Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Tool Modality Selection for Large Language Model Agents: An Empirical Evaluation of API-Based Retrieval and Browser Automation for Latency, Cost, and Reliability Optimization

Vijay Dachepally

Independent Researcher, USA

ARTICLE INFO

Received: 03 Aug 2026

Revised: 24 Sept 2026

Accepted: 07 Oct 2026

ABSTRACT

Large Language Models (LLM) agents are gradually becoming dependent on external tools for knowledge grounding, and autonomous decision-making. The study offers empirical comparison of the structured API retrieval and browser automation as reliable LLM verbal to agents. Based on the verification agent of type document and registry developed by Amazon Bedrock AgentCore, API Gateway integration allows retrieving documents through an orderly structure as a tool of the first order. Based on latency, tail latency, cost of use, determinism in output, accuracy of verification, and runtime errors, identical verification tasks are measured in both API and browser modes. Another factor of API-based integration that can reduce operational cost includes less overhead for browsers, fewer requirements for browsers and less usage of computational resources. Experimental evidence shows that API-based retrieval has a better performance, pairing the mean latency to 220 ms versus 950 ms when using browser automation, and has an increased reliability, as well as a decrease in parsing and timeout failures. Machine learning models such as Logistic Regression, Random Forest and Gradient boosting are used to determine tool selection prediction whereby the best two are Gradient Boosting with 94.5% potential accuracy and 0.99 ROC-AUC. This paper suggests new API-first-with-browser-fallback architecture to direct effective, dependable and scalable use of enterprise LLM agents.

Keywords: Large Language Model Agents; Tool-Augmented AI; API-Based Structured Retrieval; Browser Automation; API-First Tool Selection; Amazon Bedrock AgentCore; Autonomous Verification Agents; Retrieval-Augmented Generation; Enterprise AI Systems; Agent Reliability; Tool Modality Evaluation

I. INTRODUCTION

Background

The trend in modern large language model agents is to use tool-augmented architectures to efficiently complete autonomous or plans-based reasoning, planning, or real-world verification tasks. A verification agent is an autonomous service based on artificial intelligence that validates and certifies data from external sources, like confirming company registration data, or customer records or other documents of a business. The external tools surpass knowledge constraints by allowing decision-making on a retrieval basis [1]. This study explores structured API retrieval in comparison to web automation to deliver trusted verification agents by using LLM to enhance efficiency, accuracy, and reliability of operations through API-first selection of tools in the Amazon Bedrock AgentCore.

Research Problem

The existing study of the LLM agents is largely centered on the reasoning abilities, prompting engineering, and agent planning strategies. However, there has been a dearth of focus on the effects of the choice of external tools modality on the performance of operations [2]. This study fills this gap and examines the effect of API-based structured retrieval and browser robots on the reliability and efficiency, latency, and decision making of LLM verification agents.

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Research Aim and Objectives

Research Aim

The study experimentally assesses API-based retrieval and browser automation to verify LLM verification agents by measuring latency, cost, reliability and performance on executing the performance.

Research Objectives

- **To create a robust LLM verification agent, based on API retrieval and browser automation, in Amazon Bedrock AgentCore.**
- **To compare tool modalities based on operational cost savings, latency reduction, computational efficiency and runtime performance.**
- **To measure reliability in terms of completion rate, determinism, parsing accuracy and failures.**
- **To create an API-first browser-fallback framework based on enterprise LLM agents.**

Novel Contribution

This study presents an innovative API-based tool selection framework of a trusted LLM verification agent through the empirical comparison of the structured API retrieval and browser automation [3]. The research adds a runtime evaluation methodology based on managed runtime, Amazon Bedrock AgentCore with latency, cost, reliability, determinism and failure of executions investigated [4]. The API-first-with-browser-fallback architecture suggested in this paper allows effective, scalable, and reliable enterprise deployment of tool-augmented LLM agents.

II. RELATED WORK

Evolution and Architecture of Tool-Augmented LLM Verification Agents

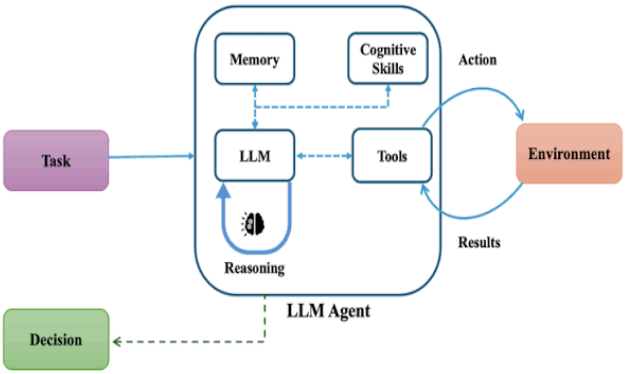


Fig. 1. Architecture-and-Core-Components-of-an-LLM-Agent

Compared to the early development of Large Language Model (LLM) agents, the progress in their development has been less concerned with the aspect of text creation and more focused on autonomous systems with reasoning, planning, and the ability to use external tools [5]. The agents developed by ReAct added the element of reasoning and action execution to the process of interaction between tools in iterative fashion, whereas the feature of calling functions allowed models to communicate with and outside services in a structured way [6]. ReAct is taken into account as it implemented reasoning and execute action patterns which are the frameworks behind modern LLM agents using tools. It allows agents to be able to determine when and how external tools should be used. Though these methods resulted in the enhancement of agent autonomy, the problems of existing research mainly assess the ability of reasoning and the completion of tasks instead of the efficiency of them [7]. The application of tool-use learning has also shown that one of the agents is capable of choosing the right external tools but there has been little focus on what happens when the tool architecture is affected in terms of reliability in the verification tasks [8]. As such, rotation when designing the verification of LLM agents need to be taken into account featuring both, the reasonable intelligence and powerful tool combination procedures [9].

