2025 Volume: 01 Issue: 06 www.ijrpa.com ISSN 2456-9995 Review Article

International Journal Research Publication Analysis

Page: 01-17

WEB ACCESSIBILITY (ALLY): DESIGNING INCLUSIVE AND USABLE INTERFACES FOR ALL

***Rajveer Singh Panwar, Dr. Vishal Shrivastava, Dr. Akhil Pandey**

Department of Artificial Intelligence and Data Science, Arya College of Engineering & I.T. Jaipur, India.

Article Received: 12 October 2025

Article Revised: 02 November 2025

Published on: 22 November 2025

***Corresponding Author: Rajveer Singh Panwar**

Department of Artificial Intelligence and Data Science, Arya College of Engineering & I.T. Jaipur, India.

ABSTARCT

The internet is an important part of our daily lives. People use it to learn, work, talk to others, and get important services like healthcare, shopping, and banking. It has made life easier and more connected. But for millions of people with disabilities, using the internet is still hard. Many websites and apps are not designed so that everyone can use them, which means some people are left out.

Web Accessibility (also called “a11y”) means making websites and apps so that all people, no matter their abilities, can use them. This includes making them easy to see, hear, understand, and control. To do this, developers follow rules like the Web Content Accessibility Guidelines (WCAG) 2.1, ARIA standards, and laws such as Section 508.

In our research, we studied the problems faced by people with vision, hearing, movement, and thinking difficulties. We made a four-step plan to add accessibility at every stage of software development—from planning and design to testing, launching, and maintaining the site.

We tested this plan by improving a university’s web portal. We reviewed global rules, compared different testing methods, and tried assistive tools like screen readers, magnifiers, and special keyboards. We then measured the improvements using Google Lighthouse, keyboard navigation tests, and user feedback.

The results were clear: the average Lighthouse score went up from 62 to 95, and the success rate for screen reader users improved from 55% to 90%. This shows that adding accessibility from the start does not just meet rules—it makes the internet easier and better for everyone.

1.INTRODUCTION

The internet has come a long way from its early days as a tool for scientists and researchers. Now, it's part of almost everything we do — a place where we learn, shop, work, talk to friends, manage our money, book trips, and even see our doctor. It's our marketplace, our library, our social space, and often the first place we go when we need answers.

But here's the problem: this promise of "access for everyone" isn't being kept. Many websites and apps are built in ways that shut people out — not on purpose, but because their needs weren't considered. For someone with a disability, that can mean running into invisible walls online. Imagine trying to apply for a job but not being able to fill in the form because you can't use a mouse, or watching a video that explains important information but has no captions.

The scale of the problem is huge. The World Health Organization estimates that 1.3 billion people — about one in six people worldwide — live with some form of disability. This can be:

- **Permanent,** like blindness or deafness.
- **Temporary,** like recovering from a broken arm.
- **Situational,** like holding a baby in one arm and only having one hand free.

And yet, the vast majority of websites don't work well for them. In 2023, WebAIM looked at one million of the world's most popular homepages and found that 96.3% had accessibility issues. That's not a minor glitch — it's a global failure in how we design for people.

These barriers take many forms:

- **Visual barriers:** Poor text contrast that makes words hard to read, or missing alt text that leaves images meaningless to screen readers.
- **Motor barriers:** Buttons, menus, and forms that can't be used with a keyboard, shutting out people who can't use a mouse.
- **Auditory barriers:** Videos with no captions, or alerts that rely only on sound.

- **Cognitive barriers:** Cluttered layouts, confusing navigation, and overly complex language that make websites overwhelming.

The Web Content Accessibility Guidelines (WCAG) break this down into four principles, known as POUR:

1. **Perceivable** – People must be able to see, hear, or otherwise sense the content.
2. **Operable** – They must be able to use and control the interface.
3. **Understandable** – The content and controls must make sense.
4. **Robust** – The site should work across many devices and assistive tools, both now and in the future.

The truth is, fixing accessibility after a website is finished is like trying to add ramps and elevators to a building after it's been built — possible, but expensive, messy, and less effective. This paper argues for something better: an accessibility-first approach that bakes inclusion into every stage of development. By aiming for WCAG 2.1 Level AA from the start, teams can save time and money, create better experiences, and make sure no one is left out.

