International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 04 | Apr 2025 www.irjet.net p-ISSN: 2395-0072

VIVI: A Personalized Virtual Assistant

Sai Patil[*1], Zoha Parkote[*2], Chinmay Mirlekar[*3], Susmita Mistry[*4], Deepti Janjani[*5], Sanjay Patil[*6]

*[*1,2,3,4] Student, Department of Artificial Intelligence and Data Science, Datta Meghe College of Engineering, Navi Mumbai, Maharashtra, India*
*[*5] Assistant Professor, Department of Artificial Intelligence and Data Science, Datta Meghe College of Engineering, Navi Mumbai, Maharashtra, India*
*[*6] Head of Department, Department of Artificial Intelligence and Data Science, Datta Meghe College of Engineering, Navi Mumbai, Maharashtra, India*

Abstract - *The accelerated development of artificial intelligence (AI) has brought about virtual assistants that simplify human-computer interaction. However, current systems are still limited by inadequate personalization, the absence of advanced task automation, and inadequate contextual retention from one interaction to another, which hinders their ability to provide truly seamless and intelligent user experiences. This work introduces VIVI, a cutting-edge virtual assistant that seeks to overcome these boundaries by bringing forth a combination of leading-edge technologies such as Generative AI and Natural Language Processing (NLP) to render dynamic, context-dependent answers and execute sophisticated tasks effortlessly. The architecture of VIVI includes a smart workflow that starts with system boot-up and hotword detection, then proceeds to command identification through speech or text inputs. These commands fall into types like system controls, media actions, web interactions, or AI-powered tasks. By enabling advanced automation of tasks and multi-platform compatibility, VIVI provides higher user flexibility, providing not only functional commands but also proactive and personalized outputs. VIVI reimagines the power of virtual assistants through enhanced user interaction through intelligent decision-making complemented by a polished task accomplishment, positioning it as a very strong tool in personal and workspaces alike.*

Key Words: Generative AI, Virtual Assistant, Task Automation, Natural Language Processing, AI-powered Systems, Speech Recognition, Personalized User Experience.

1.INTRODUCTION

VIVI addresses the shortcomings of existing virtual assistants, particularly in terms of personalization and task automation. VIVI, our AI-powered virtual assistant, aims to overcome these shortcomings by employing Generative AI, Natural Language Processing (NLP), and sophisticated automation techniques. With VIVI, users can expect personalized responses, contextual understanding, and smart task management across multiple platforms, giving a more natural and personalized experience.

The key contributions of VIVI are that it supports keeping contextual knowledge consistent across multiple interactions, allowing for more relevant and coherent conversations. Unlike traditional virtual assistants, VIVI enables dynamic user-adaptive answers through Generative AI, and conversations become more interactive and personalized. VIVI also supports higher-order task automation, and it is possible to run complex multi-step workflows and system administration tasks. Its design ensures seamless interoperability with third-party applications, allowing users to efficiently manage both personal and work-related tasks. Moreover, VIVI's user-friendly interface, supporting both voice and text inputs, ensures flexibility and ease of use across a wide range of devices and environments.

This project also elaborates on the applications of VIVI in the future for smart homes, offices, and IoT devices, rendering it a tremendous advancement in virtual assistant technology. By addressing current challenges in personalization and automation, VIVI creates a new standard in context-aware, intelligent virtual assistants.

2. LITERATURE SURVEY

Mekni, Mehdi. (2021) proposed an AI-based virtual assistant using conversational agents. The virtual assistant was designed to support multimodal communication, including text and voice-based interaction, ensuring a more intuitive user experience. The study explored the integration of Natural Language Processing (NLP) and Machine Learning (ML) techniques to enhance user interaction [1]. A. Sudhakar Reddy M, Vyshnavi, C.and Saumya designed a virtual assistant using NLP for speech and text interpretation and ML for better responses. It automated tasks like scheduling, emails, and smart home control, making user interactions easier and more efficient [2]. Baskaran, G., Raj, Harrish, Kumar, S., and Anand, R. (2021) developed an AI-powered virtual assistant for Windows OS. It used voice commands to manage applications via a command prompt and optimized system performance with user-defined models. Reinforcement learning improved adaptability based on user interactions [3].