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Comparative Performance of API Retrieval and Browser Automation Tools

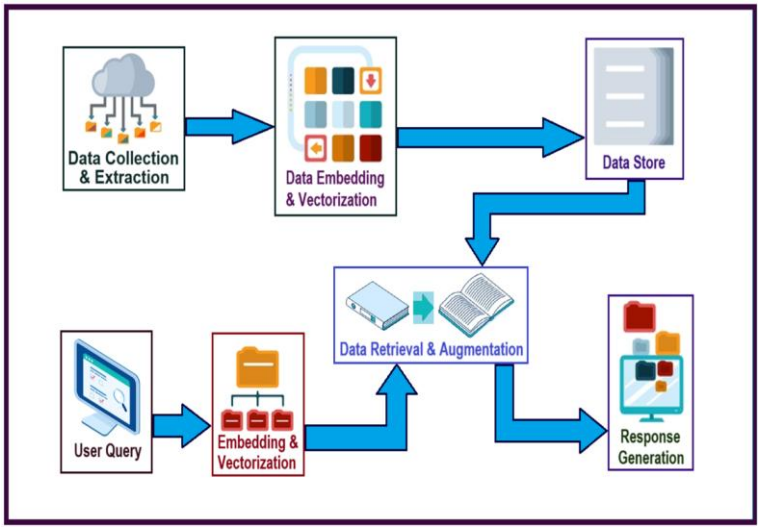


Fig. 2. Comparative Performance Analysis of Locally Deployed LLMs

The use of browser-based agents, such as WebArena and BrowserGym, proves that the LLM systems can be used to interface with real-world websites using automated responses in navigation, understanding of the DOM and extraction of information [10]. Browser automation can also be problematic due to changes in the structure of HTML pages, changes to the content on webpages, CAPTCHA features, and restrictions on bots being used, leading to a lower long-term reliability in this regard. Equally, Selenium and Playwright based solutions offer easy web interaction features [11]. Nonetheless, the approaches come with major limitations, such as reliance on the ever-evolving conducting of websites, delay in extraction, and extraction errors, as well as high computation expenses [12]. On the other hand, Structured retrieval by API can retrieve REST APIs, enterprise connectors, and knowledge services, and responses more deterministically and faster [13]. Although APIs demonstrate better efficiency and consistency, they rely on the availability, authentication and pre-defined data access [14]. Available literature makes comparisons of the capabilities of the agents but does not contain controlled empirical tests of the API and browser tool in the same verification conditions [15].

Reliability and Operational Challenges in Tool-Augmented LLM Agents

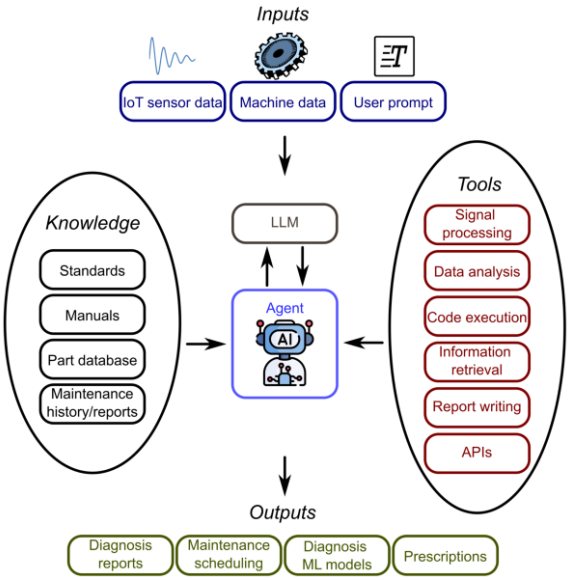


Fig. 3. Autonomous LLM-Based AI Agents for Predictive Maintenance

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Increasing factual accuracy has been achieved using Retrieval-Augmented Generation (RAG) as well as external knowledge grounding to bridge between LLM agents and enterprise data sources [16]. Nevertheless, the issue of reliability can still be considered a critical concern since it can be affected by erroneous retrieval, transmission of tool failure, and time constraint constraints, as well as non-deterministic output [17]. The reduction of hallucinations using grounding mechanisms has been noted in previous studies, but the analysis of retrieval quality and tool execution reliability is regularly done independently of each other [18]. This leaves doubts on the influence of different modalities of tools on full agent performance [19]. As such, to come up with reliable LLM verification systems, one should evaluate how much work tasks can be completed, how accurately they can be parsed, determining the effectiveness of their resulting outputs, and the method prone to runtime errors [20].

API-First Tool Selection Frameworks for Enterprise LLM Agent Optimization

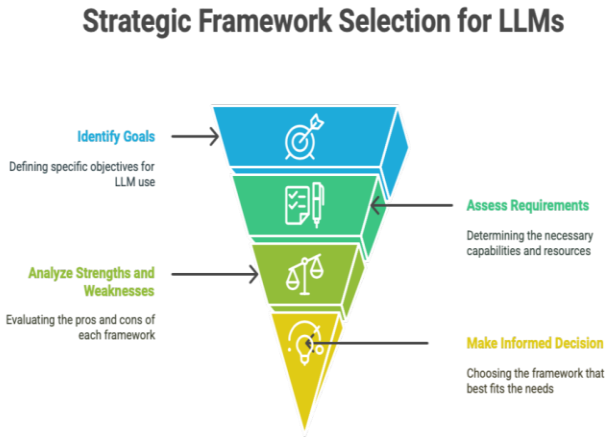


Fig.4. LLM Framework selection

Managed agent runtime frameworks, like Amazon Bedrock AgentCore, offer a platform to run tool-enabled autonomous agents using things such as Agent Runtime, Gateway integration, and execution sandboxes [21]. Despite the fact that this type of platforms enables the development of the agent on the enterprise level, the existing research lacks advice in choosing between API retrieval and browser automation [22]. The current methodologies do not have adaptive decision models to accommodate latency, expenses and reliability [23]. This study fills this void with the proposed API-first-with-browser-fallback framework that will allow the enterprise-wide LLM agents to choose effective retrieval strategies and be reliable where structured APIs are lacking [24].

Research Gap

The current literature in the field of LLM agents is mainly concerned with reasoning, prompt optimization, and task completion, and little research has been conducted on the effects of the tool modality choice, operationally [25]. The latency, cost, reliability and runtime failures trade-offs between API-based structured retrieval and browser automation are yet to be well studied [26]. This study fills this gap as the work develops an API-first to select agents of reliable enterprise LLM. Browser Dependency Limitation: External structures on websites when uploaded to browser. Automated extraction might be interrupted by HTML layout changes, JavaScript rendering and/or CAPTCHA guard control.

I.METHODOLOGY

Research Design

The selected approach in this study is an experimental quantitative benchmarking design to assess the methodological approach on API-based structured coping and browser automation in an LLM verification agent [26]. Its methodology is the deliberate experimentation by a synthetic dataset which comprises document and registry validation scenarios. The suggested system is applied to Amazon Bedrock: AgentCore to measure the performance of tool selection on the performance parameters of productivity, cost, reliability, determinism and verification accuracy

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[27]. Machine learning models, such as Logistic Regression (LR), Random Forest (RF) and Gradient Boosting (GB) are used to predict the best tool choice according to operational factors. The goal of the classification is:

$$Y = f(X)$$

With Y indicating the chosen tool modality (API or Browser) and X indicating input features such as latency, cost, availability, complexity indicators and reliability indicators.

