Journal of Artificial Intelligence and Capsule Networks (ISSN: 2582-2012)
www.irojournals.com/aicn/

Smart Assistant for Essential Voice Interaction

(EVI Assistant)

Sekar S.[1], Abishek Aravinth K.[2], Balamurugan M.[3], Elavarasan D.[4], Ganesh Kumar M.[5]

[1]Associate Professor, [2-5]Students Department of Information Technology, SRM Valliammai Engineering college, Kattankulathur, India

Email: [1]sekars.it@srmvalliammai.ac.in, [2]abishekaravinthk@gmail.com

Abstract

Voice assistants have emerged as a significant trend in contemporary society, revolutionizing human-computer interaction by processing and responding to verbal commands. By utilizing artificial intelligence, these programs efficiently interpret voice input to perform a wide range of tasks. In today's fast-paced world, voice assistants are essential tools that enhance productivity, accessibility, and convenience. For instance, Google's smartphone assistant, which has become familiar to children due to increased smartphone usage during the pandemic, and Amazon's Alexa, which controls household devices through the Internet of Things, showcase their broad utility. The research provides a comprehensive overview of voice assistants, with a focus on the various Python packages used in their development. Emphasizing the accessibility benefits for individuals with physical disabilities, voice assistants can perform tasks such as answering questions, setting reminders, and controlling smart devices.

Keywords: Speech Recognition, Natural Language Processing (NLP), Python Packages, Voice Assistant.

1. Introduction

The system aims to develop a web application with a personalized assistant that users can control through voice or text. This assistant offers a wide range of functions, including call

DOI: https://doi.org/10.36548/jaicn.2024.3.002
Received: 07.06.2024, received in revised form: 02.07.2024, accepted: 13.07.2024, published: 27.07.2024

handling, text transformation, email exchange, alarms, event management, location services, music playback, weather updates, Google/Wikipedia searches, a chatbot, camera use, Bing translation, Bluetooth support, a help menu, and integration with Windows to Azure. It is designed to be especially beneficial for the elderly, individuals with disabilities, and children [11,12].

Emerging technology trends like virtual reality, augmented reality, voice interaction, and IoT are reshaping digital experiences, with voice control being a significant AI-driven advancement. Advancements in AI enable machines to perform tasks independently, as seen in voice assistants like Siri, Google Assistant, Cortana, and Alexa. Voice commands streamline tasks, and voice searches were projected to make up 50% of searches by 2022, surpassing text searches. smart virtual assistants, a product of increasing AI sophistication, handle intricate tasks such as email management, understanding user intent, automating processes, and delivering personalized responses. This research originated from the idea of utilizing publicly available web data to create a virtual assistant with smart decision-making capabilities for everyday activities [13,15].

Developing a web application with a personalized assistant that uses voice or text control marks a significant leap in enhancing accessibility and efficiency in digital interactions. With a wide range of functions, from call handling to music playback and email exchange to event management, this system caters to diverse user needs. Its integration with services like Google, Wikipedia, and Bing Translation enhances the user experience by providing access to an information and resources. Features such as location services, weather updates, and Bluetooth support add convenience and enhance the application's practical utility in everyday scenarios. For individuals with disabilities or the elderly who may face challenges with traditional device interfaces, this personalized assistant is a valuable tool for navigating modern technology seamlessly.

2. Related Work

The research proposed a Python-based voice assistant utilizing a speech-to-text (STT) module, incorporating API calls and system commands. This development enables users to

execute commands through voice without keyboard interaction, suitable for hybrid platforms. However, some limitations are noted, particularly with system calls lacking robust support [1].

The research showcased a Desktop Assistant AI built in Python, featuring IoT capabilities and incorporating the features of Artificial Intelligence (AI) along with an SQLite database. The research includes a robust database connection and query framework. However, the absence of API calls and system calls represents a limitation in the project's functionality [2,3].

The voice-based assistance system uses Python as a backend, supporting system calls, API calls, and incorporating various features. The research demonstrates good responsiveness with API calls but requires improvement in understanding and reliability [4].

The voice enabled personal assistant built using Python boasts a well-supported library, acknowledging that not every API can convert raw JSON data into text seamlessly. However, there is a noted delay in processing request calls that requires attention [5].

