B" will have a depth of 1 and so on.

The crawler implements a "Should Visit" policy to specify whether a given URL should be crawled or not; we configured this policy to ignore URLs associated with CSS, JavaScript code, images, videos, PDF and all the content types that are not immediately relevant for the validation. Moreover, we configured the crawler in such a way that it should not follow URLs belonging to domains different from the one specified in the seed URL.

The URLs of the pages identified by the Crawler (according to the Crawler depth and Maximum Number of Pages parameters) at project's creation time are then saved in a database (3), and afterwards they are provided to the Audit Manager, which will actually evaluate them (4). As we will see later on in this section, since the actual pages associated with these URLs might depend on the execution of some JavaScript code, another module (Puppeteer) actually retrieves the HTML code currently associated with these URLs in order to perform a specific audit.

In order to support the functionality of monitoring the accessibility over time, we developed a module called Scheduler. The Scheduler is responsible for scheduling a new evaluation of the discovered pages in the time interval specified at project creation time (e.g. each Monday every week). The Validation Engine is part of the Audit Manager: it takes as input (5) the WCAG Guidelines specification defined in an XML-based language, and it exploits an HTML&CSS parser[9] to select all the page elements considered in the guidelines, and finally it verifies whether such

[8] https://github.com/yasserg/crawler4j

[9] https://github.com/radkovo/jStyleParser

elements respect the accessibility checks. In this regard, the tool has been designed in such a way to facilitate its update in case of changes in the relevant guidelines: indeed, the checking of the guidelines is not hardcoded in the tool, but it is performed by the Validation engine, which is able to check any guidelines written in an XML-based language (developed previously). The HTML&CSS parser exploits the parameter (provided by the user) specifying the target device, which defines different viewport dimensions associated with each device type. Starting from such viewport dimensions, the parser then applies the eligible media queries to the DOM elements of the Web page, to identify what content is relevant for the current validation. Thus, the accessibility evaluation for a desktop version can provide a different result from a mobile version evaluation of the same Web site.

The Validation Engine evaluates the dynamic content of Web pages by exploiting the functionalities provided by the Puppeteer Service (6), which implements the Server-Side Rendering-based validation. The tool allows the user to select between two types of validation: static validation, and server-side rendering validation. In the Static Web Page validation, the tool downloads the HTML and the CSS code of the page and then parses and validates the corresponding DOM. Instead, using the Server-Side Rendering validation, the engine does not parse just the static Web page code: indeed, it exploits the Puppeteer library (https://developers.google.com/web/tools/puppeteer) to load the HTML (and CSS) code (7) within a headless version of the Chrome browser, to simulate the page being opened, executed and rendered within a browser. In this way, the validation tool can have access to a more complete page (also populated with the results of executing the JavaScript code included in the page) and not just the original static version. The Server-Side Rendering Validation is always used by the tool in case of validations of projects (which typically imply validating one or more pages over time). Instead, when users select a single page validation (i.e. the validation of a single page done just once), they can decide the type of validation to use (either static or based on server-side rendering). While the static one is faster, the server-side rendering validation can be more complete, but it might require more time for its execution. We further extended the Validation Engine to be fully compliant with the EARL outcome specification[10] by adding the ability to calculate the WCAG techniques that have not been applied because there are no elements that are relevant for their application within the considered page.

The Monitor Manager provides users with interactive representations of the validation results (i.e. trend of results, metrics, most frequent errors). Finally, there is a Report Manager which gives the accessibility results in form of reports provided through different formats (8): through some web interfaces (e.g. the Live Preview for the end-user, the "Web developer" view), but also by providing some downloadable PDF, EARL and XML-based reports.

5 Usability study

5.1 Organisation and participants of the test

In order to gather user feedback on the tool and especially about the recently introduced features associated with the monitoring and transparency support, a usability test was carried out in which the users had to perform some tasks using the tool, and also answer related questions. We preferred to submit a specific ad-hoc questionnaire rather than a standard usability evaluation scale, to have more focused feedback from users on the usability of the tool.

