LLM-Based Usability Inspection: A Systematic Mapping of Automation Strategies, Evidence, and Validation Practices

Ana Beatriz Farias
Federal Institute of Paraíba (IFPB)
Campina Grande, PB, Brazil
ana.farias.1@academico.ifpb.edu.br

Danyllo Albuquerque
Federal Institute of Paraíba (IFPB)
Campina Grande, PB, Brazil
danyllo.albuquerque@ifpb.edu.br

Emanuel Dantas Filho
Federal Institute of Pernambuco (IFPE)
Jaboatão dos Guararapes, PE, Brazil
emanuel.filho@jaboatao.ifpe.edu.br

Ademar Neto
Federal Rural University of the Semi-Arid
Mossóro, RN, Brazil
ademarneto14@gmail.com

Abstract

Usability inspection is an important activity for identifying interaction problems, and recent advances in Large Language Models (LLMs) have created new opportunities to support or automate this process. However, it remains unclear how LLMs have been applied to usability inspection, which evidence and inspection criteria they rely on, and how their outputs are validated. This paper presents a systematic literature mapping of LLM-based usability and UX inspection. From 401 publications retrieved from Scopus, ACM Digital Library, IEEE Xplore, and SBC OpenLib, 14 primary studies published between 2023 and 2025 were selected. The studies were analyzed with respect to LLMs, prompting strategies, input artifacts, inspection criteria, validation approaches, and reported limitations. The results show a predominance of GPT-4-family models and fully automated workflows, with screenshots and structured UI representations as the most common input artifacts. Prompt engineering, particularly role prompting and chain-of-thought, is widely adopted, while validation practices remain heterogeneous. Recurring limitations include hallucinations, false positives, limited contextual judgment, and prompt dependence. These findings indicate that LLM-based usability inspection is an emerging topic in Intelligent Software Engineering, in which human-AI collaboration currently appears more realistic than full automation.

Keywords

Usability Inspection, LLM, UX Evaluation, Human-AI Collaboration, Software Quality, Intelligent Software Engineering

1 Introduction

Usability is a critical quality attribute in interactive software systems, directly affecting effectiveness, efficiency, satisfaction, engagement, conversion, and retention [7, 14]. Among usability evaluation methods, usability inspection is widely adopted because it enables specialists to identify interaction problems based on predefined principles, heuristics, or guidelines without requiring end users [13]. Although less expensive than empirical user studies, inspection still depends heavily on human expertise and interpretation, motivating increasing interest in intelligent techniques that can support or partially automate this activity.

Recent advances in Large Language Models (LLMs) have created new opportunities for Intelligent Software Engineering by enabling intelligent analysis of software artifacts throughout the software lifecycle. In usability inspection, LLMs can analyze screenshots, structured UI representations, accessibility metadata, interface descriptions, and interaction logs to identify potential usability problems, supporting AI-assisted software quality workflows.

Despite the growing number of primary studies, there is still no secondary study specifically focused on how LLMs support usability inspection from a software engineering perspective. Existing reviews discuss broader topics such as AI for usability evaluation, automated usability problem detection, or generative AI for UX testing [9, 10, 17], but they do not characterize the models, prompting strategies, evidence artifacts, validation practices, and limitations of LLM-based usability inspection.

This paper addresses this gap through a systematic literature mapping of LLM-based usability and UX inspection. The mapping selected 14 primary studies from 401 publications retrieved from Scopus, ACM Digital Library, IEEE Xplore, and SBC OpenLib. The objective is to characterize how LLMs are currently employed to support usability inspection, identify methodological trends and validation practices, and discuss implications for future research and industrial adoption. From an Intelligent Software Engineering perspective, this positions usability inspection as an AI-assisted software quality activity in which intelligent techniques must be integrated into engineering workflows while preserving reliability, traceability, and human validation.

This paper makes three main contributions. First, it provides a structured mapping of the LLMs, prompting strategies, automation levels, interface types, and input artifacts adopted in usability inspection. Second, it synthesizes how studies operationalize and validate inspection outputs using software-facing artifacts, heuristics, benchmarks, expert judgment, and quantitative metrics. Third, it identifies recurring limitations and research gaps, highlighting why human-AI collaboration remains central to reliable and evidence-grounded usability inspection.

The remainder of this paper is organized as follows. Section 2 presents background and related work. Section 3 describes the study design. Section 4 reports the mapping results. Section 5 discusses the findings in light of prior work. Section 6 presents implications for research and industry. Section 7 discusses threats to validity. Finally, Section 8 concludes the paper.