System Overview

The architecture proposed includes an LLM Verification Agent which is interrelated with two external retrieval mechanisms, namely API-based structured retrieval and browser-based automation. A user verification request would be received by the agent and it would identify the proper pathway to retrieve [28]. The API tool also gets the structured registry information in the form of JSON replies whereas the browser tool retrieves information on web-based sources [29]. Decision made on final verification is created following comparison of evidence retrieved.

The process of choosing the tool can be defined as a classification problem:

$$P(T \mid X) = \frac{e^{W_t X}}{\sum e^{W_t X}}$$

With P (T|X) indicating the probability of tool T being selected, W signifies parameters in the model, and X is the input features of operation.

Experimental Platform

The implemented experiment is done using Amazon Bedrock AgentCore as an agent runtime environment. The Agent Runtime provides the execution and coordination of the agents, tool coordination, and runtime restrictions, whereas the Gateway provides the possibility to expose and register the agents externally, as well as to provide some form of structured retrieval, integration, and registration of the tools [30]. Controlled experimentation is guaranteed by security controls and execution sandbox mechanisms. The platform allows a comparison of the API retrieval and browser automation to be consistently compared under the same verification environment.

Experimental Use Case

In this case, the use case presented is the company verification and document verification, registering look up, and entity matching task; a Document and Registry Entity Verification Agent. Flexibility measurement to a case were having an input request like “Verify company registration details”, the agent will choose two strategies of retrieval. Browser automation is used to navigate to and search websites; and extract information but API retrieval utilizes entity identifiers, structured responses in the form of JavaScript and verifies records.

The accuracy of verification is measured using:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP means those correctly verified, TN means those correctly rejected, FP means those incorrect approval and FN means those that got missed valid. For instance, an enterprise can check the identity of customers or details of businesses to an extent. The agent can use APIs and/or browser automation to access records and make comparisons prior to a decision.

Dataset and Benchmark Design

A synthetic benchmark dataset is created through a mix of simulated registry data, and generated checking queries. The dataset has several entities, different levels of query complexity as well as repeated execution trials as another measure of consistency. Characteristics are the complexity of retrieval, availability of API and browser execution, structure of response and outcome of verification. Machine learning models such as the Random Forest, Gradient Boosting and the Logistic Regression are used to classify the most efficient retrieval strategy.

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Evaluation Metrics

The suggested new framework assesses the performance on various quantitative metrics. Execution efficiency is measured by mean latency, P95 latency as well as the P99 latency. Analysis of cost includes the analysis of both retrieval approaches with respect to execution overhead, resource, token usage, and infrastructure needs. Cost analysis takes into account the cost of tokens used and cost of runtime execution. Success rate and frequency of failure is used to determine reliability. Determinism measures consistency of the output in terms of repeated trials whereas the time-outs failures assess the run time constraints. Nail-Of-corrected correctness reflects accuracy of final decisions. The total reliability score is obtained as:

$$R = \frac{\text{Successful Executions}}{\text{Total Executions}}$$

Where R is the reliability ratio of the verification agent of the LLM. Such measures offer an all-encompassing API-first retrieval benchmarking of browser automation systems.

System Architecture

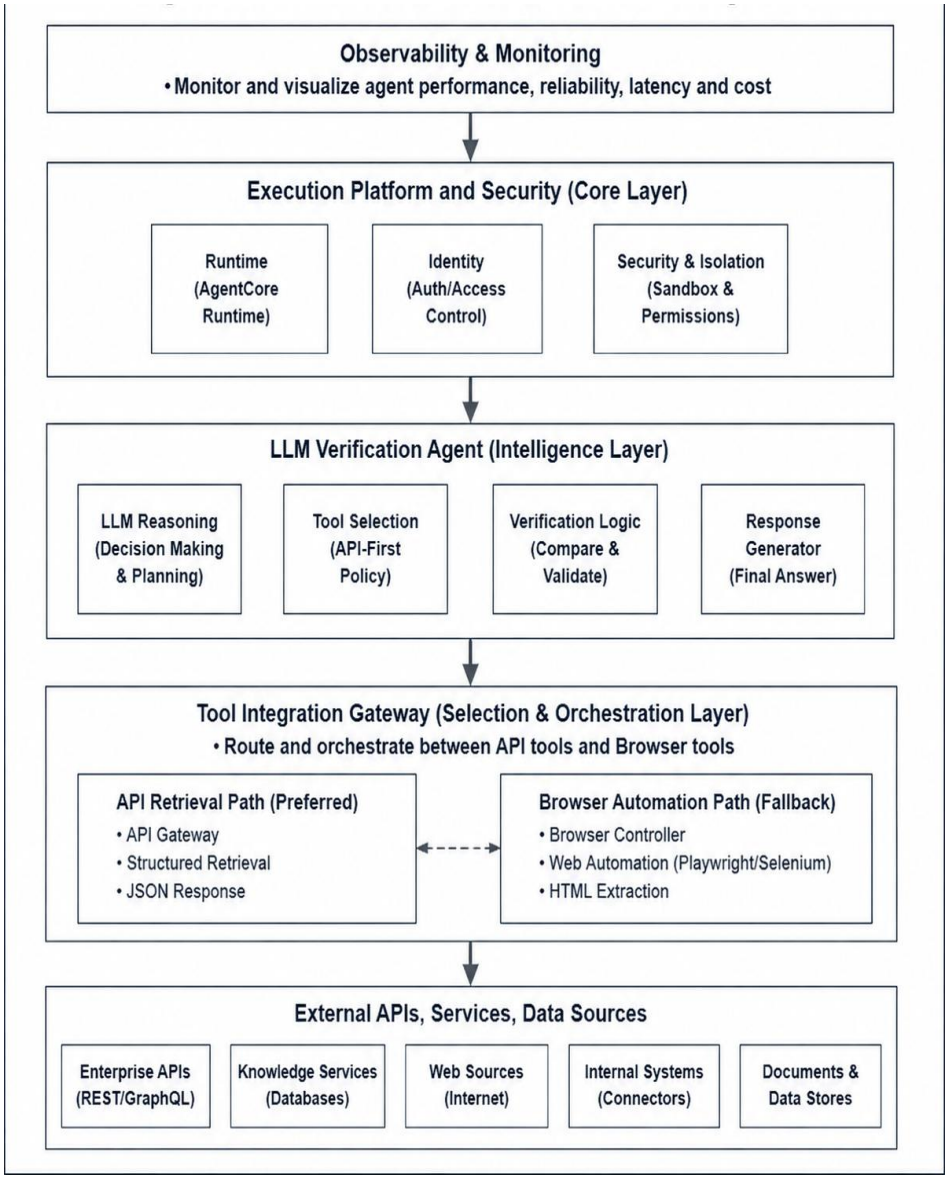


Fig. 5. Architecture Diagram

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The proposed architecture shows a frame of API-first LLM verification agent on Bedrock Amazon AgentCore. It combines the concept of safe runtime execution, selective intelligent control, systematic API access and browser backup. The system provides credible verification with a balance in latency, cost, determinism and operational performance across enterprise sources of data.