We recruited 15 users (8 females; average age: 41.6; standard deviation: 12.8), by sending an email in the research area in which our research institute belongs, inviting people with specific competencies/skills to participate in the test (i.e. being or having been a web developer or a web commissioner). To describe their profile, users were allowed to specify one or more options. In the end, eleven users selected "web developer" as one of their main profiles, five users selected "web commissioner", two selected "accessibility expert", three users selected other profiles such as UI designer or researcher. They were asked whether they have used some validation tools before the test: 8 users replied "Yes", mentioning Siteimprove (3 users), Wave (3 users), Mauve (2 users), W3C tools (2 users), Google Lighthouse (1 user); the other 7 answered "No". We also asked those who had already used some tools why they chose that tool, what the most appreciated functionalities were, and whether there was something that was missing or needed further improvements. In general, the choice of the tool was driven by convenience aspects when there was the need for an accessibility validation, and they chose those tools that were most easily found. For those who used Wave, the most appreciated features were related to the display of the reports in the form of easily understood icons that can be clicked directly on the page examined, triggering the display of the corresponding snippet of code, and an explanation of the detected problem. In the Siteimprove plugin the most appreciated functionalities were related to the possibility to filter the results based on various aspects, such as their severities. In terms of functionalities, nothing particularly novel was reported for Lighthouse. Regarding useful improvements, for WAVE the users mentioned the possibility to completely analyse the page code, to support analysis of dynamic web pages, and collect evaluation results over time. For the

[10] https://www.w3.org/TR/EARL10-Schema/#OutcomeValue

Siteimprove plugin they would like to have a better overview of the errors identified, the possibility to collect and compare results of evaluations over time, to show the errors also directly in the Web page code, and to immediately highlight them also in the page user interface. Similar features were also suggested for the Lighthouse tool.

The test was carried out remotely with the support of a videoconference system, and each test session was video-recorded with users' permission collected in an informed consent that users had to fill in and sign before the test. Some days before the test, users also received relevant documentation introducing the main aspects of the WCAG guidelines, and of the tool.

Just before starting the actual test, the moderator briefly introduced its goal. Then, the user received the link of the tool and the credentials to use for the test (all the users used the same account), and the link to a Web page which contained the tasks, alternating them with the related usability questions. In this way, users could provide their feedback on a specific feature just after having tried it in the tool. To better follow the test, the participants were asked to share the browser's window where the tool was visualised with the moderator. Since the users had to fill in the questionnaire while running the test, to prevent any influence on their responses, at the beginning the participants were asked to open the questionnaire page on a browser window different from the one that was shared with the moderator. The duration of each test session ranged between half an hour and 1 h.

5.2 Tasks and questionnaire

The tasks of the test were identified in such a way to cover the identified requirements. In the following, for each task we provide its description, also indicating the associated requirement (R1-R8). However, for requirement R8, although the tool supports it (i.e. the tool is able to support the accessibility validation of dynamic web pages), we have not gathered specific information through the test (by introducing specific tasks or questions) in order not to prolong it too much.

5.2.1 Task 1. Single page evaluation—analysis of the various views provided by the tool (R2, R7)

The first task required carrying out a "Single web page evaluation" of a Wikipedia page (https://it.wikipedia.org/wiki/Pagina_principale), using WCAG 2.1 guidelines, Level of Conformance = AAA, server-side rendering validation, and selecting "Desktop—Windows – Edge" as the device/user agent considered for the evaluation. Then, the participants had to start the validation using the tool, get an overview of the content of the four views provided by it ("Evaluation Summary", "End User", "Live preview" and "Web developer"), and answer the following related questions:

- Q1: By analysing the results, how many distinct types of problems categorised as "error" has the tool identified?
- Q2: Within the "error" category problems, which type of problem occurs most frequently? How many times does it occur?
- Q3: Did you encounter any difficulty in understanding one or more pieces of information included in the views provided by the tool ("Evaluation Summary", "End User", "Live preview" and "Web developer")? If so, what did you find difficult to understand or unclear, and why?