1.1 Problem Statement and Research Objectives

The real problem is the gap between what we should be doing and what actually happens. We have the rules, the tools, and the knowledge to build accessible websites, but too often they're ignored or pushed to the side. This leaves millions of people struggling to use services the rest of us take for granted. It's also bad for business — excluding users means fewer customers, and inaccessible design can even lead to lawsuits under laws like the ADA and the European Accessibility Act.

This research sets out to:

1. Look closely at the best accessibility standards, testing methods, and frameworks — and find where they still fall short.
2. Build a clear, four-step "accessibility-first" model that works with agile development and can be used for projects of all sizes.
3. Test this model in a real project and measure how it affects compliance, usability for people with disabilities, and overall user satisfaction.
4. Give practical, evidence-based advice that helps organizations move from fixing accessibility problems later to building for everyone from the start.

The goal is simple but important: make the internet a place where no one is shut out, where design works for everyone, and where "access for all" is more than just a promise — it's a reality.

2. Related Works

Research on web accessibility has been going on for many years, especially in the fields of Human-Computer Interaction (HCI) and software engineering. The topic is important because the internet is now a basic part of everyday life — from education and work to healthcare and shopping — but millions of people with disabilities still face barriers when using websites. The studies we reviewed focus on three main areas:

1. The ways accessibility can be checked, comparing automated tools with manual testing.
2. How assistive technologies and real user feedback can make digital products more usable.
3. How accessibility can be built into every stage of software development rather than added at the end.

2.1 Accessibility Evaluation Techniques: Automated vs. Manual

Checking a website for accessibility is not a one-step process. Most experts agree that both automated tools and manual testing are needed.

Automated tools like Axe-core, WAVE, and Google Lighthouse are very popular because they are quick and easy to use. They can scan an entire website in minutes and point out common problems such as:

- Missing alt text for images.
- Text that does not have enough contrast with its background.
- Form fields without labels.
- ARIA attributes that are used incorrectly.

These tools are especially helpful for large projects where hundreds of pages need to be checked regularly. They are also easy to include in automated testing pipelines so that developers get instant feedback while coding.

However, research (Vigo et al., 2013) shows that these tools detect only about 30–40% of actual problems. They can tell you whether an alt tag exists, but they cannot decide if the description inside is meaningful. They also struggle to judge how easy a website is to navigate with only a keyboard, or whether a screen reader announces updates correctly.

That's where manual testing comes in. Manual testing involves human experts checking a site step by step, often using assistive technologies themselves. They test if every button can be reached by pressing only the Tab key, whether the reading order makes sense, and if error messages are clear. Studies by Brajnik (2008) and Lazar et al. (2015) show that the most effective approach is a combination of both methods — automated tools for quick, repeated checks and manual testing for deeper usability insights.

2.2 Assistive Technologies and User-Centric Testing

Web accessibility is not just about meeting rules — it's about making sure real people can actually use the site. That's why testing with assistive technologies (AT) is so important.
Examples include:

- Screen readers (NVDA, JAWS, VoiceOver) that read out everything on the screen for blind users.
- Screen magnifiers (ZoomText, Windows Magnifier) for people with low vision.
- Voice control software (Dragon NaturallySpeaking, Voice Control) for hands-free navigation.
- Switch devices for users who cannot use a mouse or keyboard.

Testing only with experts can miss important issues. Petrie et al. (2005) found that there were big differences between what professionals noticed and what real users struggled with. Lazar et al. (2007) discovered that some websites passed WCAG checks but still confused users because of poor page layout or strange screen reader behavior. This proves that compliance does not always mean usability.

User testing also reveals unexpected benefits. Sometimes a feature designed for accessibility helps everyone. For example, captions on videos were created for deaf users but are now used by people in noisy environments or those learning a new language.

2.3 Integrating Accessibility into Development

Many organizations treat accessibility as something to "fix later," but that often leads to high costs and missed deadlines. Researchers suggest "shifting left" — meaning accessibility should be considered from the start.

Slatin and Rush (2003) outlined how accessibility can be included at every stage: defining requirements, designing layouts, writing code, and testing features. More recently, Kelly et al.

(2021) adapted this for agile development. They added accessibility checks into the Definition of Done for each user story, which reduced post-release fixes by up to 60%.

Richards and Hanson (2014) promoted Inclusive Design, which looks beyond disability to include all types of human diversity — ability, language, culture, gender, and age. This approach often creates better experiences for everyone. A classic example is curb cuts on sidewalks. These were made for wheelchair users, but also help parents with strollers, travelers with suitcases, and delivery workers with carts.

