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

Comparing Agentic Generative UI and Traditional Interfaces: A Controlled Study of Goal-Based AI-Generated Interfaces in Transactional Workflows

Pradeep Periasamy[1]

Submitted: 03/07/2026 **Revised**: 18/08/2026 **Accepted**: 25/08/2026

Abstract: The advent of agentic interfaces is allowing artificial intelligence to create user interfaces on the fly based on layout designs that are appropriate for the user's purpose and not for a predefined interface design created at the time of design. One practical issue then arises for product teams: does an agentic interface designed based on the user's purposes lag behind a conventional interface in terms of efficiency in a transactional process? In this research paper, results from a controlled between-subjects experiment are presented. The experiment included a multi-step process of booking the cheapest flight that arrived prior to a particular time with only one stopover. The experiment involved 52 subjects performing this task using either a traditional production-style interface or an agentically generated interface that varied how the same flight information was presented while preserving the complete dataset and all user decisions. The following measures were used to assess the time taken to complete the task, clicks made, back navigation count, perceived workload, usability, trust, perceived control, and satisfaction. The two interfaces did not differ in any of the measures mentioned, and all the subjects completed the task successfully. An analysis of equivalence using two one-sided tests with a margin of ±0.5 standard deviations proved equivalence only in terms of clicks and back navigation count, whereas experiential measures, including trust and perceived workload, were inconclusive, showing neither a reliable difference nor confirmed equivalence. These findings show that there is no disadvantage associated with using the agentically generated interface compared with a proven interface, provided that the information completeness and decision-making control of the user are maintained. The pertinent question is not whether agentically generated interfaces outperform mature interfaces but whether there is any disadvantage associated with their use; here, none was detectable.

Keywords: *Agentic user experience, Generative user interface, Human–AI interaction, Transactional workflows, Trust in automation, Usability*

1. Introduction

For most of the history of graphical computing, the mapping between information and its on-screen representation has been fixed at design time. The designer assumes that flight sets are displayed vertically as cards, prices are displayed as right-aligned numbers, and filters are in the left rail; every user is presented with this rendering. Recent developments in the field of foundation models are challenging this assumption. Language and multimodal models can now synthesize interface code, assemble components, and select layouts on demand [9], [15], and industry standardization, most visibly the Agent–User Interaction (AG-UI) protocol and generative-UI specifications such as A2UI, has reframed agent-produced interfaces as a first-class runtime capability. In this emerging paradigm, which we refer to generically as generative UI, an agent generates an interface to pursue the user's goal rather than populating a single predetermined layout.

Most excitement about generative UI has thus far been predicated on its promise to enhance conventional interfaces. However, conventional interfaces that have been refined in an established industry do not just spring up out of nowhere; rather, they have benefited from years of iteration and convergence. An established flight booking interface is an excellent starting point. The more consequential question for a team weighing whether to deploy a generative UI is therefore not whether it can beat such an interface, but whether it can match it: can an agentic, goal-based generative UI deliver comparable efficiency and experience to a traditional interface that users already trust and know how to use? Parity is a meaningful and sufficient result, because it establishes that the flexibility of generative UI need not come at the expense of the qualities that make established interfaces effective.

A central methodological commitment of this study is that the generative UI must not alter the information available to the user. While it might seem helpful to a naïve generative interface that three flights are recommended instead of ten, that is a mistake of conflating presentation with information filtering, for anything that might have been different about the decision could be blamed on having made fewer choices to begin with. In order to have the comparison mean something in terms of testing the generative interface, as opposed to curating the information for the person using the system, the agentic condition used precisely the same flight data set, along with decision affordances, as the control, differing solely in presentation. The agent was goal-aware; it knew the participant's booking objective, but it could not remove options or make the decision on the user's behalf.

This article reports a controlled between-subjects experiment in which participants completed an identical multi-step flight-booking task under one of two conditions: a traditional, production-style interface or an agentic generative UI condition in which an AI generated the interface to achieve the same booking goal. We expected that the agentic generative UI would perform just as well as the conventional user interface in terms of effectiveness and experience, since it contained the entire set of information and retained the user's decision-making power. The

[1] *Independent Researcher, Pittsburgh, PA – 15205, USA*
ORCID ID : 0009-0002-9309-5960

goals of the study are thus: (1) To propose a non-inferiority assessment of agentically generated UI in the context of the information preservation constraint; (2) To investigate, using both quantitative and qualitative assessments, if such an interface fails to perform as well as a more developed, traditional interface in a transactional task; and (3) To formulate a decision-making criterion for those contemplating the use of generative UI.

2. Related Work

2.1. Agentic LLMs: From Reasoning-and-Acting to Tool Agent User Interaction

The interfaces studied here are agentic: they are produced by models that pursue a user's goal rather than rendering a designer's fixed layout. This capability rests on an established line of work on LLM agents. ReAct demonstrated that interleaving of reasoning traces with actions aids in task completion and understandability within interactive settings [1], whereas Toolformer proved that models are able to learn when to call an external tool and incorporate its output into their workflow [2]. WebShop framed grounded web interaction as sequential decision-making and exposed both the competence and the brittleness of agents operating over multi-step interfaces under uncertainty [3]. Subsequent benchmarking expanded the approach into practical scenarios involving web and operating systems: Mind2Web for generalist web agents [4]; WebArena for self-hosted realistic web tasks [5]; AgentBench for evaluating LLMs in multiple environments as agents [6]; and OSWorld for open-ended tasks on real computer systems [7].

Specifically pertinent to this work, τ-bench studies tool-agent-user interaction within transactional domains in the real world, such as airline and retail booking, thereby moving the focus from the possibility of the agent's activity to its behavior during collaboration with the user [8]. This trajectory establishes that goal-directed agents are now capable of driving an interface in a transactional workflow such as flight booking, and reflects a field-wide pivot from raw task success toward the quality of the human-facing interaction, precisely the dimension a generative UI must satisfy if it is to match a mature conventional interface.

2.2. Generative UI and LLM-Driven Interfaces

A rapidly growing body of work studies interfaces that LLMs construct on demand rather than retrieve from a fixed design. The one that comes the closest to it is the paradigm of Generative Interfaces, where the model actively generates a UI for the task rather than a linear text response. This type of model is said to be superior to others for the offloading of cognitive tasks and the credibility of structured presentation [9]. Recent research has taken this to the user end, with the advent of malleable interfaces based on task-evolving data models where the user can curate what information gets presented and how [10], as well as with malleable overview-detail interfaces where the user can change the views generated [11]. Previous frameworks proved feasibility at the component level; DynaVis synthesized UI components for visualization editing [12], and GenerativeGUI generated dynamic interfaces on top of conversational systems [13]. The very question posed is now under active discussion within the research community [14].