5.2.2 Task 2. Analysis of the live preview (R1, R4, R7)

After completing the previous task, in the "Live preview" panel, users were asked to filter the results only to the CSS errors affecting the "perceivable" principle (thereby the parameters to set were: issue type = " errors", code type = "CSS", principle = "Perceivable"). Next, in the "Error List" section, they had to identify the errors associated with the violation of Criterion 1.4.10, SCR34 technique, analyse its first 5 occurrences, also paying attention to how the tool presents the results when the element affected by the error is or is not visible in the page. Then, they were asked to rate their agreement with the following statements using a 1–5 Likert scale (1 = I strongly disagree, 5 = I strongly agree):

- S1: When it was possible to view the error within the Web page, the view offered to the user is useful for locating the identified problem;
- S2: When the tool was able to highlight the error within the Web page, the tool also provides some information to solve the error. The view offered to the user is useful to solve the identified problem;
- S3: When the occurrence of the error is not visible within the page, the tool provides a link to the "Web developer" view for the analysis of the source code. It is useful to be able to analyse the error directly in the source code.

Then, they had to ask the question:

- Q4: Would you have any suggestions on how to improve the Live Preview, or what to change to improve it?

5.2.3 Task 3. Creation of an in-depth Project (R2)

The users had to create a new project of type "In depth", by configuring it with the following URLs (referring to a Italian University web site) and validation parameters:

5.2.4 URLs

- Homepage: https://www.unipi.it/
- Login: https://unipi.idp.cineca.it/idp/profile/SAML2/Redirect/SSO?execution=e2s1
- Forms: https://unimap.unipi.it/cercapersone/cercapersone.php
- Accessibility Statement: https://www.unipi.it/index.php/documenti-ateneo/item/14764

5.2.5 Parameters

The guidelines to select are WCAG 2.1, the Level of Conformance is AA, the audit should be repeated just once, and it should be planned for the current day, also selecting "Desktop—Windows – Chrome" for the user agent/device.

After having created the project, an audit automatically started in the tool (as associated with this newly created project): this was because it was planned for the current day (as per the specified parameters). The users had then to answer this question:

- Q5: Have you experienced particular difficulties in creating a project or in understanding the parameters to specify? If so, please indicate where.

5.2.6 Task 4. Analysis of a Single Audit (R2, R6)

Users were asked to select the newly created project and, within it, open the audit just started and analyse the information provided by it (within the "Results" and "Page Results" sections): in particular, they could analyse the details of the evaluation results associated with the various pages associated with the project using the links in the "List of Evaluated pages" section. Then, they had to rate their agreement with the following statements (using a 1–5 Likert scale, 1 = I strongly disagree, 5 = I strongly agree):

- S4: The views presented by the tool and associated with an audit are clear and understandable;

Then, they had to answer the following open-ended question:

- Q6: Would you have suggestions on what to change/how to improve the information associated with each audit?

5.2.7 Task 5. Analysis of information associated with a project, including its monitoring (R3, R5)

Differently from the previous task in which users had to assess the information provided by a single evaluation (audit) over a specific set of pages, in this task users had to assess the information that the tool provides after evaluating a set of pages over time (i.e. by carrying out audits in different times). Thus, in this task the users had to mainly use the tool features supporting monitoring activities. To allow users to perform this task during the test, in the tool's workspace of the participant's account, we made available a project containing data that simulated various audits repeated in different times of the year on a specific group of web pages. Users had to open and analyse this project, explore the various sections of the page ("Trend of results" / "Accessibility percentage and evaluation completeness" / "Most frequent errors") and they have to answer the following questions. In particular, Q8 was relevant for understanding the usability of the representation provided to support accessibility monitoring:

- Q7: In which audit was the Evaluation Completeness highest?
- Q8: Which audit had the least number of successes?
- Q9: What is the most frequent type of error found in the last audit?
- Q10: If you have experienced any difficulty in understanding one or more result included in the various views associated with the project, or in interacting with one of the graphs displayed, could you specify in what situation?
- Q11: Do you have any suggestions on what you would change to improve the info associated with each project?

They had also to rate their agreement level to this statement:

- S5: The views presented in the tool and associated with a project are clear and understandable

After this task, the questionnaire presented a number of final, open-ended questions for evaluating the overall experience of using the tool. The questions were:

- Q12: What are the three aspects of the tool you liked the most?
- Q13: What are the three aspects of the tool you liked the least?
- Q14: In general, do you think that that the tool clearly gives all the information necessary to get an overview of the accessibility problems of a site, to be able to monitor its accessibility over time?