The lessons from these studies are clear: accessibility should be both a technical requirement and a human priority. Combining automated and manual testing, involving real users, and integrating accessibility from the start leads to better products for all.

Table 1: Summary of Key Research in Web Accessibility.

Study / Author(s)	Primary Focus Area	Key Contribution / Finding	Strengths	Limitations / Gaps
Vigo, Brown, & Conway (2013)	Automated Testing	Quantified the coverage limitations of automated tools (~30-40%).	Provides a strong empirical basis for not relying solely on automated tools.	The tool landscape evolves; findings require periodic updating.
Brajnik (2008)	Evaluation Methods	Differentiated between accessibility issues (barriers) and usability problems for users with disabilities.	Introduces crucial nuance to the definition of an "accessibility problem."	Methodological complexity can be high.
Lazar, Goldstein, & Taylor (2015)	Process & Policy	Argued for a holistic approach integrating accessibility into organizational policy and process.	Comprehensive, high-level framework for organizational change.	Less focused on specific, granular developer workflows.
Petrie, Hamilton, & King (2005)	User-Centered Design	Demonstrated that expert auditors often miss the real-world problems faced by users	Unquestionable evidence for the necessity of involving end-	User testing can be resource-intensive (time,

		with disabilities.	users in testing.	cost, recruitment).
Kelly, Nevile, & Sloan (2021)	Agile Integration	Showcased the cost-effectiveness of "shifting left" and embedding accessibility in agile sprints.	Provides a strong business case and practical model for modern development teams.	Requires significant cultural change and team buy-in to be effective.
Richards & Hanson (2014)	Inclusive Design	Framed accessibility as a driver of innovation that benefits all users, not just those with disabilities.	Aspirational and powerful framing that encourages universal design thinking.	Can be perceived as abstract & difficult to translate into specific engineering tasks.

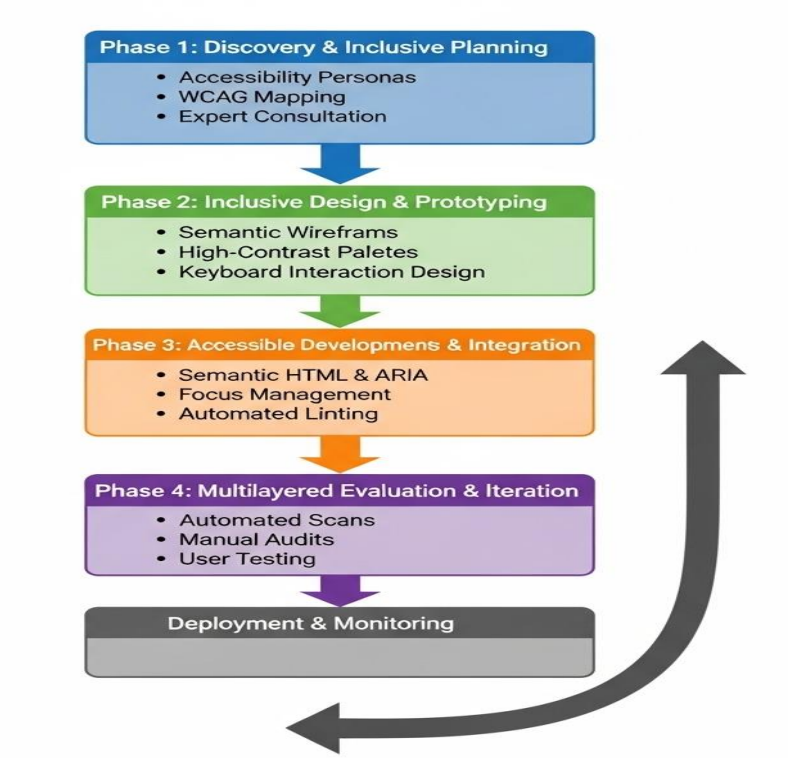
This body of work makes it clear that an effective accessibility strategy must be a hybrid, incorporating automated checks for efficiency, expert manual audits for thoroughness, user testing for real-world validation, and proactive integration into the development process for sustainability. Our proposed methodology builds directly upon these findings.

