VULCAN: WHERE AGENTS LEARN BY LIVING IN SIMULATED TOOL ENVIRONMENTS

Amir Saeidi* **Chitta Baral**
Arizona State University
{ssaeidi1,cbaral}@asu.edu

Ahmed Awadallah **Harkirat Behl**
Microsoft Research
{ahmed.awadallah,hbehl}@microsoft.com

ABSTRACT

Large Language Model (LLM) agents perform well in narrow tool-use settings but struggle to generalize across diverse environments due to the lack of high-quality training data. Existing data collection methods rely on manual environment setup and access to live systems, making them labor-intensive and difficult to scale. To address this, we propose **VULCAN**, a three-phase framework that automatically constructs executable and deterministic tool-use environments directly from tool schemas, generates diverse task variants, and collects high-fidelity agent trajectories by running LLM agents in these simulated environments. Using VULCAN, we simulate 14 environments and generate 78K high-quality training examples from only 232 tools. We conduct extensive experiments across 4 benchmarks and 2 model families, for a total of 5 models, in which we fine-tune Qwen2-3 and Phi-series models on synthesized data. Our approach yields consistent improvements of **12.2%**, **10.5%**, and **15.8%** across evaluation settings over the base versions of these models. The resulting models outperform strong baselines such as `GPT-5.2` and approach the performance of `Kimi-K2`, a significantly larger frontier model. These results demonstrate that VULCAN provides a scalable, domain-agnostic approach for improving the reasoning and generalization capabilities of tool-using LLM agents.

1 INTRODUCTION

The rapid advancement of large language models (LLMs) has enabled a new generation of agentic systems that go beyond conversational interaction to perform real-world actions through tools and external environments (Li et al., 2024; Ferrag et al., 2025; Bousetouane, 2025). These agents are now expected to manage complex workflows, interact with software systems, and operate under dynamic and stateful conditions (Su et al., 2025). Unlike traditional dialogue systems, such agents are required not only to generate fluent language but also to coordinate sequences of actions, maintain internal consistency, and adhere to domain-specific constraints (Prabhakar et al., 2025). Consequently, tool-using agents is now the central paradigm for deploying AI in real-world applications across diverse domains.

However, the practical development of such agents faces a critical gap due to the absence of accessible, executable training environments that faithfully capture real-world dynamics. This challenge is further compounded by a lack of scalable approaches for generating realistic agentic tasks. In many real-world scenarios, direct interaction with operational environments is strictly limited by cost, privacy, or safety constraints. This inaccessibility makes it nearly impossible to generate the high-quality interaction data required for training, effectively creating a barrier to building capable agents for these specific environments (Zou et al., 2025; Pan et al., 2025). Furthermore, while realistic agent behavior requires exposure to diverse, multi-step tasks involving tool coordination and evolving states, creating such task distributions at scale is labor-intensive and difficult to generalize (Zhou et al., 2023; Wang et al., 2025).

Existing works have attempted to address these challenges in isolation, yet they exhibit significant limitations. A small subset of approaches aims to simulate the environment itself (Fang et al.,

*Work done during internship at Microsoft.

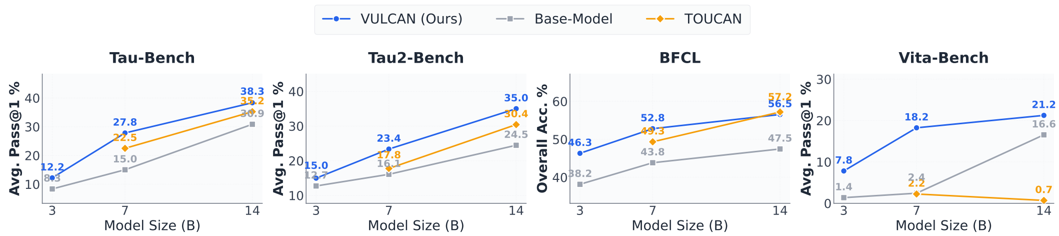

Figure 1: Comparison of VULCAN-tuned models with Qwen2.5 instruction baselines and TOUCAN models across various benchmarks. Refer to Appendix C.1 for more details.

2025; Li et al., 2025), but these methods often suffer from reliability issues and typically rely on few-shot examples or predefined use cases, making them unsuitable when no prior interaction data is available. In contrast, most other approaches focus solely on task or user-scenario generation, explicitly assuming that a reliable and executable environment is already provided (Yin et al., 2025; Prabhakar et al., 2025; Chen et al., 2025). In these settings, user behavior is commonly simulated by an LLM, introducing hallucination risks and weak grounding, often restricting interactions to simplified, single-turn tasks (Laban et al., 2025; Mishra et al., 2025; He et al., 2025a). Concurrent works (Song et al., 2026; Wang et al., 2026) rely heavily on MCP-based infrastructure, which introduces latency overhead and poses challenges for scalable deployment.

To bridge this gap, we propose **VULCAN**, a three-stage framework leveraging multi-agent collaboration to automate agentic data creation, as illustrated in Figures 2 and 3. VULCAN operates by coordinating specialized agents across three distinct phases: **environment simulation**, where executable domains are synthesized; **task generation**, where realistic user scenarios are constructed; and **agent execution**, where the system runs agents within the simulated environment to collect high-quality interaction data. By constructing chains of user queries to model complex task dependencies, this approach reduces reliance on open-ended LLM-based user simulation and mitigates hallucination risks. The framework supports two classes of environments: *stateless environments*, where tools operate independently with deterministic outputs, and *stateful environments*, which capture multi-turn interactions where shared states evolve, and tool executions influence subsequent outcomes. Empirical analysis demonstrates that VULCAN effectively scales to generate diverse, high-quality agentic data. Our main contributions are:

- We introduce VULCAN, a unified and scalable framework that enables the simulation of both stateful and stateless agentic environments and supports large-scale generation of realistic user intents.
- We conduct comprehensive experiments demonstrating that the environments synthesized by VULCAN achieve high executability rates and that the method scales to generate diverse user scenarios with and without human intervention.
- We show that supervised fine-tuning of Qwen2-3 Yang et al. (2025) and Phi-series models Abdin et al. (2024) on VULCAN-generated data yields substantial performance gains across 4 benchmarks and 5 models, achieving performance comparable to frontier models.

2 Related Works

Tool-use Benchmarks and Agents. The evaluation of tool-using agents has shifted toward complex, stateful interactions. Recent benchmarks like BFCL Patil et al. (2025), ACEBench Chen et al. (2025), and Vita-Bench He et al. (2025b) assess multi-step execution under evolving dynamic constraints. Similarly, Tau-Bench Yao et al. (2024) and MCPVerse Lei et al. (2025) investigate interactive scenarios within large action spaces. In parallel, training datasets such as API Pack Guo et al. (2024) offer large-scale supervision. However, as agents advance toward long-horizon planning, there remains a critical need for environments that support dynamic workflow composition rather than static tool invocation.

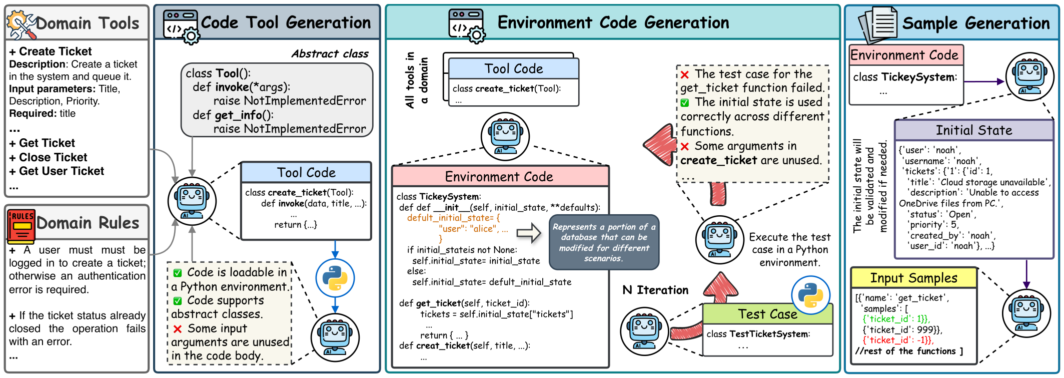

Figure 2: Overview of the environment simulation pipeline. **Stage 1** generates executable code for each tool from domain specifications and policy rules, followed by a verification loop to ensure correctness and reliability. **Stage 2** integrates all tools into a unified environment class with a shared initial state, while maintaining consistency through iterative validation. **Stage 3** samples diverse initial states and generates valid and invalid inputs for each tool to simulate a wide range of scenarios.

Agentic Data Generation. Generating synthetic training data requires robust environments, typically categorized as real-world or simulated. While real-world settings like TOUCAN Xu et al. (2025b) provide accurate feedback, they are costly and service-dependent. Fully LLM-simulated environments bypass these costs but often suffer from hallucinations and state inconsistencies. To address environment simulation limitations, current works such as EnvScaler Song et al. (2026) and AWM Wang et al. (2026) focus on scalable environment generation and actionable world modeling. However, their reliance on MCP servers for tool execution and the lack of a coherent method for task synthesis severely constrain their overall performance. Meanwhile, offline execution approaches (Barres et al., 2025; Prabhakar et al., 2025; Xu et al., 2025a) interact with local state files to bypass live APIs, yet they remain labor-intensive and struggle with user-intent hallucination. To bridge these gaps, we introduce a generalizable framework for simulating domain-specific environments, featuring fine-grained control over user intents and a highly scalable pipeline engineered to synthesize diverse, high-fidelity interaction trajectories.

3 METHOD

In this section, we present **VULCAN**, a general framework for simulating tool-use environments and generating real-world tasks to collect agentic experience for training efficient language models. We first formulate the problem mathematically and then detail the three main phases of VULCAN: **(1)** Environment Simulation, **(2)** Task Generation, and **(3)** Agent-Environment Interaction.

3.1 PROBLEM FORMULATION

Let $\mathcal{T} = \{\tau_1, \ldots, \tau_n\}$ be a set of tool specifications related to a domain D, where each tool specification defines a distinct functionality, including implicit or explicit operational rules, and some tools may perform read and write operations over shared variables. We formally define a simulator as a tuple $\mathcal{S} = (\mathcal{T}, \mathcal{X}, \mathcal{I}, \mathcal{O}, \Delta)$, where $\mathcal{T}$ is the set of tool specifications, $\mathcal{X}$ is the environment state space, $\mathcal{I}$ is the space of tool input arguments, $\mathcal{O}$ is the space of tool outputs, and $\Delta : \mathcal{X} \times \mathcal{T} \times \mathcal{I} \to \mathcal{X} \times \mathcal{O}$ is the transition function. Starting from an initial state $x^0 \in \mathcal{X}$, at each step k, a tool $\tau_k \in \mathcal{T}$ is executed with input $i_k \in \mathcal{I}$. The transition function Δ dictates how the environment evolves, producing a new state and output (x^{k+1}, o_k). A trajectory $\pi = \{(x^k, \tau_k, i_k, o_k)\}_{k=1}^{m}$ is considered valid if all tool calls adhere to their specifications and the transition function correctly implements the domain logic. When $\mathcal{X}$ is empty and tools operate independently, the simulator is defined as *stateless*; when $\mathcal{X}$ is non-empty and tools modify shared variables, the simulator is defined as *stateful*. The objective of VULCAN is to automatically synthesize the transition function Δ and the state space $\mathcal{X}$ to generate large sets of deterministic, diverse, and logically consistent trajectories π for agent training.

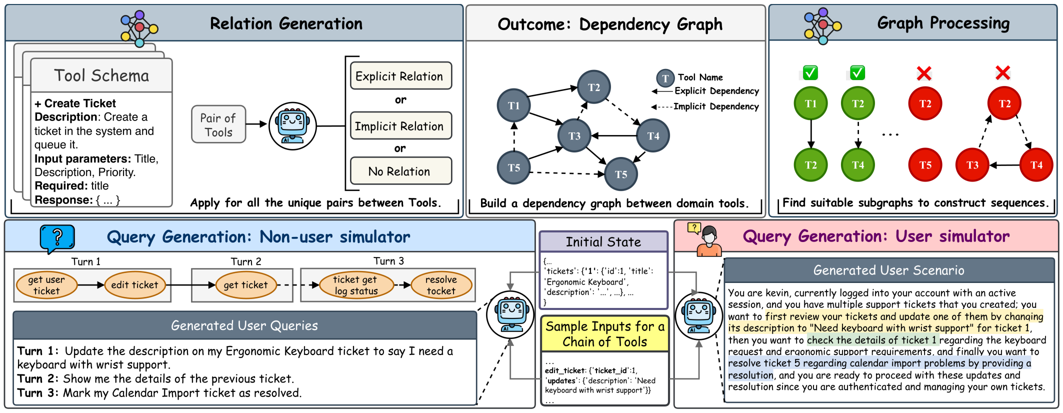

Figure 3: Overview of the task generation pipeline. Stage 1 identifies dependency types between tool specifications and constructs a directed dependency graph. Stage 2 extracts acyclic, fully connected subgraphs and composes them into valid tool sequences. Stage 3, given an initial state and input samples for each tool in a sequence, generates either a sequence of fixed user queries or a user scenario for interactive environments.

3.2 VULCAN Overview Framework

In many practical settings, executable environments are inaccessible. Additionally, realistic multi-step task distributions are difficult to construct at scale, creating a major bottleneck for training capable agents. As a result, building effective agentic systems requires jointly addressing two core components: an *executable environment* that supports interaction and a *task configuration* that defines the goal. VULCAN addresses this gap by automating the creation of both. Given a set of tool specifications, it builds a reliable, executable environment and leverages a graph-based formulation of tool dependencies to generate multi-step tasks that reflect real-world operational logic. The framework consists of three phases: **(1)** environment simulation, **(2)** task generation via dependency graphs, and **(3)** agent interaction within the simulated environment.

3.2.1 Phase 1: Environment Simulation

In this phase, we construct an executable and deterministic simulator $\mathcal{S} = (\mathcal{T}, \mathcal{X}, \mathcal{I}, \mathcal{O}, \Delta)$ by synthesizing both the transition function Δ and the state space $\mathcal{X}$ directly from tool specifications $\mathcal{T}$. We consider both stateless ($\mathcal{X} = \emptyset$) and stateful ($\mathcal{X} \neq \emptyset$) settings; in this section, we focus on the stateful case and defer stateless variants to Appendix A.