2 Background and Related Work

Usability is commonly understood as the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use [7]. In software engineering, usability is treated as a non-functional requirement and a software quality attribute. Evaluation activities

are therefore important not only for improving user experience but also for verifying whether the system supports users in achieving their goals. In this paper, the term *LLM-based usability inspection* is used as an umbrella term for studies addressing usability inspection, UX inspection, accessibility-oriented inspection, and design critique when these activities are supported or automated by LLMs.

Usability inspection methods evaluate interfaces without necessarily involving end users. Heuristic evaluation, for example, relies on specialists who inspect an interface according to usability principles such as Nielsen's heuristics [13]. Other approaches may use accessibility guidelines, cognitive walkthrough, design principles, or domain-specific criteria. These methods provide structured ways to identify usability problems, but their results depend on evaluator expertise, interpretation, and the type of evidence available during inspection.

LLMs have recently been explored as intelligent support mechanisms for software engineering tasks. In the context of usability and UX, they have been used to analyze screenshots, Figma mockups, DOM structures, accessibility metadata, screen-reader outputs, and interaction traces. Some studies use LLMs to generate design critiques, identify accessibility violations, or simulate user behavior. Others compare LLM-generated diagnoses with specialists or traditional tools. This literature suggests that LLMs can support usability inspection, but also exposes risks such as unsupported findings, false positives, hallucinations, and weak contextual judgment.

Traditional approaches to usability inspection automation have primarily relied on rule-based systems, computer vision techniques, machine learning models for specific tasks, and accessibility checking tools [8, 9, 11]. Unlike these approaches, LLMs can reason over heterogeneous software artifacts, combining textual and visual evidence to produce contextualized usability assessments. This broader reasoning capability expands the range of inspection tasks that can be supported, but also introduces challenges related to hallucinations, reproducibility, and reliability. These characteristics motivate a closer examination of how LLMs are currently being applied to usability inspection.

Previous secondary studies provide important context. A systematic mapping on LLMs in HCI-related activities discusses usability, accessibility, and UX applications [17]. A systematic review on automated usability problem detection covers a broader range of automated methods, including but not limited to LLMs [9]. Another review examines AI and generative models for UX testing across remote evaluation and qualitative data analysis [10]. In contrast, this study focuses specifically on LLM-based automation of usability and UX inspection, emphasizing models, prompts, evidence artifacts, validation practices, and reported limitations.

3 Study Design

This study follows the systematic mapping guidelines proposed by Petersen et al. [15]. A mapping study is appropriate because LLM-based usability inspection is an emerging topic, with most primary studies published only in recent years. Rather than performing a meta-analysis, the objective is to characterize the available evidence, identify research trends, and expose gaps in the field. As illustrated in Figure 1, the mapping process comprised five stages: scope definition, protocol design, search and study selection, quality assessment and data extraction, and analysis and synthesis. In what follows, each stage is described in detail.

Stage 1: Scope Definition. The first stage established the scope of the mapping study by defining its objective and formulating the research questions (RQs). Following the guidelines of Petersen et al.

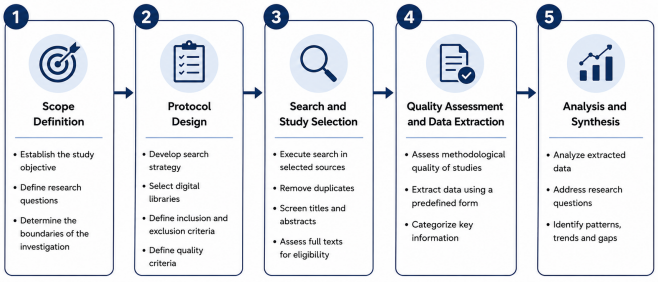

Figure 1: Systematic mapping process.

[15], the study was designed to characterize how LLMs are being applied to support usability and UX inspection from a software engineering perspective. Three RQs guided the mapping:

RQ1. Which LLMs, prompting strategies, and automation levels have been used for usability and UX inspection?
RQ2. Which inspection criteria, input artifacts, datasets, and validation practices are used to operationalize LLM-based usability inspection?
RQ3. What limitations and challenges are reported in the use of LLMs for usability and UX inspection?

