International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

# Artificial Intelligence for Accessibility and Performance Auditing: Automated Findings with Human Judgment

**Harshit Sunilkumar Vora**

**Abstract:** Automated accessibility and performance auditing tools have become integral to modern web development pipelines, yet systematic evidence shows that treating their outputs as definitive conformance verdicts leads to programs that are overconfident in coverage and underinvest in expert judgment. Deterministic rule engines reliably surface structural defects at scale but remain fundamentally constrained in their ability to evaluate success criteria requiring semantic interpretation, contextual reasoning, or natural language understanding. Established standard frameworks—structured around principles of perceivability, operability, understandability, and robustness—provide the normative foundation against which both automated and human findings must be mapped to remain institutionally credible and legally defensible. Performance auditing presents a structurally parallel set of challenges, where threshold-based metrics require human disambiguation before remediation decisions can be responsibly made. The empirical boundaries of automated detection are quantified through mutation testing and coverage analysis, confirming that no single tool is sufficient and that tools are structurally complementary rather than interchangeable. Artificial intelligence augmentation extends automated coverage into semantically demanding criteria, achieving meaningful detection rates that conventional rule engines cannot approach, while introducing anchoring risks that demand carefully designed human-in-the-loop workflows. A continuous auditing pipeline with graded confidence tiers — separating high-confidence structural findings, medium-confidence semantic assessments, and low-confidence interaction-dependent evaluations — provides the operational architecture necessary to allocate expert attention proportionally, measure program quality over time, and produce findings that are auditable, reproducible, and defensible across tool versions and evaluation cycles.

***Keywords:*** *Web Accessibility Auditing, WCAG Conformance Evaluation, Automated Detection Coverage, Large Language Model Augmentation, Continuous Auditing Pipeline*

## 1. The Incomplete Truth Behind Automated Auditing

Modern web development ecosystems have come to depend mostly on automated tooling to surface accessibility and performance problems within continuous delivery pipelines. The demand is clear: automation scales smoothly across big codebases, performs reliably on every change, and produces machine-readable results that incorporate cleanly with developer workflows. Yet the evidence suggests that treating these outputs as definitive verdicts rather than as probabilistic signals demanding structured interpretation produces programs that are systematically overconfident in their coverage and underinvested in expert judgment.

*Independent Researcher, USA*

Large-scale empirical evaluations of government web platforms reveal just how persistent and widespread this gap remains. A population-level automated assessment of all 50 U.S. state government homepages found that 90% were rated "Non-compliant" by a proprietary accessibility algorithm, with common detectable barriers including a mean of 8.4 color contrast errors per page and a mean heading structure score of only 68.1 out of 100—indicating widespread structural deficiencies across high-visibility public-sector entry points [1]. Note: [1] is a non-peer-reviewed preprint whose compliance methodology relies on a tool-specific heuristic with no regulatory standing; findings should be interpreted accordingly. Critically, not a single homepage was entirely free of detectable issues, underscoring that even well-maintained, high-priority pages fall measurably short of programmatic accessibility standards.

The paradox is that automation is simultaneously too successful and too limited. It reliably identifies large volumes of structural defects, missing label associations, insufficient contrast ratios, undeclared document languages, and empty interactive controls at a scale no human review program could match. Yet it is too limited because the issues it detects represent only a fraction of the success criteria that define genuine conformance. Controlled mutation-based evaluation demonstrates this limitation precisely: across 366 intentionally injected accessibility bugs tested against six widely used automated tools, those tools collectively failed to detect nearly 50% of injected defects, with individual tool mutation scores ranging from as low as 14% to a maximum of 52% [2].

This creates a governance risk that is often underappreciated. A more defensible framing treats automated outputs as graded-confidence hypotheses—ranked signals of varying evidential strength requiring structured triage, expert disambiguation, and measurable quality controls. This framing does not diminish the value of automation; it situates it correctly within a broader evaluation architecture where human judgment is not optional overhead but an essential epistemic component. Programs that conflate automated detection with conformance assurance will systematically miss high-impact barriers that escape detection entirely.