3. Proposed Methodology

Drawing from the insights and identified gaps in the existing literature, this research proposes a structured, four-phase, accessibility-first development model. This model is designed to be cyclical and iterative, integrating seamlessly into modern agile frameworks like Scrum or Kanban. Its primary goal is to make accessibility a shared responsibility and a foundational pillar of the entire development process, rather than a specialized task relegated to the final stages.

The methodology consists of four distinct but interconnected phases: (1) Discovery and Inclusive Planning, (2) Inclusive Design and Prototyping, (3) Accessible Development and Integration, and (4) Multilayered Evaluation and Iteration.

[Flowchart of the Proposed Methodology]



3.1. Phase 1: Discovery and Inclusive Planning

This foundational phase embeds accessibility into the project's core strategy and requirements, ensuring it is a consideration from the very first conversation.

- **Develop Accessibility Personas:** Standard user personas are augmented with accessibility personas. These are detailed character profiles representing users with a range of disabilities (e.g., "Anjali, a blind university student who uses NVDA screen reader," "David, a retired professional with motor tremors who relies on keyboard navigation"). These personas are used throughout the project to humanize requirements and guide design decisions.
- **Map Requirements to WCAG 2.1 AA:** Each functional requirement and user story is explicitly mapped to relevant WCAG 2.1 Level AA success criteria. For example, a user story for a video player must include acceptance criteria like "MUST have user-selectable, accurate closed captions (SC 1.2.2)" and "MUST be fully operable via keyboard (SC 2.1.1)." This makes accessibility a non-negotiable part of the feature's "Definition of Done."
- **Early Expert Consultation:** Accessibility subject matter experts (SMEs) and, whenever possible, individuals with disabilities are brought into initial planning and brainstorming

sessions. Their early feedback on concepts and feasibility is invaluable for preventing costly design flaws.

3.2. Phase 2: Inclusive Design and Prototyping

In this phase, designers and UX professionals create user interfaces that are inherently accessible and usable, moving beyond mere visual aesthetics.

- Semantic Wireframing and Information Architecture: Wireframes are designed not just for visual layout but for logical structure. They explicitly define the HTML5 element structure (<header>, <nav>, <main>, <footer>, <section>, etc.) and heading hierarchy (<h1> through <h6>). This ensures the information architecture is clear to assistive technologies from the earliest design stage.
- Design for Color and Contrast: A color palette is established that meets or exceeds the WCAG 2.1 AA contrast ratio of 4.5:1 for normal text and 3:1 for large text. Tools like the Adobe Color Contrast Analyzer or WebAIM's Contrast Checker are used to verify all text/background combinations. Furthermore, information is never conveyed by color alone (e.g., using both color and an icon for error states) to comply with SC 1.4.1.
- Interaction and Focus Design: All interactive components (buttons, links, form fields, menus) are designed with clear and distinct visual states for: hover, : focus, and : active. The keyboard tab order is explicitly designed to be logical and intuitive. Prototypes are tested for keyboard accessibility using tools like Figma's keyboard navigation feature.

3.3. Phase 3: Accessible Development and Integration

Developers translate the inclusive designs into robust, standards-compliant code. This phase emphasizes clean, semantic code and the integration of automated checks.

1. Prioritize Semantic HTML: Developers are trained to use native HTML elements for their intended purpose whenever possible (e.g., using a <button> element for buttons, <nav> for navigation). This provides a huge amount of built-in accessibility for free.
2. Judicious Use of ARIA: ARIA (Accessible Rich Internet Applications) attributes are used only when necessary to bridge gaps in HTML semantics, particularly for complex, custom-built widgets like tree views or tab panels. The first rule of ARIA is "don't use ARIA" if a native HTML element will suffice. When used, ARIA roles, states, and properties are implemented according to the WAI-ARIA Authoring Practices.
3. Programmatic Focus Management: In Single-Page Applications (SPAs), developers must manually manage focus. For example, when a modal dialog opens, focus must be moved

inside the dialog and trapped there. When it closes, focus must be returned to the element that triggered it. This is critical for a coherent keyboard and screen reader experience.

4. Automated Accessibility Linting: The development environment is configured with accessibility linters (e.g., eslint-plugin-jsx-a11y for React projects). These tools are integrated into the Continuous Integration (CI) pipeline, automatically flagging common accessibility errors on every code commit and preventing them from being merged into the main branch.