RQ1 focuses on the technical characteristics of existing approaches by identifying the LLMs, prompting strategies, and levels of automation adopted across the selected studies. RQ2 examines how these approaches are operationalized and evaluated, including the inspection criteria, software artifacts, datasets, and validation procedures employed to assess their effectiveness. Finally, RQ3 synthesizes the limitations and open challenges reported in the literature, providing insights into current research gaps and future directions for LLM-based usability inspection.

Stage 2: Protocol Design. The review protocol defined the search strategy, digital libraries, selection criteria, quality assessment procedure, and data extraction form before study identification began. These materials are available in the supplementary material to support transparency and reproducibility (see the Artifact Availability section).

The search strategy was organized around two conceptual groups: (i) usability and UX inspection and (ii) LLMs or generative AI. The search string combined terms such as "usability inspection", "heuristic evaluation", "UX inspection", "large language model", "LLM", and "Generative AI". For SBC OpenLib, a Portuguese version of the search string was adopted using equivalent terms. The search was performed in four digital libraries: Scopus, ACM Digital Library, IEEE Xplore, and SBC OpenLib. The protocol also defined the inclusion and exclusion criteria, the quality assessment criteria, and the attributes to be extracted from each selected study.

Stage 3: Search and Study Selection. The automated search covered publications from 2022 to 2025 and retrieved 401 studies: 203 from Scopus, 188 from ACM Digital Library, 7 from IEEE Xplore, and 3 from SBC OpenLib. All retrieved studies were managed using Parsifal. Studies were included if they (i) explicitly investigated the use of LLMs to support or automate usability or UX inspection, (ii) were peer-reviewed publications, and (iii) were written in English or Portuguese. Studies were excluded if they (i) did not address LLM-based usability or UX inspection, (ii) focused on AI applications unrelated to usability inspection, (iii) were outside the defined

publication period, (iv) were unavailable in full text, or (v) were duplicates.

After duplicate removal, the remaining studies were screened through titles, abstracts, keywords, and, when necessary, full-text reading. Inclusion decisions were independently conducted by two researchers, with disagreements resolved by a third researcher. After applying the selection and quality assessment criteria, the final corpus comprised 14 primary studies that explicitly investigated LLM-based support or automation for usability and UX inspection.

Stage 4: Quality Assessment and Data Extraction. The selected studies were assessed using five quality criteria: (i) clarity of the research objective, (ii) explicit use of an LLM for usability or UX inspection, (iii) methodological detail, (iv) validation against specialists or reference methods, and (v) discussion of limitations or threats to validity. Each criterion received a score of 1.0 (yes), 0.5 (partially), or 0.0 (no). Only studies scoring at least 3.0 points were retained for analysis.

For each selected study, data were extracted using a predefined extraction form covering the LLM employed, prompting strategy, automation level, interface type, application domain, inspection criteria, input artifacts, datasets, validation methods, evaluation metrics, and reported limitations.

Stage 5: Analysis and Synthesis. The extracted data were synthesized to answer the three research questions. Descriptive analysis was used to characterize the adoption of LLMs, prompting strategies, automation approaches, inspection criteria, validation methods, and reported limitations. Finally, recurring patterns, methodological trends, and research gaps were identified across the selected studies to provide an overview of the current state of LLM-based usability inspection.

4 Results

This section presents the results of the mapping. The final corpus comprises 14 primary studies published between 2023 and 2025, reflecting the recent emergence of LLM-based usability inspection research. Table 1 provides an overview of the selected studies, while the complete extracted dataset and supporting materials are available in the supplementary material (see the Artifact Availability section). The following subsections discuss the findings according to the three RQs.

4.1 Models, Prompting Strategies, and Automation Levels (RQ1)

This RQ characterizes the technical landscape of LLM-based usability inspection by analyzing the models, prompting strategies, and automation levels adopted across the selected studies. The selected studies show a clear predominance of OpenAI models in LLM-based usability inspection. GPT-4 was the most frequent model, appearing in six studies, followed by GPT-4o, which appeared in three studies. Other models were used more sparsely, including GPT-4 Turbo with Vision, Gemini Pro Vision, Gemini-2.5-Flash, Gemini-1.0-Pro, Claude-3-Opus, LLaMA, and LLM-based agents, each appearing in one study. This distribution indicates that current evidence is still concentrated around a small set of commercial multimodal and instruction-following models, especially from the GPT-4 family. This concentration also limits the generalizability of current evidence, as reported capabilities and limitations may reflect characteristics of a small number of proprietary models rather than the broader landscape of LLMs available for software engineering tasks.