Harshit Agrawal, Nivedita Singh, Gaurav Kumar, Dr. Diwakar Yagyasen, (2023) developed a Python-based voice assistant to execute Linux commands using voice input, reducing the need for keyboards and mice. The system improved

accessibility, efficiency, and system usability while minimizing hardware costs [4]. Sahu, Ajay, Jha, Ashish, Bhargava, (2023) explored the role of AI in virtual assistants, emphasizing their growing significance in daily life. Their study highlighted the integration of AI-driven voice assistants across various devices, enabling intelligent search and task automation. The paper also discussed the challenges and opportunities of virtual assistant technology, underlining its potential to transform human interactions and accessibility across multiple fields[5].Sinha & Siegert (2022) improved German voice assistant accuracy by combining ASR outputs using ROVER, reducing errors by 10% for clean speech but showing limited gains for noisy data[6].Kulhalli, Kshama V., Sirbi, Kotrappa & Patankar, Abhijit J. (2017) created a voice-based personal assistant that works with or without the internet. It understands voice and text commands, performs tasks, and makes mobile interactions easier using speech recognition and NLP [7].

Jain, Sakshi R. & Jason, Feon (2023) developed a Python-based desktop voice assistant using NLP, machine learning, and speech recognition APIs. It integrates tools like TensorFlow, NLTK, PyQt, and Web APIs for enhanced interaction. A healthcare case study highlighted its potential to improve efficiency and patient care [8]. Chinchane, A., Bhushan, A., Helonde, A., & Bidua, K. (2022) introduced a Python-driven voice assistant leveraging AI, NLP, and speech recognition. Unlike cloud-dependent models, their approach emphasizes local processing for enhanced privacy and faster responses, paving the way for smarter, autonomous systems [9]. Këpuska & Bohouta (2018) proposed a next-generation Virtual Personal Assistant (VPA) model using multi-modal dialogue systems. Unlike traditional assistants like Siri and Alexa, this model integrates speech, image, gesture, and video recognition to enhance human-machine interaction. The system is designed for applications in education, healthcare, robotics, home automation, and security, leveraging AI-driven conversational knowledge bases for improved adaptability and efficiency [10].

3. METHODOLOGY

Experimental Setup

The experimental setup for the VIVI AI project includes a system equipped with an AMD Ryzen 5 3500U processor, 8 GB RAM, and AMD Radeon Graphics, running on Windows 11. This hardware configuration supports multitasking, real-time AI processing, and system automation. The development is carried out using Visual Studio Code, with Git for version control and browsers like Chrome and Edge for testing. Python 3.10+ is used as the primary programming language, alongside SQLite3 for database operations and APIs like OpenWeather and Hugging Face for real-time data and NLP integration.

A variety of software libraries are used to power VIVI's capabilities. These include pyttsx3, speech recognition, eel, and pyautogui for voice interaction and GUI handling; selenium, requests, and webbrowser for web-based automation; and PIL, pytesseract, and numpy for image processing. System tasks are managed using os, psutil, subprocess, and related libraries. For AI functionalities, libraries like torch, torchvision, diffusers, transformers, and hugchat are employed, enabling deep learning and conversational abilities. Clipboard operations, regular expressions, and database handling are supported through pyperclip, re, and sqlite3.

The programming stack includes Python for backend development and AI logic, HTML/CSS and JavaScript for the interface, and SQL for database management. This combination allows for a modular, intelligent virtual assistant with features like task automation, voice commands, image processing, and real-time user interaction.

System Architecture

A structured methodology was used to develop and test VIVI as an AI-powered virtual assistant for optimizing performance in specific functionality like automation of tasks and personalization as well as context recognition. The methodology is divided into various stages beginning from system design through NLP integration to Generative AI implementation then task automation setup before user interaction testing.

