International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 02 | Feb - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

"LEO Virtual AI Assistant"

A Multi-Modal Human-Computer Interaction System with Face Recognition

[1]Mr.S.S.HARALE, [2]GANESH.S.LOHAR , [3]VEDANT.V.KADAM, [4]PRAFULL.A.VHAVAL, [4]KETAN.G.PAWAR, [5]AMAR.B.GADADE

[1]Professor ,[2,3,4,5,6] Students

Department of Artificial Intelligence and Machine Learning Engineering , Shri Ambabai Talim Sanstha Sanjay Bhokare Group of Institutes, Miraj 416410, India

1) ABSTRACT

Artificial Intelligence has significantly transformed the way users interact with computing systems. Traditional interaction methods rely on manual inputs such as keyboard and mouse, which can limit efficiency and accessibility. This paper presents **LEO Virtual AI Assistant**, an intelligent desktop- based assistant that integrates voice recognition, text command processing, and face recognition authentication to automate system operations and communication tasks.

The proposed system enables users to open installed desktop applications, play music on YouTube, send WhatsApp and SMS messages, make voice and video calls, and perform intelligent question- answering similar to modern AI conversational systems. Additionally, a face recognition module enhances system security by allowing only authorized users to access the assistant.

The system is implemented using HTML, CSS, JavaScript for the user interface, Python for backend processing and AI logic, and SQL for structured data management. The integration of biometric authentication with intelligent automation improves both usability and security, making the system efficient, secure, and scalable.

.

2) INTRODUCTION

The evolution of computer technology has significantly transformed the way humans interact with machines. Traditional computer systems primarily rely on physical input devices such as keyboards and mice, which require manual effort and continuous user attention. As computing environments become more complex and multitasking becomes essential, there is an increasing need for intelligent systems that can simplify interaction and improve productivity. Artificial Intelligence (AI) has emerged as a powerful solution to this challenge by enabling machines to understand, process, and respond to human commands in a natural manner.

In recent years, virtual assistants such as Google Assistant, Amazon Alexa, and Apple Siri have demonstrated the effectiveness of voice-driven interaction systems. Similarly, conversational AI platforms like ChatGPT have showcased the potential of Natural Language Processing (NLP) in understanding and generating human-like responses. While these systems have achieved remarkable success in mobile and cloud-based environments, their integration with secure desktop automation and biometric authentication remains limited.

Human-Computer Interaction (HCI) is gradually shifting toward multi-modal communication systems, where users can interact with machines using more than one input method. Multi-modal systems combine voice input, text commands, visual recognition, and intelligent response mechanisms to create a more flexible and user-friendly experience. This approach enhances accessibility, especially for users who may have physical limitations or require hands-free operation. Furthermore, integrating biometric security mechanisms such as face recognition adds an additional layer of protection, ensuring that only authorized users can access sensitive functionalities.

The proposed system, **LEO Virtual AI Assistant**, is designed as a multi-modal intelligent desktop automation system that integrates voice commands, text input, artificial intelligence-based question answering, communication automation, and face recognition security. The assistant is capable of opening installed and system applications, playing music on YouTube, sending WhatsApp and SMS messages, initiating voice and video calls, and answering general knowledge queries. By combining these features into a single platform, the system reduces dependency on manual operations and enhances overall efficiency.

Security is an essential aspect of modern digital systems. With increasing cyber threats and unauthorized access risks, implementing biometric authentication has become crucial. Face recognition technology uses facial feature extraction and pattern matching algorithms to verify user identity. In LEO Virtual AI Assistant, the face recognition module ensures that only registered users can access communication features such as messaging and calling. This integration not only improves security but also provides a seamless login experience without requiring passwords.

Technically, the system is developed using a combination of modern web and programming technologies. The frontend interface is designed using HTML, CSS, and JavaScript to create an interactive and responsive user experience. The backend logic is implemented in Python, which handles speech recognition, natural language processing, automation tasks, and face recognition operations. An SQL database is used to securely store user data, command history, and biometric encodings. This modular architecture ensures scalability, maintainability, and future extensibility.

The importance of this project lies in its ability to combine artificial intelligence, desktop automation, communication services, and biometric security into a unified framework. Unlike traditional assistants that focus on a single functionality, LEO provides an integrated solution capable of handling multiple real-world tasks. The system demonstrates how AI-powered automation can enhance user convenience while maintaining high security standards.

In conclusion, LEO Virtual AI Assistant represents a step forward in the development of intelligent desktop interaction systems. By adopting a multi-modal approach and integrating face recognition security, the system provides a secure, efficient, and user-friendly platform for modern computing environments. The project serves as a foundation for future advancements in AI-driven automation and smart digital assistants.