- Q15: Is there any information that you think would be useful but that you have not found in the tool?
- Q16: Do you have any further suggestion for improving the tool?

6 Results

In this section we report the users' responses to the questions indicated above.

6.1 Single page evaluation—analysis of the various views provided by the tool (R2, R7)

For question Q1, users had to indicate how many types of problems the tool identified within the "error" category: the vast majority of them (13 out of 15) correctly replied to this question. For question Q2, the users had to indicate, within the "error" category, which type of problem occurred most frequently: 11 (out of 15) correctly answered. Some of those who provided an incorrect answer just misunderstood the question statement, which included the "category" word: as a consequence, they looked in the tool where the errors are grouped "by category". Still in Q2, users had to indicate how many times the most recurrent problem occurred. Nine users correctly replied, three did not provide an answer, the remaining three provided a wrong one (which was connected to the incorrect answer provided just before).

For Q3, ten users explicitly reported that the information included in the various views was clear, and they had no particular difficulty in understanding it. As a suggestion, one of them recommended giving more visibility and structure to the left-hand panel of the "Live Preview", also expressing appreciation for the fact that in the "Web developer" view the identification and categorisation of the issues (i.e. in warnings and errors) was easy to follow, thanks to the different colours used. The remaining five reported the following comments (some provided more than one suggestion). Two reported difficulties in understanding the connections between the information provided in the "Evaluation Summary" (which shows the issues categorised in terms of techniques that resulted erroneous, successful, warnings, and not applicable) and the information available in the "End user" tab, which details the number of occurrences (categorised according to principles) for each technique of type "warning" or "error". In particular, one of them suggested adding further explanation about how to read these connected views. Two users suggested providing the possibility of ordering the column showing the number of occurrences within the "End User" view, to have more readily available the information about the most recurring error/warning. One participant suggested changing the current representation (which is a pie chart) used to show the occurrences of errors/warnings grouped by principles, because in some cases the slices may be difficult to perceive. A user reported some difficulties with the (standard W3C) acronyms (i.e. "SCR34", "G212") used in the tool to identify the various techniques/success criteria.

Next (Q4), the participants had to provide suggestions on how to improve the Live Preview, or about what to change to improve it. Six users judged the view as being already very comprehensible and useful. As for the others: i) three commented about the part supporting filtering: one suggested distinguishing more clearly the filtering part from the one presenting the results, also using different colours. Two suggested adding a button to explicitly support applying the filtering criteria (currently they are automatically applied whenever the user changes them); ii) five users gave suggestions about the type of visualisation provided by the tool: one suggested changing the eye-shaped icon of the button supporting the localisation of the selected error within the Web page when the issue cannot be visualised in the page, by using a more intuitive one (e.g. through a strikethrough eye-based icon, to highlight that the error cannot be seen); two users said that, when an issue is not visible, the link to the "Web developer" view was judged as not particularly evident; another user did not find it intuitive to use mouse-over to trigger the appearance of the tip. Another user did not realise that the eye-shaped icon is interactive.

6.2 Creation of an in-depth project (R2)

For the next question (Q5), ten users declared that there was no particular difficulty in creating a project. One user just suggested better explaining the various sections. Five users provided additional remarks to better highlight the various parts in the project creation form, also indicating the mandatory fields within it.

6.3 Analysis of a single audit (R2, R6)

For Q6, eight users declared not having any further suggestions to improve the information associated with each audit. One user suggested adding another graph visualising the number of erroneous techniques, possibly superimposed on the graph visualising the number of errors in the Page Results section (which visualises the number of error/warnings occurrences found in the page using a bar chart). The same user suggested better explaining some expressions (e.g. the "Average number of techniques" in the "Results" section did not clearly indicate which elements the average was calculated on). One user suggested making the graphs even more interactive (e.g. further details could be shown after clicking on a bar in the "Page Results" bar chart).

