UXAgent: An LLM-Agent-Based Usability Testing Framework for Web Design

Yuxuan Lu*
Northeastern University
USA

Bingsheng Yao
Northeastern University
USA

Hansu Gu
Amazon
USA

Jing Huang
Amazon
USA

Zheshen (Jessie) Wang
Amazon
USA

Yang Li
Amazon
USA

Jiri Gesi
Amazon
USA

Qi He
Amazon
USA

Toby Jia-Jun Li
University of Notre Dame
USA

Dakuo Wang
Northeastern University
USA

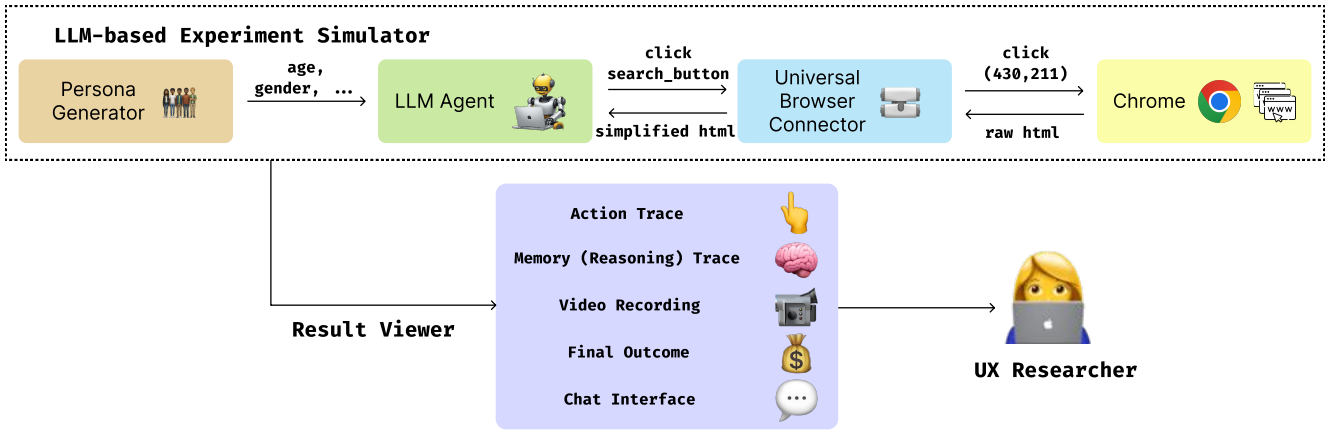


Figure 1: UXAgent System Design. An UX researcher can use the persona generator to generate thousands of user personas and feed them into the LLM Agent. The LLM Agent interacts with the target web design through a Universal Browser Connector. Collected multimodal data are presented to the UX researcher through a Result Viewer.

Abstract

Usability testing is a fundamental yet challenging research method for user experience (UX) researchers to evaluate a web design. Recent advances in Large Language Model-simulated Agent (**LLM Agent**) research inspired us to design UXAgent to support UX researchers in evaluating and reiterating their usability testing study design before they conduct the real human-subject study. Our system features an LLM Agent module and a universal browser connector module so that UX researchers can automatically generate thousands of simulated users to test the target website. The system can generate UX study results in qualitative (e.g., interviewing how an agent thinks), quantitative (e.g., # of actions), and video recording formats for UX researchers to analyze. Through a heuristic user evaluation with five UX researchers, participants praised the innovation of our system but also expressed concerns about the future of UX study with LLM Agents[1].

CCS Concepts

• **Human-centered computing** → **Empirical studies in HCI**; *Interactive systems and tools.*

*This work was done when Yuxuan was an intern, and Dakuo was a visiting scholar at Amazon. Contact email: {lu.yuxuan, d.wang}@northeastern.edu

CHI EA '25, April 26-May 1, 2025, Yokohama, Japan

ACM ISBN 979-8-4007-1395-8/2025/04
https://doi.org/10.1145/3706599.3719729

[1]Our demo and code is available at https://uxagent.hailab.io

Keywords

Usability Testing, User Simulation, Large Language Models, Simulated Agents

ACM Reference Format:

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 *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3706599.3719729

1 Introduction

Usability testing is a fundamental research method for evaluating the user experience (UX) of a new feature or a new web page design [32]. Typically, the UX researcher (or a UI/UX designer, product manager, etc.) leads the design and execution of a usability testing study: they start with a designed new feature, and they need to come up with an experiment design (e.g. A/B testing and the user tasks), recruit participants, execute the study, collect, and analyze the data. However, the current process of usability testing often faces challenges, especially in the experiment design stage and the participant recruitment stage [9, 13, 16, 23]. In the experiment design stage, it is hard for UX researchers to **evaluate and gather early feedback on their experiment design**. After days (or weeks) of experiments, an ill-designed experiment design cannot yield useful results to support the feature design iteration [32]. Another common challenge is that UX researchers often find it **difficult to recruit enough qualified participants**, especially when the target users are from a narrowly defined user group. We all acknowledge that an ill-designed experiment is irresponsible to human participants, as it ultimately wastes their valuable time. But how can we help UX researchers gather early feedback on their study design so that they can iterate it to be more responsible for the human participants?

With the advancement of Large-Language Models (LLMs), some researchers have prompted LLMs to act as autonomous agents (i.e., **LLM Agent**) that can perceive the environment and generate human-like actions to interact with the environment [37]. LLM Agents have been successfully used in various scenarios, e.g. simulating a group of 25 agents living in a village [24], simulating participants in a social science study [11, 17, 25, 29], acting as patients and clinicians in a hospital [19], and acting as employees of a software developing company [27]. The most notable effort is that Park et al. [24] illustrated a simulated town with 25 LLM Agents, who can interact with each other and with the surrounding environment; this work argues that LLM Agents can generate "believable" human-like behaviors and reasonings, where "believable" implies that the agents "appear to make decisions and act on their own volition" like a real human [4, 24]. For instance, WebAgent [10] and Claude's *Computer-Use* API[2] are two recent examples where researchers allowed the model to interact with external environments to complete tasks such as information retrieval and question answering. Although these works enable LLMs to interact with external environments (e.g., a webpage), they do not focus on the simulation of diverse user behaviors like real human users. As a result, these LLM Agents are optimized to make the final purchase using the most optimized path with the least action steps, while human users often have a twisted shopping path with lots of seemingly wasted action steps along the way.

Our project explores the possibility of utilizing LLM Agents' capabilities (i.e., simulating human behaviors and reasoning processes) to support UX researchers for web design. The usability testing guideline[1] poses the following design requirements for our system: 1) the system should be able to interact with any web environment with minimal or no customization (**easy-to-use and generalization**), 2) the system should be able to produce **both quantitative and qualitative usability data** (e.g., interviews, questionnaires, or even video recordings) so that the researcher can use their familiar analysis methods, and 3) the system should balance **group representativeness and individual uniqueness** (e.g., the female LLM Agent group and a single agent named Mary Jane, 35-year-old female, lives in Boston, etc.).

