Ability Heuristics for Conducting Accessibility Inspections

Claire L. Mitchell
The Information School
University of Washington
Seattle, Washington, USA
clairelm@uw.edu

Junhan Kong
The Information School
University of Washington
Seattle, Washington, USA
junhank@uw.edu

Jesse J. Martinez
Paul G. Allen School of Computer Science & Engineering
University of Washington
Seattle, Washington, USA
jessejm@cs.washington.edu

Shaun K. Kane
Google Research
Boulder, Colorado, USA
shaunkane@google.com

Amy J. Ko
The Information School
University of Washington
Seattle, Washington, USA
ajko@uw.edu

Alexis Hiniker
The Information School
University of Washington
Seattle, Washington, USA
alexisr@uw.edu

Jacob O. Wobbrock
The Information School
University of Washington
Seattle, Washington, USA
wobbrock@uw.edu

Abstract

The accessibility of interactive technologies is often evaluated using checklists that are low-level, numerous, and platform-specific. Such checklists are typically used by accessibility experts, leaving everyday designers and developers with little support for assessing their own interfaces. To make accessibility evaluations easier to conduct, we devised a set of nine "ability heuristics" that prompt designers to engage with accessibility throughout the design process. We empirically evaluated these ability heuristics with 37 design students, comparing them to usability heuristics and WCAG. The ability heuristics emphasized the *quality* of accessibility features compared to the other methods, and surfaced issues that were more broadly dispersed across disability groups. Further, the students found the heuristics were as easy to use as the alternative methods. We argue that the heuristics help to move beyond binary notions of accessibility, pushing designers to consider the quality of features across diverse disabilities and the range of abilities within.

CCS Concepts

• **Human-centered computing** → **Accessibility design and evaluation methods**; **Heuristic evaluations**.

Keywords

Accessibility, Ability-Based Design, Heuristic Evaluation

ACM Reference Format:
Claire L. Mitchell, Junhan Kong, Jesse J. Martinez, Shaun K. Kane, Amy J. Ko, Alexis Hiniker, and Jacob O. Wobbrock. 2026. Ability Heuristics for Conducting Accessibility Inspections. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3772318.3791504

1 Introduction

Achieving accessibility in interactive systems requires making thoughtful design choices that consider the diverse needs, abilities, and disabilities of all users, and reflects upon the contexts and environments in which the technology might be used [36, 68, 69]. It is essential to understand the true impact on accessibility of designers' choices, which are often made without accessibility in mind. This understanding can be gained through the evaluation of technology in a few ways. Traditionally, accessibility evaluation involves either empirical user testing or the utilization of extensive checklists that often require specialized expertise. While empirical testing is time-consuming, expensive, and inconvenient [35], checklists and guidelines can be used directly by designers and accessibility experts. These checklists, such as the Web Content Accessibility Guidelines (WCAG) [9], provide ways for designers and developers to ensure the accessibility of the technologies they create. However, checklists and guidelines have their limitations, especially as interactive technologies continue to proliferate with novel interaction modalities, such as those offered by virtual reality, which present unique accessibility challenges [11, 14, 28].

In contrast to empirical user testing, *inspection methods*—pioneered and most extensively developed for usability [50]—are characterized by their reliance on an evaluator's expert judgment. Such examples include heuristic evaluation [51], cognitive walkthrough [30], pluralistic walkthrough [6], guideline reviews [50], and consistency inspections [67]. Of the types of inspection methods, heuristic evaluation offers one of the most flexible methods—the number of heuristics is designed to be digestible and memorable, compared to guideline reviews, which might contain far more items to check. Moreover, heuristics are designed to be translatable across devices and scenarios unlike cognitive or pluralistic walkthroughs. One of the most prolific examples of heuristic evaluation in human-computer interaction (HCI) are Nielsen's usability heuristics, created in 1990 [46, 51] and updated in 1994 [48, 49]. Usability heuristics enabled designers and developers, rather than only usability specialists, to evaluate their own and others' interfaces during development in a low-cost and efficient fashion [25].

However, heuristic evaluation for accessibility has not been clearly established. Although accessibility testing tools provide a valuable service for detecting issues in finished systems, they are limited in scope and less useful *during* design and development. Owing to the widespread adoption of heuristic evaluation for usability, the prospect for accessibility evaluations via heuristics are promising indeed. But crucially, the right heuristics must be developed, ideally not out of "thin air" but grounded firmly in real systems and their accessibility failures. Among other things, that is what the current work contributes.

As a team of accessibility researchers, including both people with disabilities and allies, we designed a set of *ability heuristics* based on ability-based design [68, 69], providing a method for evaluating the accessibility of interactive technologies. The development of these heuristics was made primarily in two steps: (1) we conducted an initial design session that consisted of the majority of the authors, (2) we consolidated themes from the design session into heuristics through an iterative process. Prior to the design session, we engaged in two years' worth of formative work with students in three design classes who conducted an assignment on eliciting ability assumptions [68, 69]. Our analysis of these assignments helped to inform our initial brainstorms on the development of our ability heuristics. During the design session we brainstormed potential heuristics under, but not limited to, the seven principles of ability-based design—ability, accountability, availability, adaptability, transparency, performance, and context. We then qualitatively coded these potential heuristics through inductive analysis, resulting in themes that we used to combine heuristics into a smaller set. We iterated upon the remaining heuristics until we reached agreement.

We identified nine heuristics for an accessibility heuristic evaluation. These heuristics encompass themes that include, but are not limited to: personalization, I/O devices, task completion, transparency, help and support, and communication modalities. Each heuristic has a concise title that communicates the main point of the heuristic, as well as a short description to explain the heuristic. The heuristics are organized to begin with broad themes across interactive experiences, progress to specific aspects of the interactive experience, and end with ethical and social aspects of the interactive experience.

We evaluated these ability heuristics against the Web Content Accessibility Guidelines (*WCAG*) [9] and Nielsen's usability heuristics [48] in a classroom study with 37 master's students in HCI and Design, where each student was assigned one method to evaluate the accessibility of interactive technologies. Overall, students using the *Ability Heuristics* compared to the alternative methods were able to find more issues centered around the quality of an accessibility feature, as opposed to solely the presence or absence of an accessibility feature. Further, issues found from the ability heuristics had a greater emphasis than the alternative methods on the theme of the flexibility of an interactive interface. We also found the participants applied the ability heuristics more *evenly* across the set of heuristics, compared to the distribution of their use of usability heuristics. Further, these students reported experiencing a perceived workload that was equivalent to that found by students using the alternative methods.

The main research contributions of this work include providing a set of ability heuristics for conducting accessibility inspections, as well as evidence to support their use for this purpose. Our work provides designers and developers, especially those without accessibility expertise, a method for creating more accessible and inclusive technologies, especially usable during the design and development process. By emphasizing accessibility quality over binary notions of accessibility, these heuristics prompt the exploration of what it truly means to create accessible interactive technology.

2 Related Work

Previous work related to our efforts generally falls into the areas of (1) usability evaluation methods, (2) accessibility evaluation methods, and (3) ability-based design and related perspectives in HCI.

2.1 Usability Evaluation Methods

Usability evaluations offer insight into the effectiveness of interfaces by emphasizing the user experience rather than solely functional performance. These usability evaluations can be done in one of four ways: (1) empirical user tests, (2) automatic metric computation, (3) formal model calculation, and (4) by following informal general rules [50]. Of the most relevance to our work is the latter category, informal general rules or "heuristics," as our work looks to translate this technique to accessibility evaluation. This category includes inspection methods such as heuristic evaluation [51], heuristic estimation [52, 56], cognitive walkthrough [30], pluralistic walkthrough [6], feature inspection [50], consistency inspection [50], and formal usability inspection [26].

Heuristic evaluation uses general heuristics to evaluate a piece of technology. One of the most used set of guidelines for heuristic evaluation are Nielsen's usability heuristics [46, 48, 49, 51]. The usability heuristics are: (1) visibility of system status, (2) match between system and the real world, (3) user control and freedom, (4) consistency and standards, (5) error prevention, (6) recognition rather than recall, (7) flexibility and efficiency of use, (8) aesthetic and minimalist design, (9) help users recognize, diagnose, and recover from errors, and (10) help and documentation. Nielsen's usability heuristics were developed first through the classification of survey responses from 77 designers and programmers [46]. This classification was determined through the authors' experience and prior work. The authors did not evaluate the use of these heuristics until later work [51], where they asked evaluators to find accessibility problems and compared the issues they found to a known list of issues. These heuristics were iterated upon until the final set of 10 were achieved [48, 49].

Similar to these usability heuristics, Gerhardt-Powals designed design criteria, but for the reduction of cognitive load [17]. Shneiderman detailed eight golden rules of interface design [61]. With heuristic evaluation, the strength of the method depends on the thoroughness yet brevity of the set of heuristics, ensuring that all relevant aspects of the evaluation are addressed while remaining concise enough for evaluators to hold the entire set in working memory. Because they operate at an appropriately abstract level, the heuristics are broadly applicable across diverse contexts and technologies, supporting generalization beyond any single platform or specific user group.