Table 1 Summary of *min, max, median, mean and st. dev. of Likert scale ratings associated with statements S1-S5*

	Min	Max	Median	Mean	St. Dev
S1	4	5	4	4.5	0.5
S2	3	5	4	4.4	0.6
S3	4	5	5	4.5	0.5
S4	2	5	4	4.1	0.7
S5	3	5	4	4.1	0.6

6.4 Analysis of information associated with a project (R3, R5)

Users also had to indicate some results provided by the tool on the evaluation of multiple pages (corresponding to a "project") over multiple audits. This was done to understand whether such information was actually comprehensible. In particular, 14 users replied correctly to Q7 (in which audit was the Evaluation Completeness highest?), 1 incorrectly; all the users correctly answered Q8 (the audit with the least number of successes); 14 users replied correctly to Q9 (most frequent type of error in the last audit), 1 user incorrectly. So, we can infer that the key information associated with a project is generally understandable by users.

Furthermore, five users explicitly stated not having had any difficulty with the info associated with each project (Q10), while the others took this opportunity to suggest further improvements. Three users expressed concerns about the "Most frequent errors" section. In particular, they judged that the way such information was described could be improved. Indeed, to indicate respectively the most frequent/the second-most frequent/the third-most frequent type of error, we used the (probably too compact) expressions "first/second/third", which resulted a bit ambiguous for some of them (e.g.. one user thought it referred to the audits). Two users reported some slight difficulties in identifying the lowest number of successes. As for the line chart "Trend of results", one user suggested deleting the greyish area under the line and keeping just the lines; four users suggested slightly increasing the size of the circles visualised in that graph; another user suggested replacing this graph with a histogram, to further enhance its clarity.

As for Q11, nine users explicitly said that they had no particular suggestion to improve the information associated with each project. One suggested deleting the (decorative) icons currently shown under the title "List of Evaluated pages" as they do not completely reflect the actual information that is shown in that part of the page, and to use consistently the colours that are shared by other graphs (e.g. the orange colour is generally used in the tool to indicate a warning). In the section, "Accessibility percentage and evaluation completeness" one user suggested having just two series of bars (one for each of the metrics considered: accessibility percentage and evaluation completeness) instead of two bars for each audit (one bar for the first metric and one for the second). Even though currently it is possible to hide a bar series associated with a specific metric, the user interestingly pointed out that it could be useful to have a comparison between the bars referring to each metric in a more immediate manner in some cases. This would suggest that, to allow an effective use in a diverse range of situations, accessibility validation tools should support as much flexibility as possible.

6.5 User ratings

Below we report a table (see Table 1) in which we summarise the main descriptive statistics metrics for the Likert scale ratings associated with statements S1-S5.

While this table shows generally positive user appreciation of the tool, the minimum value (2) corresponds to S4 ("The views presented by the tool and associated with an audit are clear and understandable"), which was provided by an accessibility expert, who suggested using different types of visualisations (e.g. histograms) for highlighting the errors and warnings detected in each page whereas currently the tool shows a line chart ("Trend of Results" chart).

6.6 Overall experience evaluation

Finally, in the last part of the questionnaire, we asked users to provide some more general feedback about the overall experience. Among the three things they liked the most about the tool (Q12), four said that the information provided is well/clearly structured. The possibility to perform multiple audits and see the trends over time was mentioned also four times. The clearness/interactiveness of graphs was mentioned four times. Three users liked the possibility to see the issues in the rendered page and in the code, while three users liked the graphical style of the tool. Three users mentioned the clarity of the tool, two its intuitiveness, two appreciated that the tool provides objective results and in a detailed manner; two the fact that the tool provides results promptly; two liked the easy connection with related W3C documentation to help users solve an issue. Additional aspects were: the completeness of the information, the interactivity of the tool, the fact that it allows for analysing different aspects of accessibility, the possibility for even a non-expert user to carry out an analysis. They also mentioned the easy to understand results, the Web developer view, the Live Preview, the possibility to assess a single page or go deeper to assess an entire Web site, the versatility/flexibility of the tool, and the fact that it is responsive. One user also highlighted that the tool provides the user with a clear workflow to create a new project.

We also asked about the three things they liked least about the tool (Q13). Four users declared not being able to identify any aspect in this regard. Three users mentioned some issues related to the legends associated with some of the charts: while they appreciated the possibility of customising a part of a chart visualisation to better focus on specific aspects of interest (e.g. hiding a portion of the chart by interacting with the respective legend), they reported some usability issues with the way in which it was currently supported. For instance, one user complained about having to select a legend to hide the related part in a graph, another user found the legends too small (compared to the size of the graph), another suggested enhancing the contrast of labels used besides some legends (to be more visible), also noting that some labels are too spaced apart. Two users said that some functionalities are not easily visible, mentioning the link supporting the visualisation of errors within the "Web Developer" view; the same applies to the "View project"/"View audit" links. Two users mentioned adding further information to the various sections in the tool, as it is not immediately clear which aspects they refer to.