There are two key differences between this body of work and the current investigation. First, the comparison baseline is a chat or text interface, where the generative UI interface should prevail, as opposed to a fully developed graphical production interface that the participants are familiar with. Second, the evaluation process has been focused on preference, adaptability, and aesthetic judgments as opposed to a careful experiment involving efficiency, workload, trust, and perceived control, while maintaining the same amount of information content. The present study adopts the more demanding baseline and the more conservative claim: parity, not preference over chat.

2.3. Capability Versus Experience in LLM Interface Generation

A substantial body of work has established that large language models can produce interfaces from high-level inputs, and has built the datasets and metrics needed to measure how well they do so. Web2Code [15] code generation has become a benchmarking task in which visual design is paired with a reference implementation, enabling the generation of interfaces to be evaluated on both fidelity and completeness, and the problem of layout generation has been reformulated as a problem of generating code to uncover the latent structural knowledge of the language model [16]. Another line of research improves generation by using feedback rather than just data; fine-tuning models against automated compiler and visual signals increases the likelihood of compiling and rendering successfully [17]. Recent research has expanded the criteria to include more than just visual accuracy, as in the generation of interfaces accessible to users with assistive technologies [18]. All of these studies have shown that modern models are capable of synthesizing interfaces that are not only semantically correct and visually consistent, but also functional and standards-compliant.

What this work establishes is capability: that a generated interface can be correct. What it does not address is experience: whether a person completing a real task through such an interface is as efficient, feels as much in control, and has as much trust as with an interface they already know. Measures of correctness are made in comparison to reference systems or compliance criteria, rather than in comparison to a user's completion of their goal, and the comparison is usually of models or generative strategies rather than of the generated interface and an established interface system. It is exactly this that our current research addresses: capability is necessary but not sufficient for replacement, and equivalence in actual usage is what makes it possible for a generated interface to reasonably replace a standard one. We take the feasibility these results establish as given, and ask the experiential question they leave open.

2.4. Automatic and Adaptive Interface Generation

The idea that interfaces might be generated rather than hand-designed predates modern LLMs. SUPPLE [19], [20] modeled interface generation as an optimization problem that minimizes the expected amount of work a user would do, and demonstrated that automatically generated interfaces could increase speed, accuracy, and satisfaction, particularly among those who deviated from an able-bodied standard. This tradition contributes two enduring lessons: that generated interfaces should be grounded in a model of the user or task, and that their value is ultimately an empirical question to be settled against a sensible baseline. SUPPLE-style systems, however, adapt to device and ability rather than to an agent's goal-directed interpretation of content, and were not evaluated for trust or perceived control in agentic terms. We inherit

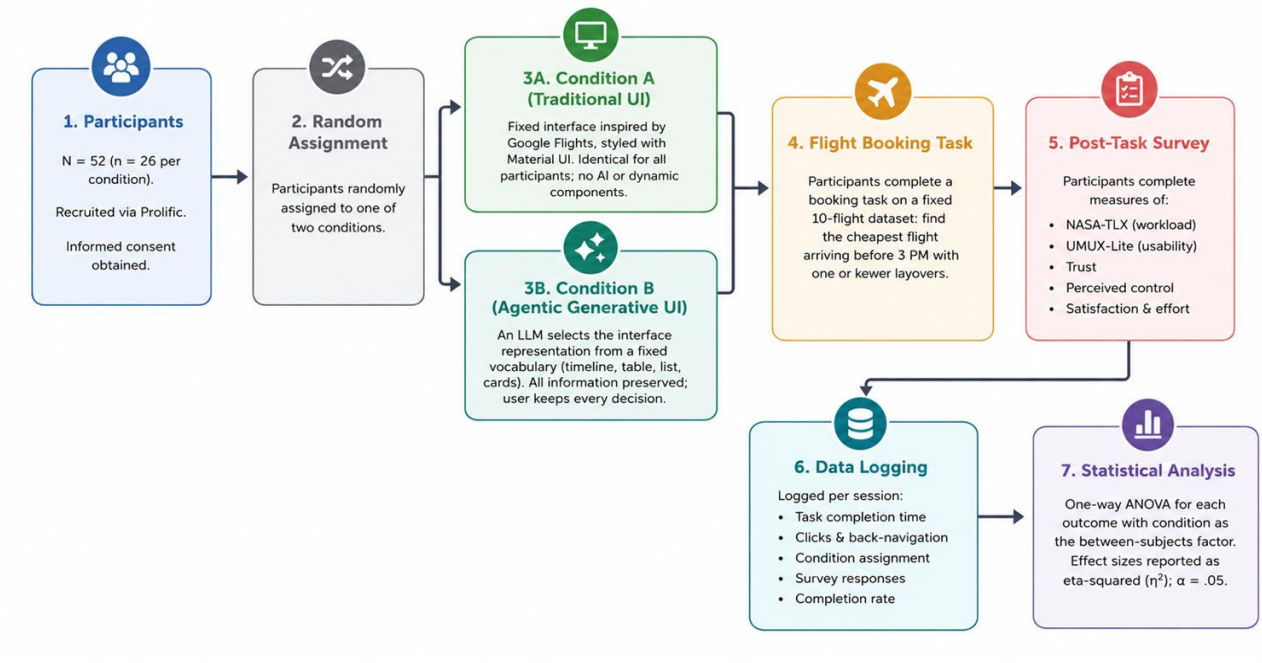

Figure 1. Study design. Participants (N = 52) were randomly assigned to a traditional fixed interface (A) or an agentic generative UI (B), then completed an identical flight-booking task and post-task survey. Outcomes were analyzed by one-way ANOVA with condition as the between-subjects factor..

the framing of generated interfaces as something to be empirically validated against a baseline, while shifting the basis of generation to a goal-aware model and the baseline to a mature production-style interface.

2.5. Familiar Interfaces, Trust, and Why Parity Is Non-Trivial

Whether a novel interface can match a familiar one is not a foregone conclusion. In terms of automation and human–AI interaction, trust depends on predictability and alignment with user expectations, not capability alone [21], [22]. Moreover, change in itself can be experienced as a cost even if objective performance is unaffected. Mixed initiative design principles stress that systems trying to reach an objective for a user must continue to be predictable and keep the user oriented [23], and human-AI interaction guidelines highlight intelligibility, controllability, and graceful failure [24]. In addition, from a human-centered approach towards AI, the importance of having human control is stressed while using automation [25]. Together, these make parity a substantive hypothesis: a goal-directed interface that changes presentation could plausibly depress trust or perceived control relative to a layout users already know, even while completing the task just as quickly. Testing whether it does across efficiency, usability, workload, trust, perceived control, and satisfaction is the empirical contribution of this work.