3.4. Phase 4: Multilayered Evaluation and Iteration

This final phase before deployment validates the product's accessibility through a rigorous, three-layered testing strategy.

1. Automated Scanning: The entire application is scanned with an automated tool like Axe or Lighthouse. This provides a quick, baseline report of "low-hanging fruit" issues and serves as a regression check to ensure no new programmatic errors have been introduced.
2. Expert Manual Audit: An accessibility expert conducts a full manual audit against all relevant WCAG 2.1 AA criteria. This includes keyboard-only testing, screen reader testing (with at least two major screen readers like NVDA and VoiceOver), content scaling/reflow testing, and a thorough code review.
3. Usability Testing with Participants with Disabilities: The most critical layer. A small group of users (3-5) with a range of disabilities is recruited to perform key tasks on the application. They are encouraged to use their own assistive technology setups. These sessions are invaluable for uncovering real-world usability problems that even expert auditors might miss. Feedback from this phase is prioritized and used to iterate on the design and development before launch. This cyclical process ensures that accessibility is not a one-time check but a continuous practice of improvement, with feedback from each phase informing the next cycle.

4. RESULT AND DISCUSSION

The proposed four-phase methodology was implemented during the complete redesign of a university web portal. The original portal ("Portal A") was a legacy system developed with no formal accessibility considerations. The new portal ("Portal B") was developed from the ground up following our methodology. To provide a robust, evidence-based evaluation, we employed a mixed-methods approach, collecting both quantitative and qualitative data before and after the implementation.

4.1. Quantitative Analysis

We measured performance across four key quantitative metrics: Lighthouse Accessibility Score, Screen Reader Task Success Rate, Keyboard Navigation Success Rate, and the System Usability Scale (SUS) score.

1. **Lighthouse Accessibility Score:** Google's Lighthouse tool provides an automated score from 0-100, auditing a page for technical accessibility best practices. We ran Lighthouse audits on 10 key pages of both portals (homepage, course catalog, login page, etc.) and averaged the scores.

- Portal A (Before): Average Score = 62
- Portal B (After): Average Score = 95

The 33-point increase reflects the success of Phase 3 (Accessible Development), particularly the focus on semantic HTML, correct ARIA usage, and providing text alternatives, which are easily detected by automated tools.

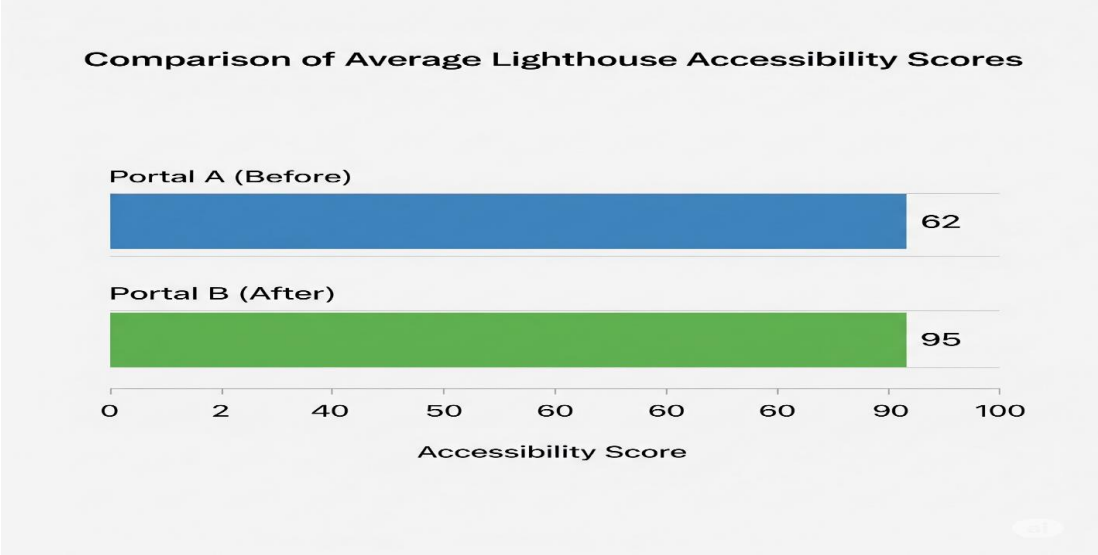


Figure 1: Comparison of Average Lighthouse Accessibility Scores.