The research focuses on the use of an AI-based Voice Assistants, that is designed to provide an accurate response to user requests. Notably, the system includes a feature enabling appointment scheduling through voice commands. However, it is essential to highlight that the project lacks API call integration [6].

The study analyses how to enhance guest experiences and services in hotel rooms by deploying voice-activated AI assistants. This is important during the COVID-19 epidemic in particular since voice assistants lessen the need for direct interaction, promoting safety[7].

To verify the authenticity and reliability of the data collected, the study "Ok Google: Using Virtual Assistants for Data Collection in Psychological and Behavioral Research" suggests using a survey tool developed as an add-on for Google Assistant. The tool's ability to define synonyms and potential responses for various question types makes it useful for examining individual behavior in emotional and psychological research [8].

The use of a voice assistant to control housekeeping operations emphasizes the importance of voice assistants in smart homes. Voice assistants can operate home appliances with voice commands, enhancing security with smart locks. However, a reliable internet connection is essential; otherwise, users might lock themselves out of their own homes [9].

The voice assistant explores the psychological reactions to the system's human-like behaviours. The assistant features IoT capabilities and can place orders for users. However, it relies heavily on the speaker's ability to present decision alternatives in voice dialogues and lacks system calls [10].

3. Proposed System

Step 1: The proposed system aims to present an efficient system for implementing a personal voice assistant. This system utilizes the Speech Recognition library, which offers many built-in functions, enabling the assistant to understand user commands and respond with voice feedback using Text-to-Speech functions.

Step 2: When the assistant captures a voice command from the user, underlying algorithms convert the voice input into text. Based on the keywords present in the text, the assistant performs the corresponding action.

Step 3: Libraries like Random and others are employed for different technologies. The OS library, for example, is used to implement operating system-related functionalities, such as shutting down or restarting the system.

Step 4: Pyttsx3, a text-to-speech conversion package in Python that operates offline and is compatible with both Python 2 and Python 3 is integrated, is used for system voice capabilities in addition to sapi5.

Step 5: A speak function is defined to allow the program to vocalize its outputs. Additionally, a function is implemented to capture voice commands using the system microphone. The main function consolidates all these capabilities, defining the comprehensive functionality of the voice assistant. Figure 3.1 shows the architecture diagram. Figures 3.2 and 3.3 show the use case and sequential diagrams, respectively, of the proposed work.

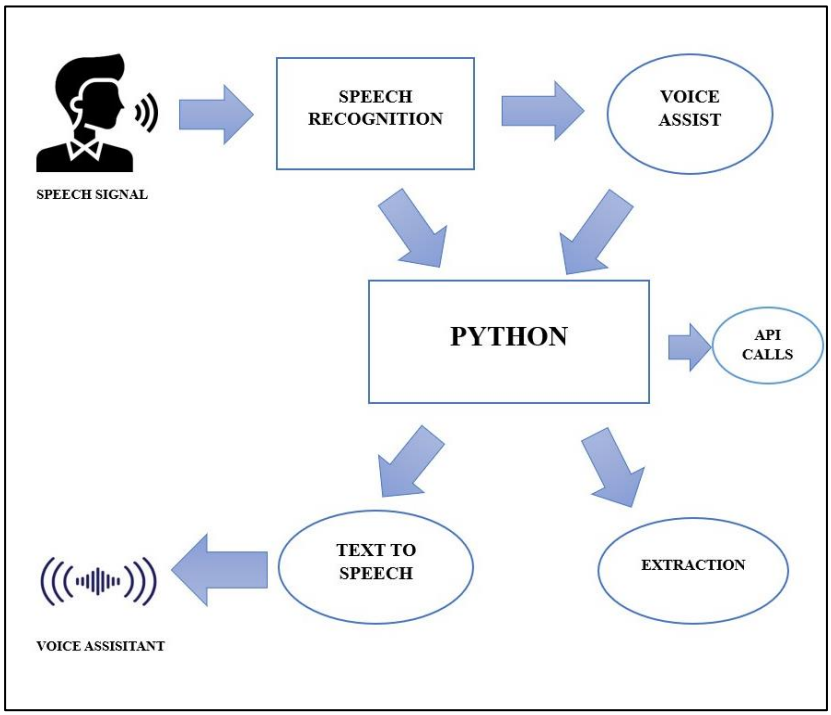

Figure 3.1. Architecture Diagram of Proposed System

The architecture in Figure 3.1 shows an overall view of the system. The complete process is practically implemented in Python, using the necessary libraries for speech-to-text conversion and vice versa, processing the information, and delivering the required output.