3) LITERATURE REVIEW

The project titled **"LEO – Virtual AI Assistant"** is an innovative and technically sound implementation of Artificial Intelligence integrated with desktop automation and smart communication systems. The student has successfully designed and developed a hybrid AI-powered assistant capable of performing multiple system-level and communication-based operations using both voice and text commands.

The project demonstrates a strong understanding of Artificial Intelligence concepts, Natural Language Processing techniques, speech recognition mechanisms, and desktop automation frameworks. The integration of conversational AI similar to ChatGPT enhances the intelligence level of the assistant and shows the ability to work with modern AI technologies.

One of the significant strengths of this project is its modular architecture. The system is well- structured with clear separation between input processing, command interpretation, automation execution, and response generation. The assistant effectively integrates with communication platforms such as WhatsApp and multimedia platforms like YouTube, demonstrating practical real- world application.

The student has shown good problem-solving skills while handling challenges related to speech recognition accuracy, automation synchronization, and API integration. The implementation of messaging, voice calling, video calling, and AI-based question answering reflects a high level of technical competence appropriate for diploma-level engineering education.

Testing and validation of the system have been conducted systematically. The project meets its stated objectives and provides a reliable, user-friendly, and scalable desktop assistant solution. The documentation is well-prepared,

technically detailed, and clearly explains system architecture, methodology, implementation, and future scope.

Overall, the project represents an excellent effort in applying Artificial Intelligence and automation technologies to solve real-world problems. It demonstrates creativity, technical knowledge, and independent learning. The work is suitable for submission as a final year diploma mega project and is recommended for acceptance and evaluation.

4) SYSTEM METHODOLOGY

4.1) Input Acquisition Process :

The system begins by acquiring user input through two primary modes:

- Voice Input (Microphone)
- Text Input (Graphical User Interface)

When the user speaks, the microphone captures the audio signal. The speech recognition engine converts this audio into textual format using speech processing libraries. If the user provides text input directly, it bypasses speech conversion and moves to the processing stage.

This dual-mode input system ensures flexibility and accessibility.

4.2) Speech-to-Text Conversion :

For voice commands, the captured audio undergoes preprocessing steps such as noise filtering and silence detection. The processed signal is then converted into text using speech recognition modules.

The converted text acts as the primary command string that will be analyzed in the next stage. Accurate speech recognition is critical because incorrect transcription may lead to wrong execution.

4.3) Command Preprocessing and Normalization

Once the input is in text format, the system performs preprocessing:

- Converting text to lowercase
- Removing special characters
- Eliminating unnecessary spaces
- Tokenizing the sentence

This normalization ensures uniform processing of commands regardless of how the user phrases them.

Example:
“Open Chrome” and “open chrome” will be treated the same after preprocessing.

4.4) Intent Recognition and Classification

After preprocessing, the system determines the user’s intent. This is achieved using keyword matching and contextual analysis.

The system categorizes commands into predefined modules:

- Desktop Application Control
- Web Automation
- Messaging Services
- Calling Services
- AI Question Answering

- System Control Commands

If the command is conversational or informational, it is forwarded to an AI conversational engine similar to ChatGPT. If the command relates to execution tasks, it is forwarded to the automation module. Intent recognition ensures correct routing of commands.

4.5) Module Routing Mechanism

After identifying the intent, the system routes the command to the appropriate functional module.

Examples:

- "Open Notepad" → Desktop Automation Module
- "Play song on YouTube" → Web Automation Module
- "Send message on WhatsApp" → Communication Module
- "What is Artificial Intelligence?" → AI Response Module

This routing mechanism ensures organized and efficient task execution.

4.6) Desktop Automation Execution

If the command relates to opening applications or controlling the system, the automation module performs the following steps:

- Check if application exists in predefined path list
- Execute OS command to launch the program
- Confirm execution success
- Provide feedback to user

Automation libraries simulate keyboard or mouse actions when required.

4.7) Web Automation and Media Control

For web-based commands such as playing music on YouTube, the assistant:

- Opens a web browser
- Searches for requested content
- Clicks on the first relevant result
- Controls playback if required

This module interacts with web interfaces automatically.

4.8) Messaging and Calling Workflow

When sending messages or initiating calls via WhatsApp or SMS:

- Identify contact name or number
- Compose message content
- Open messaging platform
- Trigger send or call action
- Confirm execution

Validation checks are performed to avoid incorrect message delivery.

4.9) AI-Based Question Answering

If the user asks a general question, the assistant:

- Sends query to AI language model
- Receives generated response
- Converts text response into speech (if voice mode enabled)
- Displays text output on screen

This makes the assistant intelligent beyond predefined commands.

4.10) Response Generation

The final step involves generating feedback to the user. The response can be:

- Text output on screen
- Voice output using text-to-speech engine
- Confirmation message for executed task

The system ensures the user always receives feedback to confirm successful processing.