Prompt engineering plays a central role in operationalizing usability inspection. The reported strategies can be grouped into three categories: instruction-oriented prompting (e.g., role and structured prompting), reasoning-oriented prompting (e.g., chain-of-thought, least-to-most, and self-refinement), and multimodal prompting, which combines textual instructions with screenshots or structured UI representations. One study reports that combining role prompting with chain-of-thought improved the identification of critical UX problems [18], while other studies used structured prompts and accessibility-related artifacts to guide the inspection process [4, 20].

Most studies proposed fully automated workflows: 13 of the 14 selected studies reported full automation, whereas one adopted a semi-automated cognitive walkthrough. The investigated interfaces primarily comprise web systems, mobile applications, and design prototypes, with comparatively little evidence for wearable, desktop, or embedded interfaces. This distribution suggests that current research has focused on visually rich interfaces while leaving several application domains largely unexplored.

Although the search string was translated into Portuguese and applied to SBC OpenLib to capture Brazilian and Portuguese-language contributions, no Portuguese-language study met the inclusion and quality assessment criteria for the final corpus. Manual inspection of the SBC OpenLib results indicated that studies addressing usability or UX evaluation in Brazilian venues rarely investigate LLM-based automation specifically; when generative AI is discussed in this context, it is more frequently framed as a design or ideation aid rather than as an inspection mechanism, or the relevant contributions are published in English even when authored by Brazilian researchers. This suggests that LLM-based usability inspection remains a nascent topic in the Portuguese-language and Brazilian software engineering literature, rather than an artifact of search string design.

Answer to RQ1. LLM-based usability inspection is dominated by GPT-4-family models and fully automated approaches. Prompt engineering is a core adaptation strategy, especially role prompting, structured prompts, chain-of-thought, few-shot prompting, and visual prompting. The field is concentrated on web, mobile, and prototyping artifacts, with little evidence for desktop or embedded interfaces.

4.2 Inspection Criteria, Evidence, and Validation Practices (RQ2)

This RQ examines how LLM-based usability inspection is operationalized and evaluated, considering the inspection criteria, input artifacts, and validation practices adopted across the selected studies. The selected studies rely on a diverse set of inspection criteria. Nielsen's heuristics are the predominant reference, appearing in several studies that use LLMs to identify interface-level usability violations [2, 5, 6]. Other approaches adopt accessibility guidelines (e.g., WCAG 2.1), visual design principles (e.g., CRAP), cognitive walkthrough, Apple Human Interface Guidelines, and UX evaluation instruments such as SUS, NASA-TLX, and UEQ. This diversity indicates that current approaches are adapted to different inspection objectives rather than following a common evaluation framework.

The evidence provided to LLMs also varies considerably. Studies employ screenshots, Figma mockups, JSON representations of UI structures, screen-reader transcripts, accessibility metadata, interaction logs, and public or study-specific datasets. Examples include DOM-like JSON structures for Figma mockups [3], TalkBack transcripts and view hierarchy metadata [20], interaction logs [6], and datasets such as UICrit and JitterWeb [2, 19]. These findings suggest

Table 1: Summary of selected studies.

Study	LLM	Automation	Interface	Criteria	Validation
Guerino et al. (2025)	GPT-4o	Full	Web	Nielsen heuristics	Specialists
Zhong et al. (2025)	GPT-4o	Full	Mobile	WCAG / MagentaA11y	Tools
Shin et al. (2025)	GPT-4	Full	Multiplatform	Nielsen severity	Qualitative analysis
Lubos et al. (2025)	Gemini-2.5-Flash	Full	Multiplatform	Generic guidelines	Result review
Pourasad and Maalej (2025)	GPT-4 Turbo Vision	Full	Mobile	Prompted criteria	Specialists
Wu et al. (2024)	GPT-4V / Claude / Gemini	Full	Web	CRAP principles	Benchmark
Hsueh et al. (2024)	GPT-4	Full	Web	Nielsen heuristics	Users / specialists
Duan et al. (2024a)	Gemini Pro Vision	Full	Mobile	Nielsen / Apple HIG	Quality score
Bisante et al. (2024)	GPT-4	Semi	Web	Cognitive walkthrough	Specialists
Xiang et al. (2024)	GPT-4 / LLaMA	Full	Wearable	SUS / NASA-TLX / UEQ	Users / designers
Duan et al. (2024b)	GPT-4	Full	Design	Nielsen / semantic	Specialists
Lu et al. (2025)	LLM Agent	Full	Web	Usability guidelines	UX researchers
Duan et al. (2023)	GPT-4	Full	Design	Nielsen heuristics	Prototype evaluation
Haraguchi et al. (2024)	GPT-4o	Full	Design	Alignment/overlap/white space	Human correlation (Pearson)

an increasing reliance on heterogeneous software artifacts rather than screenshots alone.