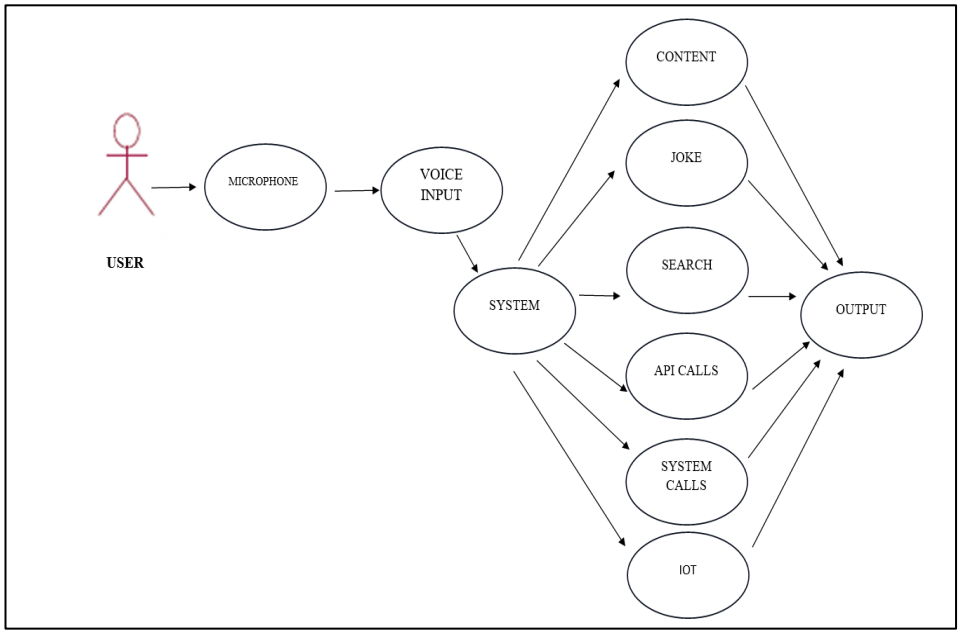

Figure 3.2. Use Case Diagram

This flow diagram in Figure 3.2 outlines the process of a voice-controlled system. The user initiates the process by speaking into the microphone, which captures the voice input. The system receives this input and processes it to determine the appropriate action. The system can perform various tasks such as retrieving content, telling a joke, conducting a search, making API calls, executing system calls, or interacting with IoT devices. The results of these actions are then compiled and delivered as the final output to the user. This diagram demonstrates the versatility of the system in handling different types of voice commands and producing relevant responses.