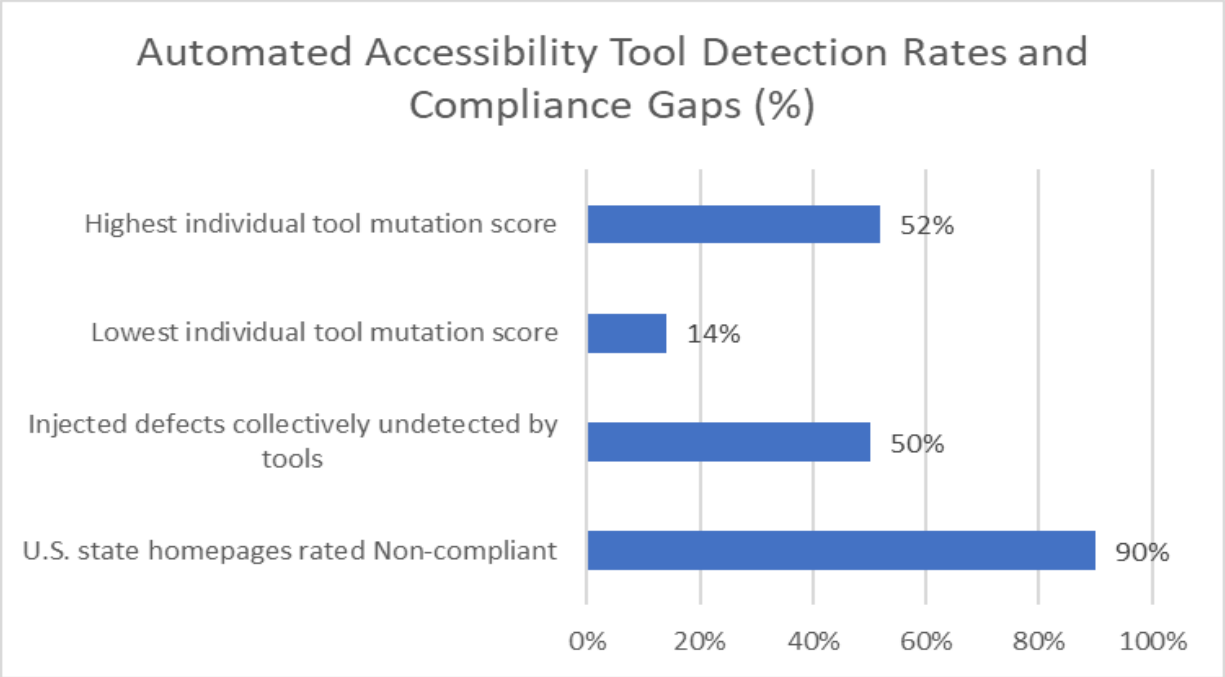


**Figure 1: Percentage-Based Accessibility Compliance and Mutation Score Metrics [1, 2]**

## 2. Standards Frameworks as the Foundation for Structured Auditing

Reliable accessibility and performance auditing require a normative anchor—a shared and formally specified definition of what conformance means, against which both automated and human findings can be mapped. Without such an anchor, audit programs risk producing outputs that are internally consistent but externally meaningless: findings that cannot be independently verified, compared across tool versions, or defended in procurement, litigation, or regulatory contexts. Established standards ecosystems provide this anchor, and understanding their structure is essential to designing auditing programs that are both technically rigorous and institutionally credible.

The most widely referenced standards baseline for web accessibility conformance is a hierarchical specification built on 4 principles—perceivability, operability, understandability, and robustness—decomposed into 13 guidelines and further into testable success criteria [3]. These standards are classified across 3 different conformance levels (A, AA, and AAA), with every tier representing conformance levels. A particularly consequential threshold is the minimum contrast ratio of 4.5:1

required for normal text under Level AA conformance, dropping to 3:1 for large-scale text—a measurable, machine-verifiable requirement that forms one of the most frequently automated checks in rule-based scanning pipelines [3]. For auditing purposes, each criterion functions as a discrete contract: a finding is meaningful only when it maps to a specific criterion, cites the implicated conformance level, distinguishes a confirmed violation from an advisory warning, and preserves evidence for independent verification.