It also ensures if each user input is intelligently routed through intent identification and NLP processing. This enables VIVI to generate personalized, context-aware responses by leveraging its AI engine, resulting in a dynamic and efficient user experience.

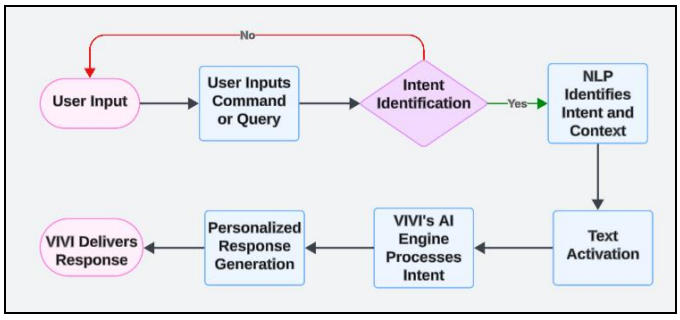


Fig -1 VIVI System Architecture

The Figure 1 depicts the process flow that is used for VIVI, a personalized virtual assistant, in its handling of user inputs along with the generation of responses. The definite process begins with "User Input." A specific command or query is entered at this location. If input isn't clear, the system repeats the entry process. Otherwise, then it proceeds onward to "Intent Identification," where the system determines whether it can fully understand all of the request. If such intent is truly not identified well enough, then that user is prompted by refining the provided input. If

the intent is successfully recognized, the system leverages multidimensional Natural Language Processing (NLP) to analyze the query, identifying its intent as well as contextual meaning.

Once the exact purpose and setting are known, the mechanism then goes to "Text Activation," and applicable data is handled. Then, VIVI's AI engine interprets upon the intent for the purpose of determining the appropriate response. This particular information is routinely passed along to the "Personalized Response Generation" phase. This perpetually and always ensures that the response is customized to all of the user's needs. Finally, the generated response is delivered up on back to the user, thus entirely completing the cycle. This planned flow surely sees that VIVI gives precise and also relevant support when improving user inputs especially for greater productivity.

4. RESULTS AND DISCUSSION

The creation of VIVI, an intelligent virtual assistant, comprised several design, implementation, and testing stages to assess its performance, functionality, and usability. This section is a comprehensive discussion of the major project outcomes such as interface design, feature execution through commands, and interactions based on artificial intelligence. All the subsections contain the pertinent visual evidence and explanatory points that highlight the features of VIVI during real-world applications.

I. Interface Design & Functionalities

Interface design and functionalities play a crucial role in shaping the overall user experience. A clean, intuitive, and responsive layout is essential to ensure smooth navigation and effective interaction through both voice and text inputs. The visual and functional elements of VIVI's UI are highlighted to demonstrate how users interact with the assistant across its core features, including general layout, settings access, activation, and exit processes.