All the users replied affirmatively to question Q14 (overview of the accessibility problems of a site, to be able to monitor its accessibility over time). While six users did not provide any further detail, five further highlighted some aspects they found important: the possibility to create projects is best suited to the kind of analysis supported, the fact that it is a tool usable by both web developers and non-web developers, it provides further explanations when needed, it offers a list of errors ordered by some priority. One user highlighted that, while it is certainly a useful tool, sometimes it uses a language that can result too specialised.

Finally (Q15-Q16), we asked whether, in user's opinion, there was any useful information that they have not found in the tool or whether they had any suggestions to improve the tool. All declared not being able to identify any further information that the tool currently does not provide. One user mentioned that it could add some information about the Web page loading time. Another mentioned that the tool could provide some information about non-working links.

7 Discussion

The results gathered through this test were overall encouraging. First, since the participants of the test had different kinds of profiles (both technical and non-technical) and were generally able to complete the various tasks in a satisfactorily manner, this result would confirm that the tool can easily support various types of stakeholders, which was one of our main requirements (R1). Also, by analysing the results gathered in association with Task1 (which dealt with a single page evaluation), Task 3 (creation of a project), and Task 4 (analysis of a single audit), overall it seems that the users satisfactorily used the tool for conducting both the evaluation of a single page and that of multiple pages (also one of the requirements, see R2). Thus, the tool seems to be able to support both the needs of those who have a specific focus on a particular web page for their own goals (e.g. a personal home page), but also the needs of organisations that require a more comprehensive, site-wide evaluation of the accessibility of their web applications. The answers provided to Q7-Q11 and the level of user's agreement to statement S5 also show that, while the tool can be improved in describing the information it provides, it is overall able to offer to its users key information to monitor how the accessibility of a site/group of pages can vary over time both in a detailed manner (i.e. in terms of number of errors/warning detected in different audits) and in terms of more general metrics that summarise the level of accessibility of a site over time (see requirements R3 and R5), as well as addressing the needs of different types of stakeholders (e.g. a summative metric can likely be of more interest for non-technical users). According to the data gathered in the ratings to statements S1, S2, S3, and the answers to question Q4, while also considering more general comments, the participants seem to appreciate the level of support that the tool offers for both localising (requirement R4) and solving (requirement R7) the errors identified during an accessibility evaluation, especially through the provided validation results views (e.g. Live Preview, Web Developer View) able to target different types of stakeholders, as well as the provision of direct link/reference to the relevant W3C documentation. However, further work should be done to provide more specific indications about how to solve some of the issues that the tool detects (even by supporting user's manual intervention).

To sum up, the participants seemed to be quite satisfied with the tool, which is particularly encouraging especially considering that the vast majority of them used it for the first time, none had used the new functionalities (e.g. the monitoring support) previously, and no familiarisation phase with the tool was carried out before the test. On the one hand, they especially appreciated the fact that it allows monitoring the evolution of different audits over time and found clear the information provided, including the graphs; they also liked that the tool provides objective results, and in a prompt and detailed manner. On the other hand, some of them suggested improving aspects connected with the clarity of the tool (e.g. improve some legends using a less specialised language), make more visible some of the provided features, and add further interactivity to the charts proposed.

8 Conclusions and future work

In this paper, we have presented a set of requirements that should characterise automatic validation tools able to address the emerging needs derived from the increasing demand for accessible Web sites and the evolution of the Web technologies. We have shown how such design dimensions can be supported and implemented in a tool, which is thus able to provide users with monitoring functionalities, clear indications of its capabilities, and validation of dynamic content. We have also reported on a first user test that provided positive feedback and suggestions for small adjustments, such as in the choice of the graphs for visualizing the validation results.