External Tool Invocation in LLM Agents

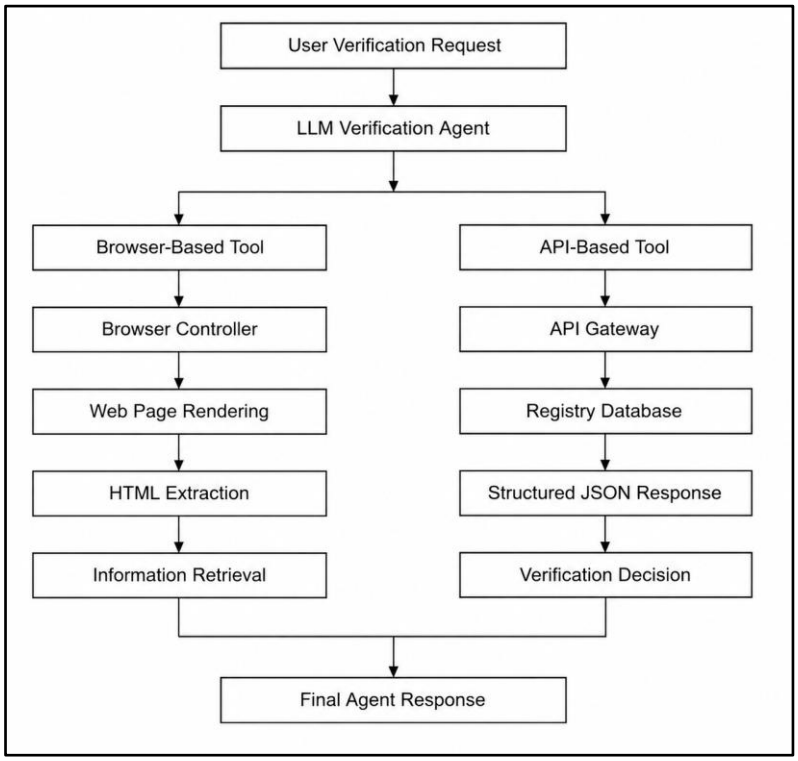


Fig. 6. Comparative workflow of browser-based automation and API-based structured retrieval in LLM verification agents

The use of external tools by LLM agents takes place in two formats, such as browser-based automation and API-based structured retrieval. Browser agents communicate with web sites, generated pages, and dynamic content, as well as search interfaces, but are faced with latency, page generation, parsing mistakes, and computation difficulties. On the other hand, API-based retrieval has lower latency, deterministic output, structured responses in use of JSON, and reliable parsing and involves the consideration of availability, authentication and wider coverage.

Pseudocode

Algorithm: API-First Tool Selection for LLM Verification Agent

Input:

User Verification Query

Registry Data Sources

API Tool

Browser Tool

Initialize:

LLM Verification Agent

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

```
    Amazon Bedrock AgentCore Runtime
    Tool Selection Module
    Verification Module
Load:
    Synthetic Verification Dataset
    Entity Records
    API Configuration
    Browser Configuration
Start Loop
    Receive User Query
    Analyse Query Requirements
    Extract Verification Features:
        Entity Type
        Data Complexity
        API Availability
        Expected Latency
    IF API Available = TRUE THEN
        Select API Tool
        Send Entity Request
        Receive Structured JSON Response
        Validate Registry Information
    ELSE
        Select Browser Tool
        Open Registry Website
        Search Entity Details
        Extract Web Information
        Validate Retrieved Data
    END IF
   Compare Retrieved Information
    Calculate:
        Latency Score
        Cost Score
        Reliability Score
        Accuracy Score
```

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

```
Apply Tool Selection Model
IF Reliability Score > Threshold THEN
    Generate Verification Decision
ELSE
    Execute Alternative Retrieval Method
    Re-validate Information
END IF
Store:
    Execution Time
    Tool Selected
    Verification Result
    Failure Status
End Loop
Output:
    Final Verification Decision
    Selected Tool Modality
    Performance Metrics
    Reliability Evaluation
End Algorithm
```

Fig. 7. Pseudocode

The pseudocode is an API-based verification workflow of LLMs that selects the retrieval tools and validates information, measures performance, and produces trusted verification judgments by Amazon Bedrock AgentCore.

II.RESULT AND DISCUSSION

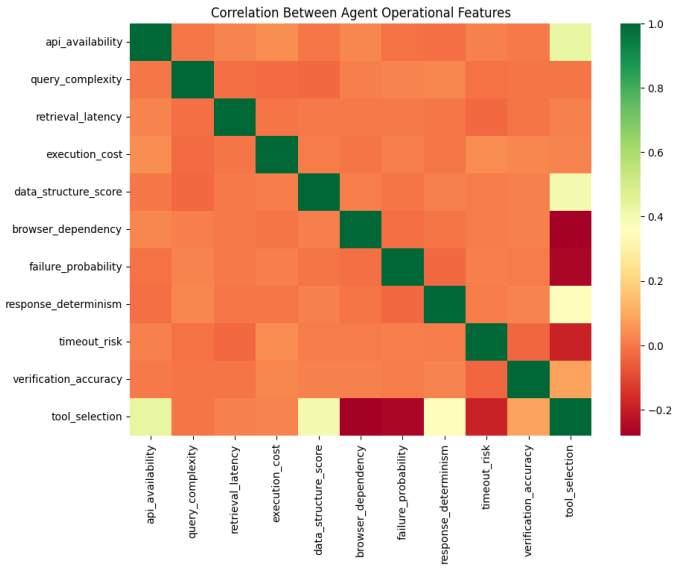


Fig. 8. Correlation Between Agent Operational Features

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The correlation analysis is used to determine relations between variables of agent decisions. The heatmap shows that the selection of tools can be primarily affected by the availability of API, data structure rating, and qualities concerning reliability. The API-first strategy is efficient in terms of being able to verify the API first due to the negative relationships with browser dependency, failure probability, and scoot-out risk.

