technologies

MDPI

Article

Evaluating Generative AI for HTML Development

Ahmad Salah Alahmad [1,*] and Hasan Kahtan [2]

[1] Accounting and MIS Department, Gulf University for Science and Technology, Mishref 32093, Kuwait
[2] Cardiff School of Technologies, Cardiff Metropolitan University, Cardiff CF5 2YB, UK; hkahtan@cardiffmet.ac.uk
* Correspondence: alahmad.s@gust.edu.kw

Abstract

The adoption of generative Artificial Intelligence (AI) tools in web development implementation tasks is increasing exponentially. This paper evaluates the performance of five leading Generative AI models: ChatGPT-4.0, DeepSeek-V3, Gemini-1.5, Copilot (March 2025 release), and Claude-3, in building HTML components. This study presents a structured evaluation of AI-generated HTML code produced by leading Generative AI models. We have designed a set of prompts for popular tasks to generate five standardized HTML components: a contact form, a navigation menu, a blog post layout, a product listing page, and a dashboard interface. The responses were evaluated across five dimensions: semantic structure, accessibility, efficiency, readability, and search engine optimization (SEO). Results show that while AI-generated HTML can achieve high validation scores, deficiencies remain in semantic structuring and accessibility, with measurable differences between models. The results show variation in the quality and structure of the generated HTML. These results provide practical insights into the limitations and strengths of the current use of AI tools in HTML development.

Keywords: generative AI; HTML development; AI code generation; frontend web components

check for updates

Academic Editors: George F. Fragulis and Debopriyo Roy

Received: 17 July 2025
Revised: 7 September 2025
Accepted: 9 September 2025
Published: 1 October 2025

Citation: Alahmad, A.S.; Kahtan, H. Evaluating Generative AI for HTML Development. *Technologies* **2025**, *13*, 445. https://doi.org/10.3390/technologies13100445

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Generative AI models have recently gained attention in software development [1], especially in areas such as HTML generation [2]. These models, built on deep learning and transfer learning techniques, can detect structural and stylistic patterns in training data and use them to generate new content [3,4]. AI uses deep and machine learning methodologies to identify patterns and relationships within existing data to facilitate the creation of new content that mimics the style, tone, or structure of the training data [5,6].

The ability of these models to generate HTML documents using straightforward textual prompts is driving interest in both academia and industry [7]. AI continues to advance in enabling users with limited coding skills to build web components, automating repetitive coding tasks, and accelerating development cycles [8–11].

The emergence of Generative AI technologies has transformed the software development industry. Recent industry reports suggest that the use of AI in code production is expanding rapidly, with some surveys estimating that up to 44% of production code may involve AI assistance [12]. For example, Microsoft has reported that approximately 30% of its codebase is written with AI support [13]. While such figures illustrate the growing role of AI in development, they should be interpreted cautiously, as methodologies for calculating these percentages are not always transparent or standardized. Nonetheless, survey data

consistently indicate widespread adoption, with over 70% of programmers reporting that they use AI assistance in their workflows [14], and business leaders increasingly expecting AI to contribute to content generation [15]. Among the most influential tools driving this change are ChatGPT, DeepSeek, Gemini, Copilot, and Claude, which have become industry leaders according to numerous studies that document their capabilities [16,17].

While industry adoption of AI-assisted code generation is rapidly increasing, this adoption also raises significant concerns that remain insufficiently addressed in current practice. AI-generated code can represent biases embedded in training data, produce insecure constructs, and obscure accountability when errors occur in deployed systems [5,10,18]. Feuerriegel, Hartmann, Janiesch and Zschech [5] emphasize the ethical and operational risks of deploying AI code without rigorous oversight, while Goldstein, Sastry, Musser, DiResta, Gentzel and Sedova [10] discuss the vulnerability of automatically generated scripts to adversarial exploitation. Consequently, systematic evaluation frameworks are necessary not only to measure functional quality but also to mitigate the inherent risks of AI-generated outputs.

The quality of AI-generated HTML code carries significant consequences for web development outcomes, directly affecting project reliability, maintainability, performance, and security [19,20]. Accordingly, selecting the most appropriate Generative AI tools is crucial for professional development. We have addressed this essential need through a systematic evaluation of five AI platforms to assess their performance across some of the most commonly used web components, i.e., contact forms, navigation menus, blog post layouts, product listings, and dashboard interfaces.

The evaluation focuses on six key aspects of HTML quality that directly impact real-world development workflows. First, we validate the HTML markup against web standards to identify syntax errors and compliance issues. Second, we analyze semantic accuracy by examining the proper use of structural elements and scoring with Lighthouse audits. Third, we assess accessibility compliance against web guidelines. Fourth, we measure code efficiency through quantitative metrics such as line counts, Document Object Model (DOM) depth, and redundancy patterns. Fifth, we evaluate readability based on indentation consistency and commenting practices. Finally, we examine SEO evaluation through meta tag implementation and heading hierarchy structure.

Our results demonstrate significant differences in how these AI models handle HTML generation tasks. While some produce clean, standards-compliant code with few errors, others require substantial corrections. The findings provide developers with practical insights into each model's strengths and weaknesses, helping them make informed decisions about integrating AI tools into their workflows. This study also identifies common patterns in AI-generated HTML that need improvement, particularly in accessibility implementation and semantic structure. These observations contribute to ongoing discussions about the role of AI in web development and highlight areas where human oversight remains essential.

By establishing this comprehensive evaluation framework, we aim to advance understanding of current AI capabilities in HTML generation while providing a foundation for future research. The methodology and results offer valuable benchmarks for both practitioners using these tools and researchers working to improve them. As Generative AI continues to evolve, studies like this will help ensure its effective application in professional web development contexts.

AI has recently been applied to web construction, enabling automated generation of HTML code through advanced language models. Although this capability accelerates development, the absence of systematic evaluation methods for AI-generated HTML introduces potential risks related to structural integrity, accessibility compliance, and search engine optimization (SEO). A review of existing studies reveals that current literature

primarily focuses on AI coding accuracy without addressing how such code performs against established standards of usability and discoverability.

Despite this rapid adoption, concerns remain underexplored in the literature, including potential biases in training data, insecure constructs, and accessibility shortcomings in AI-generated code. Existing studies primarily examine accuracy and security in general-purpose programming languages, with limited focus on HTML-specific quality factors, such as semantic integrity, accessibility, and SEO performance. This gap underscores the need for a systematic evaluation framework that benchmarks AI-generated HTML against established web standards.

This research addresses this gap by presenting a structured evaluation framework for AI-generated HTML, assessing its quality across validation, semantic consistency, accessibility, and SEO metrics. The findings aim to provide both practitioners and researchers with evidence-based guidance for ensuring that AI-assisted web construction produces reliable, accessible, and optimized outputs. To our knowledge, this is the first study to propose standardized metrics for evaluating AI-generated HTML, enabling reproducible comparisons across tools.

The rest of this article is organized as follows: Section 2 reviews related work on Generative AI in web development. Section 3 introduces the conceptual framework for evaluating AI-generated HTML. Section 4 details the methodology, including case studies and evaluation metrics. Section 5 presents the results, and Section 6 discusses their implications. Finally, Sections 7 and 8 outline limitations, conclusions, and future work.