The tool is available for open access, and future work will be dedicated to modifying it by taking into consideration the suggestions provided by users, adding functionalities that will allow accessibility experts to integrate their manual validations with those automatically generated (to decide the correctness of the elements that cannot be validated automatically), and produce accessibility reports focused on specific disabilities or application areas.

References

1. Gulliksen, J., Von Axelson, H., Persson, H., Göransson, H.: Accessibility and public policy in Sweden. Interactions **17**(3), 26–29 (2010)
2. Lazar, J., & Olalere, A. (2011). Investigation of best practices for maintaining section 508 Compliance in US federal Web sites.In: International conference on universal access in human-computer interaction pp. 498–506. Berlin: Springer.
3. Paternò, F., Schiavone, A.: The role of tool support in public policies and accessibility. ACM Interact **22**(3), 60–63 (2015)
4. EU Commission. (2016, October 26). Directive (EU) 2016/2102 of the European Parliament and of the Council. Retrieved from https://eur-lex.europa.eu: https://eur-lex.europa.eu/eli/dir/2016/2102/oj
5. Power, C., Freire, A., Petrie, H., & Swallow, D. (2012). Guidelines are only half of the story: accessibility problems encountered by blind users on the Web. In: Proceedings of the SIGCHI conference on human factors in computing systems pp. 433–442. ACM.
6. Beirekdar, A., Vanderdonckt, J., Noirhomme-Fraiture, M.: Kwaresmi–Knowledge-based Web Automated Evaluation with REconfigurable guidelineS optimisation. (Springer, Ed) DSV-IS **2545**, 362–376 (2002)
7. Beirekdar A., Keita M., Noirhomme M., Randolet F., Vanderdonckt J., Mariage C. (2005) Flexible Reporting for Automated Usability and Accessibility Evaluation of Web Sites. In: Costabile M.F., Paternò F (eds) Human-Computer Interaction - INTERACT 2005, INTERACT 2005, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg vol. 3585.
8. Fernandes, N., Kaklanis, N., Votis, K., Tzovaras, D., & Carriço, L. (2014). An analysis of spersonalised Web accessibility. In: Proceedings of the 11th web for all conference pp. 19. ACM.
9. Fuertes, J. L., González, R., Gutiérrez, E., & Martínez, L. (2009). Hera-FFX: a Firefox add-on for semi-automatic Web accessibility evaluation. In: Proceedings of the 2009 International cross-disciplinary conference on web accessibility (W4A) pp. 26–35. ACM.
10. Ivory, M.Y., Mankoff, J., Le, A.: Using automated tools to improve Web site usage by users with diverse abilities. Hum-Comput Interact Inst **2003**, 117 (2003)
11. Nietzio, A., Eibegger, M., Goodwin, M., & Snaprud, M. (2011). Towards a score function for WCAG 2.0 benchmarking.In: Proceedings of W3C online symposium on website accessibility metrics. Retrieved from https://www.w3.org/WAI/RD/2011/metrics/paper11
12. Schiavone, A., Paternò, F.: An extensible environment for guideline-based accessibility evaluation of dynamic Web applications. Univers Access Inf Soc, Springer Verlag **14**(1), 111–132 (2015)
13. Gay, G, and Qi Li, C. 2010. AChecker: open, interactive, customisable, Web accessibility checking. In: Proceedings of the 2010 International cross disciplinary conference on web accessibility (W4A). 1–2.
14. Mirri, S., Muratori, L. A., & Salomoni, P. (2011). Monitoring accessibility: large scale evaluations at a geo political level. In: The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility pp. 163–170. New York: ACM.
15. Miniukovich, A., Scaltritti, M., Sulpizio, S., and De Angeli, A. 2019. Guideline-Based Evaluation of Web Readability. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). Association for Computing Machinery, New York, NY, USA, Paper 508, pp. 1–12. DOI:https://doi.org/10.1145/3290605.3300738
16. Moreno, L., Alarcon, R., Segura-Bedmar, I., and Martínez, P. 2019. Lexical simplification approach to support the accessibility guidelines. In: Proceedings of the XX international conference on human computer interaction (interaccion '19). association for computing machinery, New York, NY, USA, Article 14, pp. 1–4. DOI:https://doi.org/10.1145/3335595.3335651
17. Abascal, J., Argue, M., Valencia, X.: Tools for Web accessibility evaluation, pp. 479–503. Springer, Web Accessibility, London (2019)
18. Manca, M., Palumbo, V., Paternò, F., Santoro, C.: The Transparency of Automatic Web Accessibility Evaluation Tools: Design Criteria, State of the Art, and User Perception. ACM Press, ACM Transaction on Accessible Computing (2022)
19. Broccia, B., Manca, M., Paternò, F., Pulina, F.: Flexible Automatic Support for Web Accessibility Validation. Proc ACM Hum Comput Interact EICS **83**(1–83), 24 (2020)
20. Pelzetter, J.: A Declarative Model for Web Accessibility Requirements and its Implementation. Frontiers Comput. Sci. **3**, 605772 (2021)
21. Vigo, M., Brown, J., and Conway, V. 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. pp. 1–10.