A complementary methodology for software accessibility conformity assessment structures the evaluation process across 4 formal functions: selection, determination, review and attestation, and surveillance [4]. This framework is directly applicable to continuous auditing programs because it clarifies that evaluation is not a single event but a repeatable, governed process. Experimental experience with this technique has shown important inter-evaluator inconsistency when checkpoints are applied without comprehensive testing practices, an outcome that strengthens the requirement of structured workflows. Practical implementations have generated checklists of 39 evaluation items drawn from combined accessibility standards, with 5 possible outcome values per checkpoint: pass, fail, partial, unknown, and not applicable [4].

The operational implication is that a mature auditing program explicitly distinguishes three tiers of findings—rule engine detections, model-assisted semantic assessments, and validated findings confirmed through structured human review—each carrying different evidence requirements, confidence levels, and remediation urgencies. The standards ecosystem provides the vocabulary and governance structure necessary to maintain this differentiation rigorously across tool versions and evaluation cycles.

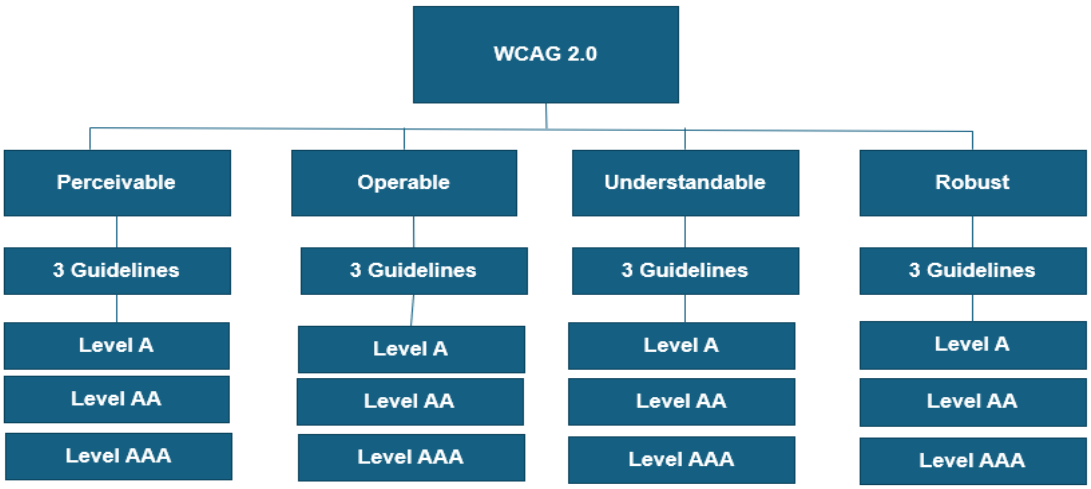


**Figure 2: WCAG Hierarchical Structure**

## 3. Performance Metrics as a Parallel Auditing Discipline

Web performance inspecting represents a structurally analogous set of problems to accessibility inspecting, making it a valuable parallel case to illustrate how automated results need human judgment rather than automatic action. In both domains, measurements depend on defective substitutions that can be over-trusted, collective thresholds can pass while user-visible harm continues, and automated signals mandate structured understanding rather than reflexive response.

Performance measurement has converged around field-measurable indicators targeting distinct dimensions of user experience: time to render the largest visible content element (LCP), the time from user interaction to the next visual update measured by Interaction to Next Paint (INP), and cumulative magnitude of unexpected layout shifts (CLS). These indicators were designed to be measurable from real user sessions at scale, making them superior to purely synthetic lab metrics as proxies for

experience. However, aggregate statistics reveal both the scale of the performance problem and the interpretive limitations of threshold-based measurement. Research on geoportal performance found that fully loaded times exceeded 10 seconds across all tested portals and surpassed 20 seconds in several cases, with PSI Performance scores remaining below 50 "poor" in the majority of measurements [6]. These findings are valuable for setting organizational targets, but cannot determine whether a specific regression is user-visible, caused by measurement artifacts, or attributable to third-party scripts. Such determinations require human judgment.

The parallel to accessibility is direct. Just as automated accessibility tools struggle to distinguish confirmed violations from noise—with studies showing only 35% of WCAG success criteria could be successfully tested by novice evaluators [5]—performance monitoring systems report threshold failures without distinguishing genuine regressions from measurement artifacts. In both cases, the automated signal is necessary but not sufficient.