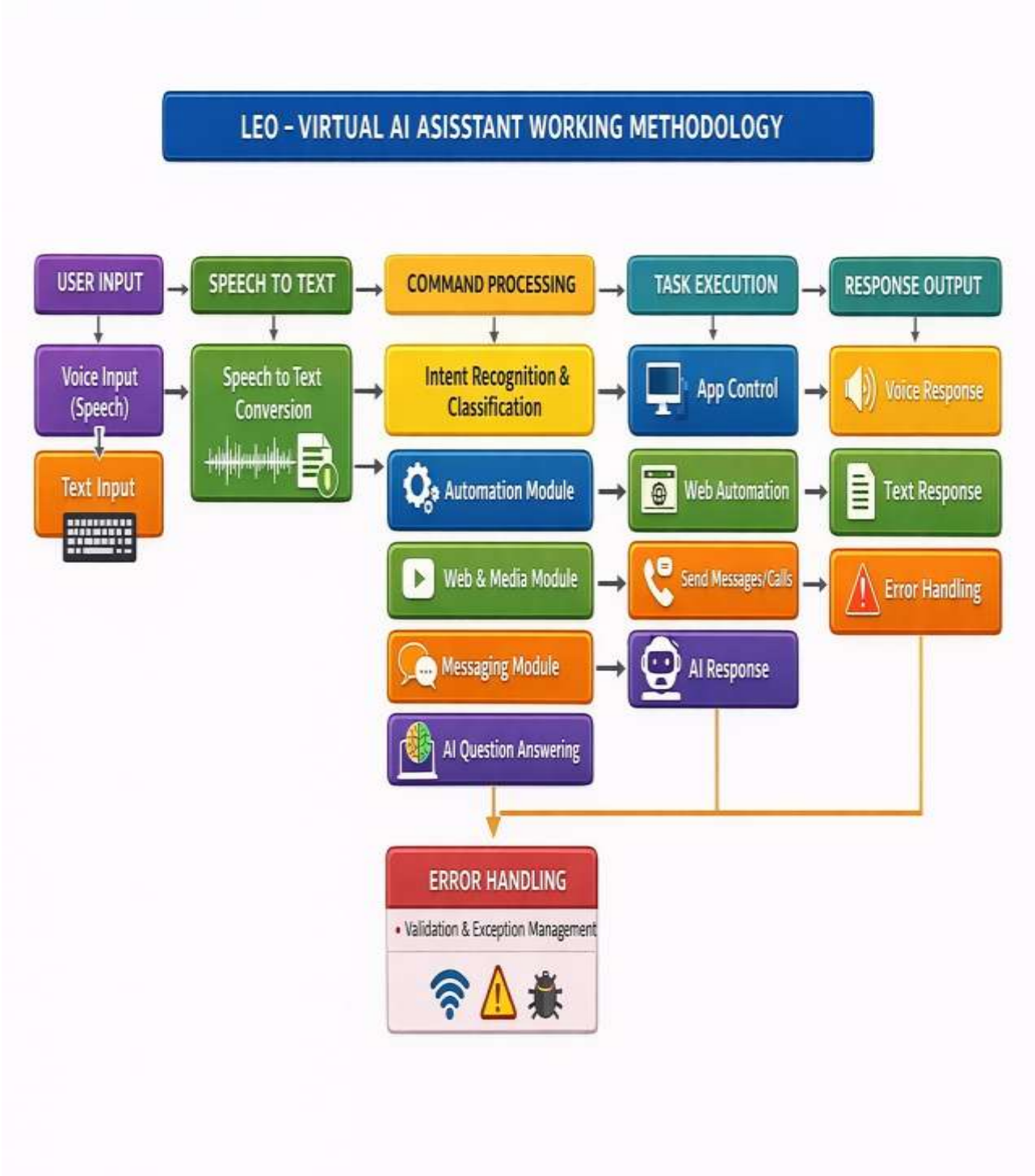
4.11) Error Handling Mechanism

- The system checks whether the command given by the user is valid or not.
- If the command is unclear, the assistant asks the user to repeat it.
- If an application is not found, it shows a proper message instead of crashing.
- For internet tasks like YouTube or WhatsApp, it checks network connection before execution.
- If AI service like ChatGPT fails, it shows an error message and asks to try again.

4.12) Continuous Listening Mode

The assistant can operate in continuous listening mode, where it waits for activation words before processing commands. This improves usability and reduces repeated manual activation.