Based on these requirements, we propose UXAgent, a system that can generate LLM Agents as usability testing participants at scale and run simulated interactions with a given web environment to collect simulated user behavioral data. The UX researcher can define a demographic distribution and use the Persona Generator to generate thousands of personas for the LLM Agents, and then use the Universal Browser Connector module to interact with webpages via Chrome, which generates large-scale user behavior traces. The Universal Browser Connector module allows LLM Agents to seamlessly parse and interpret web pages while automatically executing actions on the webpage without the need for manually predefined action spaces. The system produces video recordings, as well as features a chat interface to allow the UX researchers to view the LLM Agents' interactions with the webpage and conduct qualitative interviews in natural language conversation , empowering the UX researcher to collect qualitative data. Our position is that: **LLM Agents are not to replace human participants, rather to be more responsible to the human participants — LLM Agents can work together with UX researchers (human-AI collaboration [36]) in a simulated pilot session to provide the desired early and immediate feedback for UX researchers to further iterate the design of the human-subject study before conducting it.**

To evaluate our system, we further conducted a user study with 5 UX researchers and asked them to examine the UXAgent-generated data (video recording, action trace, memory log, and the final outcome) of how 60 LLM Agents interact with a shopping website. The user study results suggested that our UX researcher participants felt that the generated human behavior data are "not like real humans" because they are "very detailed" and "real human [users] won't think like that", but they still find the data generated from our system "very helpful", which can indeed help them iterate their experiment design. We conclude our paper with discussions of how LLM Agents may alter the future of UX research.

2 Related Work

2.1 Challenges in Usability Testing

Usability testing is a core component of UX research, used to evaluate how easily users can interact with a product to achieve their

[2]https://www.anthropic.com/news/3-5-models-and-computer-use

goals [1, 2, 18, 32]. It involves observing real users as they navigate through tasks and providing valuable feedback on design effectiveness [1]. This method helps refine products by identifying usability issues, measuring user satisfaction, and improving overall user experience. The key benefits of usability testing include validating design choices, avoiding internal biases, and ensuring the product meets user expectations.

However, UX researchers conducting usability testing on web designs have been facing multiple challenges in the experiment design stage and the participant recruitment stage [9, 13, 16, 23]. To mitigate these challenges, researchers have begun to explore using agents to simulate usability testing [28]. Prior works in agent-based tools mainly focused on pre-defined environments, like GUI [7] and games [8, 33]. Nevertheless, the potential for employing agents in web design usability testing remains underexplored, largely due to the constraints of traditional web automation technologies.

2.2 LLM Agent and LLM Web Agent

In the field of HCI, recent studies have demonstrated that LLMs have the capability to simulate human-like behaviors, making research on personalized agent behavior feasible. Unlike task-oriented agents, which are typically focused on completing specific tasks, these human agents or role-play agents [4] are designed with diverse personas that reflect not only their roles and expertise but also their preferences and habits. For example, Park et al. [24] developed a simulated town with 25 LLM Agents, each with a unique persona, and exhibited believable human behaviors through their interactions. LLM Agents are also used to study users' privacy concerns [3, 39]. For example, Chen et al. [3] introduced a privacy sandbox where users can modify an agent's persona, allowing it to generate search histories and profiles based on personal traits. Taeb et al. [34] proposed AXNav that leverages LLM to convert manual accessibility test instructions to replayable and navigatable videos. These innovations guide our work, as we aim to simulate user behavior in web environments by incorporating diverse agent personas for usability testing.

LLM Agents in web environments (LLM Web Agents) like WebGPT [22] and LASER [21] have also demonstrated that LLM Agents can perform more complex tasks within web environments, such as searching, browsing and shopping [38]. With these advancements, LLM Web Agents present a promising solution for simulating human subjects in usability testing while interacting with web environments. To this end, we propose UXAgent, an LLM-Agent-based system for automating usability testing in web environments.

3 UXAgent: LLM-Agent-Based System for Usability Testing

We propose UXAgent, a system that can generate LLM Agents as usability testing participants at scale and run simulated interactions with a given web environment to collect simulated user behavioral data. The architecture of the UXAgent system is depicted in Figure 1. The system comprises several key components: The **Persona Generator Module** generates a diverse set of large-scale personas, which are provided as input to the **LLM Agent**. The **LLM Agent** interacts with a Chrome browser through the **Universal Browser Connector Module**. This module parses the raw HTML extracted from Chrome and simplifies it for the agent. The agent outputs actions, such as `click on search_button`, which the Universal Browser Connector translates into raw actions like clicking on a specific coordinate. HTML, agents' actions and memories traces, are collected for further analysis. The **Chat Interface**, which can be loaded with an agent's memory trace, is used to provide more qualitative feedback on the web design.

3.1 Persona Generator Module Design

To support large-scale simulation of diverse user backgrounds, our system enables the generation of diverse agent personas. The user can specify a distribution of agent demographics and an example persona; the persona generator module will automatically generate the desired number of personas with random demographic information matching the specified distribution. To ensure diversity in generated personas, each time we prompt the LLM to generate personas, a randomly selected persona is used as an example. A hand-crafted persona is used as the initial seed. The prompt used in our system is detailed in Section B.1.

3.2 LLM Agent Designed for Web Environment Interactions

We designed the LLM Agent that features a **Memory Stream** and a two-loop structure, inspired by Kahneman [14]. The **Fast Loop** enables real-time interaction with the web environment, while the **Slow Loop** facilitates in-depth reasoning. Each loop contains modules, which take inputs such as the agent persona, memories retrieved from the memory stream and observations of the environment and produce outputs such as memories and actions. Figure 6 illustrates the agent's structure.

The **Fast Loop** is a fast and responsive loop in which the agent quickly operates the web environment without much in-depth thinking and has a relatively low latency between actions. It consists of the following modules: 1) Perception Module: The agent makes an observation of the web environment and produces a stream of observation memories in natural language. 2) Planning Module: The agent plans the next step based on the current state of the web environment and the recent memories. 3) Action Module: The agent takes the action on the web environment based on the plan.

The **Slow Loop** of our agent is designed to provide high-level insights and strategic guidance to the Fast Loop, ensuring that the agent remains aligned with its intent and persona. It consists of the following modules: 1) Wonder Module: The agent generates random thoughts based on the current situation, mimicking a human's "mind drifting" phenomenon. 2) Reflect Module: The agent reflects on recent memories and generates high-level insights and reasoning that will be used to guide their future actions.