Further usability inspection methods include *heuristic estimation* [53, 56], which asks for the evaluator to estimate in quantitative terms the usability of the system. *Cognitive walkthrough* [31] is a process of iterating through the steps of a task, answering predefined questions to understand what usability issues may arise. Similar to cognitive walkthrough is *pluralistic walkthrough*, which extends cognitive walkthroughs by including a user recruitment phase, as opposed to only designers and developers [50].

Features inspection requires inspecting each feature an interface offers and critiquing the steps required for that feature. *Consistency inspection* has designers compare an interface of their own design to the one of interest, whereas *standards inspection* [50] requires an expert in a specific type of interface evaluate the interface. Finally, *formal usability inspection* combines elements of a heuristic evaluation and cognitive walk-through for a formal six step process [26, 50].

These general methods provide valuable ways to evaluate technology, and in many cases can surface accessibility problems, but none prioritize accessibility at their core. With our goal to create an accessibility inspection method that can help both expertise designers as well as novice designers, achieve more accessible interfaces, we looked to follow the guidelines of other heuristic evaluations in the creation of our accessibility inspection method. Following these guidelines creates general accessibility guidelines that avoid being platform- , feature-, or task-specific as many of the above inspection methods are.

2.2 Accessibility Evaluation Methods

The evaluation of the accessibility of an interactive technology has historically taken an alternative approach from inspection methods through the use of empirical testing. But testing requires the recruitment, scheduling, and running of participants, which can be time-consuming, costly, and inconvenient. Simulation methods have been devised to overcome some of these limitations, but are therefore lacking in end-user authenticity [7, 7, 35].

In contrast to empirical testing, the Web Content Accessibility Guidelines (WCAG) detail 13 guidelines, organized under four "POUR" principles: (1) perceivable, (2) operable, (3) understandable, and (4) robust [9]. For each of the 13 principles there are success criteria, at levels A, AA, AAA. To meet a conformance level of A, all criteria of level A must be satisfied; to meet a conformance level of AA, all criteria of level AA and A must be satisfied; and to meet a conformance of level AAA, all criteria of all levels must be satisfied. In all, there are 87 individual criteria in WCAG. While WCAG has a broad view of the disabilities it seeks to cover, including but not limited to blindness and low vision, deafness and hearing loss, photosensitivity, speech disabilities, and reduced movement, it is limited in that it caters to web content, as opposed to interactive technology more broadly. This limitation highlights that there is limited feasibility in designing an accessibility testing method, as opposed to an inspection method, given that a testing method will be platform specific. The W3C also provide supplemental guidance for cognitive and learning disabilities that extends WCAG [60]. This guidance is separated into user stories that exemplify user needs, a design guide for implementation, information for how best to implement user testing, and personas.

Companies with their own mobile and interactive technologies also provide guidelines for accessibility. Google has an Android accessibility checker [20], courses for accessibility [13], accessibility scanners for Apple iOS apps [1]. Apple details its accessibility guidelines for development [2]. Meta describes their accessibility features [39] and further details virtual reality checks for applications for the Meta Horizon Store [38]. Microsoft provides multiple accessibility features and guidelines at different scopes for its users. Embedded in many of its products is an accessibility checker for users to employ to check for accessibility in their documents, emails, slides, and more [43]. This accessibility checker complements guidelines detailed for encouraging accessible writing, graphics, and websites [40]. In addition to ensuring the accessibility of the services its products help produce, Microsoft also provides accessibility features to access its devices and software. This includes adaptive input devices, as well as 3D-printable customizable input devices [42], alongside an AI-agent that helps users find accessibility features [41]. Despite the extensive resources these companies provide, they are often made in service of their specific platforms, as opposed to being broadly generalizable to different interaction devices, product functions, or interfaces.

Various automated checkers have been created to facilitate the accessibility evaluation process. There have been a number of tools and browser extension created for checking website accessibility, such as WAVE [66], Axe [24], Lighthouse [10], to name a few. Automated accessibility checkers have also been created for mobile applications, for example, the Google Accessibility Scanner [20] for Android and the Accessibility Inspector [3] for Apple iOS. ScreenAudit, and LLM-based accessibility checker, detects accessibility issues for Screen Readers in mobile applications [70]. While these checkers help to facilitate the accessibility process, they are still oriented towards individual product types, such as Android mobile applications.

Methods more aligned to inspection methods for accessibility have been recently explored. For example, Deets [12] evaluated the Johns Hopkins COVID-19 dashboard using heuristic evaluation through the principles of structure, navigation, and description, principles outlined in Zong et al. [71]. While this method provides a more generalizable technique than checklists, these principles were created to improve the screen reader experience, as opposed to a variety of input or assistive devices. Meanwhile, the game accessibility guidelines [15] provide a set of basic, intermediate, and advanced guidelines for developing accessible games that are grouped into subcategories. While these guidelines provide valuable insights into an accessible gaming experience, these guidelines are still much too numerous, as there are 123, to be used properly for a heuristic evaluation. Madan et al. [34] developed accessibility heuristics for vibe coding interfaces, enabling the evaluation of conversational programming interfaces, however, their heuristics are limited to blind and low vision individuals.

2.3 Ability-Based Design for Accessible Interactive Experiences

Ability-based design introduced a new perspective for accessible design in human-computer interaction (HCI) by emphasizing the need to design for an individual's unique and situated abilities [68, 69]. The authors proposed achieving personalized accessibility by capturing measurements of a user's abilities or by allowing the user a wide range of customizations. Ability-based design has been employed for a variety populations, including those with motor disabilities [16, 44, 45, 63], intellectual disabilities [4, 5], aging adults [57–59], and blind and low-vision disabilities [16]. In addition, it has been used for measuring situational and contextual abilities, such as during walking [19] or while under the influence of alcohol [37]. While these many implementations uphold certain principles of ability-based design, there is not yet an accompanying method to assess conformance to ability-based principles.

Ability-based design discusses the concept of *ability assumptions*, wherein interactive technologies are inherently embedded with assumptions regarding the user's abilities, and how those abilities may or may not change as the user, their environment, and their context changes [68, 69]. These assumptions may be the result of specific choices by the developers or a result of inherent biases by the developers. Our heuristics can help uncover ability assumptions in the design of technology.

Ability-based design's perspective is unique from alternative methods of accessibility design in HCI, such as universal design, inclusive design, or co-design. Universal design [32] aims to create technologies that work for as many users as possible without additional accommodation. Inclusive design [27], while similar, specifically looks to reduce barriers that prevent access to technologies. Co-design [21], also referred to as participatory design [47], is also often used as an accessibility design process by holding co-design sessions with the specific stakeholder groups.

In addition to design methodologies for accessibility, additional research discusses accessibility in HCI. Mack et al. [33] highlights the need for accessibility practice to consider the needs of chronically ill people, especially how their needs and abilities fluctuate over time. Mankoff et al. [36] discusses the intersection of disability studies and assistive technology.

While these alternative perspectives towards disability in HCI provide valuable insight towards development of an accessibility-focused heuristic evaluation, we choose to situate our heuristics through the lens of ability-based design. This choice in part reflects ability-based design's outlook on sensing and modeling, which as wearable computing becomes even more ubiquitous and adoptable for interactive technology, increases ability-based design's relevance. In addition, our choice reflects our desire to emphasize reducing user burden, especially the burden of adaptation, as ability-based design stresses [68, 69]. Although we situated the heuristics through the lens of ability-based design, we incorporated additional perspective as we created the heuristics such as considering how to account for fluctuating abilities [33]. We also incorporated into a heuristic the principle of gathering community feedback, aiming to emphasize, as participatory design does, the value of community feedback.

3 Ability Heuristics for Accessibility Inspections

From our work, we identified nine ability heuristics (Table 1). Here, we describe the methods in detail that we used to create the heuristics and the motivation for each heuristic.

3.1 Constructing Ability Heuristics

Our development of the heuristics took a multi-faceted approach to both mirror some of the steps Nielsen's usability heuristics used in their development [46, 48, 49, 51], and to ensure the robustness of the heuristics. We also integrated other practices to validate the heuristics through combining (1) an evaluation of the heuristics on an interactive technology, and (2) an evaluation of the heuristics against other techniques.

3.1.1 Ability Assumption Elicitation. As a first step in creating ability heuristics, we examined whether eliciting ability assumptions alone provided sufficient scaffolding for the generation of ability heuristics through a classroom assignment. Such assignments have also been created and analyzed for assumption elicitation more broadly, such as the CIDER technique [54, 55].

Over the course of two years we conducted variations of this assignment in three different classes. We instructed students to elicit assumptions that their chosen technology makes about the user and the context in which the user is operating. Two offerings of the class were a master's level course in HCI and design, one was an undergraduate course focused on accessibility. One offering of the master's course as well as the undergraduate course looked at this assignment from the perspective of only assumptions about the abilities of the user; the second offering of the master's course considered assumptions more broadly. Our examination of these assumptions helped us understand the types and ranges of disabilities students considered, as well as what students identified as accessibility issues. Further, our analysis affirmed that for comprehensive ability heuristics, relying solely on elicited assumptions was not sufficient for creating ability heuristics, and that eliciting ability assumptions alone is not sufficient as an inspection method for uncovering accessibility issues. Despite the limitations of this assignment, we used the themes were identified in the analysis of the assignments to help our brainstorming process.