Stateful Environment Construction. Real-world domains often require sequential tool interactions where each action modifies the environment state $x \in \mathcal{X}$, influencing subsequent transitions. To capture this behavior, we construct a unified simulator that models shared state and inter-tool dependencies.

Stage 1: Tool Schema Preprocessing. Given the set of tool specifications $\mathcal{T}$, we first normalize and validate each $\tau \in \mathcal{T}$. This includes aligning tool descriptions with their intended functionality and ensuring consistency between tool names and semantics. For tools lacking explicit output definitions, we augment their schemas by generating structured output specifications in $\mathcal{O}$. This step ensures that each tool τ can be mapped to a well-defined function $\tau : \mathcal{X} \times \mathcal{I} \to \mathcal{X} \times \mathcal{O}$.

Stage 2: Tool Implementation Synthesis. For each $\tau \in \mathcal{T}$, we synthesize executable code that implements its behavior under the transition function Δ. The generated implementation enforces three properties: (1) deterministic execution given $(x, i) \in \mathcal{X} \times \mathcal{I}$, (2) robustness to invalid inputs via explicit error handling, and (3) complete utilization of all input arguments. When required, tools are augmented with auxiliary data structures that are incorporated into the global state space $\mathcal{X}$. A verifier module evaluates each implementation against these criteria, and synthesis is iteratively refined until correctness is achieved.

Stage 3: Environment Composition. We compose all tool implementations into a single environment class that defines the global transition function Δ. This step merges individual tool behaviors into a unified simulator with shared state $x \in \mathcal{X}$. The initial state x^0 is constructed to capture relevant domain variables and inter-tool dependencies. To ensure reliability, we introduce an automated verification loop that generates test cases over (τ, i) pairs and validates whether the resulting transitions (x', o) satisfy expected behaviors. The environment is refined iteratively until all test cases pass.

Stage 4: State and Input Sampling. To enable diverse trajectory generation, we sample multiple initial states $x^0 \sim \mathcal{X}$ and corresponding input arguments $i \sim \mathcal{I}$ for each tool τ. Inputs include both valid and adversarial cases, allowing the simulator to model realistic scenarios such as incorrect user inputs, inconsistent states, and partial observability. This process defines a distribution over initial configurations, improving diversity and coverage of the trajectory space π.

3.2.2 Phase 2: Task Configuration and Ground-Truth Generation

Given the simulator $\mathcal{S}$, we generate task configurations that define valid trajectories π. We consider two settings: (1) environment-driven tasks without user interaction, and (2) interactive tasks where a user provides incremental inputs.

Stage 1: Dependency Graph Construction. We construct a directed dependency graph $G = (\mathcal{T}, E)$ over the set of tools. For each pair of tools (τ_i, τ_j), we analyze their relationship based on their specifications. We define an *explicit dependency* when the output $o_i \in \mathcal{O}$ of τ_i directly serves as a required input for τ_j. In contrast, an *implicit dependency* arises when executing τ_i modifies the environment state $x \in \mathcal{X}$ in a way that is necessary for τ_j to operate correctly, even if no direct argument is passed. If neither condition holds, no edge is introduced. Applying this analysis across all tool pairs results in a directed graph where nodes represent tools and edges encode valid execution dependencies under the transition function Δ.

Stage 2: Chain-of-Tools Construction. Given the dependency graph G, we construct structured sequences of tools to define valid multi-step trajectories. While naive random traversal of G can produce feasible paths, such sequences are often suboptimal or lack meaningful structure. Instead, we first identify candidate subgraphs that satisfy two constraints: (1) they are acyclic, and (2) they are fully connected. Each subgraph represents a coherent unit of tool interactions.

We then compose these subgraphs into longer sequences to model multi-turn tasks. A valid sequence of subgraphs $\{G_1, \ldots, G_T\}$ satisfies the following conditions: (1) tools within each subgraph are distinct, (2) there exists a dependency between the terminal node of G_t and the initial node of G_{t+1}, ensuring continuity under Δ, and (3) no cyclic dependency is introduced across the composed sequence. This construction allows explicit control over both the number of interaction turns and the number of tool calls per turn. In practice, we enumerate candidate sequences by traversing G from different starting nodes and selecting those that satisfy the above constraints, resulting in a diverse set of valid tool-call trajectories.

Stage 3: Declarative Task Generation. Prior work, such as ToolLLM Qin et al. (2023) and subsequent approaches (Yin et al., 2025; Prabhakar et al., 2025), mainly generate procedural or loosely structured queries, where actions are explicitly described or limited to short, independent tool calls. In contrast, we aim to generate *declarative* tasks that specify only the high-level objective, requiring the agent to infer the correct sequence of tool calls under Δ.

Given a tool sequence $\{\tau_1, \ldots, \tau_m\}$ and an initial state $x^0 \in \mathcal{X}$, we sample input arguments $\{i_k\}$ from $\mathcal{I}$ to ensure a valid trajectory π. The task description abstracts intermediate steps and expresses only the final intent; for example, updating a ticket containing an "Ergonomic Keyboard" without specifying its identifier requires retrieving the ticket before modification, highlighting multi-step reasoning. We support two modes: (1) *non-user simulator*, where a sequence of fixed user queries is generated, and (2) *user simulator*, where a user scenario guides a simulated user agent during interaction. Each task is paired with a ground-truth plan derived from $\{(\tau_k, i_k)\}$, ensuring solvability and grounding in the executable environment (Figure 3).

3.2.3 Phase 3: Agent–Environment Interaction

In the final phase, we execute an agent within the simulator $\mathcal{S}$ to generate trajectories π. Starting from an initial state x^0, an LLM-based agent iteratively selects tool calls (τ_k, i_k) conditioned on prior observations. The environment applies the transition function $\Delta(x^k, \tau_k, i_k)$ to produce (x^{k+1}, o_k),

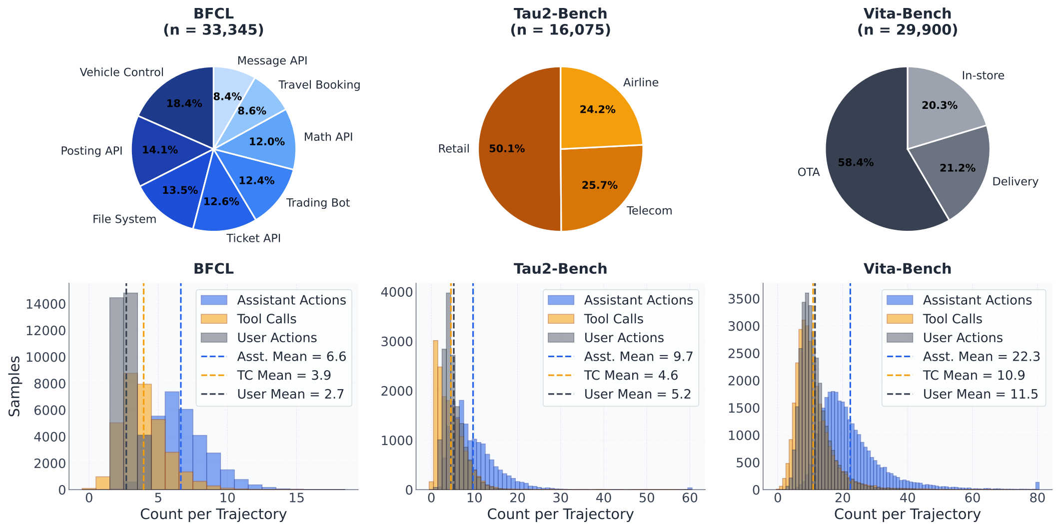

Figure 4: **Top:** Dataset composition of VULCAN across domains. **Bottom:** Distribution of agent tool calls and user queries in VULCAN datasets across domains.

which is fed back to the agent. To support interactive settings for tasks that require user involvement, a user simulator provides additional inputs at each step. This interaction continues until the task is completed or a stopping condition is met. Finally, a verification module evaluates each trajectory π against the ground-truth plan. Only trajectories that correctly follow valid transitions under Δ and achieve the task objective are retained. This process yields a large corpus of high-quality, fully executable trajectories without manual annotation.

3.3 Dataset Statistics

To generate data with VULCAN, we use `o4-mini` for stateless settings and `GPT-4.1` for stateful environments, implemented using the AgentInstruct framework Mitra et al. (2024). We generate 43K tasks for BFCL, 16K for Tau2-Bench, and 45K for Vita-Bench. We then deploy `GPT-4.1` as the agent to execute trajectories within $\mathcal{S}$, and as a user simulator where required. After execution, we filter trajectories using the ground-truth verification process and retain only successful trajectories π. Figure 4 presents statistics on tool usage, agent actions, and user interactions across different environments.

4 Experiments

We empirically evaluate the effectiveness of **VULCAN** by performing supervised fine-tuning (SFT) on open-weight language models using the synthesized datasets generated by our framework. We assess whether VULCAN improves tool-use capabilities, generalization across domains, and scalability. Our evaluation spans three standard benchmarks for tool-using agents: BFCL-v4, Tau2-Bench, and vita-bench, comparing against both base instruction models and strong frontier systems.

4.1 Experimental Setup

Models and Baselines. We fine-tune `Qwen2.5-Instruct` (3B, 7B, 14B), `Qwen3-8B`, and `Phi4-14B` on datasets generated by VULCAN. Detailed training configurations are provided in Appendix D. We compare our models against a diverse set of frontier systems, including `GPT-5.2`, `GPT-4.1`, `o4-mini`, `Gemini-2.5-Flash` (Comanici et al., 2025), `DeepSeek-V3.2` (Liu et al., 2025), `Qwen3-235B-A22B` (Yang et al., 2025), and `Kimi-K2` (Team et al., 2025), as well as their corresponding instruction-tuned baselines. We additionally compare VULCAN with prior data generation frameworks, including TOUCAN, which generates synthetic data via MCP-based

Table 1: Comparison of BFCL benchmark performance between the VULCAN-fine-tuned model and the baseline model.

Model	**Overall**	**Single Turn**		**Multi Turn**	**Hallucination**
		Non-live (AST)	*Live (AST)*		*Irrelevance*
GPT-5.2	52.67%	81.85%	70.39%	28.12%	79.42%
GPT-4.1	59.32%	82.79%	69.95%	38.88%	86.52%
o4-mini	48.19%	81.29%	70.76%	16.62%	87.25%
Gemini-2.5-Flash	60.29%	84.96%	74.39%	36.25%	93.67%
DeepSeek-V3.2	55.33%	88.54%	77.34%	29.87%	76.49%
Qwen3-235B-A22B	60.27%	87.90%	77.03%	40.12%	76.32%
Kimi-K2	66.58%	81.60%	78.68%	50.63%	87.34%
Qwen2.5-3B-Instruct	38.16%	80.08%	67.43%	6.60%	61.69%
with VULCAN	46.33%$^{+8.17\%}$	84.75%	66.32%	17.12%	75.53%
Qwen2.5-7B-Instruct	43.84%	84.19%	72.32%	12.88%	67.93%
with VULCAN	52.80%$^{+8.96\%}$	84.17%	71.95%	25.75%	83.44%
Qwen3-8B	60.32%	87.67%	80.38%	38.12%	79.56%
with VULCAN	63.51%$^{+3.19\%}$	86.29%	77.42%	45.62%	80.49%
Phi4-14B	38.24%	69.56%	60.70%	3.88%	87.55%
with VULCAN	50.47%$^{+12.23\%}$	84.35%	76.09%	22.37%	75.26%
Qwen2.5-14B-Instruct	47.46%	83.38%	73.70%	19.75%	68.46%
with VULCAN	56.54%$^{+9.08\%}$	89.12%	77.20%	32.12%	76.58%

environments, and EnvScaler and AWM, two concurrent approaches for environment simulation. This setup enables a comprehensive evaluation across models of varying scales and capabilities.

Benchmarks. We evaluate the fine-tuned models on **BFCL-v4**, **Tau-Bench**, **Tau2-Bench**, and **Vita-Bench**, which cover both single-turn and multi-turn tool-use scenarios across diverse domains. All experiments are conducted on a cluster with two nodes, each equipped with $8\times$ NVIDIA H100 GPUs. Additional benchmark details are provided in Appendix B.

4.2 Experimental Results

VULCAN improves tool-use performance. As shown in Table 1, models fine-tuned on VULCAN data consistently outperform their instruction-tuned counterparts. On BFCL, `Qwen2.5-3B`, `Qwen2.5-7B`, `Qwen2.5-14B`, `Qwen3-8B`, and `Phi4-14B` achieve improvements of 8.17%, 8.96%, 9.08%, 3.19%, and 12.23%, respectively. Notably, the VULCAN-tuned `Qwen3-8B` surpasses `GPT-4.1`, `GPT-5.2`, `Gemini-2.5-Flash`, and `DeepSeek-V3.2`, and achieves performance comparable to `Kimi-K2`. Similar improvements are observed on Tau2-Bench and Vita-Bench (Figure 1), demonstrating that VULCAN enhances both tool selection and multi-step reasoning capabilities.

Generalization across domains. To evaluate generalization, we simulate multiple domains within Tau2-Bench (retail, airline, telecom) and Vita-Bench (delivery, in-store, OTA). As shown in Figure 1, models fine-tuned on VULCAN data achieve consistent improvements across all benchmarks. Furthermore, larger models benefit more from VULCAN-generated data, indicating that the framework scales effectively with model capacity.

Comparison with existing methods. We compare VULCAN with recent approaches such as TOUCAN (Xu et al., 2025b), which relies on real-world environments and large-scale MCP-based data collection. As shown in Figure 1, VULCAN-tuned models outperform TOUCAN by 3.5%, 5.3%, 5.8%, and 20.5% on BFCL, Tau-Bench, Tau2-Bench, and Vita-Bench, respectively. We further compare `Qwen3-8B` fine-tuned with VULCAN against models trained using EnvScaler and

Table 2: Estimation of token usage and cost for simulating an environment with 27 tools, dependency graph construction, and user scenario generation using GPT-4.1.