The **Memory Stream** is used to store and retrieve the agent's memories, including its observations, actions, reflections, and thoughts. Each memory piece consists of a sentence or paragraph in natural language combined with a timestamp. The Memory Stream also serves as a bridge between the Fast and Slow loops. Following prior work [24], we design our memory retrieval module to focus on importance, relevance, and recency. Each memory is scored based on these factors, with weights tailored to prioritize recency in the

Fast Loop for quick responses and relevance in the Slow Loop for deeper reasoning, aligning with human cognitive patterns.

3.3 Universal Browser Connector Module

To seek the balance between the complexity of the observation space and the flexibility of the action space, to allow our LLM Agent to operate web browsers automatically, we implemented a Universal Browser Connector that **uses manually crafted policies to parse the raw HTML to a simplified version of HTML as the model's observation space and use auto-generated action space that can be easily generalized to different websites**. The simplified HTML is a tree structure that contains the visible information of the web page, such as the title, the product list, the product detail, and the shopping cart, while trimming all unnecessary information, such as the CSS and JavaScript code. Action space is a set of task-agnostic actions generated by the parser that the agent can take, such as "click", "type" and "back", that can be easily generalized to different websites. The agent can take actions in the action space to interact with the web environment and observe the web environment by receiving the simplified HTML structure. The definition of the action space, observation space, and the HTML parser can be found in appendix A.

3.4 Chat Interface

After the simulation session, the simulated action traces and memory traces will be loaded into the Chat Interface, allowing the UX researcher to interview the agent and collect qualitative feedback. Users can first select a persona from the persona selection page and the memory of the persona will be loaded to the chat interface. Then, in the chat page shown in Figure 2b, the user can talk and interview the agent to collect qualitative feedback. We also allow uploading images during the chat, allowing the user to get feedback on a new feature prototype, even before it is implemented.

3.5 Simulation Environment and Task for LLM Agents

To validate the effectiveness of UXAgent, we deployed the LLM Agents on two online shopping platforms: Amazon.com and Google Flights. Amazon is a publicly available online shopping environment and is used as the target experiment. We chose Amazon as as our experiment platform because it is widely used by various research projects [12]. Google Flights, a commercial platform for flight booking, is also used to demonstrate the adaptability of our system from the online shopping use case to the travel booking use case. As shopping task represents a common, highly personal activity that humans engage in daily and allows the agent to exhibit diverse behaviors, we selected the shopping task ("buy a jacket") as the agents' initial intent in our study. From the initial intent, our system is able to generate simulation results, including the agents' memory and action traces, video recording, and final outcome (such as item purchased or session quit).

In our study, we used the Persona Generator Module to generate 60 agent personas, following a uniform distribution across different gender groups (male, female, and non-binary) and across different income groups ($0-$30k, $30k-$58k, $58k-$94k, $94k-$153k, and $153k-). We then ran one simulation session with each of these 60 agent personas using UXAgent on Amazon. The simulation results revealed different purchase behaviors across income groups, with the average purchased amount increasing with income: $28.41 for the $0-$30k group, $15.99 for $30k-$58k, $54.85 for $58k-$94k, $41.03 for $94k-$153k, and $75.34 for $153k+. We provided the simulation result for the UX researcher participants to analyze as if they were running these user study sessions.

4 User Study

4.1 Participant Recruitment and Participant Task

We recruited five UX researchers as participants, each with self-reported UX experience ranging from 1 to 6 years (M = 3, SD = 1.87). Most participants rated themselves as "very familiar" with usability analysis on a 5-point scale (M = 4.2, SD = 0.83). All participants have prior experience using Large Language Models (e.g., ChatGPT, Claude). The study was conducted remotely, with participants using their computers to access the LLM Agent and communicating with the moderator via Zoom.

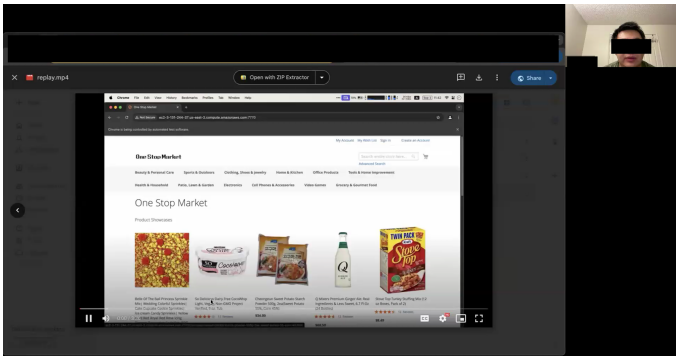
Before the study, the moderator used UXAgent to generate various personas and asked each agent to finish a simulation session of usability testing on the shopping task. The moderator then randomly sampled 60 simulation sessions from the 1,000 simulations. Examples of the generated data are shown in appendix C.

In our study, participants are instructed to first finish a pre-study survey and then receive a brief introduction to the study tasks. In the 40-minute study, participants were tasked with conducting a user study to measure customer shopping behavior on a website. Specifically, they were required to analyze simulation data generated by UXAgent, with the goal of improving the design of the user study. Participants were instructed to freely navigate through various types of data while thinking aloud. Participants are then invited to interact with a chatbot, pre-loaded with a simulated participant's memory, to "interview" the simulated participant. Participants were provided with three different types of data generated by UXAgent, including 1) agent's action trace, 2) memory trace, and 3) final outcome. Any questions regarding the tasks or the data from the participants were addressed before proceeding.

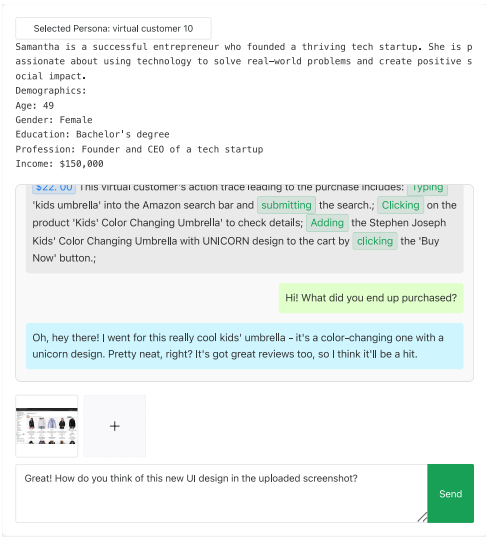
After completing the data analysis, participants filled out a Likert scale survey to assess the system's efficiency, trustworthiness, satisfaction, and helpfulness, following methodologies from prior work on LLM-generated data [15]. Participants were also asked to answer "For each type of data generated, is it helpful for designing your study?" We then conducted a semi-structured post-study interview, asking participants to share their perceptions of the different data types and how the simulated results could provide insights on the design of future user studies. We also gathered suggestions for system improvements. A screenshot of our user study can be found in Figure 2a. Each session lasted approximately 40 minutes, was video-recorded, and the transcriptions were analyzed using an inductive process [35].