3.1.2 Brainstorming and Design. We conducted a brainstorming and design session for the development of the ability heuristics, where the authors of the paper, some of whom identify as having a disability, and all whom are experts in HCI and accessibility, came together to identify possible heuristics. Prior to the brainstorming session, a subset of authors brainstormed possible heuristics. Further, the first author compiled a list of themes of accessibility in HCI alongside the principles of ability-based design [68, 69], the WCAG guidelines and its POUR principles [9], and Nielsen's usability heuristics [46, 48, 49, 51].

At the beginning of the session we discussed themes and ideas to prime our brainstorming. We reviewed Nielsen's usability heuristics [48] to ensure all the authors had a shared understanding of the scope of heuristics we sought to identify. We provided an overview of the POUR principles as well as the 13 WCAG guidelines, as well as themes discussed in literature. And finally, we went over

Table 1: Ability Heuristics

Title	Description
Adaptability	Systems should adapt or be adaptable to a user's dynamic situated abilities to reduce user burden and improve user-system-context fit. Options for personalization and customization should be apparent, accessible, flexible, expressive, predictable, changeable, undoable, and shareable, applicable to both functional and cosmetic aspects of interfaces.
Equitable Experience	Systems should provide an equitable experience for users of all abilities, regardless of utilized interaction techniques, I/O devices and channels, and interaction pathways. Satisfaction, ease, efficiency, effectiveness, and information should be equitable, even if different, for all users.
Flexible Task Completion	Common or important tasks, including error recovery, should have multiple pathways for achieving them, be able to be paused and resumed without loss of progress or context, and should not have a time dependency such that a user cannot move at their preferred pace. Extended tasks should communicate what will be required of the user.
Efficiency and Effectiveness of User Action	A user's actions should be effective in bringing about their intended result in an efficient manner. Burdensome actions, especially repetitive ones, should be replaceable in favor of preferred ones.
Multiple Modalities	Information coming to and from the user should be communicated via more than one modality and appropriately tailored to each modality. The same information should be available in each modality chosen by the user.
Understandable Messages	Textual messages, including instructions, labels, feedback, and errors, should be conveyed in plain user-centered language. Such messages should also be available in the user's preferred language. Non-textual messages such as icons, earcons, and hapticons should be as universally comprehensible as possible.
Ease of Adoption	Systems should use affordable, available, and/or open-source components that are compatible with third-party assistive technologies.
Ability Data Transparency	Systems should be transparent regarding the data they capture about a user's abilities, how that data is used and stored, the rationale for capturing that data, who has access to it, and how the user can audit, correct, prevent, and enhance the use of that data in system models and settings.
Help, Support, and Community	Systems should provide accessible help and support. Methods should be provided to access and share community knowledge. Systems should reflect a responsiveness to community feedback and needs.

the ability-based design principles. From there, our design session consisted of brainstorming possible heuristics under each of the ability-based design principles. We also allowed for ones that did not fall under any of the principles.

From there, the first author took those codes and summarized a heuristic for each code based off of the ideas that applied to the codes. Not all ideas were incorporated into their respective heuristic in order to keep the heuristics at the correct scope. Further, not all codes were used, as some did not give the correct scope. The authors reviewed these initial heuristics and then two of the authors finalized this list into the heuristics presented to the students.

3.1.3 Feedback from Ability Heuristic Evaluation. From confusion observed with the use of the heuristic *Data Transparency*, we altered the title of the heuristic to *Ability Data Transparency*. This reflects some students using the principle to talk about data privacy generally, as opposed to related to ability and disability specifically. While data transparency broadly is important, with this heuristic and for the purposes of an accessibility evaluation, we wanted the emphasis to belong around ability data.

3.2 Ability Heuristics

Here, we describe our motivation and focus for each heuristic. The motivations are summarized for each heuristic (italicized) to accompany the formal definition as provided in Table 1 to aid in the use of the heuristics. We organized the nine heuristics we identified into a structure: the first six heuristics are arranged to start from broad themes that persist across the entirety of the interactive experience, and progress to narrower themes that touch upon specific aspects of the interactive experience. The final three heuristics cover social aspects of the interwoven nature of ability, accessibility, and technology.

3.2.1 *Adaptability.*

Ability-based design emphasizes the need for systems to adapt to the user's abilities, as opposed to users adapting themselves to systems. In creating this heuristic, our goal was to emphasize this point [68, 69]. Any adaptations should be able to reflect both the changing nature of ability, whether because of situational or contextual factors, or because the user's internal abilities are changing due to disease progression, injury, medication, fatigue, or other factors. While automatic adaptations might provide the most seamless experience, it is still vital to allow the user autonomy over the adaptations.

> *Adaptability gives the user agency over their system's settings, tailoring those settings to best meet a user's needs, including if their abilities change over time or in different situations.*

3.2.2 *Equitable Experience.*

Adaptability in ability-based design results in the system adapting to the *unique* needs of a user. This adaptation may not require *equal* effort by the system or its designers, but rather that it results in a similar quality of experience for every user. Thus, our goal was to emphasize the need to design for equitability over equality. The same adaptation and accessibility feature that lets one user use an application with ease might not be sufficient, or may even be more disabling, for another user. These user-specific adaptations do not mean that the experience between users will necessarily be the same, and I/O devices and interaction pathways may be different, but the result from the user experience should be the same.

> *Providing people with disabilities a comparable user experience to those without disabilities should be a priority of every designer. Using a system with a disability should not mean having a poor user experience, even if that experience must be somewhat different from that of others.*

3.2.3 *Flexible Task Completion.*

The goal with a system that allows flexible task completion is to allow the user to choose a pathway that works the best for them. This includes flexibility with the types of interactions, such as scrolling, gesturing, or selecting, as well as the routes on the interface that one can take to get there. This flexibility should be made without sacrificing interface simplicity.

> *Allowing tasks to be completed flexibly means that users can work in the ways that best suit their abilities and that they can achieve what they set out to do at their own pace.*

3.2.4 *Efficiency and Effectiveness of User Action.*

For users with disabilities, ensuring actions are effective results in less burden on the user. Further, with efficient and effective actions, users can spend their time working towards their goals, as opposed to correcting or repeating ineffective actions.

> *By ensuring each action taken by users is efficient and effective, a system avoids frustrating or fatiguing users, and users can make progress towards their goals.*

3.2.5 *Multiple Modalities.*

By communicating information through more than one modality, one can ensure a system is adaptable to a user's available sensory channels. Further, when information quality is the same between modalities and tailored to each modality, it ensures a user is not missing information that might otherwise be conveyed only in one channel. In practice, this would allow users that otherwise might not be able to access content that is available to them to have access to the content.

> *Different users prefer to communicate and receive information using different modalities, which rely on their different senses or abilities (e.g., seeing, hearing, touching, speaking, typing, gesturing). It is important for systems to convey and receive information in ways that users prefer.*

3.2.6 *Understandable Messages.*

Messages that have consistency and simplicity, whether written or spoken, textual or non-textual, ensure that users know what to expect when they encounter the message. This means that users do not have to spend extra effort attempting to understand messages instead of working towards their goals.

> *Messages may be visual, auditory, or haptic. Whatever their form, messages must be understandable to users, avoiding jargon and unnecessary complexity.*

3.2.7 *Ease of Adoption.*

Systems that provide easier integration, especially with open-source and inexpensive assistive devices, ensure that users can easily adopt the technology. Further it ensures that users can easily personalize and customize the system, without having to adapt themselves to fit a system that is not designed for their abilities out of the box.

> *Access is not only about functionality but also about economics and availability. By using technologies that are based in and compatible with other commonly used, open-source, and/or assistive technologies, users can more easily integrate technologies into their lives and ensure they are not abandoned.*

3.2.8 *Ability Data Transparency.*

With increased ability to sense user ability through advanced sensing techniques, it becomes ever more important to ensure privacy when it comes to sensitive ability and disability data. As ability-based design often focuses on sensing to be able to understand one's abilities so that systems can adapt to them, it is essential to balance that focus with guidelines that protect the user, and provide them with agency regarding their ability data.

> *Data collected about a user's abilities is highly personal and must be always kept secure and private. It is essential that users have awareness of, access to, and agency over all data collected about themselves.*

3.2.9 *Help, Support, and Community.*

Disability communities should be the ultimate arbitrator of accessibility features that support their communities. Without this essential feedback, systems become reliant on the inherent biases of the designers, despite efforts to ensure their technologies are accessible.

When users have difficulties, they must be able to get the help they need. That help must itself be accessible, whether it comes from a company, fellow user, or community member. When an accessibility problem is raised by the community, it should be addressed promptly by the creators of the technology.

4 Evaluation of Ability Heuristics

We evaluated the effectiveness of our *ability heuristics* by studying: (1) their ability to identify accessibility issues, (2) the types of issues they identified, and (3) the effort required when using the heuristics. We performed our assessments through an evaluation of the ability heuristics against usability heuristics [48] and WCAG [9] in a between-subjects study with 37 designers.

4.1 Method

The setting for our study was a master's class in HCI and Design with professional design students. We created a homework assignment that 37 students in the class completed as part of their instruction on accessible design. This study was approved by our institution's human subjects review board.