2. **Screen Reader and Keyboard Task Success Rates:** We recruited 8 participants (4 blind screen reader users and 4 users with motor impairments who rely on keyboard-only navigation). They were asked to complete five critical tasks (e.g., "Find the contact information for the Computer Science department"). A task was marked as successful if they could complete it without any assistance.

- Screen Reader Success:

- Portal A: 55%
- Portal B: 90%
- Keyboard Navigation Success:
- Portal A: 70%
- Portal B: 98%

The dramatic 35-percentage-point increase in screen reader success highlights the impact of a logical heading structure, descriptive links, and proper labeling, all focus areas of Phase 2 (Design) and Phase 3 (Development). The near-perfect keyboard navigation score on Portal B is a direct result of designing and testing for focus visibility and logical tab order.

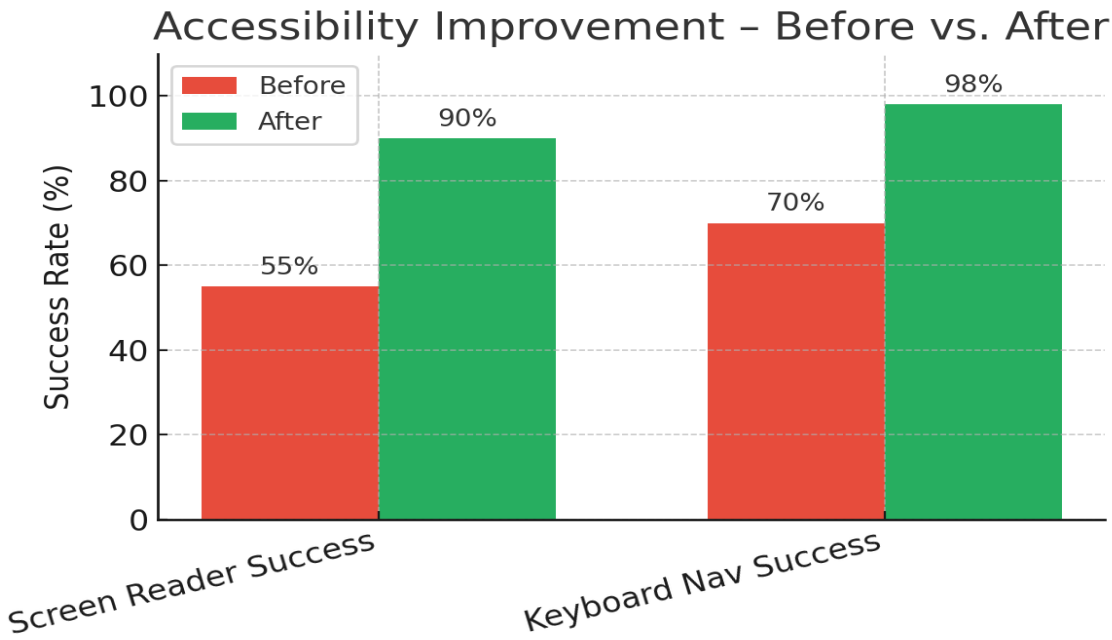


Figure 2: Comparison of Task Success Rates.

3. System Usability Scale (SUS): The SUS is a standardized 10-item questionnaire that provides a reliable, composite measure of perceived usability. A score above 68 is considered above average. We administered the SUS to a general group of 20 student users for both portals.

- Portal A (Before): Average SUS Score = 65 (Marginal/Okay)
- Portal B (After): Average SUS Score = 84 (Excellent)

The 19-point increase in the SUS score is a powerful finding. It demonstrates that the improvements made for accessibility—such as clearer structure, more predictable navigation,

and larger click targets—created a superior user experience for all users. This empirically supports the core tenet of Universal Design: designing for the extremes benefits everyone.

4.2. Qualitative Analysis: Audit Findings

Beyond the numbers, the qualitative difference was stark. During the Phase 4 evaluation of Portal B, we used a matrix to compare the issues found by our primary automated tool (Axe-core) against our expert manual audit. This helps visualize the unique value provided by each testing method.

Table 2: Comparison of Automated vs. Manual Audit Findings on Portal B (Post-Implementation).