3. Method

3.1. Task and Materials

This was a simulation of the process of booking an air travel ticket. Participants had to find the cheapest air ticket that lands before 3:00 pm and has one or zero stopovers. A fixed dataset of ten flights was used, which included price, number of stops, departure and arrival time, and refundable or non-refundable information about the flight. Participant details and payment details were pre-filled and were only read-only since the experiment involves flight ticket finding and navigation, not filling out the form. The multi-step flow was identical across conditions.

3.2. Study Design and Conditions

A controlled between-subjects experiment was conducted with interface type as the independent variable, operationalized at two levels (Fig. 1). The task structure, task content, the flight dataset, and participant goal were kept constant throughout the experiment; the only variable was the interface.

3.2.1. Condition A (Traditional Interface)

The participants interacted with a static interface that followed an interaction pattern derived from Google Flights [28] and visual design based on Material UI [29]. The same interface was shown to each participant in this condition with the layout predetermined at compile time. The flight-selection screen displayed all ten flights as a standard vertical list of flight cards, each card surfacing the same attributes in the same visual hierarchy — price, departure and arrival times, number of stops, and refundability — independent of the task. Conventional filter and sort controls were provided as optional, user-initiated affordances. The goal of the task was predetermined for the experimental setup and shown to the subjects; however, the interface was agnostic of the goal, presenting all flights equally, irrespective of the goal, without any modification to the interface. In other words, the participant had to convert the goal into the process of filtering and sorting through options, while at the same time keeping the constraints (lowest price, arriving before 3:00 PM, only one stop) in working memory as he considered each choice. There were no recommendations from an AI, no AI-driven layout, no dynamic selection of components, and no agent, chat, or reasoning text; the interface implemented the same interaction paradigm as a commercial travel booking product, without mimicking any particular service. The workflow followed a fixed sequence: trip search, flight selection, passenger information (prefilled, read-only), optional add-ons (skippable), payment (prefilled mock data, read-only), and a final

review-and-confirm screen. Because the passenger and payment fields were non-editable, the condition isolated flight selection and navigation rather than form filling. This condition served as the mature, widely accepted baseline against which the generative interface was evaluated.

3.2.2. Condition B (Agentic Generative UI)

Participants performed the exact same task using the exact same set of ten flights with the exact same pre-specified goal and displayed in the exact same manner, except that the representation used in the flight selection screen was chosen by the LLM agent dynamically. Given the stated goal, the agent chose how to present the flight information, but it did not generate arbitrary markup or freeform layout. Instead, it selected from a constrained, predefined vocabulary of five recognizable UI patterns, each implemented and validated in advance: a *ranked list*, suited to cases where a single attribute dominates such as price; a *comparison table*, for side-by-side evaluation across multiple attributes; a *timeline view*, when the temporal distribution of departures and arrivals is salient; *compact cards*, for rapid scanning when many options are present; and *metric tiles*, when the decision space is small enough that summary statistics suffice. Use of such restricted vocabulary was a design strategy that aimed at three objectives: Ecological validity, as each pattern is a real-life interface pattern that would be recognized by the participants as opposed to an imaginary layout; experimental control, as it helps in attributing the variations to the choice of representation format and not the novelty of the interface or rendering; and information retention, as all the patterns in the vocabulary can represent all flights with all attributes. The key limitation with respect to whatever choice of representation the agent made was thus that of retaining information: whatever screen was produced would show all ten flights, along with the full set of attributes and decision capabilities contained within the baseline version, without filtering or summarizing away any of it. The agent was aware of the goal, and utilized its awareness of the goal to structure the decision-making environment based on the goal; however, it did not possess any decision-making power; it could neither choose nor book a flight, nor eliminate options nor take any other action for the participant, and every downstream step (passenger information, add-ons, payment, confirmation) was identical to Condition A. In this way, the manipulation involved changing the actor that controlled the structure of the flight selection screen, a design-time one versus a runtime one, while keeping the task, the information, the entire array of choices, and the control of the decision-making process constant.

3.3. Generative UI Implementation

The agentic generative UI was realized through a single planning call made once in the initial setup of the experiment through offline inference with the large language model rather than live inference of the UI during the sessions of the participants. With the task and dataset remaining the same, there was only a need for a single call to generate a UI plan, which was deterministically embedded into the Condition B application. Therefore, each participant in Condition B received the same selected representation from the agent, thereby removing runtime stochasticity and isolating representation selection as the manipulated variable while holding the underlying information constant.

Limited vocabulary of representation. The agent did not create its own markup or formatting. It chose from a list of predetermined user interface objects, one object per step, all of which had been pre-implemented and tested to represent the entire database. For the flight-selection screen, the agent picked one of four layouts: a simple vertical list, a three-column grid of compact cards, a sortable side-by-side table, or a timeline ordered by departure time where bar length shows how long each flight takes. It also picked a default way to sort the flights: by the order they appeared in the data, by price, by how long the flight takes, or by arrival time. For the add-ons step, it chose between grouping items under category headings, showing them all in one plain list, or showing ready-made package deals first. For the confirmation step, it chose between a single all-in-one summary card or a checklist-style layout that breaks the summary into completed steps. We limited the agent to this fixed menu of options for three reasons. First, every option is a layout style people already recognize from everyday apps, so the study tests something realistic rather than something unfamiliar. Second, keeping the choices to this fixed list means any differences we find can be traced back to which layout was picked, rather than to one version simply looking rougher or more novel than the other. Third, every option on the list can still show all ten flights and every one of their details, so no matter which layout the agent picked, nothing was ever hidden or left out.

The model was presented with the task given to the participant ("Find the least expensive flight arriving before 3 PM with one stop or less"), the entire set of flights (ten flights, including all attributes), the full list of add-ons and bundles, followed by the list of components along with a choice criterion for the representation that facilitates the performance of the task given cognitive load, scannability, the task constraints, and the number of options. It was required to return a single JSON object specifying the component, default sort, and default filter for flight selection, the add-ons component, and the confirmation layout, each accompanied by a short rationale, with no prose outside the JSON.

