# WebAccessBench: Digital Accessibility Reliability in LLM-Generated Web Interfaces

Sunday 22nd February, 2026

Casey Kreer
casey@conesible.de

***Abstract*—Today I am introducing WebAccessBench, a novel benchmark for AI language models to assess accessibility quality and WCAG conformance in generated web interfaces under realistic prompting conditions. In this whitepaper, I report the current benchmark snapshot, describe the evaluation framework, and document the gap between older/smaller model families and newer flagship systems. I provide aggregate quantitative findings that are useful for model selection, risk triage, and policy design. The main empirical result is a two-regime pattern: older/smaller models show lower raw error counts but much higher element-normalized error density, while newer flagship models show higher absolute error totals with substantially lower error density per DOM structure. Guidance improves outcomes across both cohorts, but the overall data shows that guiding a model with expert-grade prompts has very little effect over a small nudge. The benchmark results suggest that objective error count is too high to rely on LLM technology at all in digital accessibility work, even under explicit expert guidance.**

## I. Introduction

I built WebAccessBench to answer a common question that software developers face on a widespread basis: If a language model is used to generate front-end code, how often does it introduce accessibility defects, and what level of prompt guidance is needed to lower that risk? I also aim to evaluate whether current-generation models are suitable for the task at all.

Current model evaluations like SWE Bench and "Humanity's Last Exam" often collapse into a single percentage-based score. For accessibility, that is inadequate. Absolute error counts, normalized error rates, and zero-defect frequency can disagree, especially when models generate outputs with very different lengths and levels of verbosity. This paper therefore emphasizes multi-metric interpretation. For convenience and comparability against other benchmarks, I include such a score regardless.

I also focus on a concrete comparison: older/smaller model behavior versus newer flagship behavior, with explicit attention to current Chinese flagship open-weight models GLM-5 and Kimi K2.5, as well as American proprietary flagships Anthropic Claude Opus 4.6 and OpenAI GPT-5.2, all released within the last three months.

## II. Evaluation Framing and Methodology

WebAccessBench evaluates model outputs under three guidance regimes:

1) **Unguided**: baseline generation behavior, explaining only what specifically to generate.
2) **Little guidance**: same prompts with a light accessibility nudge, namely "Make it accessible".
3) **Expert guidance**: same prompts with notes explicitly centered on accessibility intent, designed to highlight common implementation pitfalls.

I assess generated interfaces with axe-core, an industry-standard automated accessibility testing library, and each reported error or defect is uniquely mapped to one likely WCAG success criteria and a concrete DOM element. To preserve benchmark integrity for future models and to mitigate overfitting risk, I intentionally do not publish the exact harness composition, prompt text, or filtering internals in this paper. However, most of them prompts are trivial to recreate, and I will publish an example prompt-set on conesible.de/wab shortly.

At a high level, models are given standalone UI implementation tasks like "Develop a login form" or "Make a toggle button", and are expected to produce complete front-end artifacts directly rather than filling in bounded components inside an existing web framework. This means that models are not operating inside common framework guardrails (for example, no pre-existing design system, routing layer, or framework-enforced component contracts). They must handle structure, interaction logic, semantics, and application states in one output. This intentionally broad implementation scope decreases task complexity because no full pre-existing codebase needs to be handled, and this likely better shows true model capability as more training and documentation material exists for vanilla technologies and they are usable with much less complex APIs. This also means that real-world performance is expected to be worse than WebAccessBench reports.

All models are tested on the same 150 base tasks, with prompt framing adjusted by guidance level. Each task is evaluated twice in a fresh, uninitialized conversation. For scoring, I retain the better of the two attempts (the one with lower measured error burden). Concretely, this yields $150 \times 3 \times 2 = 900$ total generations per model, while $150 \times 3 = 450$ scored generations are used for final reporting. This best-of-two procedure provides a more stable estimate of model capability and reduces single-run variance, but it is also a cherry-picked protocol and therefore likely underestimates

real-world failure rates for one-shot usage. I intentionally settled on this approach rather than taking an average because a higher sample size would be more beneficial for this, but is not feasible cost-wise at the moment, with some single task runs costing over USD $5.

For interpretability, I report:

- average accessibility error count,
- zero-error share,
- structure-normalized error burden (errors per 100 DOM elements),
- aggregate overall benchmark score.

