iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80179

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Voice Command-Based Desktop Agent System for Automated Software Installation

Siddharth Shasane[1], Jay Raut[2], Manthan Patil[3], Prof. Sunita Naik[4]
Department of Computer Engineering, VIVA Institute of Technology, Mumbai, India

Abstract: ***Software installation on desktop systems is still a largely manual process that involves searching for the correct software, selecting a safe download source, downloading installer files, and completing multiple installation steps. This process is time-consuming, repetitive, and increases the risk of downloading malware or fake installers from untrusted websites. In large organizations, the same installation procedure must be repeated across multiple computers, which increases the workload of IT administrators and delays software deployment.***

To solve these issues, the proposed system provides a voice-based software installation assistant that automates the complete process using simple voice commands. The system captures voice input, identifies the required software, verifies trusted sources, securely downloads the installer, and performs automatic installation along with necessary configurations. Security mechanisms such as trusted domain verification and authenticity validation are included to ensure safer installation.

The proposed system also includes a centralized installation module for organizational use, where an administrator can install software on multiple computers simultaneously, ensuring consistent setup and reducing repetitive work.

Testing confirms that the system successfully installs commonly used applications such as browsers, media players, and development tools with minimal user involvement. The system reduces manual effort and installation time significantly when compared to traditional installation methods, while the centralized deployment feature improves efficiency by enabling bulk installations across multiple PCs.

Overall, the system provides a secure and efficient solution for hands-free software installation and centralized deployment, reducing user effort, improving safety, and supporting faster software setup for both individual and organizational environments.

Keywords: Software Installation, Desktop Automation, Voice Command System, Secure Downloading, Phishing Detection, GUI Interaction, System Configuration, Task Automation, Error Handling, Hands-Free Operation.

I. INTRODUCTION

In today's digital era, software applications are essential for both personal and professional computer usage. From basic utilities such as web browsers and media players to advanced tools like development environments and office suites, software plays an important role in improving productivity and efficiency. However, the process of downloading and installing software on a computer is still mostly manual and involves multiple steps such as searching for the correct application, selecting a trusted download source, downloading the installer file, running the setup file, and completing installation configurations. This manual procedure is time-consuming and can be difficult for non-technical users.

Another major issue in software installation is security. Many users unknowingly download software from unverified websites, which can lead to malware infections, corrupted installation files, and privacy risks. Fake software installers and malicious downloads are common problems when users depend on random online sources. Therefore, ensuring that software is downloaded only from trusted sources has become an important requirement in modern computer systems.

To solve these issues, this system provides a voice-based software installation mechanism that automates the complete installation process using simple voice commands. The system is capable of identifying the requested software, locating official or trusted sources, securely downloading installer files, and performing the installation automatically with required configurations. This reduces user effort, saves time, and provides a hands-free installation experience.

In addition to single-user installation, the system also includes a centralized installation module designed for large organizations. In companies, colleges, and IT departments, the same software often needs to be installed on multiple computers. Performing installations manually on every system consumes significant time and requires continuous monitoring. The centralized module allows an administrator to install software on multiple PCs at the same time, ensuring faster deployment and consistent software setup across all systems.

The system is designed to provide a secure and efficient solution for software installation automation. By combining voice command processing, secure downloading, and automated installation execution, the system improves user convenience and reduces security risks. This approach shows how automation can simplify complex installation tasks while maintaining reliability and scalability for both personal and organizational environments

II. RELATED WORK

The field of Automated system software installation has seen several researches in the past with detailed information about the previous system along with its problem definition to make the system and scope for further systems.

A. Literature Review

Recent work in the fields of desktop automation, GUI grounding, and frameworks for intelligent agents involved numerous approaches toward making Windows-based systems more robust and autonomous. Some of the earlier agent architectures, like UFO2, pioneered multi-agent OS-level automation and hybrid GUI-API interaction, together with speculative execution for optimizing task flows [1]. Complementary work on visual grounding, such as WinClick, introduced lightweight multimodal models that could detect GUI elements purely from screenshots, aided by benchmarks like WinSpot for Windows environments [2]. Others extended this to dynamic task evaluation: WorldGUI offered a large-scale benchmark with variable initial states and proposed planner-critic and actor-critic modules for error detection and correction during execution [3]. Other multi-agent coordination frameworks, such as COLA, further improved scalability by providing task schedulers, specialized pools of agents, and interactive backtracking for error recovery [4].