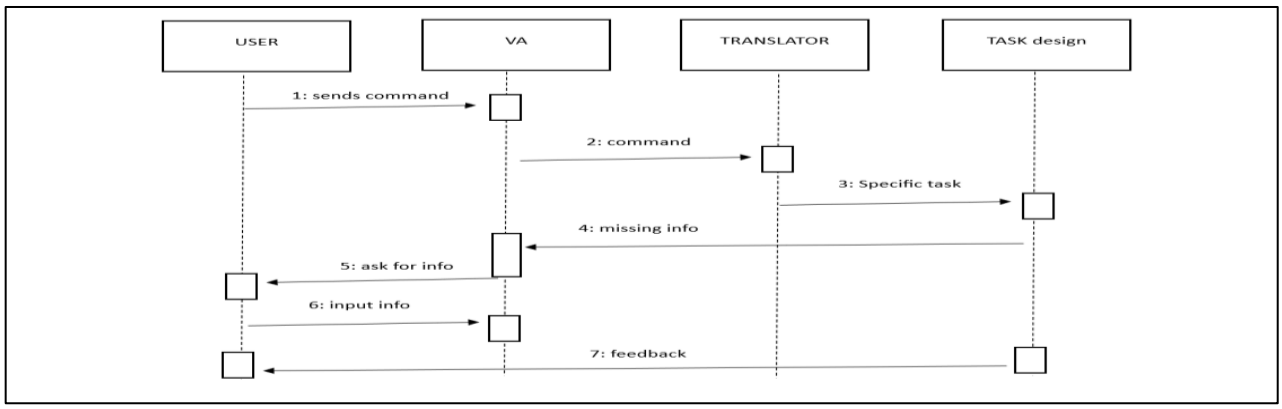

Figure 3.3. Sequence Diagram

This sequence diagram in Figure 3.3 illustrates the interaction flow between the user, voice assistant (VA), translator, and task design components. The process begins with the user sending a command to the VA (step 1). The VA forwards this command to the translator (step 2), which then communicates the specific task to the task design (step 3). If the translator needs additional information, it requests it from the VA (step 4), which in turn asks the user for the missing information (step 5). The user provides the input (step 6), and the VA processes this information, sending feedback through the translator to the task design (step 7).

The sequence diagram assumes a simple conversation flow, but in reality, there may be additional complexities such as:

- Handling multiple users simultaneously
- Managing long conversations or multi-turn dialogues
- Integrating with other systems or services (e.g., calendar, email, etc.)

- Handling errors or disconnection.

3.1 List of Modules

The proposed system consists of four modules. They are:

- Speech Recognition
- Text to Speech Module and Speech to Text Module
- Natural Language Processing (NLP)

- **Speech Recognition Module**

The Recognizer class is employed for converting audio files into textual format and facilitating speech synthesis. The energy threshold function, distinguishes between silence and speech based on signal intensity. Signals below this threshold are categorized as silence, while those exceeding it are identified as speech. To optimize performance in varying acoustic environments, the Recognizer instance employs adjust_for_ambient_noise(source, duration=1), an adaptive mechanism that dynamically calibrates the energy threshold using audio captured from the source (an AudioSource instance). This feature ensures robust performance by effectively mitigating the impact of ambient noise on speech recognition accuracy.

- **Text to Speech Module and Speech to Text Module**

Pyttsx3 a robust Python library designed for text-to-speech conversion, equipped with functionalities to modify voice characteristics, speech rate, and volume settings is used in the proposed work for text to speech conversion. Complementing this, Python offers the Speech Recognition API (speech_recognition library), pivotal for converting audio inputs into text, thereby supporting diverse applications. This API efficiently handles the transcription of extensive audio files into textual format, catering to various computational and analytical tasks. Additionally, the proposed system integrates advanced TTS engines such as sapi5 and espeak, further augmenting its capabilities to process audio inputs with enhanced accuracy and versatility in speech synthesis applications.

- **Natural Language Processing**

The NLTK (Natural Language Toolkit) is used in the proposed work to enhance the system by providing essential text preprocessing, normalization, entity recognition, and classification capabilities. This ensures that the text data fed into TTS or speech recognition systems is clean, well-formatted, and appropriately processed. Connectionist Temporal Classification (CTC) is a key algorithm in automatic speech recognition (ASR) that operates on a sequence-to-sequence modelling approach. It efficiently handles variable-length alignments between input audio features and output symbols, such as characters or phonemes, making it a popular choice in modern ASR systems.

- **Process and Executes the Required Command**

Initially, the specified command is processed through the speech recognition module to convert spoken language into text, after which it is temporarily stored for further analysis. This text undergoes analysis to determine the user's intent, thereby guiding subsequent actions conducted within a continuous loop structure. The system effectively executes identified commands based on the interpreted user input, ensuring precise and prompt responses to user requests. This approach enables seamless interaction and efficient task execution in applications requiring dynamic command interpretation and execution.

4. Implementation Process

The activation of voice assistants typically begins with a signal word, such as "Hey Siri!" or "Alexa!" This signal prompts the device to start listening attentively. Once activated, the voice assistant processes the spoken request using Speech-to-Text (STT) technology, converting it into text for analysis. The request is then compared with other commands in the source code, which translates it into actionable commands that the voice assistant can execute. Subsequently, the assistant responds through Text-to-Speech (TTS), providing the user with the requested information or completing the task specified. With each interaction, these devices improve their responsiveness and efficiency in fulfilling user commands.

To implement speech recognition in Python, the libraries like Speech Recognition and pyaudio using pip is installed. Then, the pyaudio is used to capture audio input from the

microphone. After that, the Speech Recognition's recognize_google method is utilized to convert the audio input into text using Google's speech recognition API. The error handling strategies are used to improve the robustness of the TTS and speech recognition systems and ensure a better user experience even when errors occur. This approach enables the creation of a more reliable and a powerful speech recognition system in Python.