The following model configuration was used. The prompt was created by making a single call to Claude Sonnet 4.6 (Anthropic) using zero-shot prompting at temperature 0, with an output cap of 1,024 tokens and schema-constrained JSON output format on June 7th, 2026. There were no few-shot examples, data from participants, or behavioral cues used in the creation of the prompt; the model simply made inferences about the structural nature of the task and dataset. Given that the plan had been pre-computed and embedded, no inferences were made during the sessions, and Condition B was consistent across all participants.

Resulting interface plan. For flight selection, the agent selected the FlightTimeline component with an arrival-time sort; for add-ons, the GroupedList; and for confirmation, the SummaryCard. Justification provided by the agent is that arranging the flights in terms of the arrival time will turn the 3 PM deadline into a cut-off point in the arc-bar format, enabling the user to select the flights without having to understand the actual time strings, following which the user can look for the price; the add-on list based on categories will reduce the effort required for scanning during the less important decision-making process; and a single summary is best suited for verification purposes. In order to maintain the constraint for information retention, the default filter was chosen to display all ten flights, as opposed to the “one stop or fewer” filter that the agent suggested, in order to ensure that all flights were shown. This was the only deviation from the agent's returned plan; all other selections of the FlightTimeline component, the GroupedList add-ons layout, and the SummaryCard confirmation were deployed exactly as the model specified. The filtering mechanism was thus set up in such a way as to show all the choices to the participant without withholding any flight and making no

pre-decisions on what choice to make; the filtering and sorting tools were available in full capacity, allowing the participant rather than the system to decide which choices he wants to filter or rearrange. The component's sort and filter controls were present and user-adjustable throughout, so the agent's choices set the initial view rather than restricting the options a participant could reach.

3.4. Measures

Objective measures, captured through interaction logging, comprised total task completion time, total click count, back-navigation count, per-step dwell time, and task completion. Subjective measures, collected in a post-task survey, comprised perceived workload using the NASA Task Load Index (raw, computed as the unweighted mean of six subscales) [26], perceived usability using UMUX-Lite [27], a single-item perceived-effort rating, and task-grounded Likert items (1–7) for trust, perceived control, and satisfaction [21].

3.5. Participants

Fifty-two participants (n = 26 per condition) were recruited via Prolific. To be eligible, one had to be an adult speaking fluent English and have computer literacy. Informed consent was obtained from all participants in order to compensate them fairly. Participants were randomly assigned to conditions to minimize selection bias and support causal interpretation.

3.6. Procedure

The participants initially saw the instruction page that provided the scenario about booking and the purpose of the task. Then, they performed the task based on their conditions, after which they filled out the post-task questionnaire.

3.7. Logging, Data Quality, and Exclusions

The timestamps of interactions, clicks, and navigations, time spent at each step, chosen flight, and responses to surveys were instrumented on the client side and stored locally until sending, when done. Data were not analyzed for sessions with incomplete data, invalid logs, or missing timestamps.

3.8. Analysis Plan

The effect of condition on each outcome variable was analyzed using one-way between-subjects analysis of variance (ANOVA) with condition as the factor; effect sizes are expressed as η^2. For two levels of condition, the omnibus ANOVA is identical to the independent-samples t-test; ANOVA is reported for consistency with prior work in this program. Where parametric assumptions were violated, robust or non-parametric alternatives were used and conclusions confirmed to be insensitive to the choice of test. To evaluate whether the conditions were indeed equivalent as opposed to just being non-different, two one-sided tests (TOST) were done on all outcomes based on a predetermined equivalence margin of ±0.5 standard deviations (Cohen's d=0.5), which is the general limit for an effect size of importance in case no minimum important difference has been specified for that particular domain. The criteria for equivalence were that the 90% confidence interval of the standardized mean difference would lie within ±0.5d.

No a priori power analysis was carried out; however, given the actual sample size of 26 subjects per condition, a post-hoc power analysis revealed that a two-tailed independent samples t-test (α = .05, 80% power) would be able to find a minimum effect size of d ≈ 0.79. This means that the experiment was well-powered to find large effects but had low power to detect small and medium-sized effects, a limitation that directly informs the equivalence-testing results below.

Since there are nine outcomes under consideration, it should be pointed out that none of the comparisons have achieved statistical significance even without making a correction for multiple testing; making a correction will simply increase the width of the nonsignificant interval without changing anything. In order to make it possible to interpret the data as being noninferior, the 95% confidence interval for the difference in means between the conditions is also presented for each of the outcomes, since the magnitude of the difference the data still permit is more informative than significance alone.

4. Results and Discussion

4.1. Completion and Overview of Outcomes

There was a 100% success rate among all 52 participants who conducted the reservation procedure, regardless of the type of interface used. The omnibus ANOVA statistics of the measures are given in Table 1, while Table 2 provides descriptive statistics by condition. No measure showed any significant difference between the two interfaces.

Table 4 contains the between-condition mean difference (A - B) and 95% confidence interval for each dependent measure, in the units of measurement for the respective measure. All intervals contain zero, and for the experience measures, the likely true difference is limited to within about one scale point or less.

Table 1. Omnibus one-way ANOVA results by outcome (N = 52; n = 26 per condition).

Outcome measure	df	F	p	η^2
Task completion time (s)	(1, 50)	0.29	0.592	0.006
Total clicks	(1, 50)	0.01	0.944	< .001
Back-navigation count	(1, 50)	0.01	0.916	< .001
Perceived workload (NASA-TLX)	(1, 50)	1.38	0.247	0.027
Usability (UMUX-Lite)	(1, 50)	0.36	0.553	0.007
Perceived effort	(1, 50)	0.02	0.892	< .001
Trust	(1, 50)	1.52	0.223	0.030
Perceived control	(1, 50)	0.06	0.810	0.001
Satisfaction	(1, 50)	0.70	0.407	0.014

df = Degrees of freedom are reported as (between groups, within groups), F = F-ratio from a one-way analysis of variance, p = p value, s = second, η^2= effect size, the proportion of variance in the outcome explained by condition.

Table 2. Descriptive statistics by condition: mean (standard deviation).