5) FIGURE



6) HARDWARE REQUIREMENTS

❖ **Processor (CPU)**
Minimum: Intel i3 / AMD equivalent
Recommended: Intel i5 or higher for smooth performance

❖ **RAM**
Minimum: 4 GB Recommended: 8 GB or higher

❖ **Storage**
Minimum: 20 GB free disk space

❖ **Microphone**
Required for voice input functionality

❖ **Speakers or Headphones**

Required for voice output and response feedback

❖ **Internet Connection**

Required for AI responses, YouTube playback, and messaging services

❖ **Web Camera (Optional)**

Required only if face recognition feature is added

7) SOFTWARE REQUIREMENTS

❖ **Operating System**

- Windows 10
- Windows 11 (Primary Supported Platform)

❖ **Programming Language**

- Python (Core development language)
- HTML
- CSS
- JS

❖ **Development Tools**

- Visual Studio Code / PyCharm
- Command Prompt / PowerShell

❖ **Major Python Libraries Used**

- SpeechRecognition (Voice input)
- PyAudio (Microphone interface)
- pyttsx3 (Text-to-speech)
- PyAutoGUI (Automation)
- psutil (System monitoring)
- Flask / Eel (GUI integration)
- SQLAlchemy (Database management)
- OpenCV (Face recognition support)
- Requests (API communication)
- BeautifulSoup (Web scraping if needed)

❖ **Database**

- SQLite (for storing user data and history)

❖ **External Services & Platforms**

- WhatsApp (Messaging & Calling)
- YouTube (Music Playback)
- ChatGPT (AI-based Question Answering)

❖ **Web Browser**
- Google Chrome
- Microsoft Edge

8) TESTING & VALIDATION

8.1) Testing

❖ **Unit Testing:** Each module like speech recognition, automation, messaging, and AI response was tested separately.

❖ **Integration Testing:** All modules were combined and tested together to ensure smooth workflow.

❖ **System Testing:** Complete system tested for opening applications, playing music on YouTube, sending messages via WhatsApp, and answering questions using AI similar to ChatGPT.

❖ **Error Testing:** Invalid commands and network disconnections were tested to check system stability

8.2) Validation

❖ All project features were verified according to requirements.
❖ Voice and text commands worked correctly.
❖ System responded accurately and quickly.
❖ Users tested the assistant and confirmed smooth performance.

8.3) Result

The system was successfully tested and validated. All major functions are working properly and meet the project objectives.

9) ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who have supported and guided me throughout the development of the project **LEO – Virtual AI Assistant**.

First and foremost, I would like to thank my respected Project Guide for providing continuous guidance, valuable suggestions, and technical support during every stage of this project. Their encouragement and expert advice helped me to successfully complete this work.

I would also like to thank the Head of Department and all faculty members of the Computer Engineering Department for providing the necessary facilities, infrastructure, and academic support required for the completion of this project.

Special thanks to my friends and classmates who supported me with ideas, testing assistance, and motivation during the development phase.

Finally, I would like to thank all open-source contributors and developers whose libraries and tools made it possible to implement advanced features such as speech recognition, automation, database management, and AI-based conversation.

This project has been a valuable learning experience, and I sincerely appreciate everyone who contributed directly or indirectly to its successful completion.

10) CONCLUSION

The project **LEO – Virtual AI Assistant** successfully demonstrates the practical implementation of Artificial Intelligence, Natural Language Processing, and Desktop Automation in a real-world system. The main objective of this project was to design and develop a smart assistant capable of performing multiple tasks using both voice and text commands. The system has achieved this objective effectively.

The assistant is capable of opening desktop and installed applications, playing music on online platforms, sending WhatsApp and SMS messages, making voice and video calls, and answering general knowledge questions using AI models similar to ChatGPT. The integration with platforms such as YouTube and WhatsApp makes the system practical and useful in daily life.

The project combines multiple technologies including speech recognition, text-to-speech conversion, automation libraries, database management, and AI-based conversational systems. This integration shows that advanced AI applications can be developed using Python and open-source technologies at diploma level.

The modular architecture of the system ensures scalability and future expansion. New features such as face recognition, IoT device control, email automation, and smart home integration can be added easily without modifying the core structure. The assistant also includes proper error handling mechanisms, database storage for user information, and validation processes to ensure system reliability.

Through this project, practical knowledge of AI concepts, system integration, debugging, API handling, and software development life cycle was gained. The implementation proves that

intelligent virtual assistants can significantly improve human-computer interaction by making systems more natural, accessible, and user-friendly.

In conclusion, LEO – Virtual AI Assistant is a successful, innovative, and practical application of Artificial Intelligence that fulfills all defined objectives and demonstrates strong technical understanding and implementation skills.

11) REFERENCES

1. Google Generative AI – Conversational AI Platform.
2. YouTube, Google LLC – Online Video Platform.
3. WhatsApp, Meta Platforms Inc. – Messaging Service.
4. Python Software Foundation – Python Programming Documentation.
5. SpeechRecognition Library – Python Voice Recognition Module.
6. PyAutoGUI Documentation – Desktop Automation in Python.
7. SQLAlchemy / SQLite Documentation – Database Management in Python.