International Journal of Engineering Research and Development
e- ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 22, Issue 9 (September 2026), PP 64-68

Accessibility-Aware Adaptive AI Interfaces for Intelligent Web Applications: A Human-in-the-Loop Framework

Venkata Krishna Teja Kolli
Independent Researcher, United States
Corresponding Author: Venkata Krishna Teja Kolli

ABSTRACT
Intelligent web applications increasingly personalize content, navigation and task support through artificial intelligence. However, many adaptive interfaces are optimized for engagement or efficiency rather than accessibility, which can create new barriers for users with visual, motor, cognitive or situational limitations. This paper proposes an Accessibility-Aware Adaptive AI Interface framework for intelligent web applications. The framework combines a user accessibility profile, semantic user interface analysis, WCAG-aligned rule checks, AI-generated adaptations and human feedback into a controlled adaptation loop. The goal is to support individualized experiences while preserving task intent, explainability and standards compliance. The paper contributes a reference architecture, an operational workflow, a risk-control mapping and an evaluation protocol for measuring usability, accessibility conformance, adaptation stability and user trust. The proposed framework treats accessibility not as a late-stage audit but as a runtime property of intelligent web applications.

Date of Submission: 02-09-2026 Date of acceptance: 12-09-2026

I. INTRODUCTION

Web accessibility is a core requirement for inclusive digital systems. Accessibility standards such as WCAG and WAI-ARIA provide guidance for perceivable, operable, understandable and robust web content [1], [2]. At the same time, modern web applications increasingly use artificial intelligence to personalize interfaces, summarize content, automate workflows and assist users during complex tasks. This creates an opportunity to improve accessibility, but it also creates a risk that adaptive behavior may change an interface in ways that are difficult to predict or audit.

Traditional accessibility practice often focuses on design-time and test-time conformance. Developers audit color contrast, keyboard navigation, labels, focus order and semantic markup before release. Intelligent web applications, however, may alter content ordering, generate explanations, suggest next actions or hide irrelevant controls at runtime. These adaptations can help some users while harming others. For example, simplifying a dashboard may help a user with cognitive load but may confuse a screen-reader user if landmarks, headings or control labels become inconsistent.

This paper asks: how can intelligent web applications adapt to user needs while preserving accessibility, transparency and user control? The proposed answer is an Accessibility-Aware Adaptive AI Interface framework. The framework treats accessibility as a runtime property governed by explicit rules, semantic analysis and human feedback. It is intended for web applications where AI modifies or mediates the user experience, including portals, dashboards, learning systems, support tools and agent-assisted workflows.

The paper contributes a reference architecture, a runtime workflow and an evaluation protocol. It is intentionally framed as a design-science paper rather than an experimental performance report. The goal is to provide a practical structure that researchers and developers can implement, test and extend without relying on unsupported claims about benchmark accuracy.

II. RELATED WORK

WCAG 2.2 organizes accessibility requirements around principles of perceivability, operability, understandability and robustness [1]. WAI-ARIA defines semantics that help assistive technologies interpret dynamic web interfaces [2]. These standards are essential for modern web development, but they were not designed specifically for generative AI systems that can modify content and interaction flows at runtime. Intelligent adaptation therefore requires a bridge between standards-based accessibility and AI-driven personalization.

Human-centered design also provides useful foundations. ISO 9241-210 emphasizes understanding users, tasks and environments throughout design and evaluation [4]. Accessibility research similarly shows that

process and policy matter because accessibility failures often emerge from organizational workflow, not only individual coding errors [14]. For intelligent web applications, this means that accessibility cannot be delegated entirely to a model; it must be embedded in the design, evaluation and governance process.

Recent work on language models and web agents demonstrates that models can interpret HTML, reason about page structure and perform multi-step web tasks [8], [9], [10]. These capabilities are relevant to accessibility because models can explain page content, infer missing labels, summarize complex screens and assist with navigation. However, web-agent research also shows that real interfaces remain difficult because pages are long, dynamic and domain-specific. Accessibility-aware systems must therefore combine model flexibility with deterministic checks and user confirmation.