4.2 Study Findings

4.2.1 How Did UX Researchers Use Our UXAgent System? Participants followed four categories of analysis paradigm when analyzing data generated by UXAgent: 1) gaining trust in the data, 2) making



(a) A participant in our user study is watching a video recording replaying an LLM Agent's onling shoping action trace.



(b) Chat UI. The UX researcher can select a LLM Agent to ask why it had certain behaviors.

Figure 2: User study session and an interactive chat interface with LLM Agent

sense of the data, 3) proposing hypotheses and detailed observations, and 4) drawing conclusions.

Gaining trust of data. When participants had access to all types of data generated by UXAgent, their predominant response was to examine each data type to ensure its trustworthiness. Most participants only proceeded with deeper analysis after confirming the data's trustworthiness. However, some participants began to analyze the data early on without fully verifying their trustworthiness and encountered challenges at later stages.

Making sense of data. Once trust was established, they moved on to the sense-making phase, where they tried to interpret the data and understand the relationships between different variables. For example, participants calculated the average price and number of actions among different groups in a google sheet. P4 was misled by the persona names presented to them and was trying to analyze the different shopping behaviors of LLM Agents with different names that are randomly generated: *"Most of the Jakes? There's only one Jake here that didn't buy it, but most Jakes here bought it."* This suggests a better understanding of how the data is generated is necessary for future adoption of such data.

Proposing hypotheses and detailed observations. Participants then made hypotheses on the experiment result and made detailed observations to verify their hypothesis. Most participants noticed agents of non-binary genders have a lower purchase rate than male and female agents. P4 hypothesized that maybe this is due to product availability, as there are not many unisex products. They then checked the raw action log and noticed that a non-binary user searched for a 'unisex jacket,' clicked into a 'women's jacket,' and ended up purchasing nothing, which proved their hypothesis.

Drawing Conclusions. Participants then discussed what they found during the analysis of the simulated result. They organized the findings in a document and discussed how they were going to revise and iterate their study design after the simulation. P4 suggested that they will move quickly, iterate the study design, and re-run the simulation study to test the new design: *"Since it's really quick and easy, I think it could encourage people to do a lot more iterative design."*

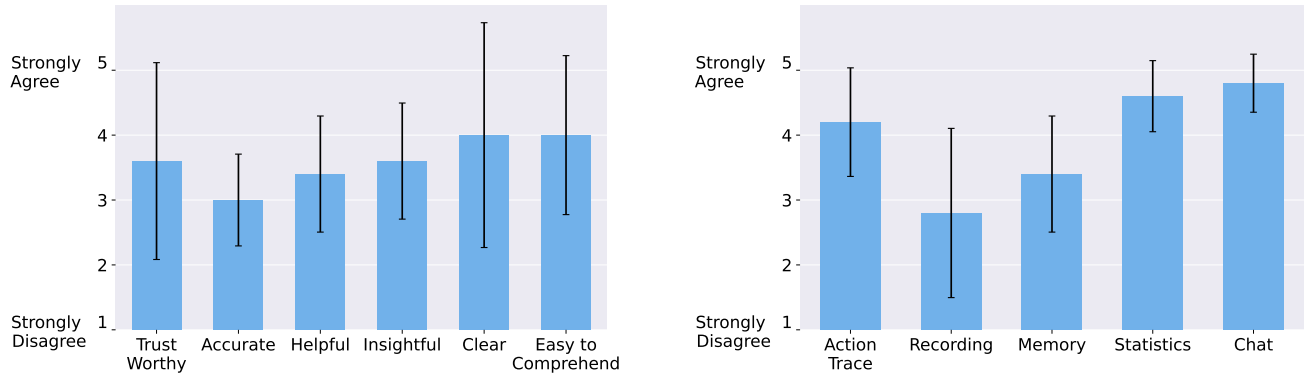
4.2.2 *Participants' Perceptions and Concerns of UXAgent.*

Participant have mixed feelings of simulated data from UXAgent. Figure 3 shows participants' perceived usage of LLM Agents generated human behavior data. Notably, although the generated data may not seem very realistic (M=3, SD=0.7), participants still found the data presented by UXAgent to be very helpful (M=3.4, SD=0.89). This perceived helpfulness was also seen as an expansion of the pilot user study design, with P1 mentioning:

> *" It's very hard to find pilot study of 60 participants or even just five or ten. So I think in this way I can conduct my pilot study with the large language model agent. "* (P1)

Participants believed that the simulation results were insightful (M=3.6, SD=0.89) and trustworthy (M=3.6, SD=1.51), sparking an interest in further investigation—an uncommon occurrence in smaller-scale pilot studies. Furthermore, the large-scale data allows them to see things from others' perspectives. Participants especially liked the ability to talk with the agent (M=4.8, SD=0.44) and collect information that you may never be able to get from real human participants, as P4 mentioned: *"I think it could help me to kind of brainstorm."* The final outcome and statistics (M=4.6, SD=0.54) and the agents' action trace (M=4.2, SD=0.84) were also rated as helpful by the participants.

Participants mentioned that the raw memory traces are hard to read and analyze. P2 mentioned that *"If you can present the thoughts*



(a) Participant's Perception towards the Usage of UXAgent in User Study

(b) Participant's Perceived Helpfulness Towards the Different Data Modality Generated by UXAgent

Figure 3: Participant's perception towards using UXAgent in User Study

in a better way, like not just right now for each agent, probably like a highlight or summary." We believe this is due to the noisy nature of all unprocessed raw data, and we further discuss the possible use of this data in Section 5.1. Participants gained trust in the generated data from watching the generated video recording of the browser operated by the LLM Agent, but they found analyzing it hard and time-consuming.

Concerns about model bias and data bias. Participants are concerned about the potential biases introduced by the data generated by LLM Agents. In our study, we identified two types of bias that are key concerns for participants when it comes to LLMs. 1) **Data Representation Bias** is where the decisions made by the agents may be influenced by biases present in the real-world data they were trained on, as stated by P2: *"because we do feel like the LLM has sort of stereotyping that. My feeling female will buy more male will buy less things"* . 2) **Algorithmic Decision Bias**, which is introduced in the pre-training and fine-tuning stage of the LLM could distract the study design, as P1 commented, *"it could be bias or it could disturb my study design"*(P1) .

Perceived unrealistic human behavior. Participants have varied opinions about whether certain types of the LLM's simulated data are realistic. Human users usually quickly skim and disregard unimportant webpage elements like ads or irrelevant results. Our agent mirrors this behavior by assigning varying "importance" scores to each memory in its memory stream. However, as the importance score is not shown to our participants, participants reported that the memory stream is "of too much detail" and "not realistic". Despite the fact that the presented memory stream may not be a full mimic of real human behavior, participants still showed their love in having these data, as these are still something they cannot collect from experiments with real human participants and can provide additional insights.