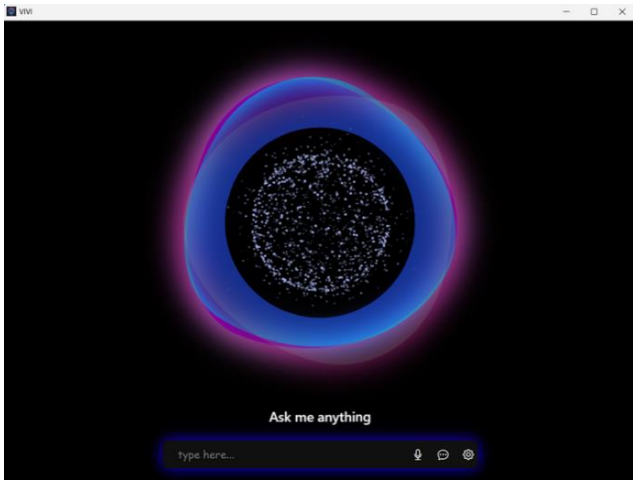


Fig - 2: User Interface Overview

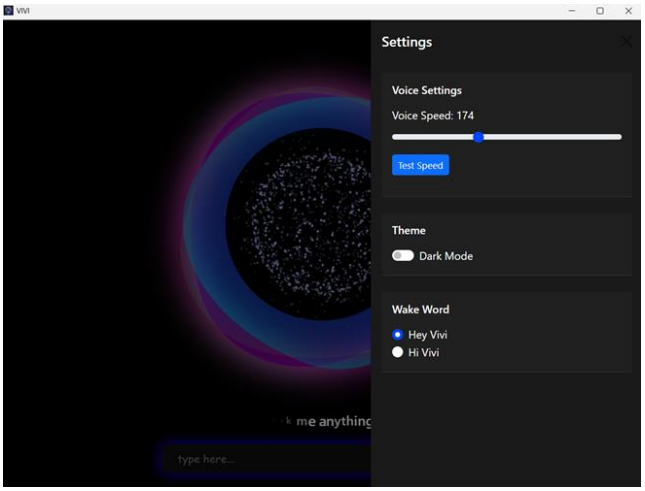


Fig - 3: System Settings Panel

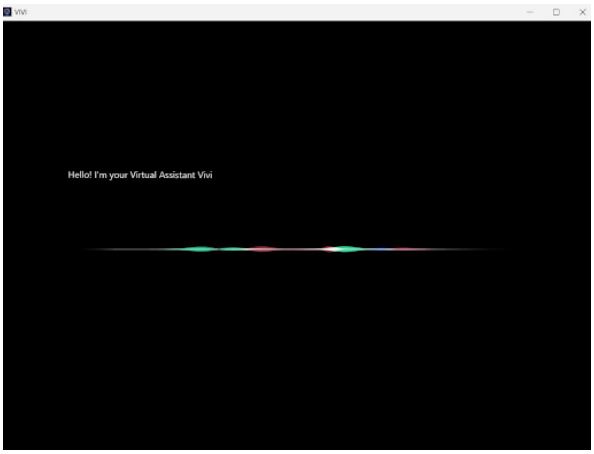


Figure 4: Assistant Wakes Up

Figure 5: Stopping and Exiting the Assistant

Figure 2 presents the main interface of the virtual assistant "VIVI", featuring a clean layout with a central circular animation that indicates active system status. It supports both voice and text input, guided by the prompt "Ask me anything," which encourages interactive communication.

Figure 3 highlights the settings panel, where users can adjust voice speed, test responses, toggle dark mode, and select a wake word either "Hey Vivi" or "Hi Vivi" to personalize activation.

Figure 4 illustrates how VIVI responds to its wake word with a greeting and dynamic waveform animation, signaling readiness to assist. This visual cue ensures users are aware when the assistant is listening. Finally, Figure 5 shows the assistant's exit prompt, where users can simply say "Stop" or "Exit" to close the session. These design elements collectively ensure a smooth, responsive, and user-friendly interaction experience.

II. Command-Based Feature Activation

Command-based feature activation is central to VIVI's functionality. This section highlights how the assistant handles essential actions from initiating the system and handling communication tasks to managing entertainment controls and providing timely informational updates such as launching from a fresh state, interacting with messaging platforms, managing media playback, and retrieving real-time information like weather updates etc. These features reflect VIVI's ability to understand user intent and perform tasks efficiently, enhancing both usability and responsiveness in practical scenarios.