Performance auditing also illustrates the importance of separating lab from field measurement, mirroring the accessibility distinction between controlled testing and real-world assistive technology configurations. A mature auditing program integrates both modalities, using human judgment to reconcile their outputs.

| Auditing Dimension | Accessibility Domain | Performance Domain |
|---|---|---|
| Measurement Proxy | WCAG Success Criteria | Core Web Vitals Indicators |
| Automated Signal Type | Violation Detection | Threshold Failure Reporting |
| Key Limitation | Semantic criteria untestable | Regression causes unattributable |
| Evaluator Dependency | Novice evaluator reliability gap | Measurement artifact ambiguity |
| Human Role | Expert confirmation | Judgment-based disambiguation |
| Testing Modality | Controlled vs. assistive technology | Lab vs. field measurement |

**Table 1: Accessibility and Performance Auditing — Structural Parallels [5, 6]**

## 4. Empirical Boundaries of Automated Accessibility Detection

The practical limits of automated accessibility auditing are not merely theoretical — they are quantified in peer-reviewed research using controlled experimental methodologies, and the numbers are consequential for how programs should be designed and governed.

One of the most direct methods for measuring automation's coverage limits is mutation testing: intentional injection of known accessibility defects into test pages, followed by measurement of how many injected defects are detected by automated tools. Research using this methodology has reported that automated tools fail to identify approximately half of deliberately introduced accessibility bugs [2]. This result has a precise operational interpretation: if a production codebase contains a representative population of accessibility defects, an automated scan passing without errors provides no better than coin-flip assurance that the codebase is defect-free.

Coverage analysis provides a complementary and equally striking picture. A metatesting study comparing nine automated accessibility tools across 121 web pages found that the most prolific tool reported roughly eight times as many issue instances as the least prolific—yet no single tool made the others redundant [8]. Each tool discovered numerous instances missed by all others, and adding a second tool to any single-tool configuration increased detected issue instances by anywhere from 13% to 767%, depending on the pairing [8]. Every tool also had at least seven issues that only it could detect, confirming that tools are structurally complementary rather than interchangeable.

Tool disagreement represents a third empirical dimension with direct operational implications. Programs relying on a single automated tool implicitly accept that tool's idiosyncratic blind spots as the boundary of their detection capability. Meanwhile, research on web performance optimization reminds us that the median mobile page size grew by nearly 600% between 2012 and 2022 [7], steadily expanding the surface area that accessibility tooling must cover—making the detection gap more consequential over time, not less.

The high volume of findings reported by large-scale scans deserves careful interpretation in this context. Automation provides genuine value for high-frequency structural issues, but a non-trivial proportion of flagged items—such as alternative text that is present but contextually meaningless—remains invisible to rule engines and requires semantic evaluation, a boundary that the following section addresses through artificial intelligence augmentation.