4.1.1 Participants. Of the 41 students in the class, 37 consented for their assignment to be used for data analysis. Of these students, 26 were women and 11 were men. We surveyed the experience of the students and found that 23 of the students had worked as a paid designer previously, seven had done a paid heuristic evaluation previously, and six had previously done paid accessibility work. As the heuristics were designed to support designers without accessibility experience, we selected this class for the study as we wanted to understand how designers-in-training and evaluators with minimal accessibility experience would use the heuristics, as well as if the heuristics provided enough detail and coverage to identify accessibility issues.

4.1.2 Experiment Design. This study was between-subjects, wherein we randomly assigned all participating students to a different interface inspection method. Of the students that granted permission for analysis of their assignments, 14 of the students were assigned *ability heuristics*, 12 were assigned *usability heuristics*, and 11 were assigned *WCAG*. Each inspection method was used as an accessibility inspection method, meaning that the *Usability Heuristics* were used to find accessibility issues, not general usability issues, and that the 13 *WCAG* guidelines were used without their myriad success criteria. For *WCAG*, we provided the 13 guidelines categorized under the POUR principles to maintain consistency. By removing the success criteria of *WCAG*, we removed the *testable* success criterion that change it from a set of guidelines into a checklist evaluation. In this way, we could ensure fair comparison to our ability heuristics. All of the heuristics across the three inspection methods are shown in Table 2.

4.1.3 Class Assignment. For the assignment, the students applied their assigned inspection method to three pieces of interactive technology: (1) Microsoft PowerPoint on a desktop or laptop, (2) Google Maps on a mobile device, and (3) Amazon.com on a web browser on a desktop or laptop. We choose devices and technologies that students would be likely to have and, if not, would be able to obtain through university resources. Further, we balanced the availability of the devices and technologies, as well as the scope of a feasible assignment, against evaluating a larger variety of devices and technologies.

For each technology, students were instructed to identify accessibility issues they found while using their inspection method. For each issue found, they recorded the issue, the principle they used to find the issue (see 2), where in the interface or device they found the issue, as well as the severity of the issue on a 1-4 scale, all on a provided template. If the students found an accessibility issue that did not correspond to a principle, they noted so.

Following their completion of accessibility issue identification, we asked students to explain their use of the principles, describing their frequency of use across their principles. In addition, they completed a NASA-TLX workload questionnaire [22] to capture the effort of using their assigned heuristics. We indicated to the students that the assignment should not take more than 10 hours.

4.1.4 Data Analysis. Evaluation of the assignment for the purposes of this research did not take place until after final grades had been submitted for the class. We evaluated the assignments using inductive thematic analysis [8], noting both the issues students found as well as their rationale for choice of heuristic. First, each issue was analyzed for its relevance to accessibility. Inclusion criteria were determined by the first and second authors, and issues that were not accessibility issues, like general usability issues with no disproportionate accessibility impact, were not considered in subsequent analyses. The remaining issues were thematically analyzed, resulting in seven themes and 81 total codes, wherein each issue could belong to more than one theme. (These themes and codes are included in Appendix A.) A subset of issues were analyzed for inter-rater reliability, achieving a Cohen's kappa agreement value of $\kappa = 0.59$ [29].

5 Results

Our evaluation of *ability heuristics* demonstrates their ability to identify numerous accessibility issues. In addition, our qualitatiave results highlight that ability heuristics identify issues related to the *quality* of accessibility features (or lack thereof), emphasizing that the presence or absence of accessibility features alone is not enough. Further, our results emphasize that ability heuristics highlight different accessibility issues compared to *usability heuristics* and *WCAG*. We also demonstrate the feasibility of the method as an "accessibility heuristic evaluation," as student reflections did not find them to be any more burdensome than the alternative methods.

5.1 Quantitative Evaluation of Heuristic Use

Across all evaluations, students identified a total of 1103 issues. Of these issues, 657 (59.6%) were found to be accessibility related and were kept. Across all inspection methods, students identified an average of 17.76 ± 12.21 issues; additional averages by interactive technology and inspection method are shown in Table 3. There was a detectable difference between the number of issues found among all methods using a nonparametric Kruskal-Wallis test ($\chi^2(1, N{=}37) = 7.92$, $p = .019$); however, post-hoc pairwise comparisons corrected with Holm's sequential Bonferroni procedure [23] revealed no significant pairwise differences between the methods.

Table 2: Heuristics Across Three Inspection Methods

Ability Heuristics	Usability Heuristics	WCAG
1. Adaptability	1. Visibility of System Status	1. Text Alternatives
2. Equitable Experience	2. Match Between the System and the Real World	2. Time-based Media
3. Flexible Task Completion	3. User Control and Freedom	3. Adaptable
4. Efficiency and Effectiveness of User Action	4. Consistency and Standards	4. Distinguishable
5. Multiple Modalities	5. Error Prevention	5. Keyboard Accessible
6. Understandable Messages	6. Recognition Rather than Recall	6. Enough Time
7. Ease of Adoption	7. Flexibility and Efficiency of User	7. Seizures and Physical Reaction
8. Ability Data Transparency	8. Aesthetic and Minimalist Design	8. Navigable
9. Help, Support, and Community	9. Help User Recognize, Diagnose, and Recover from Errors	9. Input Modalities
	10. Help and Documentation	10. Readable
		11. Predictable
		12. Input Assistance
		13. Compatible

Table 3: Issues Identified (Average and Standard Deviation)

Interactive Technology	Ability Heuristics	Usability Heuristics	WCAG	Across Inspection Methods
Microsoft PowerPoint	6.36 ± 3.00	2.45 ± 2.94	8.00 ± 6.28	5.73 ± 5.73
Google Maps	6.36 ± 2.53	3.91 ± 3.67	7.33 ± 5.26	5.95 ± 5.95
Amazon.com	5.79 ± 2.75	3.82 ± 5.31	8.50 ± 4.85	6.08 ± 6.08
Across Technologies	18.50 ± 6.76	10.18 ± 10.26	23.83 ± 14.39	17.76 ± 12.21

5.1.1 *Distribution of Issue Severity.*

For both the ability heuristics and WCAG, students indicated the most common issue severity as "Major accessibility problem," while for usability heuristics, the most common issue severity was "Minor accessibility problem" (Figure 1). Statistical analysis of these results using a two-sample chi-squared test of association revealed no significant difference of issue severity between inspection methods.

5.1.2 *Distribution of Heuristics.*

Figure 2 shows the frequency of use of each heuristic across inspection methods and how this frequency differed from proportional use. Figure 2 shows that ability heuristics and WCAG have a smoother decrease from the most popular to least popular heuristics, whereas usability heuristics have a sharp decrease from the most popular to second-most popular heuristic.

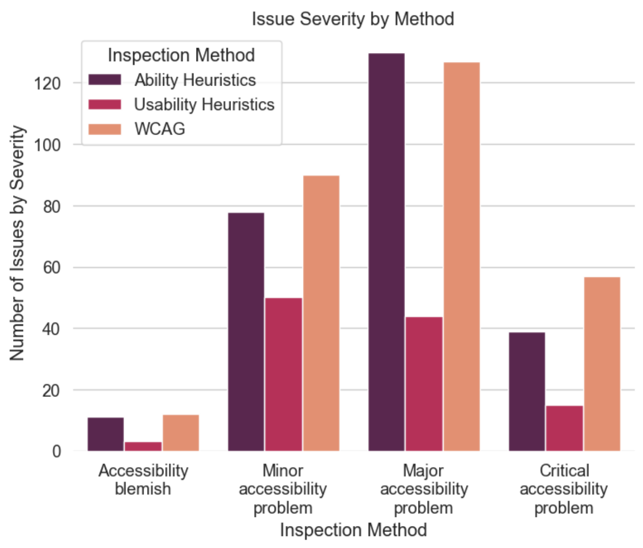

Figure 1: A bar graph of issue severity by inspection method. For both *ability heuristics* (purple) and *WCAG* (pink), the most common severity was "Major accessibility problem," followed by "Minor accessibility problem." For *usability heuristics* (orange), the most common severity was "Minor accessibility problem," followed by "Major accessibility problem."

5.2 Ability Heuristics Highlight Nuances in Accessibility Features

From our qualitative coding, we identified that students were able to find a wider range of accessibility issues using *ability heuristics* and *WCAG* as opposed to *usability heuristics.* In addition, we found that between ability heuristics and WCAG there were large differences in the types of issues they identified.

5.2.1 Quality of an Accessibility Feature. Effective accessibility features are designed to meet the needs of diverse disability groups and the heterogeneous abilities within each group. They also adapt to individual preferences and to fluctuations in users' needs and capabilities. As such, accessibility features are multifaceted and cannot be captured by simple binary compliance measures.

Throughout our qualitative analysis, we noticed issues that highlighted the *quality* of accessibility features, as opposed to issues that only noted the presence or absence of an accessibility feature. This theme regarding accessibility feature quality coalesced around a set of codes shown in Figure 3, which were mentioned more while using ability heuristics than either of the other methods. For example, students highlighted incomplete or inaccurate accessibility features, such as *"Although PowerPoint uses AI-based image recognition to automatically generate alt text, it often misses the context or key details specific to the presentation. This shifts the responsibility back to the user, adding unnecessary cognitive load and creating a frustrating experience, particularly for low vision or blind users who may struggle to manually enter accurate descriptions."* This issue highlights how just the presence of an accessibility feature is not necessarily sufficient, but that the feature must be complete so that the burden is not shifted back to the user. Another student noted about Google Maps that it only seems to emphasize wheelchair accessibility among all types of mobility disabilities: *"Limited disability options could possibly prevent users with other disabilities from being included in the user group."* This quote highlights the issue of accessibility features only catering to a narrow group of disabilities.