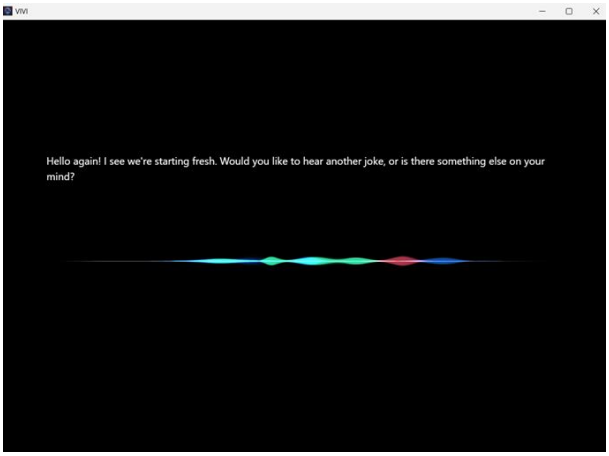
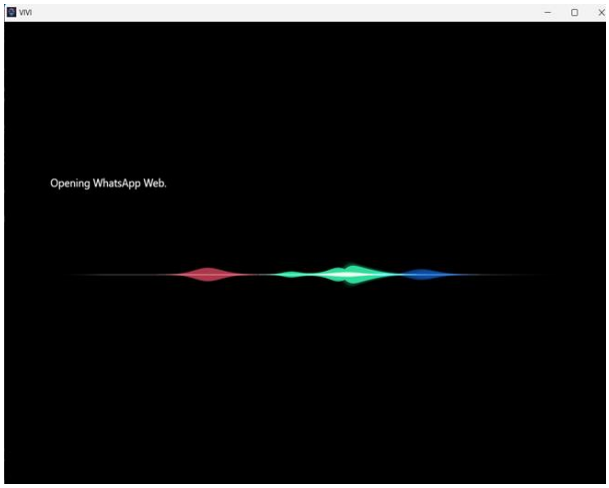


Figure 6: Launching the Assistant (Fresh Start)

Figure 7: Weather Updates and Forecast



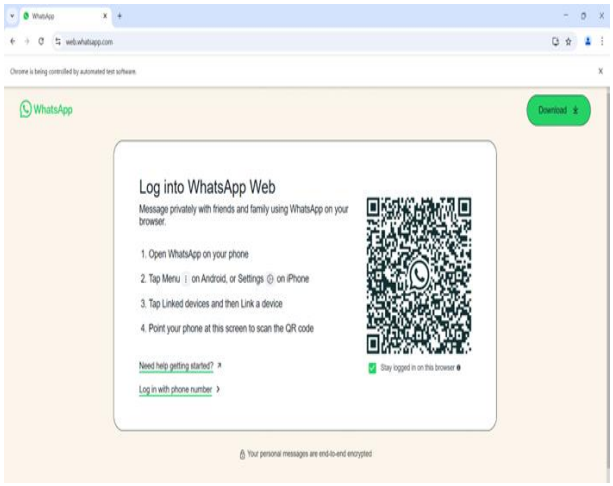
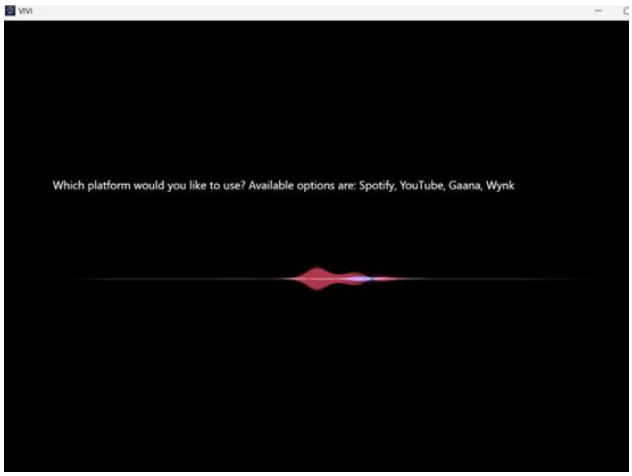