With these values, I can evaluate LLM output quality in a methodologically defensible way. Each metric captures a different failure dimension, allowing the analysis to separate models that appear similar under aggregate scoring but differ meaningfully in practical risk profiles. This framework also makes cross-model and cross-guidance comparisons more robust by exposing ranking reversals between absolute error burden and structure-normalized defect density. As a result, the interpretation is not only more precise, but also more actionable for deployment decisions.

## III. Scoring

To summarize model performance in a single number, WebAccessBench maps accessibility results to a 0–100 score. The score uses three inputs: average error count ($E$), average DOM elements ($D$), and zero-error share ($Z$, in percent).

First, element-normalized burden is computed as:

$$R_{\text{dom}} = \begin{cases} \frac{E}{D/100}, & D > 0 \\ E, & \text{otherwise.} \end{cases}$$

Then error burden is converted to quality using an exponential decay with half-life 2.0:

$$Q_{\text{err}} = 100 \cdot 2^{-E/2}, \qquad Q_{\text{dom}} = 100 \cdot 2^{-R_{\text{dom}}/2}.$$

All values are clamped to $[0, 100]$.

Per guidance condition, the reported score is:

$$S_{\text{guidance}} = 0.50 \cdot Q_{\text{dom}} + 0.50 \cdot Z.$$

The model-level overall score (across all guidance conditions) is:

$$S_{\text{overall}} = 0.50 \cdot Q_{\text{err}} + 0.30 \cdot Q_{\text{dom}} + 0.20 \cdot Z.$$

Scores are rounded to the nearest integer in $[0, 100]$.

## IV. Current Snapshot

The snapshot analyzed here is the current benchmark artifact generated on February 20, 2026 (UTC), covering 19 models.

At a global level:

- overall score range: 11 to 62,
- global mean overall score: 26.84,
- top overall model in this snapshot: `openai/gpt-5-nano` (62).

Cross-model guidance effects are directionally strong:

- mean average errors: 3.22 (Unguided), 2.01 (Little guidance), 2.17 (Expert guidance),
- mean zero-error share: 7.97%, 11.47%, and 11.16%, respectively,
- mean errors per 100 DOM elements: 12.39, 6.91, and 6.86, respectively.

All models improve from Unguided to Little guidance on average error count, and 18/19 improve from Unguided to Expert guidance.

## V. Small/Old vs New Flagship Models

### A. Cohort Definition

For this paper, I compare two explicit LLM cohorts:

- **Small/old cohort** (4): `openai/gpt-3.5-turbo`, `anthropic/claude-3-haiku`, `google/gemma-3-12b-it`, `openai/gpt-oss-20b`.
- **New flagship cohort** (4): `z-ai/glm-5`, `moonshotai/kimi-k2.5`, `anthropic/claude-opus-4.6`, `openai/gpt-5.2`.

Models with reasoning capability were allowed to reason on their default setting as provided by OpenRouter. Flagship models were selected by reported benchmark results on industry recognized benchmarks such as SWE-bench Verified. `openai/gpt-3.5-turbo` and `anthropic/claude-3-haiku` are included in the old cohort to allow for easier comparison between newer and older models. All models of the flagship cohort have been released in the last three months.

### B. Cohort-Level Contrast

It is important to note that the flagship cohort consistently produces more visually stylized outputs, while the small/old cohort generally produces more functional interfaces with limited visual styling. This also increases the amount of issues the automated detector is able to find in a given sample. However, multiple studies suggest that users with disabilities generally prefer simpler interfaces, akin to those generated by the old/small cohort. Therefore, in the final scoring, the total error count per task is weighed more aggressively.

The flagship cohort, trained specifically on agentic software tasks, performs substantially worse on raw average error count, but far better on element-normalized error density. Specifically, the small/old cohort has a higher mean overall score (31.0 vs 17.5) and lower raw average errors (Unguided 2.17 vs 5.13; Little guidance 1.65 vs 3.19; Expert guidance 1.82 vs 3.15). However, the flagship cohort produces larger outputs in DOM scope, and correspondingly lower normalized error burden (Unguided 9.69 vs 14.15 errors per 100 DOM elements; Little guidance 5.73 vs 9.38; Expert guidance 4.89 vs 9.61).