Parallel lines of research addressed automation beyond GUI control. LLM-driven installation agents like installamatic automated Python project setup through documentation parsing and Dockerfile generation, pointing out gaps in environment management capabilities [5]. Automated installer analysis systems simulated real user interactions to classify freeware installers by aggressiveness using sandboxing and image heuristics [6], while cybersecurity-focused installation automation achieved high offline installation success but underlined remaining manual steps in the deployment of the exploits [7]. RPA-based approaches demonstrated advancements in UI test automation across heterogeneous systems, notably relying on OCR and vision techniques [8]; industry-oriented computer-vision automation tools like Korat have improved the reliability of GUI interaction by conducting specialized preprocessing, considering the impact on the accuracy of OCR results [9]. Further supporting work has investigated web automation via Selenium for non-API environments [10], the position of voice assistants as modern web interfaces that shape the way information is accessed for users [11], and STT/TTS pipeline-based voice-controlled automation systems with a focus on accessibility integrated with browser automation [12]. Surveys of limited-vocabulary ASR [13], analyses of dependency update strategies in npm ecosystems [14], and deep graph learning methods like SIGL for malicious installation detection [15]

More recent studies have focused on integrating large language models into GUI automation, where surveys of LLM-based agents discussed architectures, challenges, and limitations such as reliability and hallucination [16]. Model-based GUI automation techniques provided structured approaches using event-flow models but faced challenges in dynamic environments [17], while comparative studies of GUI automation tools highlighted limitations in handling frequent interface changes [18]. Hybrid automation frameworks combining Selenium and PyAutoGUI demonstrated flexibility across web and desktop environments but suffered from screen dependency issues [19]. Finally, voice-controlled desktop automation systems enabled hands-free interaction but were limited in handling complex multi-step tasks and maintaining accuracy in noisy environments [20].

B. Problem Statement and Scope

Software installation is a time-consuming process that involves multiple manual steps such as searching for reliable sources, downloading files, verifying authenticity, and completing installation configurations. Existing automation solutions are mostly script-based and depend on GUI structures, making them unreliable in dynamic environments. Additionally, many systems lack proper security validation, exposing users to phishing risks and malicious software. Most available solutions also focus on single-user systems and do not support centralized deployment, creating challenges in organizational environments where software needs to be installed across multiple computers.

The scope of this system is to provide a voice-based software installation assistant that automates the complete installation process from software identification to secure setup. The system supports hands-free operation, trusted source selection, secure downloading, and automatic installation with required configurations. It also includes a centralized module for deploying software on multiple PCs simultaneously, improving efficiency in organizations. The system focuses on Windows environments and ensures secure, reliable, and efficient software installation, with future scope for cross-platform support and cloud-based deployment.

III. PROPOSED SYSTEM

A. *System Overview*

This system provides a completely hands-free solution for downloading and installing software on desktop computers through simple voice commands. User input through speech recognition confirms the authenticity of the software from trusted sources and drives automated installation by emulating user interactions. It has inbuilt error handling and feedback mechanisms to make operations secure, efficient, and user-friendly, reducing manual efforts and saving time while maintaining safety and system reliability.

B. *System Architecture*

1) User Interaction Layer: Voice-based interaction initiates the system, which converts the user's command into textual input through speech-to-text processing. It identifies the required software and confirms it with the user for a completely hands-free experience.

2) System Configuration Check Layer: On this layer, it checks for hardware, OS compatibility, disk space, and permissions to avoid installation failure for oncoming system readiness.

3) Download Layer: It also reduces manual effort: the system checks compatibility, securely downloads the software, detects an installer, and opens it automatically.

4) Phishing Website Detecting Layer: This will check the software URL with an ML model; if it finds any suspicious links, it does further analysis on the security of those prior to downloading.