2. Related Work

Current studies show that Generative AI can automate coding and content creation, but it comes with some challenges, such as explainability, bias, and consistency [1,21].

Recent changes and advancements in the domain of Generative AI have led to the development of tools that are changing the web development spectrum [21,22]. Solutions like ChatGPT, DeepSeek, Gemini, Copilot, and Claude have enabled users to generate functional code snippets and complete components using natural language prompts [1,21–24]. These tools used large language models trained on huge code repositories, which enabled them to produce HTML, CSS, and JavaScript [24–26]. Using AI assistants in development tasks can significantly reduce development time for common web components [1,25,27].

Recent studies have also emphasized security vulnerabilities in AI-generated code. For example, Augustine, Md, Sultan, Sharif, Sokolova, Katunina, Pysarenko, Klimova, Kovalchuk and Parikesit [23] demonstrate how large language models can produce HTML and embedded scripts prone to SQL injection, cross-site scripting, and other exploitable flaws if prompts or training sets contain insecure patterns. Furthermore, training data bias can also manifest in semantic and accessibility errors, such as non-standard use of tags, missing alternative text for images, or improper heading hierarchies [28]. These issues degrade both usability and compliance with web accessibility standards (WCAG), highlighting the urgent need for systematic evaluation frameworks that integrate security, semantic integrity, and accessibility checks when assessing AI-generated HTML.

Several studies have examined the quality and reliability of AI-generated code, with Lizcano, Martínez-Ortíz, López and Grignard [19] emphasizing maintainability and performance, and Li et al. [20] extending the focus to include security vulnerabilities. While these works advance empirical understanding, they remain primarily descriptive and tool-specific, offering limited integration with broader theories of software quality or usability. From a Human–Computer Interaction (HCI) perspective, prior work has not adequately addressed how accessibility and semantic integrity affect real-world user experience, nor how SEO impacts discoverability—both of which are central to web usability frameworks.

Similarly, software quality models, such as ISO/IEC 25010 [29], stress attributes like maintainability, reliability, and usability, yet these are rarely applied to AI-generated HTML. This lack of theoretical grounding underscores the need for a systematic evaluation model that aligns HTML-specific dimensions (semantic structure, accessibility, SEO) with established quality and usability principles. Our study bridges this gap by extending prior empirical findings into a structured framework that is both theoretically informed and practically applicable.

Recent studies such as Lizcano, Martínez-Ortíz, López and Grignard [19] and Devadas, Hiremani, Gujjar and Sapna [22] have evaluated web development, focusing on maintainability, runtime reliability, and security vulnerabilities in general-purpose programming languages. However, these efforts have not extended to markup languages like HTML, where concerns such as semantic structure and SEO optimization are central to usability and visibility, particularly in web-based applications. This study contributes by filling that gap through a focused evaluation framework specific to HTML generation.

The role of W3C validation has emerged as particularly crucial in assessing AI-generated HTML. Compliance with web standards has a direct impact on accessibility and search engine optimization, as validation errors often correlate with Web Content Accessibility Guidelines (WCAG) violations and poor SEO performance [3]. Recent work has demonstrated that even minor markup errors can significantly affect screen reader compatibility and search engine crawling efficiency [4]. This underscores the need for rigorous validation as part of any AI-assisted development workflow, particularly when generating structural components that form the foundation of accessible web experiences.

While prior studies have extensively examined AI explainability and bias (e.g., [5,10]), their implications for AI-generated HTML remain insufficiently explored. Explainability limitations hinder the ability of developers to trace how large language models select specific tags or attributes, making debugging difficult when accessibility standards are violated. Similarly, biases in training data can lead to the overuse of deprecated or non-compliant elements, producing web pages that fail validation checks or reduce search engine visibility. These gaps underscore the necessity of frameworks that assess not only the syntactic correctness of AI-generated code but also its adherence to semantic, accessibility, and SEO best practices. This paper directly addresses this deficiency by integrating these factors into a systematic evaluation model for AI-generated HTML.

Current research continues to explore methods for improving AI output quality, with promising approaches including prompt engineering techniques and hybrid human–AI development workflows. However, as our study demonstrates, significant variations persist between tools in their ability to produce standards-compliant, accessible markup without human intervention. This body of work collectively informs our methodological approach and evaluation framework.

3. Conceptual Framework

This study adopts and extends evaluation frameworks proposed by Lizcano, Martínez-Ortíz, López and Grignard [19], Vinčević and Zajmović [30]. These studies used a set of web quality dimensions, such as maintainability, standard compliance, and end-user usability, that need to be considered when evaluating web components. Building on these foundations, we propose a multi-dimensional evaluation rubric specifically tailored to assess HTML code generated by Generative AI models. Figure 1 shows the proposed conceptual framework, highlighting the six key evaluation dimensions used to assess AI-generated HTML code.

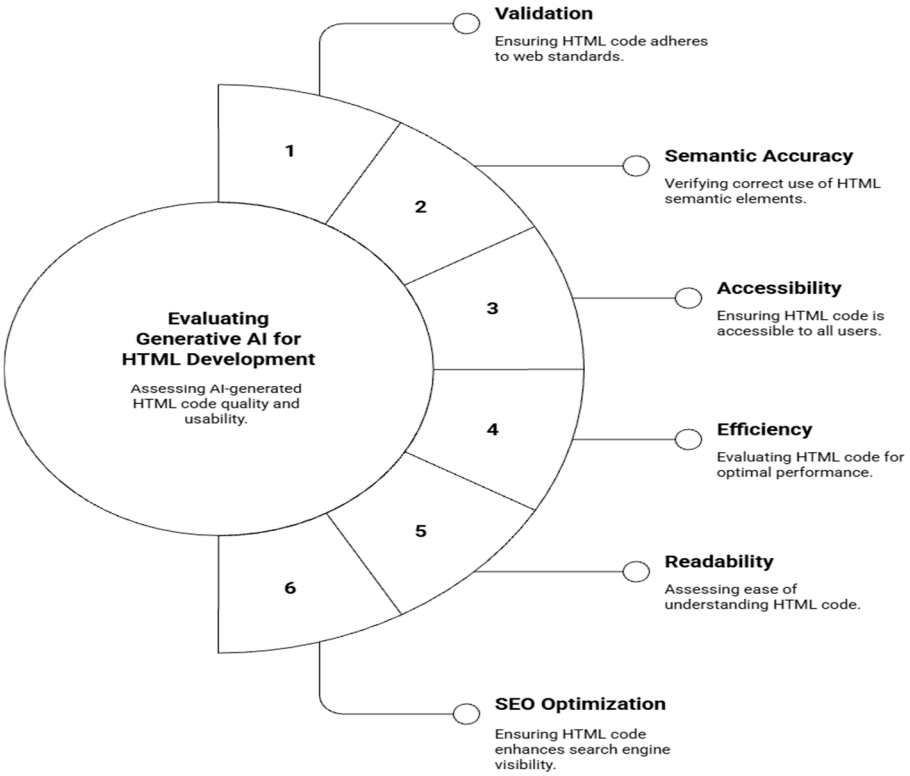


Figure 1. Evaluation framework for AI-generated HTML.