While our coding analysis identified many more accessibility quality issues with ability heuristics than WCAG (Figure 3), it is important to note that when WCAG is performed as an accessibility checklist as it was designed, it delves more into the quality of accessibility features, detailing conformance levels. However, for the purposes of a heuristic evaluation, this quality emphasis did not emerge for WCAG.

5.2.2 Types of Accessibility Features and Technology. Ability heuristics and WCAG both surfaced many issues around the theme of "accessibility features and technology." However, they differed in the types of features mentioned. Students that used WCAG found more issues surrounding accessibility features for blind and low vision individuals, such as alt text, captions, magnification, and screen readers, while the features found by students using ability heuristics focused more on the flexibility of the interface. Such issues included accessibility settings, adaptability, and personalization and customization. Thus, WCAG and ability heuristics tended to identify different types of accessibility issues and could conceivably reinforce, rather than replace, each other.

5.2.3 Accessibility of Product Outputs. One of the more interesting codes we identified throughout our coding process dealt with issues identifying accessibility problems with the main purpose or output of the software. For Google Maps this could mean accessibility problems with the routes proposed. For Microsoft PowerPoint this could mean that the presentations it produces are inaccessible. For Amazon.com, this could mean that the products it recommends are inaccessible. This theme related to product outputs was a theme that we saw emerging much more with ability heuristics than with usability heuristics or WCAG. One student who used ability heuristics highlighted this issue, saying, *"While using Google Maps, I noticed that some locations are marked with the blue wheelchair accessibility icon, but there's no clear way to know how accurate that information is. I've come across places labeled as accessible that, in reality, are not. Google doesn't indicate how many people verified the accessibility of a location or whether it has been independently confirmed."* It is important that not only these tools are themselves accessible, but the outputs they produce are accessible, too.

5.3 Workload and Ease-of-Use

NASA TLX workload response averages, as well as averages by inspection methods, are shown in Table 4. There were no detectable differences in any of the workload responses between the inspection methods. Across the methods, students on average found the mental demand of the task to be the highest (5.57 ± 0.83), while the physical demand was the lowest (2.92 ± 1.80).

Students overall indicated that the ability heuristic *Equitable Experience* was the easiest to use. One student reflected, *"It was pretty obvious when something wasn't fair or equal for users with*

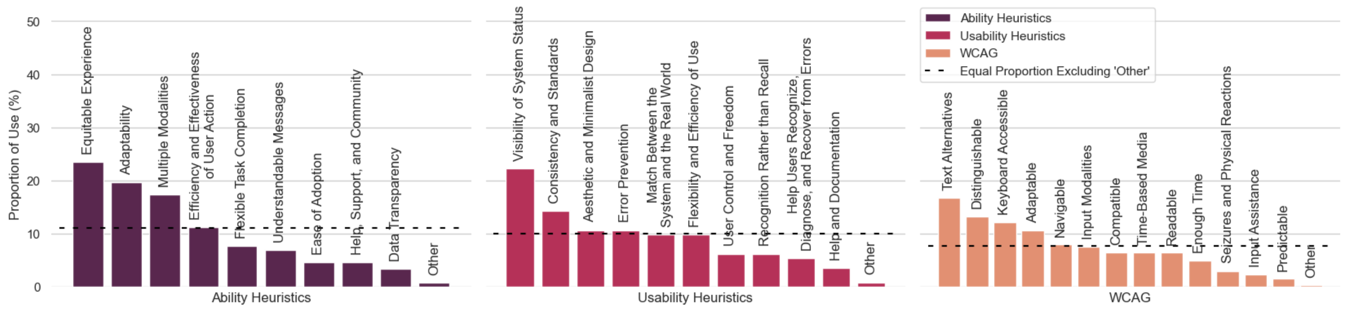

Figure 2: Bar charts of the proportion of use of heuristics by inspection method. *Ability heuristics* (left) and *WCAG* (right) demonstrate a steady decline across heuristics, while *usability heuristics* (middle) had a significant drop off between the most- and second-most common heuristics. The black line across all charts indicates equal proportion of use of heuristics. This value varies by method due to different numbers of heuristics.

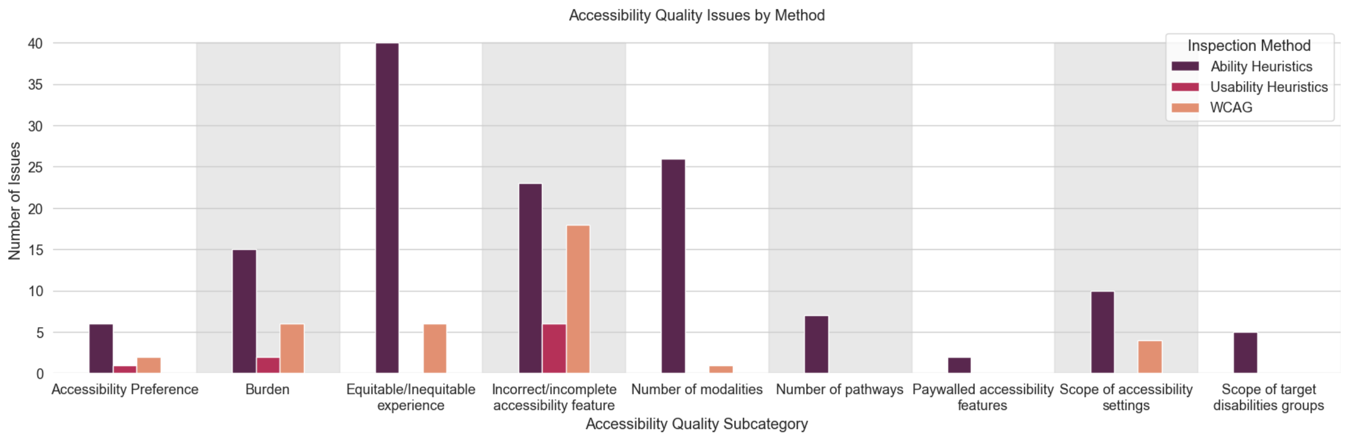

Figure 3: A bar graph of access quality issues by inspection method. *Ability heuristics* (purple) have issues in each subcategory, surpassing issue number for both *usability heuristics* (pink) and *WCAG* (orange). For *Ability Heuristics* the most frequent subcategory was Equitable/Inequitable Experience, followed by Number of modalities, then Burden.

Table 4: Workload Responses (Average and Standard Deviation)

Response Variable	Ability Heuristics	Usability Heuristics	WCAG	Across Inspection Methods
Mental Demand	5.57 ± 0.85	5.73 ± 0.65	5.42 ± 1.00	5.57 ± 0.83
Physical Demand	3.21 ± 2.01	2.82 ± 1.66	2.67 ± 1.78	2.92 ± 1.80
Temporal Demand	3.71 ± 1.20	4.55 ± 1.21	3.67 ± 1.61	3.95 ± 1.37
Performance	3.71 ± 1.44	4.18 ± 1.17	3.67 ± 1.67	3.84 ± 1.42
Effort	5.50 ± 0.94	5.45 ± 0.82	5.67 ± 0.65	5.54 ± 0.80
Frustration	4.07 ± 1.77	5.18 ± 1.33	4.42 ± 1.68	4.51 ± 1.64

disabilities. If a feature worked well for some people but created extra work, confusion, or outright barriers for others, it was a clear violation." Another student indicated that the heuristic *Equitable Experience "was fairly general, and if you considered any page of a piece of software and removed a core ability required to navigate it (whether it be physical input limitations, or limits of perceptions), often times the three products did not provide enough alternatives for it to be a truly equitable experience."*

In contrast, students indicated that the ability heuristic *Ease of Adoption* was the hardest to use, often due to unfamiliarity with the inner workings of the interface. One student reflected, *"I had limited knowledge about compatibility of these interactions with assistive technologies like screen readers, alt text displays, magnifying softwares etc."*, while another student wrote, *"Without access to the* *backend of the interface, seeing its technical implementation, it's difficult to determine if it uses components that are affordable, available, or open-source."* (Figure 4).

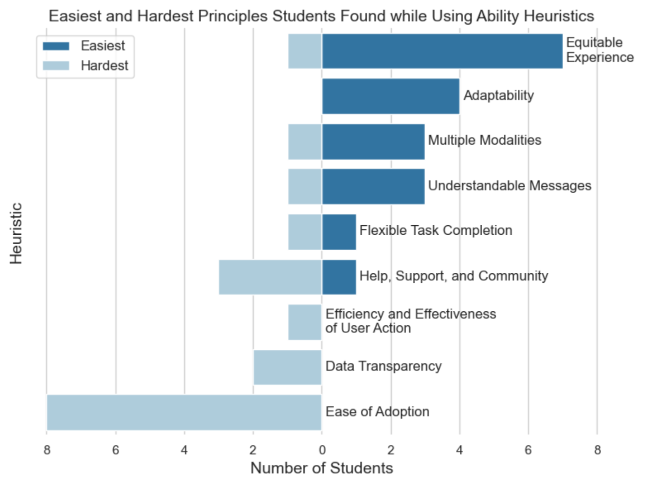

Figure 4: Count of ability heuristics students indicated as easiest (dark blue) and hardest (light blue) to use while conducting a heuristic evaluation using ability heuristics. The most common heuristic that students indicated as easiest to use was *Equitable Experience*, while *Ease of Adoption* was the hardest to use.