Figure 8 (a) and 8(b): WhatsApp Integration and Messaging



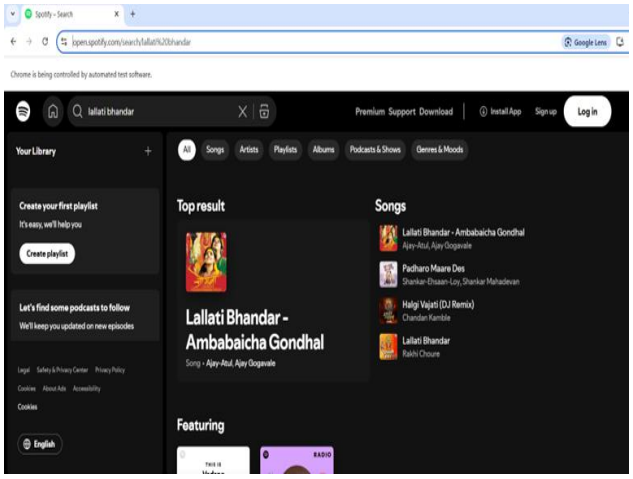


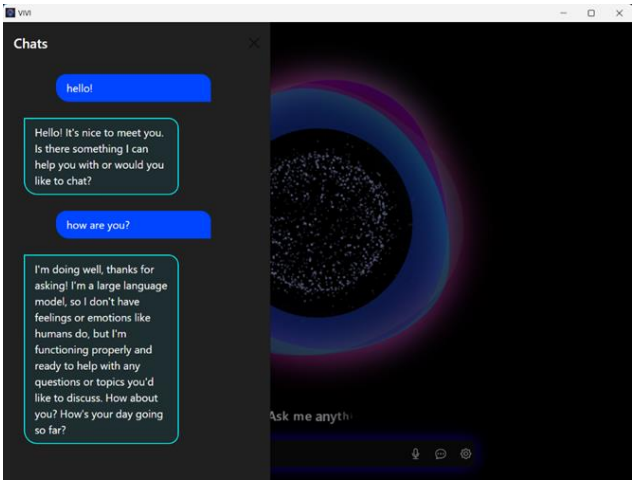
Figure 9 (a) and (b): Music Playback and Control

Figure 6 presents VIVI's launch interaction, where the assistant welcomes the user with a friendly message and animated waveform, clearly indicating that it is active and ready to assist. This responsive greeting helps establish a natural and engaging user experience. As shown in Figure 7, VIVI responds to a voice command by delivering a real-time weather update, including temperature, humidity, and sky conditions, offering convenient, hands-free access to useful information.

Figure 8 (a) and (b) demonstrates VIVI's capability to manage communication tasks, where a simple command opens WhatsApp Web and directs the user to the interface, making messaging easier. In Figure 9 (a) and (b) , the assistant handles media playback by prompting the user to select a platform such as Spotify or YouTube, and immediately opening it—highlighting VIVI's versatility in executing entertainment-related commands efficiently.

III. AI-Enabled Interactions and Learning

VIVI's AI-driven capabilities are showcased through its ability to handle more advanced, conversational interactions that simulate natural dialogue. This section outlines how the assistant processes diverse requests ranging from self-identification and status updates to delivering jokes, offering cooking suggestions, and responding to location-specific inquiries. These interactions highlight VIVI's capacity to interpret context, personalize responses, and provide meaningful engagement that extends beyond basic functionality, adding a human-like dimension to the overall user experience.