Our framework focuses on six key dimensions: validation, semantic accuracy, accessibility, efficiency, readability, and SEO optimization. These criteria are designed to capture both technical correctness and practical usability, reflecting contemporary best practices in frontend development and aligning with established web quality models.

These dimensions were selected to address common limitations in AI-generated outputs, such as a lack of semantic structure, insufficient accessibility support, and inefficient or redundant markup issues that directly impact production readiness and user experience [4,31].

4. Methodology

The study evaluated five state-of-the-art Generative AI models using the latest stable version. The selected models are ChatGPT, DeepSeek, Gemini, Copilot, and Claude. These platforms were selected based on their widespread adoption in development workflows and demonstrated capabilities in code generation tasks. Models were accessed through their web interface in April 2025, using default configurations (temperature = 0.5–0.7, max_tokens = 2048–4096, no system prompt modifications) to replicate typical developers' experiences. The evaluation is solely on HTML generation capabilities without CSS or JavaScript augmentation.

The assessment employed a systematic framework comprising automated validation tools. For each AI model, we generated five standardized web components: (1) a contact form with name, email, and message fields; (2) a responsive navigation menu with dropdown functionality; (3) a blog post layout with semantic markup; (4) a product listing page with grid layout; and (5) a dashboard interface with multiple card components. Identical prompts were administered across all platforms to ensure comparability, specifying requirements for semantic structure, accessibility features, and responsive design.

Automated validation was performed through a structured pipeline to ensure consistency and reproducibility. First, the HTML output from each model was saved in its

original form without manual modification. These files were then processed using the W3C Markup Validation Service API to identify syntax and structural errors. Validation results were parsed programmatically to count the number and type of errors, and logged for subsequent analysis. This automated process eliminated manual scoring bias and ensured uniform evaluation across all models.

All generated HTML outputs underwent rigorous validation through the W3C Markup Validation Service, which identified syntax errors, deprecated elements, and structural issues. We recorded both error counts and specific error types (e.g., unclosed tags, improper nesting, obsolete attributes) for quantitative comparison. Semantic accuracy was assessed using Chrome DevTools' Lighthouse, which evaluates proper use of HTML5 semantic elements and heading hierarchies. Accessibility compliance was measured using Lighthouse automated testing and compatibility checks.

Two experienced web developers conducted independent inspections; both used the selected tools in evaluating the generated code against industry best practices and flagging discrepancies. Inter-rater reliability was maintained through calibration sessions and a shared evaluation rubric. For example, indentation quality was assessed line-by-line using a text editor, while the presence of comments and adherence to label associations were verified manually. This hybrid approach, combining automated validation with expert review, ensured comprehensive assessment across both technical compliance and practical usability dimensions.

The experimental setup involved the utilization of multiple AI models specifically designed to generate HTML code from predefined prompts, such as creating a simple form or a blog layout. These prompts were carefully selected to represent common frontend development scenarios, enabling a comprehensive evaluation of the generated code's quality.

The evaluation process was structured around several key stages. First, the HTML pages generated by the AI models were validated using the W3C Markup Validation Service to detect syntactical errors and ensure compliance with established web standards. Following this, semantic accuracy was assessed through Google Lighthouse within Chrome DevTools, which provides insights into the semantic correctness of the markup and its alignment with best practices.

Accessibility was evaluated based on compliance with WCAG 2.1 standards [32]. This combination ensured a thorough examination of how well the generated HTML supports users with disabilities.

Additional metrics were incorporated to enrich the analysis. These included quantifying the number of validation errors and warnings, as well as categorizing the most common types of errors encountered, such as missing tags or incorrect nesting. This detailed breakdown offered insights into specific weaknesses of the AI-generated code and highlighted areas requiring improvement for production readiness.

Our methodology consists of two main phases: HTML generation and multi-dimensional evaluation, as follows:

1. HTML Generation

We prompted Generative AI models to produce HTML code for predefined tasks, such as creating a simple form or a basic blog layout. These prompts were designed to reflect common use cases in frontend development, allowing for standardized evaluation across multiple outputs.

2. Evaluation Process

The generated HTML code was assessed using a comprehensive rubric covering six core dimensions: validation, semantic accuracy, accessibility, efficiency, readability, and SEO optimization. This process consists of sub-processes to evaluate as follows:

(1) HTML Validation

The generated code was tested using the W3C Markup Validation Service to identify errors and warnings, ensuring compliance with established HTML standards.

(2) Semantic Accuracy Check

We employed Google Lighthouse (via Chrome DevTools) to assess the semantic structure and proper usage of HTML5 elements.

(3) Accessibility Compliance Check

Accessibility was evaluated according to WCAG 2.1 guidelines, using automated tools.

(4) Efficiency Evaluation

Efficiency was analyzed across three sub-metrics:

(i) Lines of Code (LoC): Measuring code length to assess verbosity.

Note: LoC counts were taken after ignoring blank lines and pure closing tags such as . HTML was evaluated in the same format as generated by each model, without additional minification, ensuring that all counts were based on consistent output conditions.

(ii) Redundancy Check: Identifying repeated or unnecessary markup.
(iii) DOM Depth: Evaluating the nesting structure to detect overly complex hierarchies.

(5) Readability Check

Code readability was examined through the following:

(i) Indentation Quality: Consistency and clarity of visual structure.
(ii) Commenting: Measured by the number of code comments present in the HTML output, if any.

(6) SEO Evaluation

SEO performance was assessed based on standard on-page optimization factors, such as the presence of meta tags, use of heading hierarchy, image alt attributes, and semantic elements that contribute to search engine indexing and discoverability.

The methodology is summarized in Figure 2.

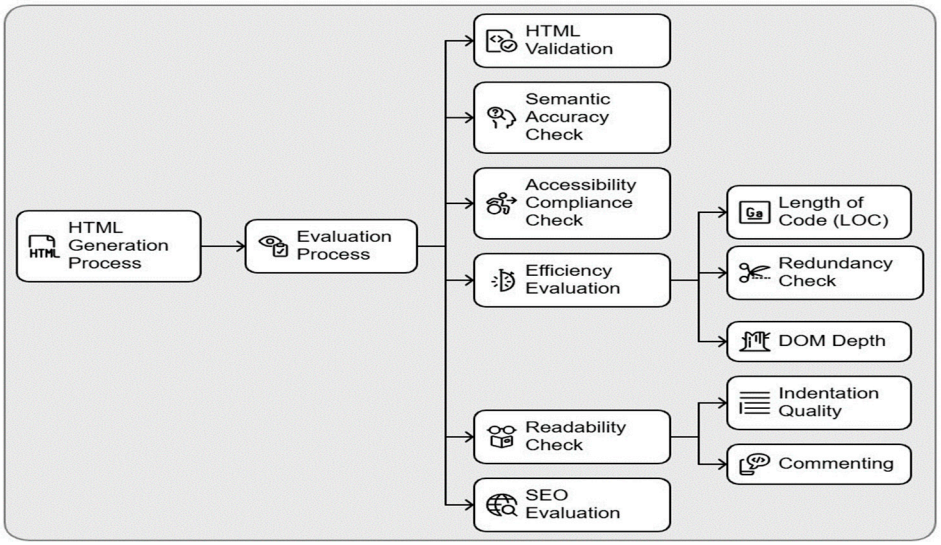


Figure 2. Research Methodology.