Metric	Environment Simulation (per env.)			Task Generation (per task)			Dep. Graph (per domain)	Total
	Pre-processing	*Tool Code Gen.*	*Env. Code Gen.*	*Init. State Gen.*	*Input Sample Gen.*	*User Scenario Gen.*	*Graph Building*	
Token	49.6K	362.6K	297.6K	2.8K	26.0K	11.5K	1,04M	**1,78M**
Cost	$0.14	$1.20	$1.03	$0.01	$0.08	$0.02	$2.31	**$4.80**

AWM, which are parallel works on environment simulation. Results in Figure 8 indicate that the VULCAN-tuned model surpasses the EnvScaler-tuned model by 15.1% on BFCL, and outperforms the AWM-tuned model by 3.7% and 4.3% on BFCL and Tau2-Bench, respectively. Additional results and analysis are provided in Appendix C.2.

4.3 Ablations and Analysis

Effectiveness of generating a sequence of fixed user queries. We hypothesize that generating a sequence of fixed user queries aligned with a tool chain is more effective than generating a single aggregated query describing the entire task. To evaluate this, we construct two settings over the same set of tool sequences: (1) a single-query setting, where one comprehensive user query encodes the full task, and (2) a sequence-based setting, where multiple fixed user queries are generated corresponding to each step of the tool chain. Since both settings are derived from the same underlying sequences, they share identical ground-truth plans. We generate trajectories by deploying `GPT-4.1` as the agent within the simulated BFCL environments.

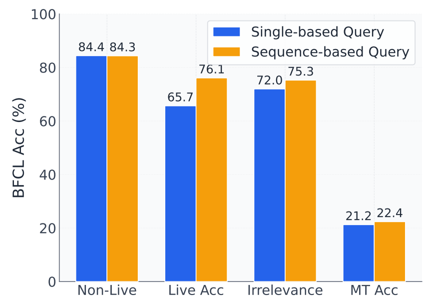

Figure 5: Performance comparison of single-query (SQ) and multi-turn query (MQ) multi-step tasks on BFCL, with `Phi4-14B` used as the instruction-tuned model

As shown in Figure 5, the sequence-based formulation consistently outperforms the single-query setting. We attribute this improvement to the fact that step-wise queries introduce controlled ambiguity and require the agent to reason over intermediate states and tool dependencies. In contrast, a single aggregated query often implicitly reveals the solution structure, reducing the need for multi-step reasoning and limiting exploration. These results validate the effectiveness of our task formulation strategy.

Cost and token usage analysis. VULCAN comprises three components: environment simulation, dependency graph construction, and task generation. To assess its efficiency, Table 2 reports the average token usage for simulating a single environment, which scales with tool complexity. We also estimate the cost of task generation and show that both environment construction and data synthesis are significantly cheaper than real-world data collection. These results highlight the scalability and cost efficiency of VULCAN.

Deduplication: overlap analysis. Following Muennighoff et al. (2025), we compute 8-gram overlap between the training data and benchmark queries. We observe minimal overlap: 0.38% for BFCL and 0.53% for Vita-Bench, while no overlap is observed for Tau2-Bench. All overlapping samples are removed from the training set to prevent potential data contamination. These results indicate that the observed performance gains are not driven by memorization, but rather by the quality and diversity of the data generated by VULCAN.

Error analysis. We analyze failure cases across benchmarks, as illustrated in Figure 6. In BFCL, the majority of failures stem from incomplete task execution, where the agent does not initiate tool calls, largely due to the absence of a simulated user to guide the interaction. In contrast, errors in Tau2-Bench are mainly caused by permission violations, reflecting the strict domain constraints imposed by the environment. For Vita-Bench, failures are dominated by hallucination and fabrication, where the agent attempts to complete tasks without invoking the appropriate tools. Overall, these findings demonstrate that failure patterns differ substantially across environments, with

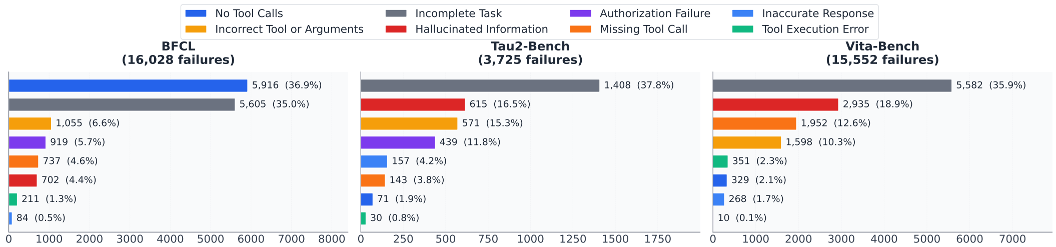

Figure 6: Distribution of agent errors in task execution across different environments.

non-interactive settings leading to under-execution, while interactive settings introduce additional challenges related to policy compliance and proper grounding. Refer to Appendix C.4 for error descriptions.

Table 3: Comparison of BFCL performance between VULCAN-MT and VULCAN-ST.

Model	Overall	Single Turn		Multi Turn	Hallucination
		Non-live (AST)	*Live (AST)*		*Irrelevance*
Phi4-14B	38.24%	69.56%	60.70%	3.88%	**87.55%**
with VULCAN-ST	49.97%	84.04%	73.87%	21.25%	78.21%
with VULCAN-MT	**50.47%**	**84.35%**	**76.09%**	**22.37%**	75.26%

Stateful environments demonstrate superior sample efficiency. We analyze the impact of environment type on data efficiency. VULCAN-MT, generated under the stateful setting, is built from only 129 tools and contains 33K agentic trajectories, whereas VULCAN-ST, generated under the stateless setting, covers 1,288 tools and contains 119K single-query examples. Despite the significantly lower tool diversity, VULCAN-MT achieves comparable fine-tuning performance to VULCAN-ST (Table 3). Notably, achieving similar performance in the stateless setting requires approximately $4\times$ more training data, highlighting that stateful, multi-turn trajectories provide a substantially more efficient training signal.

5 Conclusion

In this paper, we introduced **VULCAN**, a scalable framework for automatically synthesizing executable, state-aware environments from static tool specifications. Beyond simulation, VULCAN employs a graph-based approach to generate diverse, multi-turn user tasks that capture realistic dependencies. Our evaluation on the BFCL, Tau-Bench, Tau2-Bench, and Vita-Bench benchmarks confirms the reliability of these synthesized environments and demonstrates that supervised fine-tuning on VULCAN-generated data yields substantial performance gains over strong baselines. Notably, our results highlight the superior sample efficiency of learning from stateful, multi-turn trajectories compared to stateless interactions. Consequently, VULCAN offers a vital solution for training robust agentic systems in domains where access to real-world environments is limited, costly, or restricted by privacy concerns.

Limitations and Future Work. Although VULCAN addresses key challenges in simulating agentic environments, several directions remain for future work. Our current framework assumes independent environments, whereas many real-world tasks require coordination across multiple environments. Extending VULCAN to support cross-environment reasoning is an important next step. Additionally, while prior work highlights the benefits of reinforcement learning for tool-use (Qian et al., 2025; Tan et al., 2025), our approach focuses on supervised data generation. Integrating reinforcement learning to further improve agent behavior within simulated environments is left for future work.

ETHICS STATEMENT

We have utilized AI assistants, specifically Grammarly and ChatGPT, to correct grammatical errors and rephrase sentences.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. tau2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Fouad Bousetouane. Agentic systems: A guide to transforming industries with vertical ai agents. *arXiv preprint arXiv:2501.00881*, 2025.

Chen Chen, Xinlong Hao, Weiwen Liu, Xu Huang, Xingshan Zeng, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Yuefeng Huang, et al. Acebench: Who wins the match point in tool usage? *arXiv preprint arXiv:2501.12851*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Runnan Fang, Shihao Cai, Baixuan Li, Jialong Wu, Guangyu Li, Wenbiao Yin, Xinyu Wang, Xiaobin Wang, Liangcai Su, Zhen Zhang, et al. Towards general agentic intelligence via environment scaling. *arXiv preprint arXiv:2509.13311*, 2025.

Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. From llm reasoning to autonomous ai agents: A comprehensive review. *arXiv preprint arXiv:2504.19678*, 2025.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

Zhen Guo, Adriana Meza Soria, Wei Sun, Yikang Shen, and Rameswar Panda. Api pack: A massive multi-programming language dataset for api call generation. *arXiv preprint arXiv:2402.09615*, 2024.

Muyu He, Anand Kumar, Tsach Mackey, Meghana Rajeev, James Zou, and Nazneen Rajani. Impatient users confuse ai agents: High-fidelity simulations of human traits for testing agents. *arXiv preprint arXiv:2510.04491*, 2025a.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, et al. Vitabench: Benchmarking llm agents with versatile interactive tasks in real-world applications. *arXiv preprint arXiv:2509.26490*, 2025b.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*, 2025.

Fei Lei, Yibo Yang, Wenxiu Sun, and Dahua Lin. Mcpverse: An expansive, real-world benchmark for agentic tool use. *arXiv preprint arXiv:2508.16260*, 2025.

Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on llm-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, 2024.

Yuetai Li, Huseyin A Inan, Xiang Yue, Wei-Ning Chen, Lukas Wutschitz, Janardhan Kulkarni, Radha Poovendran, Robert Sim, and Saravan Rajmohan. Simulating environments with reasoning models for agent training. *arXiv preprint arXiv:2511.01824*, 2025.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

Venkatesh Mishra, Amir Saeidi, Satyam Raj, Mutsumi Nakamura, Jayanth Srinivasa, Gaowen Liu, Ali Payani, and Chitta Baral. How can input reformulation improve tool usage accuracy in a complex dynamic environment? a study on tau-bench. *arXiv preprint arXiv:2508.20931*, 2025.

Arindam Mitra, Luciano Del Corro, Guoqing Zheng, Shweti Mahajan, Dany Rouhana, Andres Codas, Yadong Lu, Wei-ge Chen, Olga Vrousgos, Corby Rosset, et al. Agentinstruct: Toward generative teaching with agentic flows. *arXiv preprint arXiv:2407.03502*, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Melissa Z Pan, Negar Arabzadeh, Riccardo Cogo, Yuxuan Zhu, Alexander Xiong, Lakshya A Agrawal, Huanzhi Mao, Emma Shen, Sid Pallerla, Liana Patel, et al. Measuring agents in production. *arXiv preprint arXiv:2512.04123*, 2025.

Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, et al. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. *arXiv preprint arXiv:2504.03601*, 2025.

Cheng Qian, Zuxin Liu, Akshara Prabhakar, Jielin Qiu, Zhiwei Liu, Haolin Chen, Shirley Kokane, Heng Ji, Weiran Yao, Shelby Heinecke, et al. Userrl: Training interactive user-centric agent via reinforcement learning. *arXiv preprint arXiv:2509.19736*, 2025.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

Xiaoshuai Song, Haofei Chang, Guanting Dong, Yutao Zhu, Zhicheng Dou, and Ji-Rong Wen. Envscaler: Scaling tool-interactive environments for llm agent via programmatic synthesis. *arXiv preprint arXiv:2601.05808*, 2026.

Hongjin Su, Shizhe Diao, Ximing Lu, Mingjie Liu, Jiacheng Xu, Xin Dong, Yonggan Fu, Peter Belcak, Hanrong Ye, Hongxu Yin, et al. Toolorchestra: Elevating intelligence via efficient model and tool orchestration. *arXiv preprint arXiv:2511.21689*, 2025.

Weiting Tan, Xinghua Qu, Ming Tu, Meng Ge, Andy T Liu, Philipp Koehn, and Lu Lu. Process-supervised reinforcement learning for interactive multimodal tool-use agents. *arXiv preprint arXiv:2509.14480*, 2025.

Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Zhaoyang Wang, Canwen Xu, Boyi Liu, Yite Wang, Siwei Han, Zhewei Yao, Huaxiu Yao, and Yuxiong He. Agent world model: Infinity synthetic environments for agentic reinforcement learning. *arXiv preprint arXiv:2602.10090*, 2026.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, et al. Mcp-bench: Benchmarking tool-using llm agents with complex real-world tasks via mcp servers. *arXiv preprint arXiv:2508.20453*, 2025.

Zengzhuang Xu, Bingguang Hao, Zechuan Wang, Yuntao Wen, Maolin Wang, Yang Liu, Long Chen, Dong Wang, Yicheng Chen, Cunyin Peng, et al. Funreason-mt technical report: Overcoming the complexity barrier in multi-turn function calling. *arXiv preprint arXiv:2510.24645*, 2025a.

Zhangchen Xu, Adriana Meza Soria, Shawn Tan, Anurag Roy, Ashish Sunil Agrawal, Radha Poovendran, and Rameswar Panda. Toucan: Synthesizing 1.5 m tool-agentic data from real-world mcp environments. *arXiv preprint arXiv:2510.01179*, 2025b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. tau-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Fan Yin, Zifeng Wang, I-Hung Hsu, Jun Yan, Ke Jiang, Yanfei Chen, Jindong Gu, Long Le, Kai-Wei Chang, Chen-Yu Lee, et al. Magnet: Multi-turn tool-use data synthesis and distillation via graph translation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32600–32616, 2025.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

Andy Zou, Maxwell Lin, Eliot Jones, Micha Nowak, Mateusz Dziemian, Nick Winter, Alexander Grattan, Valent Nathanael, Ayla Croft, Xander Davies, et al. Security challenges in ai agent deployment: Insights from a large scale public competition. *arXiv preprint arXiv:2507.20526*, 2025.

APPENDIX

A STATELESS ENVIRONMENT

In the main text, we introduced a stateful environment simulation and a pipeline for producing declarative user-query tasks. To further demonstrate the effectiveness of the proposed framework, we also consider a *stateless* environment setting. In contrast to the stateful case, where tools interact through a shared state $\mathcal{X}$, the stateless setting assumes $\mathcal{X} = \emptyset$, meaning that tools $\tau \in \mathcal{T}$ operate independently without persistent interactions. The main advantage of this setting is improved scalability compared to the stateful simulation, since it removes the need to maintain and update a global state across tool executions. To support this setting, we introduce a dedicated pipeline for environment construction and task generation. We further apply this pipeline to BFCL as a case study and compare its performance with a base instruction-tuned model.

