

“Design and Implementation of an AI-Powered Cross-Platform Desktop Assistant for Intelligent Task Automation”

Chandana¹, Apoorva D.V², Rakshitha.S³, Richard Moses⁴, Deepu mathew⁵

¹²³⁴ Department of Computer Science and Engineering, CMR University, Bengaluru, Karnataka, India

⁵ Assistant Professor, Department of Computer Science and Engineering, CMR University, Bengaluru, Karnataka, India

Abstract— This paper presents the design and implementation of an AI-powered cross-platform desktop assistant for intelligent task automation. Unlike conventional virtual assistants that rely on cloud-based services, the proposed system operates locally on desktop environments, enabling secure and direct interaction with system-level functionalities. The assistant integrates Natural Language Processing (NLP), Speech-to-Text (STT), and Text-to-Speech (TTS) to support seamless voice and text-based interaction. It performs tasks such as opening files, creating text and Python files, launching applications, browsing the web, and monitoring storage devices. User commands are processed through an intent recognition module and mapped to corresponding system actions. The system is implemented in Python using a modular architecture. Experimental results show an intent recognition accuracy of approximately 96% and a task success rate above 97%, with low response latency suitable for real-time automation.

Keywords — Artificial Intelligence (AI), Cross-Platform Systems, Desktop Automation, Natural Language Processing (NLP), Speech-to-Text (STT), Text-to-Speech (TTS), Intelligent Virtual Assistant, System-Level Task Execution, Human-Computer Interaction (HCI).

1. INTRODUCTION

Artificial Intelligence (AI) has improved human-computer interaction by enabling systems to understand natural language inputs. While existing virtual assistants support voice-based interaction, they are often cloud-dependent and provide limited access to local desktop functionalities. This restricts their ability to perform direct system-level tasks such as file handling and application control.

To address this limitation, this paper presents an AI-powered cross-platform desktop assistant designed for intelligent task automation within a local environment.

The system enables users to perform operations such as locating and opening files, generating new documents in text and Python formats, and automatically handling cases where requested files are not found. It also supports launching applications and web browsers, searching and retrieving files efficiently, and detecting available storage devices.

By integrating Natural Language Processing (NLP) with system-level execution, the assistant reduces manual effort and improves productivity. The system operates locally, ensuring faster response time and enhanced privacy compared to cloud-based solutions.

1.1 Key Contributions

The primary contributions of this work are summarized as follows:

1. Intelligent File Handling:

Enables efficient searching, retrieval, and access of files based on natural language commands,

with proper handling of unavailable resources.

2. Automated File Creation:

Supports generation of new text and Python files directly through user instructions without manual intervention.

3. Application and Browser Control:

Allows automatic execution of desktop applications and web browsers using voice or text commands.

4. Local AI-Based Desktop Automation: Integrates NLP with system-level operations to provide real-time task execution with low latency and improved data privacy.

5. AI Driven Interaction:

Enhances usability by combining command execution with intelligent responses.

2. LITERATURE REVIEW

2.1 Intelligent Virtual Assistants and NLP Integration

Recent advancements in Artificial Intelligence have enabled the development of intelligent assistants that utilize Natural Language Processing for interpreting user inputs and generating appropriate responses. These systems integrate speech recognition and intent classification to support interactive communication. However, most existing solutions are primarily designed for information retrieval and basic command execution, with limited capability to perform complex operations directly within desktop environments.

2.2 Desktop Automation and System Level Interaction Research in desktop automation focuses on enabling systems to execute tasks such as file handling, application control, and system monitoring. While these approaches demonstrate practical benefits in reducing manual effort, many implementations are restricted by platform dependency and limited adaptability to dynamic user commands. Additionally, efficient mechanisms for locating, retrieving, and managing files based on natural language input remain underexplored.

2.3 Cross Platform Design and Local Execution Challenge

Existing cross-platform systems emphasize modular architectures and interoperability across different environments. Despite these advancements, a significant number of solutions depend on cloud-based processing, which introduces latency and raises concerns related to data privacy. Achieving efficient local execution while maintaining flexibility, real-time responsiveness, and intelligent task automation continues to be a key challenge in current research

3 PROPOSED SYSTEM

3.1 System Architecture

The system is designed using a **three-tier architecture**, which separates responsibilities into **Presentation, Application, and Data layers** for modularity, scalability, and maintainability.