Validation practices remain equally heterogeneous. Most studies compare LLM-generated findings against specialists, users, designers, traditional accessibility tools, or benchmark datasets. Reported evaluation measures include precision, recall, F1-score, accuracy, agreement, coverage, similarity, and Cohen's kappa. Although several studies report encouraging results—for example, precision between 0.61 and 0.66 or specialist agreement reaching 93.55% [1, 16]—the diversity of validation protocols and metrics limits direct comparison across studies.

Answer to RQ2. LLM-based usability inspection is operationalized through heterogeneous criteria and artifacts. Nielsen's heuristics remain the most common reference, but studies also use accessibility guidelines, design principles, cognitive walkthrough, and UX instruments. Evidence ranges from screenshots to structured UI data, accessibility metadata, and interaction logs. Validation is commonly based on comparison with specialists, tools, users, or benchmarks, but metrics and protocols remain inconsistent.

4.3 Limitations and Challenges (RQ3)

This RQ investigates the main limitations and challenges reported in the use of LLMs for usability and UX inspection. The selected studies consistently report challenges that affect the reliability of LLM-based usability inspection. Hallucinations and false positives are among the most frequently cited issues. For example, one study reports that 24.3% of the usability problems identified by GPT-4o were false positives [5]. Other studies show that LLMs may overapply guidelines, generate plausible but unsupported diagnoses, or overlook important contextual aspects of the interface. These findings indicate that current models still struggle to distinguish genuine usability problems from plausible but incorrect assessments.

Another recurring challenge is the limited ability of LLMs to reproduce human contextual judgment. Studies report difficulties in interpreting abstract guidelines, dynamic interactions, cultural aspects, and task-specific contexts, while simulated users or synthetic data often fail to adequately represent real user behavior [12, 20]. These limitations become particularly evident when usability issues depend on interaction flows, user goals, or domain knowledge that cannot be inferred directly from the available artifacts.

Prompt dependence also emerges as a methodological concern. Many approaches rely on carefully engineered prompts, and variations in wording, examples, reasoning instructions, or output formats may substantially influence the generated findings. As a result, both reproducibility and comparability remain limited across existing studies. It is important to note that the frequencies reported in Figure 2 correspond to the total number of mentions of each limitation category identified across the corpus, rather than the number of distinct studies reporting it. A single primary study could report a given limitation, such as hallucination, multiple times, in relation to different inspection tasks, model outputs, or findings within the same paper. As a result, category frequencies (e.g., 17 mentions of "hallucination") can exceed the number of primary studies (n=14), and the figure should be read as a mention-level frequency count rather than a study-level count.

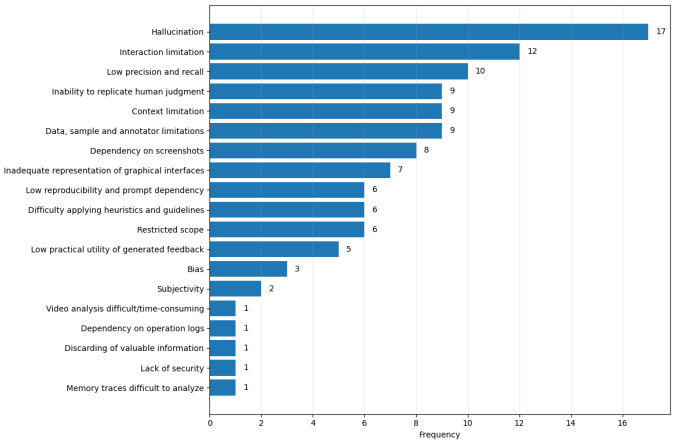

Figure 2: Frequency of reported limitation categories across the selected studies.

Answer to RQ3. Current evidence consistently highlights hallucinations, false positives, limited contextual judgment, and prompt dependence as the main barriers to reliable LLM-based usability inspection. These challenges suggest that LLMs are better suited to supporting expert evaluators than replacing them, reinforcing the importance of human-AI collaboration in usability inspection workflows.