5 Discussion

5.1 Design Considerations for LLM Agent Systems in User Study

The shift towards using LLM to assist UX research has demonstrated immense potential. Our work UXAgent demonstrated that LLM Agents can support user studies with faster design iterations through user behavior simulation, reducing costs and improving the quality of study design [6]. Our participants suggested that the **future systems should automatically generate high-level insight summaries** as the generated LLM Agents' raw memory is hard to analyze. Thus, future systems should explore how to automatically utilize the current data to generate high-level insights and summaries for users.

5.2 Resolving Risk and Privacy Conerns

While LLM-assisted UX study systems offer substantial benefits for user studies, they also introduce significant risks and concerns, particularly around privacy and ethical use [20, 31]. A notable concern about sensitive data might arise in applying LLM Agents in user studies in fields such as healthcare [26]. Our participants viewed data from LLM Agents as supplementary rather than definitive, which is consistent with prior findings that LLMs cannot fully replace human subjects in psychological experiments [5]. This preference reflects their ability to detect biases, limiting acceptance of LLMs as accurate representations of outcomes [30].

Another critical issue is the risk of researchers misusing AI-generated data [26]. Some may be tempted to use this simulated data as a substitute for actual human participant data, potentially bypassing real user studies altogether. This could lead to flawed conclusions, as the AI-generated data, while useful for simulations, cannot fully replicate the complexity of real human behavior. We believe that addressing these risks requires a clear understanding of the limitations of LLM-generated data and establishing rigorous ethical guidelines and robust privacy safeguards to ensure responsible and effective integration of LLM in UX research.

5.3 Limitations and Future Works

There are several limitations in our work. First, the limited number of participants in our heuristic evaluation may not fully capture the diversity of user experiences or the generalizability of the findings. A larger scale of rigorous user testing in the future will help us further explore the effectiveness of UXAgent. Also, our current work focuses only on the qualitative analysis of the participants' perspectives toward LLM Agent simulation, but not on quantitative evaluations of the LLM Agents' versus real humans' shopping behaviors. It is crucial to conduct a systematic analysis comparing LLM-generated simulations with real human participants in order to further understand how UX researchers could benefit from using such agents. Furthermore, our current system and agent design only process semantic or textual information (i.e. HTML) available on the webpage, while visual elements such as images and visual layout information are excluded during parsing. Incorporating Multimodal LLMs (MLLMs) to interpret and utilize both textual and visual information could enable the system to better understand page context, enhancing its adaptability in real-world scenarios where visual cues and layout are also critical for effective user interaction modeling.

Our current results confirm the effectiveness of UXAgent in web-related user research, as well as the potential of LLMs in simulating user behavior to assist rapid iteration of user research. In the future, this work could be expanded to encompass a broader range of user research contexts beyond web applications. This might include adaptation to UI/UX design in desktop, mobile, or mixed-reality environments, where LLM Agents could simulate user interactions in complex, multi-modal interfaces. Also, we only tested giving the agents an explicit intention of "buy a jacket". Experimenting with other intents, such as window shopping, can also help understand the ability and limits of LLM Agent. Additionally, the action space of the agent is currently limited to basic actions such as "click" and "type". Supporting other types of actions, such as "read", "scroll" and "mouse hover" can allow the experimenter to collect more information useful to their study.

6 Conclusion

In this work, we designed UXAgent, a system enabling researchers to conduct simulated user studies, thereby facilitating iterative refinement of their UX study designs. The results of our evaluation study suggest UXAgent can support UX researchers in designing and refining their user studies to improve UX study quality and reduce risk for real human subjects. Our study is the first step towards a promising future of designing LLM agents to collaborate with UX researchers to achieve "human-AI collaboration" in the field of UX research.

References

[1] Carol M Barnum. 2020. **Usability Testing Essentials: Ready, Set... Test!** Morgan Kaufmann.

[2] J.M. Christian Bastien. 2010. Usability Testing: A Review of Some Methodological and Technical Aspects of the Method. **International Journal of Medical Informatics** 79, 4 (April 2010), e18–e23. https://doi.org/10.1016/j.ijmedinf.2008.12.004

[3] Chaoran Chen, Weijun Li, Wenxin Song, Yanfang Ye, Yaxing Yao, and Toby Jia-Jun Li. 2024. An Empathy-Based Sandbox Approach to Bridge the Privacy Gap among Attitudes, Goals, Knowledge, and Behaviors. In **Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24)**. Association for Computing Machinery, New York, NY, USA, 1–28. https://doi.org/10.1145/3613904.3642363

[4] Chaoran Chen, Bingsheng Yao, Ruishi Zou, Wenyue Hua, Weimin Lyu, Toby Jia-Jun Li, and Dakuo Wang. 2025. Towards a Design Guideline for RPA Evaluation: A Survey of Large Language Model-Based Role-Playing Agents. **arXiv preprint arXiv:2502.13012** (2025).

[5] Ziyan Cui, Ning Li, and Huaikang Zhou. 2024. Can AI Replace Human Subjects? A Large-Scale Replication of Psychological Experiments with LLMs. **arXiv preprint arXiv:2409.00128** (2024).

[6] Heidi Decker-Maurer. 2012. Method Madness: A Usability Testing Pilot Research Case Study. **Commun. Des. Q. Rev** 13, 1 (March 2012), 14–18. https://doi.org/10.1145/2424837.2424839

[7] Juha Eskonen, Julen Kahles, and Joel Reijonen. 2020. Automating GUI Testing with Image-Based Deep Reinforcement Learning. In **2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)**. 160–167. https://doi.org/10.1109/ACSOS49614.2020.00038

[8] Pedro M. Fernandes, Manuel Lopes, and Rui Prada. 2021. Agents for Automated User Experience Testing. In **2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)**. 247–253. https://doi.org/10.1109/ICSTW52544.2021.00049

[9] Asbjørn Følstad, Effie Law, and Kasper Hornbæk. 2012. Analysis in Practical Usability Evaluation: A Survey Study. In **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)**. Association for Computing Machinery, New York, NY, USA, 2127–2136. https://doi.org/10.1145/2207676.2208365

[10] Izzeddin Gur, Hiroki Furuta, Austin V. Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2023. A Real-World WebAgent with Planning, Long Context Understanding, and Program Synthesis. In **The Twelfth International Conference on Learning Representations**. https://openreview.net/forum?id=9JQtrumvg8

[11] Onder Gurcan. 2024. LLM-Augmented Agent-Based Modelling for Social Simulations: Challenges and Opportunities. https://doi.org/10.48550/arXiv.2405.06700 arXiv:2405.06700 [physics]

[12] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. WebVoyager: Building an End-to-End Web Agent with Large Multimodal Models. arXiv:2401.13919 [cs] http://arxiv.org/abs/2401.13919

[13] Morten Hertzum and Niels Ebbe Jacobsen. 2003. The Evaluator Effect: A Chilling Fact About Usability Evaluation Methods. **International Journal of Human-Computer Interaction** (Feb. 2003). https://doi.org/10.1207/S15327590IJHC1501_14

[14] Daniel Kahneman. 2011. Thinking, fast and slow. **Farrar, Straus and Giroux** (2011).