Generative AI risk guidance is also relevant. OWASP identifies risks such as prompt injection, insecure output handling, excessive agency and overreliance in LLM applications [7]. NIST AI RMF and the Generative AI Profile emphasize governance, measurement and human oversight across the AI lifecycle [5], [6]. In accessibility-aware adaptation, these risks appear when a model changes the interface, gives incorrect guidance or hides essential controls. The proposed framework responds by separating analysis, policy, adaptation and user review.

III. PROPOSED FRAMEWORK

The proposed framework is built around an adaptation loop. The loop begins with a user accessibility profile, continues through semantic user interface analysis and WCAG-aligned checks, produces AI-supported adaptation proposals and ends with human review and feedback. The framework does not assume that a model should directly rewrite the interface. Instead, AI-generated adaptations are treated as proposals constrained by accessibility policy and reviewed by the user or application owner when appropriate.

A. Architectural Components

The first component is the user profile layer. It stores accessibility preferences such as text size, contrast preference, reduced motion, keyboard-only use, screen-reader use, reading support needs and preferred explanation style. The second component is the semantic UI analyzer. It observes the page structure, document object model, headings, landmarks, labels, focus order, dynamic regions and current task state. The third component is the accessibility rule layer, which checks relevant WCAG and WAI-ARIA rules before and after adaptation.

The fourth component is the AI adaptation engine. It can summarize long content, reorder optional panels, generate plain-language explanations, suggest keyboard paths, propose labels for unlabeled controls, or identify likely barriers. The fifth component is the adaptive interface policy. This policy determines which adaptations are allowed automatically, which require user confirmation and which are blocked. The final component is the human review and feedback layer, which lets users accept, reject or report adaptations.

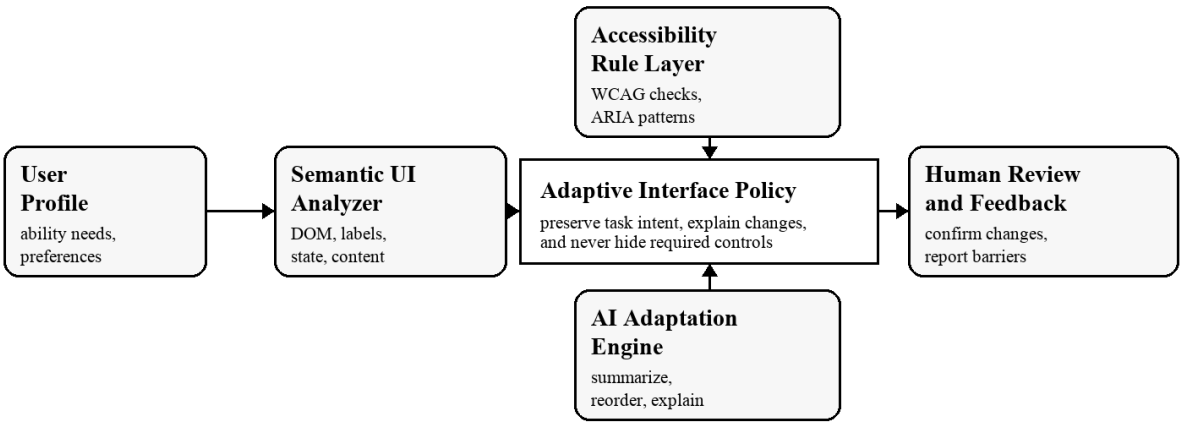

Figure 1: Accessibility-aware adaptation loop for intelligent web applications

B. Adaptation Principles

The framework follows four principles. First, preserve task intent: adaptation should support the user's goal rather than changing it. Second, preserve semantics: generated changes must maintain labels, roles, headings, relationships and focus behavior. Third, explain adaptation: the user should know when the interface has been simplified, summarized or reorganized. Fourth, maintain reversibility: the user should be able to return to the original interface or disable a class of adaptations.