5 Discussion

The findings should be interpreted in the context of previous work on automated usability evaluation and AI-assisted software engineering. Earlier secondary studies have broadly discussed AI for usability, accessibility, and UX evaluation [9, 10, 17]. This mapping complements this literature by specifically characterizing how LLMs are employed to support usability inspection. Collectively, the three research questions reveal a field that is evolving rapidly

but remains methodologically fragmented, with substantial variation in models, software artifacts, inspection criteria, and validation practices.

The predominance of GPT-4-family models reflects the broader adoption of multimodal and instruction-following LLMs in software engineering. However, the concentration of evidence on a small number of proprietary models limits the generalizability of current findings. Although studies have started exploring alternatives such as Gemini, Claude, LLaMA, and agent-based systems, comparative evidence across models remains scarce, making it difficult to determine whether reported performance is model-specific or representative of LLM-based usability inspection more broadly.

Another important finding concerns the role of software artifacts in inspection quality. The mapped studies employ screenshots, structured UI representations, accessibility metadata, and interaction traces as evidence for LLM-based reasoning. This suggests that inspection performance depends not only on the underlying model, but also on how software artifacts are represented and provided to the LLM. Consequently, improving artifact representation may become as important as improving the models themselves for reliable AI-assisted usability inspection.

The results also highlight an important methodological gap. Although studies employ diverse inspection criteria, datasets, and evaluation metrics, there is little consensus regarding validation protocols or benchmark datasets. This heterogeneity limits reproducibility, complicates comparisons across studies, and hinders the accumulation of evidence. Establishing standardized datasets, evaluation procedures, and reporting guidelines therefore appears to be a key research direction for advancing the field.

Finally, the findings challenge the notion of fully autonomous usability inspection. Although nearly all selected studies propose fully automated workflows, most still validate their outputs against specialists, users, or traditional inspection methods, while simultaneously reporting hallucinations, false positives, and limited contextual judgment as recurring limitations. Together, these findings indicate that current LLMs are better suited to supporting expert evaluators than replacing them. Their most promising role is as AI-assisted inspectors that generate candidate findings, support triage, and reduce manual inspection effort while preserving human oversight.

6 Main Implications

The mapping indicates that LLM-based usability inspection is evolving into an emerging research area within Intelligent Software Engineering. While the selected studies demonstrate the potential of LLMs to support usability inspection, they also reveal methodological fragmentation, heterogeneous validation practices, and limited benchmarking. The implications of these findings are discussed from both research and industrial perspectives.

Implications for Research. The results highlight the need for stronger methodological standardization. Future studies should systematically report prompt templates, model versions, input artifacts, output schemas, evaluation protocols, and the criteria used to distinguish true and false findings. Such information is essential to improve reproducibility and enable meaningful comparisons across studies.

The field would also benefit from public benchmarks that integrate representative software artifacts, expert annotations, and standardized evaluation metrics. Existing studies rely on different interface types, inspection criteria, datasets, and validation procedures, making cumulative evidence difficult. Common benchmarks would facilitate more rigorous and comparable evaluations of LLM-based usability inspection approaches.

Based on the mapped evidence, three research directions appear particularly relevant. First, future work should investigate how different software artifacts, such as screenshots, structured UI representations, accessibility metadata, and interaction logs, affect inspection reliability. Second, evaluation protocols should better distinguish visually observable usability issues from those requiring interaction, user goals, or domain knowledge. Third, future studies should assess defect detection performance, explainability, traceability, and usefulness of LLM-generated findings during human review.

Implications for Industry. The findings suggest that LLMs should be viewed as decision-support tools rather than replacements for expert evaluators. Their most realistic role is to assist specialists by identifying candidate usability issues, organizing findings, explaining potential violations, and prioritizing inspection activities, while leaving final decisions to human reviewers.

Organizations adopting LLM-based usability inspection should therefore prioritize human-AI workflows instead of fully autonomous pipelines. Effective tools should provide transparent evidence supporting each reported issue, enable reviewers to validate or reject suggestions, and maintain traceability between inspection findings and the underlying software artifacts. Such workflows have the potential to improve inspection efficiency while preserving the reliability of expert judgment.

7 Threats to Validity

As with any systematic mapping study, the results should be interpreted considering the following threats to validity.