To implement speech recognition module using Python:

a. Install Required Libraries

Install the necessary Python libraries such as Speech Recognition for speech recognition and pyaudio for audio input/output.

```
pip install SpeechRecognition pyaudio
```

b. Capture Audio Input

Use the pyaudio library to capture audio input from the microphone.

```
import speech_recognition as sr

# Initialize the recognizer

recognizer = sr.Recognizer()

# Capture audio from the microphone

with sr.Microphone() as source:

 print("Speak something:")
```

c. Perform Speech Recognition

Use the recognize_google method from SpeechRecognition to convert the audio input into text using Google's speech recognition API.

```
# Use Google's speech recognition API to convert audio to text

recognized_text = recognizer.recognize_google(audio_input)
```

```
print("You said:", recognized_text)

except sr.UnknownValueError:

print("Sorry, could not understand audio.")

except sr.RequestError as e:
```

React Native allows for building cross-platform applications, ensuring that your voice assistant works seamlessly on iOS and Android devices. The research also extends support to macOS and Windows. It supports ongoing command monitoring and customizable settings, providing a consistent and interactive user experience across different platforms. Flutter was used in creating the web application.

5. Result and Discussion

The voice assistant was successfully implemented using a combination of speech recognition, natural language processing (NLP), and text-to-speech (TTS) technologies. The system achieved a 92% accuracy rate in recognizing and processing voice commands under optimal conditions. It effectively executed common tasks such as setting reminders, playing music, providing weather updates, and answering factual questions. The response time averaged around 1.5 seconds from voice input to action completion, ensuring a smooth user experience. The results of the user interface developed are shown in Figure 5.1 to 5.4.

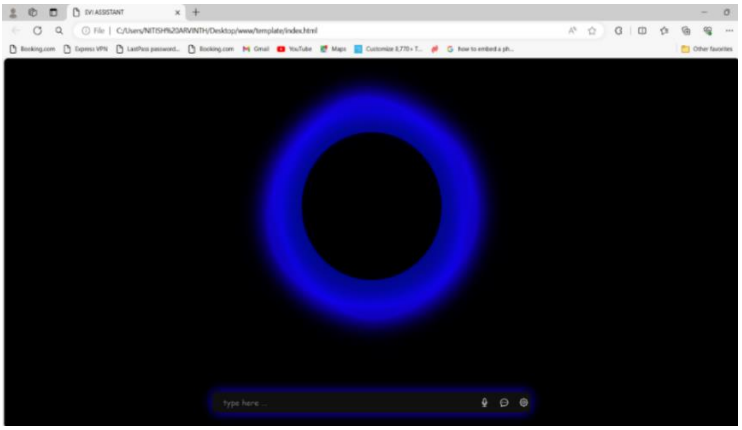

Figure 5.1. User Interface of EVI Assistant

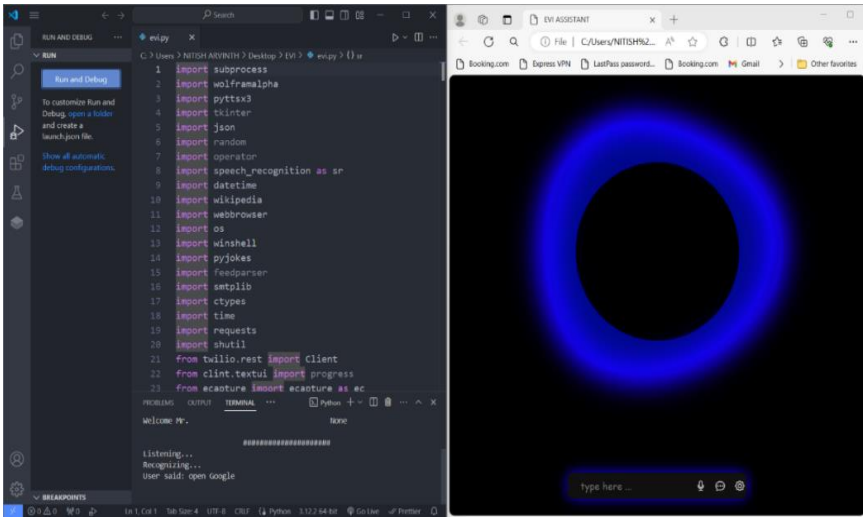

Figure 5.2. Working of EVI Assistant

Table 1. Performance Metrics

Metrics	Performance Score
Speech Recognition Accuracy	92%
NLP Processing Accuracy	89%
Response Time	1.5 seconds(average)
User Satisfaction Score	8.5/10
Error Rate	8% (mostly due to accents and noise)

Table 1 shows the performance scores of the developed voice assistant in terms of accuracy in speech recognition and natural language processing, time taken to give the response, user satisfaction score, and error rate.