Table 1: Accessibility risks and framework controls

Risk	Example in Intelligent Web Application	Framework Control
Semantic loss	AI-generated simplification removes headings, labels or landmark structure.	Run semantic checks before publishing or displaying adapted views.
Navigation instability	Controls move unpredictably after each generated adaptation.	Use stable regions, focus preservation and user-approved layout changes.
Over-personalization	The system hides information it assumes the user does not need.	Never hide required controls; provide reversible adaptation states.
Incorrect assistance	The assistant gives an invalid keyboard path or misleading description.	Validate against live DOM state and allow user feedback.
Prompt injection	External content influences accessibility guidance or generated instructions.	Treat page content as data; separate trusted accessibility policy from untrusted text.

IV. OPERATIONAL WORKFLOW

The workflow begins when a user activates adaptive assistance or when the application detects a relevant accessibility preference. The semantic UI analyzer captures the current state of the page and produces a structured representation containing headings, regions, controls, labels, focus order, error messages and dynamic content. This representation is smaller and safer than sending the full page to a model because it emphasizes accessibility-relevant structure.

The accessibility rule layer then performs deterministic checks. Examples include missing labels, ambiguous link text, insufficient heading structure, keyboard traps and dynamic regions without appropriate announcement behavior. These checks establish a baseline. The AI adaptation engine can then propose improvements such as a plain-language summary, a task-specific navigation path, a reordered non-critical panel group or a suggested label for developer review.

Before adaptation reaches the user, the adaptive interface policy applies constraints. A low-risk adaptation, such as increasing spacing or generating a summary alongside the original text, may appear immediately. A medium-risk adaptation, such as reordering optional sections, may require confirmation. A high-risk adaptation, such as hiding fields or changing form submission behavior, should be blocked unless reviewed by a developer or explicitly requested by the user. The workflow ends by capturing feedback so the system can improve future recommendations.

V. EVALUATION PROTOCOL

Evaluation should measure both accessibility conformance and user experience. Automated conformance checks are necessary but insufficient because many accessibility problems depend on task context, assistive technology behavior and user preference. The proposed protocol combines standards checks, task completion, adaptation stability, user trust and qualitative barrier reporting.

A test suite should include representative tasks such as finding information, completing a form, comparing options, resolving an error and navigating a dashboard. Each task should be tested in three modes: original interface, AI-assisted interface without adaptation and accessibility-aware adaptive interface. Reviewers should include users with different access needs where possible, and results should record whether adaptations improved or disrupted task completion.

Table 2: Evaluation dimensions for accessibility-aware adaptation

Dimension	Metric	Evaluation Question
Conformance	WCAG checks before and after adaptation.	Did adaptation preserve required accessibility properties?
Task success	Completion rate, errors and assistance requests.	Did users complete the intended workflow more effectively?
Stability	Unexpected layout, focus or label changes across sessions.	Does adaptation remain predictable and reversible?
Comprehension	User understanding of generated summaries or guidance.	Did AI assistance make the interface easier to understand?
Trust	User confidence and willingness to continue using adaptation.	Does the user feel in control of the changed interface?

VI. ILLUSTRATIVE APPLICATION SCENARIOS

The first scenario is an intelligent learning platform. A student asks for help understanding a dense assignment page. The system generates a plain-language explanation and a step-by-step task outline while preserving the original content and headings. The student can switch between original and adapted views. This supports cognitive accessibility without hiding source material.

The second scenario is an enterprise dashboard. A keyboard-only user needs to complete a multi-step approval task. The semantic UI analyzer identifies the current focus path and the assistant explains the next

keyboard actions. If the dashboard contains an inaccessible control, the system reports it and offers an alternate accessible route when one exists. The adaptation does not submit or approve anything without explicit user action.

The third scenario is a public service form. A user encounters an error message after submitting a field. The assistant maps the error to the related input, explains the required correction and ensures that the message is available through the same semantic relationship used by assistive technologies. This scenario shows how AI guidance can reinforce, rather than replace, accessible markup.