A.1 PHASE 1: ENVIRONMENT SIMULATION

Given a tool set $\mathcal{T}$, we construct a stateless simulator $\mathcal{S} = (\mathcal{T}, \emptyset, \mathcal{I}, \mathcal{O}, \Delta)$ through the following four steps:

1. **Tool specification refinement.**
   Each tool specification $\tau \in \mathcal{T}$ is standardized to ensure semantic correctness and completeness. When necessary, the specification is augmented with a structured response schema, ensuring that the output space $\mathcal{O}$ is well-defined and deterministic.
2. **Code synthesis and verification.**
   Based on the refined specifications, an implementation of the transition function Δ is constructed for each tool. During this process, the tool logic is analyzed to determine whether auxiliary storage is required. For example, write-oriented functions such as `save` may require extending the input space $\mathcal{I}$ with an additional `data` field. Each implementation is then verified with respect to three criteria: executability, correct argument usage, and compliance with a predefined abstract interface.
3. **Mock data construction.**
   To ensure scalability and robustness, we avoid relying on a single monolithic database. Instead, we construct multiple independent `buckets` of mock data. Each bucket contains a set of input arguments from $\mathcal{I}$ and, when required, a lightweight isolated database instance. This allows the simulator to represent diverse tool behaviors across different contexts without introducing shared state.
4. **Code validation via forward–backward loop.**
   To ensure that the implemented transition function Δ is correct, we employ a forward–backward validation loop. In the forward pass, tools are executed on sampled mock inputs. In the backward pass, the resulting outputs, or error traces, are compared against the expected behavior defined by the tool specifications. If inconsistencies are detected, the implementation is revised. This process repeats until the implementation satisfies the required constraints, systematically improving the reliability of the simulator.

A.2 PHASE 2: TASK CONFIGURATION AND GROUND TRUTH GENERATION

With the simulated environment $\mathcal{S}$ established, the next phase constructs task configurations together with their ground-truth solutions. In addition to the user query, this phase automatically produces a ground-truth scenario used to evaluate the agent's behavior. If the scenario is *normal* (solvable), the ground truth consists of the ordered list of tool calls required to achieve the goal. If the scenario is *abnormal* (e.g., missing tools, insufficient arguments, or invalid logic), the ground truth consists of the expected behavioral explanation, such as clarifying why the request cannot be fulfilled. This formulation ensures that the agent is evaluated not only on tool execution, but also on its ability to correctly handle diverse user intents. We categorize tasks into **single-turn** and **multi-turn** user queries.

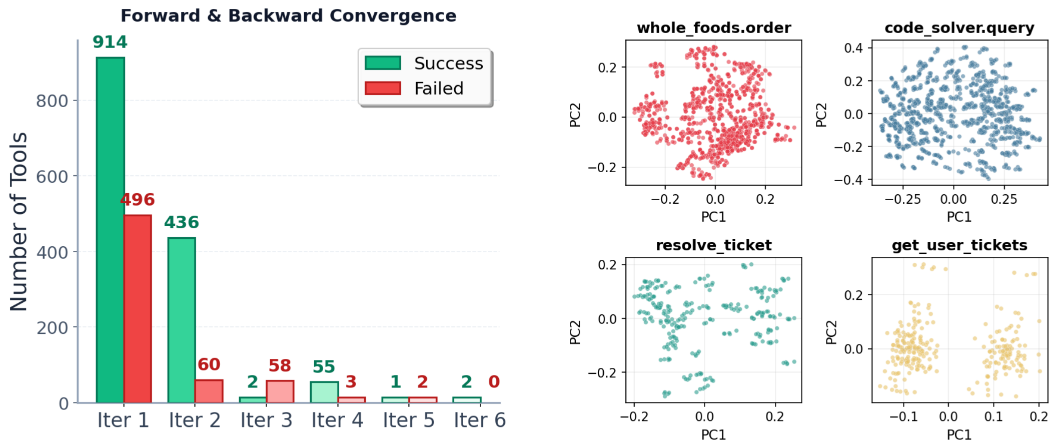

Figure 7: **Left:** Convergence behavior of the forward–backward validation process, showing the number of iterations until stabilization of generated tool implementations and inputs across 1,410 BFCL tools. **Right:** Embedding visualization of samples generated for four representative tools, illustrating the diversity of input instances produced for each tool.

Single-Turn Tasks. In single-turn scenarios, the agent must correctly map a user request to a specific tool $\tau \in \mathcal{T}$, or determine that the request cannot be fulfilled. These queries are designed to evaluate tool selection accuracy, identification of missing functions, and detection of missing arguments. More specifically, they test whether the requested capability is absent from $\mathcal{T}$, or whether the request does not provide sufficient information from $\mathcal{I}$ to execute the corresponding tool.

We classify these tasks into:

1. **Simple:** The task requires exactly one tool call and all required arguments are present.
2. **Multiple:** The task requires invoking the same tool multiple times with different arguments.
3. **Independent Parallel:** The task requires multiple different tools, but their execution is parallelizable since $\mathcal{X} = \emptyset$.

A.3 Phase 3: Agent–Environment Interaction

In the final phase, we collect the actual agent interaction trajectories. Because the environment is stateless, each tool execution depends only on the selected tool $\tau_k \in \mathcal{T}$ and the corresponding input $i_k \in \mathcal{I}$, and produces an output $o_k \in \mathcal{O}$ according to Δ, without modifying any shared state. An agent is instantiated with a generated user query and the list of available tools, and attempts to solve the task by interacting with the simulator $\mathcal{S}$. Since the environment is fully executable, every tool call returns concrete feedback in the form of outputs or error messages, which the agent can use to guide subsequent actions. After task completion, the final result is compared against the ground-truth answer derived during task generation. Only trajectories π that are successfully solved and verified are retained. This automated pipeline enables scalable collection of high-quality training data without manual annotation.

A.4 BFCL: Case Study

We implement VULCAN on top of the AgentInstrcut Mitra et al. (2024) framework to maximize concurrency. We evaluate VULCAN on the Berkeley Function Calling Leaderboard (BFCL) benchmark to demonstrate its ability to construct executable tool environments and generate agentic interaction data under the stateless setting. We extract all tool specifications from BFCL-v3, which contains 1,410 unique tools. These tools fall into two categories. The first category consists of tools provided only as specifications without executable implementations; these are primarily used to evaluate an agent's tool-selection capability. For the stateless setting, we synthesize executable environments for the tools that lack implementations. BFCL contains 1,288 tools without executable implementations. We simulate these tools and generate agentic tasks through a four-stage pipeline.

Stage 1: Specification normalization and code synthesis. Tool specifications are first normalized. Our analysis indicates that 8% of the tools contain ambiguous or underspecified descriptions. Missing response schemas are introduced to explicitly define the output structure when necessary. Based on the resulting specification and a predefined abstract interface, a Python class is synthesized for each tool. Each class exposes two methods: `invoke`, which implements the tool logic, and `info`, which returns the tool schema. All generated code is iteratively verified to ensure (i) executability, (ii) complete usage of all input arguments, and (iii) conformance to the abstract interface. We observe that different coding models exhibit varying tendencies regarding database integration: specifically, `o4-mini` integrates an extra database for 44.5% of the tools, whereas `gpt-4.1` considers a database necessary for 85.2% of the tools.

Stage 2: Input and database synthesis. For each tool, we generate 10 candidate values per input argument and, when required, a lightweight mock database as a bucket. This process is repeated for 15 iterations to promote diversity (see Figure 7, Right). All generated artifacts must conform to a strict dictionary schema; invalid outputs trigger automatic regeneration. For tools without input arguments, no database or sample pool is created.

Stage 3: Execution-based validation. All tools are executed on the synthesized inputs within a secure Python `exec` environment, and their outputs are recorded. We randomly sample 30 executions per tool for verification. We define four failure modes: execution timeouts, logical inconsistencies in outputs, malformed inputs or database states, and runtime errors. When a failure is detected, the implementation is revised, and the validation loop repeats until all constraints are met. Figure 7 shows that over 64% of the synthesized tools satisfy all executability and validation criteria within eight iterations.

Stage 4: Input stratification and task construction. We systematically partition tool inputs into three categories using a rule-based approach that inspects the required and optional arguments defined in each tool's schema. The categories are: (i) **complete inputs**, where all arguments are provided (50% ratio); (ii) **missing required inputs**, generated by programmatically identifying and removing at least one required argument (20% ratio); and (iii) **minimal inputs**, where only the strictly required arguments are retained (30% ratio). This deterministic stratification enables the systematic generation of diverse user behaviors, including ambiguous or incomplete requests that challenge an agent's error-handling capabilities. Finally, these stratified inputs are combined with personas sampled from PersonaHub (Ge et al., 2024) to construct rich, realistic user scenarios.

B Details of the Evaluation Benchmarks

B.1 Berkeley Function-Calling Leaderboard (BFCL)

The Berkeley Function-Calling Leaderboard (BFCL) is a benchmark for evaluating tool-use capabilities of language models across Single-Turn, Hallucination, and Multi-Turn settings (Patil et al., 2025). In BFCL-v4, the Live setting contains 1,351 samples, the Non-Live setting contains 1,150 samples, the Hallucination Measurement setting contains 1,122 samples, and the Multi-Turn setting contains 800 samples. The Live and Non-Live settings evaluate single-turn tool calling, where models must select the correct tool and generate valid arguments; these predictions are evaluated using AST-based matching. The Hallucination setting measures whether models correctly abstain from tool calls when the available tools are irrelevant or insufficient. The Multi-Turn setting evaluates sequential tool use in executable environments that require reasoning over multiple interaction steps. For evaluation, we follow the BFCL-v4 scoring formulation while excluding the agentic component. The overall score is computed as:

$$\text{Overall Score} = (\text{Multi-Turn} \times 0.50) + (\text{Live} + \text{Non-Live} + \text{Hallucination}) \times 0.50$$

B.2 Tau2-Bench

Tau2-bench is a multi-domain benchmark designed to evaluate tool-using conversational agents in realistic, interactive settings where both the agent and the user can act on a shared environment. It spans three domains, retail, airline, and telecom, with each domain defined by a set of tools, policies, and task configurations (Barres et al., 2025). The retail domain contains 13 agent tools (7 write and 6 read) and 115 tasks, the airline domain contains 12 agent tools (6 write and 6 read) and 50 tasks, and the telecom domain contains 13 agent tools (6 write and 7 read), 30 user tools (15 write and 15 read), and 114 evaluation tasks sampled from a larger pool of 2,285 programmatically generated tasks (Barres et al., 2025). The benchmark follows a dual-control setup in which both the agent and a simulated user can invoke tools and modify the environment state, requiring coordination, communication, and policy compliance. Each task consists of a multi-turn interaction where the agent must gather information, guide the user when necessary, and execute tool calls to achieve a shared objective. For evaluation, we report the Pass@1 metric, which measures the proportion of tasks successfully solved by the agent. It is defined as:

$$\text{Pass@1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\text{success}_i),$$

where N is the total number of tasks and $\mathbf{1}(\text{success}_i)$ is an indicator function that equals 1 if the agent successfully completes task i, and 0 otherwise.

B.3 VITA-BENCH

Vita-bench is a benchmark for evaluating tool-using agents in interactive, multi-turn environments across real-world domains, including delivery, in-store consumption, and online travel (OTA) (He et al., 2025b). It provides a unified simulation framework with a total of 66 tools, distributed across domains with 24 tools for delivery, 21 for in-store, and 21 for OTA. These tools can be composed to form both single-domain and cross-domain tasks, enabling agents to reason across heterogeneous environments. The benchmark includes 300 single-domain tasks and 100 cross-domain tasks, each requiring multi-step reasoning, tool selection, and interaction with the environment. Compared to Tau2-bench, which adopts a dual-control setting with explicit user-agent coordination and domain policies, Vita-bench focuses on agent-driven execution within a unified environment, where task complexity arises from cross-domain composition and inter-tool dependencies. For evaluation, we report the Pass@1 metric, which measures the proportion of tasks successfully solved by the agent. It is defined as:

$$\text{Pass@1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\text{success}_i),$$

where N is the total number of tasks and $\mathbf{1}(\text{success}_i)$ is an indicator function that equals 1 if the agent successfully completes task i, and 0 otherwise.

C MORE EXPERIMENTS AND DETAILS

In this section, we present additional experiments and provide detailed comparisons with prior and concurrent works.

C.1 DETAILED BFCL, TAU2-BENCH AND VITA-BENCH RESULTS

We compared VULCAN with TOUCAN across multiple benchmarks using different sizes of Qwen2.5 models. Tables 4, 5 and 6 report detailed performance across domains for BFCL, Tau2-bench, and vita-bench, respectively. To ensure a fair comparison on Tau2-bench, we follow the TOUCAN setup and use `GPT-4o` as the user simulator. For vita-bench, we use `GPT-4.1` as the user simulator. We report the scores for TOUCAN on BFCL and Tau2-bench as provided in the original paper, and additionally evaluate TOUCAN models on vita-bench.

C.2 DETAILED COMPARISON WITH EXISTING WORKS

We also compare VULCAN with AWM Wang et al. (2026) and EnvScaler Song et al. (2026), which fine-tune Qwen3-series models to demonstrate performance improvements. Figure 8 presents the results for the `Qwen3-8B` model as reported in the AWM paper. We do not perform separate evaluations for these methods and instead rely on their reported results. As shown, VULCAN consistently outperforms both approaches across BFCL and Tau2-bench, highlighting the effectiveness of our framework.

Table 4: Comparison of VULCAN- and TOUCAN-tuned models on BFCL.

Model	**Overall**	**Single Turn**		**Multi Turn**	**Hallucination**
		Non-live (AST)	*Live (AST)*		*Irrelevance*
Qwen2.5-3B-Instruct	38.16%	80.08%	**67.43%**	6.60%	61.69%
with VULCAN	**46.33%**	**84.75%**	66.32%	**17.12%**	**75.53%**
Qwen2.5-7B-Instruct	43.85%	**84.19%**	72.32%	12.88%	67.93%
with TOUCAN	49.34%	78.52%	**74.50%**	22.62%	75.18%
with VULCAN	**52.80%**	84.17%	71.95%	**25.75%**	**83.44%**
Qwen2.5-14B-Instruct	47.46%	83.38%	73.70%	19.75%	68.46%
with TOUCAN	**57.19%**	85.42%	76.01%	**35.25%**	75.96%
with VULCAN	56.54%	**89.12%**	**77.20%**	32.12%	**76.58%**

Table 5: Comparison of VULCAN- and TOUCAN-tuned models on Tau-Bench and Tau2-Bench.