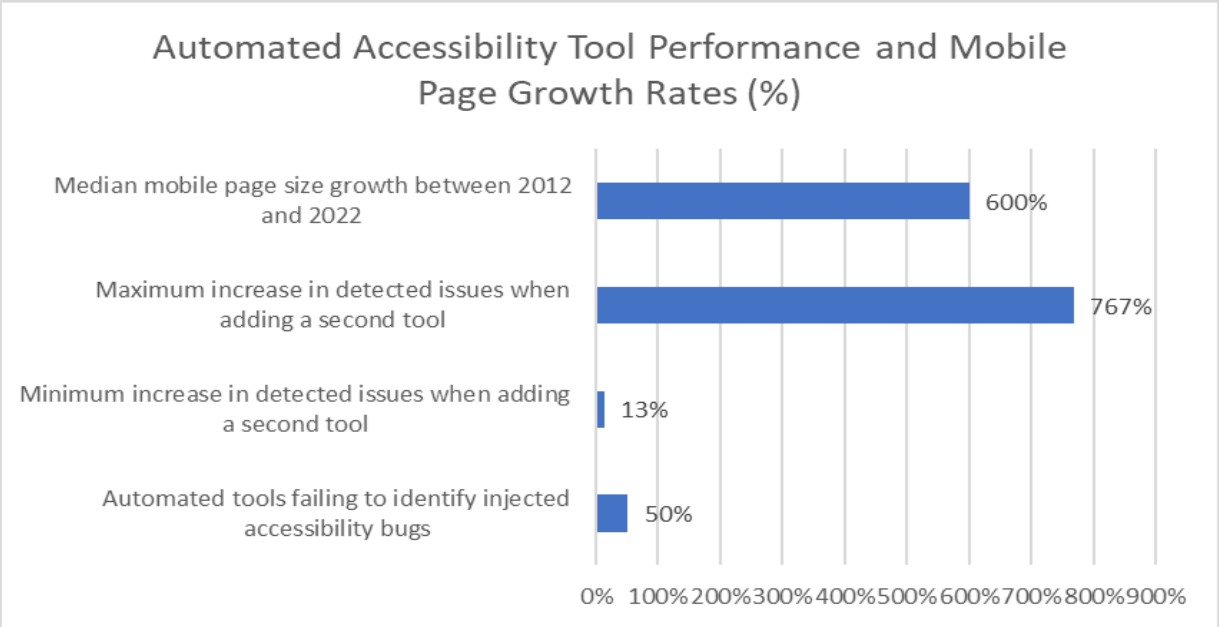


**Figure 3: Percentage-Based Detection Gaps and Coverage Gains in Automated Accessibility Tools (% [7, 8]**

## 5. The Role of Artificial Intelligence in Semantic Audit Augmentation

The limitations of deterministic rule engines create a well-defined opportunity for artificial intelligence augmentation: extending automated coverage into success criteria that require semantic interpretation, contextual judgment, or natural language understanding. Rather than replacing human expertise, this augmentation layer expands the boundary of what can be evaluated computationally—reducing the volume of issues requiring full human review by providing structured, evidence-backed hypotheses about semantic quality and concentrating expert attention where automated confidence is lowest.

Research applying large language model-based evaluation to specific accessibility success criteria demonstrates that this augmentation is practically achievable at meaningful accuracy levels. A study targeting three success criteria historically classified as requiring manual evaluation reported detection performance exceeding 87% across test cases, at a time when conventional automated evaluators achieved near-zero accuracy on intentional failures for those same criteria [11]. This represents a qualitative expansion of automated coverage—particularly significant given that automated tools currently cover only approximately one-sixth of all WCAG success criteria, with even the broadest combination of six engines covering only 39% of WCAG 2.1 success criteria [9].

Broader studies of generative AI applied to accessibility detection have identified violation types beyond those captured by rule-based tools when evaluated against real production websites [12]. For programs managing large surface areas, this represents a meaningful reduction in the proportion of the evaluation space requiring human review. Especially, manually evaluating a single webpage can take up to one hour depending on the complexity of the page [9]—a figure that highlights the scalability benefit AI augmentation delivers, especially when automated testing has been shown to decrease long-run expenses [10].

The hands-on deployment of AI in auditing pipelines needs proper attention to input quality and

output handling. The evaluation of model-based semantics performs reliably when offered with structured, evidence-rich inputs: accessibility tree extracts, DOM snapshots, and rule identifiers representing the criterion being calculated. When these inputs are absent or ambiguous, models produce outputs that may be superficially plausible but are not grounded in the specific evidence needed for auditable findings.

| Augmentation Dimension | Deterministic Rule Engines | AI Augmentation Layer |
|---|---|---|
| Coverage Scope | Structural and syntactic criteria | Semantic and contextual criteria |
| Evaluation Capability | Rule-based pattern matching | Natural language understanding |
| Human Review Dependency | Low for structural findings | Moderate with confirmation required |
| Output Type | Binary pass or fail signal | Graded-confidence hypothesis |
| Expert Attention Allocation | Distributed uniformly | Concentrated at the lowest confidence |
| Scalability Across Large Surfaces | High | High with reduced human burden |

**Table 2: AI Augmentation Layer — Purpose and Positioning in Accessibility Auditing [9, 11]**

## 6. Engineering a Continuous Auditing Pipeline With Graded Confidence

Translating the empirical findings reviewed in preceding sections into operational practice requires a pipeline architecture designed from the outset to reflect the true nature of automated outputs: probabilistic signals of varying evidential strength requiring structured triage, not binary verdicts requiring only remediation or dismissal. The architecture described here integrates deterministic rule engines, model-assisted semantic evaluation, and human review into a sequenced evidence-accumulation process with measurable quality controls at each stage.