The fourth scenario is a commerce site with many filters and recommendations. The system can reduce visual clutter by grouping optional recommendations, but it cannot hide checkout, price, shipping or accessibility-critical controls. The user can review what was hidden or summarized. This prevents over-personalization from becoming a barrier.

VII. DISCUSSION

The proposed framework positions accessibility as an active part of intelligent web application behavior. In static applications, accessibility can often be reviewed against a known interface. In adaptive applications, the interface can change based on model output, user state or task context. The framework therefore adds a policy and validation layer between AI generation and the final user experience.

There are trade-offs. Runtime accessibility analysis adds engineering complexity and may affect performance. User profiles must be handled carefully because accessibility preferences can reveal sensitive information. AI-generated explanations may be helpful but can also be wrong or overly confident. For these reasons, deterministic checks, transparency and human feedback are necessary. The framework is not a substitute for accessible design; it is a way to keep accessibility intact when intelligent behavior is added.

The framework also clarifies that intelligent web applications require separate governance for data, actions and interface changes. Accessibility-aware adaptation governs the user interface itself: what is reorganized, summarized, explained or preserved during runtime personalization. This perspective helps teams avoid a common failure mode in which AI features are added to an application without a corresponding policy for how the user experience may change.

The main limitation of this paper is that it presents a reference framework rather than a user study. Future work should implement the framework in a real application and evaluate it with users who rely on screen readers, keyboard navigation, magnification, captions or cognitive support. Another useful direction is to develop machine-readable adaptation policies that can be shared across design systems and audited during continuous integration.

VIII. CONCLUSION

AI can make web applications more responsive to individual needs, but unmanaged adaptation can also introduce accessibility barriers. This paper proposed an Accessibility-Aware Adaptive AI Interface framework that combines user profiles, semantic UI analysis, standards-aligned checks, AI-generated adaptation proposals and human feedback. The framework helps developers build intelligent web applications that adapt without losing accessibility, predictability or user control.

The central conclusion is that accessibility should be treated as a runtime design constraint for intelligent web applications. Adaptive interfaces must preserve semantics, explain changes and remain reversible. By evaluating conformance, task success, stability, comprehension and trust, developers can move beyond one-time accessibility audits toward intelligent web systems that are inclusive by design and accountable in operation.

REFERENCES

[1] World Wide Web Consortium, Web Content Accessibility Guidelines (WCAG) 2.2, W3C Recommendation, 2023.
[2] World Wide Web Consortium, Accessible Rich Internet Applications (WAI-ARIA) 1.2, W3C Recommendation, 2023.
[3] World Wide Web Consortium, Authoring Tool Accessibility Guidelines (ATAG) 2.0, W3C Recommendation, 2015.
[4] International Organization for Standardization, ISO 9241-210: Human-Centred Design for Interactive Systems, 2019.
[5] National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, 2023.
[6] National Institute of Standards and Technology, Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile, NIST AI 600-1, 2024.
[7] OWASP Foundation, OWASP Top 10 for Large Language Model Applications, OWASP GenAI Security Project, 2025.
[8] I. Gur et al., Understanding HTML with Large Language Models, arXiv:2210.03945, 2022.
[9] X. Deng et al., Mind2Web: Towards a Generalist Agent for the Web, arXiv:2306.06070, 2023.
[10] S. Zhou et al., WebArena: A Realistic Web Environment for Building Autonomous Agents, arXiv:2307.13854, 2023.
[11] N. F. Liu et al., Lost in the Middle: How Language Models Use Long Contexts, Transactions of the Association for Computational Linguistics, 2024.
[12] Y. Gao et al., Retrieval-Augmented Generation for Large Language Models: A Survey, arXiv:2312.10997, 2023.

[13] K. Greshake et al., Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection, arXiv:2302.12173, 2023.
[14] J. Lazar, D. F. Goldstein and A. Taylor, Ensuring Digital Accessibility through Process and Policy, Morgan Kaufmann, 2015.
[15] S. Abou-Zahra, Web Accessibility Evaluation Tools: Overview and Selection, World Wide Web Consortium Web Accessibility Initiative, 2024.