22. Burkard, A., Zimmermann, G., Schwarzer, B.: Monitoring systems for checking websites on accessibility. Front Comput Sci **3**(2021), 2 (2021)
23. Abduganiev S.G.: Towards automated web accessibility evaluation: a comparative study. Int J Inf Technol Comput Sci (IJITCS) **9**(9), 18–44 (2017)
24. Pădure, M., and Pribeanu, C. Exploring the differences between five accessibility evaluation tools (2019)
25. Frazão, T., and Duarte, C. Comparing accessibility evaluation plug-ins. In: Proceedings of the 17th International web for all conference (W4A '20). association for computing machinery, New York, NY, USA, Article 20, 1–11 (2020). https://doi.org/10.1145/3371300.3383346
26. Yesilada, Y., Brajnik, G., Vigo, M., Harper, S.: Exploring perceptions of Web accessibility: a survey approach. Behav Inf Technol **34**(2), 119–134 (2015)
27. Brajnik, G., Yesilada, Y., Harper, S.: Is accessibility conformance an elusive property? A study of validity and reliability of WCAG 2.0. ACM Trans Access Comput (TACCESS) **4**(2), 1–28 (2012)
28. Molinero, A.M., Kohun, F.G., Morris, R.: Reliability in Automated evaluation tools for web accessibility standards compliance. Issues Inf Syst **7**(2), 218–222 (2006)
29. Petrie, H., King, N., Velasco, C., Gappa, H., Nordbrock, G. (2007): The usability of accessibility evaluation tools, In: Stephanidis, C. ed UAHCI 2007, LNCS, vol. 4556, pp. 124–132. Springer Heidelberg, https://doi.org/10.1007/978-3-540-73283-9_15
30. Salehnamadi, N., Alshayban, A., Lin, JW, Ahmed, I., Branham, S., and Malek, S. 2021. Latte: Use-Case and Assistive-Service Driven Automated Accessibility Testing Framework for Android. In: CHI Conference on Human Factors in Computing Systems (CHI '21), May 8– 13, 2021, Yokohama, Japan. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3411764.3445455
31. Paternò F., Pulina F., Santoro C., Gappa H., Mohamad Y. (2020) Requirements for Large Scale Web Accessibility Evaluation. In: Miesenberger K., Manduchi R., Covarrubias Rodriguez M., Peňáz P. (eds) Computers Helping People with Special Needs. ICCHP 2020. Lecture Notes in Computer Science, vol. 12376. Springer, Cham. https://doi.org/10.1007/978-3-030-58796-3_33
32. Siteimprove, QualWeb. https://chrome.google.com/webstore/detail/siteimprove-accessibility/djcglbmbegflehmbfleechkjhmedcopn, http://qualweb.di.fc.ul.pt/evaluator/. Accessed 6 Sept 2022
33. Achecker, https://achecker.achecks.ca/checker/index.php. Accessed 6 Sept 2022
34. Experte, https://www.experte.com/accessibility. Accessed 6 Sept 2022
35. IBM Accessibility Checker, https://addons.mozilla.org/en-US/firefox/addon/accessibility-checker/. Accessed 6 Sept 2022
36. Lighthouse, https://web.dev/lighthouse-accessibility/. Accessed 6 Sept 2022
37. Shadi Abou-Zahra. Evaluation and Report Language (EARL) (2017). Retrieved February 2, 2017 from https://www.w3.org/TR/EARL10-Schema/#OutcomeValue