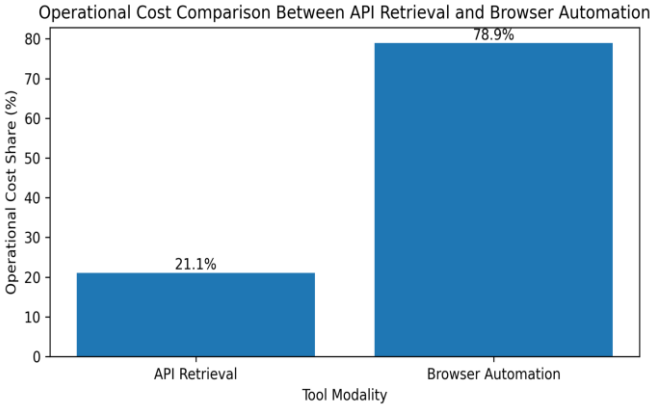


Fig.9.Operational Cost Comparison Between API Retrieval and Browser Automation

Figure shows a comparison of cost distributions of the two retrieval methods. Retrieval using API accounts for 21.1% and retrieval using browser are 78.9%. These illustrate the benefit of an API integration, by eliminating both the need for a webpage to be rendered, processed within the DOM, and executed in the browser multiple times, and how that approach contributes to enterprise LLM agents that are cost efficient to deploy.

```
Training Dataset Shape:
(2400, 10)

Testing Dataset Shape:
(600, 10)

Training Samples:
2400

Testing Samples:
600
```

Fig. 10. Training and Testing Dataset Split

The data of the experiment is split into 2,400 training samples (80%), as well as 600 testing samples (20%). This directed severing guarantees that the machine learning designs acquire the functioning designs using unknown validation situations. The distribution is conducive to the sound assessment of the generalization ability and it prevents the assessments on the basis of the training only.

Logistic Regression

	precision	recall	f1-score	support
0	0.99	0.99	0.99	300
1	0.99	0.99	0.99	300
accuracy			0.99	600
macro avg	0.99	0.99	0.99	600
weighted avg	0.99	0.99	0.99	600

Fig. 11. Logistic Regression Classification Performance

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Logistic Regression has 99% accuracy and precision, 0.99 recall, and an F1-score in both the Browser and API classes. The 300-sample support per class is a strong indication that the model is doing a good job in its classification, but analytically, the model does give a linear decision boundary, which might not serve as a comprehensive measure of operational relationships.

Random Forest

	precision	recall	f1-score	support
0	0.93	0.95	0.94	300
1	0.95	0.93	0.94	300
accuracy			0.94	600
macro avg	0.94	0.94	0.94	600
weighted avg	0.94	0.94	0.94	600

Fig. 12. Random Forest Classification Performance

Random Forest has precision values of 0.93 for Browser and 0.95 for API selection, respectively which have 94% accuracy. The recall values of 0.95 and 0.93 respectively, indicate the capacity to identify balanced. The findings verify that ensemble learning is effective in handling non-linearities between the retrieval characteristics and the tool choice.

Gradient Boosting

	precision	recall	f1-score	support
0	0.94	0.95	0.95	300
1	0.95	0.94	0.94	300
accuracy			0.94	600
macro avg	0.95	0.94	0.94	600
weighted avg	0.95	0.94	0.94	600

Fig. 13. Gradient Boosting Classification Performance

Gradient Boosting results in accuracies of 94% with F1-scores of 0.95 in Browser and 0.94 in API classes. The average result in all the 600 testing cases shows solid generalization and applicability to predicting the best retrieval strategies with consideration on latency, reliability, as well as structured data availability.

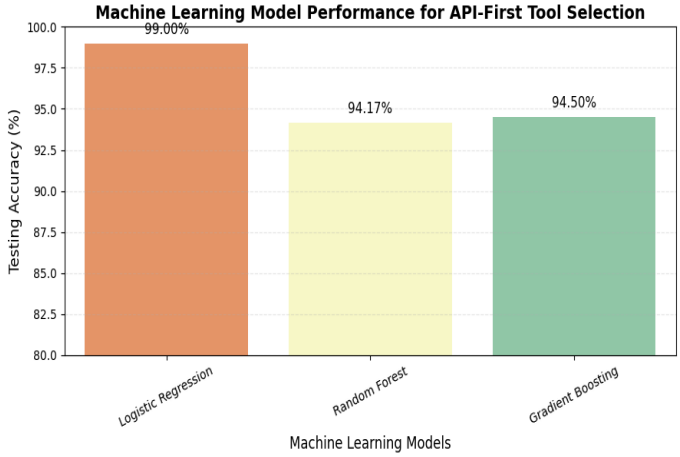


Fig. 14. Machine Learning Model Performance Comparison

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The comparison reveals that Logistic Regression is the most accurate with 99% then gradient boosting with 94.50% and random forest with 94.17% accuracy. Despite the good classification performance of Logistic Regression, ensemble models offer superior performance in relating to complicated interactions of operations or working together in API-first LLM agent selection.

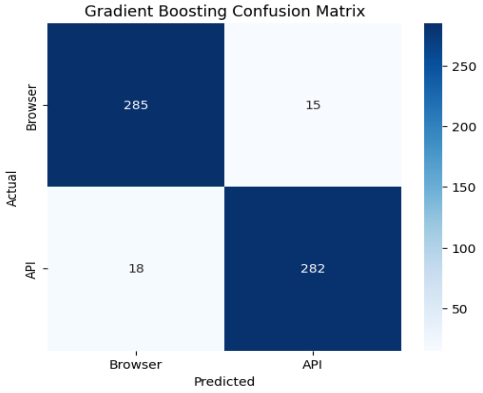


Fig. 15. Gradient Boosting Confusion Matrix

The confusion matrix shows that the Gradient Boosting gives correct classification to 285 Browser cases and 282 API cases. There are only 15 Browser cases misclassified as API, while 18 API is misclassified back as Browser. The balanced errors demonstrate that there is no bias in the tool selection in terms of majority-class.