Model	Tau-Bench			Tau2-Bench			
	Avg.	*Airline*	*Retail*	*Avg.*	*Airline*	*Retail*	*Telecom*
Qwen2.5-3B-Instruct	8.35%	8.00%	8.70%	12.72%	**18.00%**	11.40%	8.77%
with VULCAN-MT	**12.22%**	**14.00%**	**10.43%**	**14.97%**	16.00%	**14.91%**	**14.00%**
Qwen2.5-7B-Instruct	15.03%	8.75%	21.30%	16.08%	14.00%	17.54%	16.70%
with TOUCAN	22.48%	15.50%	29.46%	17.77%	20.00%	22.80%	**10.50%**
with VULCAN	**27.83%**	**20.00%**	**35.65%**	**23.37%**	**28.00%**	**33.33%**	8.77%
Qwen2.5-14B-Instruct	30.85%	17.25%	44.46%	24.46%	12.00%	41.20%	20.18%
with TOUCAN	35.24%	22.00%	48.48%	30.43%	22.00%	**49.10%**	20.18%
with VULCAN	**38.35%**	**28.00%**	**48.69%**	**35.01%**	**34.00%**	48.24%	**22.80%**

C.3 More Findings

Failure analysis reveals challenges in long-horizon planning. We conduct a rigorous failure analysis using an LLM-as-a-judge framework. For stateless scenarios, we categorize errors as summarized in Table 8. As illustrated in Figure 9, the predominant failure mode is *incorrect tool invocation*, followed by *violations of ground-truth constraints*, suggesting that while VULCAN improves general capability, precise adherence to complex constraints remains a critical area for future optimization.

Across scalability. We also present the statistics for input-output pairs generated by `o4-mini` and `GPT-4.1` in Table 7. Although VULCAN datasets are derived from random sampling of these pairs, the results exemplify VULCAN's capacity to synthesize an arbitrary volume of diverse tool-calling scenarios, thereby demonstrating the inherent scalability of the framework.

C.4 Details of the Error Analysis

In this section, we present the error categories used to evaluate single-turn tasks generated under stateless settings (Table 8). We also present the error categories for multi-turn tasks generated under stateful settings, as shown in Table 9.

D Fine-Tuning Hyperparameters

We fine-tune Qwen-series models and `Phi4-14B` using VULCAN on two nodes of a supercomputing cluster equipped with NVIDIA H100 GPUs. The maximum sequence length is set to 16,384

Table 6: Comparison of VULCAN- and TOUCAN-tuned models on Vita-Bench.

Model	Vita-Bench				
	Avg.	*Delivery*	*In Store*	*OTA*	*Cross Scenario*
Qwen2.5-3B-Instruct	1.38%	4.40%	1.10%	0.00%	0.00%
with VULCAN	**7.83%**	**19.00%**	**10.30%**	**2.00%**	0.00%
Qwen2.5-7B-Instruct	2.45%	3.03%	5.05%	0.00%	1.70%
with TOUCAN	2.25%	7.00%	1.00%	1.00%	0.00%
with VULCAN	**18.25%**	**26.00%**	**31.00%**	**13.00%**	3.00%
Qwen2.5-14B-Instruct	16.55%	27.00%	22.00%	10.70%	**6.50%**
with TOUCAN	0.78%	0.00%	2.00%	0.00%	1.10%
with VULCAN	**21.25%**	**30.00%**	**30.00%**	**22.00%**	3.00%

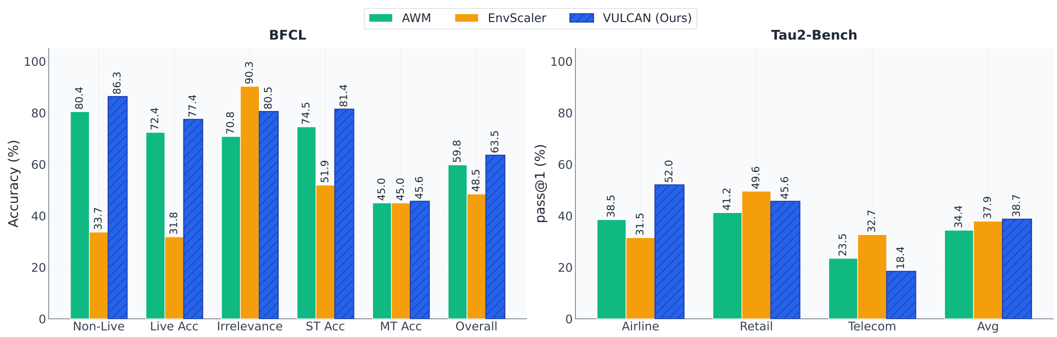

Figure 8: Comparison of VULCAN with AWM and EnvScaler on BFCL and Tau2-Bench.

for `Phi4-14B` and 32,768 for the Qwen2.5 and Qwen3 models. Detailed hyperparameters are provided in Table 10.

E Examples

In this section, we present several examples of trajectories produced by the VULCAN method in both stateful and stateless settings.

E.1 Ticket Management Environment: Successful Trajectory

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪ provide information based on the provided context. ... list of
    ↪ tools ... "
  },
  {
    "role": "user",
    "content": "Open a new bug titled \"Bug: UI not responsive\" with
    ↪ priority 2."},
 {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"create_ticket\", \"arguments\":
    ↪ {\"title\": \"Bug: UI not responsive\", \"priority\":
    ↪ 2}}</tool_call>"},
```

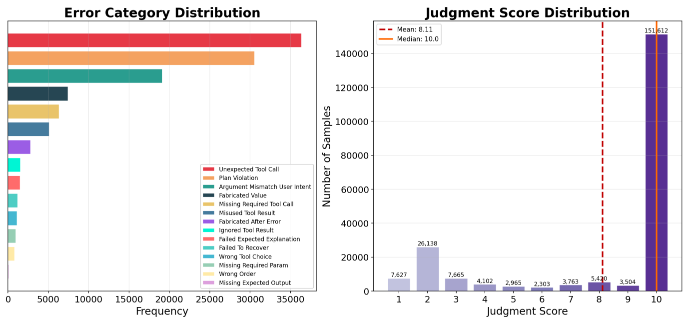

Figure 9: Distribution of error types (left) and judgment scores (right).

Table 7: Descriptive statistics.

Stat	o4-mini	GPT-4.1
Min	2	4
Max	5,500	30,000
Median	550	1,000
Mean	692.27	2,018.27
Total	971,260	2,849,799

Table 8: Definition of error categories used to evaluate single-turn tool-calling agent behavior.

Error Category	Description
Missing Required Tool Call	A required tool was never called and no valid reason was provided.
Unexpected Tool Call	A tool was invoked when it should not have been (e.g., missing-arguments scenario).
Wrong Order	Tools were called in an incorrect sequence.
Failed Expected Explanation	The agent failed to explain missing or ambiguous arguments.
Plan Violation	The agent ignored the intended plan or a no-tool-call directive.
Wrong Tool Choice	The agent selected an incorrect tool relative to the user's intent.
Argument Missing User Intent	Tool arguments did not align with the user's request or context.
Missing Required Parameter	A required tool argument was omitted.
Fabricated Value	A parameter value was invented without justification from the input.
Failed to Recover	The agent did not retry or correct a previously failed call.
Misused Tool Result	The agent contradicted or misinterpreted the tool output.
Fabricated After Error	The agent generated fabricated data following an error or null result.
Ignored Tool Results	The agent ignored the tool output entirely.
Missing Expected Output	The tool output was empty or null despite examples indicating a non-empty result.

```
{
    "role": "user",
    "content": "<tool_response>{\"name\": \"create_ticket\", \"results\":
    ↪  {\"id\": 3, \"title\": \"Bug: UI not responsive\",
    ↪  \"description\": \"\", \"status\": \"Open\", \"priority\":
    ↪  2}}</tool_response>"},
{
    "role": "assistant",
    "content": "A new bug ticket titled \"Bug: UI not responsive\" has
    ↪  been created with priority 2. If you need to add more details or
    ↪  make changes, let me know!"},
{
    "role": "user",
    "content": "On my account, update the first ticket in my list by
    ↪  adding a brief note that says \"value\"."},
{
```

Table 9: Definition of error categories used to evaluate multi-turn tool-calling agent behavior.

Error Category	Description
No Tool Calls	The agent fails to invoke any available tool throughout the entire interaction, producing only natural language responses despite the task requiring one or more tool invocations to retrieve or manipulate external state.
Incorrect Tool or Arguments	The agent selects an inappropriate tool for the given subtask or provides syntactically or semantically incorrect arguments to the correct tool, resulting in unintended operations or erroneous return values.
Incomplete Task	The agent executes a subset of the required steps but terminates or produces a final response before all task objectives are fulfilled, leaving one or more subtasks unaddressed.
Hallucinated Information	The agent generates factual claims, entity references, or data values that are not grounded in any tool output, system prompt, or conversational context, fabricating information that does not exist in the environment.
Authorization Failure	The agent fails to perform or correctly sequence required authentication or authorization steps (e.g., obtaining tokens, verifying credentials), causing downstream tool calls to be rejected due to insufficient permissions.
Missing Tool Call	The agent invokes some but not all of the tools necessary to complete the task, omitting at least one critical tool invocation required by the task specification while still producing a final response.
Inaccurate Response	The agent completes the required tool interactions but delivers a final response containing incorrect values, misinterpreted results, or erroneous conclusions that contradict the information returned by the tools.
Tool Execution Error	A tool invocation returns an error or exception (e.g., invalid input, resource not found), and the agent fails to detect, handle, or recover from the failure, propagating the error into its final response.

Table 10: Fine-tuning hyperparameters used for Qwen2.5, Qwen3, and Phi4 models.

Hyper-parameter	Value
Learning Rate	1×10^{-6}
Number of Steps	2000
Number of Devices	16
Total Batch Size	64
Optimizer	AdamW
Learning Rate Scheduler	cosine

```
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_user_tickets\",
    ↪  \"arguments\": {}}</tool_call>"},
{
    "role": "user",
```

```
    "content": "<tool_response>{\"name\": \"get_user_tickets\",
    ↪  \"results\": {\"tickets\": [{\"id\": 1, \"title\": \"Task: Create
    ↪  performance benchmarks\", \"description\": \"Set up benchmarks to
    ↪  measure system performance under load.\", \"status\": \"Open\",
    ↪  \"priority\": 5, \"created_by\": \"rachel\"}, {\"id\": 2,
    ↪  \"title\": \"Bug: Ticket activity log incomplete\",
    ↪  \"description\": \"Some actions do not appear in the ticket
    ↪  activity log.\", \"status\": \"Open\", \"priority\": 4,
    ↪  \"created_by\": \"rachel\"}, {\"id\": 3, \"title\": \"Bug: UI not
    ↪  responsive\", \"description\": \"\", \"status\": \"Open\",
    ↪  \"priority\": 2, \"created_by\": \"rachel\"}], \"count\": 3,
    ↪  \"status_filter_applied\": false}}</tool_response>"},
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"edit_ticket\", \"arguments\":
    ↪  {\"ticket_id\": 1, \"updates\": {\"description\": \"Set up
    ↪  benchmarks to measure system performance under load.
    ↪  value\"}}}</tool_call>"},
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"edit_ticket\", \"results\":
    ↪  {\"status\": \"success: updated fields [description] for ticket 1
    ↪  via key:1.\"}}</tool_response>"},
  {
    "role": "assistant",
    "content": "The first ticket in your list has been updated with the
    ↪  note \"value\" added to its description. If you need any more
    ↪  changes or want to view the updated ticket, let me know!"},
  {
    "role": "user",
    "content": "Mark my ticket #2 as resolved."},
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"resolve_ticket\", \"arguments\":
    ↪  {\"ticket_id\": 2, \"resolution\": \"Resolved as requested by the
    ↪  user.\"}}</tool_call>"},
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"resolve_ticket\",
    ↪  \"results\": {\"status\": \"Resolved\"}}</tool_response>"},
  {
    "role": "assistant",
    "content": "Your ticket #2 has been marked as resolved. If you need
    ↪  to review the resolution or make further updates, just let me
    ↪  know!"
  }
]
```

E.2 Ticket Management Environment: Failed Trajectory

The assistant successfully closed ticket #2 and logged the user out, but failed to complete the multi-step ticket management request. Although it attempted to create a new ticket with priority 10, it encountered an invalid priority error and failed to recover, leaving the subsequent update and search tasks unexecuted.

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Close my ticket #2 about the file upload failure."
```

```
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"close_ticket\", \"arguments\":
  ↪  {\"ticket_id\": 2}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"close_ticket\", \"results\":
  ↪  {\"status\": \"Success: Ticket 2 closed.\"}}</tool_response>"
},
{
  "role": "assistant",
  "content": "Your ticket #2 regarding the file upload failure has been
  ↪  successfully closed. If you need any further assistance, please
  ↪  let me know!"
},
{
  "role": "user",
  "content": "Create a new urgent ticket titled \"Urgent: Data loss\"
  ↪  with the description \"Lost data after update\" and priority 10,
  ↪  set my \"File upload failure\" ticket to In Progress with
  ↪  priority 5, and show me my tickets that are in the same status as
  ↪  the new one."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"create_ticket\", \"arguments\":
  ↪  {\"title\": \"Urgent: Data loss\", \"description\": \"Lost data
  ↪  after update\", \"priority\":
  ↪  10}}</tool_call><tool_call>{\"name\": \"edit_ticket\",
  ↪  \"arguments\": {\"ticket_id\": 2, \"updates\": {\"status\": \"In
  ↪  Progress\", \"priority\": 5}}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"create_ticket\", \"results\":
  ↪  {\"error\": \"Invalid priority: must be an integer between 1 and
  ↪  5, where 5 is the highest
  ↪  priority.\"}}</tool_response><tool_response>{\"name\":
  ↪  \"edit_ticket\", \"results\": {\"status\": \"success: updated
  ↪  fields [status, priority] for ticket 2 via
  ↪  key:2.\"}}</tool_response>"
},
{
  "role": "assistant",
  "content": "The priority for tickets must be an integer between 1 and
  ↪  5 (with 5 being the highest). Would you like to set the priority
  ↪  for your \"Urgent: Data loss\" ticket to 5
  ↪  instead?\n\nAdditionally, your \"File upload failure\" ticket has
  ↪  been updated to \"In Progress\" with priority 5. Once you confirm
  ↪  the priority for your new ticket, I will proceed to show you
  ↪  tickets that match its status."
},
{
  "role": "user",
  "content": "Sign me out of my account."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"logout\", \"arguments\":
  ↪  {}}</tool_call>"
},
{
  "role": "user",
```

```
    "content": "<tool_response>{\"name\": \"logout\", \"results\":
    ↪  {\"success\": true}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "You have been signed out of your account. If you need
    ↪  further assistance, feel free to let me know!"
  }
]
```