Construct validity. The adopted search terms and selected digital libraries may not have captured all relevant studies, particularly recent preprints or publications outside the selected venues. This threat was mitigated by searching four major digital libraries using search strings that combined terms related to usability inspection, UX inspection, LLMs, and generative AI, including a Portuguese-language version of the search string applied to SBC OpenLib to capture Brazilian and Portuguese-language contributions. Nonetheless, no Portuguese-language study met the inclusion and quality assessment criteria for the final corpus. This is interpreted as reflecting the genuine scarcity of Brazilian contributions to this specific topic, rather than a limitation of the search strategy itself, since Brazilian venues rarely investigate LLM-based automation of usability inspection specifically and relevant Brazilian work, when it exists, is frequently published in English. Even so, this absence remains a threat worth monitoring as the field matures, particularly given the authors' own research context in Brazil. In addition, the complete review protocol, including the search strings, is available in the supplementary material to support transparency and reproducibility.

Internal validity. Study selection, quality assessment, and data extraction involve researcher judgment and may introduce bias. To mitigate this threat, predefined inclusion, exclusion, and quality assessment criteria were adopted. Two researchers independently screened the studies, and disagreements were resolved by a third researcher. Data extraction followed a structured protocol derived from the research questions.

External validity. The mapping is based on 14 primary studies published between 2023 and 2025. Although this reflects the emerging nature of LLM-based usability inspection, it limits the generalizability of the findings to other contexts, application domains, or future generations of LLMs. Consequently, the reported trends should be interpreted as a snapshot of the current state of the field.

Conclusion validity. The conclusions depend on the quality, completeness, and consistency of reporting in the selected primary studies. To reduce this threat, only studies satisfying predefined quality criteria were included, and the synthesis was based on evidence extracted through a structured protocol. Furthermore, the review protocol and extracted dataset are publicly available, facilitating verification, reproducibility, and future updates of the mapping.

8 Conclusion

This paper presented a systematic mapping of LLM-based usability and UX inspection from an Intelligent Software Engineering perspective. Following established systematic mapping guidelines, 401 publications retrieved from four digital libraries were screened, resulting in a final corpus of 14 primary studies published between 2023 and 2025.

The results show that current research is dominated by GPT-4-family models and predominantly by fully automated inspection workflows. Prompt engineering has become a fundamental mechanism for adapting LLMs to usability inspection, while Nielsen's heuristics remain the most frequently adopted evaluation criterion. The studies also reveal increasing use of heterogeneous software artifacts, including screenshots, structured UI representations, accessibility metadata, and interaction traces. However, validation practices remain highly heterogeneous, limiting the comparability of existing approaches.

These findings indicate that LLM-based usability inspection is rapidly emerging as an AI-assisted software quality activity rather than merely a UX evaluation technique. At the same time, recurring challenges—including hallucinations, false positives, prompt dependence, and limited contextual judgment—suggest that current LLMs are better suited to supporting expert evaluators than replacing them. Human-AI collaboration therefore appears to be the most promising direction for integrating LLMs into usability inspection workflows.

Future research should focus on establishing standardized benchmarks, evaluation protocols, and reporting guidelines to improve reproducibility and comparability across studies. Additional work is also needed to evaluate a broader range of LLMs, investigate the impact of different software artifacts on inspection quality, and assess LLM performance in dynamic interaction scenarios and complex software systems.

Artifact Availability

The supplementary material is available in an anonymized repository: https://zenodo.org/records/21892377, including the review protocol, extracted data, and supporting materials.

Generative AI Use

Generative AI tools were used to support language editing and figure preparation. All scientific content was reviewed and approved by the authors, who take full responsibility for the manuscript.

References

[1] Alba Bisante, Venkata Srikanth Varma Datla, Emanuele Panizzi, Gabriella Trasciatti, and Stefano Zeppieri. 2024. Enhancing Interface Design with AI: An Exploratory Study on a ChatGPT-4-Based Tool for Cognitive Walkthrough Inspired Evaluations. In *Proceedings of the 2024 International Conference on Advanced Visual Interfaces* (Arenzano, Genoa, Italy) *(AVI '24)*. Association for Computing Machinery, New York, NY, USA, Article 41, 5 pages. doi:10.1145/3656650.3656676

[2] Peitong Duan, Chin-Yi Cheng, Gang Li, Bjoern Hartmann, and Yang Li. 2024. UICrit: Enhancing Automated Design Evaluation with a UI Critique Dataset. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, USA) *(UIST '24)*. Association for Computing Machinery, New York, NY, USA, Article 46, 17 pages. doi:10.1145/3654777.3676381

[3] Peitong Duan, Jeremy Warner, and Bjoern Hartmann. 2023. Towards Generating UI Design Feedback with LLMs. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23 Adjunct)*. Association for Computing Machinery, New York, NY, USA, Article 70, 3 pages. doi:10.1145/3586182.3615810