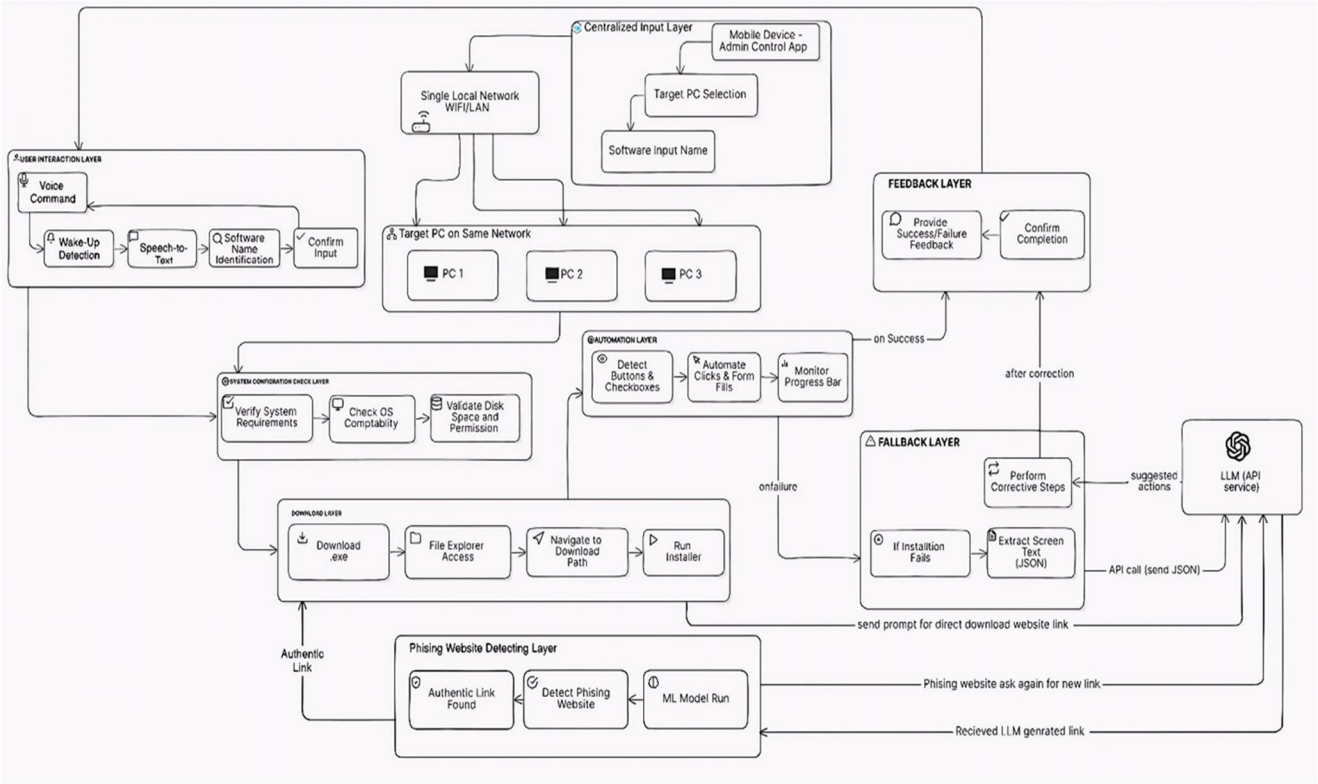


Fig. 1.System Architecture

1) Automation Layer: This layer is where the application installation is automated, simulating user-like actions such as clicking, filling in forms, and checking on progress.

2) Feedback Layer: Based on the installation of the software, success or failure is reported to the user.

3) Fallback Layer: In case there is any error, this layer analyzes on-screen text for the problem, and based on corrective suggestions, it retries installation.

4) LLM Service: This will aid in fallback layer error analysis and further adaptive recovery measures in case of unexpected issues.

5) Centralized Installation Layer: This layer is designed for large organizations. It allows an administrator to deploy selected software on multiple computers simultaneously from a centralized interface. The module ensures uniform installation and reduces manual workload for IT administrators by supporting bulk deployment across multiple systems at the same time.

C. System Methodology

Follows a structured, multi-phase approach to develop a reliable voice-based agent.
Begins with a literature review of existing automation and installation systems.
Identifies key gaps in current methods::

1) Low fault acceptance
2) Limited security mechanisms
3) Inflexible, non-adaptive automation

System structured into multiple functional layers:

- User Interaction Layer – handles voice commands and feedback
- System Check Layer – verifies compatibility and environment readiness
- Secure Download Layer – ensures trusted software sources
- Automation Layer – executes installation steps automatically
- Error Handling & Feedback Layer – detects, reports, and corrects issues
- LLM Service – verify trusted software source, error handling
- Centralized Layer – handle organization level control multiple pc on single network

IV. RESULTS AND DISCUSSION

A. Partial Implementation

The system will include a voice-based agent, which will perform automated installations of the Python 3.12 desktop system software. It will utilize libraries for recognizing voices, GUI automation, and causing system-level interactions to execute processes with precision. Then, it captures the user's voice commands, checks operating system compatibility, securely downloads installer files, and then automates the installation. Meanwhile, phishing model checks sources for safety. Overall, the system performs secure, efficient, hands-free software installation.