This pattern indicates two different failure regimes:

1) **Compact-output regime (small/old):** lower absolute error totals, but higher error concentration per DOM structure.

2) **Large-structure regime (flagship):** higher absolute totals, but lower error concentration once normalized by DOM scope.

In practical terms, developers that optimize only for raw defect counts may prefer compact models; developers that optimize for defect density per DOM structure may prefer newer flagships.

### C. Focused Flagship Analysis

A common pattern in the requested flagship set is large Unguided → Guided improvement, but with different preferred guidance styles. GPT-5.2 improves from 3.48 average errors (Unguided) to 1.98 (Little guidance), a 43.1% reduction. Claude Opus 4.6 improves from 5.17 to 2.02 under Little guidance, a 60.9% reduction. GLM-5 improves from 4.75 to 3.79 with Expert guidance as best, a 20.2% reduction. Kimi K2.5 improves from 7.11 to 3.66 with Expert guidance as best, a 48.5% reduction.

*a) GPT-5.2.:* GPT-5.2 shows a strong response to lightweight guidance ($3.48 \rightarrow 1.98$ average errors), with weaker performance under expert-style prompts (2.40). Its element-normalized error rates are moderate relative to this benchmark (4.32–7.56 errors per 100 DOM elements), indicating that guidance improves density as well as absolute burden.

*b) Claude Opus 4.6.:* Claude Opus 4.6 has the largest relative gain in this subset under Little guidance (60.9% reduction from Unguided). It also shows a strong element-normalized improvement ($11.81 \rightarrow 4.17$ errors per 100 DOM elements under Little guidance; 3.51 under Expert guidance), but still accumulates non-trivial absolute defects.

*c) GLM-5.:* GLM-5 is atypical in this set: it benefits more from Expert guidance than Little guidance (best 3.79 vs 4.23 avg errors). This suggests that GLM-5 may require stronger intent specification to realize accessibility improvements. Even with this gain, its absolute errors remain high compared with top-performing models in the full board.

*d) Kimi K2.5.:* Kimi K2.5 shows the highest Unguided burden in this subset (7.11 avg errors), then a substantial drop under guidance, with Expert guidance best (3.66). The improvement is large but still leaves a high residual burden versus the field median.

### D. Counterpoint from Older/Smaller Models

The small/old cohort produces smaller DOM footprints and, in this snapshot, frequently higher zero-error rates. For example, `openai/gpt-3.5-turbo` posts strong raw accessibility outcomes (Unguided 1.40, Little 0.86, Expert 0.91 avg errors; zero-error share up to 64.67%). However, because its DOM scope is small, normalized error density remains relatively high compared to the flagship cohort.

This is exactly why I treat absolute and normalized metrics as complementary rather than substitutable.

## VI. Comparison to human-written code

Every year, WebAIM of Utah State University conducts an automated analysis of one million home pages using WAVE, tooling that is methodologically similar to WebAccessBench and features approximate results. Across 2019–2025, WebAIM reports approximately 50–60 detected accessibility errors per page; in 2025 specifically, the reported averages are 51 errors and 1257 DOM elements per page, corresponding to $\approx 4.06$ errors per 100 DOM elements (range using 50–60: $\approx 3.98$–$4.77$).

Against this reference, WebAccessBench shows the expected absolute-count gap but a different normalized picture. Pooled across all benchmarked models, generated artifacts are much smaller and contain fewer absolute errors per sample, yet remain relatively defect-dense when normalized by structure. Cohort-level analysis sharpens this result. The small/old cohort has lower absolute error burden (overall mean 1.88 errors/sample) but very small DOM scope (17.02 elements/sample), yielding $\approx 11.05$ errors per 100 elements. The flagship cohort has higher absolute burden (overall mean 3.82 errors/sample) but substantially larger outputs (57.34 elements/sample), yielding $\approx 6.77$ errors per 100 elements. Thus, the cohort ordering reverses under element normalization: small/old models look better on raw counts, while flagships look better on structural error density.