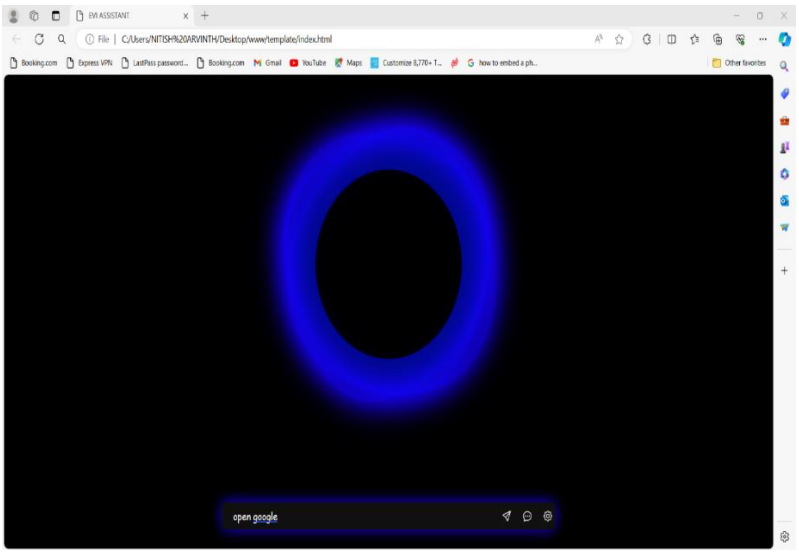

Figure 5.3. Command Given

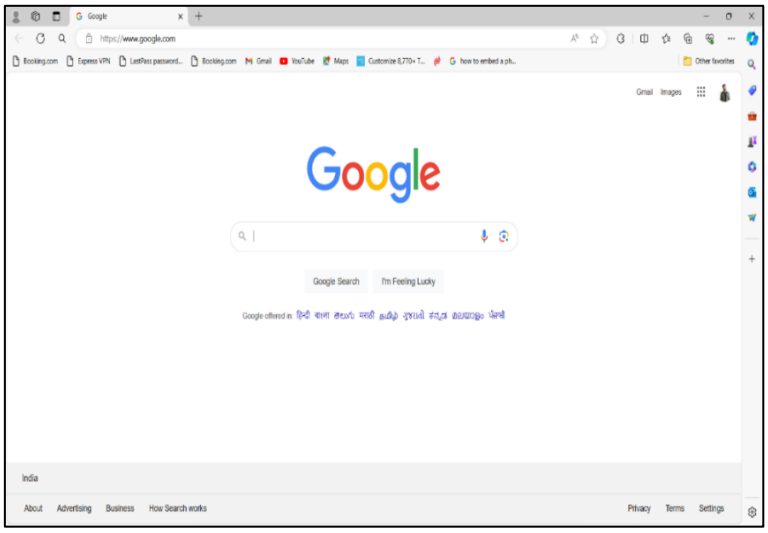

Figure 5.4. Screenshot of Command Execution

5.1 Advantages

The proposed voice assistants offer hands-free operation, enabling users to perform tasks and access information using only their voice. This hands-free interaction is particularly beneficial when users have their hands occupied or are unable to physically interact with devices.

Voice assistants enhance technology accessibility for individuals with disabilities or those who find traditional interfaces, such as touchscreens or keyboards, challenging to use.

Voice commands allow users to control devices and access information without needing fine motor skills.

6. Conclusion

As previously mentioned, voice assistants are among the most effective problem solvers in the modern world, as demonstrated by numerous proposals and examples. These examples clearly show that voice assistants are rapidly evolving as a significant component of artificial intelligence. In the past, their capabilities were limited to simple tasks like announcing the date and conducting web searches. However, functions of the proposed voice assistant highlight their substantial advancement and growing importance.