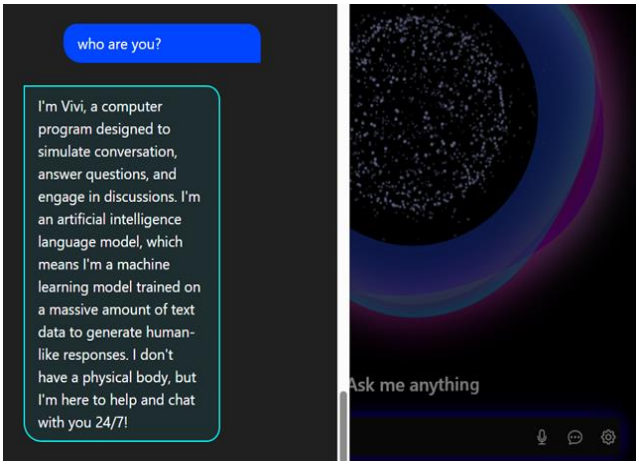


Figure 10 (a) and (b): Assistant Self-Identification and Status Inquiry

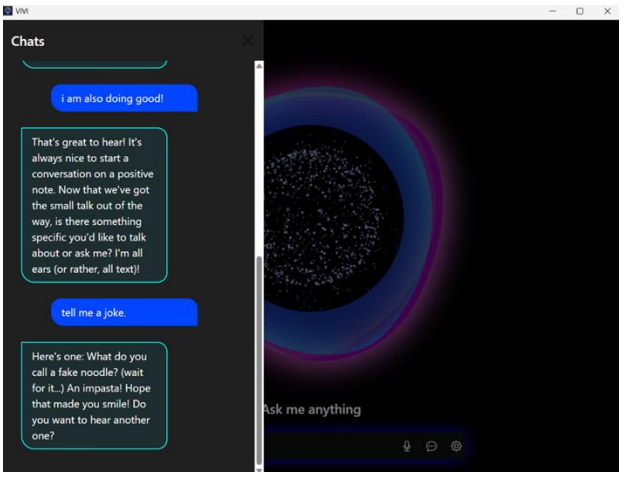


Figure 11: Entertainment Feature: Joke Generation

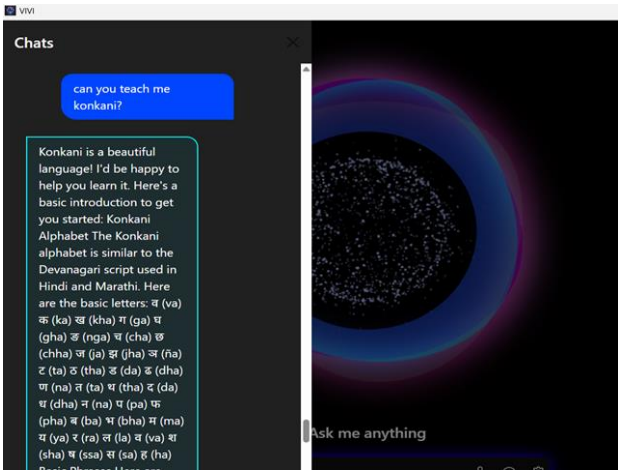
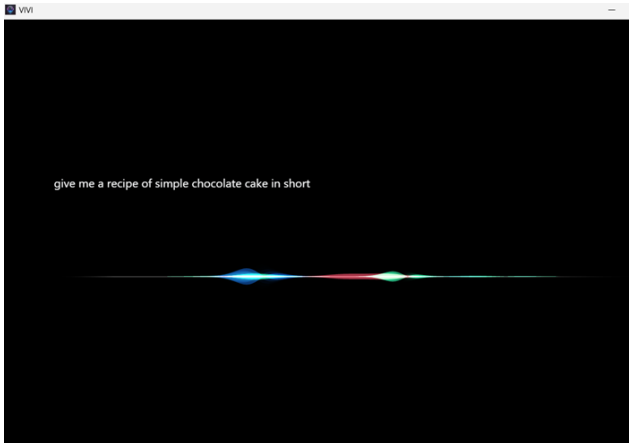


Figure 12: Location-Based Inquiry: Konkan Information



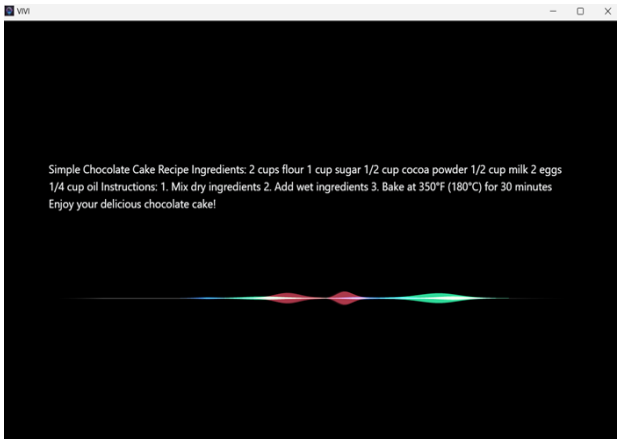


Figure 13 (a) and (b): Recipe Suggestion and Cooking Assistance

Figure 10 (a) and (b) shows VIVI introducing itself after a user greeting, responding in a friendly and informative way that makes the interaction feel more natural. In Figure 11, VIVI responds to a fun command by telling a joke, adding a playful touch to the conversation and making the experience more enjoyable.