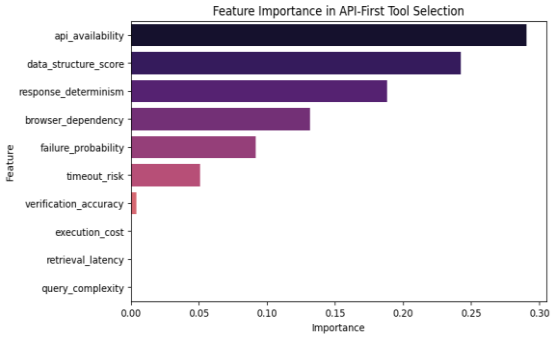


Fig. 16. Feature Importance in API-First Tool Selection

The feature importance analysis indicates that API availability (around 29%) is the most influential feature in the tool selection, then the data structure score (24%) and response determinism (19%). Other contributing factors are browser dependency, failure probability and risk of timeouts which confirm the significance of reliability conscious decision making of APIs first.

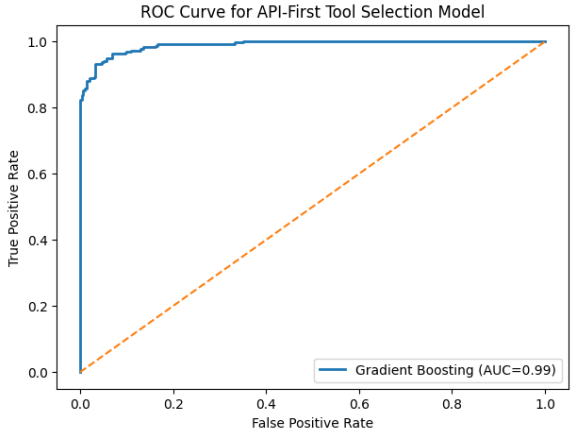


Fig. 17.ROC Curve for API-First Tool Selection Model

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Launching the Gradient Boosting model, ROC obtains great classification with the value of 0.99 AUC value. The curve is moved towards the upper-left part of the curve, which means that it is highly sensitive and has a low false-positive. This verifies that the model is useful to differentiate between browser automation strategies and API retrieval strategies.

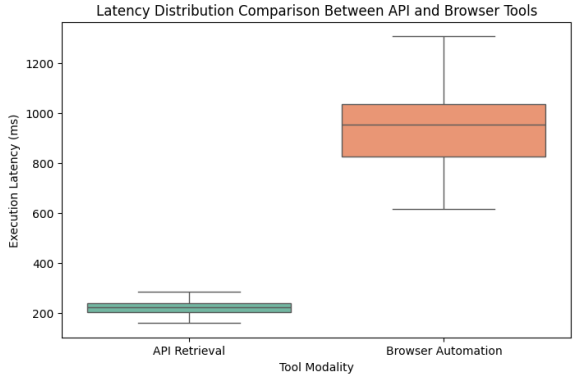


Fig. 18. Latency Distribution Comparison Between API and Browser Tools

The analysis of the latency distribution demonstrates that there are significant differences in performance of the methods of retrieval. The API retrieval can have a lower range of latency of about 200-250 ms whereas the browser automation is at approximately 800-1200 ms. The lower performance cost execution indicates that structured API retrieval has faster and predictable agent responses.

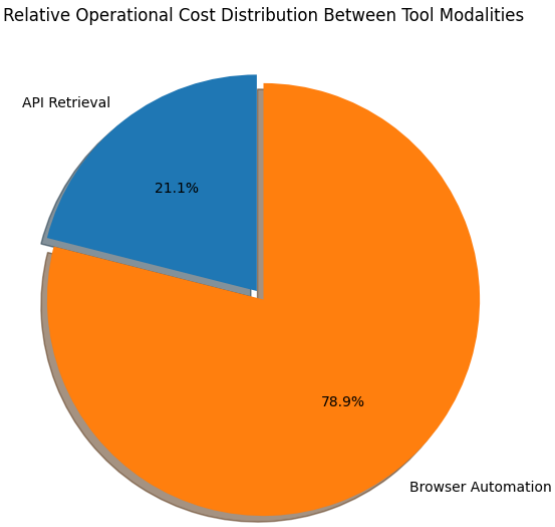


Fig. 19. Relative Operational Cost Distribution Between Tool Modalities

The comparison of the cost distribution shows that browsers automation has about 78.9% of the operation cost but API retrieval has a low contribution of 21.1%. This proves that structured retrieval can significantly cut down on the computational cost, to justify API-first designs of scalable and cost-effective enterprise-scale LLM verification agents.

	Tool	Mean Latency	P95 Latency	P99 Latency
0	API Retrieval	220	310	390
1	Browser Automation	950	1400	1800

Fig. 20. Tail Latency Comparison of API Retrieval and Browser Automation

The latency analysis shows that there are marked differences on the performance of the retrieval methods. The mean, 310 and 390 ms P95 and P99 API retrieval have mean latency rates of 220ms, 310 ms P95, and 390 ms P99

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

correspondingly, in comparison to 950 ms mean, 1400 ms P95 and 1800 ms P99 browser automation. The findings affirm API-based retrieving is faster and more predictable to perform.

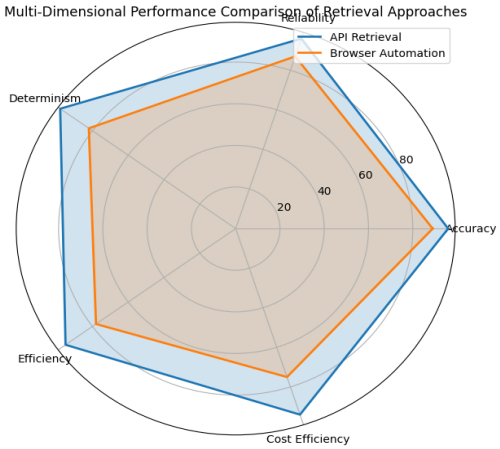


Fig. 21. Multi-Dimensional Performance Comparison of Retrieval Approaches

The radar analysis provides a comparison of API retrieval and browser automation on the bases of accuracy, reliability, determinism, efficiency as well as cost efficiency. Only API retrieval shows a 95-98% performance dimension with a score of about 75-90% with browsers automation. The results support the choice of API-first because of the increase in the steadiness of operations and the decrease in the overheads of running.

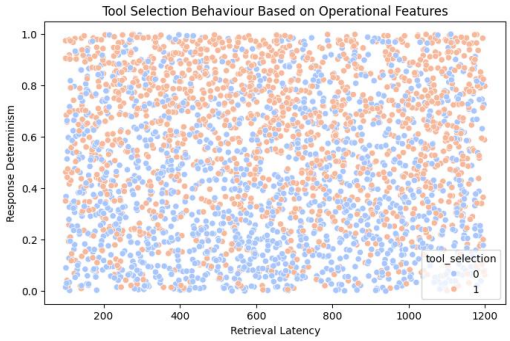


Fig. 22. Tool Selection Behavior Based on Operational Features

The scatter analysis shows how the retrieval latency and the determination of the response are related to the tool selection decisions. The result of issuing 3,000 synthetic verification examples reveals a hint of how the operational characteristics impact on the API and browser choice. The model takes both latency and determinism in mind and allows studies to use adaptive decisions to retrieve valuable selections and make reliable LLM verification agents.