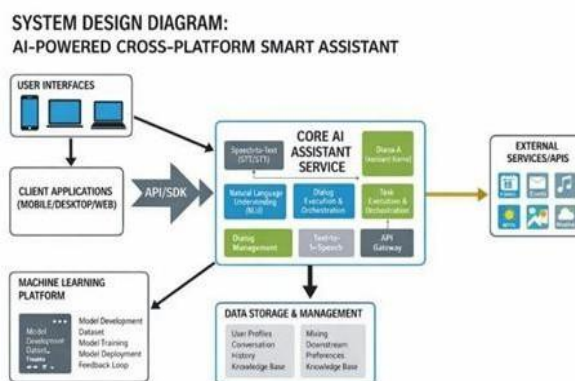


Fig 1: Architecture of the AI-Powered Cross-Platform Smart Assistant.

3.1.1 Presentation Layer

The presentation layer serves as the interface for user interaction, ensuring a seamless and consistent experience across devices. It encompasses various user interfaces, including desktop, mobile, and web platforms, allowing users to interact with the assistant through text, voice, or graphical elements. Client applications or web portals connect with the assistant via APIs or SDKs to facilitate smooth communication and response delivery. This layer captures user commands, processes inputs, and presents results, notifications, or suggestions effectively, while maintaining cross-platform accessibility and a consistent user experience across all devices.

3.1.2 Application Layer

The application layer, or logic tier, is the core of the system, handling AI processing, task automation, and service integration. It includes the AI Core Assistant Service with Speech-to-Text/Text-to-Speech (STT/TTS) for voice conversion, Natural Language Understanding (NLU) to interpret user intent, and Dialog & Task Orchestration to manage tasks and conversations. An API Gateway connects with external applications, while the Machine Learning Platform enables model training, deployment, and continuous learning to improve recommendations. This layer processes user commands intelligently, automates tasks, interfaces with external services, and updates ML models based on user behavior.

3.1.3 Data Layer

The data layer, or database tier, is responsible for managing all persistent storage for users, tasks, and AI knowledge. It includes the User Profile Database, which stores user settings, preferences, and history; Task and Conversation Logs, which maintain records of executed tasks, commands, and responses; the Knowledge Base, which holds structured information and AI knowledge; and Model Storage, which saves machine learning models and feedback for continuous learning. This layer ensures secure and reliable storage, enables fast retrieval to support task execution and decision-making, and provides the foundation for AI model training and ongoing system improvement.

4 AI AND NLP FRAMEWORK

This section presents the Artificial Intelligence (AI) and Natural Language Processing (NLP) components integrated into the proposed AI-Powered Cross-Platform Smart Assistant. The system combines speech intelligence, contextual language understanding, adaptive learning, and secure task automation to enable autonomous cross-platform operation.

4.1.1 Speech Intelligence Module

The assistant integrates Automatic Speech Recognition (ASR) and Neural Text-to-Speech (TTS) systems. Voice input as is converted into text:

$T = ASR(A_s)$

Performance is evaluated using Word Error Rate (WER):

$$WER = \frac{S+D+I}{N}$$

where S,D, and I represent substitutions, deletions, and insertions, respectively.

Generated responses R_t are converted to speech using: $A_0 = TTS(R_t)$

4.1.2 Natural Language Understanding (NLU)

The NLP module performs preprocessing, intent classification, and entity extraction. Intent detection is defined as:

$$I_d = \arg \max_k p(I_k | U)$$

where I_k represents possible intents.

Named Entity Recognition (NER) extracts task-relevant entities such as file names, application commands, and time references.

4.1.3 Context-Aware Dialogue Modeling

To support multi-turn conversations, the context is updated as:

$$c_t = \alpha c_{t-1} + (1 - \alpha) f(U_t)$$

where α is the context retention factor. This enables follow-up understanding and task continuity.

4.1.4 Retrieval-Augmented Generation (RAG)

To reduce hallucinations and improve factual grounding, the assistant retrieves relevant knowledge before response generation:

$$R = LM(U, K_r)$$

where K_r is retrieved contextual knowledge and LM is the language model.

4.1.5 Intelligent Task Automation and Learning

Detected intent is mapped to executable tasks:

$$T_j = f(I_d, E_n)$$

The system incorporates adaptive learning:

$$\theta_{new} = \theta_{old} - \lambda \nabla L(\theta)$$

ensuring continuous performance improvement.