To enhance the transparency and replicability of our evaluation framework, we explicitly map each quality metric to the tools and procedures used for its assessment. This mapping clarifies how dimensions such as validation, semantic accuracy, accessibility, efficiency, readability, and SEO optimization were evaluated in practice within the study. The associated tools, including W3C Validator and Lighthouse, are linked to specific measurement criteria for each metric. A detailed overview of this mapping is provided in Table 1.

Table 1. Mapping Between the Metrics and The Evaluation Tools.

Metric	Tool(s)	Details/How It's Measured
HTML Validation	W3C Markup Validation Service (https://validator.w3.org accessed on 15 April 2025) [33]	Number of errors and warnings. Types of common issues (missing tags, incorrect nesting, obsolete elements).
Semantic Accuracy	Lighthouse (Chrome DevTools) [34,35]	Checks semantic errors and warnings. Best Practices score
Accessibility Compliance	Lighthouse (Chrome DevTools) [34,35]	Accessibility score.
Efficiency	Lines of Code (LoC) [36–38]	Count the total number of lines. Count the total number of tags used.
	Redundancy Check [39]	Count the number of repeated code blocks (e.g., repeated styles or elements that could be simplified).
	DOM Depth [40,41]	Measure the maximum depth of the DOM tree (easy via browser DevTools). Shallow DOM = more efficient.
Readability	Indentation Quality [42]	Check if proper consistent indentation is used (spaces/tabs per level).
	Commenting [43–45]	Count the number of code comments (if any). Comments in generated HTML are rare, but if they exist = plus.
SEO Evaluation	Lighthouse (Chrome DevTools) [46,47]	Presence of meta tags. Correct heading structure (h1, h2, etc.). Link text meaningfulness.

Each metric was selected to capture both technical correctness and user-centric quality, ensuring a well-rounded evaluation of AI-generated HTML code. For instance, Lighthouse audits offer quantifiable insights into best practices, accessibility, and SEO areas often overlooked in traditional code evaluation. Manual checks, such as indentation consistency and the presence of comments, complement automated tools by accounting for developer readability and maintainability. This combined approach ensures that the evaluation reflects real-world development concerns, bridging the gap between automated generation and production-level readiness.

Lighthouse evaluations were conducted using Chrome DevTools (Lighthouse version 100.0.0.4) under default settings. This configuration applies the standard audit categories—performance, accessibility, best practices, and SEO, without custom performance budgets or modified accessibility thresholds. Using the default configuration ensures that results are comparable to other studies and reflect widely accepted baseline measurements.

Furthermore, to minimize subjectivity in evaluating efficiency and readability, we adopted measurable metrics such as code length, DOM tree depth, semantic tag usage ratio, and indentation consistency.

A. Proposed Redundancy Metrics:

1. Style Redundancy Score: Count duplicate CSS properties (e.g., repeated padding, margin, color).

$$\text{CSS Redundancy (\%)} = (\text{Number of duplicate properties})/(\text{Total unique properties}) \times 100$$

$$\text{CSS Score} = 100 - \text{CSS Redundancy (\%)}$$

2. HTML Structure Redundancy: Count unnecessary wrapper elements.

$$\text{HTML Redundancy (\%)} = (\text{Redundant wrappers})/(\text{Total structural tags}) \times 100$$

$$\text{HTML Score} = 100 - \text{HTML Redundancy (\%)}$$

$$\text{Redundancy Score: (CSS Score + HTML Score)}/2$$

B. Proposed length of code after ignoring blank lines and pure closing tags like :

$$\text{LoC (\%)} = (\text{LoC} - \text{Lowest LoC})/(\text{Highest LoC} - \text{Lowest LoC}) \times 100$$

$$\text{LoC Score} = 100 - \text{LoC (\%)}$$

C. Proposed DOM Depth score:

$$\text{DOM Depth (\%)} = ((\text{Highest DOM Depth} - \text{Lowest DOM Depth})/(\text{DOM Depth} - \text{Lowest DOM Depth})) \times 100$$

$$\text{DOM Depth Score} = 100 - \text{DOM Depth (\%)}$$

D. Proposed Indentation Quality score:

$$\text{Indentation Error (\%)} = ((\text{Highest Errors} - \text{Lowest Errors})/(\text{Indentation Errors} - \text{Lowest Errors})) \times 100$$

$$\text{Indentation Quality Score} = 100 - \text{Indentation Error (\%)}$$

E. Proposed Commenting score:

$$\text{Commenting (\%)} = ((\text{Highest Comments} - \text{Lowest Comments})/(\text{Comments} - \text{Lowest Comments})) \times 100$$

$$\text{Commenting Quality Score} = 100 - \text{Commenting (\%)}$$

5. Results

The comprehensive evaluation of five leading Generative AI models—ChatGPT, DeepSeek, Gemini, Copilot, and Claude—yielded significant insights into their capabilities for HTML generation across fundamental web components. Our methodology employed a rigorous, multi-dimensional assessment framework examining validation, semantic, accessibility, efficiency, readability, and SEO optimization. This systematic approach allowed for direct comparison between models while controlling for prompt consistency and evaluation criteria, providing reliable data on their relative strengths and weaknesses in real-world development scenarios.

The analysis focused on five representative web components (contact form, navigation menu, blog post layout, product listing page, and dashboard layout) that collectively encompass the core structural requirements of modern web development [48–50]. The tasks were designed to reflect varying complexity and common web development requirements. These components were selected to test the models' ability to handle diverse HTML challenges from basic form elements to complex, nested layouts while adhering to web standards and best practices. The following results reveal how each AI platform performed against these critical benchmarks, highlighting both the current state of AI-assisted HTML generation and areas needing improvement.

HTML case studies were created to generate a list of standard web components via AI, as shown in Table 2 below. The prompt used was as follows:

- Generate a valid, well-structured HTML webpage for a [specific use case]. The HTML must follow best practices and include proper semantic elements. The page should be structured as follows: [The structure from Table 2].

Table 2. Case Studies Used for Evaluation.

Case Study	The Structure
A simple contact form	Fields for name, email, message, and a submit button Link to home page Logo Map for the address General details about the website
A navigation menu	Logo Horizontal navigation bar with links (e.g., Home, About, Services, Contact) Dropdown menu under one link (e.g., Services) Responsive design
A blog post layout	Blog title Author name and post date Featured image Blog content with headings, paragraphs, and images Comment section at the bottom
A product listing page	Grid of products Each product with image, name, price, and "Add to Cart" button Filter/sort options (e.g., by category or price) Pagination controls at the bottom
A dashboard layout	Sidebar navigation (e.g., Dashboard, Profile, Settings) Top navigation bar with user profile picture and logout button Main area with cards showing KPIs (e.g., Sales, Users, Revenue) Embedded charts or graphs Notifications panel

Our evaluation of five Generative AI models revealed distinct performance patterns in HTML generation across key quality metrics. The results demonstrated in Table 3 show substantial variation between tools in their ability to produce valid, semantic, and accessible markup for fundamental web components, with notable strengths and weaknesses emerging across validation compliance, structural accuracy, and implementation efficiency. These findings provide empirical evidence of how current AI technologies handle core web development tasks while highlighting persistent challenges that require attention.