Outcome measure	A: Traditional, M (SD)	B: Generative UI, M (SD)	A: Mdn (IQR)	B: Mdn (IQR)
Task completion time (s)	278.6 (139.5)	305.8 (215.0)	248 (173–298)	251 (147–371)
Total clicks	14.6 (6.7)	14.4 (8.8)	11 (11–18)	11 (11–14)
Back-navigation count	1.9 (3.3)	1.8 (4.5)	0 (0–4)	0 (0–2)
Perceived workload (NASA-TLX, 0–20)	5.81 (3.60)	4.58 (3.93)	5.08 (2.92–8.63)	3.00 (1.54–7.12)
Usability (UMUX-Lite, 0–100)	88.8(13.9)	86.5 (13.1)	91.7 (83.3–100.0)	83.3 (83.3–100.0)
Perceived effort	3.42 (2.21)	3.50 (1.82)	3.0 (1.2–5.8)	3.0 (2.0–5.0)
Trust (1–7)	6.23 (0.82)	5.92 (0.98)	6.0 (6.0–7.0)	6.0 (5.2–7.0)
Perceived control (1–7)	6.08 (1.16)	6.00 (1.13)	6.0 (6.0–7.0)	6.0 (5.0–7.0)
Satisfaction (1–7)	6.38 (0.94)	6.15 (1.05)	7.0 (6.0–7.0)	6.0 (6.0–7.0)

Note. M = Mean; SD = Standard Deviation; Mdn = Median; IQR = interquartile range (25th–75th percentile). Both summaries are reported for all outcomes; for the right-skewed time- and count-based measures, the median is the more representative estimate of central tendency.

4.2. Equivalence Testing

Since ANOVA is not significant and hence does not provide any evidence of equivalence, TOST tests were performed against a margin of ±0.5 d (Table 3). Total clicks (p=.045) and back navigation count (p=.048) were within the margin criterion of 90% confidence interval, while the rest did not show any equivalence. This includes trust, perceived workload, usability, perceived control, satisfaction, and task completion time, as their 90% confidence intervals either failed to exclude zero or fell within the margin. The data therefore indicate an absence of detectable differences but do not constitute positive evidence of equivalence for the experiential measures at a medium margin; the observed standardized differences for trust (d = 0.34) and workload (d = 0.33), though non-significant, remain compatible with effects up to roughly d = 0.8.

Table 3. Equivalence test (TOST) results by outcome against a ±0.5 d margin (N = 52; n = 26 per condition).

Outcome measure	Cohen's d	90% CI (d)	p (TOST)	Equivalent?
Task completion time (s)	-0.15	[−0.61, 0.32]	0.106	No
Total clicks	0.02	[−0.45, 0.48]	0.045	Yes
Back-navigation count	0.03	[−0.44, 0.49]	0.048	Yes
Perceived workload (NASA-TLX)	0.33	[−0.14, 0.79]	0.266	No
Usability (UMUX-Lite)	0.17	[−0.30, 0.63]	0.117	No
Perceived effort	-0.04	[−0.50, 0.43]	0.051	No
Trust	0.34	[−0.12, 0.81]	0.286	No
Perceived control	0.07	[−0.40, 0.53]	0.062	No
Satisfaction	0.23	[−0.23, 0.70]	0.169	No

Note. d = Cohen's d (standardized mean difference, Condition A − Condition B); 90% CI = 90% confidence interval of d, consistent with the two one-sided tests (TOST) at α = .05; p (TOST) = the larger of the two one-sided p values. Equivalence was concluded when the 90% CI fell entirely within the pre-specified margin of ±0.5 d. "No" indicates an inconclusive result — neither a significant difference nor confirmed equivalence — rather than evidence of a difference.

Table 4. Between-condition mean differences (Condition A − Condition B) with 95% confidence intervals, in original units (N = 52; n = 26 per condition).

Outcome measure	Mean difference (A − B)	95% CI
Task completion time (s)	-27.1	[−128.1, 73.8]
Total clicks	0.2	[−4.2, 4.5]
Back-navigation count	0.12	[−2.08, 2.31]
Perceived workload (NASA-TLX, 0–20)	1.22	[−0.87, 3.32]
Usability (UMUX-Lite, 0–100)	2.2	[−5.3, 9.8]
Perceived effort	-0.08	[−1.20, 1.05]
Trust (1–7)	0.31	[−0.19, 0.81]
Perceived control (1–7)	0.08	[−0.56, 0.72]
Satisfaction (1–7)	0.23	[−0.32, 0.79]

Note. A positive difference indicates a higher mean in the traditional interface (Condition A); a negative difference indicates a higher mean in the generative UI (Condition B). All confidence intervals include zero. The direction of "better" depends on the measure: for trust, usability, perceived control, and satisfaction, higher is better; for completion time, clicks, back navigation, workload, and effort, lower is better.

4.3. Efficiency: Time, Clicks, and Navigation

Task completion time did not differ significantly between conditions, $F(1, 50) = 0.29$, $p = .592$, $\eta^2 = .006$, with means of 278.6 s for the traditional interface and 305.8 s for the generative UI. Neither total clicks, $F(1, 50) = 0.01$, $p = .944$, nor back-navigation count, $F(1, 50) = 0.01$, $p = .916$, differed. Because the time- and count-based measures were right-skewed, medians are more representative than means (Table 2); the near-identical medians for completion time (248 vs 251 s), total clicks (11 vs 11), and back-navigation (0 vs 0) reinforce the absence of an efficiency difference. As can be seen from Fig. 2, the confidence intervals of both conditions have significant overlap for all efficiency metrics. Therefore, the use of the agentic generative user interface did not incur any efficiency penalty compared to the mature one.

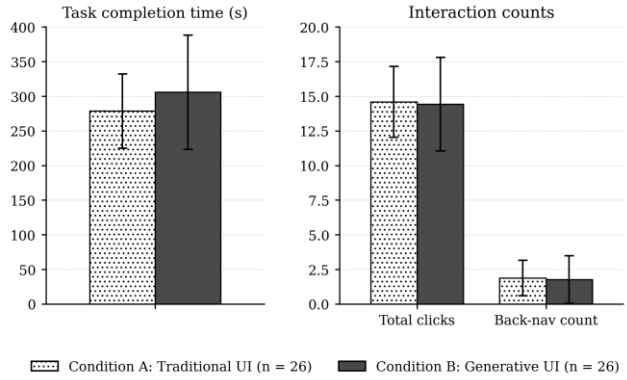

Fig 2. Mean task completion time and interaction counts by condition, with 95% confidence intervals (n = 26 per condition). Panels are shown on their respective scales.

4.4. Workload, Usability, and Effort

Perceived workload did not differ significantly, $F(1, 50) = 1.38$, $p = .247$, $\eta^2 = .027$, although the generative UI showed a directional, non-significant reduction (4.58 versus 5.81 on the 0–20 scale). Usability, $F(1, 50) = 0.36$, $p = .553$, and perceived effort, $F(1, 50) = 0.02$, $p = .892$, likewise did not differ, and usability ratings were high in both conditions (88.8 and 86.5 on a 0–100 scale). Fig. 3 displays these outcomes, each on its native scale; the overlapping confidence intervals confirm the absence of reliable differences.