[15] Emily Kuang, Ehsan Jahangirzadeh Soure, Mingming Fan, Jian Zhao, and Kristen Shinohara. 2023. Collaboration with Conversational AI Assistants for UX Evaluation: Questions and How to Ask Them (Voice vs. Text). In **Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)**. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3544548.3581247

[16] Emily Kuang, Xiaofu Jin, and Mingming Fan. 2022. "Merging Results Is No Easy Task": An International Survey Study of Collaborative Data Analysis Practices Among UX Practitioners. In **Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)**. Association for Computing Machinery, New York, NY, USA, 1–16. https://doi.org/10.1145/3491102.3517647

[17] Kyuwon Lee, Simone Paci, Jeongmin Park, Hye Young You, and Sylvan Zheng. [n. d.]. Applications of GPT in Political Science Research. ([n. d.]).

[18] James R Lewis. 2012. Usability Testing. **Handbook of human factors and ergonomics** (2012), 1267–1312. https://doi.org/10.1002/9781118131350.ch46

[19] Junkai Li, Siyu Wang, Meng Zhang, Weitao Li, Yunghwei Lai, Xinhui Kang, Weizhi Ma, and Yang Liu. 2024. Agent Hospital: A Simulacrum of Hospital with Evolvable Medical Agents. https://doi.org/10.48550/arXiv.2405.02957 arXiv:2405.02957 [cs]

[20] Tianshi Li, Sauvik Das, Hao-Ping Lee, Dakuo Wang, Bingsheng Yao, and Zhiping Zhang. 2024. Human-Centered Privacy Research in the Age of Large Language Models. https://doi.org/10.48550/arXiv.2402.01994 arXiv:2402.01994 [cs]

[21] Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, Wenhao Yu, and Dong Yu. 2024. LASER: LLM Agent with State-Space Exploration for Web Navigation. https://doi.org/10.48550/arXiv.2309.08172 arXiv:2309.08172 [cs]

[22] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. WebGPT: Browser-assisted Question-Answering with Human Feedback. https://doi.org/10.48550/arXiv.2112.09332 arXiv:2112.09332 [cs]

[23] Mie Nørgaard and Kasper Hornbæk. 2006. What Do Usability Evaluators Do in Practice? An Explorative Study of Think-Aloud Testing. In **Proceedings of the 6th Conference on Designing Interactive Systems (DIS '06)**. Association for Computing Machinery, New York, NY, USA, 209–218. https://doi.org/10.1145/1142405.1142439

[24] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In **Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)**. Association for Computing Machinery, New York, NY, USA, 1–22. https://doi.org/10.1145/3586183.3606763

[25] Joon Sung Park, Carolyn Q. Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S. Bernstein. 2024. Generative Agent Simulations of 1,000 People. arXiv:2411.10109 http://arxiv.org/abs/2411.10109

[26] Charith Peris, Christophe Dupuy, Jimit Majmudar, Rahil Parikh, Sami Smaili, Richard Zemel, and Rahul Gupta. 2023. Privacy in the Time of Language Models. In **Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23)**. Association for Computing Machinery, New York, NY, USA, 1291–1292. https://doi.org/10.1145/3539597.3575792

[27] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. ChatDev: Communicative Agents for Software Development. https://doi.org/10.48550/arXiv.2307.07924 arXiv:2307.07924 [cs]

[28] Yuqing Ren and Robert E Kraut. 2014. Agent based modeling to inform the design of multiuser systems. In **Ways of Knowing in HCI**. Springer, 395–419.

[29] Samuel Schmidgall, Rojin Ziaei, Carl Harris, Eduardo Reis, Jeffrey Jopling, and Michael Moor. 2024. AgentClinic: A Multimodal Agent Benchmark to Evaluate AI in Simulated Clinical Environments. arXiv:2405.07960 [cs] http://arxiv.org/abs/2405.07960

[30] Albrecht Schmidt, Passant Elagroudy, Fiona Draxler, Frauke Kreuter, and Robin Welsch. 2024. Simulating the Human in HCD with ChatGPT: Redesigning Interaction Design with AI. **interactions** 31, 1 (Jan. 2024), 24–31. https://doi.org/10.1145/3637436

[31] Yijia Shao, Tianshi Li, Weiyan Shi, Yanchen Liu, and Diyi Yang. 2024. PrivacyLens: Evaluating Privacy Norm Awareness of Language Models in Action. https://doi.org/10.48550/arXiv.2409.00138 arXiv:2409.00138 [cs]

[32] Debora Shaw. 1996. Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests. **Journal of the American Society for Information Science** 47, 3 (March 1996), 258–259. https://doi.org/10.1002/(SICI)1097-4571(199603)47:3{$<$}258::AID-ASI18{$>$}3.0.CO;2-#

[33] Samantha . Stahlke, Atiya Nova, and Pejman Mirza-Babaei. 2019. Artificial Playfulness: A Tool for Automated Agent-Based Playtesting. In **Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)**. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3290607.3313039

[34] Maryam Taeb, Amanda Swearngin, Eldon Schoop, Ruijia Cheng, Yue Jiang, and Jeffrey Nichols. 2024. AXNav: Replaying Accessibility Tests from Natural Language. In **Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)**. Association for Computing Machinery, New York, NY, USA, 1–16. https://doi.org/10.1145/3613904.3642777

[35] David R. Thomas. 2006. A General Inductive Approach for Analyzing Qualitative Evaluation Data. **American Journal of Evaluation** 27, 2 (2006), 237–246. https://doi.org/10.1177/1098214005283748 arXiv:https://doi.org/10.1177/1098214005283748

[36] Dakuo Wang, Justin D. Weisz, Michael Muller, Parikshit Ram, Werner Geyer, Casey Dugan, Yla Tausczik, Horst Samulowitz, and Alexander Gray. 2019. Human-AI Collaboration in Data Science: Exploring Data Scientists' Perceptions of Automated AI. **Proceedings of the ACM on Human-Computer Interaction** 3, CSCW (Nov. 2019), 1–24. https://doi.org/10.1145/3359313 arXiv:1909.02309 [cs]

[37] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. 2024. A Survey on Large Language Model Based Autonomous Agents. **Frontiers of Computer Science** 18, 6 (Dec. 2024), 186345. https://doi.org/10.1007/s11704-024-40231-1 arXiv:2308.11432 [cs]