Table 3. Evaluation Results.

Case	Category	ChatGPT	DeepSeek	Gemini	Copilot	Claude
1	HTML Validation	1—Warning: Section lacks heading.	No errors or warnings.	1—Error: Bad value 100% for attribute width on element iframe: Expected a digit but saw % instead.	No errors or warnings.	No errors or warnings.

Table 3. *Cont.*

Case	Category		ChatGPT	DeepSeek	Gemini	Copilot	Claude
1	Semantic	Checks semantic errors and warnings.	1 <frame> or <iframe> elements do not have a title.	1 <frame> or <iframe> elements do not have a title.	1 <frame> or <iframe> elements do not have a title.	1 <frame> or <iframe> elements do not have a title.	No errors or warnings.
		Best Practices score	74	74	74	74	96
	Accessibility		92	96	92	92	96
	Efficiency	Code Length	100	73.42	65.82	83.54	0
		Redundancy Check	97.22	87.12	84	94.38	75.67
		DOM Depth	100	50	50	100	0
	Readability	Indentation Quality	40	80	0	60	100
		Commenting	0	0	0	0	0
	SEO Optimization		91	100	91	91	91
2	HTML Validation		1—Warning: Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.	No errors or warnings.	No errors or warnings.	No errors or warnings.	No errors or warnings.
	Semantic	Checks semantic errors and warnings.	No errors or warnings.	No errors or warnings.	No errors or warnings.	No errors or warnings.	No errors or warnings.
		Best Practices score	96	78	78	96	96
	Accessibility		100	100	100	100	100
	Efficiency	Code Length	85.2	0	63	100	74.1
		Redundancy Check	88.3	92.9	86.3	92.2	82.6
		DOM Depth	50	0	50	100	50
	Readability	Indentation Quality	40	100	0	60	20
		Commenting	0	100	0	0	0
	SEO Optimization		90	90	91	90	90

Table 3. *Cont.*

Case	Category		ChatGPT	DeepSeek	Gemini	Copilot	Claude
	HTML Validation		1—Warning: Consider using the h1 element as a top-level heading only (all h1 elements are treated as top-level headings by many screen readers and other tools). 2—Warning: Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.	1—Error: End of file seen when looking for tag name. Ignoring tag. 2—Error: End of file seen, and there were open elements. 3—Error: Unclosed element form. 4—Error: Unclosed element section. 5—Error: Unclosed element main.	No errors or warnings.	No errors or warnings.	1—Warning: Consider using the h1 element as a top-level heading only (all h1 elements are treated as top-level headings by many screen readers and other tools).
3	Semantic	Checks semantic errors and warnings.	No errors or warnings.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.
		Best Practices score	96	96	78	96	96
	Accessibility		100	96	95	96	96
	Efficiency	Code Length	77.0	23.0	100	87.8	0
		Redundancy Check	90.3	84.1	95.5	93.2	83.4
		DOM Depth	66.7	33.3	100	100	0
	Readability	Indentation Quality	20	100	0	40	60
		Commenting	0	0	100	0	0
	SEO Optimization		91	100	91	91	91

Table 3. *Cont.*

Case	Category		ChatGPT	DeepSeek	Gemini	Copilot	Claude
	HTML Validation		1—Warning: Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.	No errors or warnings.	1—Warning: Section lacks heading. Consider using h2–h6 elements to add identifying headings to all sections, or else use a div element instead for any cases where no heading is needed.	1—Warning: Section lacks heading. Consider using h2–h6 elements to add identifying headings to all sections, or else use a div element instead for any cases where no heading is needed.	13
4	Semantic	Checks semantic errors and warnings.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.
		Best Practices score	96	96	96	96	96
	Accessibility		96	96	94	94	96
	Efficiency	Code Length	0	100	90.6	64.9	0
		Redundancy Check	83.7	95.6	93.5	90.1	86.4
		DOM Depth	0	100	100	50	0
	Readability	Indentation Quality	80	40	60	20	100
		Commenting	60	0	0	100	100
	SEO Optimization		91	91	91	91	91
5	HTML Validation		1—Warning: Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.	No errors or warnings.	1—Warning: Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.	No errors or warnings.	8

Table 3. *Cont.*

Case	Category		ChatGPT	DeepSeek	Gemini	Copilot	Claude
5	Semantic	Checks semantic errors and warnings.	Background and foreground colors do not have a sufficient contrast ratio.	Background and foreground colors do not have a sufficient contrast ratio.	Buttons do not have an accessible name.	Heading elements are not in a sequentially descending order	Background and foreground colors do not have a sufficient contrast ratio.
		Best Practices score	96	74	96	96	96
	Accessibility		96	92	88	98	95
	Efficiency	Code Length	84.9	0	44.6	100	0
		Redundancy Check	90.4	90.9	83.2	100	94.8
		DOM Depth	75	25	50	100	0
	Readability	Indentation Quality	80	100	0	60	90
		Commenting	0	0	0	0	0
	SEO Optimization		91	91	91	91	91

6. Discussion

This study provides a multi-dimensional evaluation of HTML code generated by five leading Generative AI models. The results demonstrated in Table 3 present differences in model performance across the six quality dimensions: validation, semantic accuracy, accessibility, efficiency, readability, and SEO optimization. These findings offer actionable insights for both researchers and practitioners seeking to adopt AI tools for frontend development tasks, and they validate the importance of comprehensive evaluation frameworks such as the one proposed in this study.

The quality metrics used provide foundational metrics for AI-generated HTML. Future implementations could extend this framework with computational efficiency measures (e.g., generation time, resource consumption) and comparisons to human-coded baselines, particularly valuable for assessing real-world viability while maintaining the quality standards.

In the validation dimension, as shown in Table 4, which presents a subset of the data from Table 3 related to W3C validation error counts, Claude achieved the most reliable results, producing valid HTML with minimal warnings or errors across all five case studies. ChatGPT and DeepSeek also maintained strong performance, with only minor issues such as missing headings or unnecessary trailing slashes. In contrast, Gemini occasionally produced invalid structures—for example, unclosed elements in the blog post layout—indicating that structural integrity remains an area for improvement in some models. These differences underline the necessity of using HTML validators during AI-assisted development to prevent subtle syntax issues that may impact rendering or accessibility.

For semantic accuracy, ChatGPT and Claude showed consistent use of HTML5 structural tags, with correctly implemented heading hierarchies and appropriate sectioning elements. However, semantic warnings were observed across several models, particularly

involving the use of <iframe> elements without title attributes and contrast ratio issues in visual elements. While such findings are flagged under semantic evaluation in this study, they are also relevant to broader design considerations, especially for user experience and accessibility. These results suggest that although generative models have absorbed semantic conventions from their training data, additional refinement is needed for more complex or interactive components.

Table 4. W3C Validation Error Counts.