[4] Peitong Duan, Jeremy Warner, Yang Li, and Bjoern Hartmann. 2024. Generating Automatic Feedback on UI Mockups with Large Language Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 6, 20 pages. doi:10.1145/3613904.3642782

[5] Guilherme Guerino, Luiz Rodrigues, Bruna Capeleti, Rafael Ferreira Mello, André Freire, and Luciana Zaina. 2025. Can GPT-4o Evaluate Usability Like Human Experts? A Comparative Study on Issue Identification in Heuristic Evaluation. arXiv:2506.16345 [cs.HC] https://arxiv.org/abs/2506.16345

[6] Nien-Lin Hsueh, Hsuen-Jen Lin, and Lien-Chi Lai. 2024. Applying Large Language Model to User Experience Testing. *Electronics* 13, 23 (2024). doi:10.3390/electronics13234633

[7] International Organization for Standardization. 2018. ISO 9241-11:2018 — Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts. Disponível em: https://www.iso.org/standard/63500.html.

[8] Melody Y. Ivory and Marti A. Hearst. 2001. The State of the Art in Automating Usability Evaluation of User Interfaces. *Comput. Surveys* 33, 4 (2001), 470–516.

[9] Eduard Kuric, Peter Demcak, Matus Krajcovic, and Jan Lang. 2025. Systematic Literature Review of Automation and Artificial Intelligence in Usability Issue Detection. *arXiv preprint arXiv:2504.01415* (2025).

[10] Jia Liu. 2025. AI-Powered Automated and Remote UX Evaluation Methods: A Systematic Literature Review. In *Proceedings of the 43rd ACM International Conference on Design of Communication (SIGDOC '25)*. Association for Computing Machinery, New York, NY, USA, 10–16. doi:10.1145/3711670.3764614

[11] Yuwen Lu, Yuewen Yang, Qinyi Zhao, Chengzhi Zhang, and Toby Jia-Jun Li. 2024. AI Assistance for UX: A Literature Review Through Human-Centered AI. *arXiv preprint arXiv:2402.06089* (2024).

[12] Yuxuan Lu, Bingsheng Yao, Hansu Gu, Jing Huang, Zheshen Jessie Wang, Yang Li, Jiri Gesi, Qi He, Toby Jia-Jun Li, and Dakuo Wang. 2025. UXAgent: An LLM Agent-Based Usability Testing Framework for Web Design. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. ACM, 1–12. doi:10.1145/3706599.3719729

[13] Jakob Nielsen and Robert L. Mack. 1994. *Usability Inspection Methods*. John Wiley & Sons, New York, NY, USA.

[14] Donald A. Norman. 2006. *O design do dia-a-dia*. Rocco, Rio de Janeiro.

[15] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. 2008. Systematic Mapping Studies in Software Engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE)*. 68–77.

[16] Ali Ebrahimi Pourasad and Walid Maalej. 2025. Does GenAI Make Usability Testing Obsolete? arXiv:2411.00634 [cs.SE] https://arxiv.org/abs/2411.00634

[17] Simone Santos, Robson Alves, Francisco Oliveira, and Francisco Araujo. 2025. Investigating LLM-based tools to support Usability, Accessibility, User Experience in HCI activities: A Systematic Literature Mapping. In *Proceedings of the 31st Brazilian Symposium on Multimedia and the Web* (Rio de Janeiro/RJ). SBC, Porto Alegre, RS, Brasil, 631–645. doi:10.5753/webmedia.2025.16160

[18] Subin Shin, Jeesun Oh, and Sangwon Lee. 2025. Can LLMs See What I See? A Study on Five Prompt Engineering Techniques for Evaluating UX on a Shopping Site. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, Article 125, 7 pages. doi:10.1145/3706599.3720079

[19] Jason Wu, Yi-Hao Peng, Amanda Li, Amanda Swearngin, Jeffrey P. Bigham, and Jeffrey Nichols. 2024. UIClip: A Data-driven Model for Assessing User Interface Design. arXiv:2404.12500 [cs.HC] https://arxiv.org/abs/2404.12500

[20] Mingyuan Zhong, Ruolin Chen, Xia Chen, James Fogarty, and Jacob O. Wobbrock. 2025. ScreenAudit: Detecting Screen Reader Accessibility Errors in Mobile Apps Using Large Language Models. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 1163, 19 pages. doi:10.1145/3706598.3713797