The pipeline operates as an arrangement of gates, always producing structured evidence records rather than normal pass/fail results. A code change event activates automated scanning across different registered tool engines, with every engine producing results tagged with rule identifiers, impact element references, convenience tree snapshots, and rule version metadata. Results are deduplicated across different tools and clustered by the category of issue, with agreement across various independent engines contributing to confidence and disagreement flagging mandatory human review. High-confidence structural results are qualified for automated release gating, while unclear or semantic results enter a triage queue ordered by influence likelihood and confidence grade [8].

The confidence grading model assigns each finding to one of three tiers based on evidence strength, reproducibility, and impact likelihood. Structural findings with deterministic evidence occupy the high-confidence tier. Semantic findings—alternative text quality, link purpose evaluations, and language appropriateness judgments—occupy the medium-confidence tier and require human confirmation, a design choice grounded in evidence that LLM-based scripts achieved only 78%, 85%, and 100% accuracy across three individually tested success criteria, yielding an overall detection rate of 87.18% across 39 test cases [11]. Interaction-dependent findings occupy the low-confidence tier and require end-to-end assistive technology testing before conclusions can be drawn [9].

The triage rubric integrates the confidence tier with an impact severity dimension to produce prioritized action assignments. This rubric ensures expert time is allocated in proportion to both potential user impact and the uncertainty of the automated signal [10]. The pipeline must also measure its own quality over time. Key metrics include precision of automated findings confirmed as genuine issues—GenA11y demonstrated 94.5% precision across 3,302 reported violations on real websites—and recall proxies, where the same tool achieved 87.61% recall across 589 injected accessibility issues, detecting on average eight more violation types per page than all existing tools combined [12]. Rule sets and model prompts must be versioned with the same rigor applied to application code, enabling retrospective analysis and preventing drift that erodes the interpretability of long-running quality programs.