Case	ChatGPT	DeepSeek	Gemini	Copilot	Claude
1	1 warning	0	1 error	0	0
2	1 warning	0	0	0	0
3	1 warning	1 error	2 errors	0	0
4	1 warning	0	1 warning	1 warning	0
5	1 warning	0	1 warning	0	0

The results in the accessibility dimension were highly promising, as shown in Table 5, which presents a subset of the data from Table 3 related to Lighthouse accessibility scores. All models demonstrated strong performance, with Claude and ChatGPT achieving near-perfect Lighthouse accessibility scores in most components. This reflects the ability of Generative AI models to incorporate accessibility best practices—such as proper use of label tags, image alt text, and structured headings—even without specific prompting. Notably, the navigation menu, contact form, and dashboard components were all generated with elements that promote inclusive access. These results support the view that Generative AI has matured in its capacity to contribute to accessible web development and that it can serve as a valuable assistant for developers aiming to meet WCAG 2.1 standards [3,30].

Table 5. Lighthouse Accessibility Scores.

Case	ChatGPT	DeepSeek	Gemini	Copilot	Claude
1	92	96	92	92	96
2	100	100	100	100	100
3	100	96	95	96	96
4	96	96	94	94	96
5	96	92	88	98	95

The efficiency analysis revealed clear trade-offs between completeness and conciseness. Claude and DeepSeek generated compact, optimized code with minimal redundancy and shallow DOM structures. ChatGPT, while semantically rich, often produced longer outputs with deeper nesting and more wrapper elements, which may affect maintainability and performance in production environments. The application of our custom metrics, such as redundancy percentage and DOM Depth scores summarized in Table 6, enabled a precise comparison that moves beyond subjective judgments. These metrics not only expose structural inefficiencies but also offer practical benchmarks for selecting the most suitable AI model for performance-sensitive projects.

In terms of readability, Claude again stood out by producing consistently indented and cleanly structured code. DeepSeek also performed well in this regard, while Gemini lacked indentation altogether in some outputs, negatively impacting code legibility. Commenting, which is essential for maintainable and collaborative code, was rare across all

models. Although not a primary requirement for machine-generated HTML, adding basic inline documentation could enhance usability for developers, especially in educational or team environments. The inclusion of indentation and commenting metrics in this study strengthens the evaluation framework and draws attention to overlooked dimensions of code quality.

Table 6. DOM Depth Comparisons.

Case	ChatGPT	DeepSeek	Gemini	Copilot	Claude
1	100	50	50	100	0
2	50	0	50	100	50
3	66.7	33.3	100	100	0
4	0	100	100	50	0
5	75	25	50	100	0

The evaluation metrics demonstrate clear tool differentiation. Claude achieved excellent W3C validation in 4/5 test cases (Table 3) with Lighthouse accessibility scores consistently $\geq$96%, indicating strong compliance with automated checks. ChatGPT's generated components showed deeper DOM structures (avg. depth = 66.7) through nested elements (observed in Case 3's blog layout HTML output), while Gemini produced the most compact code (65.82 efficiency score) but with occasional validation errors. These patterns provide developers with empirically grounded selection criteria.

Finally, all five models showed consistently strong results in SEO optimization. Essential elements such as meta tags, heading hierarchies, image alt attributes, and meaningful link text were generally present across outputs. These elements were especially well implemented in static components like blog layouts and product listing pages. The consistency of SEO features across different models suggests that such elements are strongly represented in training datasets and easily generalized by language models. These results demonstrate that AI-generated HTML can support not only technical functionality but also content discoverability—a critical aspect of modern web design.

Overall, the results confirm that while Generative AI tools have become increasingly capable in producing HTML, their performance is multi-dimensional and context-dependent. No model was universally superior, but Claude delivered the most balanced results across dimensions, followed by ChatGPT, which emphasized semantic richness, and Copilot, which favored concise and readable output. This study's structured evaluation contributes both a replicable assessment model and practical guidance for developers seeking to integrate AI into web development workflows.

Although our evaluation focused on technical dimensions of HTML quality, the findings also intersect with broader debates in Generative AI. Issues such as explainability, potential security vulnerabilities embedded in training data, and the ethical implications of human–AI co-creation remain critical. These concerns highlight that even when AI models achieve strong technical outputs, their use in professional development must be accompanied by careful oversight, transparency, and accountability.

Beyond its applied benchmarking value, the framework also aligns with established perspectives in software quality and HCI, as it emphasizes usability, accessibility, and readability alongside efficiency and semantic accuracy. By framing these measures within the broader discourse on human–AI collaboration, this study contributes not only to practice but also to ongoing debates about the role of Generative AI in software engineering.

This study also illustrates how Generative AI reshapes software engineering practices by influencing code quality, usability, and accessibility, which are central to broader software

engineering discourse. By highlighting each model's strengths and limitations, the findings help inform strategic tool selection and encourage more targeted use of AI in frontend engineering. At the same time, these results must be viewed with caution: over-reliance on AI-generated code may introduce risks related to accessibility compliance, maintainability, and security, underscoring the continued need for human oversight in production environments. Practically, Claude's consistency makes it suitable for accessibility-first projects, while ChatGPT's semantic richness suits content-heavy applications, but neither model eliminates the need for professional review and validation.

7. Limitations

This study establishes key metrics for evaluating the quality of AI-generated HTML and notes several important limitations. This study focused on static HTML markup and intentionally excluded CSS styling and JavaScript to allow for accurate evaluation of structural semantics. While this controlled design offers valuable insights into the quality of the markup, it does not address critical aspects of modern web development, such as dynamic functionality and responsive design.

The evaluation was conducted at the component level using a rigorous comparative approach, but it did not extend to full-page generation. As a result, the findings do not cover how AI models manage semantic relationships across a complete document or ensure consistency between components. In addition, all testing used model versions available in April 2025. Given the rapid pace of change in Generative AI, these findings should be considered a snapshot of that time rather than a long-term assessment.

Furthermore, although standardized prompts allowed for consistent comparison across models, they did not account for differences in how each model responds to prompts (for example, ChatGPT's responsiveness to instructional nuance compared to Claude's preference for structured output). This approach prioritized consistency in the experiment over possible prompt optimization, reflecting a trade-off between methodological rigor and practical applicability. Future research should also consider testing prompt variations to illustrate their potential effects on model outputs and involve consultation with experienced developers to design richer and more representative evaluation tasks.

The study compares AI models relative to each other but lacks expert-crafted reference implementations for absolute benchmarking against professional standards, which is an important direction for future work. These constraints, although deliberate, provide focused benchmarks and point to clear opportunities for further research on full-stack HTML generation, including component interaction, layout consistency, and dynamic features.

8. Conclusions & Future Work

This study presents a comprehensive evaluation of five leading Generative AI models, i.e., ChatGPT, DeepSeek, Gemini, Copilot, and Claude, in the context of HTML development. Our findings demonstrate that while these tools show significant promise in automating web component generation, their performance varies considerably across critical quality metrics. Claude emerged as the most consistent in producing valid HTML with minimal errors, while ChatGPT excelled in semantic structure but often at the cost of code efficiency. At the same time, notable weaknesses remain, including the frequent absence of comments, recurring contrast ratio issues, and deep DOM structures that may reduce maintainability. All models also exhibited gaps in accessibility compliance, particularly in dynamic components, underscoring the continued need for human oversight in AI-assisted development workflows. The results provide developers with actionable insights for selecting appropriate tools based on specific project requirements, balancing the trade-offs between speed, correctness, and maintainability.