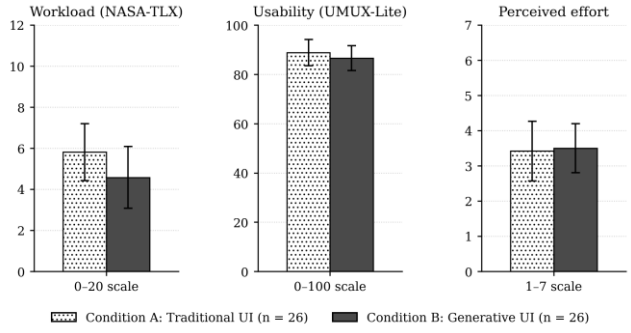

Fig 3. Mean perceived workload (NASA-TLX), usability (UMUX-Lite), and perceived effort by condition, with 95% confidence intervals (n = 26 per condition).

4.5. Trust, Perceived Control, and Satisfaction

Trust did not differ significantly between conditions, $F(1, 50) = 1.52$, $p = .223$, $\eta^2 = .030$, and remained high in both (6.23 and 5.92 on a 1–7 scale). Perceived control, $F(1, 50) = 0.06$, $p = .810$, and satisfaction, $F(1, 50) = 0.70$, $p = .407$, were similarly high and showed no statistically detectable difference, as shown in Fig. 4. Notably, the agent-selected interface evaluated here did not erode trust or perceived control, the dimensions on which delegating authority to an AI most often exacts a cost.

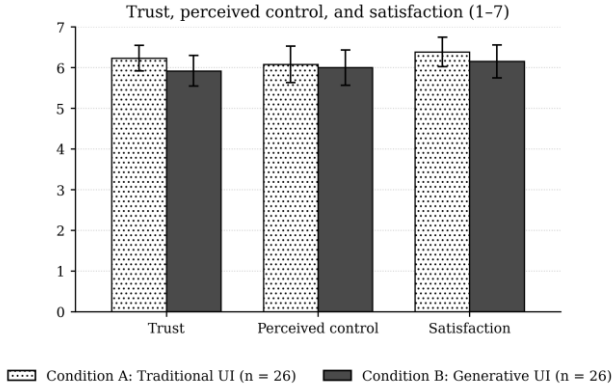

Fig 4. Mean trust, perceived control, and satisfaction ratings (1–7 scale) by condition, with 95% confidence intervals (n = 26 per condition).

5. Discussion

The strength of this non-inferiority claim rests on the standing of the baseline against which it was measured. Condition A was not an arbitrary or deliberately weakened control but an interface whose interaction model was drawn from Google Flights [28], one of the most widely used flight-search products in the world, and whose visual treatment followed an established design system, Material UI [29]. Matching such a baseline is a more stringent test than the chat-versus-generative comparisons common in the generative-UI literature, where the reference interface is a linear text exchange and structured presentation is expected to outperform. Since users came to the test with established expectations regarding the expected look and behavior of a flight booking interface, any confusion resulting from AI selection of the interface representation had ample time to manifest itself in slower task completion, greater numbers of user backtracking, and reduced levels of trust. The absence of such factors suggests that in this case at least the AI-generated interface is not only acceptable but also interchangeable with an established reference design.

On the other hand, the trend of directional and non-significant differences should be read with caution. The generative user interface did not perform any better than its traditional counterpart on any of the efficiency metrics and demonstrated a slight non-significant decrease in workload perception, whereas the latter had a slight lead in terms of trust and satisfaction. In relation to the current sample, this is entirely within the bounds of variation due to sampling; while we have carried out equivalence testing, the sample size was not sufficient to prove equivalence with a medium margin in the experiential variables, so for these results, the proper interpretation is absence of difference rather than equivalence. The implication is clear, nevertheless: since representation may be delegated to an agent without a reliable penalty, the question to explore further is not one of the safety of the use of generative UI within a transactional process, but rather that of the situation where the flexibility of generative UI provides some real advantage such as personalization to user goals, adaptation across devices, or accessibility, while respecting the information-preservation constraint identified here as a design invariant.

The difference intervals between conditions for both trust and workload (Table 4), like those in the equivalence tests, are sufficiently large that a genuine difference can still exist, even adding to the above cautionary interpretation.

6. Limitations

There were several limitations in support of these findings. For instance, the research was done using only one task during one testing session; therefore, it did not address the question of how non-inferiority changes upon repeated trials or with other kinds of tasks. Additionally, there were two conditions tested without an intermediate one; for example, an AI recommendation with a user's ability to override it in order to determine the place, if any, where generative assistance starts to be helpful or harmful. The sample size, while consistent with controlled human–computer interaction studies, was a primary limitation. A sensitivity analysis showed that the design (n = 26 per condition) could detect only effects of d ≈ 0.79 or larger at 80% power; it was thus underpowered for the small-to-medium effects most relevant to a non-inferiority claim. This is not a peripheral caveat: the two largest observed differences, trust (6.23 vs 5.92; d = 0.34) and perceived workload (5.81 vs 4.58, a directional ~21% reduction; d = 0.33), were each accompanied by an achieved power of only about 0.21–0.23, meaning the study had little ability to adjudicate whether these represent real differences or noise. The non-significant ANOVAs and the inconclusive TOST results for these measures should therefore be read as an absence of evidence rather than evidence of absence; confirming equivalence on trust and workload would require a substantially larger sample (on the order of 70–100 per condition for a ±0.5 d margin). The manipulation by means of agentic behavior was constrained by the representation and its ordering at the beginning; since the interface created in the experiment itself defined a default order according to the task, the representation and ordering are inseparable within this particular experimental setup, and the results cannot be generalized to other generative interfaces that do not preserve information.

A second primary limitation concerns the generality of the agentic condition. The interface was produced by a single deterministic language model call (temperature 0), made once during study setup, and was identical for all participants. This was a deliberate methodological choice: by constructing the interface once and then embedding it, we were able to eliminate variability in the performance of the model across runs, ensure that all participants in the condition would experience the exact same well-designed and fully functional interface, and separate the manipulation of interface representation from the stochasticity of live construction. The trade-off is external validity: the study evaluates a single realized interface rather than a sample from the distribution of interfaces that such a system could produce. A different prompt, model, or random seed could yield a different representation, so the present results speak to the specific FlightTimeline-based design the agent selected, not to agent-generated interfaces in general. Establishing whether non-inferiority holds across the space of interfaces a generative system can produce would require sampling many generated interfaces, ideally with live per-participant generation, and is an important direction for future work.