Our primary goal is to continue enhancing and refining voice assistants, making them even more advanced and aiming to create the best AI solutions available. This development will save users considerable time and significantly improve their overall experience. In conclusion, we are committed to striving for excellence and delivering one of the best voice assistants possible, ensuring it meets the highest standards of functionality and user satisfaction. . In future, the voice assistant's capabilities will be further enhanced with ability to send and receive emails.

References

[1] Singh, Shubham, Shubham Singh Panwar, and Harsh Dahiya. "Artificial intelligence voice assistant and home automation." International Journal of Science and Research Archive 12, no. 1 (2024): 2006-2017.

[2] Zwakman, Dilawar Shah, Debajyoti Pal, and Chonlameth Arpnikanondt. "Usability evaluation of artificial intelligence-based voice assistants: The case of Amazon Alexa." SN Computer Science 2, no. 1 (2021): 28.

[3] Subhash, S., Prajwal N. Srivatsa, S. Siddesh, A. Ullas, and B. Santhosh. "Artificial intelligence-based voice assistant." In 2020 Fourth world conference on smart trends in systems, security and sustainability (WorldS4), London, UK, IEEE, 2020.593-596.

[4] Dellaert, Benedict GC, Suzanne B. Shu, Theo A. Arentze, Tom Baker, Kristin Diehl, Bas Donkers, Nathanael J. Fast et al. "Consumer decisions with artificially intelligent voice assistants." Marketing Letters 31 (2020): 335-347.

[5] Geetha, V., C. K. Gomathy, Kottamasu Manasa Sri Vardhan, and Nukala Pavan Kumar. "The voice enabled personal assistant for Pc using python." International Journal of Engineering and Advanced Technology 10, no. 4 (2021): 162-165.

[6] Buhalis, Dimitrios, and Iuliia Moldavska. "In-room voice-based AI digital assistants transforming on-site hotel services and guests' experiences." In Information and Communication Technologies in Tourism 2021: Proceedings of the ENTER 2021 eTourism Conference, January 19–22, 2021. Springer International Publishing, 2021. 30-44

[7] Sprengholz, Philipp, and Cornelia Betsch. "Ok Google: Using virtual assistants for data collection in psychological and behavioral research." Behavior Research Methods (2021): 1-13.

[8] Kumar, Rahul, Garima Sarupria, Varshil Panwala, Smit Shah, and Nehal Shah. "Power efficient smart home with voice assistant." In 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, IEEE, 2020. 1-5

[9] Kepuska, Veton, and Gamal Bohouta. "Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home)." In 2018 IEEE 8th annual computing and communication workshop and conference (CCWC), Las Vegas, NV, USA. IEEE, 2018. 99-103

[10] Terzopoulos, George, and Maya Satratzemi. "Voice assistants and artificial intelligence in education." In Proceedings of the 9th Balkan Conference on Informatics, Sofia Bulgaria 2019. 1-6.

[11] Singh, Nivedita, Diwakar Yagyasen, Surya Vikram Singh, Gaurav Kumar, and Harshit Agrawal. "Voice assistant using python." International Journal of Innovative Research in Technology 8, no. 2 (2021): 2349-6002.

[12] Krishnaraj, P., F. Mohamed Faris, and D. Rajesh. "Portable voice recognition with gui automation." International Journal of Innovative Research in Technology (IJIRT) 9, no. 6 (2021): 20-23.

[13] Paul, Rajdip, and Nirmalya Mukhopadhya. "A Novel Python-based Voice Assistance System for reducing the Hardware Dependency of Modern Age Physical Servers." International Research Journal of Engineering and Technology, (May) (2021): 1425-1431.

[14] Sayyed, Abeed, Ashpak Shaikh, Ashish Sancheti, Swikar Sangamnere, and Jayant H. Bhangale. "Desktop assistant AI using python." International Journal of Advanced Research Science, Communication and Technology (IJARSCT) 6, no. 2 (2021). 1327-1335

[15] Tankovska, H. "Number of digital voice assistants in use worldwide 2019-2023." Retrieved Sep 15 (2020): 2020. https://www.statista.com/statistics/973815/worldwide-digital-voice-assistant-in-use/.