Figure 12 shows VIVI answering a location-related question by providing information about the Konkani language, highlighting its ability to respond to specific cultural or regional queries. In Figure 13 (a) and (b), VIVI helps with a cooking task by sharing a quick and easy chocolate cake recipe, showing how it can assist users with daily activities.

3. CONCLUSIONS

The research emphasizes the design and implementation of an intelligent virtual assistant that can comprehend and react to user inputs via text and voice-based interactions. Emphasis was on developing a friendly interface, with key features such as messaging, entertainment control, information retrieval, and voice commands. The assistant was implemented with priority on real-time response, personalization, and simplicity, showing its capability to execute various tasks effectively in one platform.

The project reinforces the growing relevance of AI-driven tools in simplifying routine digital tasks and enhancing user engagement. By successfully merging contextual understanding with command execution, the assistant delivers an experience that feels both natural and practical. This research establishes a strong groundwork for expanding the assistant's capabilities, offering opportunities for future enhancements in fields like automation, accessibility, and cross-platform integration.

REFERENCES

[1] Mekni, Mehdi. (2021). An Artificial Intelligence Based Virtual Assistant Using Conversational Agents. Journal of Software Engineering and Applications. 14. 455-473. 10.4236/jsea.2021.149027.

[2] Sudhakar Reddy M, Vyshnavi, C. Raju Kumar, and Saumya,"Virtual Assistant Using Artificial Intelligence" in J ETIR March 2020, Volume 7, Issue 3 ISSN-2349-5162.

[3] Baskaran, G & Raj, Harrish & Kumar, S & Anand, R. (2021). To Build A Virtual Assistant By Using Artificial Intelligence. The Open Artificial Intelligence Journal. 2. 1134. 10.6084/m9.figshare.14446467.

[4] Harshit Agrawal, Nivedita Singh, Gaurav Kumar, Dr. Diwakar Yagyasen, Mr. Surya Vikram Singh. "Voice Assistant Using Python" An International Open Access-revied, Refereed Journal. Unique Paper ID: 152099, Publication Volume & Issue: Volume 8, Issue 2, Page(s): 419-423.

[5] Sahu, Ajay and Jha, Ashish and Bhargava, Ritik and Priya, Priyanshu and Kumari, Rupa, Voice Assistant Using Artificial Intelligence (March 10, 2023). Proceedings of the International Conference on Innovative Computing & Communication (ICICC) 2022.

[6] Sinha, Yamini & Siegert, Ingo. (2022). Improving the Accuracy for Voice-Assistant conversations in German by combining different online ASR-API outputs. 10.6094/UNIFR/223823.

[7] Dr. Kshama, V. Kulhalli, Dr. Kotrappa Sirbi, Mr.Abhijit J. Patankar, "Personal Assistant with Voice Recognition Intelligence," International Journal of Engineering Research and Technology. vol 10, no.1, pp. 416-418, (2017).

[8] S. R. Jain and F. Jason, "Personal Desktop Voice Assistant - Research Paper," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 12, no. 3, pp. 126, Mar. 2023. DOI: 10.17148/IJARCCE.2023.12324.

[9] Chinchane, A., Bhushan, A., Helonde, A., & Bidua, K. (2022). SARA: A Voice Assistant Using Python. *International Journal for Research in Applied Science & Engineering Technology (IJRASET), 10*(VI), 3567. https://www.ijraset.com.

[10] V. Këpuska and G. Bohouta, "Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)," 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2018, pp. 99-103, doi: 10.1109/CCWC.2018.8301638.