7. Conclusion

Because of the ability of foundation models to enable the generation of interfaces dynamically as opposed to designing them beforehand, one question that arises when thinking about the use of such interfaces is whether an agent-oriented and goal-based generative UI is outperformed by the user interfaces that are currently being used. The results of a between-subjects study performed on a multi-step flight booking task revealed that a single interface selected by an agent was not significantly inferior to a full-fledged production interface in terms of efficiency, workload, usability, trust, perceived control, and satisfaction, with all participants completing the task in both conditions. Equivalence testing confirmed statistical equivalence for the interaction-effort measures, while the experiential measures of trust and workload were inconclusive at a medium margin and await confirmation from larger samples. The results have redefined the core issue for generative UIs from superiority to non-inferiority. Within the framework of the constraint that requires preservation of all information and of the user's decision-making power, an interface generated by AI was not shown to have incurred any experiential cost in comparison to a conventional interface, which has been used for a long time. It is essential to create the floor of non-inferiority before using such interfaces in any transaction. This is how we move towards future research, which will include long-term usage, governance through intermediaries, and large samples.

7.1. Appendix: Reproducibility - Generative UI Prompt and Resulting Output

To support reproducibility of the Condition B (agentic generative UI) stimulus, this appendix provides the verbatim prompt issued to the model and the complete interface plan it returned. As described in §3.3, the plan was produced by a single zero-shot call to Claude Sonnet 4.6 (Anthropic) at temperature 0, with a 1,024-token limit and schema-constrained JSON output, executed once on June 7, 2026. The returned plan was embedded deterministically into the Condition B application; no model inference occurred during participant sessions.

7.1.1. Verbatim prompt

You are an interface designer selecting the optimal UI representation for an HCI flight booking study.
Participant task:
"Find the cheapest flight that arrives before 3 PM and has 1 stop or fewer."
Flight Data (10 options): [the fixed ten-flight dataset, each with id, label, price, stops, departure_time, arrival_time, airline, refundability, duration].
Available Add-Ons: [the eleven-item add-on catalog, each with id, label, price, group].
Available Bundles: [three pre-composed bundles, each with id, label, addon_ids, price, description].
Component Menu - choose one option per step:
Flight Selection Component (choose one): FlightList - standard vertical list; CompactCards - three-column card grid; ComparisonTable - sortable attribute table; FlightTimeline - departure-ordered timeline with duration arc bars.
Add-Ons Component (choose one): GroupedList - items under category headers; FlatList - single ungrouped list; BundleCards - pre-built bundles shown first.
Flight Selection Default Sort (choose one): best (dataset order); price (cheapest first); fastest (shortest duration); arrival (earliest arrival first).
Flight Selection Default Filter (choose one): all (show all 10

flights); nonstop (nonstop only); 1stop (one stop or fewer).
Confirmation Layout (choose one): SummaryCard - single consolidated card; ChecklistStyle - completed-checklist sections.
Selection Criteria: Choose the interface that best helps a user efficiently accomplish the specific task above. Consider cognitive load, information scannability, the constraints (price + 3 PM arrival + ≤1 stop), and number of options.
Return ONLY a valid JSON object [with fields flight_selection {component, default_sort, default_filter, reasoning}, addons {component, reasoning}, confirmation {layout, reasoning}], no prose, no markdown fences.

7.1.2. Resulting interface plan (model output)

The model returned the following selections (reasoning fields abbreviated):
flight_selection: component = "FlightTimeline", default_sort = "arrival". Reasoning (abridged): the two hard constraints are arrival before 3 PM and ≤1 stop; sorting by arrival time makes the 3 PM cutoff visually obvious in the timeline's arc-bar layout, after which price can be scanned among qualifying options.
addons: component = "GroupedList". Reasoning (abridged): organizing eleven items under four category headers (Seating, Baggage, Meals, Extras) reduces scanning effort during a secondary, lower-stakes decision.
confirmation: layout = "SummaryCard". Reasoning (abridged): a single consolidated card lets the user verify the selected flight against the original constraints before finalizing.

7.1.3. Deviation from the returned plan

One field was overridden at deployment. The model proposed default_filter = "1stop", which would have hidden the single two-stop flight on load. To uphold the information-preservation constraint, this was set to "all" so that every flight remained visible; the filter and sort controls themselves remained user-adjustable. All other selections (FlightTimeline, arrival sort, GroupedList, SummaryCard) were deployed exactly as returned. This is the only modification made to the model's output.

7.1.4. Interface screenshots

Figures 5 and 6 show the flight-selection screen as presented in Condition A (traditional vertical list) and Condition B (agent-selected FlightTimeline), respectively, both displaying the identical ten-flight dataset.

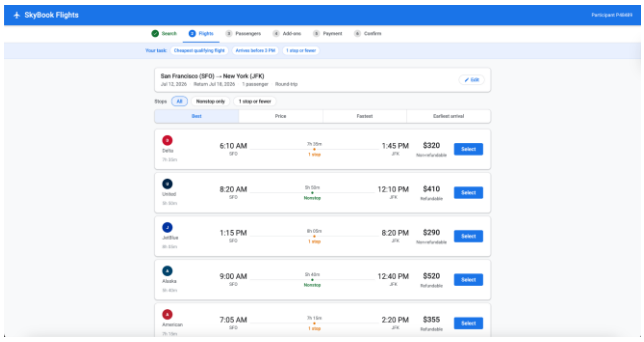

Fig 5. Condition A: flight-selection screen showing all ten flights as a standard vertical card list with filter and sort controls.

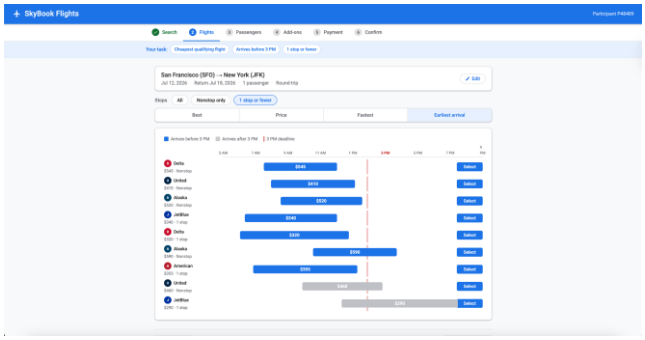

Fig 6. Condition B: the same ten flights in the agent-selected FlightTimeline representation, ordered by arrival time, with all options visible and controls user-adjustable.