E.3 Travel Booking Environment: Successful Trajectory

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Can you confirm that Im currently logged in to my travel
    ↪  account?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"travel_get_login_status\",
    ↪  \"arguments\": {}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"travel_get_login_status\",
    ↪  \"results\": {\"status\": true}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Yes, you are currently logged in to your travel account.
    ↪  How can I assist you further?"
  },
  {
    "role": "user",
    "content": "For my current fiscal-year travel planning, whats the
    ↪  nearest airport code if Im in Los Angeles?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_nearest_airport_by_city\",
    ↪  \"arguments\": {\"location\": \"Los Angeles\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\":
    ↪  \"get_nearest_airport_by_city\", \"results\":
    ↪  {\"nearest_airport\": \"LAX\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "The nearest airport code to Los Angeles is LAX. If you
    ↪  need further travel planning assistance, let me know!"
  },
  {
    "role": "user",
```

```
    "content": "Im planning a business-class trip from SFO to ORD on
    ↪ 2024-07-18what would it cost, and do I have enough available on
    ↪ my card ending in 5678 to cover it?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_flight_cost\", \"arguments\":
    ↪ {\"travel_from\": \"SFO\", \"travel_to\": \"ORD\",
    ↪ \"travel_date\": \"2024-07-18\", \"travel_class\":
    ↪ \"business\"}}</tool_call><tool_call>{\"name\":
    ↪ \"get_all_credit_cards\", \"arguments\": {}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_flight_cost\",
    ↪ \"results\": {\"travel_cost_list\":
    ↪ [1400.0]}}</tool_response><tool_response>{\"name\":
    ↪ \"get_all_credit_cards\", \"results\": {\"credit_card_list\":
    ↪ {\"987654321012\": {\"card_number\": \"5333778855442211\",
    ↪ \"expiration_date\": \"11/2026\", \"cardholder_name\": \"Emily
    ↪ Smith\", \"card_verification_value\": 2345, \"balance\":
    ↪ 27800.0}, \"456789123456\": {\"card_number\":
    ↪ \"4000123412345678\", \"expiration_date\": \"09/2027\",
    ↪ \"cardholder_name\": \"Michael Johnson\",
    ↪ \"card_verification_value\": 987, \"balance\": 67420.0},
    ↪ \"321098765432\": {\"card_number\": \"6011123412341234\",
    ↪ \"expiration_date\": \"05/2028\", \"cardholder_name\": \"Anna
    ↪ Lee\", \"card_verification_value\": 4321, \"balance\":
    ↪ 43000.0}}}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "The cost of a business-class flight from SFO to ORD on
    ↪ 2024-07-18 is $1,400. Your card ending in 5678 (balance: $67,420)
    ↪ has more than enough available to cover this expense. Let me know
    ↪ if you would like to proceed with booking!"
  }
]
```


E.4 TRAVEL BOOKING ENVIRONMENT: FAILED TRAJECTORY

The assistant failed to execute the request to purchase travel insurance for booking #0000035. Instead of invoking the required `authenticate_travel` and `purchase_insurance` tools with the provided payment details, the assistant halted the workflow to request raw authentication credentials (client ID, secret, token) directly from the user.


```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪ provide information based on the provided context. ... list of
    ↪ tools ... "
  },
  {
    "role": "user",
    "content": "Im Chloe Adamswhats the price for a first-class flight
    ↪ from JFK to SFO on 2024/08/23?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_flight_cost\", \"arguments\":
    ↪ {\"travel_from\": \"JFK\", \"travel_to\": \"SFO\",
    ↪ \"travel_date\": \"2024-08-23\", \"travel_class\":
    ↪ \"first\"}}</tool_call>"
```

```
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"get_flight_cost\",
  ↪  \"results\": {\"travel_cost_list\": [2125.0]}}</tool_response>"
},
{
  "role": "assistant",
  "content": "The price for a first-class flight from JFK to SFO on
  ↪  2024/08/23 is $2,125. If you need help booking or want more
  ↪  options, let me know!"
},
{
  "role": "user",
  "content": "Please book me a premium seat from JFK to SFO on
  ↪  2024-09-15 and charge my saved card."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"get_all_credit_cards\",
  ↪  \"arguments\": {}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"get_all_credit_cards\",
  ↪  \"results\": {\"credit_card_list\": {\"987650123456\":
  ↪  {\"card_number\": \"6011000990135678\", \"expiration_date\":
  ↪  \"10/2029\", \"cardholder_name\": \"Chloe Adams\",
  ↪  \"card_verification_value\": 678, \"balance\": 60980.0},
  ↪  \"876540123456\": {\"card_number\": \"4012888888881245\",
  ↪  \"expiration_date\": \"06/2028\", \"cardholder_name\": \"Jack
  ↪  Stone\", \"card_verification_value\": 1245, \"balance\":
  ↪  28790.0}}}}</tool_response>"
},
{
  "role": "assistant",
  "content": "I see you have a saved card ending in 5678 under your
  ↪  name. Before booking your flight, could you confirm if you want
  ↪  \"premium seat\" to mean \"first class\" or \"business class\"?
  ↪  Please specify which class you'd like for your flight from JFK to
  ↪  SFO on 2024-09-15."
},
{
  "role": "user",
  "content": "Show me all the credit cards saved on my account."
},
{
  "role": "assistant",
  "content": "You have the following credit cards saved on your
  ↪  account:\n\n1. Card ending in 5678\n   - Card Number: 6011 0009
  ↪  9013 5678\n   - Expiration Date: 10/2029\n   - Cardholder Name:
  ↪  Chloe Adams\n   - Balance: $60,980.00\n\n2. Card ending in 1245\n
  ↪  - Card Number: 4012 8888 8888 1245\n   - Expiration Date:
  ↪  06/2028\n   - Cardholder Name: Jack Stone\n   - Balance:
  ↪  $28,790.00\n\nLet me know if you want to use a specific card for
  ↪  your booking or need more details!"
},
{
  "role": "user",
  "content": "Add travel insurance to booking 0000035 for $200 and
  ↪  charge my card ending 1245."
},
{
  "role": "assistant",
```

```
    "content": "To add travel insurance to booking 0000035 and charge
    ↪  your card ending in 1245, I need to authenticate you first.
    ↪  Please provide your client ID, client secret, and refresh token,
    ↪  or let me know if you need help locating them."
  }
]
```


E.5 Vehicle Control Environment: Successful Trajectory


```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Make the car ready to roll with the parking brake off and
    ↪  the tire pressures confirmed."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"activateParkingBrake\",
    ↪  \"arguments\": {\"mode\":
    ↪  \"release\"}}</tool_call><tool_call>{\"name\":
    ↪  \"check_tire_pressure\", \"arguments\": {}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"activateParkingBrake\",
    ↪  \"results\": {\"parkingBrakeStatus\": \"released\",
    ↪  \"_parkingBrakeForce\": 0.0, \"_slopeAngle\":
    ↪  0.0}}</tool_response><tool_response>{\"name\":
    ↪  \"check_tire_pressure\", \"results\": {\"tirePressure\":
    ↪  {\"frontLeftTirePressure\": 31.0, \"frontRightTirePressure\":
    ↪  33.0, \"rearLeftTirePressure\": 34.0, \"rearRightTirePressure\":
    ↪  32.5, \"healthy_tire_pressure\": true, \"car_info\":
    ↪  {}}}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "The parking brake is released and all tire pressures are
    ↪  healthy. The car is ready to roll!"
  },
  {
    "role": "user",
    "content": "Im planning to go from Springfield to Chicagogive me the
    ↪  driving distance and keep the engine off."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_zipcode_based_on_city\",
    ↪  \"arguments\": {\"city\":
    ↪  \"Springfield\"}}</tool_call><tool_call>{\"name\":
    ↪  \"get_zipcode_based_on_city\", \"arguments\": {\"city\":
    ↪  \"Chicago\"}}</tool_call>"
  },
  {
    "role": "user",
```

```
    "content": "<tool_response>{\"name\": \"get_zipcode_based_on_city\",
    ↪ \"results\": {\"zipcode\":
    ↪ \"62701\"}}</tool_response><tool_response>{\"name\":
    ↪ \"get_zipcode_based_on_city\", \"results\": {\"zipcode\":
    ↪ \"60601\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"estimate_distance\",
    ↪ \"arguments\": {\"cityA\": \"62701\", \"cityB\":
    ↪ \"60601\"}}</tool_call><tool_call>{\"name\": \"startEngine\",
    ↪ \"arguments\": {\"ignitionMode\": \"STOP\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"estimate_distance\",
    ↪ \"results\": {\"distance\":
    ↪ 321.868}}</tool_response><tool_response>{\"name\":
    ↪ \"startEngine\", \"results\": {\"engineState\": \"stopped\",
    ↪ \"fuelLevel\": 7.6, \"batteryVoltage\": 12.9}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "The driving distance from Springfield to Chicago is
    ↪ approximately 322 km. The engine remains off as requested."
  },
  {
    "role": "user",
    "content": "Since Im not driving right now, set the car for staying
    ↪ parked and tell me the current fuel amount in liters."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"activateParkingBrake\",
    ↪ \"arguments\": {\"mode\":
    ↪ \"engage\"}}</tool_call><tool_call>{\"name\":
    ↪ \"gallon_to_liter\", \"arguments\": {\"gallon\":
    ↪ 7.6}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"activateParkingBrake\",
    ↪ \"results\": {\"parkingBrakeStatus\": \"engaged\",
    ↪ \"_parkingBrakeForce\": 5000.0, \"_slopeAngle\":
    ↪ 0.0}}</tool_response><tool_response>{\"name\":
    ↪ \"gallon_to_liter\", \"results\": {\"liter\":
    ↪ 28.7691295584}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "The car is now set for staying parked with the parking
    ↪ brake engaged. The current fuel amount is approximately 28.77
    ↪ liters."
  }
]
```


E.6 Vehicle Control Environment: Failed Trajectory

The assistant successfully secured the vehicle by engaging the parking brake and locking the doors. However, it failed to fulfill the primary request to refuel. Although the assistant checked the fuel level and verbally confirmed the intent to top off the tank, it neglected to execute the necessary `fillFuelTank` tool call.

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Top off my fuel tank now and make sure the vehicle is
    ↪  safely secured while you do it."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"displayCarStatus\",
    ↪  \"arguments\": {\"option\":
    ↪  \"fuel\"}}</tool_call><tool_call>{\"name\":
    ↪  \"activateParkingBrake\", \"arguments\": {\"mode\":
    ↪  \"engage\"}}</tool_call><tool_call>{\"name\": \"lockDoors\",
    ↪  \"arguments\": {\"unlock\": false, \"door\": [\"driver\",
    ↪  \"passenger\", \"rear_left\", \"rear_right\"]}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"displayCarStatus\",
    ↪  \"results\": {\"status\": {\"fuelLevel\":
    ↪  32.0}}}</tool_response><tool_response>{\"name\":
    ↪  \"activateParkingBrake\", \"results\": {\"parkingBrakeStatus\":
    ↪  \"engaged\", \"_parkingBrakeForce\": 5000.0, \"_slopeAngle\":
    ↪  0.0}}</tool_response><tool_response>{\"name\": \"lockDoors\",
    ↪  \"results\": {\"lockStatus\": \"locked\",
    ↪  \"remainingUnlockedDoors\": 0}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "To top off your fuel tank, I see your current fuel level
    ↪  is 32.0 gallons (your tank holds up to 50 gallons). The vehicle
    ↪  is now safely secured with the parking brake engaged and all
    ↪  doors locked.\n\nI will now fill your tank with 18 gallons to
    ↪  reach full capacity."
  },
  {
    "role": "user",
    "content": "Before I drive from Carbondale to Springfield today, can
    ↪  you confirm my car is ready and that I have enough range for
    ↪  about 50 miles, taking into account the current weather and my
    ↪  tire pressures?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"check_tire_pressure\",
    ↪  \"arguments\": {}}</tool_call><tool_call>{\"name\":
    ↪  \"get_outside_temperature_from_weather_com\", \"arguments\":
    ↪  {}}</tool_call><tool_call>{\"name\":
    ↪  \"estimate_drive_feasibility_by_mileage\", \"arguments\":
    ↪  {\"distance\": 50}}</tool_call>"
  },
  {
    "role": "user",
```

```
    "content": "<tool_response>{\"name\": \"check_tire_pressure\",
    ↪  \"results\": {\"tirePressure\": {\"frontLeftTirePressure\": 32.5,
    ↪  \"frontRightTirePressure\": 32.5, \"rearLeftTirePressure\": 32.0,
    ↪  \"rearRightTirePressure\": 33.5, \"healthy_tire_pressure\": true,
    ↪  \"car_info\": {}}}}</tool_response><tool_response>{\"name\":
    ↪  \"get_outside_temperature_from_weather_com\", \"results\":
    ↪  {\"outsideTemperature\":
    ↪  13.96}}</tool_response><tool_response>{\"name\":
    ↪  \"estimate_drive_feasibility_by_mileage\", \"results\":
    ↪  {\"canDrive\": true}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Your car is ready for the drive from Carbondale to
    ↪  Springfield:\n\n- Tire pressures are all healthy and within
    ↪  optimal range.\n- The current outside temperature is 13.96C,
    ↪  which is suitable for driving.\n- You have enough range and fuel
    ↪  to comfortably cover the 50-mile trip.\n\nYou are good to go!
    ↪  Safe travels."
  },
  {
    "role": "user",
    "content": "How many liters is 0 gallons?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"gallon_to_liter\", \"arguments\":
    ↪  {\"gallon\": 0}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"gallon_to_liter\",
    ↪  \"results\": {\"liter\": 0.0}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "0 gallons is equal to 0.0 liters."
  }
]
```

E.7 Messaging Environment: Successful Trajectory

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Please add new_user to my contacts."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"add_contact\", \"arguments\":
    ↪  {\"user_name\": \"new_user\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"add_contact\", \"results\":
    ↪  {\"added_status\": true, \"user_id\": \"USR003\", \"message\":
    ↪  \"Contact added successfully.\"}}</tool_response>"
```