6 Discussion

In this work, we presented *ability heuristics*, which are used to evaluate the accessibility of interactive technology. This methodology contains nine heuristics that cover important accessibility considerations. In this section, we reflect on the strengths and weaknesses of ability heuristics, especially in relation to ability-based design and to other inspection methods.

6.1 Effectiveness of Ability Heuristics

Our ability heuristics demonstrate a level of effectiveness at finding accessibility issues across the interfaces and devices we chose to evaluate. These heuristics provide a unique focus for accessibility evaluations by highlighting how the quality of an accessibility feature matters to ensuring accessibility, and by focusing on the flexibility of interactive interfaces.

6.2 Significance for Accessibility

A heuristic evaluation for accessibility provides an additional tool for designers to ensure accessible interactive technology, beyond laborious technology-specific checklists such as *WCAG*. Our evaluation of the heuristics prove their ease to be learned in a classroom setting, enabling them to be used as a tool to teach accessibility. With laws such as the European Accessibility Act [65], it becomes critical that all designers and developers, not solely accessibility experts, have training in accessibility. Other tools, such as WCAG, are more difficult for non-experts to implement due to the numerous items one has to remember. Further, these heuristics are not intended as a replacement for empirical user testing; rather, they are meant to provide a foundation for accessibility, enabling user testing to be reserved for situations where it offers unique value.

WCAG's perspective on data privacy centers around managing user entered information that potentially could contain sensitive data in the context of redundant entry and user inactivity [9]. While this viewpoint is important, it does not account for ability information that systems can infer through modeling user interaction. We aimed to specifically address this in our heuristics, including it as it's own heuristic, *Ability Data Transparency*. The inclusion of this principle is also supported by ability-based design principles [68, 69] as well as the General Data Protection Regulation (GDPR) [64]. As society moves to store more self-identifying information online through social media, the electronification of medical records, and as devices are increasingly able to sense more regarding human abilities, providing control over this data becomes paramount.

6.3 Connection to Ability-Based Design

Our ability heuristics are deeply linked to ability-based design [68, 69], as it was through that lens that the heuristics were developed. This perspective is reflected in the heuristics themselves, through their emphasis on adapting to the user through personalization, multiple pathways, integration with accessibility technology, and more. This idea further manifests in the issues that students identified. For example, the focus on interface flexibility ties into the emphasis that ability-based design places on ensuring the burden of adaptation lies with the technology as opposed to the user.

6.4 Comparison of *Ability Heuristics* to Related Principles

Several of the ability heuristics bear similarities to existing design heuristics. For example, the heuristic *Equitable Experience* is similar to universal design's [32] *Equitable Use* principle. However, *Equitable Experience* emphasizes that the techniques used to achieve an equitable experience are likely to be different, while universal design's equitable use emphasizes providing identical means when possible [62]. Further, *Adaptability* in our ability heuristics also parallels *Flexibility in Use* in universal design, both emphasizing choice and abilities; however, *Adaptability* focuses more on the dynamic nature of ability, while *Flexibility in Use* only mentions user pace.

Help, Support, and Community from our ability heuristics has similarities to *Help and Documentation* from Usability Heuristics [48], in that both emphasize that it is necessary for help to be easy to find. However, *Help, Support, and Community* expands beyond *Help and Documentation*, emphasizing that this help should be accessible and that communities of interest should be able to impact changes that make for a more accessible product.

6.5 Alternative Use of *WCAG* and *Usability Heuristics*

In our evaluation of our ability heuristics, we sought to: (1) only evaluate interactive technology for its accessibility, (2) ensure we were comparing against other heuristics so that the issues identified were of similar scope, and (3) provide an equal-effort assignment across the methods students were assigned. For these reasons, with both the usability heuristics and WCAG, we altered the methods traditionally used. For the usability heuristics, we asked the students to use those heuristics through the lens of accessibility, not general usability. While undoubtedly this method would surface fewer overall issues than if those heuristics were used through the lens of usability, our goal was solely to identify accessibility issues. If the usability heuristics used this way identified more accessibility issues than the ability heuristics, it would have indicated issues with the ability heuristics. Given that our results indicated that the usability heuristics surfaced fewer issues, and that the issues identified did not encompasses accessibility issues as broadly as our ability heuristics, we can say that the usability heuristics are not as suitable as the ability heuristics for identifying accessibility issues.

We also altered the base methodology used when employing WCAG. While each guideline of WCAG has multiple success criteria that provide more specificity, we provided the guidelines to the students without these success criteria so that WCAG would be closer to a heuristic evaluation. This helped keep the workload across the inspection methods similar, while maintaining the focus of WCAG.

6.6 Limitations and Future Work

A limitation of this work is that it presents only an initial exploration of ability heuristics. Although these heuristics were created with extensive expertise in accessibility research and design, as well as by individuals with disabilities, we expect and welcome these heuristics to morph as the concept of "accessibility inspections" is explored through other work, and as both WCAG and usability heuristics have also changed with extensive use and technology evolution [9, 46, 48, 49, 51].

Another limitation is that this work did not evaluate the use of these heuristics with users with disabilities. While such an evaluation would undoubtedly uncover additional issues, our goal was to understand how designers and developers could effectively use the heuristics to find accessibility issues. We made this choice as these heuristics are not meant to replace feedback from users with disabilities, but rather to provide a tool for designers and developers, especially ones without accessibility training or access to accessibility experts. With this limitation in mind, we ensured that some of the authors involved in the project identified as having disabilities.

Future work should explore how the heuristics are used in practice and ultimately effect the accessibility of products. For example, it could be valuable to explore the kinds of modifications that are made when using the ability heuristics, especially in comparison to other accessibility evaluation tools.

Lastly, we examined our ability heuristics only with three pieces of interactive technology and two types of devices to provide a feasible assignment for our students, making sure that the assignment (1) took a feasible amount of time, (2) was able to be easily performed on devices the students had easy access to, and (3) used interactive technologies the students could easily access. While we limited our the number of interactive technologies, we ensured they were different types: one a mobile app, one a desktop app, and one a website. Future work should explore additional types of interactive technologies and devices, such as AR/VR devices, gaming consoles, or interactive kiosks.

Future work should explore applying our heuristics to additional pieces of interactive technology, ensuring their robustness. This will help to establish even more validity for these heuristics, and solidify their formulation. As other work explores the integration of accessibility and generative artificial intelligence (GAI) [18], we see the opportunity to integrate these heuristics with GAI to help provide an initial evaluation of the accessibility of a interface, which designers and developers can then use to probe deeper.

7 Conclusion

In this work, we present accessibility inspections using *ability heuristics*, which are nine ability-based heuristics for evaluating the accessibility of technology. As a form of heuristic evaluation, this approach differs from guideline reviews that assess accessibility as a checklist of items to fulfill. Our results show that ability heuristics are an effective method for conducting accessibility inspections, especially capable of assessing the *quality* of accessibility features and barriers, rather than just their presence or absence. With the established familiarity of usability inspection methods in HCI, it is our hope that accessibility inspections using ability heuristics can be a useful lightweight method for designers and developers to employ to ensure the widespread accessibility of their creations.

Acknowledgments

The authors thank the students of the University of Washington Master of Human-Computer Interaction and Design program. This work was supported in part by a National Science Foundation Graduate Research Fellowship. Any opinions, findings, conclusions or recommendations expressed in our work are those of the authors and do not necessarily reflect those of any supporter.