8. References

[1] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in Proc. 11th Int. Conf. Learning Representations (ICLR), 2023.

[2] T. Schick, J. Dwivedi-Yu, R. Dessì, et al., "Toolformer: Language models can teach themselves to use tools," in Advances in Neural Information Processing Systems 36 (NeurIPS), 2023.

[3] S. Yao, H. Chen, J. Yang, and K. Narasimhan, "WebShop: Towards scalable real-world web interaction with grounded language agents," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 35, 2022.

[4] X. Deng, Y. Gu, B. Zheng, et al., "Mind2Web: Towards a generalist agent for the web," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 36, Datasets and Benchmarks Track, 2023.

[5] S. Zhou, F. F. Xu, H. Zhu, et al., "WebArena: A realistic web environment for building autonomous agents," in Proc. 12th Int. Conf. Learn. Represent. (ICLR), 2024.

[6] X. Liu, H. Yu, H. Zhang, et al., "AgentBench: Evaluating LLMs as agents," in Proc. 12th Int. Conf. Learn. Represent. (ICLR), 2024.

[7] T. Xie, D. Zhang, J. Chen, et al., "OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments," in Advances in Neural Information Processing Systems 38, Datasets and Benchmarks Track (NeurIPS), 2024.

[8] S. Yao, N. Shinn, P. Razavi, and K. Narasimhan, "τ-bench: A benchmark for tool–agent–user interaction in real-world domains," in Proc. 13th Int. Conf. Learning Representations (ICLR), 2025.

[9] J. Chen, Y. Zhang, Y. Zhang, Y. Shao, and D. Yang, "Generative interfaces for language models," arXiv:2508.19227, 2025.

[10] Y. Cao, B. Min, A. Chen, and H. Xia, "Generative and malleable user interfaces with generative and evolving task-driven data models," in Proc. CHI Conf. Human Factors Comput. Syst. (CHI), 2025, doi: 10.1145/3706598.3713285.

[11] B. Min, A. Chen, Y. Cao, and H. Xia, "Malleable overview-detail interfaces," in Proc. CHI Conf. Human Factors Comput. Syst. (CHI), Art. no. 688, 2025, doi: 10.1145/3706598.3714164.

[12] P. Vaithilingam, E. L. Glassman, J. P. Inala, and C. Wang, "DynaVis: Dynamically synthesized UI widgets for visualization editing," in Proc. 2024 CHI Conf. Human Factors in Computing Systems (CHI '24), Art. 985, ACM, 2024, doi: 10.1145/3613904.3642639.

[13]N. Hojo et al., "GenerativeGUI: Dynamic GUI Generation Leveraging LLMs for Enhanced User Interaction on Chat Interfaces," in Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, in CHI EA '25. New York, NY, USA: Association for Computing Machinery, 2025. doi: 10.1145/3706599.3719743.

[14] S. Lindley et al., "What does generative UI mean for HCI practice?", presented at the CHI 2026 Workshop on Generative UI, 2026.

[15] S. Yun, H. Lin, R. Thushara, et al., “Web2Code: A large-scale webpage-to-code dataset and evaluation framework for multimodal LLMs,” in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 37, Datasets and Benchmarks Track, 2024.
[16] Z. Tang, C. Wu, J. Li, and N. Duan, “LayoutNUWA: Revealing the hidden layout expertise of large language models,” in Proc. 12th Int. Conf. Learn. Represent. (ICLR), 2024.
[17] J. Wu, E. Schoop, A. Leung, T. Barik, J. P. Bigham, and J. Nichols, “UICoder: Fine-tuning large language models to generate user interface code through automated feedback,” in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol. (NAACL-HLT), 2024, pp. 7511–7525, doi: 10.18653/v1/2024.naacl-long.417.
[18] J. Yoon, J. Cho, J. Kim, J. Chung, J. Jeon, and Y. Yu, “A11YN: Aligning LLMs for Accessible Web UI Code Generation,” 2025.
[19] K. Z. Gajos, D. S. Weld, and J. O. Wobbrock, “Automatically generating personalized user interfaces with SUPPLE,” Artif. Intell., vol. 174, no. 12–13, pp. 910–950, 2010, doi: 10.1016/j.artint.2010.05.005.
[20] K. Z. Gajos, J. O. Wobbrock, and D. S. Weld, “Automatically generating user interfaces adapted to users’ motor and vision capabilities,” in Proc. 20th ACM Symp. User Interface Softw. Technol. (UIST), 2007, pp. 231–240, doi: 10.1145/1294211.1294253.
[21] J. D. Lee and K. A. See, “Trust in automation: Designing for appropriate reliance,” Hum. Factors, vol. 46, no. 1, pp. 50–80, 2004, doi: 10.1518/hfes.46.1.50.30392.
[22] R. Parasuraman and V. Riley, “Humans and automation: Use, misuse, disuse, abuse,” Hum. Factors, vol. 39, no. 2, pp. 230–253, 1997, doi: 10.1518/001872097778543886.
[23] E. Horvitz, “Principles of mixed-initiative user interfaces,” in Proc. SIGCHI Conf. Human Factors Comput. Syst. (CHI), 1999, pp. 159–166, doi: 10.1145/302979.303030.
[24] S. Amershi, D. Weld, M. Vorvoreanu, et al., “Guidelines for human-AI interaction,” in Proc. CHI Conf. Human Factors Comput. Syst. (CHI), 2019, doi: 10.1145/3290605.3300233.
[25] B. Shneiderman, Human-Centered AI. Oxford, U.K.: Oxford Univ. Press, 2022, doi: 10.1093/oso/9780192845290.001.0001.
[26] S. G. Hart and L. E. Staveland, “Development of NASA-TLX: Results of empirical and theoretical research,” in Human Mental Workload, P. A. Hancock and N. Meshkati, Eds. Amsterdam, The Netherlands: Elsevier, 1988, pp. 139–183, doi: 10.1016/S0166-4115(08)62386-9.
[27] J. R. Lewis, B. S. Utesch, and D. E. Maher, “UMUX-LITE: When there’s no time for the SUS,” in Proc. SIGCHI Conf. Human Factors Comput. Syst. (CHI), 2013, pp. 2099–2102, doi: 10.1145/2470654.2481287.
[28] Google LLC, “Google Flights.” [Online]. Available: https://www.google.com/travel/flights. Accessed: Jun. 15, 2026.
[29] MUI, “Material UI – React component library.” [Online]. Available: https://mui.com/material-ui/. Accessed: Jun. 15, 2026.