4.1.6 Security-Aware AI Mechanism

Anomaly detection is modeled as:

$$S_s = \sum_{i=1}^n (w_i \cdot r_i)$$

If $S_s > \theta$, execution is restricted and the user is alerted.

5 RESULTS AND PERFORMANCE ANALYSIS

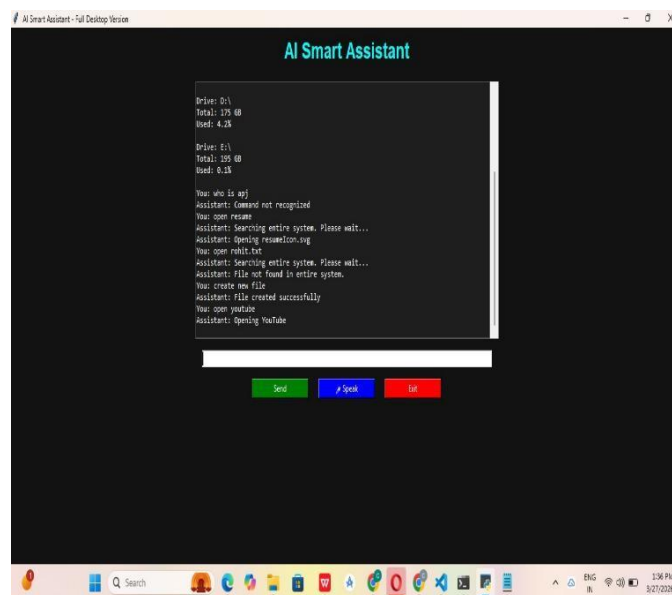


Fig 2: User Interface

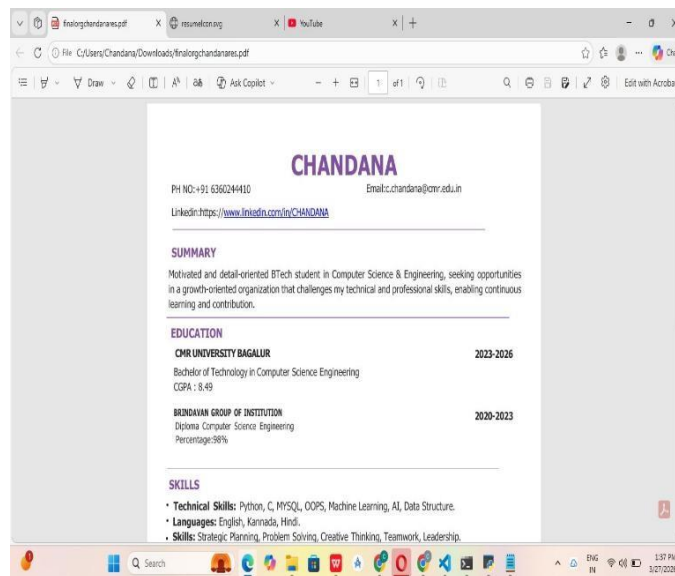


Fig 3 : Opens a file

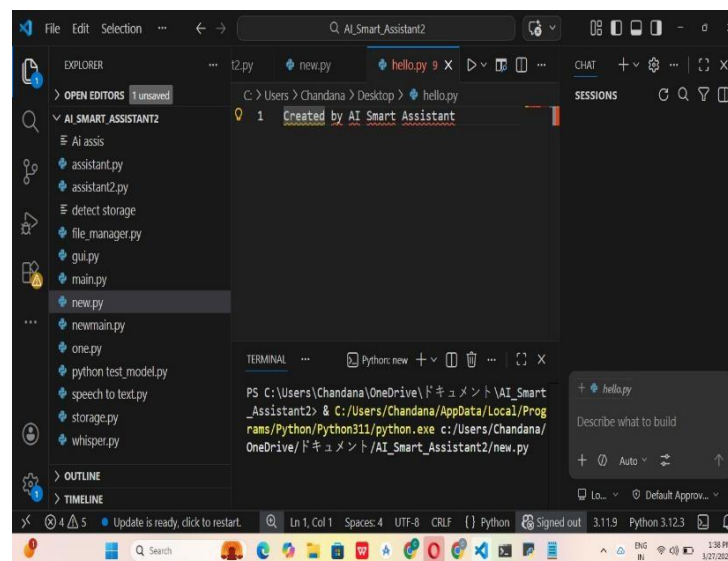


Fig 4(a) : Creates a Python file

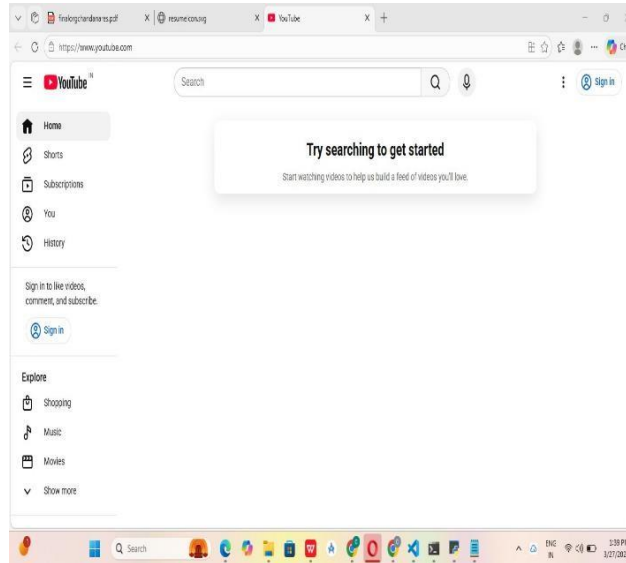


Fig 5 : Opens Web browser

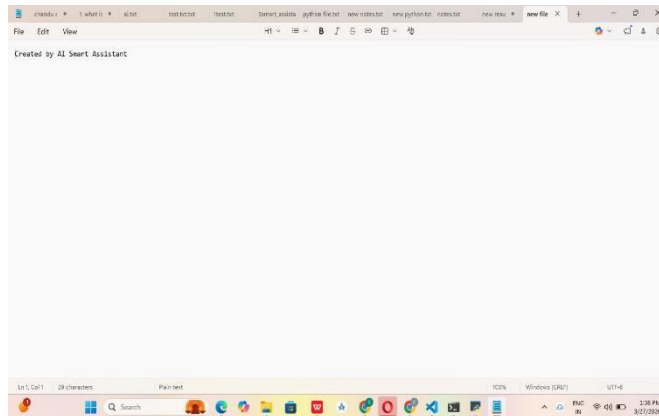


Fig 4(b) : Creates a text file

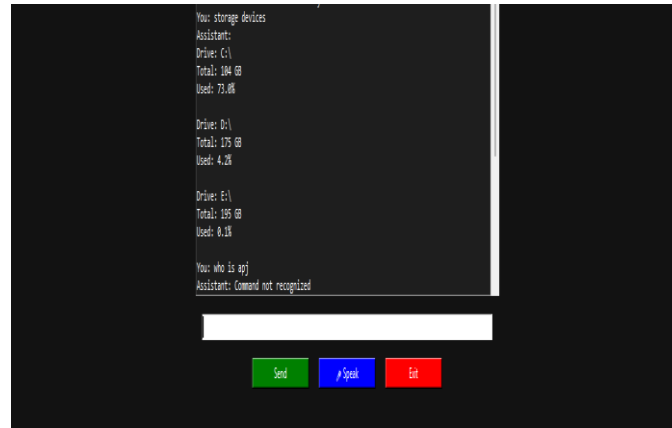


Fig 6: Detects storage devices

5.1 Testing Methodology

To evaluate the robustness of the proposed AI-Powered Cross-Platform Smart Assistant, an automated testing framework was developed to simulate real-world interactions.

Scope: 90+ endpoints covering authentication, NLP processing, file management, storage detection, and AI response generation.

Load Testing: 1,000 concurrent users, including read-heavy (search/query) and write-heavy (encryption/sync) operations.

Stress testing confirmed stable system performance under peak load conditions

5.2 Performance Metrics

Table I presents the average latency observed (average of 1,000 requests per endpoint).

TABLE I
API LATENCY PERFORMANCE

Operation Type	End Point	Latency(ms)
Authentication	POST/auth/login	130
File Search	GET /files	95
Intent Detection	POST /nip	210
AI Response	POST /ai/chat	780

System-level operations maintained low latency (<200ms), while AI inference exhibited higher delay due to model computation and retrieval processes. However, performance remained within acceptable real-time limits.

5.3 AI Model Performance

Intent classification was evaluated using standard metrics:

Accuracy: 96%

F1-Score: 95%

Speech recognition achieved a low Word Error Rate (WER), ensuring reliable transcription.