References

[1] 2025. google/GSCXScanner. https://github.com/google/GSCXScanner original-date: 2019-02-27T22:10:27Z.
[2] Apple. 2025. Accessibility. https://developer.apple.com/design/human-interface-guidelines/accessibility
[3] Apple. 2026. Accessibility Inspector. https://developer.apple.com/documentation/accessibility/accessibility-inspector
[4] Andrew A. Bayor. 2019. HowToApp: Supporting Life Skills Development of Young Adults with Intellectual Disability. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '19)*. Association for Computing Machinery, New York, NY, USA, 697–699. doi:10.1145/3308561.3356107
[5] Andrew A. Bayor, Laurianne Sitbon, Bernd Ploderer, Filip Bircanin, and Margot Brereton. 2019. "TechShops" Engaging Young Adults with Intellectual Disability in Exploratory Design Research. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, 1–8. doi:10.1145/3290607.3299056
[6] Randolph G. Bias. 1994. The pluralistic usability walkthrough: coordinated empathies. In *Usability inspection methods*. John Wiley & Sons, Inc., USA, 63–76.
[7] Pradipta Biswas and Peter Robinson. 2008. Automatic evaluation of assistive interfaces. In *Proceedings of the 13th international conference on Intelligent user interfaces (IUI '08)*. Association for Computing Machinery, New York, NY, USA, 247–256. doi:10.1145/1378773.1378806
[8] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (Jan. 2006), 77–101. doi:10.1191/1478088706qp063oa
[9] Alastair Campbell, Chuck Adams, Rachael Bradley Montgomery, Michael Cooper, and Andrew Kirkpatrick. 2024. Web Content Accessibility Guidelines (WCAG) 2.2. https://www.w3.org/TR/WCAG22/
[10] Chrome. 2025. Introduction to Lighthouse. https://developer.chrome.com/docs/lighthouse/overview

[11] Chris Creed, Maadh Al-Kalbani, Arthur Theil, Sayan Sarcar, and Ian Williams. 2024. Inclusive AR/VR: accessibility barriers for immersive technologies. *Universal Access in the Information Society* 23, 1 (March 2024), 59–73. doi:10.1007/s10209-023-00969-0

[12] Stuart M. Deets. 2023. Accessibility of a COVID-19 Interactive Display: An Evaluation of the Johns Hopkins COVID-19 Dashboard. In *Proceedings of the 41st ACM International Conference on Design of Communication (SIGDOC '23)*. Association for Computing Machinery, New York, NY, USA, 269–270. doi:10.1145/3615335.3623049

[13] Android Developers. [n. d.]. Make your Android app more accessible. https://developer.android.com/courses/pathways/make-your-android-app-accessible

[14] John Dudley, Lulu Yin, Vanja Garaj, and Per Ola Kristensson. 2023. Inclusive Immersion: a review of efforts to improve accessibility in virtual reality, augmented reality and the metaverse. *Virtual Reality* 27, 4 (Dec. 2023), 2989–3020. doi:10.1007/s10055-023-00850-8 Company: Springer Distributor: Springer Institution: Springer Label: Springer.

[15] Barrie Ellis, Gareth Ford-Williams, Lynsey Graham, Dimitris Grammenos, Ian Hamilton, Ed Lee, Jake Manion, and Thomas Westin. 2017. Game Accessibility Guidelines. https://gameaccessibilityguidelines.com/

[16] Krzysztof Z. Gajos, Jacob O. Wobbrock, and Daniel S. Weld. 2007. Automatically generating user interfaces adapted to users' motor and vision capabilities. In *Proceedings of the 20th annual ACM symposium on User interface software and technology (UIST '07)*. ACM Press, New York, NY, USA, 231–240. doi:10.1145/1294211.1294253

[17] Jill Gerhardt-Powals. 1996. Cognitive engineering principles for enhancing human-computer performance. *International Journal of Human–Computer Interaction* 8, 2 (April 1996), 189–211. doi:10.1080/10447319609526147

[18] Kate S Glazko, Momona Yamagami, Aashaka Desai, Kelly Avery Mack, Venkatesh Potluri, Xuhai Xu, and Jennifer Mankoff. 2023. An Autoethnographic Case Study of Generative Artificial Intelligence's Utility for Accessibility. In *The 25th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, New York NY USA, 1–8. doi:10.1145/3597638.3614548

[19] Mayank Goel, Leah Findlater, and Jacob Wobbrock. 2012. WalkType: using accelerometer data to accomodate situational impairments in mobile touch screen text entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. Association for Computing Machinery, New York, NY, USA, 2687–2696. doi:10.1145/2207676.2208662

[20] Google. 2026. Get started with Accessibility Scanner - Android Accessibility Help. https://support.google.com/accessibility/android/answer/6376570?hl=en

[21] Joan Greenbaum and Morten Kyng (Eds.). 1992. *Design at work: cooperative design of computer systems*. L. Erlbaum Associates Inc., USA.

[22] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Human Mental Workload*, P. A. Hancock and N. Meshkati (Eds.). Vol. 52. North Holland Press, Amsterdam, The Netherlands, 139–183. https://www.sciencedirect.com:5037/science/chapter/bookseries/abs/pii/S0166411508623869

[23] Sture Holm. 1979. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics* 6, 2 (1979), 65–70. https://www.jstor.org/stable/4615733

[24] Deque Systems Inc. 2025. Axe DevTool. https://chromewebstore.google.com/detail/axe-devtools-web-accessib/lhdoppojpmngadmnindnejefpokejbdd?hl=en-US&pli=1

[25] Robin Jeffries, James R. Miller, Cathleen Wharton, and Kathy Uyeda. 1991. User interface evaluation in the real world: a comparison of four techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '91)*. Association for Computing Machinery, New York, NY, USA, 119–124. doi:10.1145/108844.108862

[26] Michael J. Kahn and Amanda Prail. 1993. Formal Usability Inspections. In *Usability Inspection Methods*, Jakob Nielsen and Robert L. Mack (Eds.). Hewlett-Packard, 141–171.

[27] Simeon Keates and P. John Clarkson. 2002. Countering design exclusion through inclusive design. *SIGCAPH Comput. Phys. Handicap.* 73-74 (June 2002), 69–76. doi:10.1145/960201.957218

[28] Melanie Jo Kneitmix and Jacob O. Wobbrock. 2025. From Screen Reading to "Scene Reading" in SceneVR: Touch-Based Interaction Techniques for Use in Virtual Reality by Blind and Low-Vision Users. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '25)*. Association for Computing Machinery, New York, NY, USA, 1–18. doi:10.1145/3663547.3746364

[29] J. Richard Landis and Gary G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics* 33, 1 (1977), 159–174. doi:10.2307/2529310

[30] Clayton Lewis, Peter G. Polson, Cathleen Wharton, and John Rieman. 1990. Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. Association for Computing Machinery, New York, NY, USA, 235–242. doi:10.1145/97243.97279

[31] Clayton Lewis and Cathleen Wharton. 1997. Chapter 30 - Cognitive Walkthroughs. In *Handbook of Human-Computer Interaction (Second Edition)*, Marting G. Helander, Thomas K. Landauer, and Prasad V. Prabhu (Eds.). North-Holland, Amsterdam, 717–732. doi:10.1016/B978-044481862-1.50096-0

[32] Ron. L. Mace, G. J. Hardie, and J. P. Place. 1991. Accessible environments: Toward universal design. In *Design Intervention: Toward a More Humane Architecture*, Wolfgang F. E. Preiser, Jacqueline C. Vischer, and Edward T. White (Eds.). John Wiley & Sons Inc.

[33] Kelly Mack, Emma J. McDonnell, Leah Findlater, and Heather D. Evans. 2022. Chronically Under-Addressed: Considerations for HCI Accessibility Practice with Chronically Ill People. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '22)*. Association for Computing Machinery, New York, NY, USA, 1–15. doi:10.1145/3517428.3544803

[34] Shalini Madan, Sreelakshmi Surabiyil Bindu, and Venkatesh Potluri. 2025. Accessibility Heuristics for Vibe Coding Interfaces. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '25)*. Association for Computing Machinery, New York, NY, USA, 1–5. doi:10.1145/3663547.3759729

[35] J. Mankoff, H. Fait, and R. Juang. 2005. Evaluating accessibility by simulating the experiences of users with vision or motor impairments. *IBM Systems Journal* 44, 3 (2005), 505–517. doi:10.1147/sj.443.0505

[36] Jennifer Mankoff, Gillian R. Hayes, and Devva Kasnitz. 2010. Disability studies as a source of critical inquiry for the field of assistive technology. In *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility - ASSETS '10*. ACM Press, New York, NY, USA, 3. doi:10.1145/1878803.1878807

[37] Alex Mariakakis, Sayna Parsi, Shwetak N. Patel, and Jacob O. Wobbrock. 2018. Drunk User Interfaces: Determining Blood Alcohol Level through Everyday Smartphone Tasks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 1–13. doi:10.1145/3173574.3173808

[38] Meta. 2020. Introducing the Accessibility VRCs. https://developers.meta.com/horizon/blog/introducing-the-accessibility-vrcs/

[39] Meta. 2025. Accessibility at Meta. https://transparency.meta.com/features/accessibility-at-meta

[40] Microsoft. 2022. Accessibility Guidelines and Requirements. https://learn.microsoft.com/en-us/style-guide/accessibility/accessibility-guidelines-requirements

[41] Microsoft. 2025. Ask Microsoft Accessibility. https://askma.microsoft.com/?source=SMCA11y

[42] Microsoft. 2026. Adaptive accessories help & learning. https://support.microsoft.com/en-us/adaptiveaccessories

[43] Microsoft. 2026. Improve accessibility with the Accessibility Checker - Microsoft Support. https://support.microsoft.com/en-us/office/improve-accessibility-with-the-accessibility-checker-a16f6de0-2f39-4a2b-8bd8-5ad801426c7f

[44] Claire Mitchell, Gabriel Cler, Susan Fager, Paola Contessa, Serge Roy, Gianluca De Luca, Joshua Kline, and Jennifer Vojtech. 2022. Ability-Based Keyboards for Augmentative and Alternative Communication: Understanding How Individuals' Movement Patterns Translate to More Efficient Keyboards: Methods to Generate Keyboards Tailored to User-Specific Motor Abilities. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems (CHI EA '22)*. ACM Press, New York, NY, USA, 1–7. doi:10.1145/3491101.3519845

[45] Claire L. Mitchell, Gabriel J. Cler, Susan K. Fager, Paola Contessa, Serge H. Roy, Gianluca De Luca, Joshua C. Kline, and Jennifer M. Vojtech. 2022. Ability-Based Methods for Personalized Keyboard Generation. *Multimodal Technologies and Interaction* 6, 8 (Aug. 2022), 67. doi:10.3390/mti6080067

[46] Rolf Molich and Jakob Nielsen. 1990. Improving a human-computer dialogue. *Commun. ACM* 33, 3 (March 1990), 338–348. doi:10.1145/77481.77486

[47] Michael J. Muller and Sarah Kuhn. 1993. Participatory design. *Commun. ACM* 36, 6 (June 1993), 24–28. doi:10.1145/153571.255960

[48] Jakob Nielsen. 1994. Enhancing the explanatory power of usability heuristics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Boston Massachusetts USA, 152–158. doi:10.1145/191666.191729

[49] Jakob Nielsen. 1994. Heuristic evaluation. In *Usability inspection methods*. John Wiley & Sons, Inc., USA, 25–62.