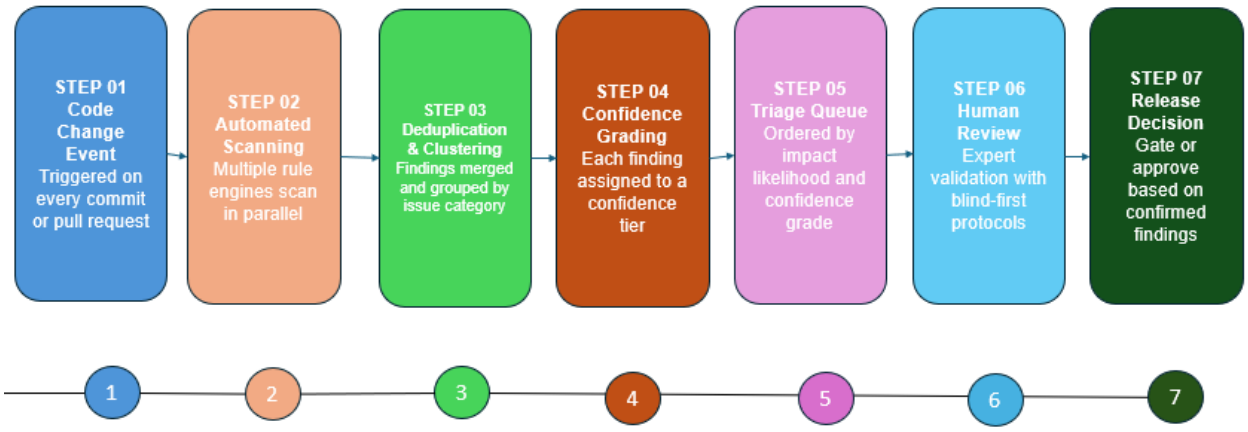


**Figure 4: Continuous Auditing Pipeline with Graded Confidence Tiers**

## Conclusion

Automated accessibility and performance auditing tools deliver genuine value within continuous delivery pipelines by surfacing structural defects at a scale and consistency no human review program could match, yet the evidence is unambiguous that their outputs represent graded-confidence hypotheses rather than conformance verdicts, and governance programs that conflate the two will systematically miss high-impact barriers that escape detection entirely. Established standards frameworks provide the normative vocabulary necessary to distinguish confirmed violations from advisory warnings and to map findings to specific conformance levels in ways that are independently verifiable and institutionally defensible. Performance auditing reinforces this lesson through parallel evidence: aggregate threshold measurements require human disambiguation before remediation decisions can be responsibly made, mirroring the same interpretive obligations that apply to accessibility signals. The empirical boundaries of automated detection—documented through mutation testing, coverage analysis, and cross-tool disagreement rates—confirm that tool complementarity is structural rather than incidental and that programs relying on any single engine implicitly accept its idiosyncratic blind spots as the boundary of their detection capability. Artificial intelligence augmentation meaningfully extends coverage into semantically demanding criteria but introduces anchoring risks that demand blind-first review protocols, disconfirming-evidence prompts, and independent verification requirements before automated conclusions are accepted. A continuous auditing pipeline with graded confidence tiers provides the operational architecture that reconciles these demands—allocating expert attention proportionally to both potential user impact and evidential uncertainty, versioning rule sets and model prompts with the same discipline applied to application code, and treating the auditing system itself as a measurable, governed artifact whose quality must be tracked over time rather than assumed at deployment.

## References

[1] Tolu Adedoja, "Automated Evaluation of Detectable Accessibility Issues on U.S. State Government Homepages: A Baseline Assessment Ahead of the 2026–2027 ADA Title II Deadlines," Research Square, 2026. [Online]. Available: https://www.researchsquare.com/article/rs-8663556/v1

[2] Mahan Tafreshipour et al., "Ma11y: A Mutation Framework for Web Accessibility Testing," ACM Digital Library, 2024. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3650212.3652113

[3] Ben Caldwell et al., "Web Content Accessibility Guidelines (WCAG) 2.0," W3C Recommendation, World Wide Web Consortium, Dec. 2008. [Online]. Available: https://www.w3.org/TR/WCAG20/

[4] Fernando Alonso, "Requirements for a Method of Software Accessibility Conformity Assessment," Proc. Int. Conf. Computers for Handicapped Persons (ICCHP), Linz, Austria, 2008. [Online]. Available: https://oa.upm.es/2425/1/INVE_MEM_2008_55928.pdf

[5] André Pimenta Freire et al. “Accessibility Inspections Using the Web Content Accessibility Guidelines by Novice Evaluators: An Experience Report,” ACM Digital Library, 2024. https://doi.org/10.1145/3702038.3702040

[6] Karol Król and Wojciech Sroka. “Internet in the Middle of Nowhere: Performance of Geoportals in Rural Areas According to.” MDPI, 2023. https://doi.org/10.3390/ijgi12120484

[7] Juho Vepsäläinen et al., “Overview of Web Application Performance Optimization Techniques,” arXiv, 2024. https://arxiv.org/pdf/2412.07892

[8] Jonathan Robert Pool, “Accessibility Metatesting: Comparing Nine Testing Tools.” W4A '23: Proceedings of the 20th International Web for All Conference, 2023. https://doi.org/10.1145/3587281.3587282

[9] Thomas Fischer et al., “Coverage of web accessibility guidelines provided by automated checking tools.” Universal Access in the Information Society, 2025. https://link.springer.com/content/pdf/10.1007/s10209-025-01263-x.pdf

[10] Heidilyn V. Gamido, Marlon V. Gamido. "Comparative review of the features of automated software testing tools.” International Journal of Electrical and Computer Engineering, 2019. https://www.researchgate.net/profile/Heidilyn-Gamido/publication/335928031

[11] Juan-Miguel López-Gil and Juanan Pereira, "Turning manual web accessibility success criteria into automatic: an LLM-based approach," Universal Access in the Information Society, vol. 24, pp. 837–852, March 2025. https://doi.org/10.1007/s10209-024-01108-z

[12] ZIYAO HE et al. (2025). “Enhancing web accessibility: Automated detection of issues with generative AI,” Proceedings of the ACM on Software Engineering. https://doi.org/10.1145/3729371