B. Functional Results

The current working version includes several operational screens representing key system functions.

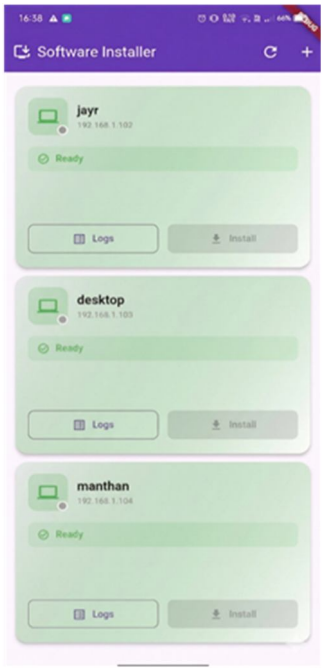


Fig. 2. Admin Panel

The Fig. 2 demonstrates how the admin panels look and the devices on the same network are enlisted.

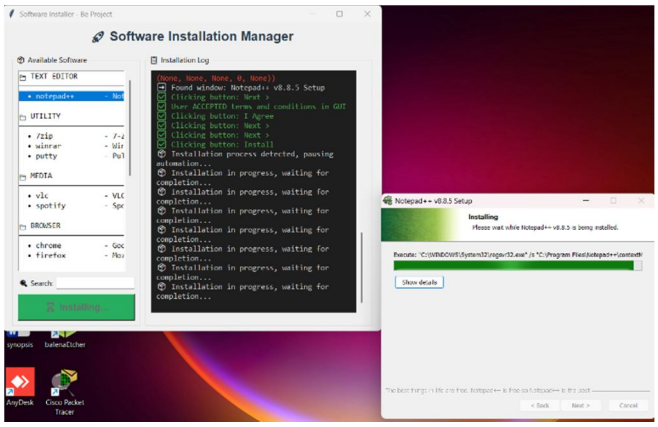


Fig. 3. Installation Process Started

Fig. 3 shows the start of the automated software installation process.

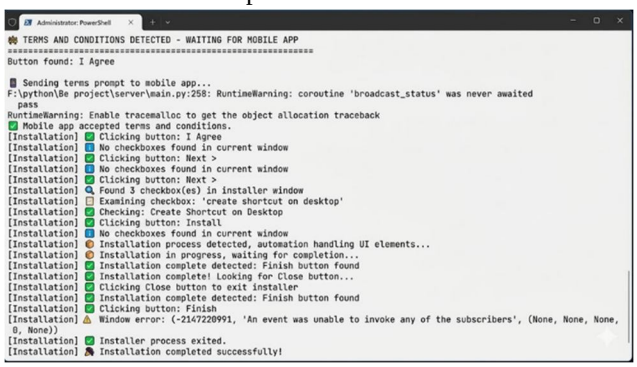


Fig. 4. Installation Logs

Fig. 4 shows the logs during Installation of any Software.

A. Comparative Evaluation

This chapter compares existing automation systems with the developed desktop agent system. The comparison underlines the main differences in terms of functionality, safety, adaptability, and user interaction. The comparison reveals how the developed system allows for automating software installation in a more effective, secure, and user-friendly way compared to the traditional solutions or those based on scripts.

Table. 1. Comparison Analysis Between Existing Systems and the Proposed System

Sr. No.	Existing Systems	Proposed System
1	Limited task automated required human interaction for setup. [1], [2], [3], [4], [16]	Automates Full downloading process from finding links, download as well as setup.
2	Depends on random or third-party websites for downloading. [6], [15]	Download only from trusted and verified official sources.
3	Most systems support single-system automation only. [1]–[20]	Supports centralized installation across multiple PCs for organizations.
4	Automation tools depend heavily on GUI layout and screen recognition [9], [17], [18], [19]	Uses controlled installer automation for better reliability
5	Limited fault tolerance and error recovery mechanisms [5], [17], [18]	Includes fallback mechanism with LLM-based error correction

The above comparison highlights that existing systems focus on limited automation, lack security validation, and do not support scalable deployment. In compare, the proposed system provides a complete, secure, and reliable solution with voice-based automation, error handling, and centralized multi-PC installation.