[50] Jakob Nielsen and Robert L. Mack (Eds.). 1994. *Usability Inspection Methods*. John Wiley & Sons, Inc.

[51] Jakob Nielsen and Rolf Molich. 1990. Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. Association for Computing Machinery, New York, NY, USA, 249–256. doi:10.1145/97243.97281

[52] Jakob Nielsen and Victoria L. Phillips. 1993. Estimating the relative usability of two interfaces: heuristic, formal, and empirical methods compared. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems (CHI '93)*. Association for Computing Machinery, New York, NY, USA, 214–221. doi:10.1145/169059.169173

[53] Jakob Nielsen and Victoria L. Phillips. 1993. Estimating the relative usability of two interfaces: heuristic, formal, and empirical methods compared. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing*

Systems (CHI '93). Association for Computing Machinery, New York, NY, USA, 214–221. doi:10.1145/169059.169173

[54] Alannah Oleson. 2022. CIDER: A Method to Teach Practical Critical Software Design Skills. In *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 2 (ICER '22, Vol. 2)*. Association for Computing Machinery, New York, NY, USA, 7–9. doi:10.1145/3501709.3544295

[55] Alannah Oleson, Meron Solomon, Christopher Perdriau, and Amy Ko. 2023. Teaching Inclusive Design Skills with the CIDER Assumption Elicitation Technique. *ACM Trans. Comput.-Hum. Interact.* 30, 1 (March 2023), 6:1–6:49. doi:10.1145/3549074

[56] Peter Polson, John Rieman, Cathleen Wharton, Judith Olson, and M. Kitajima. 1992. Usability inspection methods: rationale and examples. In *The 8th Human Interface Symposium (HIS92)*, Vol. 1992. 377–384. https://www.colorado.edu/ics/sites/default/files/attached-files/92-07.pdf

[57] Sayan Sarcar, Jussi Jokinen, Antti Oulasvirta, Xiangshi Ren, Chaklam Silpasuwanchai, and Zhenxin Wang. 2017. Ability-Based Optimization: Designing Smartphone Text Entry Interface for Older Adults. In *Human-Computer Interaction – INTERACT 2017 (Lecture Notes in Computer Science)*, Regina Bernhaupt, Girish Dalvi, Anirudha Joshi, Devanuj K. Balkrishan, Jacki O'Neill, and Marco Winckler (Eds.). Springer International Publishing, Cham, 326–331. doi:10.1007/978-3-319-68059-0_22

[58] Sayan Sarcar, Jussi PP Jokinen, Antti Oulasvirta, Zhenxin Wang, Chaklam Silpasuwanchai Asian Institute of Technology, Asian Institute of Technology, and Xiangshi Ren. 2018. Ability-Based Optimization of Touchscreen Interactions. *IEEE Pervasive Computing* 17, 1 (2018), 15–26. www.computer.org/pervasive

[59] Sayan Sarcar, Jussi Joklnen, Antti Oulasvirta, Chaklam Silpasuwanchai, Zhenxin Wang, and Xiangshi Ren. 2016. Towards Ability-Based Optimization for Aging Users. In *Proceedings of the International Symposium on Interactive Technology and Ageing Populations (ITAP '16) (ITAP '16)*. Association for Computing Machinery, New York, NY, USA, 77–86. doi:10.1145/2996267.2996275

[60] Lisa Seeman-Horwitz, Rachael Bradley Montgomery, Steve Lee, and Ruoxi Ran. 2021. Making Content Usable for People with Cognitive and Learning Disabilities. https://www.w3.org/TR/coga-usable/

[61] Ben Shneiderman, Catherine Plaisant, Maxine Cohen, Steven Jacobs, Niklas Elmqvist, and Nicholas Diakopoulos. 2016. *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (6th ed.). Pearson. http://www.cs.umd.edu/hcil/DTUI6

[62] Molly Follette Story. 2011. The Principles of Universal Design. In *Universal design handbook* (2nd ed ed.), Wolfgang F. E. Preiser and Korydon H. Smith (Eds.). McGraw-Hill, New York. OCLC: 702446007.

[63] Daniil Umanski and Yael Avni. 2017. PLAY-ABLE: Developing Ability-Based Play Activities for Children with Special Needs. In *Proceedings of the 11th International Convention on Rehabilitation Engineering and Assistive Technology (i-CREATe 2017)*. Singapore Therapeutic, Assistive & Rehabilitative Technologies (START) Centre, Midview City, SGP, 1–4.

[64] European Union. 2016. General Data Protection Regulation. https://gdpr.eu/tag/gdpr/

[65] European Union. 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 (Text with EEA relevance). http://data.europa.eu/eli/dir/2019/882/oj

[66] WebAIM. 2026. WAVE Web Accessibility Evaluation Tools. https://wave.webaim.org/

[67] Dennis Wixon, Sandra Jones, Linda Tse, and George Casaday. 1994. Inspections and Design Reviews: Framework, History, and Reflection. In *Usability Inspection Methods*, Jakob Nielsen and Robert L. Mack (Eds.). John Wiley & Sons, Inc., 141–171.

[68] Jacob O. Wobbrock, Krzysztof Z. Gajos, Shaun K. Kane, and Gregg C. Vanderheiden. 2018. Ability-Based Design. *Commun. ACM* 61, 6 (May 2018), 62–71. doi:10.1145/3148051

[69] Jacob O. Wobbrock, Shaun K. Kane, Krzysztof Z. Gajos, Susumu Harada, and Jon Froehlich. 2011. Ability-based design: Concept, principles and examples. *ACM Transactions on Accessible Computing* 3, 3 (April 2011), 1–27. doi:10.1145/1952383.1952384

[70] Mingyuan Zhong, Ruolin Chen, Xia Chen, James Fogarty, and Jacob O. Wobbrock. 2025. ScreenAudit: Detecting Screen Reader Accessibility Errors in Mobile Apps Using Large Language Models. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3706598.3713797

[71] Jonathan Zong, Crystal Lee, Alan Lundgard, JiWoong Jang, Daniel Hajas, and Arvind Satyanarayan. 2022. Rich Screen Reader Experiences for Accessible Data Visualization. *Computer Graphics Forum* 41, 3 (2022), 15–27. doi:10.1111/cgf.14519

A Codebook

Here, we include our final set of codes for our analysis, each organized into its subcategory.

- Disability Type
  - Aging
  - Cognitive
  - Deaf/hard of hearing
  - Fatigue
  - Motor
  - Neurological
  - Sensory processing
  - Situational/Contextual abilities
  - Speech
  - Vision
- I/O Modalities, Technologies, and Actions
  - AR
  - Audio
  - Clicking
  - Dragging
  - Error Recovery
  - Eye-tracking
  - Feedback
  - Gestures
  - Haptics
  - Keyboard
  - Mouse
  - Pinching
  - Pointing
  - Scrolling
  - Text entry
  - Touch
  - Trackpad
  - Verification
  - Voice
- Accessibility Features/Technology
  - Accessibility checkers
  - Accessibility settings
  - Accessibility/Ability data
  - Adaptability
  - Alt text
  - Alternative I/O
  - Alternative layout
  - Alternative navigation
  - Assistive devices
  - Braille
  - Captions/Transcriptions
  - Community accessibility input
  - Help
  - Personalization/Customization
  - Product related service accessibility
  - Screen readers
  - Shortcuts
  - Smart input features
  - Zooming/Magnification
- Accessibility Quality
  - Accessibility preference
  - Burden
  - Equitable/Inequitable experience
  - Incorrect/incomplete accessibility feature
  - Number of modalities

  - Number of pathways
  - Paywalled accessibility features
  - Scope of accessibility settings
  - Scope of target disabilities groups
- Element Features
  - Brightness
  - Color
  - Contrast
  - Effect
  - Size
- UI Elements
  - Animation
  - Charts/Graphs
  - Icons
  - Images
  - Interactive widgets
  - Tables
  - Text
  - Visual content
- Layout
  - Consistency
  - Density
  - Flexibility
  - Hierarchy
  - Navigation
  - Simplicity
- Other
  - Data privacy
  - Focus
  - Repetition
  - Time
  - Triggers