For future work, several directions merit exploration. Developing specialized prompt engineering techniques could improve model outputs for accessibility and semantic correctness, while integrating real-time validation tools directly into AI coding assistants may help address structural and readability issues during generation. Expanding the evaluation framework to include CSS and JavaScript generation would provide a more complete picture of AI capabilities in frontend development. Among these, prioritizing statistical validation of results and incorporating human-coded baselines may have the greatest impact, as they would establish stronger benchmarks and improve the reliability of comparisons.

While the contribution is primarily practice-oriented, it also connects to wider academic debates on human–AI interaction and software quality, offering a foundation for more theory-driven studies in the future.

Additionally, systematic investigation of prompt engineering techniques, including model-specific optimizations and their impact on accessibility compliance, would valuably extend this work. Finally, longitudinal studies tracking how these models improve over time could offer valuable insights into the evolution of AI-assisted web development. As Generative AI continues to advance, such research will be crucial for ensuring these tools meet professional standards while unlocking their full potential to enhance developer productivity.

Author Contributions: Conceptualization, A.S.A. and H.K.; methodology, A.S.A. and H.K.; software, A.S.A.; validation, A.S.A. and H.K.; formal analysis, A.S.A. and H.K.; investigation, A.S.A.; resources, A.S.A.; data curation, A.S.A.; writing—original draft preparation, A.S.A.; writing—review and editing, H.K.; visualization, A.S.A.; supervision, H.K.; project administration, A.S.A.; funding acquisition, A.S.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The datasets supporting this study, including all AI-generated HTML samples, are publicly available at https://drive.google.com/file/d/1A0ornjIH7yCHa1ub9pwUySw3351o1plW/view?usp=sharing (accessed on 20 July 2025).

Acknowledgments: The authors gratefully acknowledge the financial support provided by the Gulf University for Science and Technology (GUST) for funding this research.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Sauvola, J.; Tarkoma, S.; Klemettinen, M.; Riekki, J.; Doermann, D. Future of software development with generative AI. *Autom. Softw. Eng.* **2024**, *31*, 26. [CrossRef]
2. Abu Doush, I.; Kassem, R. Can generative AI create accessible web code? A benchmark analysis of AI-generated HTML against accessibility standards. *Univers. Access Inf. Soc.* **2025**, 1–24. [CrossRef]
3. AlDahoul, N.; Hong, J.; Varvello, M.; Zaki, Y. Towards a World Wide Web powered by generative AI. *Sci. Rep.* **2025**, *15*, 7251. [CrossRef] [PubMed]
4. Palmer, Z.B.; Oswal, S.K. Constructing websites with generative AI tools: The accessibility of their workflows and products for users with disabilities. *J. Bus. Tech. Commun.* **2025**, *39*, 93–114. [CrossRef]
5. Feuerriegel, S.; Hartmann, J.; Janiesch, C.; Zschech, P. Generative ai. *Bus. Inf. Syst. Eng.* **2024**, *66*, 111–126. [CrossRef]
6. Fui-Hoon Nah, F.; Zheng, R.; Cai, J.; Siau, K.; Chen, L. Generative AI and ChatGPT: Applications, challenges, and AI-human collaboration. *J. Inf. Technol. Case Appl. Res.* **2023**, *25*, 277–304. [CrossRef]
7. Lim, C.V.; Zhu, Y.-P.; Omar, M.; Park, H.-W. Decoding the relationship of artificial intelligence, advertising, and generative models. *Digital* **2024**, *4*, 244–270. [CrossRef]
8. Binhammad, M.H.Y.; Othman, A.; Abuljadayel, L.; Al Mheiri, H.; Alkaabi, M.; Almarri, M. Investigating how generative AI can create personalized learning materials tailored to individual student needs. *Creat. Educ.* **2024**, *15*, 1499–1523. [CrossRef]

9. Al-Ahmad, A.; Kahtan, H.; Tahat, L.; Tahat, T. Enhancing Software Engineering with AI: Key Insights from ChatGPT. In Proceedings of the 2024 International Conference on Decision Aid Sciences and Applications (DASA), Manama, Bahrain, 11–12 December 2024; pp. 1–5.
10. Goldstein, J.A.; Sastry, G.; Musser, M.; DiResta, R.; Gentzel, M.; Sedova, K. Generative language models and automated influence operations: Emerging threats and potential mitigations. *arXiv* **2023**, arXiv:2301.04246. [CrossRef]
11. Yadav, S.S.; Kumar, Y. Generative AI in Streamlining Software Development Life Cycles Within Programming Environments. *Int. J. Sci. Innov. Eng.* **2024**, *1*, 29–36.
12. EliteBrains. AI-Generated Code Statistics 2025. Available online: https://www.elitebrains.com/blog/aI-generated-code-statistics-2025 (accessed on 25 May 2025).
13. Novet, J. Satya Nadella Says as Much as 30% of Microsoft Code Is Written by AI. Available online: https://www.cnbc.com/2025/04/29/satya-nadella-says-as-much-as-30percent-of-microsoft-code-is-written-by-ai.html (accessed on 30 April 2025).
14. Overflow, S. Stack Overflow Annual Developer Survey 2024: AI Results. Available online: https://survey.stackoverflow.co/2024/ai (accessed on 12 January 2025).
15. AIPRM. AI Statistics: Latest Trends, Usage, and Adoption Rates. Available online: https://www.aiprm.com/ai-statistics/ (accessed on 23 May 2025).
16. Kerimbayev, N.; Menlibay, Z.; Garvanova, M.; Djaparova, S.; Jotsov, V. A Comparative Analysis of Generative AI Models for Improving Learning Process in Higher Education. In Proceedings of the 2024 International Conference Automatics and Informatics (ICAI), Varna, Bulgaria, 10–12 October 2024; pp. 271–276.
17. Cho, K.; Park, Y.; Kim, J.; Kim, B.; Jeong, D. Conversational AI forensics: A case study on ChatGPT, Gemini, Copilot, and Claude. *Forensic Sci. Int. Digit. Investig.* **2025**, *52*, 301855. [CrossRef]
18. Reyero Lobo, P.; Daga, E.; Alani, H.; Fernandez, M. Semantic Web technologies and bias in artificial intelligence: A systematic literature review. *Semant. Web* **2023**, *14*, 745–770. [CrossRef]
19. Lizcano, D.; Martínez-Ortíz, A.L.; López, G.; Grignard, A. End-user modeling of quality for web components. *J. Softw. Evol. Process* **2023**, *35*, e2256. [CrossRef]
20. Li, T.; Huang, R.; Cui, C.; Towey, D.; Ma, L.; Li, Y.-F.; Xia, W. A Survey on Web Application Testing: A Decade of Evolution. *arXiv* **2024**, arXiv:2412.10476. [CrossRef]
21. Mahon, J.; Mac Namee, B.; Becker, B.A. Guidelines for the Evolving Role of Generative AI in Introductory Programming Based on Emerging Practice. In Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1, Milan, Italy, 8–10 July 2024; pp. 10–16.
22. Devadas, R.M.; Hiremani, V.; Gujjar, P.; Sapna, R. Unveiling the Potential of Generative Approaches in AI-Infused Web Development for Design, Testing, and Maintenance. In *Generative AI for Web Engineering Models*; IGI Global: Hershey, PA, USA, 2025; pp. 107–128.
23. Augustine, N.; Md, A.B.; Sultan, M.H.O.; Sharif, K.Y.; Sokolova, Y.; Katunina, O.; Pysarenko, N.; Klimova, I.; Kovalchuk, O.; Parikesit, B.A.B. Enhancing SQL injection (SQLi) mitigation by removing malicious SQL parameter values using long short-term memory (LSTM) neural networks. *J. Theor. Appl. Inf. Technol.* **2025**, *103*, 3070–3083.
24. Islam, F. AI Driven Development of Modern Web Application. Bachelor's Thesis, Tampere University of Applied Sciences, Tampere, Finland, 2024.
25. APVisor. Choosing the Right AI Tool: A Comparison of DeepSeek, Claude, ChatGPT, Gemini, Copilot, and Perplexity. Available online: https://apvisor.com/?p=117 (accessed on 30 April 2025).
26. Auger, T.; Saroyan, E. *Generative AI for Web Development: Building Web Applications Powered by OpenAI APIs and Next.js*; Springer Nature: Berlin/Heidelberg, Germany, 2024.
27. Nettur, S.B.; Karpurapu, S.; Nettur, U.; Gajja, L.S. Cypress copilot: Development of an AI assistant for boosting productivity and transforming web application testing. *IEEE Access* **2024**, *13*, 3215–3229. [CrossRef]
28. Tsuchiya, M. Performance impact caused by hidden bias of training data for recognizing textual entailment. *arXiv* **2018**, arXiv:1804.08117. [CrossRef]
29. *ISO (2011) ISO/IEC 25010*; Systems and Software Engineering—Systems and Software Quality—Requirements and Evaluation (SQuaRE)—System and Software Quality Models. International Organization for Standardization: Geneva, Switzerland, 2011. Available online: https://www.iso.org/standard/35733.html (accessed on 3 April 2025).
30. Vinčević, V.; Zajmović, M. Quality models and web application validation methods. *Ann. Fac. Eng. Hunedoara-Int. J. Eng.* **2024**, *22*, 13–21.
31. Preston, M.; Rezwana, J. The Accessibility Landscape of Co-Creative AI Systems: Analysis, Insights and Recommendations. In Proceedings of the ACM IUI Workshops 2025, Cagliari, Italy, 24–27 March 2025.
32. Web Content Accessibility Guidelines (WCAG) 2.1, World Wide Web Consortium Recommendation. Available online: https://www.w3.org/TR/WCAG21/ (accessed on 8 September 2025).