[38] Shunyu Yao, Howard Chen, John Yang, and Karthik R. Narasimhan. 2022. WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents. In **Advances in Neural Information Processing Systems**. https://openreview.net/forum?id=R9KnuFlvnU

[39] Zhiping Zhang, Bingcan Guo, and Tianshi Li. 2025. Privacy Leakage Overshadowed by Views of AI: A Study on Human Oversight of Privacy in Language Model Agent. https://doi.org/10.48550/arXiv.2411.01344 arXiv:2411.01344 [cs]

A Browser Environment

Observation Space. The observation space in the Universal Browser Connector is represented with a JSON object that contains:

- `url`: The URL of the current webpage
- `page`: The simplified HTML of the current webpage
- `clickables`: A list of clickable elements on the webpage, such as buttons and links
- `inputs`: A list of input elements on the webpage, such as text fields and dropdowns
- `error_message`: An error message if an error occurs, for example, if the agent clicked on a button that does not exist

To remove redundant information, we only keep the visible information of each HTML tag, such as the text content, the attributes, and child elements. An example simplified HTML is shown in Figure 4. The parser takes the raw HTML and a recipe and parses the HTML based on the recipe. The recipe specifies how to parse the HTML, such as which tags to keep and how to extract the text content and attributes of each tag. If the recipe has any child elements, the parser will recursively parse the child elements with the child recipe.

Handling of Text Text of an HTML element is extracted if the recipe specifies a `add_text` field. The three possible parse methods of text are `default`, `text_selector` and `text_js`. `default` extracts all of the text content of the tag. `text_selector` extracts the text content of a child element based on a CSS selector. `text_js` serves like a fallback method, where the recipe species a JavaScript function that returns the text content. In this way, for most cases the recipe can be easily created by selecting the text element. In rare cases, for example, the text of a radio button should be the text of a sibling `label` tag, the flexibility of the `text_js` method allows the parser to handle these cases.

Sometimes, the visual information and the textual information of an element differ, especially in cases where the HTML's accessibility features are not well-implemented. For example, on the product detail page, the review score information is often shown as stars. While an alternative text like "4.2 out of 5 stars" exists, there is sometimes no clue that the scores here is the product review. We allow the recipe to specify a `text_format`, for example, "`Rating: {}`" in this case, to supply the necessary context. With `text_format` like this, the abovementioned product review score will be parsed as "Rating: 4.2 out of 5 stars", allowing easier understanding by the LLM Agent.

Handling of Tag Name and Attributes Attributes of HTML elements include 1) the visual information of the element, such as style and class; 2) the semantic information of the element, such as role, href, and src; 3) the information of the element's state, such as checked and selected. The tag name of an HTML element also sometimes conveys semantic information, such as `a` tag is used for hyperlink, and `img` tag is used for the image.

In our case, as most visual information is not needed, we remove all visual information attributes from the HTML and keep the semantic information and the state information. For tag name, the default behavior is to keep the tag name as is, and the recipe can supply a `tag_name` field to override the tag name, for example, replace `div` with `button` for better readability and accessibility.

Specifically, the `class` attribute mostly conveys visual information such as color and alignment. However, they sometimes also convey the semantic information, such as `product-item-details`. Thus, the recipe can supply a `keep_attr` option to keep the attributes of the tag. To further enhance flexibility, the recipe can supply a `override_attr` option to keep the attributes of the tag based on a JavaScript function. This way, the parser can handle a wide range of scenarios, from simple attribute extraction to more complex cases where attributes need to be dynamically determined.

The JavaScript state of the HTML element, for example, the `selected` attribute and the `value` of an input box, is invisible in the HTML, yet important for the agent to know. Thus, for `select` and `input` elements, we extract their JavaScript state, and add it to the corresponding attribute simplified HTML, for example, a `selected` attribute is added to the `select` element, and a `value` attribute is added to the `input` element.

Handling of Child Elements

To develop complex animation to indicate states like hover and focus, UI designers use many nested elements and CSS classes. For example, a button is often implemented using the `button` or `div` tag, but to support styles like shadow and animations, web designer often use multiple nested `div` element inside the button. While these redundant tags are necessary for better styling, they do not contribute to the semantic information to the button and may affect the agent's understanding.

To simplify the HTML, we "flatten" it by removing redundant levels of elements. For instance, in the HTML shown in Figure 5a, a nested "Child-wrapper" is used. If the recipe specifies a `children` field, the parser will select the deep child element ("Child" in this case) and remove the intermediate "Child-wrapper".

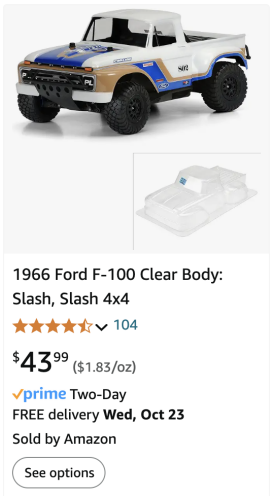
If the recipe does not specify a `children` field, all child elements are removed from the HTML. Additionally, as discussed in Section A, if the recipe specifies an `add_text` field, the text from the deep child elements is extracted and added to the corresponding parent tag, which is shown in Figure 7. This process reduces the number of tags in the HTML, making it more readable.

In this way, the HTML presented to the LLM Agent is simplified and more readable, while still preserving the same semantic information of the HTML.

Action Space. The action space of the web browser includes the following actions:

- `click`: perform a mouse click on the target element
- `type`: type the text into the input box
- `type_and_submit`: type the text into the input box and submit the form, just like hitting`Enter` key after typing the text
- `clear`: clear the text in the input box
- `back`: go back to the previous page
- `terminate`: terminate the current session, denoting user closes the browser

For all interactive elements, a `name` attribute is appended to the HTML tag to assist the agent in identifying the target element. The `name` attribute is constructed by traversing the parent tree. For instance, if a button is named `add_to_cart` in the recipe, and it is a child of a `div` named `product`, the complete name of the resulting HTML element will be `product.add_to_cart`. Specifically, in our testing online shopping scenario, all the forms only have



(a) Example of a part of a web page processed by the Perception Module

```
<div name="search_results.1966_ford_f_100_clear_body_slash_slash_4x4"
class="search-result">
    <a name="search_results.1966_ford_f_100_clear_body_slash_slash_4x4.view_product"
       class="product-name">
        1966 Ford F-100 Clear Body: Slash, Slash 4x4
    </a>
    <div class="product-review">
        <span class="product-rating">4.3 out of 5 stars</span>
        <span class="product-rating-count">103 reviews</span>
    </div>
    <div class="product-price"><span>$43.99</span></div>
    <div class="product-delivery">FREE delivery Mon, Oct 14</div>
</div>
```

(b) The simplified HTML structure parsed by the browser environment

```
The search results display a product listing for the '1966 Ford F-100 Clear Body:
 Slash, Slash 4x4'. This product has a rating of 4.3 out of 5 stars based on
 103 reviews and is priced at \$43.99 with free delivery on Monday, October 14.
```

(c) The observation memory produced by the Perception Module

Figure 4: Example of a part of a web page processed by the Univerasl Web Connector and the Perception Module. The Univerasl Web Connector parses the raw web page (left) into a simplified HTML structure (right), and the Perception Module generates a descriptive observation memory (bottom) for each segment.

one input (e.g. search box), thus to reduce the number of action needed and to reduce hullucination, we also provide a composite action `type_and_submit`, which is equivlent to typing and hitting the "enter" key. Agents can also `type` then `click` on the submit button.