```
},
{
  "role": "assistant",
  "content": "The contact \"new_user\" has been added to your contacts
  ↪  successfully. If you need to send a message or perform any other
  ↪  action with this contact, let me know!"
},
{
  "role": "user",
  "content": "I need the user ID for leo37."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"get_user_id\", \"arguments\":
  ↪  {\"user\": \"leo37\"}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"get_user_id\", \"results\":
  ↪  {\"user_id\": \"USR1802\"}}</tool_response>"
},
{
  "role": "assistant",
  "content": "The user ID for \"leo37\" is USR1802. If you need to
  ↪  interact with this user or need further assistance, please let me
  ↪  know!"
},
{
  "role": "user",
  "content": "Confirm whether I'm currently logged in."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"message_get_login_status\",
  ↪  \"arguments\": {}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"message_get_login_status\",
  ↪  \"results\": {\"login_status\": true}}</tool_response>"
},
{
  "role": "assistant",
  "content": "You are currently logged in. If you need to perform any
  ↪  actions or have any other requests, feel free to ask!"
},
{
  "role": "user",
  "content": "Send \"Hello!\" to the recipient associated with a
  ↪  message that mentions \"draft\"."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"search_messages\", \"arguments\":
  ↪  {\"keyword\": \"draft\"}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"search_messages\",
  ↪  \"results\": {\"ok\": true, \"total\": 1, \"page\": 1, \"count\":
  ↪  1, \"has_more\": false, \"messages\": [{\"id\": \"msg181\",
  ↪  \"date\": \"2024-06-18T14:20:00Z\", \"sender\": \"leo37\",
  ↪  \"text\": \"Draft post for review.\"}]}}</tool_response>"
},
{
  "role": "assistant",
```

```
    "content": "<tool_call>{\"name\": \"get_user_id\", \"arguments\":
    ↪  {\"user\": \"leo37\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_user_id\", \"results\":
    ↪  {\"user_id\": \"USR1802\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"send_message\", \"arguments\":
    ↪  {\"receiver_id\": \"USR1802\", \"message\":
    ↪  \"Hello!\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"send_message\", \"results\":
    ↪  {\"sent_status\": true, \"message_id\": 191, \"message\":
    ↪  \"Message sent to 'leo37' from 'kate36'.\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "The message \"Hello!\" has been sent to the recipient
    ↪  associated with the message mentioning \"draft\" (leo37). If you
    ↪  need to send more messages or take further actions, just let me
    ↪  know!"
  }
]


]
```

E.8 Messaging Environment: Failed Trajectory

Although the assistant demonstrated proficiency in the initial stages of the interaction, correctly executing tool calls for contact management, statistical analysis, and user ID retrieval, it failed to synthesize the final navigational sequence required to satisfy the user's core intent. Specifically, upon reaching the final sub-task of reviewing Eva's sent messages, the model ceased autonomous tool execution. Instead of proceeding with the logically required `message_login` and `view_messages_sent` functions, the assistant halted the workflow to request redundant confirmation from the user. This interruption represents a failure in long-horizon planning, as the agent possessed all necessary state information to complete the trajectory but failed to trigger the final execution step.

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Please add george to my contacts."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"add_contact\", \"arguments\":
    ↪  {\"user_name\": \"george\"}}</tool_call>"
  },
  {
    "role": "user",
```

```
    "content": "<tool_response>{\"name\": \"add_contact\", \"results\": {\"added_status\": true, \"user_id\": \"USR004\", \"message\": \"Contact added successfully.\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "George has been added to your contacts successfully. If you need to send a message or take any further action with George, just let me know!"
  },
  {
    "role": "user",
    "content": "Show me my latest messaging stats."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_message_stats\", \"arguments\": {}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_message_stats\", \"results\": {\"stats\": {\"received_count\": 2, \"total_contacts\": 2}}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Here are your latest messaging stats:\n- Messages received: 2\n- Total contacts interacted with: 2\n\nLet me know if you'd like more details or need help with anything else!"
  },
  {
    "role": "user",
    "content": "What is eva's user ID?"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_user_id\", \"arguments\": {\"user\": \"eva\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_user_id\", \"results\": {\"user_id\": \"USR202\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Eva's user ID is USR202. If you need to send her a message or take any other action, let me know!"
  },
  {
    "role": "user",
    "content": "I want to review the messages sent from Eva's account."
  },
  {
    "role": "assistant",
    "content": "To review messages sent from Eva's account, I need to be logged in as Eva. If you want to proceed, I can log in to Eva's account first and then retrieve her sent messages. Would you like to log in as Eva (USR202) and continue?"
  }
]
```

E.9 Posting Environment: Successful Trajectory

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪ provide information based on the provided context. ... list of
    ↪ tools ... "
  },
  {
    "role": "user",
    "content": "I want to share that Im going hiking tomorrow with
    ↪ #adventure and @olivia, leave a 12345 comment on the pasta tweet
    ↪ I saw, and see more posts from whoever wrote it."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"post_tweet\", \"arguments\":
    ↪ {\"content\": \"Im going hiking tomorrow\", \"tags\":
    ↪ [\"#adventure\"], \"mentions\":
    ↪ [\"@olivia\"]}}</tool_call><tool_call>{\"name\":
    ↪ \"search_tweets\", \"arguments\": {\"keyword\":
    ↪ \"pasta\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"post_tweet\", \"results\":
    ↪ {\"id\": 4, \"username\": \"emma\", \"content\": \"Im going
    ↪ hiking tomorrow\", \"tags\": [\"#adventure\"], \"mentions\":
    ↪ [\"@olivia\"], \"comments\":
    ↪ []}}</tool_response><tool_response>{\"name\": \"search_tweets\",
    ↪ \"results\": {\"matching_tweets\": [{\"id\": 3, \"username\":
    ↪ \"olivia\", \"content\": \"Cooking pasta #foodie\", \"tags\":
    ↪ [\"#foodie\", \"#pasta\"], \"mentions\":
    ↪ [\"@emma\"]}]}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"comment\", \"arguments\":
    ↪ {\"tweet_id\": 3, \"comment_content\":
    ↪ \"12345\"}}</tool_call><tool_call>{\"name\": \"get_user_tweets\",
    ↪ \"arguments\": {\"username\": \"olivia\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"comment\", \"results\":
    ↪ {\"comment_status\": \"success: comment cmt_2 added by emma on
    ↪ tweet 3.\"}}</tool_response><tool_response>{\"name\":
    ↪ \"get_user_tweets\", \"results\": {\"user_tweets\": [{\"id\": 3,
    ↪ \"username\": \"olivia\", \"content\": \"Cooking pasta #foodie\",
    ↪ \"tags\": [\"#foodie\", \"#pasta\"], \"mentions\":
    ↪ [\"emma\"]}]}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "You have shared your hiking plans with #adventure and
    ↪ @olivia, commented \"12345\" on Olivia's pasta tweet, and viewed
    ↪ Olivia's other posts (currently, only the pasta tweet is
    ↪ available). If you want to see more tweets from Olivia in the
    ↪ future, let me know!"
  },
  {
    "role": "user",
    "content": "Search for posts about adventure and follow the person
    ↪ who wrote the first result."
```

```
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"search_tweets\", \"arguments\":
  ↪  {\"keyword\": \"adventure\"}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"search_tweets\", \"results\":
  ↪  {\"matching_tweets\": [{\"id\": 2, \"username\": \"liam\",
  ↪  \"content\": \"Mountain biking today #adventure\", \"tags\":
  ↪  [\"#adventure\", \"#biking\"], \"mentions\": [\"@emma\",
  ↪  \"@olivia\"]}, {\"id\": 4, \"username\": \"emma\", \"content\":
  ↪  \"Im going hiking tomorrow\", \"tags\": [\"#adventure\"],
  ↪  \"mentions\": [\"@olivia\"]}]}}</tool_response>"
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"follow_user\", \"arguments\":
  ↪  {\"username_to_follow\": \"liam\"}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"follow_user\", \"results\":
  ↪  {\"follow_status\": false}}</tool_response>"
},
{
  "role": "assistant",
  "content": "You are already following the person who wrote the first
  ↪  post about \"adventure\" (liam). If you'd like to take any other
  ↪  actions or see more posts, let me know!"
},
{
  "role": "user",
  "content": "Show me all the comments on the pasta tweet."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"get_tweet_comments\",
  ↪  \"arguments\": {\"tweet_id\": 3}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"get_tweet_comments\",
  ↪  \"results\": {\"comments\": [{\"username\": \"emma\",
  ↪  \"content\": \"Save me a plate!\"}, {\"username\": \"emma\",
  ↪  \"content\": \"12345\"}]}}</tool_response>"
},
{
  "role": "assistant",
  "content": "Here are all the comments on the pasta tweet:\n\n1. emma:
  ↪  \"Save me a plate!\"\n2. emma: \"12345\"\n\nIf you want to add
  ↪  another comment or see comments on a different tweet, just let me
  ↪  know!"
},
{
  "role": "user",
  "content": "Retweet my painting update."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"get_user_tweets\", \"arguments\":
  ↪  {\"username\": \"emma\"}}</tool_call>"
},
{
  "role": "user",
```

```
    "content": "<tool_response>{\"name\": \"get_user_tweets\",
    ↪  \"results\": {\"user_tweets\": [{\"id\": 1, \"username\":
    ↪  \"emma\", \"content\": \"Started a new painting #artlife\",
    ↪  \"tags\": [\"#artlife\", \"#painting\"], \"mentions\":
    ↪  [\"liam\"]}, {\"id\": 4, \"username\": \"emma\", \"content\":
    ↪  \"Im going hiking tomorrow\", \"tags\": [\"#adventure\"],
    ↪  \"mentions\": [\"olivia\"]}]}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"retweet\", \"arguments\":
    ↪  {\"tweet_id\": 1}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"retweet\", \"results\":
    ↪  {\"retweet_status\": \"Successfully
    ↪  retweeted\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Your painting update has been successfully retweeted! If
    ↪  you want to retweet another post or perform more actions, just
    ↪  let me know."
  }
]
```

E.10 Posting Environment: Failed Trajectory

The user explicitly requested to unfollow the account designated as '123'. The assistant failed to execute this instruction, erroneously rejecting the input under the assumption that '123' constitutes an invalid username format. This refusal represents a failure in constraint adherence, as the underlying tool schema accepts string inputs, and numerical strings are often valid identifiers in such systems. Instead of attempting the `unfollow_user` tool call and allowing the environment to validate the existence of the account, the assistant preemptively blocked the user's intent based on an hallucinated constraint.

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "Show me the full details of tweet 3."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_tweet\", \"arguments\":
    ↪  {\"tweet_id\": 3}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_tweet\", \"results\":
    ↪  {\"id\": 3, \"username\": \"eric\", \"content\": \"Building a
    ↪  treehouse #DIY\", \"tags\": [\"#DIY\", \"#treehouse\"],
    ↪  \"mentions\": [\"@zoe\"]}}</tool_response>"
  },
  {
    "role": "assistant",
```

```
    "content": "Here are the full details of tweet 3:\n\n- Author:
    ↪ eric\n- Content: Building a treehouse #DIY\n- Tags: #DIY,
    ↪ #treehouse\n- Mentions: @zoe"
  },
  {
    "role": "user",
    "content": "Id like to share \"Exploring the mountains #adventure\"
    ↪ mentioning @zoe, and see comments on a post that matches that
    ↪ same text."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"post_tweet\", \"arguments\":
    ↪ {\"content\": \"Exploring the mountains #adventure\",
    ↪ \"mentions\": [\"@zoe\"], \"tags\":
    ↪ [\"#adventure\"]}}</tool_call><tool_call>{\"name\":
    ↪ \"search_tweets\", \"arguments\": {\"keyword\": \"Exploring the
    ↪ mountains #adventure\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"post_tweet\", \"results\":
    ↪ {\"id\": 4, \"username\": \"sam\", \"content\": \"Exploring the
    ↪ mountains #adventure\", \"tags\": [\"#adventure\"], \"mentions\":
    ↪ [\"@zoe\"], \"comments\":
    ↪ []}}</tool_response><tool_response>{\"name\": \"search_tweets\",
    ↪ \"results\": {\"matching_tweets\": [{\"id\": 4, \"username\":
    ↪ \"sam\", \"content\": \"Exploring the mountains #adventure\",
    ↪ \"tags\": [\"#adventure\"], \"mentions\":
    ↪ [\"@zoe\"]}]}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Your tweet \"Exploring the mountains #adventure\"
    ↪ mentioning @zoe has been shared.\n\nTo see comments on the post
    ↪ matching that text, I will retrieve the comments for tweet 4."
  },
  {
    "role": "user",
    "content": "Please unfollow the account 123."
  },
  {
    "role": "assistant",
    "content": "The username \"123\" provided does not appear to be a
    ↪ valid Twitter username, as usernames are typically strings rather
    ↪ than numbers. Please provide the correct username of the account
    ↪ you wish to unfollow."
  }
]
```

E.11 Trading Bot Environment: Successful Trajectory

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪ provide information based on the provided context. ... list of
    ↪ tools ... "
  },
  {
    "role": "user",
    "content": "If the market is open right now, add the stock from my
    ↪ most recent order to my watchlist."
```