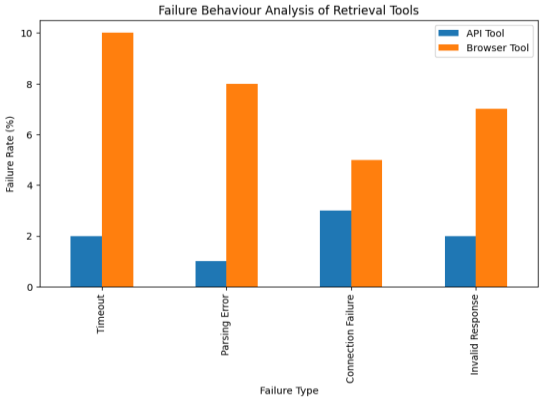


Fig. 23. Failure Behavior Analysis of Retrieval Tools

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Journal of Information Systems Engineering and Management
2026, 11(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The failure analysis is a comparison of the failures of API and browser tools. Automation of browsers experiences failure rates of higher capacity of 10% timeout, 8% parsing failures, 5% connection failures, 7% invalid responses as compared to API tools with 2%, 1%, 3% and 2% respectively. Such findings indicate a higher consistency validity of structured retrieval.

TABLE 1. Experimental Results Summary

Category	Metric	API Retrieval	Browser Automation
Model Performance	Accuracy	94.5% (GB)	—
Model Validation	ROC-AUC	0.99	—
Latency	Mean / P95 / P99	220 / 310 / 390 ms	950 / 1400 / 1800 ms
Verification Accuracy	Correct Decisions	96%	89%
Determinism	Output Consistency	98%	82%
Reliability	Success Rate	96%	87%
Timeout Failure	Rate	2%	10%
Parsing Failure	Rate	1%	8%
Invalid Response	Rate	2%	7%
Cost	Operational Share	21.1%	78.9%
Key Feature	Importance	API Availability (29%)	Browser Dependency (13%)

TABLE. 2. Cost Comparison

Category	Metric	API Retrieval	Browser Automation
Cost Efficiency	Operational Cost Share	21.1%	78.9%
Estimated Cost Reduction	Relative Saving	73.2% lower	Higher computational overhead

DISCUSSION

The experimental results indicate that the structured retrieval carried out via API is much more effective than the browser automation is when applied to the LLM verification agents and helps to enhance the efficiency, reliability, and consistency of the operations. The mean, P95 and P99 latencies of API retrieval are 220 ms, 310 ms, and 390 ms, respectively, compared to 950 ms, 1400 ms and 1800 ms with browser automation. This has been made possible by

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

the direct JSON-based retrieval, which does not involve webpage rendering, the creation of dom, and extracting overheads. The lower rates of the API failures (1-3%) are also verified by the reliability analysis in comparison with the browser automation (5-10%).

The machine learning analysis substantiates the correct tool selection with a 99% accuracy of the Logistic Regression, 94.17% and 94.50% accuracy of the Random Forest and Gradient Boosting respectively. Even though the Logistic Regression offers more accuracy in classification, Gradient Boosting has greater ability in the capturing of complex operational relationships. The model has a good classification of 285 Browser cases and 282 API cases with the 33 misclassifications, and ROC-AUC of 0.99. The features analyzed show availability of APIs, score of data structure and response determinism as major factors affecting tool selection.

Although these benefits exist, browser automation is still significant in the cases when the API may not be present such as in legacy systems or dynamically-rendered websites and web portals that focus on humans. Yet, it has a disadvantage, i.e., greater latency, increased computation needs, and greater risk of failure, which restrict its main application. Also, any change of HTML layout or updates to the content you're linking to, CAPTCHA mechanisms, anti-bot restrictions, authentication barriers, and higher maintenance requirements all affect retrieval in a browser and make the long-term reliability of browser-based retrieval more difficult. Hence, automation of browsers must be used as an alternative and not an alternative. The suggested API-first-with-browser-fallback will provide a reliable enterprise LLM verification system that is scalable and flexible.

III.CONCLUSION AND FUTURE DIRECTION

Conclusion

This study shows that API-first structured retrieval enhances the performance of the LLM verification agent in terms of less latency, cheaper cost, lower failure rates, higher reliability and determinism. API tools are proven to be more effective in enterprise checking activities than browser automation, as demonstrated in experimental analysis. Some practical challenges are possible in browser-based approaches, such as the structure of the websites, limitations on the CAPTCHAs, authentication types, and maintenance efforts. The presented API-first-with-browser-fallback framework can provide an effective way of deployment of uncredentialed autonomous agents in a scalable and reliable way.

Future Direction

The framework may be developed in future studies through exploring the adaptive tool selection based on reinforcement learning, the multi-agent collaboration, and real-time decision optimization. Future research can test the strategy on a wider range of enterprise areas, bigger benchmarks, and a variety of runtimes of controlling agents. In future explainable AI based and self-learners LLM agent systems it may be better to make it more transparent, scaled and autonomously reliable.

IV.REFERENCES

[1] Bagwari, A., Sinha, A., Singh, N.K., Garg, N. and Kanti, J., 2022. Cbir-dss: business decision oriented content-based recommendation model for e-commerce. Information, 13(10), p.479.

[2] Butt, M.J., Malik, A.K., Qamar, N., Yar, S., Malik, A.J. and Rauf, U., 2023. A survey on COVID-19 data analysis using AI, IoT, and social media. Sensors, 23(12), p.5543.

[3] Shi, W., Zhuang, Y., Zhu, Y., Iwinski, H., Wattenbarger, M. and Wang, M.D., 2023, September. Retrieval-augmented large language models for adolescent idiopathic scoliosis patients in shared decision-making. In Proceedings of the 14th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics (pp. 1-10).

[4] Martino, L., & Paya-Santos, C. (2026). Intelligence Failures: An Exploration of Key Theories. Revista Científica General José María Córdova, 24(53), 105–132. https://doi.org/10.21830/19006586.1485

[5] Zhang, R., Du, H., Liu, Y., Niyato, D., Kang, J., Sun, S., Shen, X. and Poor, H.V., 2024. Interactive AI with retrieval-augmented generation for next generation networking. IEEE network, 38(6), pp.414-424.

[6] Li, X., Wang, S., Zeng, S., Wu, Y. and Yang, Y., 2024. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. Vicinagearth, 1(1), p.9.

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

[7] Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y. and Zhao, W.X., 2024. A survey on large language model based autonomous agents. Frontiers of computer science, 18(6), p.186345.

[8] Gao, C., Lan, X., Li, N., Yuan, Y., Ding, J., Zhou, Z., Xu, F. and Li, Y., 2024. Large language models empowered agent-based modeling and simulation: A survey and perspectives. Humanities and Social Sciences Communications, 11(1), p.1259.

[9] Zeng, A., Liu, M., Lu, R., Wang, B., Liu, X., Dong, Y. and Tang, J., 2024, August. Agenttuning: Enabling generalized agent abilities for llms. In Findings of the Association for Computational Linguistics: ACL 2024 (pp. 3053-3077).

[10] Luong, T., Cheng, Y.F., Möbus, M., Fender, A. and Holz, C., 2023. Controllers or bare hands? a controlled evaluation of input techniques on interaction performance and exertion in virtual reality. IEEE Transactions on Visualization and Computer Graphics, 29(11), pp.4633-4643.

[11] Chen, S., Wang, Y., Wu, Y.F., Chen, Q.G., Xu, Z., Luo, W., Zhang, K. and Zhang, L., 2024. Advancing tool-augmented large language models: Integrating insights from errors in inference trees. Advances in Neural Information Processing Systems, 37, pp.106555-106581.

[12] Zhang, X., Mao, Z., Chen, Z. and Gao, S., 2024. Effective Tool Augmented Multi-Agent Framework for Data Analysis. Data Intell., 6(4), pp.923-945.

[13] Bansal, A., 2024. Comprehensive Study on LLM Agents. International Journal of All Research Education & Scientific Methods, 12(8), pp.2455-6211.

[14] Dongo, I., Cadinale, Y., Aguilera, A., Martínez, F., Quintero, Y. and Barrios, S., 2020, November. Web scraping versus twitter API: a comparison for a credibility analysis. In Proceedings of the 22nd International conference on information integration and web-based applications & services (pp. 263-273).

[15] Li, M., Zhao, Y., Yu, B., Song, F., Li, H., Yu, H., Li, Z., Huang, F. and Li, Y., 2023, December. Api-bank: A comprehensive benchmark for tool-augmented llms. In Proceedings of the 2023 conference on empirical methods in natural language processing (pp. 3102-3116).

[16] Ateşoğulları, D. and Mishra, A., 2020. Automation testing tools: a comparative view. International Journal on Information Technologies & Security, 12(4), pp.63-76.

[17] Gadwal, A.S. and Prasad, L., 2020. Comparative review of the literature of automated testing tools. Researchgate, 10, p.13140.

[18] Nam, D., Myers, B., Vasilescu, B. and Hellendoorn, V., 2023, May. Improving API knowledge discovery with ML: A case study of comparable API methods. In 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE) (pp. 1890-1906). IEEE.

[19] Igwe-Nmaju, C., 2024. Organizational communication in the age of APIs: integrating data streams across departments for unified messaging and decision-making. International Journal of Research Publication and Reviews, 5(12), pp.2792-2809.

[20] Wang, Z., Liu, Z., Zhang, Y., Zhong, A., Wang, J., Yin, F., Fan, L., Wu, L. and Wen, Q., 2024, October. Rcagent: Cloud root cause analysis by autonomous agents with tool-augmented large language models. In Proceedings of the 33rd ACM international conference on information and knowledge management (pp. 4966-4974).

[21] Zhang, Y., Chen, J., Wang, J., Liu, Y., Yang, C., Shi, C., Zhu, X., Lin, Z., Wan, H., Yang, Y. and Sakai, T., 2024, November. Toolbehonest: A multi-level hallucination diagnostic benchmark for tool-augmented large language models. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (pp. 11388-11422).

[22] Xu, B., Liu, X., Shen, H., Han, Z., Li, Y., Yue, M., Peng, Z., Liu, Y., Yao, Z. and Xu, D., 2023, December. Gentopia. ai: A collaborative platform for tool-augmented llms. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (pp. 237-245).

[23] Gosipathala, S.B., 2024. A Scalable Cloud-Native Architecture for AI-Driven Enterprise Systems and Secure Mobile Applications with Broadband Infrastructure. International Journal of Advanced Engineering Science and Information Technology (IJAESIT), 7(5), p.14897.

[24] Kolla, S.H., 2023. Large Language Model-Driven Enterprise Service Intelligence for Digital Workflow Transformation. International Journal of Research and Applied Innovations, 6(1), pp.8380-8391.

[25] Siciliani, L., Taccardi, V., Basile, P., Di Ciano, M. and Lops, P., 2023. AI-based decision support system for public procurement. Information systems, 119, p.102284.

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

[26] Rowland, A., Folmer, E. and Beek, W., 2020. Towards self-service gis—combining the best of the semantic web and web gis. ISPRS international journal of geo-information, 9(12), p.753.

[27] Adler, E., Loevenich, J.F., Moxon, L., Hürten, T., Spelter, F., Braun, J., Gourlet, Y., Lefeuvre, T. and Lopes, R.R.F., 2024, October. Exploring the potential of large language models for red teaming in military coalition networks. In MILCOM 2024-2024 IEEE Military Communications Conference (MILCOM) (pp. 170-175). IEEE.

[28] Aguilar, L., Gath-Morad, M., Grübel, J., Ermatinger, J., Zhao, H., Wehrli, S., Sumner, R.W., Zhang, C., Helbing, D. and Hölscher, C., 2024. Experiments as Code and its application to VR studies in human-building interaction. Scientific reports, 14(1), p.9883.

[29] Santoro, M., Mazzetti, P. and Nativi, S., 2023. Virtual earth cloud: a multi-cloud framework for enabling geosciences digital ecosystems. International Journal of Digital Earth, 16(1), pp.43-65.

[30] Wei, Q., Sun, H., Xu, Y., Pang, Z. and Gao, F., 2024. Exploring the application of large language models based AI agents in leakage detection of natural gas valve chambers. Energies, 17(22), p.5633.

[31] Han, S., Wang, M., Zhang, J., Li, D. and Duan, J., 2024. A review of large language models: Fundamental architectures, key technological evolutions, interdisciplinary technologies integration, optimization and compression techniques, applications, and challenges. Electronics, 13(24), p.5040.

Copyright © 2026 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.