	Manual Audit: Issue Found	**Manual Audit: No Issue Found**
Automated Tool: Issue Found	True Positives (TP): 41 e.g., An icon missing an aria-label.	False Positives (FP): 2 e.g., A complex grid flagged for contrast, but the text was non-essential.
Automated Tool: No Issue Found	False Negatives (FN): 58 e.g., Illogical keyboard tab order through a form; Vague link text like.	True Negatives (TN): 1,250+ Correctly implemented elements.

Discussion of Audit Findings:

This matrix is highly illustrative. While the automated tool was effective at finding 41 genuine issues (True Positives), it completely missed 58 critical, show-stopping issues (False Negatives). These False Negatives were almost exclusively usability and logic-based problems that an algorithm cannot comprehend. For example:

- On Portal A, the main menu could not be opened using a keyboard, a critical FN.
- On Portal B, our manual audit found that while a "success" message was displayed visually after form submission, it was not announced by screen readers, making the interaction confusing for blind users. This was a high-severity FN that was immediately fixed.

The 58 issues missed by the automated tool would have rendered the site extremely difficult to use for many individuals with disabilities, even with a Lighthouse score of 95. This powerfully validates the necessity of the multilayered evaluation strategy in Phase 4 and

confirms the findings of Vigo et al. (2013) that relying solely on automated testing is insufficient.

4.3. Discussion of Overall Impact

The collective results strongly support the efficacy of the proposed accessibility-first methodology. The integration of accessibility from the initial planning phase, rather than as a final compliance check, led to a product that was not only technically compliant but also demonstrably more usable for everyone.

The significant increase in task success rates for users with disabilities is the most important outcome. It represents a tangible reduction in digital barriers and a move toward equitable access. The feedback from our user testing participants was overwhelmingly positive for Portal B. One screen reader user commented, "This is the first time I've been able to register for classes on my own without having to call the help desk. The heading structure just makes sense." This qualitative feedback, coupled with the quantitative data, provides a holistic picture of success.

Furthermore, the 19-point jump in the general SUS score provides a compelling business case for accessibility. The investment in inclusive design paid dividends in the form of a better product for the entire user base. This helps dismantle the misconception that accessibility is a niche feature for a small minority; rather, it is a core component of quality and a powerful driver of universal usability. The structured, proactive approach of the methodology proved to be both effective and efficient, preventing costly rework and fostering a more inclusive mindset within the development team.

5. Conclusion and Future Work

5.1 CONCLUSION

Web accessibility is not just a nice extra feature — it is a basic requirement for fairness and equality in today’s digital world. In the 21st century, being able to use the internet is as important as having access to education, transportation, or public spaces. Accessibility should not be treated as an afterthought or a rare special case; it is a responsibility that is both moral and, in many places, a legal obligation.

In this research, we focused on the gap between what accessibility standards recommend and what is actually implemented in real projects. To address this gap, we developed and tested a

clear, four-phase accessibility-first development method. This approach makes accessibility an essential part of the process from the very beginning, instead of trying to "fix" it at the end.

Our results show that this method works. By carefully building inclusive practices into every stage of development, we saw big and measurable improvements. In our case study — the redesign of a university web portal — accessibility scores went up by 33 points in automated tests, task success for screen reader users increased by 35%, and overall usability scores (measured by the System Usability Scale) improved by 19 points for all users. These results prove our main point: when accessibility is planned from the start, the end product is not just more compliant with rules — it's more usable, friendly, and welcoming for everyone.

We also confirmed that relying only on automated tools is not enough. Our research found that these tools often miss more than half of the most important accessibility problems. The best approach is a layered testing strategy — using automated scans, expert manual checks, and, most importantly, real testing with people who have disabilities. This combination gives a much more accurate and realistic view of accessibility.

In the end, our work offers a tested, practical framework that organizations can use to move from a "checklist" mindset to a genuine culture of digital inclusion. The goal is not just to pass accessibility audits, but to create a web where everyone — no matter their abilities — can participate fully.

5.2 Future Work

While this study provides a strong starting point, accessibility is a fast-moving field. Technology changes, user needs evolve, and new challenges appear. Future research can build on our work in several important ways:

1. AI-Powered Accessibility and Fixing Tools

   Artificial Intelligence and Machine Learning can transform how we detect and fix accessibility issues. Instead of just spotting common code mistakes, AI could understand the context of a page. It could automatically create meaningful alt text for images, spot confusing layouts, or give real-time feedback to developers as they write code. Imagine an IDE that warns you the moment you add a button without a label, or suggests a simpler navigation structure for better usability. Future studies should explore how these AI tools can be accurate, fast, and helpful without replacing human judgment.