B Prompt

B.1 Prompt for Persona Generatoin

```
Generate a persona using the above examples. The
persona should be different from previous personas to
ensure diversity. The persona should:

- Have the age of {age}
- Be {gender}
- Have an income between ${income_range[0]} and
${income_range[1]}

Provide the persona in the same format as the
examples.

Only output the persona, no other text.
```

C Example Data

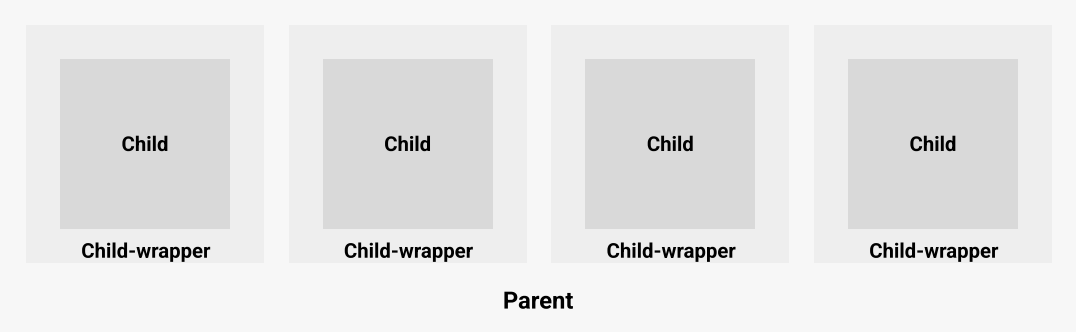
C.1 Action Trace

```
Action 1: type_and_submit, description: Typing
'woman's jacket' into the search input field and
submitting the form.
Action 2: click, description: Clicking on the product
'Jackets For Women Womens Hooded Fleece Line Coats
Parkas Faux Fur Jackets with Pockets' to view its
details.
Action 3: click, description: Clicking on the 'Navy'
color option for the jacket.
Action 4: click, description: Clicking on the
'Medium' size option for the jacket.
Action 5: click, description: Clicking on the 'Add to
Cart' button to add the chosen product to the cart.
```

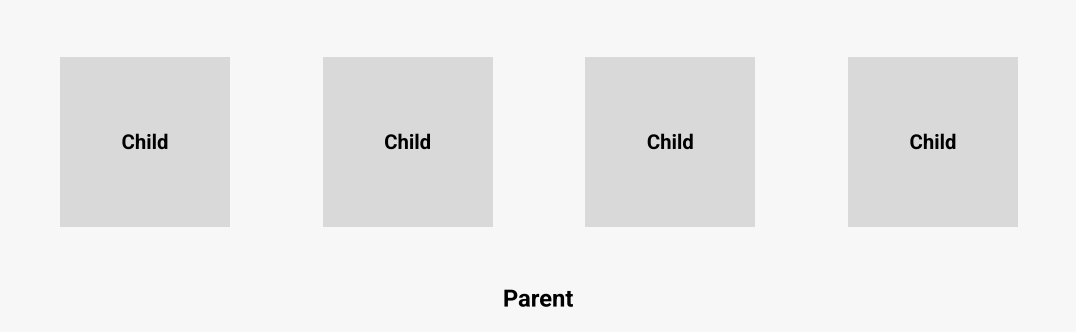
C.2 Memory Trace

This example is filtered for readability.

```
For action 1, I will: Typing 'woman's jacket' into the
search input field and submitting the form.
Before action 2, I thought: I really need to find that
navy blue jacket soon. It would be perfect for those
networking coffee meet-ups. I hope I can get it on
sale.
```



(a) HTML structure before parsing



(b) HTML structure after parsing

Figure 5: HTML Parser Example of multiple nested elements.

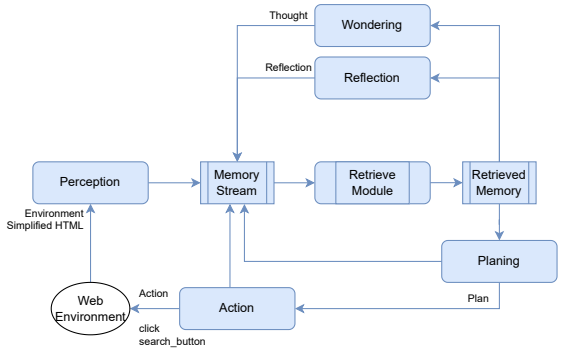
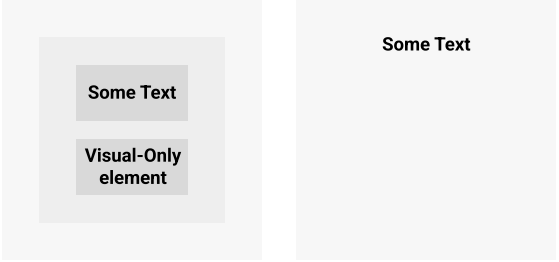


Figure 6: The structure of the LLM Agent's Reasoning Architecture



(a) HTML structure before parsing (b) HTML structure after parsing

Figure 7: HTML parser example of text in child elements.

```
Before action 2, I thought: If I don't find a job
soon, I might have to reconsider my budget for
essentials. It feels so overwhelming at times.
Before action 2, I thought: I wonder if I should try
making some more connections through social media or
if that's just a waste of time. It feels like
everything is so competitive right now.
```

```
Before action 2, I thought: The action to search for
'woman's jacket' was successful, as the new
observation confirms that the search results for this
query have been displayed.
Before action 2, I thought: Next, I need to browse
through the search results to find a navy blue,
medium-sized women's jacket. It will be essential to
closely examine the available product options to see
if they meet these specific criteria.
```

```
Before action 2, I thought: Additionally, I should pay
attention to the pricing and consider narrowing down
my search to ensure that I am selecting the most
affordable option that fits my needs and budget.
Before action 2, I saw: The page title reads 'Search
results for: 'woman's jacket''. This provides context
for the user about the ongoing search query.
```