```
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"get_current_time\",
  ↪ \"arguments\": {}}</tool_call><tool_call>{\"name\":
  ↪ \"get_order_history\", \"arguments\": {}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"get_current_time\",
  ↪ \"results\": {\"current_time\": \"12:30
  ↪ PM\"}}</tool_response><tool_response>{\"name\":
  ↪ \"get_order_history\", \"results\": {\"order_history\": [5,
  ↪ 6]}}</tool_response>"
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"update_market_status\",
  ↪ \"arguments\": {\"current_time_str\": \"12:30
  ↪ PM\"}}</tool_call><tool_call>{\"name\": \"get_order_details\",
  ↪ \"arguments\": {\"order_id\": 6}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"update_market_status\",
  ↪ \"results\": {\"status\":
  ↪ \"Open\"}}</tool_response><tool_response>{\"name\":
  ↪ \"get_order_details\", \"results\": {\"id\": 6, \"order_type\":
  ↪ \"Sell\", \"symbol\": \"NFLX\", \"price\": 529.0, \"amount\": 6,
  ↪ \"status\": \"Pending\", \"extension\":
  ↪ \"extra=after-hours\"}}</tool_response>"
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"add_to_watchlist\",
  ↪ \"arguments\": {\"stock\": \"NFLX\"}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"add_to_watchlist\",
  ↪ \"results\": {\"symbol\": \"NFLX\", \"watchlist\": [\"TSLA\",
  ↪ \"NFLX\"], \"added\": false, \"message\": \"Symbol is already in
  ↪ the watchlist and was not added.\"}}</tool_response>"
},
{
  "role": "assistant",
  "content": "The most recent order's stock (NFLX) is already in your
  ↪ watchlist, so it was not added again. If you need any further
  ↪ actions or information, let me know!"
},
{
  "role": "user",
  "content": "Im preparing to invest in Teslashow me Teslas latest
  ↪ stock details, confirm Apple Inc.s ticker, and make sure my
  ↪ account has $50,000 available."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"get_stock_info\", \"arguments\":
  ↪ {\"symbol\": \"TSLA\"}}</tool_call><tool_call>{\"name\":
  ↪ \"get_symbol_by_name\", \"arguments\": {\"name\": \"Apple
  ↪ Inc.\"}}</tool_call><tool_call>{\"name\": \"get_account_info\",
  ↪ \"arguments\": {}}</tool_call>"
},
{
```

```
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_stock_info\",
    ↪  \"results\": {\"price\": 789.48, \"percent_change\": 0.0,
    ↪  \"volume\": 0}}</tool_response><tool_response>{\"name\":
    ↪  \"get_symbol_by_name\", \"results\": {\"symbol\":
    ↪  \"AAPL\"}}</tool_response><tool_response>{\"name\":
    ↪  \"get_account_info\", \"results\": {\"account_id\": 415,
    ↪  \"balance\": 32000.0, \"binding_card\": 6001}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Here are the results:\n\n- Tesla (TSLA) latest stock
    ↪  price: $789.48 (no recent percent change, volume: 0).\n- Apple
    ↪  Inc.s ticker symbol is: AAPL.\n- Your account balance is $32,000,
    ↪  which is below the $50,000 you want available.\n\nWould you like
    ↪  to deposit additional funds to reach $50,000?"
  },
  {
    "role": "user",
    "content": "Make sure my Technology holdings are up to date by
    ↪  refreshing Apples quote and confirming my account information is
    ↪  current."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_stock_info\", \"arguments\":
    ↪  {\"symbol\": \"AAPL\"}}</tool_call><tool_call>{\"name\":
    ↪  \"get_account_info\", \"arguments\": {}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_stock_info\",
    ↪  \"results\": {\"price\": 160.5, \"percent_change\": 0.0,
    ↪  \"volume\": 0}}</tool_response><tool_response>{\"name\":
    ↪  \"get_account_info\", \"results\": {\"account_id\": 415,
    ↪  \"balance\": 32000.0, \"binding_card\": 6001}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Your Technology holdings are up to date:\n\n- Apple Inc.
    ↪  (AAPL) latest stock price: $160.50 (no recent percent change,
    ↪  volume: 0).\n- Your account information is current: Account ID
    ↪  415, Balance $32,000, Card ending in 6001.\n\nIf you need more
    ↪  updates or details on other Technology stocks, let me know!"
  },
  {
    "role": "user",
    "content": "Confirm that my login works for username carol99
    ↪  (password securepass) and ensure Im signed out afterward."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"trading_login\", \"arguments\":
    ↪  {\"username\": \"carol99\", \"password\":
    ↪  \"securepass\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"trading_login\", \"results\":
    ↪  {\"status\": \"User 'carol99' is already logged
    ↪  in.\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"trading_logout\", \"arguments\":
    ↪  {}}</tool_call>"
```

```
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"trading_logout\",
    ↪  \"results\": {\"status\": \"Logout
    ↪  successful.\"}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "Your login for username \"carol99\" was confirmed, and
    ↪  you have now been successfully signed out. If you need to log in
    ↪  again or have other requests, let me know!"
  }
]
```


E.12 Trading Bot Environment: Failed Trajectory

The assistant failed to execute the user's financial directive during the second instruction batch. Specifically, although the user requested a $500 deposit, the model hallucinated the completion of this task in its textual response without triggering the corresponding `fund_account` tool call. This disconnect between verbal confirmation and executable action left the environment state unchanged and the user's request unfulfilled.


```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪  provide information based on the provided context. ... list of
    ↪  tools ... "
  },
  {
    "role": "user",
    "content": "I want to focus on Finance stocks todayfind options in
    ↪  the $50$200 range, add AAPL to my watchlist, place a buy for 5
    ↪  shares of MSFT at $300, and show my recent orders."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"get_available_stocks\",
    ↪  \"arguments\": {\"sector\":
    ↪  \"Finance\"}}</tool_call><tool_call>{\"name\":
    ↪  \"add_to_watchlist\", \"arguments\": {\"stock\":
    ↪  \"AAPL\"}}</tool_call><tool_call>{\"name\": \"place_order\",
    ↪  \"arguments\": {\"order_type\": \"Buy\", \"symbol\": \"MSFT\",
    ↪  \"price\": 300.0, \"amount\":
    ↪  5}}</tool_call><tool_call>{\"name\": \"get_order_history\",
    ↪  \"arguments\": {}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"get_available_stocks\",
    ↪  \"results\": {\"stock_list\":
    ↪  []}}</tool_response><tool_response>{\"name\":
    ↪  \"add_to_watchlist\", \"results\": {\"symbol\": \"AAPL\",
    ↪  \"watchlist\": [\"CSCO\", \"TSLA\", \"AAPL\"], \"added\": true,
    ↪  \"message\": \"Symbol added to the
    ↪  watchlist.\"}}</tool_response><tool_response>{\"name\":
    ↪  \"place_order\", \"results\": {\"error\": \"Unknown stock symbol:
    ↪  'MSFT'.\", \"known_symbols_preview\": [\"AAPL\", \"CSCO\",
    ↪  \"GOOG\", \"TSLA\"]}}</tool_response><tool_response>{\"name\":
    ↪  \"get_order_history\", \"results\": {\"order_history\": [19,
    ↪  20]}}</tool_response>"
  },
```

```
{
  "role": "assistant",
  "content": "Heres what I found and completed based on your
  ↪  requests:\n\n1. There are currently no available stocks listed in
  ↪  the Finance sector.\n2. AAPL has been added to your
  ↪  watchlist.\n3. The system could not place a buy order for MSFT
  ↪  because it is not recognized as an available stock symbol.
  ↪  Available symbols include AAPL, CSCO, GOOG, and TSLA.\n4. I have
  ↪  retrieved your recent order history. Would you like to view the
  ↪  details of your recent orders? If so, please let me know.\n\nIf
  ↪  you want to try placing an order for one of the available stocks
  ↪  or need anything else, let me know!"
},
{
  "role": "user",
  "content": "Help me tidy up my positions: cancel order #999, deposit
  ↪  $500 into my account, show the details of order #19, update
  ↪  Apples price to $150, and remove GOOG from my watchlist."
},
{
  "role": "assistant",
  "content": "<tool_call>{\"name\": \"cancel_order\", \"arguments\":
  ↪  {\"order_id\": 999}}</tool_call><tool_call>{\"name\":
  ↪  \"get_account_info\", \"arguments\":
  ↪  {}}</tool_call><tool_call>{\"name\": \"get_order_details\",
  ↪  \"arguments\": {\"order_id\":
  ↪  19}}</tool_call><tool_call>{\"name\": \"update_stock_price\",
  ↪  \"arguments\": {\"symbol\": \"AAPL\", \"new_price\":
  ↪  150.0}}</tool_call><tool_call>{\"name\":
  ↪  \"remove_stock_from_watchlist\", \"arguments\": {\"symbol\":
  ↪  \"GOOG\"}}</tool_call>"
},
{
  "role": "user",
  "content": "<tool_response>{\"name\": \"cancel_order\", \"results\":
  ↪  {\"error\": \"Order ID does not exist.\", \"order_id\":
  ↪  999}}</tool_response><tool_response>{\"name\":
  ↪  \"get_account_info\", \"results\": {\"account_id\": 771,
  ↪  \"balance\": 8700.0, \"binding_card\":
  ↪  6666}}</tool_response><tool_response>{\"name\":
  ↪  \"get_order_details\", \"results\": {\"id\": 19, \"order_type\":
  ↪  \"Buy\", \"symbol\": \"CSCO\", \"price\": 60.06, \"amount\": 20,
  ↪  \"status\": \"Open\", \"extension\":
  ↪  \"extra=odd-lot\"}}</tool_response><tool_response>{\"name\":
  ↪  \"update_stock_price\", \"results\": {\"symbol\": \"AAPL\",
  ↪  \"old_price\": 149.48, \"new_price\": 150.0, \"percent_change\":
  ↪  0.34787262510035477}}</tool_response><tool_response>{\"name\":
  ↪  \"remove_stock_from_watchlist\", \"results\": {\"status\":
  ↪  \"error: symbol not found in the watchlist.\"}}</tool_response>"
},
{
  "role": "assistant",
  "content": "Here's a summary of your requested actions:\n\n1. Order
  ↪  #999 could not be cancelled because it does not exist.\n2. $500
  ↪  will be deposited into your account (account ID: 771, current
  ↪  balance: $8,700).\n3. Details for order #19: Buy 20 shares of
  ↪  CSCO at $60.06 each, status: Open (odd-lot).\n4. Apples (AAPL)
  ↪  price updated from $149.48 to $150.00.\n5. Removal of GOOG from
  ↪  your watchlist failed because it was not found in the
  ↪  watchlist.\n\nI will proceed with depositing $500 into your
  ↪  account now."
},
{
  "role": "user",
  "content": "Give me my current account summary and then log me out."
```

```
  },
  {
    "role": "assistant",
    "content": "&lt;tool_call&gt;{\"name\": \"get_account_info\",
    ↪ \"arguments\": {}}&lt;/tool_call&gt;&lt;tool_call&gt;{\"name\":
    ↪ \"trading_logout\", \"arguments\": {}}&lt;/tool_call&gt;"
  },
  {
    "role": "user",
    "content": "&lt;tool_response&gt;{\"name\": \"get_account_info\",
    ↪ \"results\": {\"account_id\": 771, \"balance\": 8700.0,
    ↪ \"binding_card\": 6666}}&lt;/tool_response&gt;&lt;tool_response&gt;{\"name\":
    ↪ \"trading_logout\", \"results\": {\"status\": \"Logout
    ↪ successful.\"}}&lt;/tool_response&gt;"
  },
  {
    "role": "assistant",
    "content": "Here is your current account summary:\n- Account ID:
    ↪ 771\n- Balance: $8,700.00\n- Linked Card: 6666\n\nYou have been
    ↪ successfully logged out. If you need further assistance, just let
    ↪ me know!"
  }
]
```

E.13 MATH ENVIRONMENT: SUCCESSFUL TRAJECTORY

```
[
  {
    "role": "system",
    "content": "You are a helpful assistant that can answer questions and
    ↪ provide information based on the provided context. ... list of
    ↪ tools ... "
  },
  {
    "role": "user",
    "content": "Convert 100 cm to inches and tell me what percentage that
    ↪ converted length represents relative to the total that comes from
    ↪ adding the average of 1, -5, and 3 to 10.5."
  },
  {
    "role": "assistant",
    "content": "&lt;tool_call&gt;{\"name\": \"imperial_si_conversion\",
    ↪ \"arguments\": {\"value\": 100, \"unit_in\": \"cm\",
    ↪ \"unit_out\": \"in\"}}&lt;/tool_call&gt;&lt;tool_call&gt;{\"name\": \"mean\",
    ↪ \"arguments\": {\"numbers\": [1, -5, 3]}}&lt;/tool_call&gt;"
  },
  {
    "role": "user",
    "content": "&lt;tool_response&gt;{\"name\": \"imperial_si_conversion\",
    ↪ \"results\": {\"result\":
    ↪ 39.37007874015748}}&lt;/tool_response&gt;&lt;tool_response&gt;{\"name\":
    ↪ \"mean\", \"results\": {\"result\":
    ↪ -0.3333333333333333}}&lt;/tool_response&gt;"
  },
  {
    "role": "assistant",
    "content": "&lt;tool_call&gt;{\"name\": \"add\", \"arguments\": {\"a\":
    ↪ -0.3333333333333333, \"b\":
    ↪ 10.5}}&lt;/tool_call&gt;&lt;tool_call&gt;{\"name\": \"percentage\",
    ↪ \"arguments\": {\"part\": 39.37007874015748, \"whole\":
    ↪ 10.166666666666666}}&lt;/tool_call&gt;"
  },
  {
```

```
    "role": "user",
    "content": "<tool_response>{\"name\": \"add\", \"results\":
    ↪  {\"result\":
    ↪  10.166666666666666}}</tool_response><tool_response>{\"name\":
    ↪  \"percentage\", \"results\": {\"result\":
    ↪  387.24667613269656}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "100 cm is equal to approximately 39.37 inches. This
    ↪  converted length represents about 387.25% relative to the total
    ↪  that comes from adding the average of 1, -5, and 3 to 10.5."
  },
  {
    "role": "user",
    "content": "What is 0 raised to the power of -1?"
  },
  {
    "role": "assistant",
    "content": "0 raised to the power of -1 is mathematically undefined.
    ↪  In general, raising 0 to a negative exponent involves dividing by
    ↪  zero, which is not defined in mathematics."
  },
  {
    "role": "user",
    "content": "Convert 10 meters to kilometers and round the result to 3
    ↪  decimal places."
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"si_unit_conversion\",
    ↪  \"arguments\": {\"value\": 10, \"unit_in\": \"m\", \"unit_out\":
    ↪  \"km\"}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"si_unit_conversion\",
    ↪  \"results\": {\"result\": 0.01}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "<tool_call>{\"name\": \"round_number\", \"arguments\":
    ↪  {\"number\": 0.01, \"decimal_places\": 3}}</tool_call>"
  },
  {
    "role": "user",
    "content": "<tool_response>{\"name\": \"round_number\", \"results\":
    ↪  {\"result\": 0.01}}</tool_response>"
  },
  {
    "role": "assistant",
    "content": "10 meters is equal to 0.01 kilometers, rounded to 3
    ↪  decimal places."
  }
]
```