Guidance effects are also cohort-specific under this normalization. In the small/old cohort, errors per 100 elements decrease from 14.15 (Unguided) to 9.38 (Little guidance) and 9.61 (Expert guidance). In the flagship cohort, they decrease from 9.69 (Unguided) to 5.73 (Little guidance) and 4.89 (Expert guidance). Even the strongest flagship condition remains above the WebAIM reference range, indicating that lower absolute error counts in generated artifacts are largely explained by reduced interface scope rather than superior accessibility quality per unit of DOM structure.

This remains a non-isomorphic comparison: WebAIM measures real deployed pages with heterogeneous legacy complexity, whereas WebAccessBench measures bounded generated artifacts under controlled prompts. Nevertheless, element-normalized results support the same operational conclusion: current AI-generated web code still requires expert human accessibility review before deployment.

## VII. Interpretation and Practical Guidance

I take six operational conclusions from this snapshot:

1) **Prompt guidance is non-optional.** Across all models, unguided prompting shows materially worse results.
2) **Guidance type should be model-specific.** Lightweight guidance is consistently beneficial, while expert-style guidance is beneficial only for specific models (for example, GLM-5 and Kimi K2.5).
3) **Absolute vs normalized metrics can reverse rankings.** If this is ignored, model selection can be systematically wrong for a given workload profile.
4) **Flagship scale does not guarantee lower raw accessibility defects.** In this assessment, several flagships trail older/smaller LLMs on raw error counts and zero-error frequency.

5) **Flagships can still be attractive for large interfaces.** Their lower element-normalized error density suggests higher local quality per DOM structure.
6) **Current model quality is still insufficient for autonomous accessibility-critical delivery.** Even under a cherry-picked best-of-two protocol, error rates remain high enough to require human accessibility review.

Overall, this means that current-generation LLMs are not yet suitable for unsupervised development of accessible web applications. As AI-assisted development is adopted at scale, accessibility defects can spread quickly and may feed back into future training corpora. This creates material societal risk for people with disabilities by increasing the probability of exclusion from key domains including education, healthcare, finance, leisure, and work.

Model vendors must improve web UI output quality substantially, not only visually but also semantically and behaviorally. Policymakers should enforce existing digital accessibility legislation and develop additional accountability mechanisms for both model vendors and software developers.

## VIII. Benchmark Limitations

WebAccessBench remains an automated high-throughput system, so I keep the following caveats explicit:

- Automated assessment cannot fully substitute for expert manual accessibility review. The tooling used here captures only a subset of standards defined in WCAG 2.2 and EN 301 549.
- Prompt buckets are behaviorally useful but do not isolate all causal factors.
- Model and provider behavior are time-varying; results are timestamped measurements, not permanent rankings.

## IX. Conclusion

WebAccessBench provides a structured measurement framework for accessibility reliability in LLM-generated web interfaces, using absolute error burden, zero-error frequency, element-normalized burden, and an aggregate score that combines these signals. In this snapshot (19 models; 150 tasks across three guidance conditions; best-of-two scoring), prompt guidance consistently improves outcomes relative to unguided generation (mean errors: 3.22 Unguided, 2.01 Little guidance, 2.16 Expert guidance), but improvement is heterogeneous by model and guidance style. This confirms that accessibility performance is not a monotonic function of model scale and that model-specific prompt policy remains necessary.

Cross-model comparisons also show that metric choice changes interpretation. Some models perform better on absolute defect counts, while others perform better on normalized burden due to broader DOM scope. The scoring framework therefore treats these measures as complementary rather than interchangeable. At the same time, external comparison to WebAIM indicates that absolute counts alone can be misleading: although benchmark artifacts have fewer total errors per output, they also contain far fewer DOM elements. Under element-normalized comparison, the benchmarked AI outputs remain more defect-dense than the large-scale human-written baseline.

The practical implication is urgent and political, not merely technical. If AI-generated interfaces are deployed at scale without rigorous accessibility safeguards, exclusion will be industrialized: people with disabilities will face compounded barriers to education, healthcare, employment, public services, finance, and democratic participation. Accessibility defects in generated code do not stay in a prototyping stage; they replicate across products, organizations, and future training data, turning preventable design failures into systemic discrimination. This is why accessibility review cannot be optional, and why responsibility must extend beyond individual developers to model vendors, platform operators, regulators, and public institutions.

## X. Addendum

Due to time pressure, this whitepaper and the campaign website with the full benchmark results have been written with the assistance of an LLM. The actual research has been handcrafted.