33. Gupta, V.; Singh, H. Web Content Accessibility Evaluation of Universities' Websites-A Case Study for Universities of Punjab State in India. In Proceedings of the 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 17–19 March 2021; pp. 546–550.
34. Król, K. Between Truth and Hallucinations: Evaluation of the Performance of Large Language Model-Based AI Plugins in Website Quality Analysis. *Appl. Sci.* **2025**, *15*, 2292. [CrossRef]
35. Macakoğlu, Ş.S.; Peker, S.; Medeni, İ.T. Accessibility, usability, and security evaluation of universities' prospective student web pages: A comparative study of Europe, North America, and Oceania. *Univers. Access Inf. Soc.* **2023**, *22*, 671–683. [CrossRef]
36. Margea, R. Content Management Systems for Institutional Academic Websites. The Feaa Timişoara Experience. *Ann. 'Constantin Brancusi' Univ. Targu-Jiu* **2017**, *2*, 41–51. Available online: https://www.utgjiu.ro/revista/ec/pdf/2017-02/05_Margea.pdf (accessed on 25 April 2025).
37. Kim, M.C. Learning Early-Stage Web Development at Scale: Exploring Methods to Assess Learning Through Analysis of HTML and CSS. Ph.D. Thesis, Drexel University, Philadelphia, PA, USA, 2021.
38. Nuñez-Varela, A.S.; Pérez-Gonzalez, H.G.; Martínez-Perez, F.E.; Soubervielle-Montalvo, C. Source code metrics: A systematic mapping study. *J. Syst. Softw.* **2017**, *128*, 164–197. [CrossRef]
39. Hague, M.; Lin, A.W.; Ong, C.-H.L. Detecting redundant CSS rules in HTML5 applications: A tree rewriting approach. In Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, Pittsburgh, PA, USA, 25–30 October 2015; pp. 1–19.
40. Deng, X.; Shiralkar, P.; Lockard, C.; Huang, B.; Sun, H. Dom-lm: Learning generalizable representations for html documents. *arXiv* **2022**, arXiv:2201.10608. [CrossRef]
41. Moreno-Lumbreras, D. Enhancing HTML Structure Comprehension: Real-Time 3D/XR Visualization of the DOM. In Proceedings of the 2024 IEEE Working Conference on Software Visualization (VISSOFT), Flagstaff, AZ, USA, 6–7 October 2024; pp. 127–132.
42. Buse, R.P.; Weimer, W.R. Learning a metric for code readability. *IEEE Trans. Softw. Eng.* **2009**, *36*, 546–558. [CrossRef]
43. Elnaggar, M. Measuring the Readability of Web Documents. Bachelor's Thesis, Duisburg-Essen University, Duisburg, Germany, 7 September 2023.
44. Musciano, C.; Kennedy, B. *HTML & XHTML: The Definitive Guide*; O'Reilly Media, Inc.: Santa Rosa, CA, USA, 2002.
45. Hanenberg, S.; Morzeck, J.; Werger, O.; Gries, S.; Gruhn, V. Indentation and Reading Time: A Controlled Experiment on the Differences Between Generated Indented and Non-indented JSON Objects. In Proceedings of the International Conference on Software Technologies, Rome, Italy, 10–12 July 2023; pp. 50–75.
46. Tavosi, M.; Naghshineh, N. Google SEO score and accessibility rank on the American University Libraries' websites: One comparative analysis. *Inf. Discov. Deliv.* **2023**, *51*, 241–251. [CrossRef]
47. Csontos, B.; Heckl, I. Five years of changes in the accessibility, usability, and security of Hungarian government websites. *Univers. Access Inf. Soc.* **2025**, *24*, 2757–2781. [CrossRef]
48. Brown, D.M. *Communicating Design: Developing Web Site Documentation for Design and Planning*; New Riders: Los Angeles, CA, USA, 2010.
49. Zuga-Divre, L. Creating a Modern and Responsive Customer Tracking Dashboard. 2016. Available online: https://www.theseus.fi/handle/10024/119779 (accessed on 25 April 2025).
50. Hristov, I. Web Design for Small Businesses: Enhancing User Experience. 2024. Available online: https://www.theseus.fi/handle/10024/869156 (accessed on 20 April 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.