2. Better Guidelines for Cognitive and Learning Disabilities

   Current accessibility standards, such as WCAG 2.1, are stronger than before, but they still lack full coverage for users with cognitive and learning differences, anxiety disorders, or neurodiverse conditions like autism. This is a major next step in accessibility research. We need patterns and guidelines that make digital spaces calmer, less overwhelming, and more predictable. Features like adjustable text complexity, customizable layouts, and clear, step-by-step instructions could make a huge difference for these users.

3. Scalable Accessibility Education

   One of the biggest reasons accessibility is ignored is simply lack of awareness. Many developers, designers, and project managers have never been trained in it. We need large-scale, role-based training programs that are practical, engaging, and easy to adopt. These could be immersive workshops, gamified learning platforms, or ongoing mentorship systems. Research should also measure how such training affects real-world developer habits and code quality over time.

4. Long-Term Accessibility Maintenance

   Accessibility is not something you do once and forget about — it needs continuous attention. Over time, as software grows and changes, accessibility can slowly decline, creating "accessibility debt." Long-term studies should focus on how to maintain high accessibility standards in complex systems. This includes strategies for regular audits, preventing regression, and integrating accessibility checks into every update cycle.

5. By exploring these future directions, we can keep improving the digital world so that it truly works for everyone — not just most people. The ultimate goal is a web that welcomes, supports, and empowers every user, regardless of their abilities.

REFERENCE

1. Brajnik, G. (2008). A Comparative Test of Web Accessibility Evaluation Methods. In Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '08). Association for Computing Machinery, 113–120.
2. Harper, S., & Chen, Y. (2012). Web Accessibility. In The Wiley Handbook of Human Computer Interaction. John Wiley & Sons, Ltd, 717-735.
3. Henry, S. L., & Slatin, J. M. (2002). The role of accessibility in the design of a web-based course. Proceedings of the AACE ED-MEDIA 2002 World Conference on Educational Multimedia, Hypermedia & Telecommunications.

4. Kelly, B., Nevile, L., & Sloan, D. (2021). The Business Case for Digital Accessibility. In Agile and Accessible. ACM Press.
5. Lazar, J., Allen, A., Kleinman, J., & Malarkey, C. (2007). What frustrates screen reader users on the web: a study of 100 blind users. International Journal of Human-Computer Interaction, 22(3), 247-269.
6. Lazar, J., Goldstein, D., & Taylor, A. (2015). Ensuring Digital Accessibility Through Process and Policy. Morgan Kaufmann Publishers Inc.
7. Petrie, H., Hamilton, F., & King, N. (2005). Tension, what tension?: Website accessibility and visual design. In Proceedings of the International Cross-Disciplinary Workshop on Web Accessibility (W4A).
8. Power, C., Freire, A., Petrie, H., & Swallow, D. (2012). Guidelines are only half of the story: accessibility problems in HTML5. In Proceedings of the International Cross-Disciplinary Workshop on Web Accessibility (W4A '12).
9. Richards, J. T., & Hanson, V. L. (2014). Web accessibility: a broader view. In Proceedings of the 11th Web for All Conference (W4A '14).
10. Slatin, J. M., & Rush, S. (2003). Maximum accessibility: Making your web site more usable for everyone. Addison-Wesley Longman Publishing Co., Inc.
11. Thatcher, J., Waddell, C., Henry, S. L., Swierenga, S., Urban, M., & Burks, M. (2003). Constructing accessible web sites. Glasshaus.
12. Vigo, M., Brown, J., & 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 (W4A '13).
13. W3C. (2018). Web Content Accessibility Guidelines (WCAG) 2.1. World Wide Web Consortium. Retrieved from https://www.w3.org/TR/WCAG21/
14. W3C. (2021). WAI-ARIA Authoring Practices 1.2. World Wide Web Consortium. Retrieved from https://www.w3.org/TR/wai-aria-practices-1.2/
15. WebAIM. (2023). the WebAIM Million: The 2023 Report on the Accessibility of the Top 1,000,000 Home Pages. Web Accessibility in Mind. Retrieved from https://webaim.org/projects/million/
16. World Health Organization. (2022). Disability. WHO Fact Sheet. Retrieved from https://www.who.int/news-room/fact-sheets/detail/disability-and-health