V. CONCLUSION

The developed system successfully automates the process of software downloading and installation using voice commands and centralized deployment. The system integrates speech recognition, secure download verification, automated installer execution, and intelligent error-handling mechanisms to provide a reliable and efficient installation workflow. By eliminating the need for manual searching, downloading, and setup interaction, the system significantly reduces user effort and installation time.

In addition to individual voice-based installation, the centralized deployment module enables administrators to install software on multiple computers simultaneously, making the system suitable for organizational environments. Security measures such as phishing detection and trusted source verification ensure that software is downloaded from safe sources, reducing potential security risks.

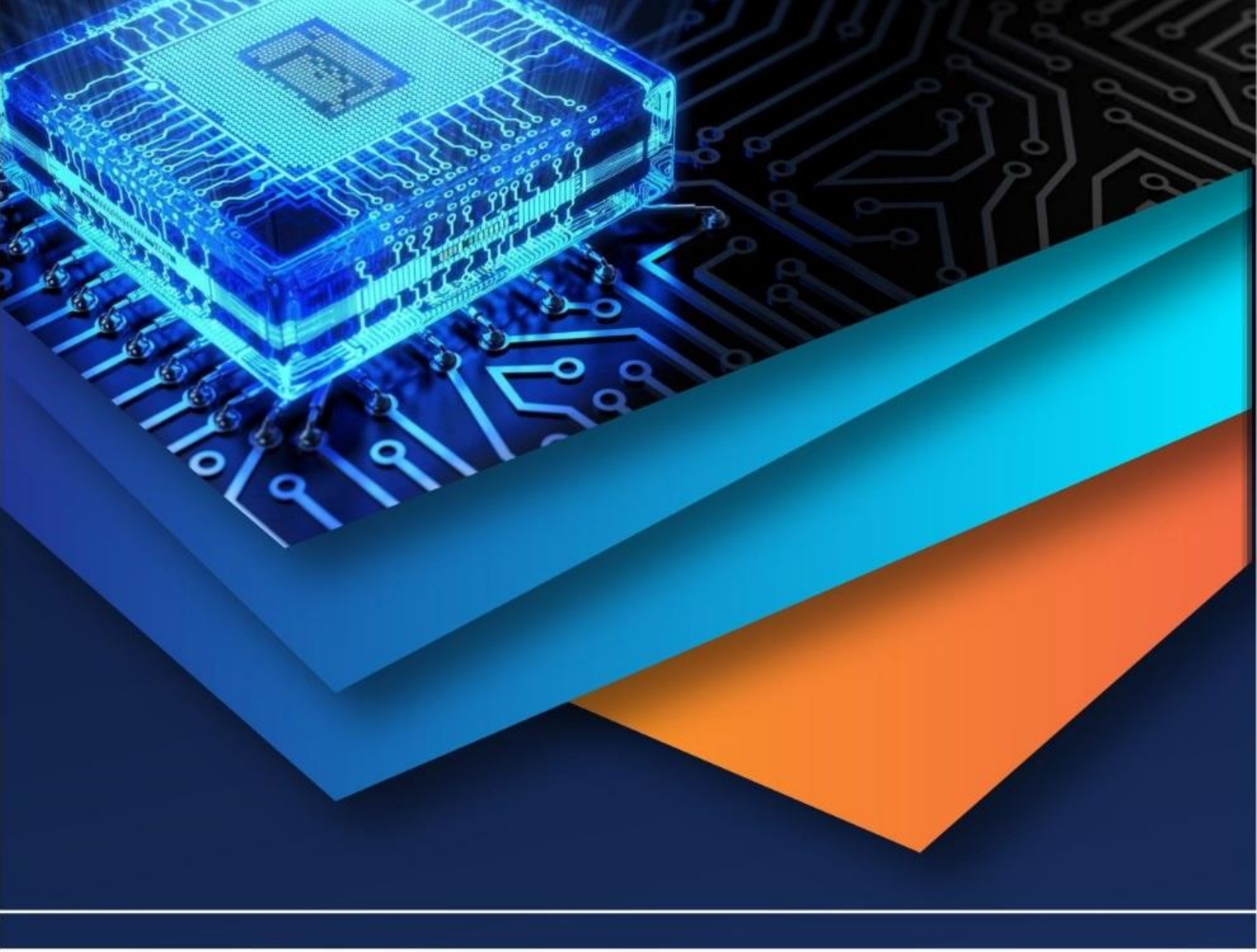
Overall, the system demonstrates how automation and voice interaction can simplify complex software installation tasks while improving efficiency, security, and scalability for both individual users and enterprise environments.