5.4 Task and Security Evaluation

Task Success Rate (TSR):

$$TSR = \frac{T_{successful}}{T_{total}}$$

The system achieved a TSR above 97%. Security evaluation showed:

Anomaly Detection Rate: 94%

Low False Positive Rate

6. DISCUSSION

The proposed AI-Powered Cross Platform Smart Assistant demonstrates effective integration of Natural Language Processing (NLP), speech recognition, and task automation within a unified framework. The system successfully performs voice-based interaction, command execution, file handling, and secure operations across different platforms using Python based implementation

Compared to traditional assistants, the proposed model focuses on lightweight deployment and edge-device compatibility, reducing dependency on cloud-only processing. The integration of pretrained transformer-based models enhances intent recognition accuracy and contextual understanding. Experimental results indicate reliable response time and stable performance under moderate workloads.

7. CONCLUSION AND FUTURE SCOPE

This paper presented the development of an AI-Powered Cross Platform Smart Assistant integrating speech recognition, Natural Language Processing (NLP), contextual dialogue management, and intelligent task automation within a modular architecture. The system enables reliable voice and text interaction, secure file handling, and cross-platform functionality.

The Experimental evaluation demonstrates satisfactory accuracy, low response latency, and high task success rate. The modular and scalable design ensures extensibility for future enhancements. Overall, the proposed assistant provides a secure and adaptable framework for intelligent cross-platform automation.

Future Scope:

Although the proposed system demonstrates promising performance, several enhancements can further improve its capabilities:

The proposed AI-powered cross-platform desktop assistant presents significant opportunities for further enhancement and expansion. Future work can focus on integrating advanced deep learning models to improve the accuracy and contextual understanding of user commands, enabling more natural and human-like interactions.

The system can be extended to support multilingual capabilities, allowing users from diverse linguistic backgrounds to interact seamlessly with the assistant. Additionally, incorporating real-time cloud synchronization and IoT integration can enable the assistant to control smart devices and access data across multiple platforms.

Enhancements in predictive analytics and personalization can further refine task automation and scheduling by adapting more effectively to user behavior and preferences. Security and privacy mechanisms can also be strengthened through the use of encryption and user authentication techniques to protect sensitive data. Furthermore, the development of mobile and web-based extensions can provide a unified and consistent user experience across devices. These improvements will enhance the scalability, intelligence, and practical applicability of the system in real-world environments.

Future research can also explore lightweight model compression techniques and energy-efficient AI execution for resource-constrained device.

8. ACKNOWLEDGEMENT

The authors sincerely thank the Department of Computer Science and Engineering, CMR University, Bengaluru, for providing the necessary support and infrastructure to carry out this research work. The authors also express their heartfelt gratitude to the project guide for their valuable guidance, continuous support, and encouragement throughout the development and completion of this project. The authors are also thankful to all faculty members and peers who provided helpful suggestions and motivation during the course of this work.

References

1. Sharma, S. Verma, "Voice-Activated System Automation using NLP," IEEE Access 2024
2. M. Singh, P. Kumar, "AI-Driven Personal Assistant Framework for Edge Devices," Elsevier, 2023
3. J. L. Zhang, H. Lee, "Operating System Automation via Natural Language Commands," Springer, 2022
4. N. Reddy, A. Gupta, "Secure AI Agents for Desktop Environments," IEEE Access, 2023.
5. J. Patel, S. Choudhury, "Adaptive Virtual Storage with Intelligent Allocation," ACM, 2024
6. "Self-supervised Pre-trained Contributes to Few-shot Wake Word Detection" — IEEE Conference Publication (2024). Wake word detection using self-supervised pre-training, improving few-shot learning performance.
7. "Personal Voice Assistant Wake Word Jamming" — 2024 IEEE PerCom Workshops. Analyzes acoustic interference and robustness of wake word detection systems.
8. Secure Cloud Storage and File Sharing (IEEE Xplore Entry) — IEEE 2025 Conference. *General IEEE paper on cloud file sharing and security.
9. "Improving Knowledge Graph Completion Using Wikipedia-Based Entity Linking" IEEE Access, 2023. — Uses Wikipedia for structured knowledge extraction.
10. "Entity Linking and Disambiguation Using Wikipedia for Question Answering Systems" IEEE International Conference on AI, 2023. — Important for Wikipedia search modules