REFERENCES

[1] Zhang, C. et al. (2025). UFO2: The Desktop AgentOS. ACM.
UFO2: The Desktop AgentOS - Microsoft Research
[2] Hui, Z. et al. (2025). WinClick: GUI Grounding with Multimodal Large Language Models. Association for Computational Linguistics.
WinSpot: GUI Grounding Benchmark with Multimodal Large Language Models - ACL Anthology
[3] Zhao, H.H. et al. (2025). WorldGUI: An Interactive Benchmark for Desktop GUI Automation from Any Starting Point. National University of Singapore.
WorldGUI: An Interactive Benchmark for Desktop GUI Automation from Any Starting Point
[4] Zhao, D. et al. (2024). COLA: A Scalable Multi-Agent Framework for Windows UI Task Automation. Academy of Military Sciences.
[2503.09263] COLA: A Scalable Multi-Agent Framework For Windows UI Task Automation
[5] Milliken, L. et al. (2024). Beyond pip install: Evaluating LLM Agents for the Automated Installation of Python Projects. ResearchGate.
(PDF) Beyond pip install: Evaluating LLM Agents for the Automated Installation of Python Projects
[6] Geniola, A., Antikainen, M. & Aura, T. (2018). Automated analysis of freeware installers promoted by download portals. Computers & Security, 77, 209-225.
Automated analysis of freeware installers promoted by download portals - ScienceDirect
[7] Järvinen, T., Karkkonen, K., & Kalliomäki, S. (2022). Automating software installation for cyber security research and testing public exploits in CRATE. Swedish Defence Research Agency.
Automating Software Installation for Cyber Security Research and Testing - DocsLib
[8] Cernat, M., Staicu, A. & Stefanescu, A. (2020). Improving UI Test Automation using Robotic Process Automation. ResearchGate.
(PDF) Improving UI Test Automation using Robotic Process Automation
[9] Cheng, Y.P., Li, C.W. & Chen, Y.C. (2021). Apply computer vision in GUI automation for industrial applications. Mathematical Biosciences and Engineering, 16(6), 7526-7545.
Apply computer vision in GUI automation for industrial applications
[10] Lakshmi, G.R.S. & Chary, V.S.R. (2021). Web Scraping with Python and Selenium. IOSR Journal of Computer Engineering, 21(1), 1-13.
[11] Natale, S. & Cooke, H. (2021). Browsing with Alexa: Interrogating the impact of voice assistants as web interfaces. Media, Culture & Society, 43(6), 1000-1016.
Browsing with Alexa: Interrogating the impact of voice assistants as web interfaces
[12] Sharma, K. et al. (2022). ARA- A Voice Assistant for Disabled Personalities. International Journal of Engineering Applied Sciences and Technology, 7(1), 106-109.
172-175, Tesma0701,IJEAST.pdf
[13] Fendji, J.L.K.E. et al. (2022). Automatic Speech Recognition Using Limited Vocabulary: A Survey. Applied Artificial Intelligence, 36(1), 2095039.
[14] Jafari, A.J. et al. (2023). Dependency Update Strategies and Package Characteristics. ACM Transactions on Software Engineering Methodology.
[15] Han, X. et al. (2021). SIGL: Securing Software Installations Through Deep Graph Learning. Proceedings of the 30th USENIX Security Symposium, 2345-2362.
[16] Y. Zhang et al. (2024) Large Language Model-Brained GUI Agents: A Survey. ACM Computing Surveys, 2024, pp. 1–38.
[17] M. Memon et al. (2001) Model-Based GUI Automation. IEEE Software, 2001, pp. 20–28.
[18] R. C. Bryce et al. (2010) GUI Automation Testing Tools: A Comparative Study. Software Testing, Verification and Reliability, 2010, pp. 1–23.
[19] P. Sharma et al. (2022) Hybrid Desktop Automation Using Selenium and PyAutoGUI. International Conference on Advances in Computing and Communication Engineering (ICACCE), 2022, pp. 312–317.
[20] V. Kulkarni, et al. (2020) Interactive Voice-Controlled Assistant for Desktop Automation. International Journal of Computer Applications, 2020, pp. 15–20.

Dataset:

[1] A. Prasad and S. Chandra. "PhiUSIIL Phishing URL (Website)," UCI Machine Learning Repository, 2024. [Online]. Available: https://doi.org/10.1016/j.cose.2023.103545 last accessed on 2 March 2026.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY